Sub-unit Cell Modeling and Manufacturing of Non-Orthogonal Three-Dimensional Braided Structures

By

Evan R. J. Batten

A thesis submitted in partial fulfilment of the requirements of the degree of

Master of Science

Department of Mechanical Engineering

University of Alberta

Abstract

Three-dimensional (3D) braiding is commonly considered an advantageous method for producing near-net composites that provide improved transverse properties relative to two-dimensional (2D) braiding. Modeling of 3D braids is an ever-growing field of research – particularly the use of sub-unit cell models. The production of complex braid shapes and patterns – and the modeling of the same – serves to allow 3D braids to be used more widely, such as various mechanical, infrastructure, or medical applications. This research focused improving the functionality of a purpose-built braiding machine and on the improvement of sub-unit cell models used for predictive modeling through an expansion of modeling capabilities and through validation. In this research, the braiding machine's maximum capacity was upgraded from 9 to 49 motors, 24 to 161 yarns, and 3 to 50 motor drivers. The control and take-up systems were also dramatically improved by the addition of 112 new I/O connections and the ability to vary or pause the take-up speed respectively. Through these improvements, many new braid shapes were possible – including three braid shapes novel to this work: isosceles triangle, right triangle, and curve approximation. The modeling application was expanded to match the capabilities of the braiding machine. In doing so, predictions of stiffness for various orthogonal braid shapes were made possible and several novel sub-unit cells were identified. The novel sub-unit cells included the isosceles triangle edge, upper corner, and lower corner and the right triangle edge, upper corner, and lower corner. It was predicted that as the isosceles triangle had a larger braid angle than orthogonal shapes (and the right triangle larger again) that the longitudinal stiffness of the triangular shapes would be reduced relative to orthogonal shapes, but that transverse properties would be improved. The interior braid angle and braid pitch predicted by the model were validated using a micro-computed tomography (micro-CT) analysis. The relative errors between the model predictions and micro-CT results for interior braid angle and braid pitch were found to be 13.00% and 6.16% respectively. It was suggested that additional measurements to normalize the model data could further reduce these errors in the future.

Preface

Some of the research conducted for this thesis forms a part of a research collaboration. The collaboration is led by Professors J. Carey and C. Ayranci at the University of Alberta and by G. Melenka at York University. The micro-computerized tomography analysis referred to in chapter 5 was performed by A. Gholami under the supervision of G. Melenka. The sample preparation and analysis of results in chapter 5 was my own work. Similarly, the construction of the braiding machine and model covered in chapter 3 and 4 was my original work.

Some work that would have normally been included in this thesis had to be redefined due to the COVID-19 pandemic. Due to laboratory shutdowns and limitations some experimentation had to be removed from the scope of this thesis. This work is discussed in the future work section.

To my grandpa, Verne Jones.

He achieved his Master of Science in Civil Engineering from the University of Alberta in 1962.

He was my inspiration:

To be an engineer,

To pursue a Master of Science,

And to maintain a sense of humor and a quick wit even in the face of overwhelming odds.

He sadly passed before he could see me finish my degree.

I hope he's at peace knowing his legacy lives on.

Acknowledgements

This thesis could not have been completed without the continual support and guidance from many sources.

First, my supervisor Dr. Cagri Ayranci and co-supervisor Dr. Jason Carey were pivotal in providing mentorship and leadership throughout my pursuit of the Master of Science degree. Both have provided key advice for my thesis and for general life. I cannot understate how much I appreciate their efforts, and particularly their support and understanding during difficult life events.

Second, I would also like to thank my colleagues both in and out of the lab for keeping me sane and on track to the best of their ability. Additionally, many staff members at the University of Alberta were pivotal in assuring the completion of my degree. To name two (among many) Rick Conrad with his help on sourcing electronics components and Gail Dowler with her advice in many aspects of my journey.

Lastly, my thanks for the support from my parents and family cannot go unsaid. My parents have always been outstandingly supportive of my pursuit of higher learning. My family had to suffer through me describing my work to them many times and I will be forever grateful for that. I'd like to especially thank my mom – who took the time to assist with revisions of my thesis when I no longer had the strength to do so.

Contents

List of Tables

List of Figures

List of Symbols

θ	Braid Angle
r	Yarn Radius
SF	Spacing Factor
z	Braid Pitch
n	Number of Sub-unit Cells by Type
E	Young's Modulus
A	Cross-sectional Area of the Braid

List of Abbreviations

1D, 2D, 3D	One-dimensional, two-dimensional, three dimensional
Micro-CT	Micro-Computed Tomography
ITE	Isosceles Triangle Edge (Sub-unit cell)
RTE	Right Triangle Edge (Sub-unit cell)
UITC	Upper Isosceles Triangle Corner (Sub-unit cell)
LITC	Lower Isosceles Triangle Corner (Sub-unit cell)
URTC	Upper Right Triangle Corner (Sub-unit cell)
LRTC	Lower Right Triangle Corner (Sub-unit cell)
FEA	Finite Element Analysis
FGM	Fabric Geometry Model
FIM	Fiber Inclination Model
CLPT	Classical Laminated Plate Theory

Chapter 1 Introduction

1.1.1 Motivation

Three-dimensional (3D) braids can be loosely defined as intertwined yarns with a 3D cross-sectional shape, as opposed to a rope or flat fabric [1]. These braids are often used as preforms for composite structures [2]. Recently, 3D braids have started to be studied more due to some key advantages over two-dimensional (2D) braids. These advantages include increased delamination resistance, improved near-net shape capability and increased out-of-plane strength [3].

Though 3D braids are commonly stated to be advantageous for applications such as aerospace, automotive, or medical, they have failed to be implemented industrially [4]. This failure is in part due to the relatively low speed of manufacture, the difficulty of accurately modeling mechanical properties, and the limited possible braid shapes and patterns. The low manufacturing speed of braids limits the feasibility of using 3D braids in mass-produced products such as automotive or construction components. This speed can be addressed both by increasing the throughput of the machine and by decreasing the human effort required to design, model, and test the braids. The modeling of braids is difficult partially due to the excessive undulations of the yarns and the variable braiding angle throughout the cross-sections. A method of modeling is required that allows both fast and accurate results for highly tailorable braids. It is also required to construct a braiding machine that can produce the tailorable braids. The ability to produce purpose-tailored braids will greatly increase the number of braid shapes and patterns that are possible, thus the number of potential applications is similarly improved. This work was motivated by these limitations.

The long-term goal for this project is to provide a predictive modeling method for arbitrary braid materials, patterns, and shapes. Through the research of braids, it has become evident that the advantages of 3D braiding lie mainly in the many possibilities for customization. As such, this fully customizable modeling method will allow 3D braiding technology to be used to its fullest capacity.

1.2 Thesis Scope

In previous work, it was shown that fast and accurate modeling was feasible for square braids by using a combination of sub-unit cells and finite element analysis [5]. These braids were manufactured using a two-step rotary 3D braiding machine. This thesis aims to improve the 3D braiding manufacturing process and capability of the braiding machine, as well as to expand modeling capability. Specifically, the physical, electrical, and electronic upgrades to the braiding machine to enable full control of braid shapes and patterns was undertaken. Similarly, the specific changes to the previously developed modeling application [5] to allow modeling of these newly possible shapes and patterns will be described. Further, the implications of these braid shapes and patterns on braid angle and other factors related to the mechanical properties of the braided composites will be explored.

1.3 Thesis Outline

This thesis is divided into six chapters. The second chapter contains background on existing 3D braiding machines and processes, previous modeling attempts, and suggested applications for 3D braids. The advantages and disadvantages of various braiding technologies are discussed to support design changes to the constructed braiding machine. Similarly, previously explored modeling methods are explained. The potential applications for 3D braids and 3D braided composites are presented to further reinforce design changes as well as to provide motivation for future expansion.

The third chapter focuses on the braiding machine itself. The design of the braiding machine in previous work is presented, along with its inherent disadvantages. Then, the updated braiding machine is explained. The physical updates to the braiding machine such as increased capacity and improved tolerance are described. Electrical and electronic updates to facilitate the improved functionality of the machine are listed. Finally, the updates made to the braiding process are explored.

The fourth chapter focuses on the braiding software, including the modeling and control software. Like chapter three, this chapter describes the previous modeling and control software and provides in-depth explanations for all updates. These updates include changes to the modeling GUI, control GUI, and microcontroller firmware.

The fifth chapter provides a comparison of the presented modeling software with micro-CT results. Within this chapter, the methods of the micro-CT process are discussed as well as suggested causes for error.

Finally, the sixth chapter presents the conclusions that were drawn from this thesis. Also, the limitations of this work are discussed as well as suggested future work.

The appendices are divided as follows. Appendix A contains datasheets for major machine components. Appendix B contains articles related to the physical construction of the braiding machine such as CAD drawings. Appendix C contains articles related to the electrical and electronic components including datasheets and schematics. Appendix D contains all computer code used in this body of work. Appendix E contains additional documentation related to the models developed in this work.

Chapter 2 Background

2.1 Introduction to Three-dimensional Braids

Composite structures are being increasingly studied due to their high specific strength and stiffness properties [6], [7]. Braiding is one method of manufacturing preforms for composite structures consisting of "continuous, intertwined, fiber tows" [6]. Further, braiding is distinct from other textile methods such as weaving or knitting. Weaving results from orthogonally interlacing yarns and knitting requires inter-looping yarns, whereas braiding necessitates neither loops nor orthogonal yarns [1]. Note that in some literature, the words "braiding" and "weaving" are used interchangeably. This literature review focused on works that used the aforementioned description of a braid (i.e. lacking orthogonal yarns) regardless of the label of "braiding" or "weaving."

Two-dimensional (2D) braids have been studied extensively in literature. 3D braids can be differentiated simply from 2D in that a 2D braid describes either a braided single-layer sheet or thin-walled tube and a 3D braid involves further braiding in the third dimension, i.e. through thickness dimension, [1]. As described extensively in [1], this simple definition is lacking when two or more single-layer sheets are braided together in a shape such as a T, L, C, I, or box. The definition used throughout this work will be the same as in [1]. The definition is as follows:

> "*3D braided fabric is a fully integrated assembly of two or more yarn layers, in which each layer comprising two or three sets of yarns having different directions is intertwined with at least one adjacent layer by a 3D braiding process*" [1].

This means that the "branches" of a shape (such as a T, L, C, I, or box) must be at least 2 layers thick – and these layers must be interconnected through the braiding process (i.e. not stitched or glued together) – in order for these shapes to be considered a 3D braid. This distinction is important because a 3D braiding machine (or 3D braider) can also produce 2D braids, but this work will focus only on those that match the description of a 3D braid. Similarly, this literature review focused only on machines, methods, and applications that included 3D braids.

2.2 Applications of Three-dimensional Braids

Composite materials, and specifically three-dimensional braided composites have been proposed for many applications. Some of these applications include aerospace [4], automotive[8], infrastructure and prosthetics [9], ballistics [3], [4], and smart structures [2]–[4]. These applications will be discussed more thoroughly in the following sections. Braiding has limitations in preform width and manufacturing speed [2]–[4]. However, braiding is still proposed for many different applications because it is very versatile. Specifically, the ability to create many different cross-sectional shapes in near-net preforms is frequently referenced as an advantage [10].

2.2.1 Transportation Applications

Transportation applications such as those found in the aerospace, automotive, and nautical industries are grouped together here due to their similar requirements. The most commonly listed limitations for such applications are the relatively slow braiding speed and the limitations of braid cross-sectional size [1]. The cross-sectional size of a 3D braid is limited by the size of the braiding machine baseplate, the number of carriers, and the yarn denier. Each of these limitations can be individually rectified, but at the cost of creating other limitations. By increasing the baseplate size, the maximum cross-sectional shape inherently increases. This has been achieved previously through using modular braiding machine designs such as [1], [11], [12]. The immediately obvious downsides to such a solution are increased space requirements and cost. Additionally, the increased distance from the outer carriers to the braiding point can cause issue with either too much slack or too much tension in the yarns [3]. As mentioned further in Section 2.3.2, the number of carriers can be increased by such methods as hexagonal horngears or switching mechanisms [3]. However, more densely packed carriers require smaller bobbins to avoid jamming the machine. As a result, there is a trade-off in maximum braid length achievable with these types of machines. There is a similar trade-off in length when the yarn denier is increased. These factors make it simple to conclude that with current 3D braiding technology it is not feasible to produce large area braids economically – as suggested in [3], [4].

On the other hand, for specific smaller cross-section applications, 3D braiding has been shown to be useful. For example, in [13]–[15] it was stated that 3D braids were very useful for applications where flexibility and damage tolerance were needed such as wing stiffening elements or curved fuselage frames. Further, the ability for complexly shaped, near-net preforms has been identified as a unique benefit of the 3D braiding process [16]. The complexity of braided shapes has an inverse relationship with the speed of production. This is due to longer braid paths and the need for more complex control mechanisms. As a result, 3D braiding is not yet suitable for mass-produced components for any of the transportation applications discussed herein. However, for applications where high complexity, high specific stiffness, and high damage tolerance are needed, 3D braids are still worth investigating.

2.2.2 Infrastructure Applications

The goal of using 3D braided composites in infrastructure would primarily be to replace existing building products with stronger and lighter alternatives. One potential application would be to replace existing steel beams. The most common cross-sectional shapes of steel beams are tee (T), channel (C), bearing pile (H), angle (L), I-beam (I), pipe, or hollow (box) [4]. Many of these shapes have been previously attempted using other composite manufacturing methods such as layup of fiber mats. However, as evidenced by the invention of the deltoid noodle shown in [16], this method often leaves gaps between the assembled mats. As a result, it is considered beneficial to be able to manufacture these shapes in a continuous cross section or near-net shape [17]–[21]. Although these shapes have been achieved previously, the effect of variable braid patterns within the existing shapes has not been studied. Additionally, the method for creating deltoid noodles as shown in [22], [23] requires cutting a notch in a prepreg mat. It was considered advantageous for this thesis to explore the possibility of creating similar shapes without necessitating cutting of the fibers.

2.2.3 Niche Applications

Niche applications include the aforementioned applications such as prosthetics, ballistics, and smart structures. In general, niche applications benefit predominantly from the ability to produce variable shapes and patterns. Other medical applications such as braided scaffolds [24]–[30], orthopedics [17], and stents [18]–[20] will be discussed further in this section.

3D braids have been reportedly used to produce scaffolds for repairing articular cartilage [21], ligament or tendon repair [18], [19], and bone repair [19]. For some instances of ligament or tendon repair, the reported 3D braids appeared closer to what has been defined in this thesis as 1D braids [20]. In fact, in the case of [17], the braids were manufactured completely manually rather than using a braiding machine. However, where 3D braiding (as defined in this thesis) was used, it was stated to be beneficial due to the ability to tailor the mechanical properties [17]. The ability to tailor the mechanical properties of the resulting composite was also stated as an advantage for the repair of articular cartilage [17]. In the case of [21], a circular 4-step braiding machine was used. As a result, the parameters that were varied were limited to the braid angle (based on spaces moved in a step), yarn density, and the number of layers of braided yarns [22], [23]. A 4-step braiding process was also used for the proposed method of bone repair [22]. The use of 4-step braiding machines limits the ability to tailor the mechanical properties of a braid as will be discussed in Section 2.3.1.

Contrary to scaffold braids, which are typically intended to be biodegraded and replaced with natural tissue, the orthopedic braids covered in this literature review were intended to be permanent fixtures [23]. As identified in [25]–[30], it is critical for the stiffness of a permanent fixture in the human body to closely match the tissue it is replacing (i.e. bone). This means that the ability to control the mechanical properties of the braid in such situations is even more crucial than for soft-tissue replacement. In [28], it was identified that it is also important to match the shape of the implant to the tissue it is replacing. Matching the shape allows for the implant to better approximate the stresses that would be normally experienced by the natural tissue and surrounding area.

The use of braids to manufacture stents is a more established practice than either scaffolds or orthopedics. However, stents are predominantly designed using 2D braiding processes and shape memory alloys (i.e. metal fibers). 3D Braided stents with diverging tubes have also been proposed in [11]. Regardless, it is commonly regarded that the use of FEA is useful to predict the behaviour of the stents inside the body [28]. These finite element processes – like many early 3D braiding models – often neglect the effect of fiber undulation. Instead, the areas where fibers overlap are constrained to act almost as a single node, with the exception that rotational movement is not constrained. In one case, the stent was manufactured to allow different braiding angles at the ends [29]. This allowed for the stent to achieve both the appropriate radial stiffness to prevent collapse, while also limiting the extension of the stent. Both of these parameters had been previously shown to cause issues when stents were implanted. In this case, the different braid angles were achieved by modifying the braid after production [29]. To further validate the use of 3D braiding machines for the purpose of manufacturing stents, one group used a 4-step circular 3D braider to produce their stents [30]. They were able to produce multi-layer intertwined stents and vary the braid angle. It was also noted that braid angle had the largest impact on the function of the stents [11]. This group later produced shape memory polymer braids as well [11], and were able to show that by manipulating braid parameters the polymer braids could be equally effective to the metal.

2.3 Three-dimensional Braiding Machines and Methods

Three-dimensional braiding machines (3D braiders) are commonly categorized as either row-and-column or rotary [1]. The invention of Bluck [31] in 1969 has been suggested as the first instance of both row-and-column and rotary 3D braiding [1]. However, it is interesting to note that the invention of Blaisdell [32] in 1930 described a method to intertwine two separate layers of two yarns each together in order to produce brake linings. This suggests that the benefits of "three-dimensional" braiding have been sought after for a long time, even if it was not referred to as such.

2.3.1 Row-and-column Braiding Machines

In the early study of 3D braiding, the primary style of machine was row-and-column [1]. Braiders within this category, such as the invention of Florentine [33] in 1980, were originally designed in either rectangular or circular configurations and braid by shifting an entire row or column. This method is further described in [34]. Although originally referred to as "Magnaweave" [34] the invention [33] would be better categorized as a 4-step rectangular 3D braid [1]. The 4 steps of such a braider in a rectangular configuration – as described in [35] – are as follows. First, alternating actuation of carriers in the column direction. This means alternating actuators move entire columns "up" or "down." Second, alternating actuation in the row direction. Similarly, this means alternating actuators move entire rows either "right" or "left." Third and fourth, these motions are repeated with directions reversed. That is, actuators that moved “up” now move “down,” actuators that moved “right” now move “left” and vice versa. For a circular configuration, these same steps apply except that the row movement is replaced by ring movement [35]. In both configurations, the primary variables with which to affect the braid pattern are the number of spaces of movement in the column (radial) or row (ring) directions [34]. For a rectangular configuration a designation of a 2 x 1 pattern would equate to 2 spaces of movement in the row direction and 1 space of movement in the column direction [35]. The same 2 x 1 designation for a circular configuration would equate to 2 spaces of movement in the radial direction and 1 space of movement in the ring direction [35]. Similar machines have been designed to manufacture using two or six steps and have been described extensively in [35]. While early row-and-column braiders could often produce simple braids relatively quickly, they lacked the ability to manufacture braids with complex braid patterns. As a result, multi-step braiding was explored.

Beyond 4, 2, or 6-step row-and-column braiding, another area of interest was “multi-step” braiding. Multi-step braiding was introduced by Kostar and Chou in 1994 [36], [37]. Multi-step braiding allowed for the movement of individual rows or columns a variable number of spaces [35]–[37]. The primary advantage of multi-step braiding was the large number of new “braid cycles” [36] that were newly possible. Note that braid cycles as referenced in [36] are referred to as braid patterns throughout the rest of this work. Multi-step braiding allowed for significant improvement in braid pattern complexity versus traditional row-and-column braiding, but it still required an entire row, column, or ring to be moved in a step. As a result, control of an individual carrier was still limited.

2.3.2 Rotary Braiding Machines

The braiding machine developed in this work, and previously in [5], is a rotary braider. Rotary 3D braiders are a natural extension of Maypole or lace braiding into three dimensions [1]. An overview of Maypole and lace braiding is presented in [1] and a detailed explanation in [38]. As such, this section will be reserved for the description of 3D rotary braiding. However, it is worth noting that the primary difference between the two is that Maypole braiding is a continuous process, while lace braiding is discrete. It follows that the braider developed in this work is more closely tied to lace braiding than to Maypole.

The modern rotary braider was developed as an alternative to the 4-step row-and-column braider [39]. This braider used "rotors" (hereinafter referred to as horngears) with 4 orthogonal recesses to hold fiber carriers [39]. Although 4 recesses were used in all figures of the patent, it was suggested that any number of recesses could be used. Later, this would be realized in the hexagonal braider presented in [11]. Each horngear in [39] was attached to an individual drive system that allowed each to rotate in a desired angle in either direction. The stated benefit of the individual drive systems was the ability to braid uneven cross-sections. The braiding mechanism of the invention [39] was to have two groups of horngears rotate in opposite directions in alternating steps. As such, this invention could be considered a 2-step rotary braiding machine. The 2-step method is further explored in a follow up invention [40]. The difference provided in this updated invention was the ability to vary the braiding angle by halting the rotation of specific horngears while the other continue in the 2-step pattern [40]. This ability was proposed to be beneficial for applications where the load throughout the braided structure varies. By introducing steps where the horngears follow a different pattern – such as halting briefly – this expansion to the original invention could be classified as a multi-step rotary braider in a similar fashion as the multi-step row-and-column braiders above. However, even with multi-step braiders the horngears were typically driven in 2 groups turning in opposite directions. A rotary braiding machine has not been shown to have each horngear have individual control for direction and rotation in any given step.

The evolution to hexagonal horngears [11], as well as switching mechanisms as described in [1], [11], [12], are intended to increase the yarn density in the rotary 3D braider design. Switching mechanisms also increase the ability for a braider to quickly move a carrier to a desired position and allow for bifurcated structures to be more easily produced [11], [12]. The design of the 3D braider described herein does not include switching mechanisms at present, but they would be a valuable addition in future work.

2.4 Three-dimensional Braid Modeling and Analysis

The history of analytical 3D braid modeling extending to classical laminate plate theory has been well documented in [41] and in previous work [5]. As such, only the most pertinent models to this work will be explored herein. These include the fabric geometry model, fiber inclination model, and sub-unit cell models. These will be discussed in the sections below. Finite Element Analysis (FEA) is often used in conjunction with sub-unit cell models for prediction of mechanical properties – as was done in previous work [5], [42]. This thesis did not focus on improving the FEA portion of the model, therefore discussion on FEA models was not included for succinctness.

2.4.1 Fabric Geometry Model

The Fabric Geometry Model (FGM) – first described in [43] and later updated in [44] – was a model that treated the combination of fibers and matrix as individual composites rods. It was determined that these rods could have 3 possible orientations, which depended heavily on the braid shape and pattern [43]. By using braiding parameters such as the number of spaces a carrier was moved, it could be determined how many yarns were in each orientation. From there, strength could be determined using simple geometric equations for force acting on an angled fiber and the rule of mixtures. The update to the FGM served mainly to solve two drawbacks of the original model. These drawbacks were that the original model failed to emulate transverse isotropy and that some fiber orientations would result in undefined values in the equations [44]. This updated model also demonstrated two methods for determining the composite stiffness matrix; one is termed "stiffness averaging" and uses a volumetric average of the individual rod stiffness matrices, while the other is termed "compliance averaging" and achieves the same goal using the rod compliance matrices instead. It was shown that stiffness averaging generally overpredicts the stiffness of the composite while compliance averaging generally underpredicts. However, stiffness averaging was shown to achieve much better results when compared to experimental data [44]. The FGM idealises fibers as straight rods, and does not account for undulation, but it is still useful as a quick method to predict the elastic behavior of braided composites. It was used as such to provide a baseline in the development of the model used in this work [5].

2.4.2 Fiber Inclination Model

The Fiber Inclination Model (FIM) was one of the earlier attempts at an analytical model for 3D braids [45]. FIM was based on modifications to Classical Laminated Plate Theory (CLPT) as explored in [46]. It used a unit cell similar to the one recreated in Figure 2-1. As with all unit cell models, this cell could be repeated as needed to fully describe the full composite structure (hereinafter the composite structure will be assumed to be a 3D braid, though that is not a requirement of the model) [46]. The model was formed based on the following concepts. First, it was assumed that after matrix impregnation all parallel representative fibers within the same z-layer could be treated as a single inclined lamina. Second, fibers were assumed to be straight and unidirectional within the aforementioned lamina and factors caused by a change in direction at the corner of a unit cell (i.e. interlocking or bending) were not taken into account. Third, the unit cell was assumed to be a combination of 4 equal thickness laminae with no intersection. The unit cell fiber volume fraction was thus assumed equal to that of the braid.

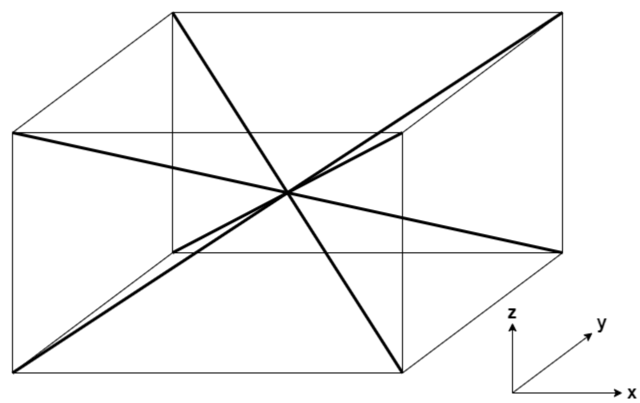


Figure 2-1: Representation of a 3D braid unit cell as described in [46]

This model was reported to be a reasonable approximation for predicting elastic properties when compared to experimental data at the time [46]. However, it is apparent that the assumptions of fibers following linear paths with no interactions are an oversimplification of the true geometry of a braid. Additionally, this model fails to account for the difference in yarn paths internal versus external to the braid. These, along with the reported deficiency of inaccurate Poisson's Ratio estimation [46], lead to the study of the models discussed below.

2.4.3 Sub-Unit Cell Models

The deficiency presented in the FIM of not being able to model the differing yarn paths internal or external to the braid eventually led to the study of small "unit-cell-like substructures" [47], which will be referred to as sub-unit cells throughout this work. In [47], 3 sub-unit cells were identified: interior, surface, and corner. These sub-unit cells could be combined to generate the larger unit cell. It is worth noting that it was demonstrated that the sub-unit cell structure depended entirely on the braiding method. Notably, the shape of the braided preform was suggested as one of the characterizing parameters [47]. Since [47], many other works have included the concept of sub-unit cells. A sample of these works and their achievements are summarized in Table 2-1 below. It is clear from Table 2-1 that this area of research has almost exclusively been performed with 4-Step row-and-column braiding machines and with the assumption of linear yarn segments. Further, the shapes covered in these papers are almost entirely simple rectangles, with only some expanding into other orthogonal braid shapes [48], [49]. Even in these studies, only L shape braids were explored.

Table 2-1: Sample of literature on Sub-unit cell models

Year	**Reference Number**	**Braiding Machine Type**	**Number of Sub-Unit Cells**	**Contribution**
1995	[50]	Row-and-column, 4-Step	3	Model determines the simplified topology of yarns is fully characterized by braiding angle and pitch. Uses mixed-volume averaging of stiffness and compliance. Linear yarns are assumed
1996	[51]	Row-and-column, 4-Step	3	Model uses different unit cells than previous work. Center unit cell is equivalent to FIM. Linear yarns are assumed
1999	[52]	Row-and-column, 4-Step	3	Model uses multiphase finite element model to analyse 3 sub unit cells. Surface and corner were found to be stiffer than interior. Linear yarn segments were assumed
2001	[53]	Row-and-column, 4-Step	3	Model defines finite elements for yarns and uses these to simulate the braiding process. It was observed that yarns curved through the interior of the braid as well as on the exterior. However, the yarn properties were not preserved so mechanical properties could not be predicted

2007	[48]	Row-and-column, 4-Step	6	Model determines the microstructure of braid with complex rectangular shapes. Some curved yarn segments are used
2008	[49]	Row-and-column, 4-Step	6	Model provides a method for calculating fiber volume fraction of complex rectangular shapes using 6 sub-unit cells
2009	[54]	Row-and-column, 4-Step	3	Model provides stiffness and strength analysis of 3D braids including axial yarns using 3 sub-unit cells. Linear yarn segments were assumed
2010	[55]	Row-and-column, 4-Step	4	Model uses 4 sub-unit cells of the interior cell rather than different cells on the surface or at the corner. Parametric finite element analysis is performed and a hexagonal cross section of the yarn is used. Linear yarn segments were assumed
2011	[56]	Row-and-column, 4-Step	4	Model is a follow-up to the method that uses 4 sub-unit cells of the interior cell. Again parametric finite element analysis is performed, but this time with axial yarns included. Linear yarn segments were assumed
2012	[57]	Row-and-column, 4-Step	4	Model provides an analytical method for determining mechanical properties of braid. A distinction in made between exterior and interior surface cells. Exterior surface cells contain curved yarn segments, others remain linear
2013	[58]	Row-and-column, 4-Step	3	Model provides finite element analysis of all 3 sub-unit cells. Surface and corner cells were found to be stiffer than interior. Linear yarn segments were assumed
2014	[59]	Row-and-column, 4-Step	5	Model presents 5 sub unit cells to describe the microstructure of the braid namely: interior, interior and exterior surface, and interior and exterior corner. Linear yarn segments were assumed
2016	[60]	Row-and-column, 4-Step	5	Model is a follow-up to the method that uses 5 sub-unit cells. In this case an analytical method is presented to predict mechanical properties. Exterior corner and surface cells are shown to have

				higher stiffness. Linear yarn segments are assumed
2016	[61]	Surface-core, 2-Step	3	Model provides sub-unit cells for new surface-core braiding technique. Linear yarns are assumed
2017	[62]	Row-and-column, 4-Step	3	Model uses 3 sub-unit cells to analyse high velocity impact. Linear yarn segments are assumed
2017	[42]	Rotary, 2-Step	3	Model allows for undulation in yarns allowing unit cells to closely approximate true braid paths

2.5 Conclusion

Through this literature review, some gaps in existing 3D braid manufacturing and modeling processes were established. For manufacturing, it was found that current braiding machines lack the ability to create complex shapes and braid patterns due to the inflexibility of the existing horngear control. Similarly, the ability to vary the braid angle within a single braid through take-up or braid pattern manipulation has not been satisfactorily demonstrated. For modeling, it was found that current models generally oversimplify the paths of the yarns throughout the braid. As a result, some important effects – such as undulation – are often missed. Further, the previous work for this project did not provide validation to ensure the modeled yarn paths match the paths produced with the braiding machine. Therefore, the goals of this thesis were as follows: (1) to produce a 3D braiding machine with the ability to create complex new braid shapes and patterns through the addition of granular control of the horngears and take-up mechanism; (2) to improve the model to allow for complex braid shapes and patterns; and (3) to provide validation of the modeled yarn paths through a micro-CT analysis of the braids.

Chapter 3 Design of a Multi-step, Variable Take-up 3D Braiding Machine to Produce Complex Braid Shapes and Patterns

3.1 Introduction

Previously, a rotary 3D braiding machine was designed with the capacity to produce 2-step, rectangular braids with a maximum yarn count of 24 [5]. For this thesis, and later work with this research group, this capacity needed to be expanded. Specifically, the first goal of this thesis is to produce a 3D braiding machine with the ability to create complex new braid shapes and patterns. This chapter is divided into two main sections giving an overview of the machine design and features and discussing the improved capacity of the machine respectively.

3.2 Overview of the 3D Braiding Machine Design

The first section of this chapter serves to provide an overview of the 3D braiding machine design and functionality.

The braiding machine developed in this work was a direct continuation of previous work [5], using many of the same components. As the design considerations for these components were not made during the course of this thesis, only a brief summary of these components is included in Table 3-1. More information on these components can be found in [5] and Appendix A.

Table 3-1: Specifications for Pre-existing Components

Component	Specific Product Information
Baseplate	1/2" Thick Aluminum Sheet, Max. Capacity: 49 Motors
Frame	Standard 4-channel T-Slot Aluminum
Yarn spool/ Tensioning device	Standard Metal Sewing Machine Bobbin
Motor	17HS19-1684S-PG19, Nema 17 Stepper Motor with gear box, Rato: 19.203:1
Motor Driver	12x RB-Dfr-727 TB6600 Stepper Motor Driver
Microcontroller	Arduino Mega 2560

This 3D braiding machine contains four subsystems, which will be discussed further in the subsections below. These sub-systems are: the braiding subsystem, which consists of the horngears, carriers, motors and drivers and serves to move the yarns through the determined braid paths; the take-up sub-system, which consists of the take-up motors and collection mechanism and serves to apply tension to the yarns; the control subsystem, which consists of the microcontroller and I/O expanders and serves to send required information to the braiding and take-up subsystems; and the power subsystem, which consists of the power bus and power supplies and serves to distribute power to all of the other subsystems. A comparison between the previous and redesigned subsystems can be found in Table 3-2.

Table 3-2: Comparison of Previous and Redesigned Braiding Machine Subsystems

Subsystem	Component	Previous	Redesigned
Braiding	Horngears	9	49
Braiding	Carriers	24	112 (+ 49 Axial)
Braiding	Motors	9	49
Braiding	Drivers	3	50
Take-up	Motors	2	No change
Take-up	Collection Mech.	In air	Collection Ring
Control	Microcontroller	Arduino Mega 2560	No change
Control	I/O Expanders	None	7 (Additional 112 I/O pins)
Power	Power Bus	None	Capacity for 6 Circuits at 50A each
Power	Power Supplies	2	5

3.2.1 Braiding Subsystem

The braiding subsystem consists of the horngears, carriers, motors, and motor drivers. This subsystem is responsible for the braiding of yarns by moving the carriers to new locations based on a user-determined braid path. The motor drivers receive information from the control system that determines whether a motor is to be enabled, the direction in which the motor should rotate, and the angle through which the motor should rotate. This information is passed to the motors, which in turn rotate the horngears. Each horngear is surrounded by four carriers, which are then moved into position by the rotation of the horngears.

In the redesign of the braiding machine, some changes were made to the design of the horngears. These changes can be seen in Figure 3-1. Most notably, the rectangular slots added to the horngears allow for them to be aligned at the start of braiding and allow for an axial yarn bobbin to be mounted directly to the horngear.

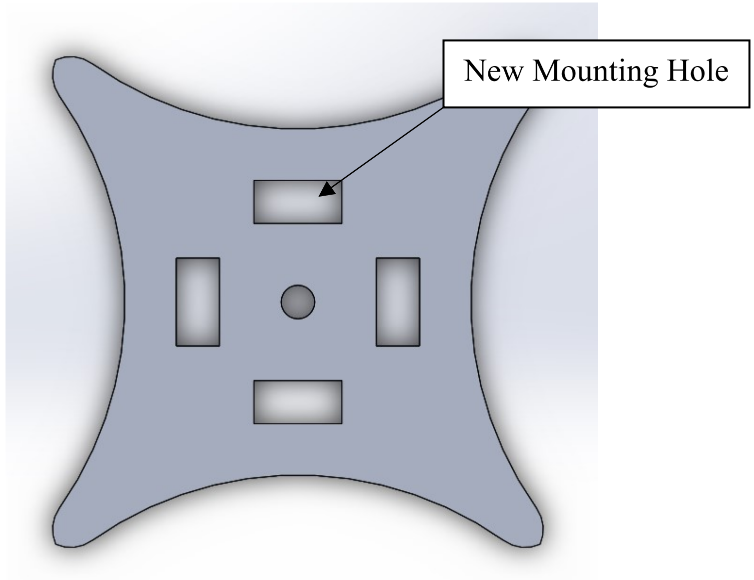


Figure 3-1: Redesigned horngear including mounting holes for axial yarn bobbin

3.2.2 Take-up Subsystem

The take-up subsystem consists of the take-up motors and collection mechanism. This subsystem is responsible for applying tension to the yarns to allow the braid to form. Like the braiding subsystem, the take-up mechanism is operated by the motors receiving information from the control system after interpretation from a motor driver. Previously, the take-up motors would only receive direction information from the motor driver as they were always enabled while the braiding machine was operating. However, in the redesigned braiding machine it was considered valuable to allow the take-up system to be disabled for certain steps and to allow the take-up speed to be varied. This variability allows for more control over the braid angle within the braid as will be discussed further in Section 3.3.4.

3.2.3 Control Subsystem

The control subsystem consists of the microcontroller and I/O expanders. This subsystem is responsible for interpreting information sent by a user into binary information for the braiding and take-up subsystems. The general flow of information is from the user's computer to the microcontroller, then to the I/O expanders, and finally to the motor drivers. The motor drivers used in this work require 3 pieces of information to operate. They have one pin for each of enable, direction, and clock. Enable turns the motor on (binary 0) or off (binary 1), direction determines clockwise (0) or counter clockwise (1) rotation, and clock pulses 1 and 0 at a frequency that determines the speed of operation. In the redesigned braiding machine, there is a clock line for the take-up subsystem and 7 clock lines are used to run the other motor drivers – with each line servicing 7 motors. This allows for the take-up motors to be operated at a different speed than the braiding motors.

The previous braiding machine only used 2 motor drivers, so only 6 I/O pins were used on the microcontroller (the Arduino Mega has 50 I/O pins). Therefore, no I/O expanders were necessary. However, even with the common clock lines for the braiding motors 108 I/O pins were required with 50 drivers. In order to facilitate this, it was decided to use I/O expanders. Specifically, the expanders chosen were 16 port I/O expanders (MCP23017-E/SP, Adafruit) that use the I^2C protocol to communicate with the microcontroller. I/O expanders were chosen for several reasons. Up to 8 I/O expanders can be connected to a single Arduino Mega using the I^2C bus, though the current design only uses 7. With 7 16-bit I/O expanders, the maximum capacity for I/O pins is 112. The current setup uses 100 of these pins (the other 8 pins are used directly from the microcontroller), which allows for some expansion at a later date. Additionally, I/O expanders were in part due to the simultaneous sending of data. All I/O expanders are connected to a communal clock line, so no information is sent out until they are all updated. This happens on the order of milliseconds and ensures that no motors will be out-of-sync.

In order to connect the I/O expanders to the Arduino Mega, the circuit boards shown in Figure 3-2 were manufactured manually using perfboard. Manually manufacturing was chosen over having boards manufactured due to the ability to rapidly change parameters if needed. Wires for the control of the motor drivers were soldered directly to the perfboard. The I/O expanders are connected to the Arduino Mega I^2C bus with pull-up resistors. This means that the default state of the I^2C pins is pulled to "high" or 1 in binary. This ensures that the square waves containing binary data are not attenuated over long distances due to wire impedance. The power and ground buses of the I/O expander boards were connected to an external 5V power source that allows for a total current draw of 2A. The use of an external power source lowers the risk of damaging the microcontroller chip if there are any issues with the power draw of the motor drivers. The complete electronics schematic of the control system, including all connections to the I/O expander boards is included in Appendix C.

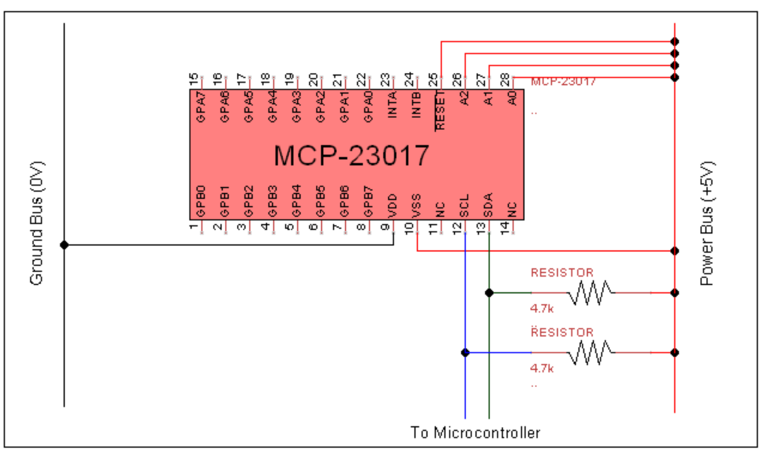


Figure 3-2: Schematic diagram of I/O custom expander circuits mounted on perfboard

3.2.4 Power Subsystem

The power subsystem consists of the power bus and power supplies. This system is responsible for supplying DC power to all other subsystems. Power flows through this subsystem from the wall outlet to the power supplies, then to the power bus. The power bus is used to distribute power to the motor drivers, while allowing for future changes.

In the previous machine configuration, the motors were arranged in series to reduce the current needed in the power supply. However, when many motors are arranged in series, there is the possibility of them engaging in a cascading fashion due to each motor receiving power one at a time. As such, it was decided to change the power configuration to a parallel arrangement. As a result, significantly lower voltage is required – limited to the max voltage of the motor drivers (12 V) – but significantly higher current is required as a trade-off. To lower the potential current through the wires, the 50 motor drivers and motors were divided into 5 separate circuits. Even so, each circuit had the potential to draw up to 43.636 A. Detailed calculations for this value are shown in Appendix C. The power supplies selected (SE-600-12, Mean Well USA Inc., purchased from Digi-Key Canada) were sized accordingly. Each power supply can supply up to 50A at 12V.

The power bus was built to allow for the connection of multiple motor drivers in parallel without welding them together. This allows the drivers to be moved around in later configurations. The power bus uses steel bars as the connection point to ensure that the electricity flow through the connection does not cause excess heat. The wires used for power connection were intentionally oversized at 14 gauge to further improve safety and reduce heat being generated.

3.3 Production of Complex Shapes and Patterns

This section introduces the newly available braid shapes and patterns resulting from the redesign of the braiding machine discussed in the previous section. Both shapes and patterns that have been previously produced with other braiding machines in literature and those that were novel will be discussed.

3.3.1 Capacity to Include Axial Yarns

Axial yarns are one of the most common ways to increase the longitudinal stiffness of a braid. This is useful for applications where the applied load is in the longitudinal direction. Some examples of this – as discussed in Section 2.2 – are wing stiffening elements, beams, or ligament replacements. As such, it was clear that the capability to include axial yarns was vital for the braiding machine.

The capability to include axial yarns was added with a modification to the horngears. As shown in Figure 3-1, the newly added rectangular holes allow for easy mounting of the axial yarn bobbin holders.

3.3.2 Capacity to Produce Shapes for Infrastructure

The braiding machine previously used a maximum 3 x 3 configuration of horngears [5]. The 3 x 3 configuration was limited both by the maximum overall size of the braids that could be produced and by the shapes and patterns that could be produced as a 3D braid. The definition of a 3D braid (as presented in Section 2.1) states that there must be a minimum of two layers intertwined in the braid shape in order for it to be considered a 3D braid [1]. As such, steel beam replacement shapes such as T, C, H, L, I, or box produced with a 3 x 3 setup could not be considered 3D braids as they would only have a single layer. Figure 3-3 shows this limitation.

Therefore, in order to align with the goal to create complex braid shapes and patterns a logical step was to expand the capacity of the braiding machine bed. The updated braiding machine incorporates a maximum 7 x 7 horngear setup, which allows for 112 carriers or 161 maximum yarns if axial yarns are included. With the expanded capacity, the shapes for steel beam replacement can be produced as 3D braids. This allows for the study of possible infrastructure applications as introduced in Section 2.2.2 Additionally, these shapes can be created in many sizes and configurations as shown for the T-shape in Figure 3-4.

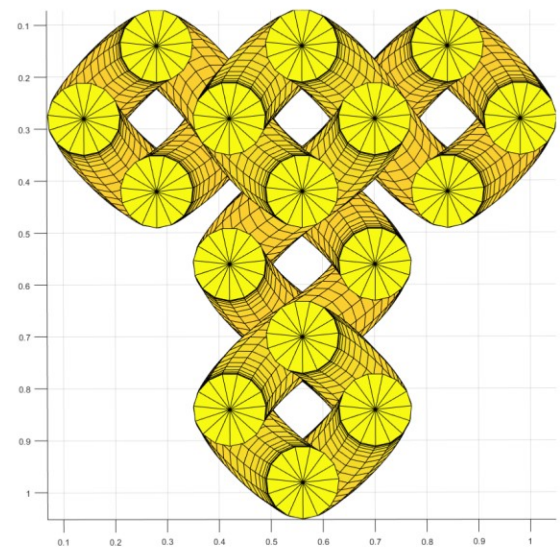


Figure 3-3: Single-layer T-shape orthogonal braid modeled with 3 x 3 horngear setup. With the 3 x 3 setup, only a single layer can be produced, which is not typically considered a 3D braid. Units are in mm.

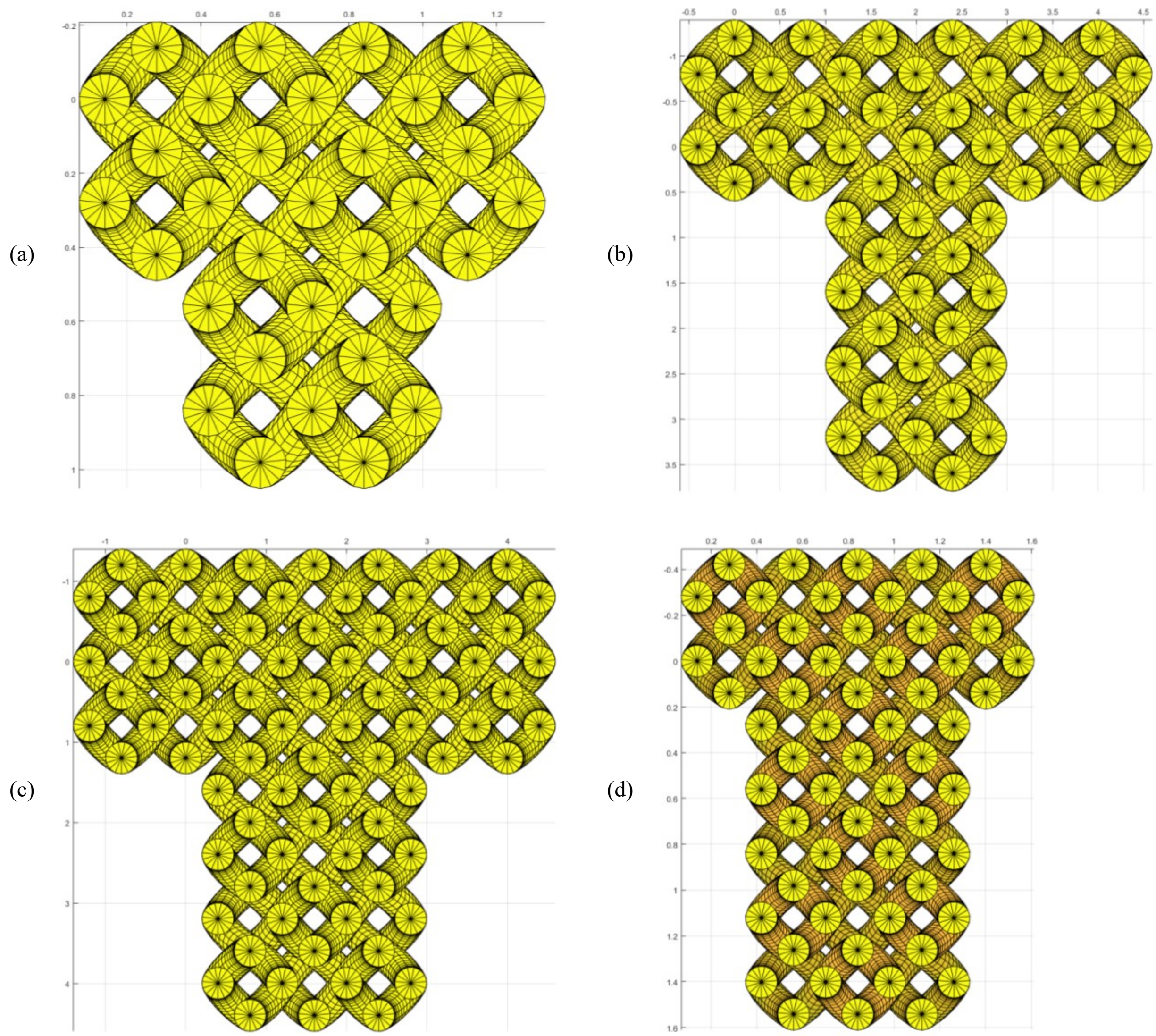


Figure 3-4: Examples of various sizes and configurations of the T-shape 3D braid made possible by the expansion of the braiding machine bed. (a) uses 4 horngears in each major direction and 2 for the thickness, (b) uses 6 horngears in each major direction and 2 for the thickness, (c) 7 horngears in each major direction and 3 for the thickness, and (d) 5 horngears for the "top" of the T, with 2 horngears for the thickness and 7 horngears for the "leg" of the T, with 3 horngears for the thickness. Units are in mm.

3.3.3 Capacity to Produce Non-orthogonal Shapes

Although the capabilities to produce orthogonal shapes for infrastructure and to include axial yarns were important, they have been previously achieved by many other researchers. As the goal was to produce new and complex shapes, it was important to also produce non-orthogonal shapes that have not been previously demonstrated. In order to do so, motor drivers were added so that each motor has an assigned motor driver. The previous configuration of the braid machine with even and odd motor groups was only capable of orthogonal shapes with the same braid pattern. The addition of individual motor drivers for each motor allowed for motors to be enabled or disabled at any time – as well as providing a method for changing the direction of an individual motor. This change was critical for the development of triangular braid shapes.

The most exciting new shape that was now possible was the isosceles triangular cross-section. As discussed in Section 2.2.2, a triangular cross-section could be used as a noodle to reinforced I-beams made with fiber layup processes [16]. Triangular shapes could be approximated with 2-step rotary braiding, but as shown in Figure 3-5 there are disconnects along the hypotenuse of the triangle where the yarns do not intertwine. It was suspected that these disconnects would result in lower transverse mechanical properties as the yarns were no longer intertwined with other layers. As transverse properties are one of the key advantages to 3D braids, this was considered unacceptable. However, with multi-step rotary braiding, these disconnects can be braided together by swapping the yarns. Figure 3-6 shows the end result of this method, and Table 3-3 shows the steps required for this shape to be made. By doing so, the yarns were then intertwined throughout, thus theoretically improving the function of the braid.

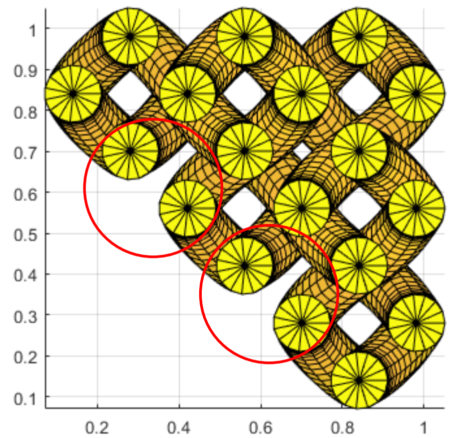


Figure 3-5: Isosceles triangle modeled using a traditional 2-step braiding method on a 3 x 3 braiding machine bed. Units are in mm. Note the disconnects circled in red

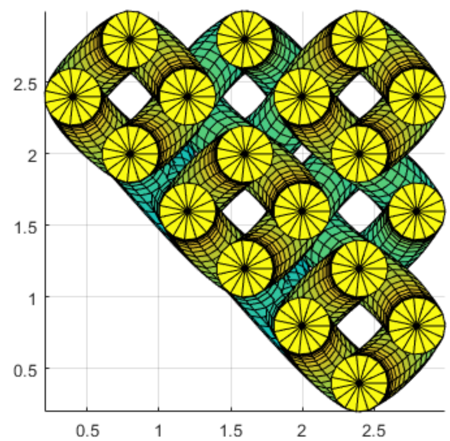


Figure 3-6: Isosceles triangle modeled using a multi-step (7 steps) braiding method on a 3 x 3 braiding machine bed. Units are in mm. Note that there are no disconnects in the yarn paths

Table 3-3: The 7 distinct steps required to braid the isosceles triangle shown in Figure 3-6. Note that the motor numbers correspond to the middle 3 x 3 square in the 7 x 7 braiding machine

Step	**Enabled Motors**								
	19		**25**		**31**		**33**		**Take-up**
1	*Direc.*	*Angle*	*Direc.*	*Angle*	*Direc.*	*Angle*	*Direc.*	*Angle*	*Speed*
	CCW	90°	CCW	90°	CCW	90°	CCW	90°	1 mm/min
	26		**32**						**Take-up**
2	*Direc.*	*Angle*	*Direc.*	*Angle*					*Speed*
	CW	90°	CW	90°					1 mm/min
	18		**24**						**Take-up**
3	*Direc.*	*Angle*	*Direc.*	*Angle*					*Speed*
	CW	180°	CW	180°					0 mm/min
	17								**Take-up**
4	*Direc.*	*Angle*							*Speed*
	CCW	180°							0 mm/min
	18		**24**						**Take-up**
5	*Direc.*	*Angle*	*Direc.*	*Angle*					*Speed*
	CCW	180°	CCW	180°					0 mm/min
	17								**Take-up**
6	*Direc.*	*Angle*							*Speed*
	CW	180°							0 mm/min
	18		**24**						**Take-up**
7	*Direc.*	*Angle*	*Direc.*	*Angle*					*Speed*
	CCW	90°	CW	90°					0 mm/min

Other right-angle triangular cross-sections can be made using a similar procedure. The result is shown in Figure 3-7. Again, this shape could be approximated with 2-step braiding, but this would result in larger sections of the braid not intertwining. Therefore, it's reasonable to say that this right-angle triangle shape could not be sufficiently produced without the use of multi-step braiding. In addition to the benefit of improved transverse properties, one of the biggest advantages of 3D braids is the ability to produce near-net shapes. Near-net shapes allow for composites to be produced without joining procedures such as gluing or stitching. This ability is an advantage because the joint is often the first point of failure in a composite. As such, the ability to produce a right-angle triangle near net shape with this braiding machine allows for 3D braids to be used in more applications. One such application could be potentially braiding wing-flaps for aircraft.

Further, this switching procedure for intertwining loose corners can be extrapolated to make a rough approximation of a curve. For example, a combination of the 45° angle from the isosceles triangle and the more acute angle from the general right-angle triangle could be used to approximate a curved shape as shown in Figure 3-8. Where before curved shapes were often made by over-braiding a 2D braid over a curved mandrel, near-net curved shapes could possibly be made directly with 3D braids. This possibility could allow the advantages of 3D braids to be brought into fields such as orthopedic braids, where the braids must conform to organic shapes.

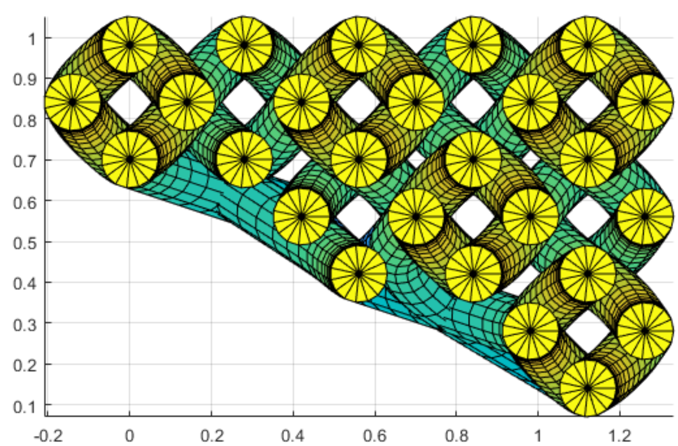


Figure 3-7: Right-angle triangle modeled using a multi-step (11 steps) braiding method. Requires a minimum braiding machine bed size of 5 x 4 Units are in mm.

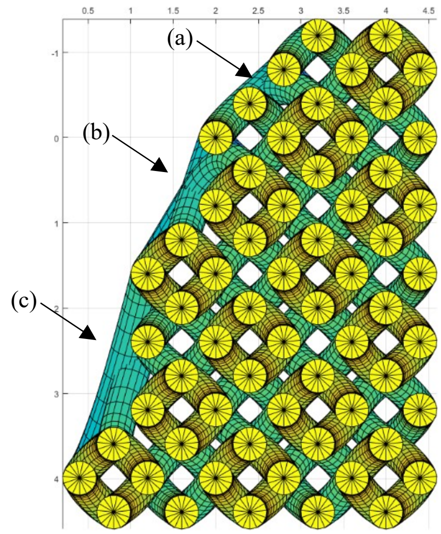


Figure 3-8: Rough approximation of a curve modeled using a multi-step (9 step) braiding method. Formed by combining a (a) 1-1 switch, (b) 2-1 switch, and (c) 3-1 switch. Minimum machine bed size is 7 x 6. Units are in mm.

3.3.4 Capacity to Manipulate Braid Angle

As the braid angle is widely accepted as one of the dominating factors affecting the mechanical properties of the braid, it is evident that the ability to manipulate the braid angle is highly beneficial. Contrary to 2D braids, which generally have one braid angle throughout, 3D braids typically have differing braid angles between the exterior of the braid and the interior. Previously, the equations for internal and external braid angle for a 3 x 3 2-step braid have been defined as follows [5]:

$$\theta_{int} = \tan^{-1}\left(\frac{4\sqrt{2}rSF}{z}\right) \tag{3.1}$$

$$\theta_{ext} = tan^{-1}\left(\frac{2\sqrt{2}rSF}{z}\right) \tag{3.2}$$

In these equations, *r* represents the radius of the yarn, *z* represents the pitch of the braid, and *SF* represents a spacing factor. The spacing factor is determined as follows [5]:

$$SF = \frac{z}{\sqrt{z^2 - 16r^2}} \quad (3.3)$$

In the case where the radius of the yarn is significantly smaller than the pitch of the braid (as is the case throughout this work) the spacing factor can be approximated as 1. In these equations, only the transverse ($2\sqrt{2r * SF}$) and axial ($z/2$, z respectively) distances travelled by the yarn affect the braid angle. Therefore, any methods for manipulating braid pattern should result in a change of either the transverse or axial distance travelled.

The transverse distance travelled will generally stay consistent within a given braiding subsystem setup. However, the controlling the take-up speed will inherently affect the axial distance travelled. In the redesigned braiding machine, it is possible to change the take-up speed at the step level. That means that any given braiding step can have a different braid angle than the last. When combined with the ability to change the braid pattern at any point, the ability to control take-up becomes a massively effective tool for creating highly tailorable braids. These tailored braids could be effective in niche or specialized transportation applications as discussed in Sections 2.2.1 and 2.2.3.

3.4 Conclusions

Through the redesign of the braiding machine, a multi-step, variable take-up machine was produced. This redesigned braiding machine has significantly improved capacity braid shapes and patterns when compared to the previous design. The additional capacities of axial yarns and larger rectangular shapes allowed this braiding machine to be comparable with other machines that exist in literature. Most notably, the conversion to a multi-step braiding machine and the ability to vary the take-up speed have allowed this braiding machine to produce complex braid shapes and patterns and to be able to vary the braid angle within those shapes. These new capabilities make this braiding machine a far better candidate for the research and production of braids for niche applications that require very specific braid properties and shapes.

Chapter 4 Development of Novel Sub-unit Cell Models for 3D Braids using an Integrated Modeling and Control Application

4.1 Introduction

Accurate geometric models of 3D braids have been a pursuit of many research groups since 3D braids have been produced. Some, such as the geometric model used in the Fiber Inclination Model, were highly simplified by assuming yarns were straight within a unit cell, ignoring interactions between yarns, and using a single unit cell throughout the braid structure [63]. Later, it was realized that the yarn paths are different between the yarns in the center of the braid and those on the surface or corners. Thus, the 3 sub-unit cell model was developed [50]. This model included separate unit cells for the center, surface, and corner of the braid. However, these unit cells still assumed that the yarns travelled in straight paths within the unit cell and ignored interactions between yarns. Many further models using sub-unit cells were explored [42], [51]–[55], [57]–[62], [64]. Some of these models incorporated curved yarn paths for select unit cells [48], [57], however it wasn't until quite recently that the unit cells were generated using code to allow for the inclusion of undulations [42]. This chapter is a continuation of that model to allow for the generation of non-rectangular shapes and braids with variable braid patterns.

4.2 Model Development

The model presented in this work was developed as a continuation of the work presented in [5], [42]. The redesigned application was created in MATLAB® (MATLAB 2019b, Mathworks, Inc.). This application operates on the concept of machine emulation. Machine emulation is a process with which a computer can approximate the motion of the yarns throughout the braiding process.

The general flow of information for machine emulation in the developed software is as follows. First, the application is used to create and store information about each step of the braid. This information is stored in two forms: the position of each yarn before braiding is stored in a standard array and all other information is stored in a structure array. Next, both arrays, as well as some user-input data such as the yarn denier, are sent to a braid generation function built in MATLAB®. This function then calls sub-functions for determining the locations of each yarn in every step (machine emulation sub-function) and for generating 3D plot points with which to model the braid (path plotting sub-function). These functions are described in more detail below, and the MATLAB® scripts can be found in Appendix D.

4.2.1.1 Function of the Application

The application has four tabs: "Carrier Setup", "Braid Pattern Setup", "Modeling", and "Braid Control" and the workflow generally proceeds thus. The Carrier setup tab allows for the selection of carriers to be loaded with yarn. The chosen selection of yarns populates a 15 x 15 matrix denoting the positions of each yarn. Figure 4-1 shows two examples of this (a) showing a 3 x 3 braid setup without axial yarns and (b) showing a 3 x 3 braid setup with axial yarns.

0	4	0	11	0	18	0
1	0	8	0	15	0	22
0	5	0	12	0	19	0
2	0	9	0	16	0	23
0	6	0	13	0	20	0
3	0	10	0	17	0	24
0	7	0	14	0	21	0

(a)

0	4	0	14	0	24	0
1	5	11	15	21	25	31
0	6	0	16	0	26	0
2	7	12	17	22	27	32
0	8	0	18	0	28	0
3	9	13	19	23	29	33
0	10	0	20	0	30	0

(b)

Figure 4-1: Examples of yarn matrix populated using the MATLAB® modeling application. (a) shows the yarns in a 3 x 3 setup without axial yarns and (b) shows the same setup with axial yarns. Matrices are cropped to the areas of interest.

The Braid Pattern Setup tab is used to create and save individual braiding steps for modeling or control purposes. The Braid Pattern Setup tab is shown in Figure 4-2. The parameters that are stored in each step are the state of each motor (enabled/disabled) and the direction and angle of rotation, plus the speed and state of the take-up mechanism. This information is stored in a structure array for easy reference throughout the modeling process. The format of the structure array is shown in Figure 4-4.

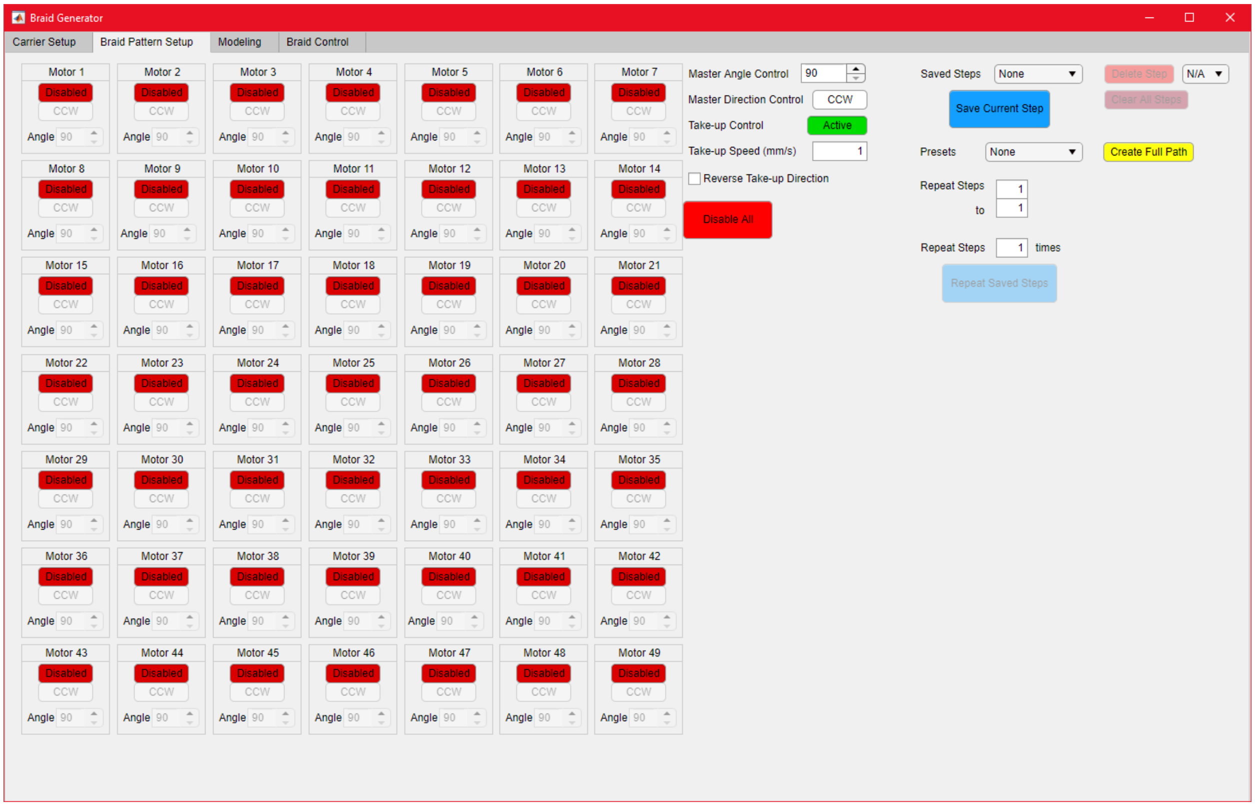


Figure 4-2: Braid Pattern Setup Tab in the developed MATLAB® modeling application. This tab allows for granular control all parameters necessary to define the braid formation for modeling or control purposes

The Modeling tab allows for the manipulation of various modeling parameters. Notably, the material properties of the matrix and yarn can be selected, and the number of knots in the splines for approximating the yarn paths can be entered.

The Control tab is used simply to connect to the microcontroller for the braiding machine.

4.2.1.2 Yarn Array and Step Structure Arrays

The yarn selection tool directly populates the yarn array with a 1 wherever a yarn is selected in the Carrier Setup tab. These ones are later replaced with integers from 1 to the total number of yarns to allow the tracking of each individual yarn. The array is always 15 x 15 to represent the total possible positions of fibers. The main advantage of this method is that the possible yarn motions are limited by the physical size of the braiding machine rather than the size of the desired braid. This allows for yarns to move far outside the braid boundaries if needed during a switching step. This change was crucial for allowing the many new braid shapes and patterns that have been described herein.

The horngear centers were described by (row, column) vectors. This allowed for flexibility in motor directionality and the inclusion of axial yarns. While the previous modeling method could add axial yarns, but only as an “all or nothing” addition. This new method meant that individual axial yarns could be selected or deselected as needed to further improve the flexibility of modeling. An example step with axial yarns only populated in the corners of a 3 x 3 setup is shown in Figure 4-3. Patterns such as this could be used to further tailor the properties of a braid. In this example, it could be expected that the corners of the braid would be stiffer due to the axial yarns, but the braid would have a lower overall density when compared with a braid with all axial yarns populated.

0	4	0	13	0	20	0
1	5	10	0	17	21	26
0	6	0	14	0	22	0
2	0	11	0	18	0	27
0	7	0	15	0	23	0
3	8	12	0	19	24	28
0	9	0	16	0	25	0

Figure 4-3: Examples of yarn matrix populated using the MATLAB® modeling application. Axial yarns are only populated in the corner horngear positions – designated with red circles. The empty axial yarn positions are circled in black. Matrix cropped to the area of interest.

The format of the structure array that contains all data for the braid steps is as follows. The highest level of the step structure is designated *Step*. This container holds fields for each step that was created using the GUI. Each *Step* field holds two sub-structures: one designated *motor* and one designated *TakeUp*. The *motor* sub-structure has 49 fields (one for each motor), which each further contain the fields *Enable*, *Direction*, and *Rotation*. The *Enable*, *Direction*, and *Rotation* fields correspond to the enable, direction, and angle buttons in the GUI. The *TakeUp* sub-structure has one field, which contains *Enable* and *Speed* fields. The *Enable* field here corresponds to the active/paused take-up button, and the *Speed* field corresponds to the speed fillable box in the GUI. Figure 4-4 shows a schematic of the entire structure arrangement. One major benefit of using a structure array is readability. To determine if motor 35 in step 7 was enabled, the query *Step(7).motor(35).Enable* can be used. This greatly improves the ability for future changes to be made to the code. The use of the structure format also allows for simple creation of pre-set braid steps. These can be manually written into the code to allow for easy access at a later date. This allows for reduced time consumption when producing common or repeated braids.

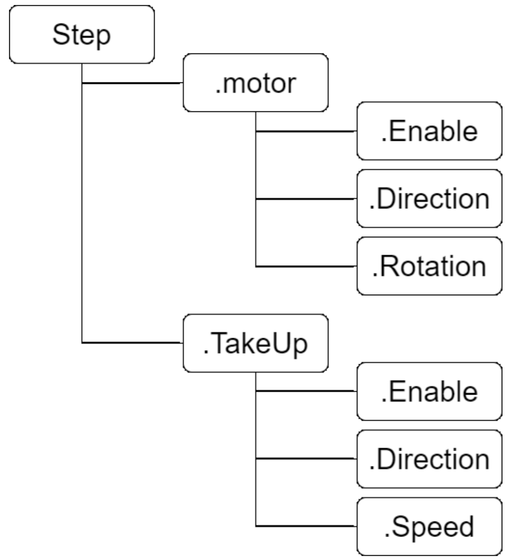


Figure 4-4: A graphical representation of the structure format used to store braid parameters

4.2.1.3 Machine Emulation

In the new method of machine emulation, the centers of all horngears are predefined as (row, column) vectors. The process of machine emulation is then performed roughly as described in previous work [5]. A 3D array is produced where each z-layer represents the resting place of each yarn after each step. The key difference here is that rather than all odd or even motors being rotated in a set direction on alternating steps, the rotation is now dependent on which motors are enabled, the direction they are specified to turn, and the angle through which they are to rotate. This process provides a framework for the plotting of the approximated yarn paths.

The next step of the machine emulation is to plot the approximated yarn paths. This is done by adding interim points between z-layers in the 3D array mentioned above. The application allows the user to enter the number of interim points. The x and y locations of these interim points are generated relative to the start and end positions of the yarn by simulating the rotation of a point in a 90° circular arc centered at the center of the horngear. The simulated rotation is performed in two steps, as shown in Figure 4-5. First, the quarter circle is drawn using the desired number of interim points at regular intervals. This step correlates to the path of the yarn and carrier through the braiding step. Second, the quarter circle is converted into a chord using the same intervals. This step correlates to the path of the yarn after tension is applied. The z locations of the interim points are determined based on the take-up speed. This is an additional feature that was not previously available. As described in prior work, the yarn paths are post-processed by running a zero-phase filter [5]. The filter serves to smooth the yarn paths to convert the straight lines previously generated into the undulating paths in the final model. The choice of a zero-phase filter ensures that the yarn paths aren't artificially shifted in the smoothing process. The feature of including the take-up speed also allows for modelling of braids where the take-up is paused. The ability for take-up to be paused allows for switching yarns without creating loose braids. The downside of this method is that when braids are switched, no interim points are created. Thus, the yarns directly intersect with one another as shown in Figure 4-6. The method developed for avoiding this issue is discussed in the following section.

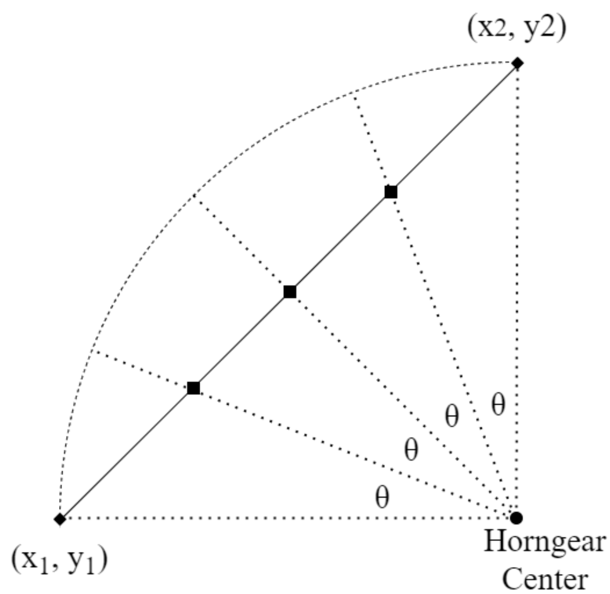


Figure 4-5: Example of simulated yarn path in a single braiding step before the application of the zero-phase filter. Shown here using 3 interim points.

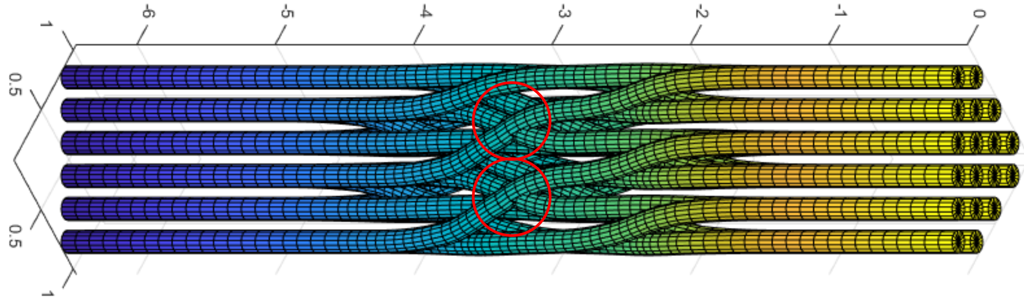


Figure 4-6: Model representation of the newly proposed isosceles triangle braid shape. Intersecting yarns are circled in red. Units are in mm.

4.2.1.4 Intersecting Yarns in Advanced Braids

One of the main benefits of the model created in [5] was that is was able to account for the undulation of yarns through the braid, thus avoiding artificial intersections. As such, it was vital to avoid new modeling parameters resulting in intersecting yarns. One parameter that could easily result in intersecting yarns was the ability to pause take-up during certain steps. This parameter was required for producing braids where it was necessary to switch yarns from one braid path to another. Switching was used to produce novel braids such as the triangle braid shapes discussed further in Section 4.3.2.

To avoid switching causing artificially intersecting yarns, a secondary smoothing step was added. This step finds any locations in the braid where yarns have switched and adjusts the (x, y, z) coordinates before and after the intersection point. Figure 4-7 shows the resulting model, which now lacks the intersecting yarns. This is done in a similar manner to the plotting of the approximate yarn paths described above. However, the start and end points of the arc were determined differently. In the typical procedure, the start and end points were taken to be the location of the yarn in the previous and current step respectively. If this was done for the switching steps, it would result in a highly localized change in braid angle at the exact point of the previous intersection. To avoid this, the secondary smoothing step chooses the start and end points to be halfway to the previous and subsequent steps respectively. This procedure ensures that the change in braid angle is less localized and allows for the model to reflect the real-world influences of the braid angles before and after the switch. Figure 4-8 shows that the application of the secondary smoothing step allowed the model to closely resemble visually what happens naturally with a real braid. Further characterization is needed to fully support this method.

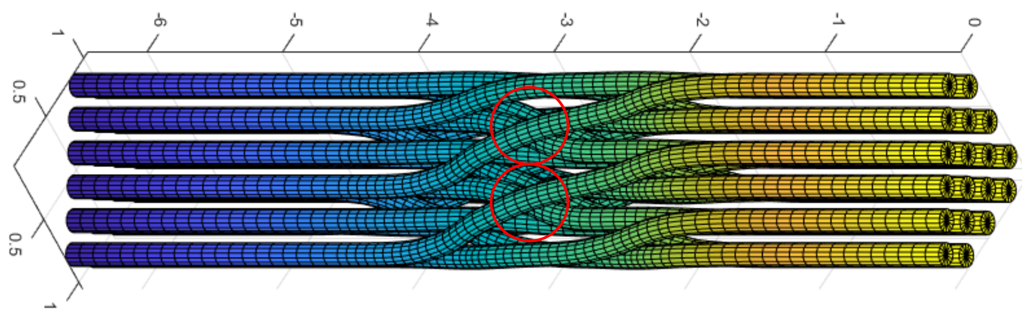


Figure 4-7: Model representation of the newly proposed isosceles triangle braid shape after secondary smoothing step applied to avoid intersecting yarns. Locations where yarns were previously intersecting are circled in red. Units are in mm.

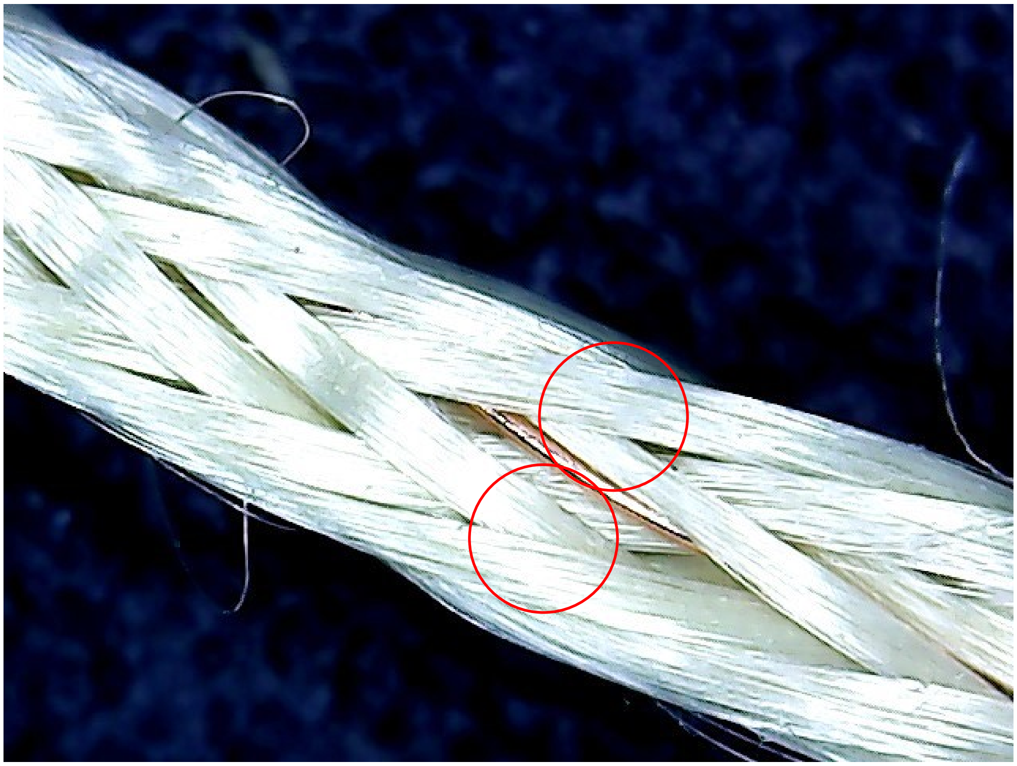


Figure 4-8: Fabricated isosceles triangle braid using a 3 x 3 braid setup at a takeup speed of 1 mm/min. Note the paths of the yarns at the switching location as circled in red.

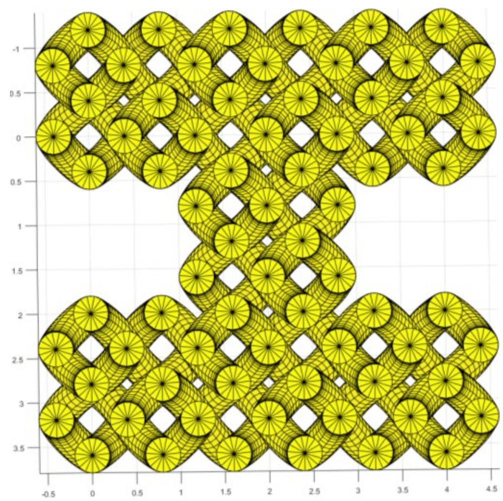
4.3 Results and Discussion

A great many braid shapes and patterns have been realized in literature. In particular the rotary braiding machines identified in [1], [11], [12] which use switching mechanisms seem to be the most flexible in terms of braid shapes. The most common braid shapes mentioned in literature are the I, T, L, C, box, and hat shapes. Typically, the use of sub-unit cell models has been used for square or rectangular shapes. Additionally, the previously developed method for sub-unit cell analysis involving undulations was only used for a 2-step, square, 3 x 3 braid shape [5], [42]. To further test the efficacy of this modeling method it was necessary to expand the model to allow for other shapes. Many other shapes can now be modeled with the application developed in this work, which will be discussed in the sections following.

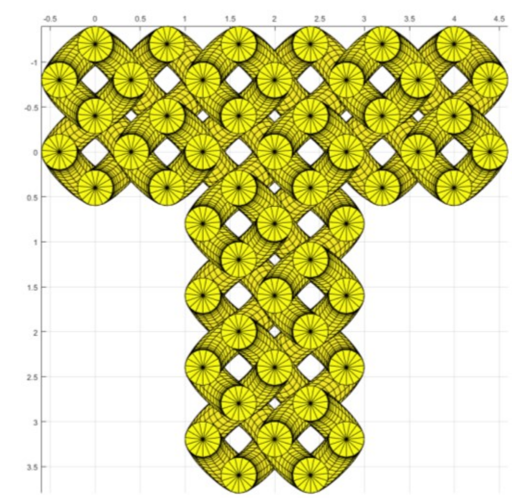
4.3.1 2-Step Orthogonal Shapes

Most of the common shapes in literature can be realized with simple 2-step braid modeling. For this work the most important shapes to model considered to be the I, T, L, C, H, and box shapes. This decision was made due to these shapes being commonly referenced in other 3D braiding work and due to these shapes being the most widely used for beams in infrastructure. These shapes, as generated using the modeling application developed herein are shown in Figure 4-9. Due to the simplicity of the modeling application, the models shown in Figure 4-9 can be produced in under a minute.

(a)



(b)



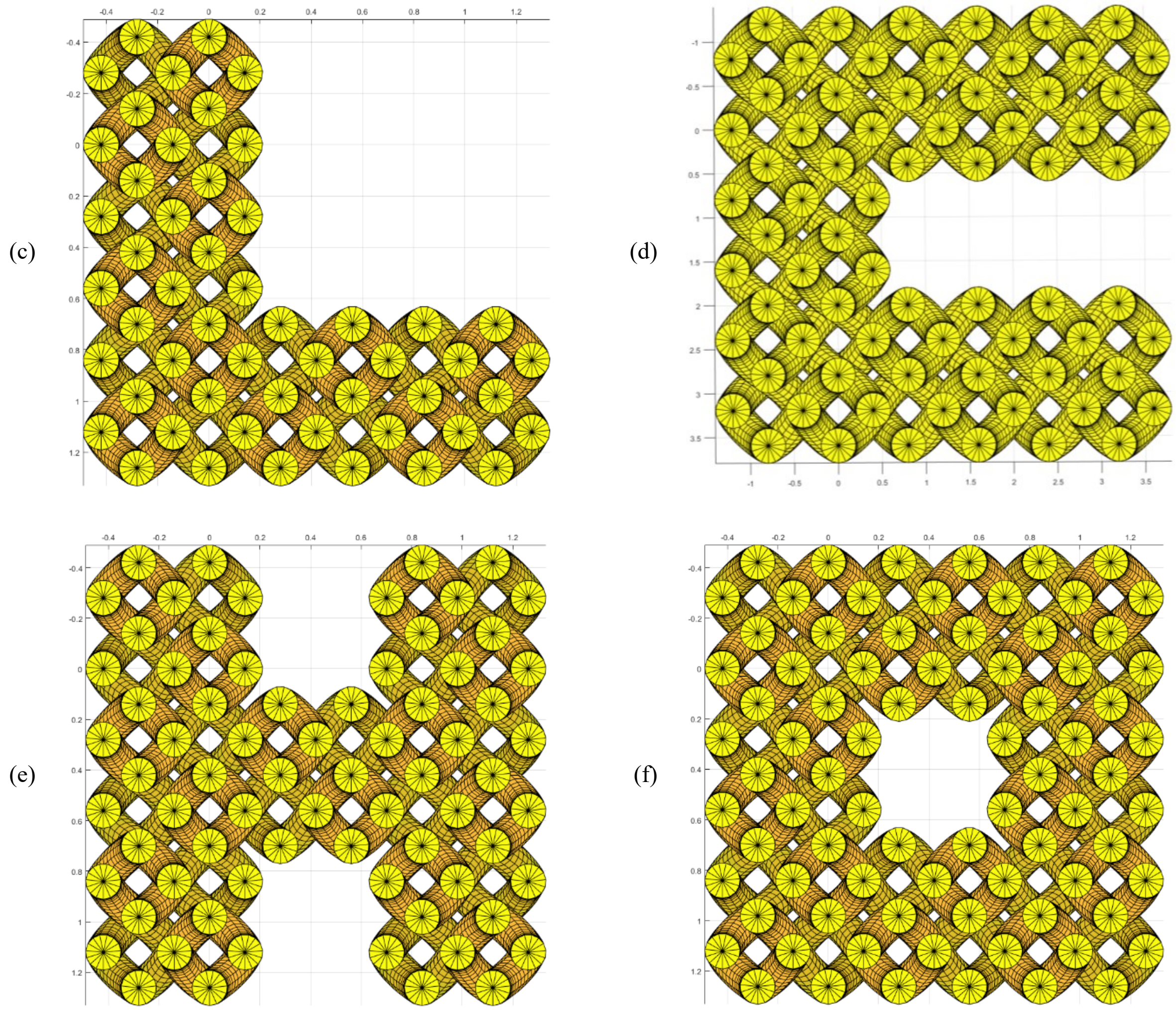


Figure 4-9: Figures of orthogonal, 2-step braids designed as possible replacements for existing steel beams. All shapes are made within a 6 x 6 horngear setup with 2 horngears for the thickness. Units are in mm

Previously, three sub-unit cells were used to model the 3D braids manufactured with this machine. These sub-unit cells are shown in Figure 4-10. With these three unit-cells – hereinafter referred to as corner, edge, and middle – any solid orthogonal 2-step braids can be modeled. Previously, these sub-unit cells were used to predict the axial Young's Modulus of a 2-step, square braids of various sizes [5]. To do so, the Young's Modulus of each sub-unit cell was estimated using finite element analysis (FEA). The sub-unit cells were then approximated as springs with the determined Young's

Moduli and combined to form the complete braid. The equation used for these predictions was as follows:

$$E_{composite,axial} = \frac{n_c E_c A_c + n_e E_e A_e + n_m E_m A_m}{A_{composite,xy}} \tag{4.1}$$

In this equation the subscripts c, e, and m, represent the corner, edge, and middle sub-unit cells respectively and *n* represents the number of said unit cell present in the braid. This equation was derived by approximating the behaviour of the sub-unit cells as a system of parallel springs [5]. In literature, it has been found that the corner and edge sub-unit cells generally have higher stiffness than the middle [52], [60], [65]. As such, it can be predicted that a braid with a larger proportion of edge and corner sub-unit cells would generally have a higher Young's Modulus than a braid with predominantly middle unit cells. Using this information, the predictive ranking shown in Table 4-1 was generated. Figures depicting the distribution of sub-unit cells within these braid shapes can be found in Appendix E.

Table 4-1: Predictive Stiffness Ranking for Various Braid Shapes

Shape	**Dimensions (L x W x T)**	n_c	n_e	n_m	**% Middle**	**Stiffness Ranking**
L	6 x 6 x 2	5	14	1	5%	1
C	6 x 6 x 2	6	18	2	8%	2
T	6 x 6 x 2	6	12	2	10%	3
Box	6 x 6 x 2	4	24	4	13%	4
I	6 x 6 x 2	8	16	4	14%	5
H	6 x 6 x 2	8	16	4	14%	5
Square	6 x 6	4	16	16	44%	7

(a) (b) (c)

Figure 4-10: Render of CAD models of sub-unit cells as developed in previous work [5]. (a) represents the corner sub-unit cell, (b) represents the edge sub-unit cell, and (c) represents the middle sub-unit cell.

4.3.2 Novel Multi-Step Braid Shapes

The most promising possibilities of the braiding machine described herein are the new shapes such as the isosceles or right-angle triangle. The isosceles triangle was originally investigated as a solution to the issue in traditional lay-up I-beams. The lay-up process leaves a gap at the bottom, which is typically filled with a cylindrical "noodle" [16]. The cylindrical shape still leaves gaps due to the more triangular shape of the hole. Therefore, the possibility to directly provide a triangular noodle would be useful in this scenario. The right-angle triangle similarly provides more possibilities for the use of 3D braiding machines industrially. One potential example would be flaps or ailerons on an airplane wing. Though braiding is still limited in cross sectional area by the size of the braiding machine bed, it is an exciting possibility to be able to potentially provide better near-net shapes. In a larger braiding machine, the combination of various angle triangles could be combined to approximate curved profiles. An example of this is shown in Figure 3-8. In a machine of this size the approximation of curves is still quite rough, but as the size of the braiding machine increases this approximation with improve relatively. It is also worth noting that the approximation of curves is not possible at all with a simple 2-step braid pattern.

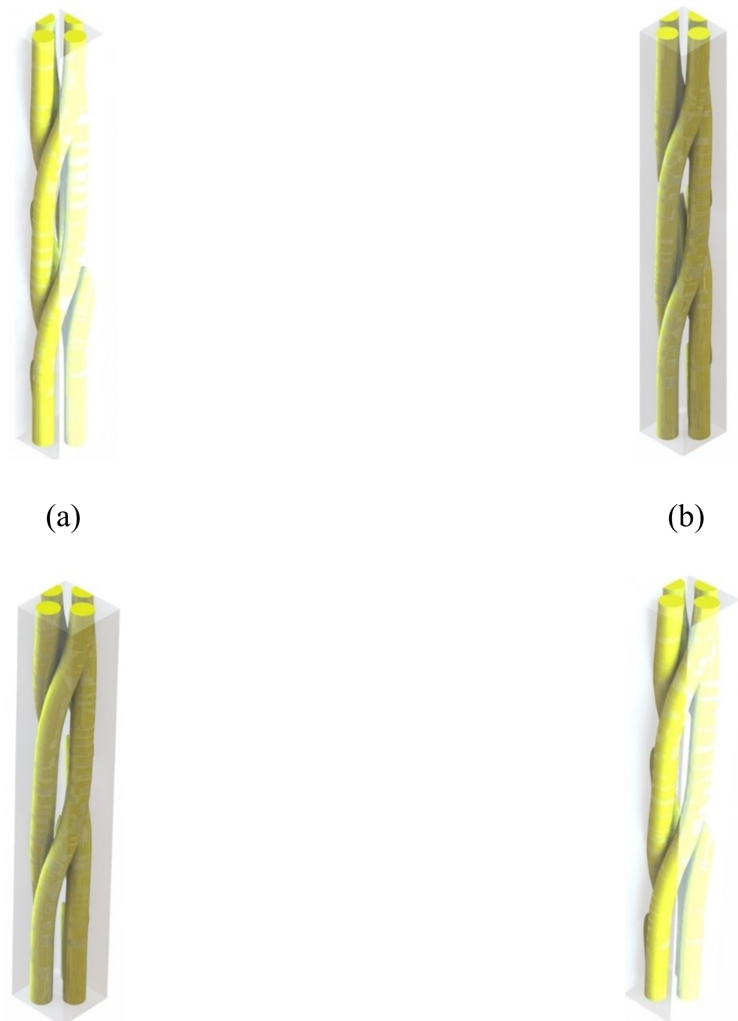
As these multi-step, angled braids have not been previously studied, it was required to generate new sub-unit cells in order for them to be modeled. While similar, the sub-unit cells for the isosceles triangle edge (ITE) and for the right triangle edge (RTE) differ because the angle of the yarn motion is inherently different due to the difference in angle of the edges themselves. Figure 4-11 shows the ITE and RTE sub-unit cells.

(a)

(b)

Figure 4-11: Novel Isosceles Triangle Edge (a) and Right Triangle Edge (b) sub-unit cells developed in this work

Due to the switching steps, the nearby corner sub-unit cells of both triangular braids were also affected. Where previously the corner sub-unit cells could be rotated as needed to match the braid paths for each corner, the corners for each triangle were unique. Therefore, further sub-unit cells were developed. These were designated "upper isosceles triangle corner" (UITC), "lower isosceles triangle corner" (LITC), "upper right triangle corner" (URTC), and "lower right triangle corner" (LRTC). Figure 4-12 shows these unit cells.



(a) (b)

(c) (d)

Figure 4-12: Novel Upper Isosceles Triangle Corner (a), Lower Isosceles Triangle Corner (b), Upper Right Triangle Corner (c), and Lower Right Triangle Corner (d) sub-unit cells developed in this work

Equations for the approximate braid angles for the surface yarns in the ITE and RTE sub-unit cells were developed based on Equations (3.1) and (3.2). For the ITE sub-unit cell, the transverse and axial distance travelled for the exterior yarn is the same distance travelled by an interior yarn in a typical 2-step braid. As such, the equation for external braid angle for the ITE sub-unit cell was equivalent to Equation (3.2). On the other hand, the exterior angle for the RTE sub-unit cell was significantly different due to the much-increased transverse travel. The equation for the exterior braid angle of the RTE sub-unit cell is as follows:

$$\theta_{ext} = tan^{-1}\left(\frac{4\sqrt{2-\sqrt{2}}rSF}{z}\right) \tag{4.2}$$

Using this information, it can be stated that for the same yarn radius, spacing factor, and braid pitch, both the ITE and RTE braid angles will both be larger than that of the edge sub-unit cell, with the RTE exterior angle being larger again than the ITE. As a smaller braid angle is associated with a higher longitudinal stiffness, it can be extrapolated that the triangular braids will likely have lower longitudinal stiffness than similarly sized orthogonal braids. On the other hand, larger braid angles equate to higher transverse stiffness. Therefore, triangular braids could prove to be useful in circumstances where the braid is facing multiaxial loads. Further, the method of switching used for triangular braids in this work could be incorporated into come orthogonal shapes. This would allow the orthogonal braids to maintain the higher longitudinal stiffness provided by the internal unit cells, while adding beneficial transverse stiffness on the exterior of the braid.

4.4 Conclusions and Future Work

Through this chapter, an improved method of facilitating sub-unit cell modeling for 3D braids has been presented. It was demonstrated that the improved method allows for the modelling of all orthogonal, 2-step braid shapes that could be used as replacements for existing industrial beams. Further, the relative stiffness rankings of the orthogonal, 2-step shapes were predicted using existing knowledge of the stiffness of sub-unit cells. Two new, multistep, triangular braids were also demonstrated. These braid shapes have not been previously shown in literature and required the development of new sub-unit cells. Using the estimated braid angles for these new shapes, it was predicted that they would have lower stiffness than orthogonal braids of similar size. Experimental validation of these predictions has yet to occur. Additionally, validation of the braid paths in the new braid shapes is required to support the assumptions made in the associated extra smoothing steps to prevent intersecting yarns.

Chapter 5 Micro-CT Analysis of Yarn Paths in 3D Braids

5.1 Introduction

Three-dimensional (3D) braiding is a method of producing textile preforms for composite materials. It is a continuation of two-dimensional (2D) braiding. A 3D braid differs from a 2D braid in that a 2D braid is composed of a single layer of intertwined yarns, whereas a 3D braid contains multiple. Further, the multiple layers of a 3D braid must be directly intertwined with one another with a 3D braiding process [1]. This differentiates 3D braiding from other multi-layer textile preforms such as stitched or woven fabrics.

These braids have been proposed for many applications. The most commonly suggested applications are replacement for existing composite structures in the aerospace [4] or automotive [8] industries, replacement of existing support structures in infrastructure [9], and replacement for various medical structures such as braided scaffolds [24]–[30], orthopedics [17], or stents [18]–[20]. For all of these applications it is critical that the mechanical properties of the braids can be accurately predicted. To do so, many attempts have been made to model 3D braids.

One of the most promising methods of modeling 3D braids is sub-unit cell modeling [47]. Sub-unit cell modeling recognizes that 3D braids have distinct yarn paths in the corner, edge, and middle of the braid. Since the development of the sub-unit cell model, many modifications to the model have been explored [42], [51]–[55], [57]–[62], [64]. In these models yarns are typically approximated as following straight paths within a sub-unit cell, although some have incorporated curved yarns into the corner or edge sub-unit cells [48], [57]. Recently, it was proposed that incorporating the undulation of the yarns into all sub-unit cells would result in more accurate predictions of sub-unit cell stiffness [42]. In this recent sub-unit cell model, a method of machine emulation was used to produce solid models of the sub-unit cells, which were then processed using finite element analysis to predict the stiffness properties [42]. However, the machine emulated model was not validated. As the undulations of the yarns were lauded as the primary advantage of this model, it was vital to provide validation for the paths of the yarns throughout the braid.

Micro-computed tomography (Micro-CT) has been used for validation of a variety of braid properties in literature. It has been shown to be effective for examining void content [66]–[68], damage mechanisms [69], [70], scaffold structure [17], and yarn paths [71], [72]. When using Micro-CT for braid paths, typically the tracing yarn was a different material than the remainder of the braid (i.e. glass fiber tracing yarn in a carbon fiber braid). This was done to increase the contrast between the tracing yarn and the braid based on the differing densities of the materials.

5.2 Methods

5.2.1 Braid Preparation

Two square, 3 x 3 braid samples were prepared as follows. The braid preforms were made using the braiding machine and application discussed throughout this work. The 3 x 3 horngear setup was fully populated with 24 yarns and carriers. A basic 2-step braiding pattern was selected as it was desired to validate the analysis process with a simple braid before continuing with more complex shapes and patterns.

The resin mixture used to impregnate the finished braid preforms was composed of Epon 826 (Hexion Inc., Ohio, USA) and Lindau LS-81K (Lindau Chemicals Inc., South Carolina, USA). This mixture was chosen to maintain consistency with 2D braids examined in previous work [73], [74]. This will allow knowledge from 2D braid experiments to be directly contrasted with results from 3D braid testing.

The mixture is made in a 1:1 ratio by weight in small quantities. Care is taken at this point to ensure that the braids are not unduly twisted. The resin mixture is massaged into the braids by hand along the full length of the braids in a consistent direction.

Curing is performed in an oven in 3 steps. Step one is 1.5 hours long and the oven temperature is set to 66°, step two is one hour long and the oven temperature is 85, and step three is three hours long and the oven temperature is 150°. This cycle is taken exactly from the datasheet provided for the mixture [75].

For a tracing yarn, it was decided to use extremely thin copper wire (approximately 28 gauge) wrapped loosely around the usual Kevlar® yarn. Figure 5-1 shows the wrapped tracer yarn.

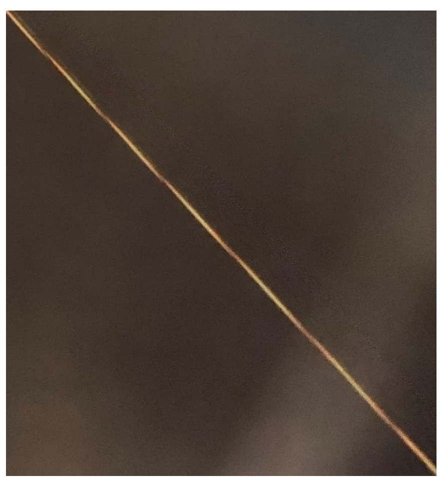


Figure 5-1: Copper-wrapped Kevlar® Tracer Yarn for micro-CT imaging

The decision to use copper-wrapped Kevlar® (Dupont, Delaware, USA) as the tracing yarn – rather than a glass or similar fiber – was made to facilitate viewing of the yarn in with the micro-CT machine while minimally changing the braiding process. It was assumed that the use of a fiber with a substantially different stiffness could unintentionally affect the braiding process, thus affecting the applicability to the model.

5.2.2 Model Preparation

The model of a single yarn path through the 2-step, 3 x 3 square braid was produced using the modeling application developed in previous work [5]. This application uses a method of machine emulation to produce a CAD model of the yarns within the braid. This method provides (x, y) coordinates of the yarn at a given z-height based on the emulation of the movement of the horngears. The x and y-coordinates are scaled using the radius of the yarn and a spacing factor, which is determined based on Equation 3.3. As the radius of the yarn was significantly smaller than the braid pitch, the spacing factor was approximated as equal to 1 for this study. The z-coordinates are scaled relative to a measured pitch for a braid formed with 1 mm/min take-up speed. The pitch was measured as the distance from the base of one yarn to the beginning of the next along the z-axis. It was measured in four locations and averaged to reduce the discrepancy caused by the inability to exactly select the start and end points of the measurement lines. Figure 5-2 shows the measurements used for this calculation and Table 5-1 contains the results of these measurements. The image was processed using ImageJ, which is an open-source image processing software that is widely used for similar applications. The ImageJ software uses the defined scale to measure the drawn lines based on the length of these lines in pixels. The measured pitch was thus 1.218 mm with a standard deviation of 0.041 mm, which corresponds to a relative error of 3.67%. The low error from this method of measurement shows that it is internally consistent. Measurements of more braids and over a longer section of braid are needed to determine the overall error in this measurement.

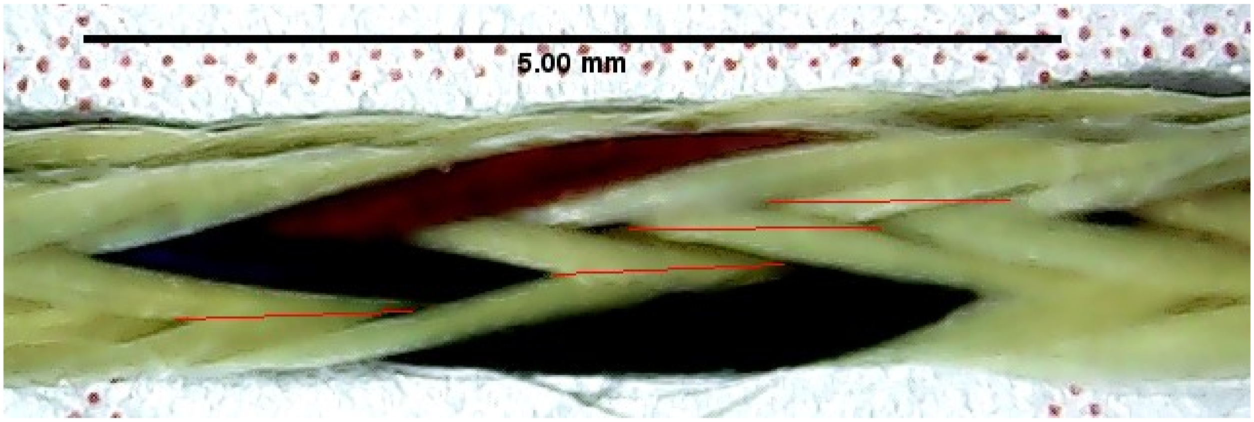


Figure 5-2: ImageJ measurement of braid pitch at 4 different locations (red lines). This braid was manufactured using the in-house braiding machine with a 3 x 3 horngear configuration at 1 mm/min take-up speed.

Table 5-1: Pitch measurements from Figure 5-2 and average and standard deviation information

	Pitch (mm)
1	1.272
2	1.174
3	1.222
4	1.203
Ave.	1.218
Std. Dev.	0.041

5.2.3 Micro-CT Analysis

Micro-CT Analysis was performed by Ali Gholami under the supervision of Dr. Garrett Melenka at York University. Table 5-2 shows the braid parameters used for the micro-CT analysis. The test parameters used for the micro-CT imaging are shown in Table 5-3.

Table 5-2: Micro-CT input parameters

Parameter	Predicted Value
Braid Width	1.3 mm
Geometry	Square
Angle	14-23°
Number of Yarns	24
Number of Tracing Yarns	1
Denier	200

Material	Specific Gravity
Yarn	1.44
Tracing Yarn (copper)	8.96
Matrix	1.16

Table 5-3: Micro-CT machine testing parameters

Parameter	Value
Source Voltage (kV)	50
Source Current (μA)	200
Image Pixel Size (μm)	4.999994
Result Image Width (pixels)	696
Result Image Height (pixels)	696
Exposure (ms)	798
Rotation Step (deg)	0.100
Frame Averaging	ON (5)
Flat Field Correction	ON
Filter	No Filter
Scan Duration	3h:13m:00s

The braid was held in place using a purpose-designed apparatus as shown in Figure 5-3.

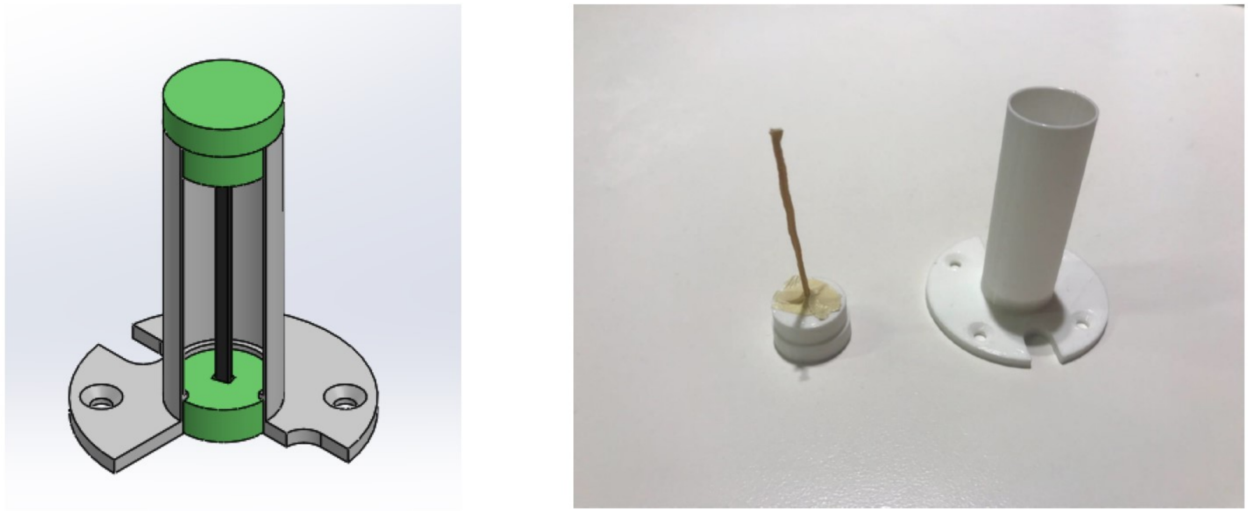


Figure 5-3: Apparatus for holding braid during micro-CT imaging

5.2.4 Numerical Comparison of Micro-CT Data and Model

The braid pitch and interior braid angle were chosen for numerical comparison between the modeled yarn path and the path as analysed using the Micro-CT images. Interior braid angle was chosen because for larger square braids, the interior braid angle has been shown to be the dominant angle affecting the braid stiffness [5]. The braid pitch was chosen because it is directly related to both interior and exterior braid angle as shown in Equations (3.1) and (3.2).

The micro-CT data originally included 3015 data points while the model only included 145. This was a result of the micro-CT taking images at ~0.5 mm increments. To enable more direct comparison of the two datasets, the micro-CT data was edited to only include points where the z-coordinate of a micro-CT point corresponded with the z-coordinate of a point in the modeled data. The angles of the two datasets at a z-coordinate (Point i) were estimated by taking the average of the angle of the Point (i+1) and the Point (i-1) measured relative to the z-axis. Figure 5-4 shows this procedure with an example point.

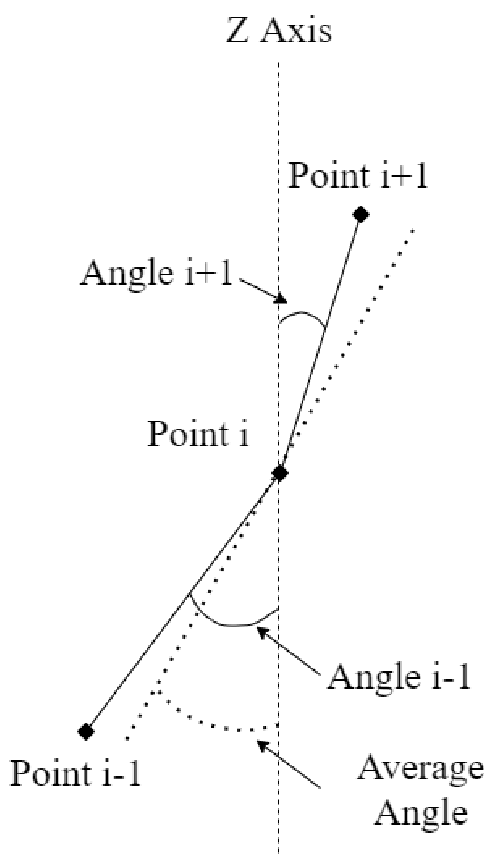


Figure 5-4: Representation of average angle approximation. Difference in angle is exaggerated to provide easier visibility. Shown in two dimensions for simplicity.

The average angles were plotted relative to the distance along the z-axis to give a visual representation of the change in angle throughout the yarn path. The interior braid angle should correspond with a long, relatively constant angle. These sections were quite clear for the modeled data shown in Figure 5-5. To estimate the interior braid angles using this plot, the areas of the braid where the angle remains relatively constant were grouped. This was done by grouping points together if the angle of either point (i+5) or point (i-5) was within 3°. The grouped sections can be seen in Figure 5-6. It was decided to compare with points ±5 z-steps away because this was equivalent of half of a braiding step in the model. It was assumed that if the angle remained relatively constant through half of a braiding step, then the angle could be considered constant. The allowable difference of up to 3° was similarly chosen based on the model as this was the maximum difference between local maxima and minima in the constant-angle sections of the modeled data. Using this method, the average estimated braid angle of the two constant-angle sections was 18.169°. The estimated value using Equation 3.1 was 18.034°. This resulted in a relative error between the graphical solution and the predicted of 0.75%, therefore this method was considered acceptable for predicting the interior braid angle.

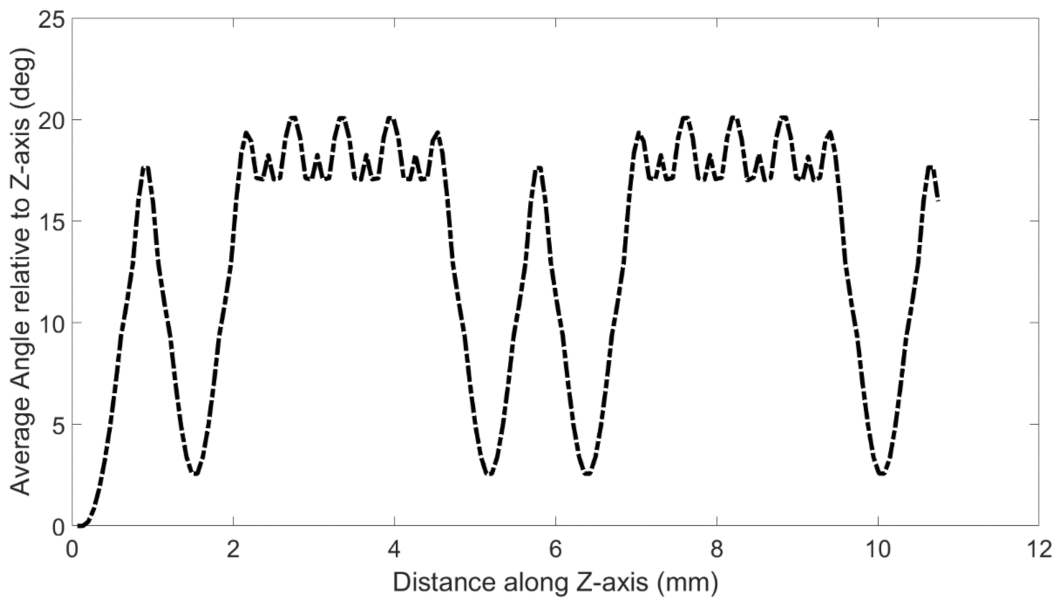


Figure 5-5: The change in braid angle as the yarn path travels along the z-axis. Note that the long, constant-angle sections in the modeled data correspond to the interior braid angle.

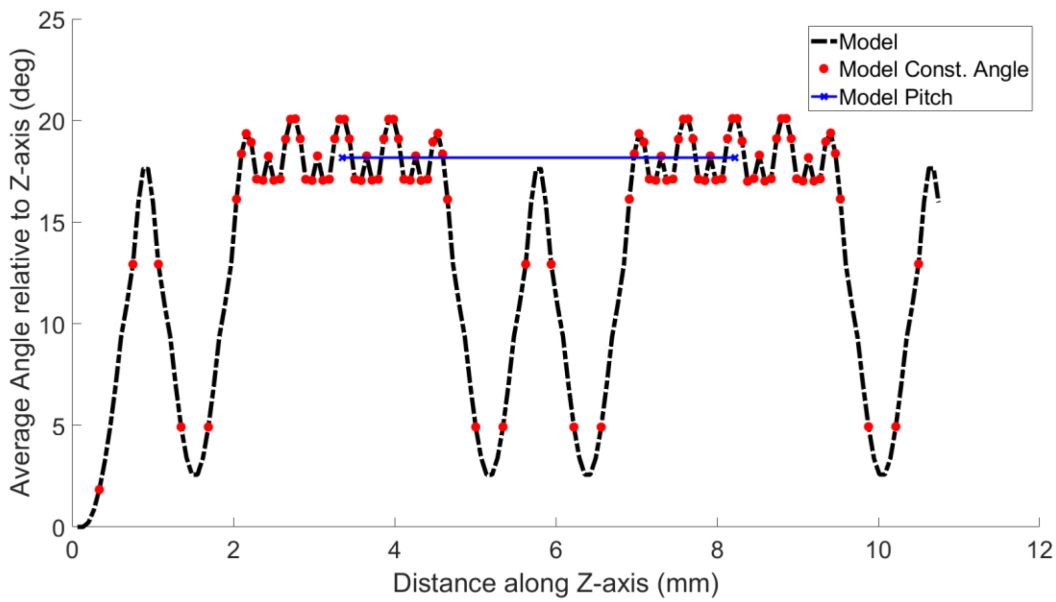


Figure 5-6: Modeled data with constant-angle sections grouped for pitch and internal braid angle approximation. Sections were grouped if points (i+5) or (i-5) were within 3°. The length used for the braid pitch estimation is shown at the estimated internal angle.

Each step in the braiding process was assumed to take half of the braid pitch to complete in the model. However, it was difficult to distinguish the start and end points of an individual step from the braid angle graph. Instead, the distance between the centers of the constant-angle sections was measured. This length involved 8 steps in the 3 x 3 model setup. Therefore, by dividing by 4 the pitch was estimated and found to match exactly with the measured pitch of 1.218. This result affirmed that the method of measuring the pitch from the centers of the constant-angle sections could be used for further analysis.

5.3 Results and Discussion

The yarn paths for the first sample were validated based on two parameters: the interior braid angle and the braid pitch. The yarn paths for the second sample have yet to be validated. The diagonal path in the first sample was considered to be more representative of the internal braid angle, therefore it was considered a better option for this study as a proof of concept. With the success of this first attempt using this method, the square path in the second sample will be analysed in future work. The predicted braid path for the first sample is shown in Figure 5-7. Figure 5-7a shows the cross-sectional path as the coordinate data output from the modeling application, while Figure 5-7b shows the CAD model generated from said data. When overlaid with the CAD model from the micro-CT analysis in Figure 5-8, it can be seen that the model achieves reasonably good agreement with the real braid. However, it is worth noting that as the micro-CT results travel in the axial direction, the discrepancy to the modeled path generally increases. It is suspected that this is due to a twist that was observed during braid formation. The twist in the braid was caused by an inadequate collection ring and inconsistency in the tensioning mechanisms of the bobbins. These issues will be rectified for later tests.

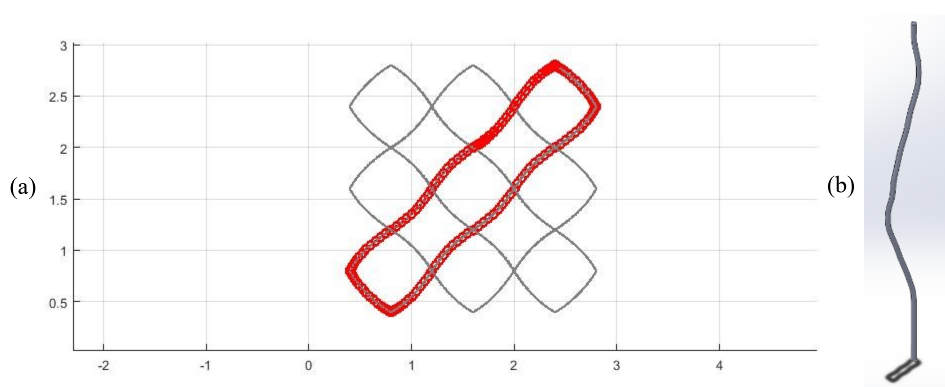


Figure 5-7: Predicted diagonal path of yarn through a 2-step, 3 x 3 square braid. (a) shows the model coordinate output from the top view. (b) shows the model CAD output from an isometric view

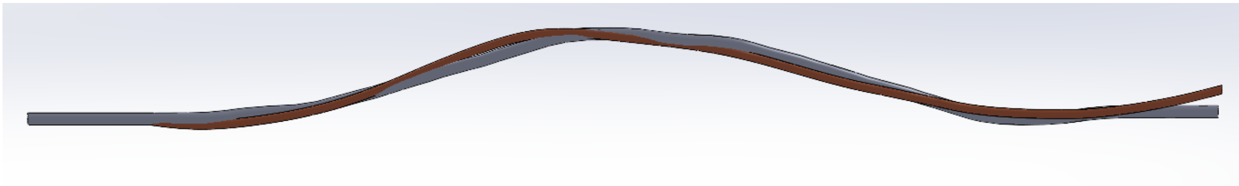


Figure 5-8: Model of single yarn along diagonal path in a 3 x 3 square braid overlaid with micro-CT results

The procedure described in Section 5.2.4 above was followed for numerical comparison between the model and the acquired micro-CT data. The general trend of the micro-CT braid angle compared with the expected trend from the model is shown in Figure 5-9. Generally, the micro-CT data follows a similar trend to the modeled data. Two relatively constant-angle sections are visible in both datasets as well as lower braid angles in between these sections. However, the constant-angle sections in the micro-CT data are constant over significantly smaller portions of the z-axis travel. This was expected as a real yarn is influenced by the external sections of the yarn both before and after the internal section. Another discrepancy was that where the model data showed large variation in the braid angle between the constant-angle sections, the micro-CT data showed a more consistent trend – including some additional constant-angle sections. These additional constant-angle sections correspond to the corner braid angle.

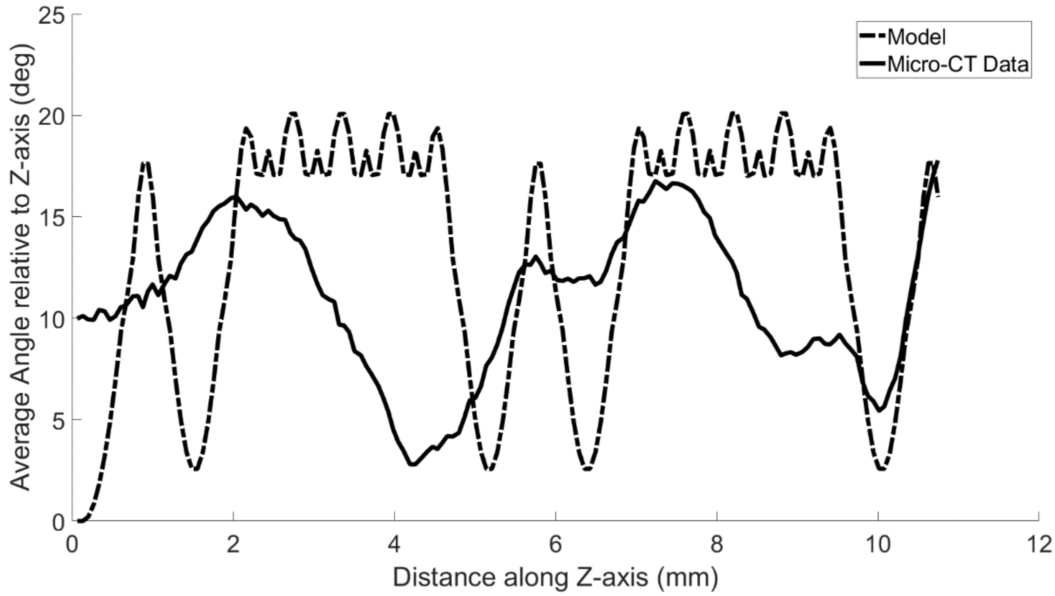


Figure 5-9: Comparison of the trend in braid angle with respect to the z-axis of the braid. Both model and actual yarns were produced in a 3 x 3 braid setup with a 1 mm/min take-up speed.

For the model, the difference between local maxima and minima in the constant-angle sections was found to be approximately 3°. For the micro-CT data, this difference was approximately 1° instead. Therefore, to locate the constant-angle sections in the micro-CT data, points were included if points (i+5) or (i-5) were within 1°. Figure 5-10 shows the results of this grouping. The data points recognized as part of the two, peak, constant-angle sections were averaged to determine the interior braid angle of the micro-CT data. The average for the first section was 15.312° and the average for the second was 16.303°, thus the overall average internal braid angle was estimated as 15.807°. When compared with the estimated angle of the model (18.169°) the micro-CT data had a relative error of 13.00%. As this result is determined from only two constant-angle sections, it is difficult to determine whether this error is truly representative. One further potential source of this error is that currently the model is defined relative to the measured pitch of 1.218 mm. As this pitch was measured from a single braid over a relatively small section of the braid, it is possible that this measured pitch could be an anomaly. The braid pitch and angle are intrinsically linked, so an error in one will inherently propagate error to the other.

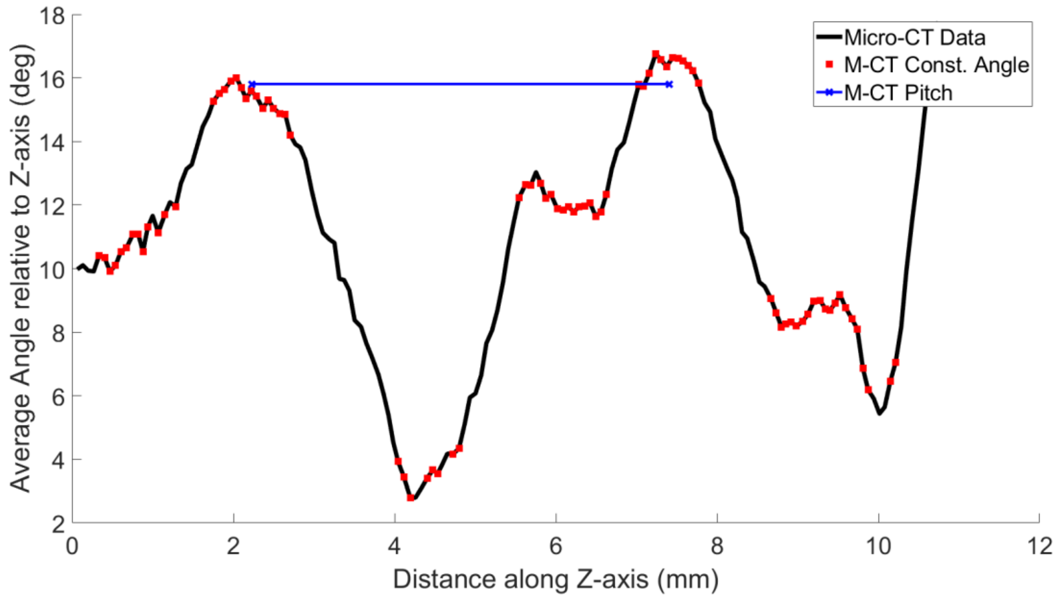


Figure 5-10: Modeled data with constant-angle sections grouped for pitch and internal braid angle approximation. Sections were grouped if points (i+5) or (i-5) were within 1°. The length used for the braid pitch estimation is shown at the estimated internal angle.

The pitch of the micro-CT data was analysed by measuring the distance between the average centers of the two constant-angle sections and dividing by 4 as described previously. The resulting length is shown in Figure 5-10 at the estimated internal braid angle. Using this method, the pitch was estimated to be 1.293 mm. When compared with the modeled pitch (1.218 mm), the relative error was found to be 6.16%. As mentioned previously, this error is likely caused due to both using a single braid for reference in the model and only acquiring a single pitch measurement from the results.

5.4 Conclusions and Future Work

Micro-CT analysis of braids by using thin copper wrapped around the yarn was presented. A 2-step, 3 x 3 braid was analysed using this method. It was shown to be effective at improving the contrast between the tracing yarn and the rest of the braid. Micro-CT results were compared with machine emulated models. The produced CAD models showed promising results to confirm the trajectory of the yarn through the braid. However, the paths were increasingly dissimilar as the distance in the z-direction of the braid increased. It was suggested that this was due to an applied twist during the manufacturing process. For future braids, twist in the braid needs to be minimized to ensure comparison with the model is possible for long sections of a braid. To aid in this endeavour, a new collection ring mechanism and curing rack have been designed.

Analysis of the internal braid angle determined that the modeled and micro-CT data differed by a relative error of 13.00%. Similarly, analysis of the braid pitch determined a relative error of 6.16% between the micro-CT data and the model. These errors were posited as resulting from a combination of using a single measured braid pitch for reference in the model and measuring only two internal angles and a single pitch in the micro-CT data. That these errors could be achieved under such conditions shows that this method of analysis has the potential to be beneficial for determining braid pitch and internal angle. In future tests, the pitch of many braids under the same conditions should be measured and the average used as the reference for the model. Additionally, it would be beneficial to analyse more braids in the micro-CT – and a longer length of each – to determine a better approximation of the internal braid angle. Further, a sensitivity study to determine the effect of an error in pitch measurement on the prediction of braid angle is vital to determine the acceptable level of error in the model.

Chapter 6 Conclusions, Limitations, and Future Work

6.1 Conclusions

The goals of this thesis were to produce a 3D braiding machine with the ability to create complex new braid shapes and patterns, to improve the method of modeling to allow for the modeling of these complex braid shapes and patterns, and to provide validation of the modeled yarn paths through a micro-CT analysis of the braids.

The braiding machine was shown to have greatly improved the ability for producing braid shapes and patterns. The increased bed size allows for both thicker cross-sections and complex switching mechanisms. The additional motor drivers allow for more direct control over individual fiber carriers to produce many new braid shapes and patterns. The braiding machine can produce all braid shapes and patterns that were found in literature, as well as some shapes that have not yet been presented elsewhere such as unique triangles and curve approximations. The changes to the braiding process and take-up mechanism were suggested as improvements to the consistency in the final braids.

The modeling software for the machine was updated to allow for the modeling of all newly possible braiding shapes and patterns. This included the ability to select individual yarns to load, the ability to enable individual motors to rotate through a selectable angle in either direction, and the ability to halt take-up for more advanced switching procedures. The modeling procedure was modified to allow for switching procedures while minimizing yarn intersection. The sub-unit cell modeling procedure was improved to allow for the modeling of more and more complex braiding shapes and patterns. Additionally, it was shown that the prediction of various braid angles as a result of new braiding patterns could be accomplished.

The micro-CT analysis provided promising results for comparison of CAD models developed with the micro-CT data and developed with the modeling application. However, confirmation of the cross-sectional braid path and interior braid angle were inconclusive. Suggestions were made for improving these comparisons in the future. The ability to accurately predict the yarn paths within the braid will serve to improve finite element models and feasibility for highly tailored braids for uses such as the medical field or aerospace.

6.2 Limitations

Sub-unit cell models were produced for a variety of new braid shapes and their relative stiffnesses were predicted. However, this thesis did not provide experimental or finite element validation of the stiffness values due to restrictions caused by the COVID-19 pandemic.

This section will discuss some limitations of the current braiding machine and modeling software. As discussed in Section **Error! Reference source not found.**5.3, the micro-CT result were promising but there were errors potentially caused by the twist in the braid. Until braids can be produced consistently, it will be difficult to fully determine the effectiveness of the model. Micro-CT results were also limited to only one braid sample as a proof of concept. In order to get a more statistically significant confirmation of the yarn paths throughout the braid, many samples should be analysed.

As discussed in Section 4.3.2, the smoothing of intersecting yarns for triangle shapes was performed by matching the model visually to braids. While this method is effective in reducing intersection of yarns, more work is needed to ensure that the smoothed path truly represents the reality of the final braids.

6.3 Future Work

Future work with the developed braiding machine and application should focus on validation of the models in multiple ways. Predicted stiffness values of the newly developed triangular shapes should be tested using finite element software. Overall predictions of the mechanical properties of the braid predicted using the modeling application need to be verified using experimental data. Braid pitch and estimated braid angles should be further explored using micro-CT analysis. Particularly, future micro-CT analysis should include further focus on minimizing twist in the braid during production, obtaining more measurements of pitch with which to normalize the model data, and a sensitivity study to determine the relative effect of an error in estimated braid angle on the resulting error in mechanical properties.

To mitigate the issue of braid twist in the future, a 3-axis maneuverable collection ring design has been developed. Figure 6-1 shows the CAD prototype of the 3-axis collection ring. This design uses materials that are already widely used in the braiding machine such as T-slotted aluminum to increase ease of use through familiarity. It also features a Teflon ® collection ring to reduce twisting caused by friction with the ring. The benefit of having 3-axis maneuverability is the 'x' and 'y' motion allows for the ring to be centered around the ideal collection point for any braid shape, while the 'z' motion allows for the ring to be moved for different take-up speeds. This collection ring was intended to be implemented in the current braiding machine design but had to be delayed due to lab closures.

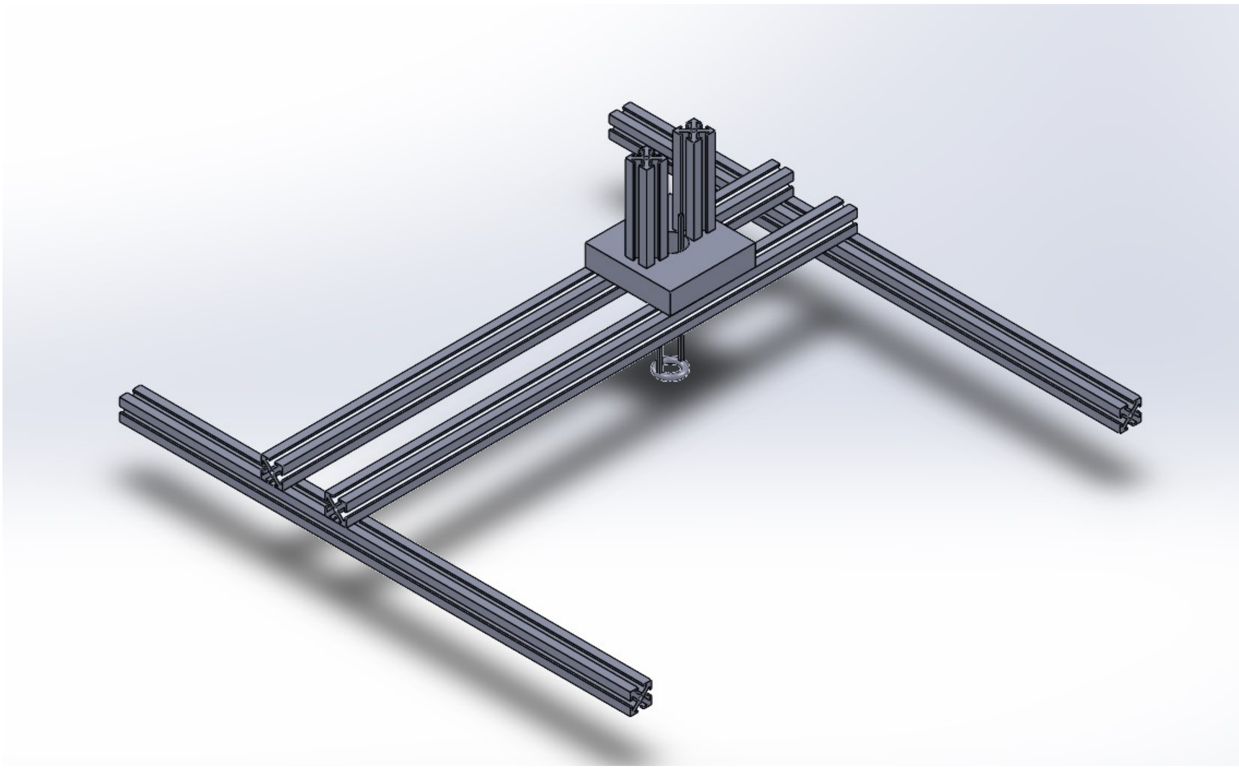


Figure 6-1: CAD model of a proposed 3-axis collection ring mechanism to allow for collection to occur at any location or height within the braiding machine build volume

Bibliography

[1] A. E. Bogdanovich, “An overview of three-dimensional braiding technologies,” in *Advances in Braiding Technology*, Elsevier, 2016, pp. 3–78.

[2] P. M. Wambua and R. Anandjiwala, “A Review of Preforms for the Composites Industry,” *Journal of Industrial Textiles*, vol. 40, no. 4, pp. 310–333, Apr. 2011, doi: 10.1177/1528083709092014.

[3] M. B. Dow and H. B. Dexter, “Development of Stitched, Braided and Woven Composite Structures in the ACT Program and at Langley Research Center,” Hampton, VA, 1997. [Online]. Available: https://ntrs.nasa.gov/search.jsp?R=19980000063.

[4] A. P. Mouritz, M. K. Bannister, P. J. Falzon, and K. H. Leong, “Review of applications for advanced three-dimensional fibre textile composites,” *Composites Part A: Applied Science and Manufacturing*, vol. 30, no. 12, pp. 1445–1461, Dec. 1999, doi: 10.1016/S1359-835X(99)00034-2.

[5] D. R. Aldrich, “Design and Development of a New Model for Predicting the Mechanical Properties of Three-Dimensional Braided Composite Materials,” 2018.

[6] C. Ayranci and J. Carey, “2D braided composites: A review for stiffness critical applications,” *Composite Structures*, vol. 85, no. 1, pp. 43–58, Sep. 2008, doi: 10.1016/j.compstruct.2007.10.004.

[7] G. W. Melenka, C. M. Pastore, F. K. Ko, and J. P. Carey, “Advances in 2-D and 3-D braided composite material modeling,” in *Handbook of Advances in Braided Composite Materials: Theory, Production, Testing and Applications*, Elsevier Inc., 2017, pp. 321–363.

[8] K. Bilisik, “Two-dimensional (2D) fabrics and three-dimensional (3D) preforms for ballistic and stabbing protection: A review,” *Textile Research Journal*, vol. 87, no. 18, pp. 2275–2304, Nov. 2017, doi: 10.1177/0040517516669075.

[9] L. Heller, D. Vokoun, P. Šittner, and H. Finckh, “3D flexible NiTi-braided elastomer composites for smart structure applications,” *Smart Materials and Structures*, vol. 21, no. 4, p. 045016, Mar. 2012, doi: 10.1088/0964-1726/21/4/045016.

[10] D. Mungalov and A. Bogdanovich, “Automated 3-D braiding machine and method,” US6439096B1, Nov. 28, 2000.

[11] F. Schreiber, F. K. Ko, H. J. Yang, E. Amalric, and T. Gries, “NOVEL THREE-DIMENSIONAL BRAIDING APPROACH AND ITS PRODUCTS,” IOM Communications, 2009. Accessed: Jun. 07, 2020. [Online].

[12] T. Sontag, H. Yang, T. Gries, and F. Ko, “Recent advances in 3D braiding technology,” in *Advances in 3D Textiles*, Elsevier, 2015, pp. 153–181.

[13] K. Brakefield, “The Various Types of Structural Steel Shapes,” *Swanton Welding & Machining*. https://blog.swantonweld.com/the-various-types-of-structural-steel-shapes (accessed Jun. 15, 2020).

[14] “The Various Types of Structural Steel Shapes,” *Midwest Steel*. https://www.midweststeel.com/about/news/The-Various-Types-of-Structural-Steel-Shapes_AE37.html (accessed Jun. 15, 2020).

[15] "Types of Structural Steel Shapes," *Thomas Publishing Company*. https://www.thomasnet.com/articles/metals-metal-products/types-of-steel-beams-and-shapes/ (accessed Jun. 15, 2020).

[16] J. Bremmer, R. A. Lacko, J. G. Sauer, P. H. Denavit, W. E. Hovan, and E. J. Fabian, "Conformal deltoid noodle for a composite structure," US8440045B2, Sep. 14, 2013.

[17] H. Ahn, K. J. Kim, S. Y. Park, J. E. Huh, H. J. Kim, and W. R. Yu, "3D braid scaffolds for regeneration of articular cartilage," *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 34, pp. 37–46, Jun. 2014, doi: 10.1016/j.jmbbm.2014.01.004.

[18] J. G. Barber, A. M. Handorf, T. J. Allee, and W. J. Li, "Braided nanofibrous scaffold for tendon and ligament tissue engineering," *Tissue Engineering - Part A*, vol. 19, no. 11–12, pp. 1265–1274, Jun. 2013, doi: 10.1089/ten.tea.2010.0538.

[19] U. A. Gurkan, X. Cheng, V. Kishore, J. A. Uquillas, and O. Akkus, "Comparison of morphology, orientation, and migration of tendon derived fibroblasts and bone marrow stromal cells on electrochemically aligned collagen constructs," *Journal of Biomedical Materials Research Part A*, vol. 9999A, no. 4, p. NA-NA, Sep. 2010, doi: 10.1002/jbm.a.32783.

[20] S. Madhavarapu, R. Rao, S. Libring, E. Fleisher, Y. Yankannah, and J. W. Freeman, "Design and characterization of three-dimensional twist-braid scaffolds for anterior cruciate ligament regeneration," *TECHNOLOGY*, vol. 05, no. 02, pp. 98–106, Jun. 2017, doi: 10.1142/s2339547817500066.

[21] X. Yuan, A. F. T. Mak, and J. Li, "Formation of bone-like apatite on poly(L-lactic acid) fibers by a biomimetic process," *Journal of Biomedical Materials Research*, vol. 57, no. 1,

pp. 140–150, Oct. 2001, doi: 10.1002/1097-4636(200110)57:1<140::AID-JBM1153>3.0.CO;2-G.

[22] Y. Z. Wan *et al.*, “Characterization of three-dimensional braided carbon/Kevlar hybrid composites for orthopedic usage,” *Materials Science and Engineering A*, vol. 398, no. 1–2, pp. 227–232, May 2005, doi: 10.1016/j.msea.2005.03.010.

[23] H. Luo *et al.*, “Preparation of three-dimensional braided carbon fiber-reinforced PEEK composites for potential load-bearing bone fixations. Part I. Mechanical properties and cytocompatibility,” *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 29, pp. 103–113, Jan. 2014, doi: 10.1016/j.jmbbm.2013.09.003.

[24] M. Henry *et al.*, “Treatment of renal artery aneurysm with the multilayer stent,” *Journal of Endovascular Therapy*, vol. 15, no. 2, pp. 231–236, Apr. 2008, doi: 10.1583/07-2222.1.

[25] C. Shanahan, P. Tiernan, and S. A. M. Tofail, “Looped ends versus open ends braided stent: A comparison of the mechanical behaviour using analytical and numerical methods,” *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 75, pp. 581–591, Nov. 2017, doi: 10.1016/j.jmbbm.2017.08.025.

[26] M. de Beule *et al.*, “Virtual optimization of self-expandable braided wire stents,” *Medical Engineering and Physics*, vol. 31, no. 4, pp. 448–453, May 2009, doi: 10.1016/j.medengphy.2008.11.008.

[27] M. Frost, P. Sedlák, A. Kruisová, and M. Landa, “Simulations of self-expanding braided stent using macroscopic model of NiTi shape memory alloys covering R-Phase,” in *Journal of Materials Engineering and Performance*, 2014, vol. 23, no. 7, pp. 2584–2590, doi: 10.1007/s11665-014-0966-z.

[28] X. Y. Ni, C. W. Pan, and B. Gangadhara Prusty, “Numerical investigations of the mechanical properties of a braided non-vascular stent design using finite element method,” *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 18, no. 10, pp. 1117–1125, Jul. 2015, doi: 10.1080/10255842.2013.873420.

[29] J. H. Kim, T. J. Kang, and W. R. Yu, “Mechanical modeling of self-expandable stent fabricated using braiding technology,” *Journal of Biomechanics*, vol. 41, no. 15, pp. 3202–3212, Nov. 2008, doi: 10.1016/j.jbiomech.2008.08.005.

[30] J. Hyun Kim, T. Jin Kang, and W. R. Yu, “Simulation of mechanical behavior of temperature-responsive braided stents made of shape memory polyurethanes,” *Journal of Biomechanics*, vol. 43, no. 4, pp. 632–643, Mar. 2010, doi: 10.1016/j.jbiomech.2009.10.032.

[31] R. M. Bluck, “High speed bias weaving and braiding,” Dec. 1969. Accessed: May 24, 2020. [Online].

[32] S. B. Blaisdell, “BRAIDED BRAKE LINING AND MACHINE FOR MAKING SAME,” Aug. 1930. Accessed: May 24, 2020. [Online].

[33] R. A. Florentine, “Apparatus for weaving a three-dimensional article,” US4312261, May 27, 1980.

[34] R. A. Florentine, “Magnaweave Process—From Fundamentals to Applications,” *Textile Research Journal*, vol. 53, no. 10, pp. 620–623, Oct. 1983, doi: 10.1177/004051758305301008.

[35] K. Bilisik, “Cartesian 3D braiding,” in *Advances in Braiding Technology*, Elsevier, 2016, pp. 107–145.

[36] T. D. Kostar and T. W. Chou, “Process simulation and fabrication of advanced multi-step three-dimensional braided preforms,” *Journal of Materials Science*, vol. 29, no. 8, pp. 2159–2167, Apr. 1994, doi: 10.1007/BF01154695.

[37] T. D. Kostar and T.-W. Chou, “Microstructural Design of Advanced Multi-Step Three-Dimensional Braided Preforms,” *Journal of Composite Materials*, vol. 28, no. 13, pp. 1180–1201, Jul. 1994, doi: 10.1177/002199839402801301.

[38] Y. Kyosev, *Braiding technology for textiles: Principles, design and processes*. Elsevier Inc., 2014.

[39] M. Tsuzuki, M. Kimbara, K. Fukuta, and A. Machii, “Three-dimensional fabric woven by interlacing threads with rotor driven carriers,” US5067525, Dec. 28, 1989.

[40] M. Tsuzuki, “Three-dimensional woven fabric with varied thread orientations,” US5348056, Apr. 23, 1992.

[41] Q. Gu, Z. Quan, J. Yu, J. Yan, B. Sun, and G. Xu, “Structural modeling and mechanical characterizing of three-dimensional four-step braided composites: A review,” *Composite Structures*, vol. 207. Elsevier Ltd, pp. 119–128, Jan. 01, 2019, doi: 10.1016/j.compstruct.2018.09.065.

[42] D. R. Aldrich and C. Ayranci, “EFFECTS OF YARN PATH AND SUB-UNIT CELLS ON THE CHARACTERIZATION OF MECHANICAL PROPERTIES OF 3D BRAIDED COMPOSITE MATERIALS,” Aug. 2017, Accessed: May 31, 2020. [Online].

[43] C. M. P. FK Ko, J. Vinson, and M. Taya, *Structure and Properties of an Integrated 3-D Fabric for Structural Composites*. West Conshohocken, PA: ASTM International, 1985.

[44] C. M. Pastore and Y. A. Gowayed, “Self-consistent fabric geometry model: modification and application of a fabric geometry model to predict the elastic properties of textile composites,” *Journal of Composites Technology and Research*, vol. 16, no. 1, pp. 32–36, Jan. 1994, doi: 10.1520/CTR10392J.

[45] T. Ishikawa and T. W. Chou, “Stiffness and strength behaviour of woven fabric composites,” *Journal of Materials Science*, vol. 17, no. 11, pp. 3211–3220, Nov. 1982, doi: 10.1007/BF01203485.

[46] J.-M. Yang, C.-L. Ma, and T.-W. Chou, “Fiber Inclination Model of Three-Dimensional Textile Structural Composites,” *Journal of Composite Materials*, vol. 20, no. 5, pp. 472–484, Sep. 1986, doi: 10.1177/002199838602000505.

[47] Y. Q. Wang and A. S. D. Wang, “On the topological yarn structure of 3-D rectangular and tubular braided preforms,” *Composites Science and Technology*, vol. 51, no. 4, pp. 575–586, Jan. 1994, doi: 10.1016/0266-3538(94)90090-6.

[48] W. Zhang, X. Ding, and Y. Li, “Microstructure of 3D Braided Preform for Composites with Complex Rectangular Cross-section,” *Journal of Composite Materials*, vol. 41, no. 25, pp. 2975–2983, Dec. 2007, doi: 10.1177/0021998307082179.

[49] Wei Zhang, Xin Ding, and Yuling Li, “Calculation and Design of Parameters for Four-step 3D Braided Preform with Complex Rectangular Cross Sections,” *Journal of Industrial Textiles*, vol. 38, no. 2, pp. 139–150, Oct. 2008, doi: 10.1177/1528083707087838.

[50] Y. Q. Wang and A. S. D. Wang, “Microstructure/property relationships in three-dimensionally braided fiber composites,” *Composites Science and Technology*, vol. 53, no. 2, pp. 213–222, Jan. 1995, doi: 10.1016/0266-3538(95)00021-6.

[51] D. L. Wu, “Three-cell model and 5D braided structural composites,” *Composites Science and Technology*, vol. 56, no. 3, pp. 225–233, Jan. 1996, doi: 10.1016/0266-3538(95)00136-0.

[52] L. Chen, X. M. Tao, and C. L. Choy, “Mechanical analysis of 3-D braided composites by the finite multiphase element method,” *Composites Science and Technology*, vol. 59, no. 16, pp. 2383–2391, Dec. 1999, doi: 10.1016/S0266-3538(99)00087-1.

[53] Y. Wang and X. Sun, “Digital-element simulation of textile processes,” *Composites Science and Technology*, vol. 61, no. 2, pp. 311–319, Feb. 2001, doi: 10.1016/S0266-3538(00)00223-2.

[54] D. sen Li, Z. xing Lu, L. Chen, and J. lu Li, “Microstructure and mechanical properties of three-dimensional five-directional braided composites,” *International Journal of Solids and Structures*, vol. 46, no. 18–19, pp. 3422–3432, Sep. 2009, doi: 10.1016/j.ijsolstr.2009.05.013.

[55] D. sen Li, J. L. Li, L. Chen, Z. X. Lu, and D. N. Fang, “Finite element analysis of mechanical properties of 3D four-directional rectangular braided composites part 1: Microgeometry and 3D finite element model,” *Applied Composite Materials*, vol. 17, no. 4, pp. 373–387, Aug. 2010, doi: 10.1007/s10443-010-9126-2.

[56] D. sen Li, D. N. Fang, N. Jiang, and Y. Xuefeng, “Finite element modeling of mechanical properties of 3D five-directional rectangular braided composites,” *Composites Part B:*

Engineering, vol. 42, no. 6, pp. 1373–1385, Sep. 2011, doi: 10.1016/j.compositesb.2011.05.042.

[57] M. M. Shokrieh and M. S. Mazloomi, “A new analytical model for calculation of stiffness of three-dimensional four-directional braided composites,” *Composite Structures*, vol. 94, no. 3, pp. 1005–1015, Feb. 2012, doi: 10.1016/j.compstruct.2011.09.010.

[58] C. Zhang and X. Xu, “Finite element analysis of 3D braided composites based on three unit-cells models,” *Composite Structures*, vol. 98, pp. 130–142, Apr. 2013, doi: 10.1016/j.compstruct.2012.11.003.

[59] K. Xu and X. Qian, “Microstructure Analysis and Multi-Unit Cell Model of Three Dimensionally Four-Directional Braided Composites,” *Applied Composite Materials*, vol. 22, no. 1, pp. 29–50, May 2014, doi: 10.1007/s10443-014-9396-1.

[60] K. Xu and X. Qian, “Analytical prediction of the elastic properties of 3D braided composites based on a new multiunit cell model with consideration of yarn distortion,” *Mechanics of Materials*, vol. 92, pp. 139–154, Jan. 2016, doi: 10.1016/j.mechmat.2015.09.007.

[61] J. Sun, G. Zhou, C. Zhou, and X. Wang, “In-plane shear investigation of 3D surface-core braided composites,” *Composites Science and Technology*, vol. 135, pp. 54–66, Oct. 2016, doi: 10.1016/j.compscitech.2016.09.011.

[62] C. Zhang, J. L. Curiel-Sosa, and E. A. Duodu, “Finite element analysis of the damage mechanism of 3D braided composites under high-velocity impact,” *Journal of Materials Science*, vol. 52, no. 8, pp. 4658–4674, Apr. 2017, doi: 10.1007/s10853-016-0709-7.

[63] J.-M. Yang, C.-L. Ma, and T.-W. Chou, "Fiber Inclination Model of Three-Dimensional Textile Structural Composites," *Journal of Composite Materials*, vol. 20, no. 5, pp. 472–484, Sep. 1986, doi: 10.1177/002199838602000505.

[64] D. sen Li, D. N. Fang, N. Jiang, and Y. Xuefeng, "Finite element modeling of mechanical properties of 3D five-directional rectangular braided composites," *Composites Part B: Engineering*, vol. 42, no. 6, pp. 1373–1385, Sep. 2011, doi: 10.1016/j.compositesb.2011.05.042.

[65] C. Zhang and X. Xu, "Finite element analysis of 3D braided composites based on three unit-cells models," *Composite Structures*, vol. 98, pp. 130–142, Apr. 2013, doi: 10.1016/j.compstruct.2012.11.003.

[66] G. W. Melenka, B. M. Bruni-Bossio, C. Ayranci, and J. P. Carey, "Examination of voids and geometry of bio-based braided composite structures," *IOP Conference Series: Materials Science and Engineering*, vol. 406, p. 012012, 2018, doi: 10.1088/1757-899x/406/1/012012.

[67] G. W. Melenka, E. Lepp, B. K. O. Cheung, and J. P. Carey, "Micro-computed tomography analysis of tubular braided composites," *Composite Structures*, vol. 131, pp. 384–396, Nov. 2015, doi: 10.1016/j.compstruct.2015.05.057.

[68] G. W. Melenka and C. Ayranci, "Advanced measurement techniques for braided composite structures: A review of current and upcoming trends," *Journal of Composite Materials*, vol. 54, pp. 3895–3917, 2020, doi: 10.1177/0021998320903105.

[69] H. Zhou, C. Li, L. Zhang, B. Crawford, A. S. Milani, and F. K. Ko, "Micro-XCT analysis of damage mechanisms in 3D circular braided composite tubes under transverse impact,"

Composites Science and Technology, vol. 155, pp. 91–99, Feb. 2018, doi: 10.1016/j.compscitech.2017.11.025.

[70] Y. Chen *et al.*, “Crack initiation and propagation in braided SiC/SiC composite tubes: Effect of braiding angle,” *Journal of the European Ceramic Society*, vol. 40, no. 13, pp. 4403–4418, Oct. 2020, doi: 10.1016/j.jeurceramsoc.2020.04.060.

[71] Y. bo Wang, Z. guo Liu, N. Liu, L. Hu, Y. chen Wei, and J. jun Ou, “A new geometric modelling approach for 3D braided tubular composites base on Free Form Deformation,” *Composite Structures*, vol. 136, pp. 75–85, Feb. 2016, doi: 10.1016/j.compstruct.2015.09.036.

[72] L. Hu, G. Tao, Z. Liu, Y. Wang, and J. Ya, “Investigation on the Yarn Squeezing Effect of Three Dimensional Full Five Directional Braided Composites,” *Applied Composite Materials*, vol. 26, pp. 371–387, 2019, doi: 10.1007/s10443-018-9697-x.

[73] A. Ead, “Experimental Study of Creep Behaviour of Two Dimensional Kevlar Braided Composites,” 2018, doi: 10.7939/R35Q4S28F.

[74] G. W. Melenka and J. P. Carey, “Experimental analysis of diamond and regular tubular braided composites using three-dimensional digital image correlation,” *Journal of Composite Materials*, vol. 51, no. 28, pp. 3887–3907, Dec. 2017, doi: 10.1177/0021998317695418.

[75] “Starting Formulation SF 8017 Epoxy Resin System for Pultrusion or Filament Winding EPON^{TM} Resin 826 or EPON^{TM} Resin 862 / LS-81K Anhydride Curing Agent (MTHPA).” Accessed: Jun. 16, 2020. [Online].

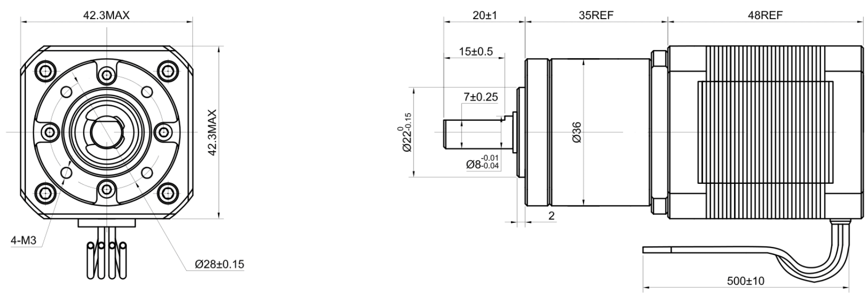
[76] “Structure array - MATLAB.” https://www.mathworks.com/help/matlab/ref/struct.html (accessed Jun. 16, 2020).

[77] J. Valdez and J. Becker, “Understanding the I2C Bus,” Jun. 2015. Accessed: Jun. 16, 2020. [Online]. Available: www.ti.com.

[78] “Arduino - BlinkWithoutDelay.” https://www.arduino.cc/en/Tutorial/BlinkWithoutDelay (accessed Jun. 16, 2020).

Appendix A Machine Component Datasheets

A.1 Nema 17 Stepper Motor with gear box



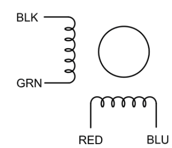
SPECIFICATION \ CONNECTION	BIPOLAR
AMPS/PHASE	1.68
RESISTANCE/PHASE(Ohms)@25°C	1.80±10%
INDUCTANCE/PHASE(mH)@1KHz	3.20±20%
HOLDING TORQUE w/o GEARBOX(Nm)[lb-in]	0.52[4.60]
GEAR RATIO	$19\frac{38}{187}$
EFFICIENCY	81.00%
STEP ANGLE w/o GEARBOX(°)	1.80
BACKLASH@NO-LOAD	<=1°
MAX.PERMISSIBLE TORQUE(Nm)	3.00
MOMENT PERMISSIBLE TORQUE(Nm)	5.00
SHAFT MAXIMUM AXIAL LOAD(N)	50.00
SHAFT MAXIMUM RADIAL LOAD(N)	100.00
WEIGHT(Kg)[lb]	0.60[1.32]
TEMPERATURE RISE:MAX.80°C（MOTOR STANDSTILL;FOR 2PHASE ENERGIZED）	
AMBIENT TEMPERATURE -10°C~50°C[14°F~122°F]	
INSULATION CLASS B 130°C[266°F]	

TYPE OF CONNECTION (EXTERN)		MOTOR	
PIN NO	BIPOLAR	LEADS	WINDING
1	A —	BLK	A
2	A\ —	GRN	A\
3	B —	RED	B
4	B\ —	BLU	B\

FULL STEP 2 PHASE-Ex. ,
WHEN FACING MOUNTING END (X)

STEP	A	B	A\	B\
1	+	+	-	-
2	-	+	+	-
3	-	-	+	+
4	+	-	-	+

CW ↓ CCW ↑

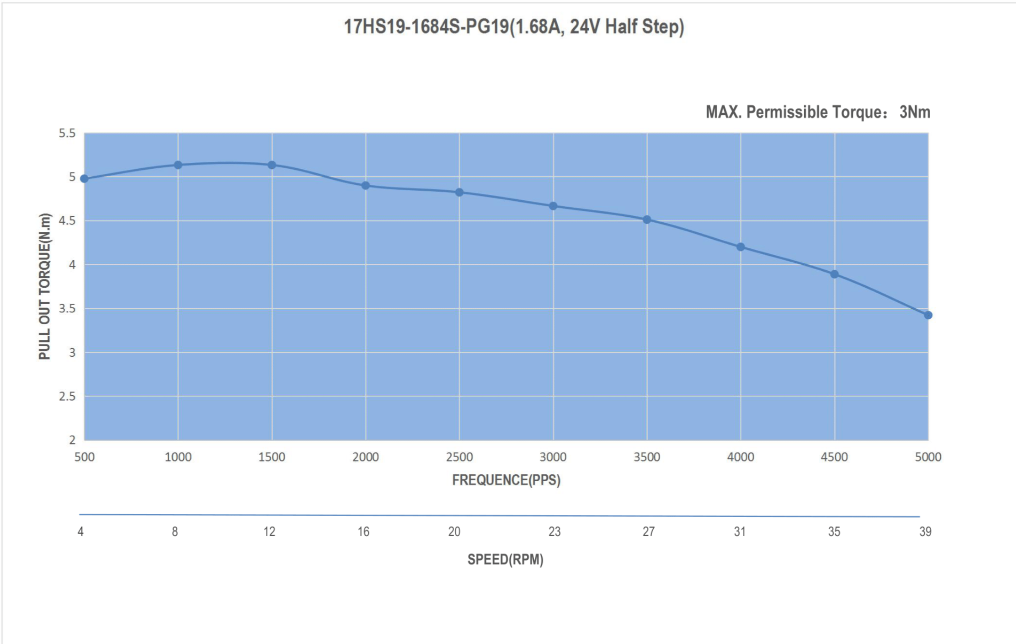


STEPPERONLINE®

	APVD		8.18.2018
	CHKD		
1:1	DRN		
SCALE	SIGNATURE		DATE

STEPPER MOTOR

17HS19-1684S-PG19



17HS19-1684S-PG19(1.68A, 24V Half Step)
MAX. Permissible Torque：3Nm
PULL OUT TORQUE(N.m)
5.5
5
4.5
4
3.5
3
2.5
2
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
FREQUENCE(PPS)
4
8
12
16
20
23
27
31
35
39
SPEED(RPM)

A.2 TB6600 Stepper Motor Driver

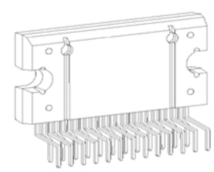
TOSHIBA BiCD Integrated Circuit Silicon Monolithic

TB6600HG

PWM Chopper-Type bipolar
Stepping Motor Driver IC

The TB6600HG is a PWM chopper-type single-chip bipolar sinusoidal micro-step stepping motor driver.
Forward and reverse rotation control is available with 2-phase, 1-2-phase, W1-2-phase, 2W1-2-phase, and 4W1-2-phase excitation modes.
2-phase bipolar-type stepping motor can be driven by only clock signal with low vibration and high efficiency.

TB6600HG



HZIP25-P-1.00F

Weight:
HZIP25-P-1.00F: 7.7g (typ.)

Features

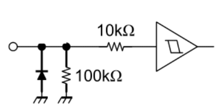
- Single-chip bipolar sinusoidal micro-step stepping motor driver
- Ron (upper + lower) = 0.4 Ω (typ.)
- Forward and reverse rotation control available
- Selectable phase drive (1/1, 1/2, 1/4, 1/8, and 1/16 step)
- Output withstand voltage: Vcc = 50 V
- Output current: I_{OUT} = 5.0 A (absolute maximum ratings, peak)
  I_{OUT} = 4.5 A (operating range, maximal value)
- Packages: HZIP25-P-1.00F
- Built-in input pull-down resistance: 100 kΩ (typ.), (only TQ terminal: 70 kΩ (typ.))
- Output monitor pins (ALERT): Maximum of I_{ALERT} = 1 mA
- Output monitor pins (MO): Maximum of I_{MO} = 1 mA
- Equipped with reset and enable pins
- Stand by function
- Single power supply
- Built-in thermal shutdown (TSD) circuit
- Built-in under voltage lock out (UVLO) circuit
- Built-in over-current detection (ISD) circuit

Pin Functions

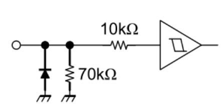
Pin No.	I/O	Symbol	Functional Description	Remark
1	Output	ALERT	TSD / ISD monitor pin	Pull-up by external resistance
2	—	SGND	Signal ground	
3	Input	TQ	Torque (output current) setting input pin	
4	Input	Latch/Auto	Select a return type for TSD.	L: Latch, H: Automatic return
5	Input	Vref	Voltage input for 100% current level	
6	Input	Vcc	Power supply	
7	Input	M1	Excitation mode setting input pin	
8	Input	M2	Excitation mode setting input pin	
9	Input	M3	Excitation mode setting input pin	
10	Output	OUT2B	B channel output 2	
11	—	N_{FB}	B channel output current detection pin	
12	Output	OUT1B	B channel output 1	
13	—	PGNDB	Power ground	
14	Output	OUT2A	A channel output 2	
15	—	N_{FA}	A channel output current detection pin	
16	Output	OUT1A	A channel output 1	
17	—	PGNDA	Power ground	
18	Input	ENABLE	Enable signal input pin	H: Enable, L: All outputs off
19	Input	RESET	Reset signal input pin	L: Initial mode
20	Input	Vcc	Power supply	
21	Input	CLK	CLK pulse input pin	
22	Input	CW/CCW	Forward/reverse control pin	L: CW, H:CCW
23	—	OSC	Resistor connection pin for internal oscillation setting	
24	Output	V_{reg}	Control side connection pin for power capacitor	Connecting capacitor to SGND
25	Output	MO	Electrical angle monitor pin	Pull-up by external resistance

<Terminal circuits>

Input pins
(M1, M2, M3,CLK, CW/CCW,
ENABLE, RESET, Latch/Auto)

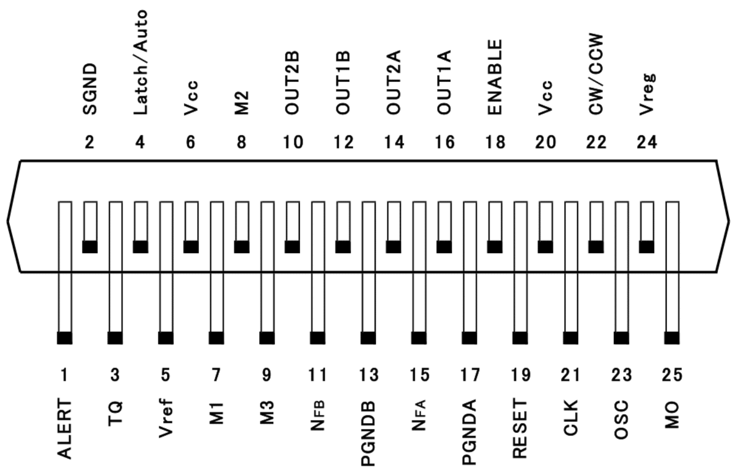


Input pins
(TQ)

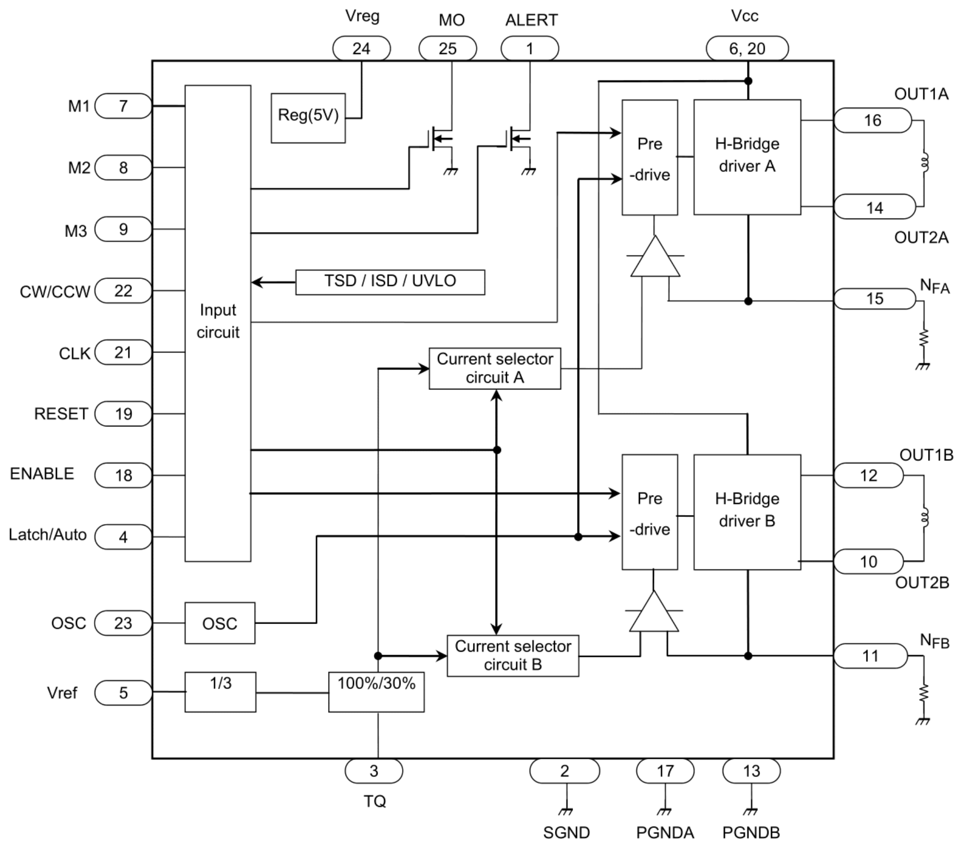


Pin Assignment

（Top View）



Block Diagram



Setting of Vref

Input	Voltage ratio
TQ	
L	30%
H	100%

Description of Functions

1. Excitation Settings

The excitation mode can be selected from the following eight modes using the M1, M2 and M3 inputs. New excitation mode starts from the initial mode when M1, M2, or M3 inputs are shifted during motor operation. In this case, output current waveform may not continue.

Input			Mode (Excitation)
M1	M2	M3	
L	L	L	Standby mode (Operation of the internal circuit is almost turned off.)
L	L	H	1/1 (2-phase excitation, full-step)
L	H	L	1/2A type (1-2 phase excitation A type) (0%, 71%, 100%)
L	H	H	1/2B type (1-2 phase excitation B type) (0%, 100%)
H	L	L	1/4 (W1-2 phase excitation)
H	L	H	1/8 (2W1-2 phase excitation)
H	H	L	1/16 (4W1-2 phase excitation)
H	H	H	Standby mode (Operation of the internal circuit is almost turned off.)

Note: To change the exciting mode by changing M1, M2, and M3, make sure not to set M1 = M2 = M3 = L or M1 = M2 = M3 = H.

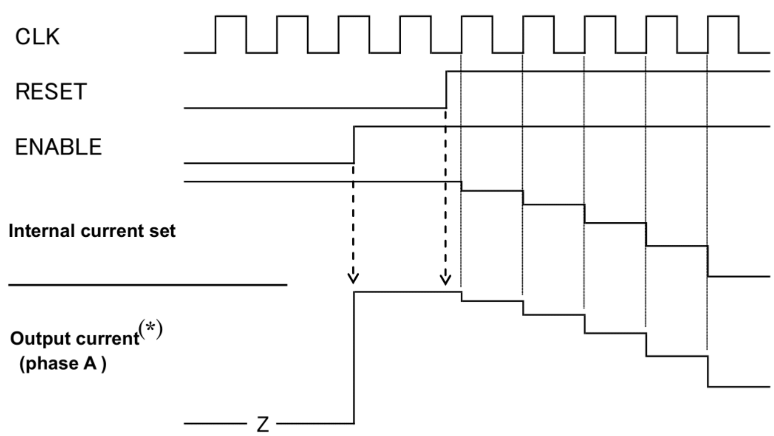
Standby mode

The operation mode moves to the standby mode under the condition M1 = M2 = M3 = L or M1 = M2 = M3 = H.
The power consumption is minimized by turning off all the operations except protecting operation.
In standby mode, output terminal MO is HZ.
Standby mode is released by changing the state of M1=M2=M3=L and M1=M2=M3=H to other state.
Input signal is not accepted for about 200 μs after releasing the standby mode.

2. Function

(1)To turn on the output, configure the ENABLE pin high. To turn off the output, configure the ENABLE pin low.
(2) The output changes to the Initial mode shown in the table below when the ENABLE signal goes High level and the RESET signal goes Low level. (In this mode, the status of the CLK and CW/CCW pins are irrelevant.)
(3) As shown in the below figure of Example 1, when the ENABLE signal goes Low level, it sets an OFF on the output. In this mode, the output changes to the initial mode when the RESET signal goes Low level. Under this condition, the initial mode is output by setting the ENABLE signal High level. And the motor operates from the initial mode by setting the RESET signal High level.

(Example 1)



(*: Output current starts rising at the timing of PWM frequency just after ENABLE pin outputs high.)

Input				Output mode
CLK	CW/CCW	RESET	ENABLE	
↑	L	H	H	CW
↑	H	H	H	CCW
X	X	L	H	Initial mode
X	X	X	L	Z

Command of the standby has a higher priority than ENABLE. Standby mode can be turned on and off regardless of the state of ENABLE.
X: Don’t Care

3. Initial Mode

When RESET is used, the phase currents are as follows.

Excitation Mode	Phase A Current	Phase B Current
1/1 (2-phase excitation, full-step)	100%	-100%
1/2A type (1-2 phase excitation A type) (0%, 71%, 100%)	100%	0%
1/2B type (1-2 phase excitation B type) (0%, 100%)	100%	0%
1/4 (W1-2 phase excitation)	100%	0%
1/8 (2W1-2 phase excitation)	100%	0%
1/16 (4W1-2 phase excitation)	100%	0%

current direction is defined as follows.
OUT1A → OUT2A: Forward direction
OUT1B → OUT2B: Forward direction

4. 100% current settings (Current value)

100% current value is determined by Vref inputted from external part and the external resistance for detecting output current. Vref is doubled 1/3 inside IC.

I_o (100%) = (1/3 × Vref) ÷ R_{NF}

The average current is lower than the calculated value because this IC has the method of peak current detection.

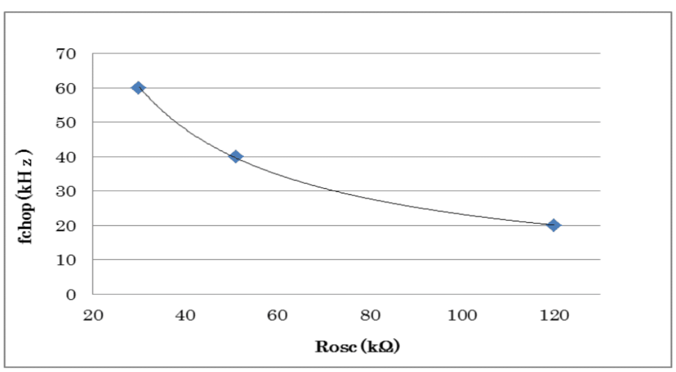
Pleas use the IC under the conditions as follows;

$0.11\Omega \leq R_{NF} \leq 0.5\Omega$, $0.3V \leq Vref \leq 1.95V$

5. OSC

Triangle wave is generated internally by CR oscillation by connecting external resistor to OSC terminal. Rosc should be from 30kΩ to 120kΩ. The relation of Rosc and fchop is shown in below table and figure. The values of fchop of the below table are design guarantee values. They are not tested for pre-shipment.

Rosc(kΩ)	fchop(kHz)		
	Min	Typ.	Max
30	-	60	-
51	-	40	-
120	-	20	-



6. Decay Mode

It takes approximately five OSCM cycles for charging-discharging a current in PWM mode. The 40% fast decay mode is created by inducing decay during the last two cycles in Fast Decay mode.
The ratio 40% of the fast decay mode is always fixed.
The relation between the master clock frequency (fMCLK), the OSCM frequency (fOSCM) and the PWM frequency (fchop) is shown as follows:

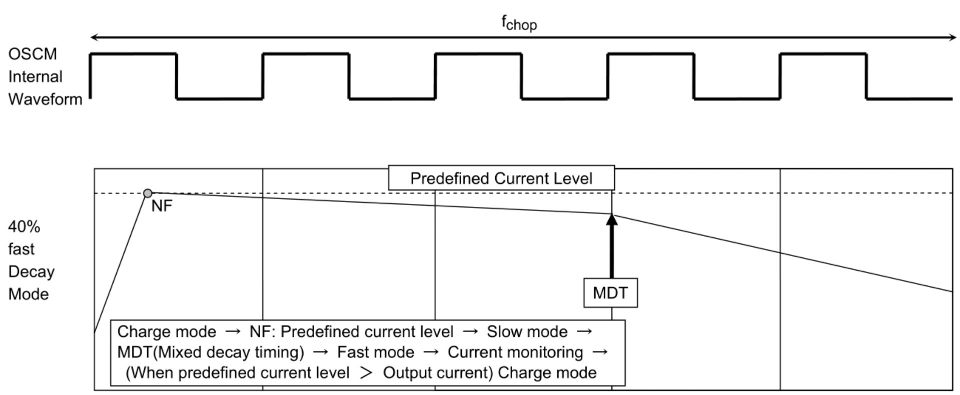
fOSCM = 1/20 ×fMCLK
fchop = 1/100 ×fMCLK

When Rosc=51kΩ, the master clock=4MHz, OSCM=200kHz, the frequency of PWM(fchop)=40kHz.

6-1. Current Waveform and Mixed Decay Mode settings

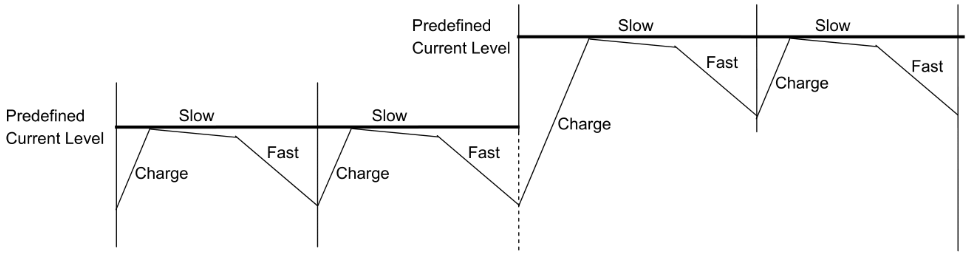
The period of PWM operation is equal to five periods of OSCM.
The ratio 40% of the fast decay mode is always fixed.
The "NF" refers to the point at which the output current reaches its predefined current level.

MDT means the point of MDT (MIXED DECAY TIMMING) in the below diagram.

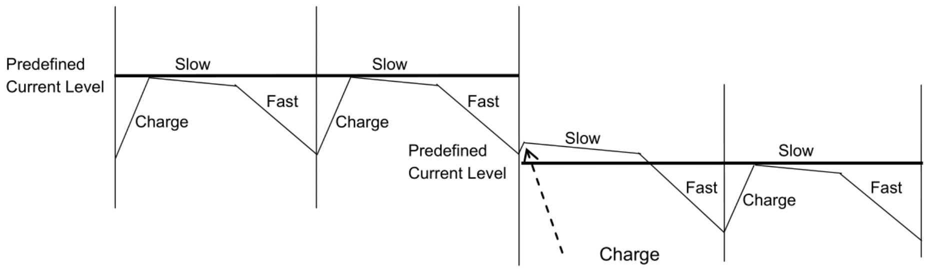


6-2. Effect of Decay Mode

- Increasing the current (sine wave)

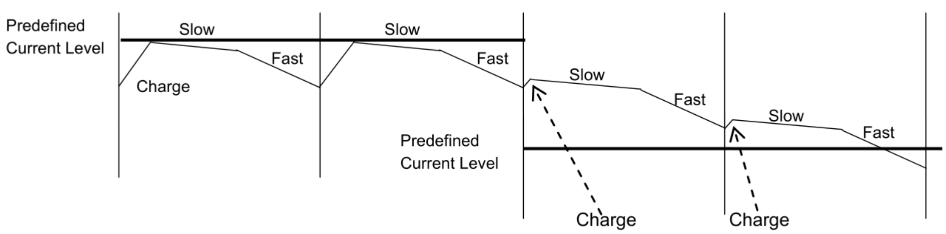


- Decreasing the current (In case the current is decreased to the predefined value in a short time because it decays quickly.)



Even if the output current rises above the predefined current at the RNF point, the current control mode is briefly switched to Charge mode for current sensing.

- Decreasing the current (In case it takes a long time to decrease the current to the predefined value because the current decays slowly.)

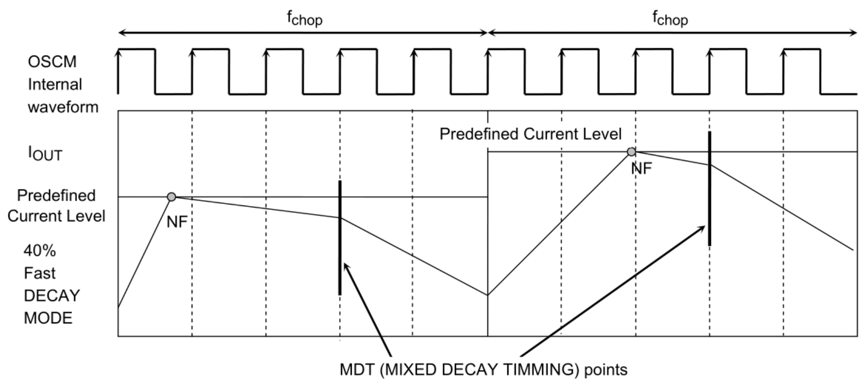


Even if the output current rises above the predefined current at the RNF point, the current control mode is briefly switched to Charge mode for current sensing.

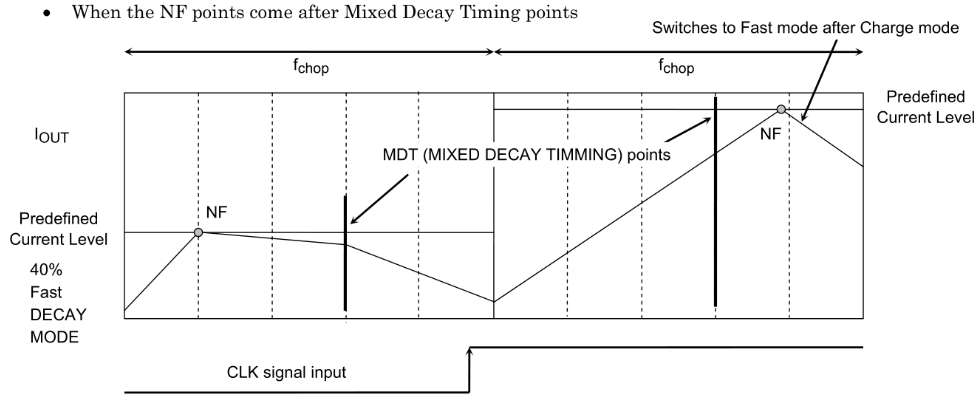
During Mixed Decay and Fast Decay modes, if the predefined current level is less than the output current at the RNF (current monitoring point), the Charge mode in the next chopping cycle will disappear (though the current control mode is briefly switched to Charge mode in actual operations for current sensing) and the current is controlled in Slow and Fast Decay modes (mode switching from Slow Decay mode to Fast Decay mode at the MDT point).

Note: The above figures are rough illustration of the output current. In actual current waveforms, transient response curves can be observed.

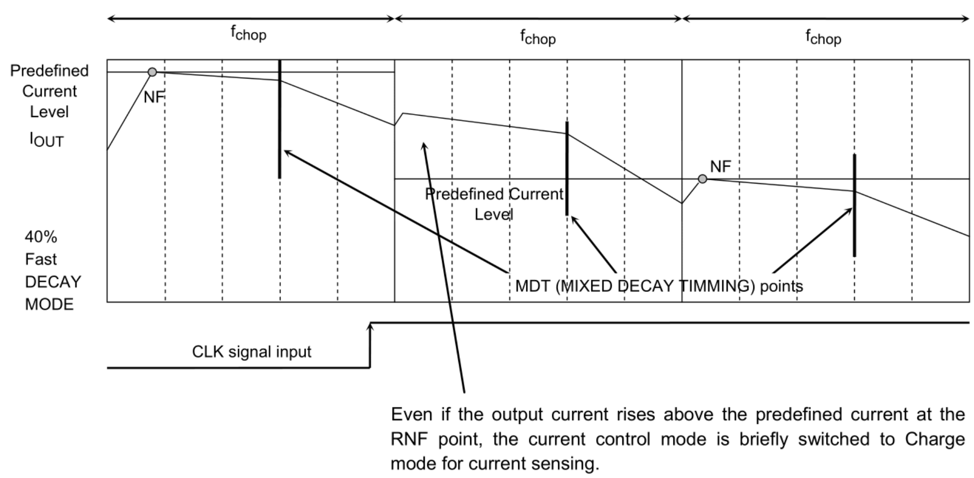
6-3. Current Waveforms in Mixed Decay Mode



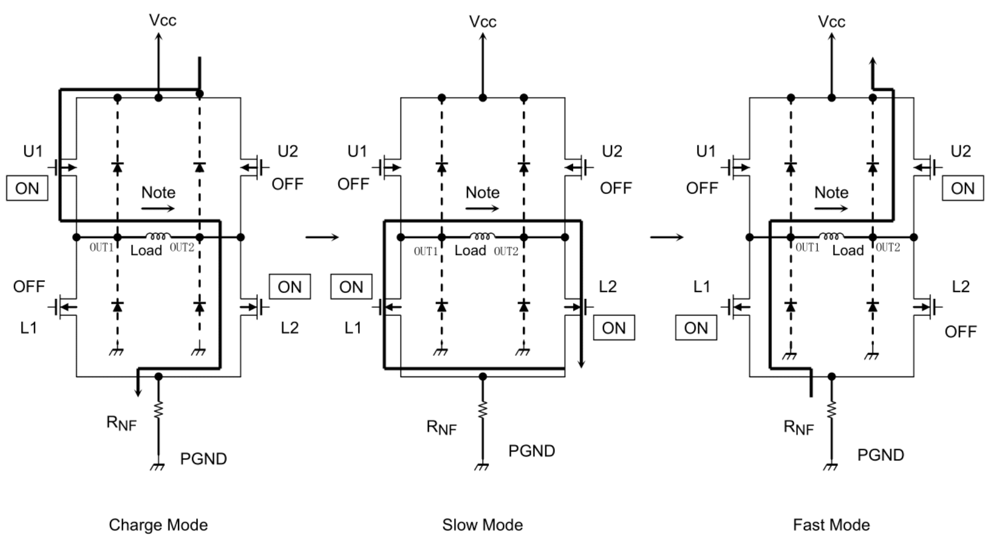
- When the NF points come after Mixed Decay Timing points



- When the output current value > predefined current level in Mixed Decay mode



Output Stage Transistor Operation Mode



Output Stage Transistor Operation Functions

CLK	U1	U2	L1	L2
CHARGE	ON	OFF	OFF	ON
SLOW	OFF	OFF	ON	ON
FAST	OFF	ON	ON	OFF

Note: The above chart shows an example of when the current flows as indicated by the arrows in the above figures. If the current flows in the opposite direction, refer to the following chart:

CLK	U1	U2	L1	L2
CHARGE	OFF	ON	ON	OFF
SLOW	OFF	OFF	ON	ON
FAST	ON	OFF	OFF	ON

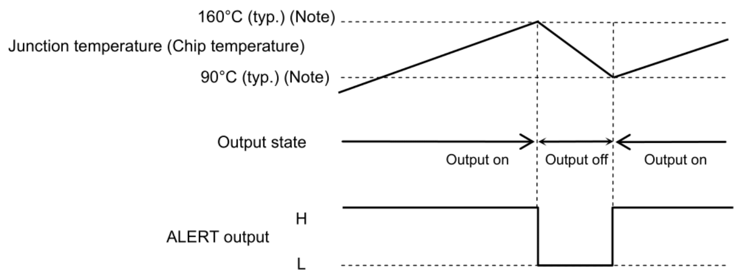
Upon transitions of above-mentioned functions, a dead time of about 300 ns (Design guarantee value) is inserted respectively.

Thermal Shut-Down circuit (TSD)

(1) Automatic return

TSD = 160°C (typ.) (Note)

TSDhys = 70°C (typ.) (Note)



Automatic return has a temperature hysteresis shown in the above figure.

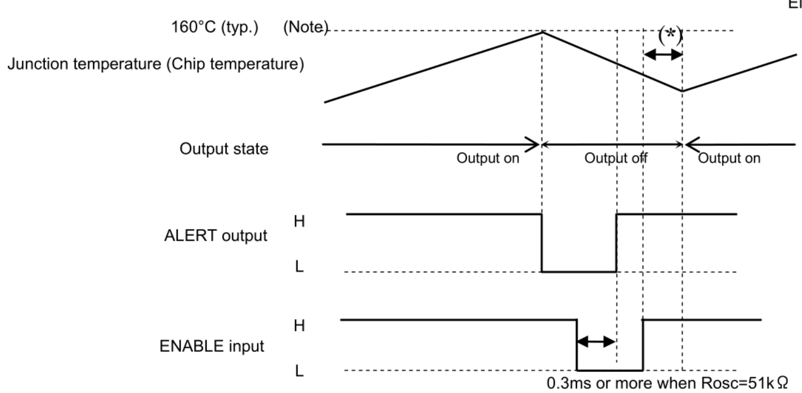
In case of automatic return, the return timing is adjusted at charge start of fchop after the temperature falls to the return temperature (90°C (typ.) in the above figure).

The return period after the temperature falls corresponds to one cycle to two cycles of fchop.

(2) Latch type

TSD = 160°C (typ.) (Note)

(*)Output current starts rising at the timing of PWM frequency just after ENABLE pin outputs high.



The operation returns by programming the ENABLE as H → L → H shown in above figure or turning on power supply and turning on UVLO function. In this time, term of L level of ENABLE should be 0.3ms or more.

To recover the operation, the junction temperature (the chip temperature) should be 90°C or less when ENABLE input is switched from L to H level. Otherwise, the operation does not recover.

Note: Pre-shipment testing is not performed.

・State of internal IC when TSD circuit operates.

The states of the internal IC and outputs, while the shutdown circuit is operating, correspond to the state when ENABLE is L.

The state after automatic return corresponds to the state when ENABLE is H. Please configure the Reset L to rotate the motor from the initial state.

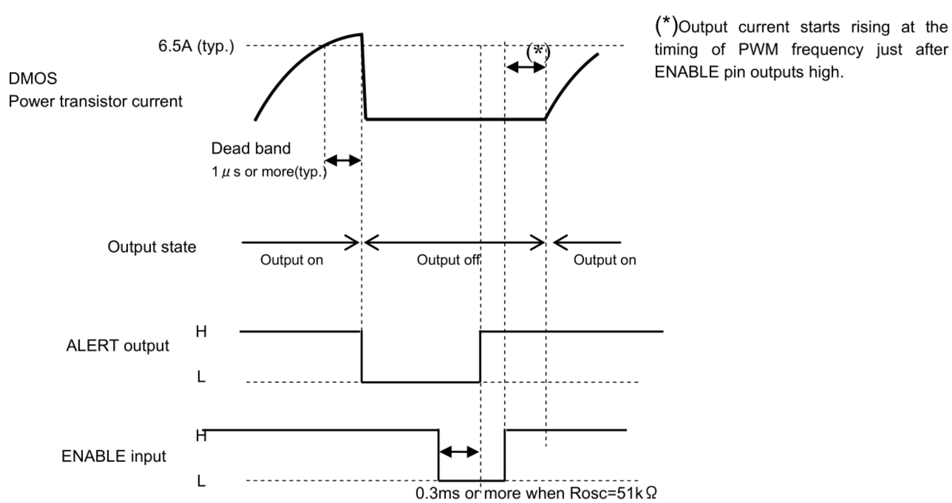
Latch/Auto is an input pin for determining the return method of TSD.
If Latch/Auto pin outputs low, TSD function returns by either of turning on power supply again or programming the ENABLE as H → L → H.
If Latch/Auto pin outputs high, it returns automatically.
In standby mode, TSD function returns automatically regardless of the state of the Latch/Auto pin.
When power supply voltage Vcc is less than 8V, TSD function cannot operate regardless of the state of the Latch/Auto pin.

ISD (Over current detection)

Current that flows through output power MOSFETs are monitored individually. If over-current is detected in at least one of the eight output power MOSFETs, all output power MOSFETs are turned off then this status is kept until ENABLE signal is input. In this time, term of L level of ENABLE should be 0.3ms or more.

Masking term of 1μs or more (typ. when Rosc=51kΩ) (Note) should be provided in order to protect detection error by noise. ISD does not work during the masking term.

Over current detection value ISD=6.5 A (Note)



The operation returns by programming the ENABLE as H → L → H shown in above figure or turning on power supply and turning on UVLO function.

Note: Pre-shipment testing is not performed.

·State of internal IC when ISD circuit operates.

The states of the internal IC and outputs, while the over current detection circuit is operating, correspond to the state when ENABLE is L.

The state after automatic return corresponds to the state when ENABLE is H. Please configure the Reset L to rotate the motor from the initial state.

Return method of ISD

ISD function returns by either of turning on power supply again or programming the ENABLE as H → L → H regardless of the state of the Latch/Auto pin.

In standby mode, ISD function cannot operate.

When power supply voltage Vcc is less than 8V, ISD function cannot operate.

Under Voltage Lock Out (UVLO) circuit

Outputs are shutoff by operating at 5.5 V (Typ.) of Vcc or less.
It has a hysteresis of 0.5 V (Typ.) and returns to output when Vcc reaches 6.0 V (Typ.). The following values are design guarantee values.

・State of internal IC when UVLO circuit operates.

The states of the internal IC and outputs correspond to the state in the ENABLE mode and the initial mode at the same time.
After a return, it can start from the initial mode.

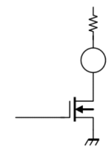
When Vcc falls to around 5.5 V and UVLO operates, output turns off.
It recovers automatically from the initial mode when both Vcc rise to around 6.0 V or more. The following values are design guarantee values.

ALERT output

ALERT terminal outputs low in detecting either TSD or ISD.

ALERT terminal is connected to power supply externally via pull-up resistance.
V_{ALERT} = 0.5 V (max) at 1 mA

TSD	ISD	ALERT
Under TSD detection	Under ISD detection	
Normal	Under ISD detection	Low
Under TSD detection	Normal	
Normal	Normal	Z

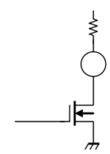


Applied voltage to pull-up resistance is up to 5.5 V. And conducted current is up to 1 mA.
It is recommended to gain 5 V by connecting the external pull-up resistance to Vreg pin.

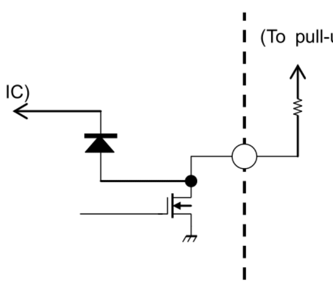
MO output

MO turns on at the predetermined state and output low.
MO terminal is connected to power supply externally via pull-up resistance.
V_{MO} = 0.5 V (max) at 1 mA

State	MO
Initial	Low
Not initial	Z



Applied voltage to pull-up resistance is up to 5.5 V. And conducted current is up to 1 mA.
It is recommended to gain 5 V by connecting the external pull-up resistance to Vreg pin.

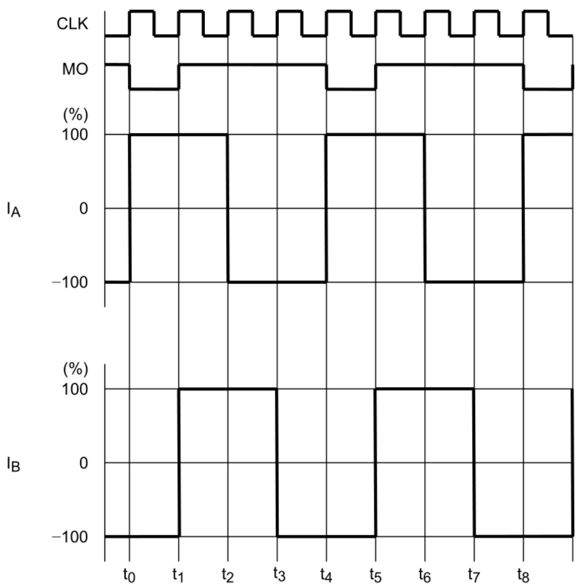


Voltage pull-up of MO and ALERT pins

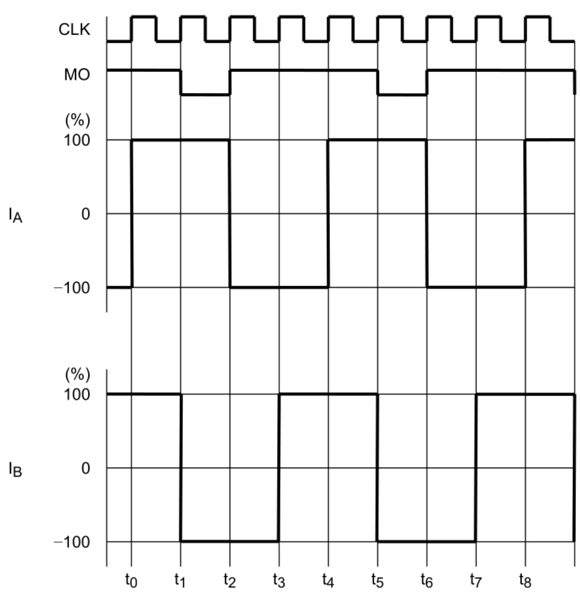
・It is recommended to pull-up voltage to Vreg pin.
・In case of pull-up to except 5 V (for instance, 3.3 V etc.), it is recommended to use other power supply (ex. 3.3 V) while Vcc output between the operation range. When Vcc decreases lower than the operation range and Vreg decreases from 5 V to 0 V under the condition that other power supply is used to pull-up voltage, the current continues to conduct from other power supply to the IC inside through the diode shown in the figure. Though this phenomenon does not cause destruction and malfunction of the IC, please consider the set design not to continue such a state for a long time.
・As for the pull-up resistance for MO and ALERT pins, please select large resistance enough for the conducting current so as not to exceed the standard value of 1 mA.
Please use the resistance of 30 kΩ or more in case of applying 5 V, and 20 kΩ or more in case of applying 3.3 V.

Sequence and current level in each excitation mode

1/1-step Excitation Mode (M1: L, M2: L, M3: H, CW Mode)

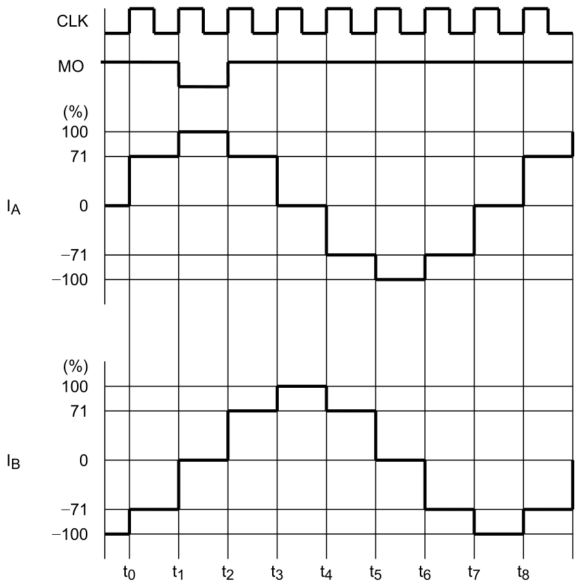


1/1-step Excitation Mode (M1: L, M2: L, M3: H, CCW Mode)

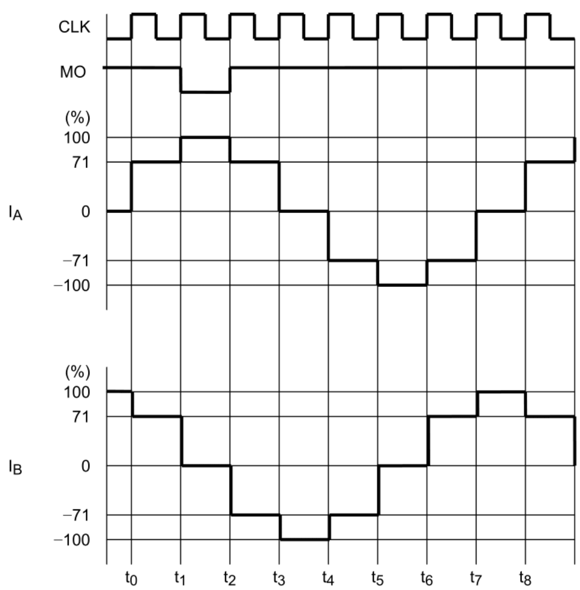


It operates from the initial state after the excitation mode is switched.

1/2-step Excitation Mode (A type) (M1: L, M2: H, M3: L, CW Mode)

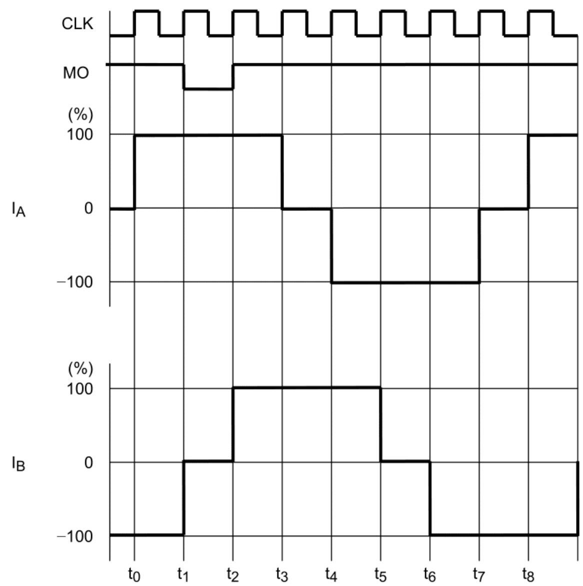


1/2-step Excitation Mode (A type) (M1: L, M2: H, M3: L, CCW Mode)

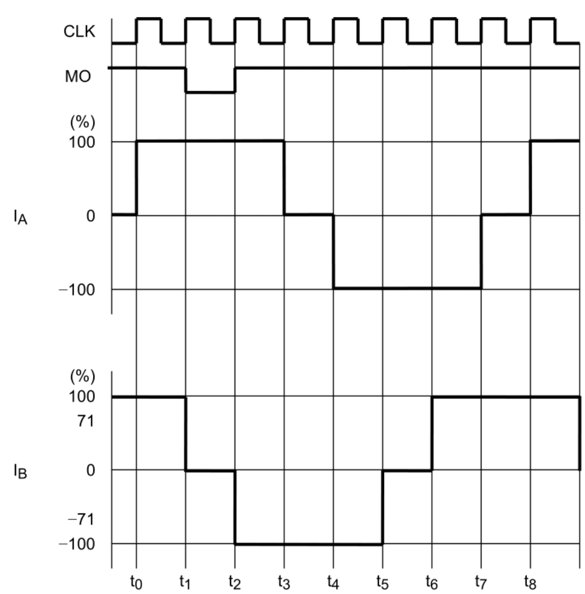


It operates from the initial state after the excitation mode is switched.

1/2-step Excitation Mode (B type) (M1: L, M2: H, M3: H, CW Mode)

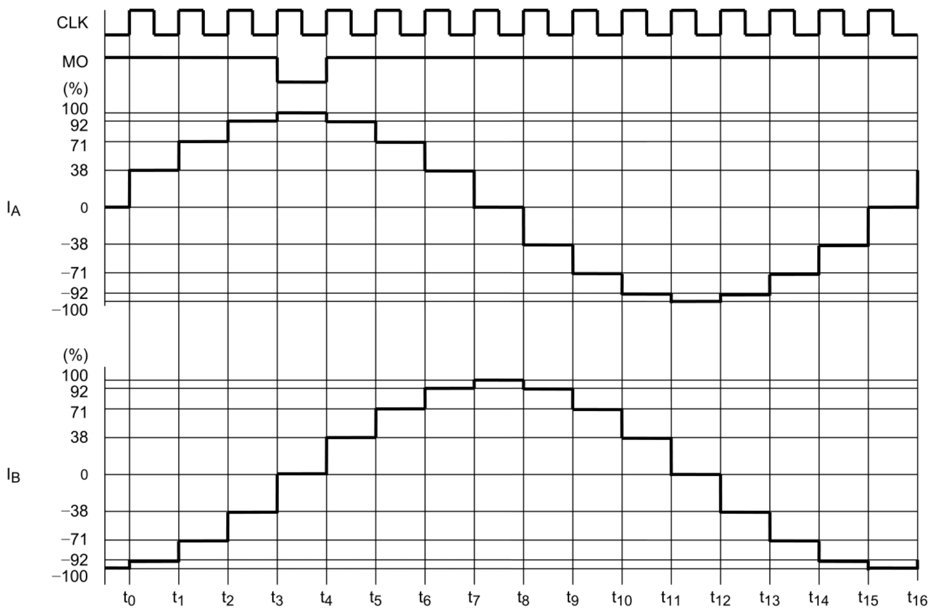


1/2-step Excitation Mode (B type) (M1: L, M2: H, M3: H, CCW Mode)

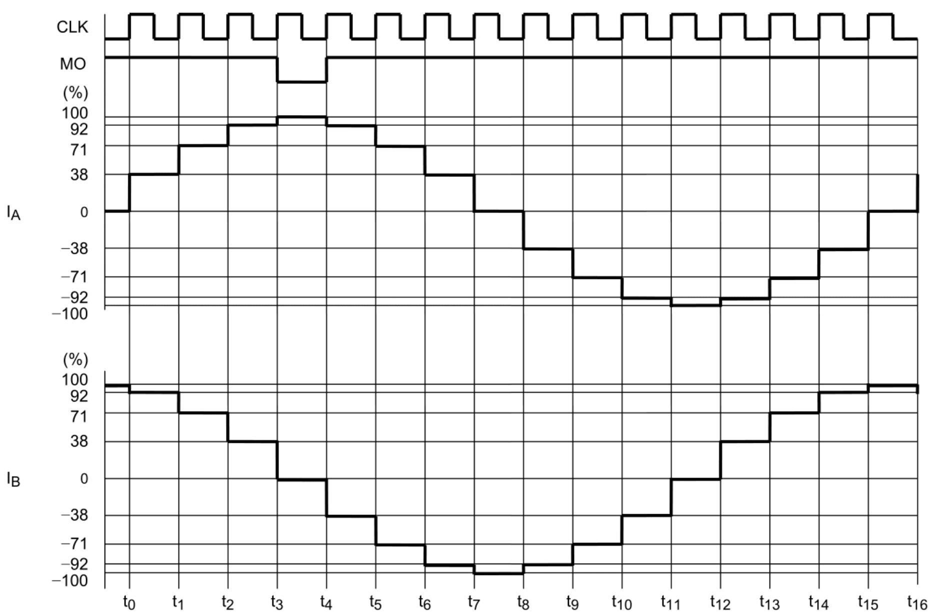


It operates from the initial state after the excitation mode is switched.

1/4-step Excitation Mode (M1: H, M2: L, M3: L, CW Mode)

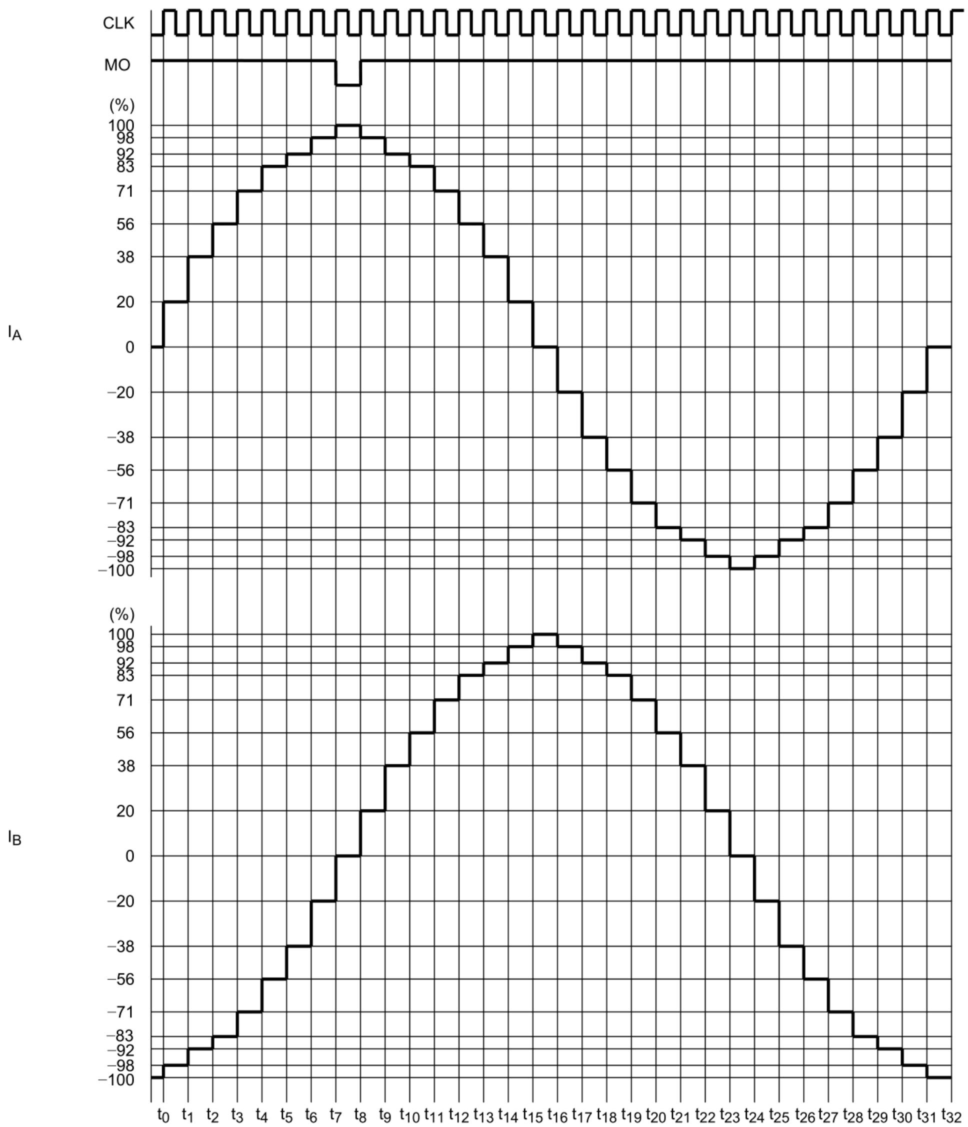


1/4-step Excitation Mode (M1: H, M2: L, M3: L, CCW Mode)



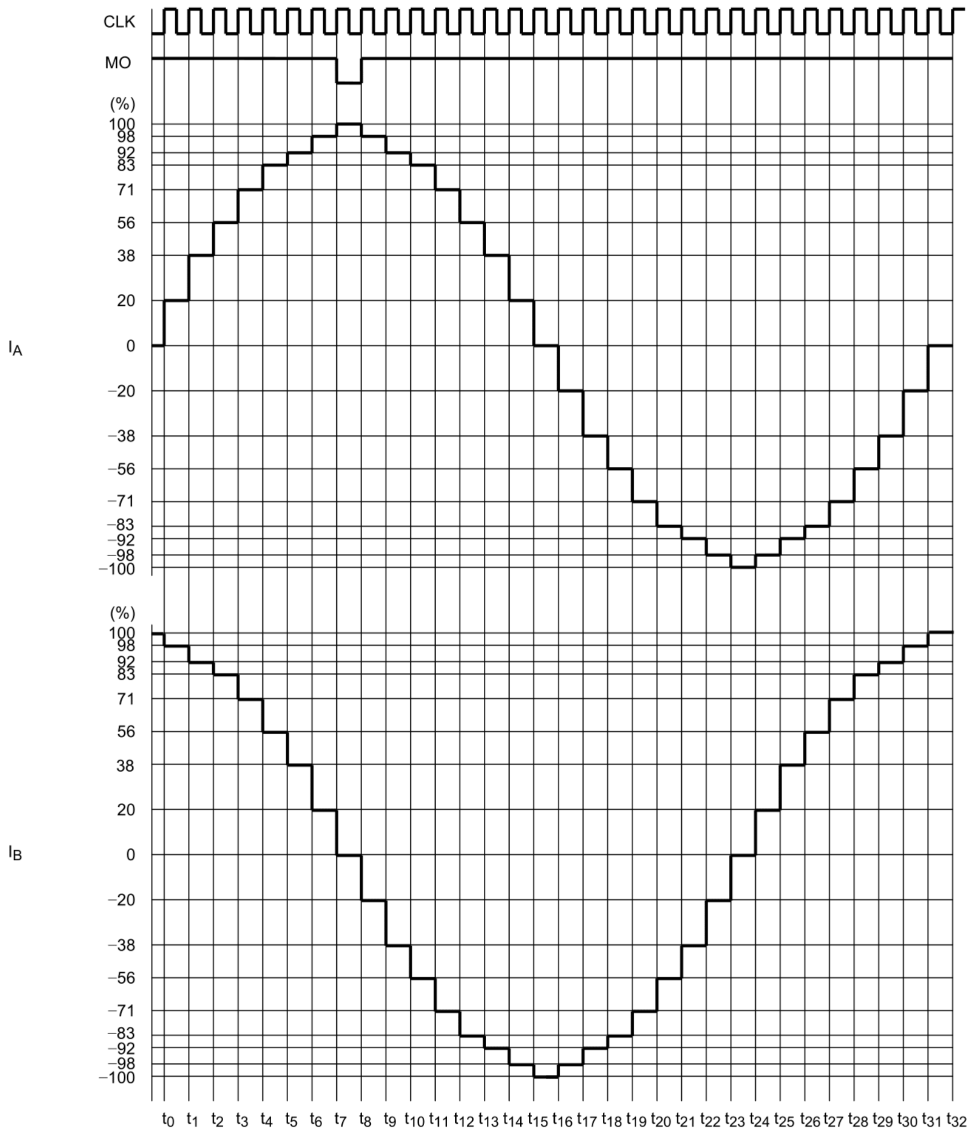
It operates from the initial state after the excitation mode is switched.

1/8-Step Excitation Mode (M1: H, M2: L, M3: H, CW Mode)



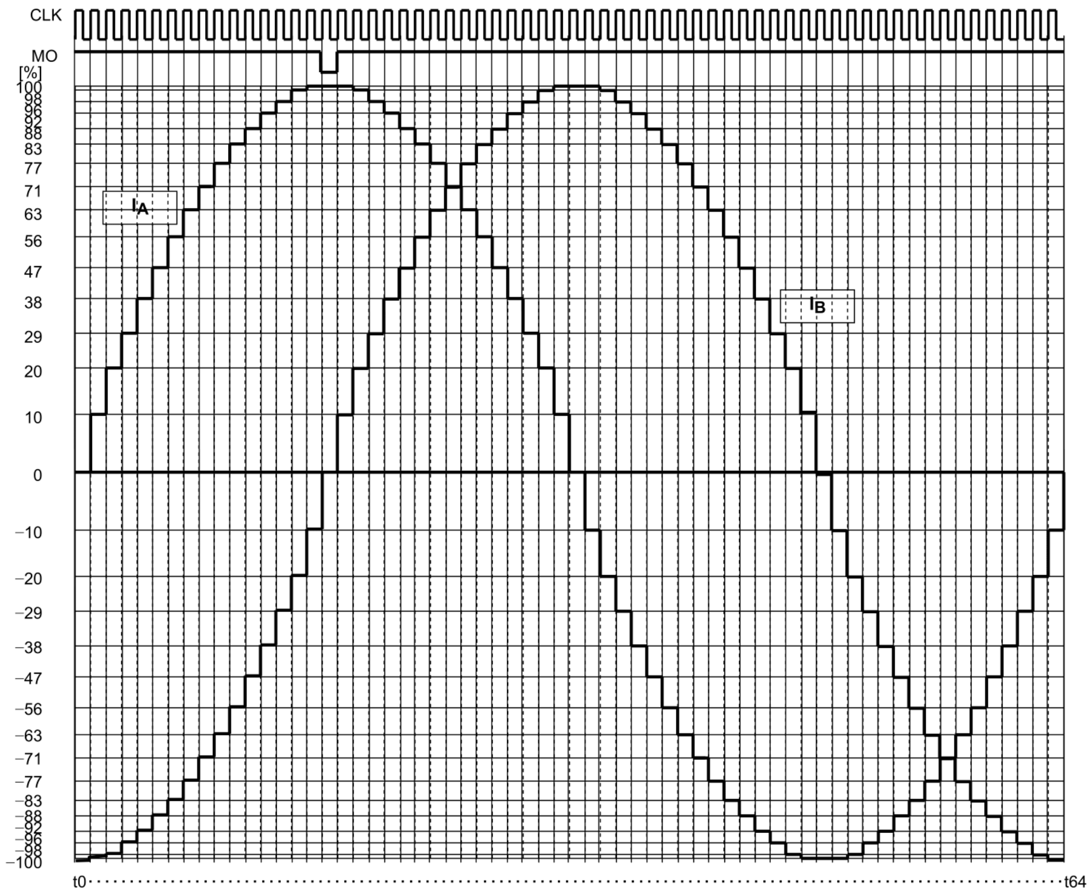
It operates from the initial state after the excitation mode is switched.

1/8-Step Excitation Mode (M1: H, M2: L, M3: H, CCW Mode)



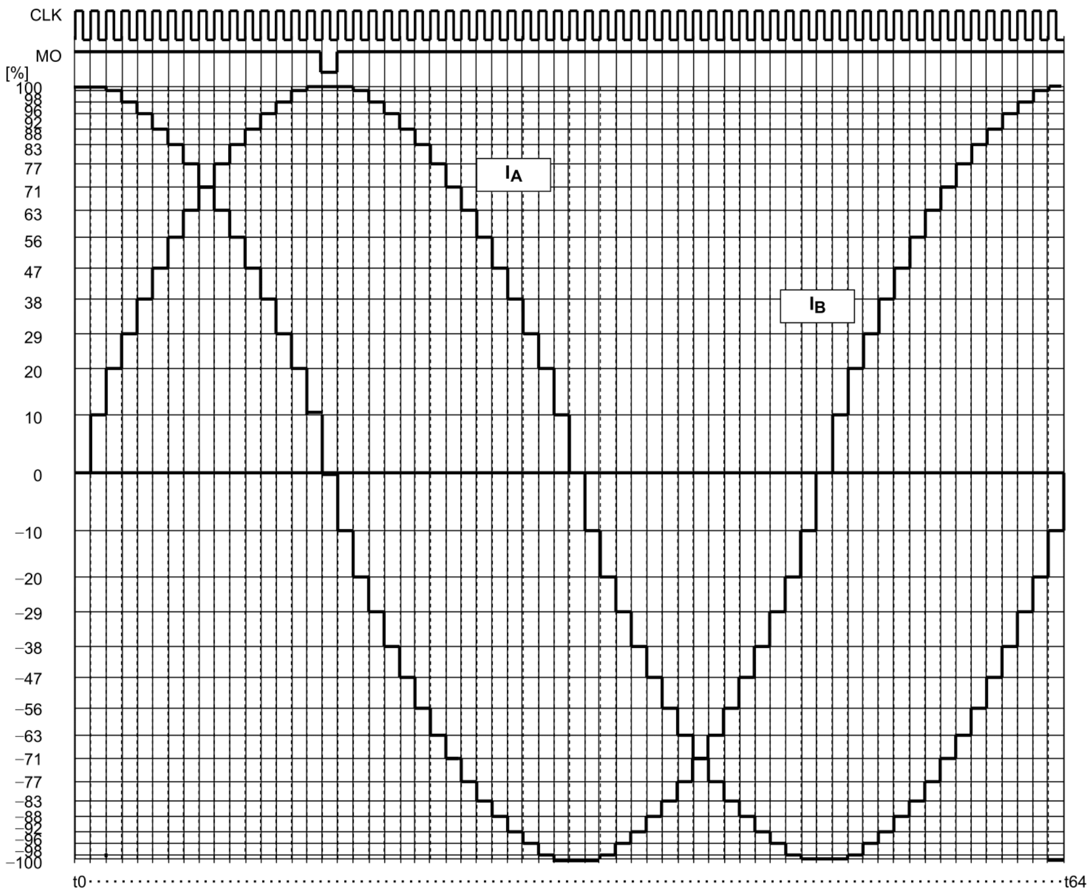
It operates from the initial state after the excitation mode is switched.

1/16-step Excitation Mode (M1: H, M2: H, M3: L, CW Mode)



It operates from the initial state after the excitation mode is switched.

1/16-step Excitation Mode (M1: H, M2: H, M3: L, CCW Mode)



It operates from the initial state after the excitation mode is switched.

Current level

2-phase, 1-2-phase, W1-2-phase, 2W1-2-phase, 4W1-2-phase excitation (unit: %)

Current level (1/16, 1/8, 1/4, 1/2, 1/1)

1/16, 1/8, 1/4, 1/2, 1/1	Min.	Typ.	Max.	Unit
θ16	---	100.0	---	%
θ15	95.5	99.5	100.0	
θ14	94.1	98.1	100.0	
θ13	91.7	95.7	99.7	
θ12	88.4	92.4	96.4	
θ11	84.2	88.2	92.2	
θ10	79.1	83.1	87.1	
θ9	73.3	77.3	81.3	
θ8	66.7	70.7	74.7	
θ7	59.4	63.4	67.4	
θ6	51.6	55.6	59.6	
θ5	43.1	47.1	51.1	
θ4	34.3	38.3	42.3	
θ3	25.0	29.0	33.0	
θ2	15.5	19.5	23.5	
θ1	5.8	9.8	13.8	
θ0	---	0.0	---	

Absolute Maximum Ratings (Ta = 25°C)

Characteristic	Symbol	Rating	Unit
Power supply voltage	Vcc	50	V
Output current (per one phase)	I_O (PEAK)	5.0	A
Drain current (ALERT, MO)	I (ALERT)	1	mA
	I (MO)		
Input voltage	V_{IN}	6	V
Power dissipation	P_D	3.2 (Note 1)	W
		40 (Note 2)	
Operating temperature	T_{opr}	-30 to 85	°C
Storage temperature	T_{stg}	-55 to 150	°C

Note 1: Ta = 25°C, No heatsink

Note 2: Ta = 25°C, with infinite heatsink.

The absolute maximum ratings of a semiconductor device are a set of ratings that must not be exceeded, even for a moment. Do not exceed any of these ratings.
Exceeding the rating (s) may cause the device breakdown, damage or deterioration, and may result injury by explosion or combustion.
Please use the IC within the specified operating ranges.

Operating Range (Ta = −30~85°C)

Characteristic	Symbol	Test Condition	Min	Typ.	Max	Unit
Power supply voltage	Vcc	—	8.0	—	42	V
Output current	I_{OUT}	—	—	—	4.5	A
Input voltage	V_{IN}	—	0	—	5.5	V
	V_{ref}	—	0.3	—	1.95	V
Clock frequency in logical part	f_{CLK}	—	—	—	200	kHz
Chopping frequency	f_{chop}	See page 7.	20	40	60	kHz

Note: Two Vcc terminals should be programmed the same voltage.
The maximum current of the operating range can not be necessarily conducted depending on various conditions because output current is limited by the power dissipation P_D.
Make sure to avoid using the IC in the condition that would cause the temperature to exceed Tj (avg.) =107°C.

The power supply voltage of 42 V and the output current of 4.5 A are the maximum values of operating range. Please design the circuit with enough derating within this range by considering the power supply variation, the external resistance, and the electrical characteristics of the IC. In case of exceeding the power supply voltage of 42 V and the output current of 4.5 A, the IC will not operate normally.

Electrical Characteristics (Ta = 25°C, Vcc = 24 V)

Characteristic		Symbol	Test Condition	Min	Typ.	Max	Unit
Input voltage	High	$V_{IN\ (H)}$	M1, M2, M3, CW/CCW, CLK, RESET, ENABLE, Latch/Auto, TQ	2.0	—	5.5	V
	Low	$V_{IN\ (L)}$		-0.2	—	0.8	
Input hysteresis voltage		V_H		—	400	—	mV
Input current		$I_{IN\ (H)}$	M1, M2, M3, CW/CCW, CLK, RESET, ENABLE, Latch/Auto V_{IN} = 5.0 V	—	50	75	μA
			TQ, V_{IN} = 5.0 V	—	70	105	
		$I_{IN\ (L)}$	M1, M2, M3, CW/CCW, CLK, RESET, ENABLE, Latch/Auto, TQ V_{IN} = 0 V	—	—	1	
Vcc supply current		Icc_1	Output open, RESET: H, ENABLE: H、 M1:L, M2:L, M3:H (1/1-step mode) CLK:L	—	4.2	7	mA
		Icc_2	Output open, RESET: L, ENABLE: L M1:L, M2:L, M3:H (1/1-step mode) CLK:L	—	3.6	7	
		Icc_3	Standby mode (M1:L, M2:L, M3:L)	—	1.8	4	
Vref input circuit	Current limit voltage	V_{NF}	Vref = 3.0 V(Note 1), TQ=H	0.9	1.0	1.1	V
	Input current	$I_{IN}(V_{ref})$	Vref = 3.0 V(Note 1)	—	—	1	μA
	Divider ratio	V_{ref}/V_{NF}	Maximum current: 100%, TQ=H	—	3	—	—
Minimum CLK pulse width		tw_{CLKH}	CLK	2.2	—	—	μs
		tw_{CLKL}					
Output residual voltage		V_{OL} MO	I_{OL} = 1 mA	—	—	0.5	V
		V_{OL} ALERT					
Internal constant voltage		Vreg	External capacitor = 0.1 μF (in standby mode)	4.5	5.0	5.5	V
Chopping frequency		f_{chop}	Rosc=51kΩ	28	40	52	kHz

Note 1: Though Vref of the test condition for pre-shipment is 3.0V, make sure to configure Vref within the operating range which is written in page 26 in driving the motor.

Electrical Characteristics (Ta = 25°C, Vcc = 24 V)

Characteristic		Symbol	Test Condition	Min	Typ.	Max	Unit
Output ON resistor		$Ron_U + Ron_L$	I_{OUT} = 4 A	—	0.4	0.6	Ω
Output transistor switching characteristics		t_r	V_{NF} = 0 V, Output: Open	—	50	—	ns
		t_f		—	500	—	
Output leakage current	Upper side	I_{LH}	Vcc = 50 V	—	—	5	μA
	Lower side	I_{LL}		—	—	5	

Timing Waveforms and Names

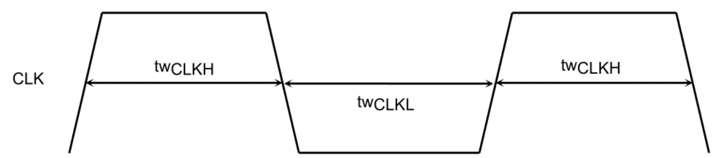


Figure 1 Timing Waveforms and Names

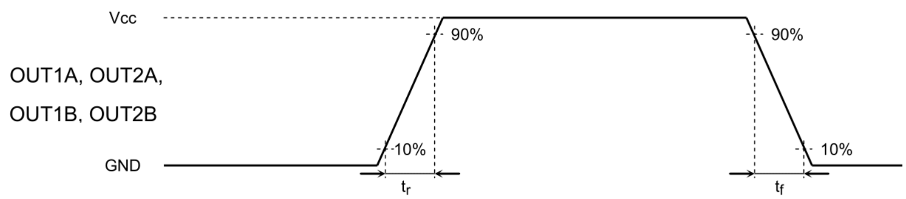
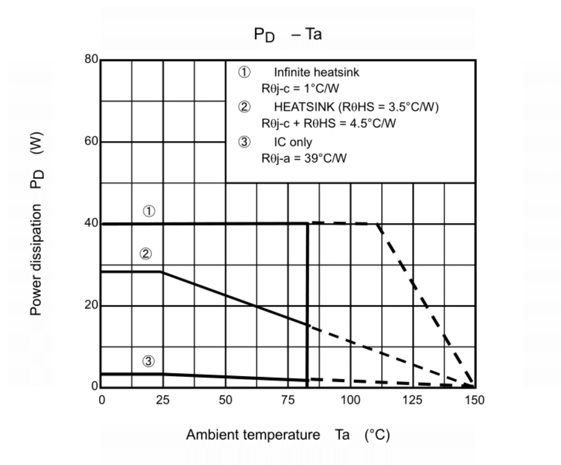


Figure 2 Timing Waveforms and Names

Power Dissipation

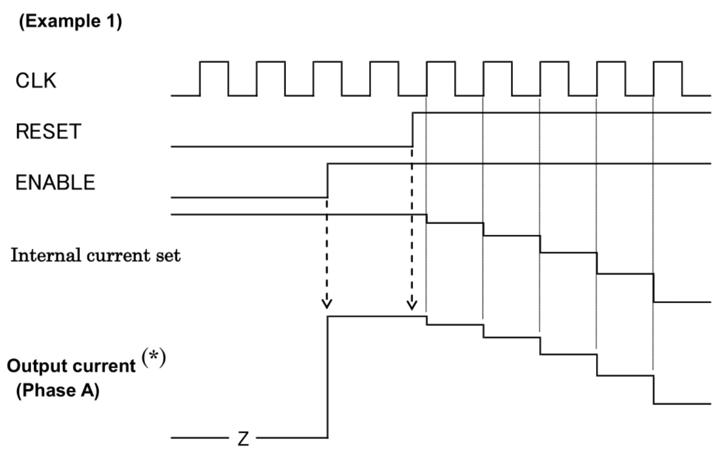
TB6600HG

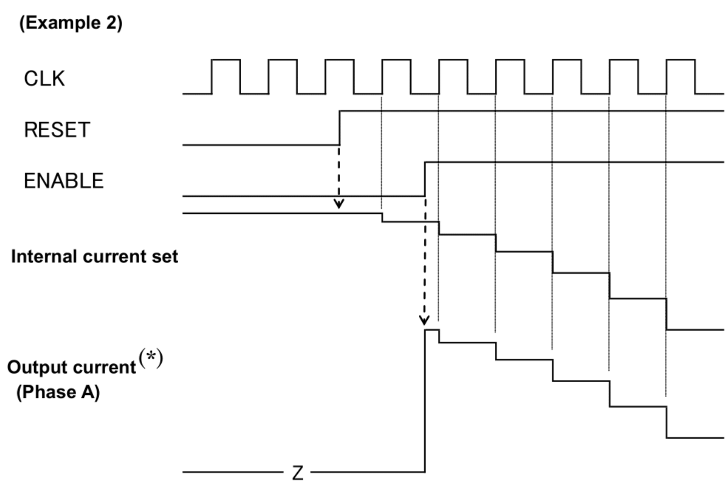


1. How to Turn on the Power

In applying Vcc or shutdown, ENABLE should be Low.
See Example 1(ENABLE = High → RESET = High) and Example 2(RESET = High → ENABLE = High) as follows. In example 1, a motor can start driving from the initial mode.
(1) CLK: Current step proceeds to the next mode with respect to every rising edge of CLK.
(2) ENABLE: It is in Hi-Z state in low level. It is output in high level.
RESET: It is in the initial mode (Phase A=100% and Phase B=0%) in low level.
①ENABLE=Low and RESET=Low: Hi-Z. Internal current setting is in initial mode.
②ENABLE=Low and RESET=High: Hi-Z. Internal current setting proceeds by internal counter.
③ENABLE=High and RESET=Low: Output in the initial mode (Phase A=100% and Phase B=0%).
④ENABLE=High and RESET=High: Output at the value which is determined by the internal counter.

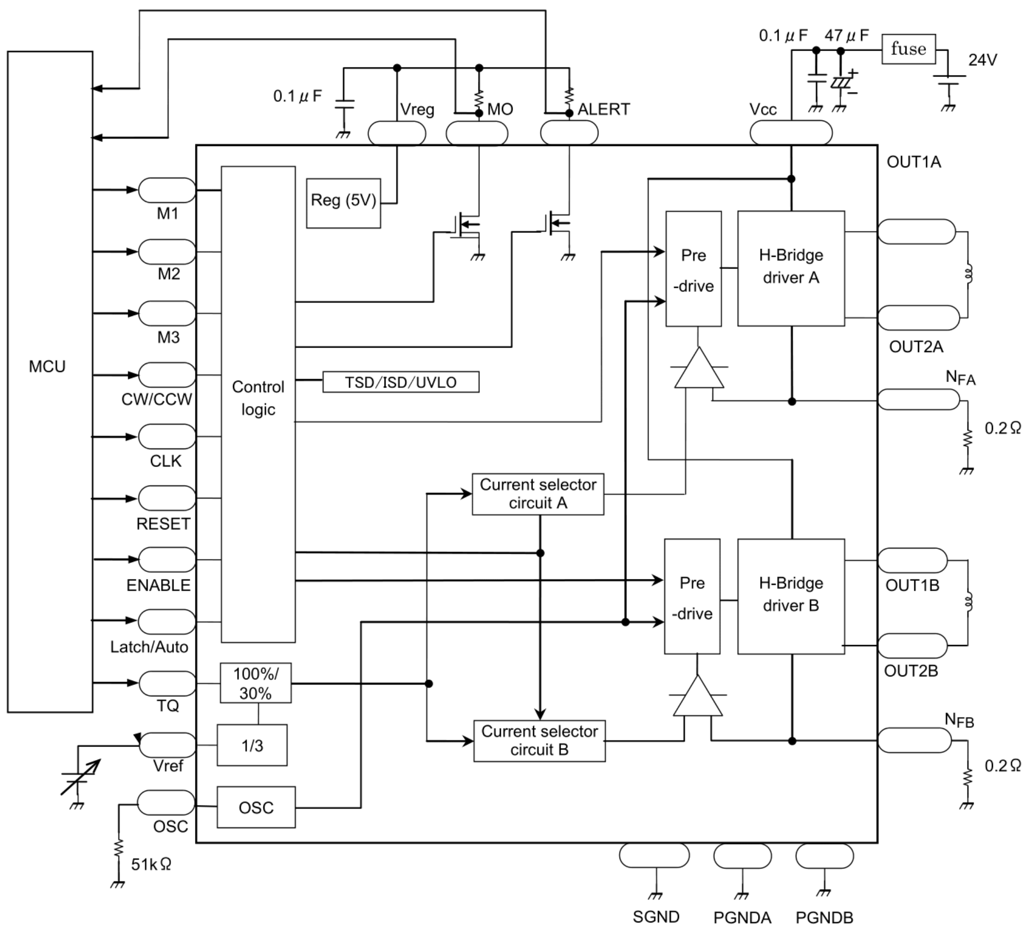
<Recommended control input sequence>





(* : Output current starts rising at the timing of PWM frequency just after ENABLE pin outputs high.)

Application Circuit



Note 1: Capacitors for the power supply lines should be connected as close to the IC as possible.

Note 2: Current detecting resistances (R_{NFA} and R_{NFB}) should be connected as close to the IC as possible.

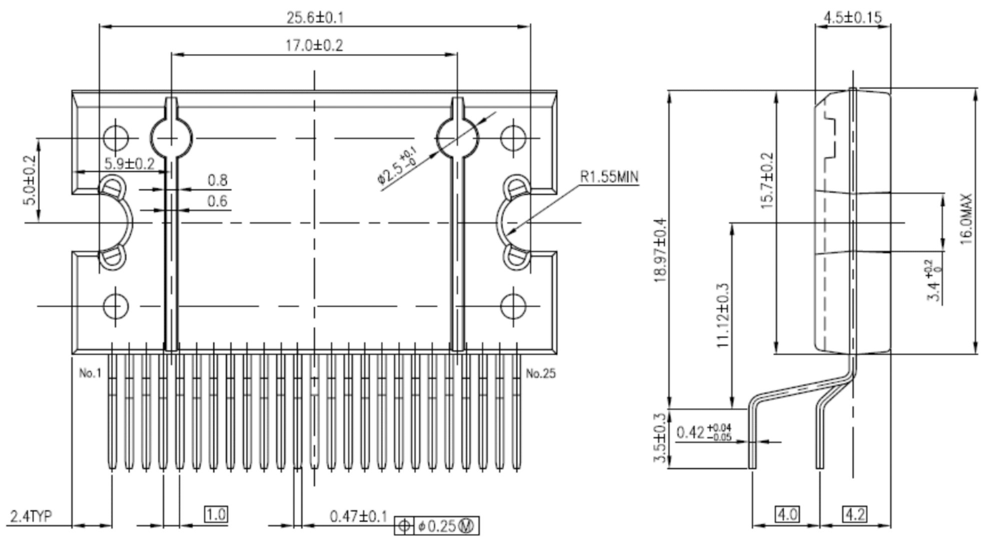
Note 3: Pay attention for wire layout of PCB not to allow GND line to have large common impedance.

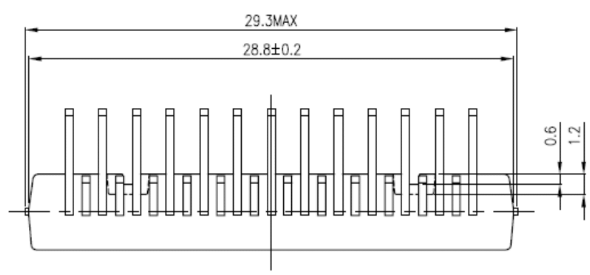
Note 4: External capacitor connecting to Vreg should be 0.1μF. Pay attention for the wire between this capacitor and Vreg terminal and the wire between this capacitor and SGND not to be influenced by noise.

Note 5: The IC may not operate normally when large common impedance is existed in GND line or the IC is easily influenced by noise. For example, if the IC operates continuously for a long time under the circumstance of large current and high voltage, the number of clock signals inputted to CLK terminal and that of steps of output current waveform may not proportional. And so, the IC may not operate normally. To avoid this malfunction, make sure to conduct Note.1 to Note.4 and evaluate the IC enough before using the IC.

Package Dimensions

Weight: 7.7 g (typ.)
Unit: mm





Notes on Contents

1. Block Diagrams

Some of the functional blocks, circuits, or constants in the block diagram may be omitted or simplified for explanatory purposes.

2. Equivalent Circuits

The equivalent circuit diagrams may be simplified or some parts of them may be omitted for explanatory purposes.

3. Timing Charts

Timing charts may be simplified for explanatory purposes.

4. Application Circuits

The application circuits shown in this document are provided for reference purposes only. Thorough evaluation is required, especially at the mass production design stage.

Toshiba does not grant any license to any industrial property rights by providing these examples of application circuits.

5. Test Circuits

Components in the test circuits are used only to obtain and confirm the device characteristics. These components and circuits are not guaranteed to prevent malfunction or failure from occurring in the application equipment.

IC Usage Considerations

Notes on handling of ICs

[1] The absolute maximum ratings of a semiconductor device are a set of ratings that must not be exceeded, even for a moment. Do not exceed any of these ratings.
Exceeding the rating(s) may cause the device breakdown, damage or deterioration, and may result injury by explosion or combustion.

[2] Use an appropriate power supply fuse to ensure that a large current does not continuously flow in case of over current and/or IC failure. The IC will fully break down when used under conditions that exceed its absolute maximum ratings, when the wiring is routed improperly or when an abnormal pulse noise occurs from the wiring or load, causing a large current to continuously flow and the breakdown can lead smoke or ignition. To minimize the effects of the flow of a large current in case of breakdown, appropriate settings, such as fuse capacity, fusing time and insertion circuit location, are required.

[3] If your design includes an inductive load such as a motor coil, incorporate a protection circuit into the design to prevent device malfunction or breakdown caused by the current resulting from the inrush current at power ON or the negative current resulting from the back electromotive force at power OFF. IC breakdown may cause injury, smoke or ignition.
Use a stable power supply with ICs with built-in protection functions. If the power supply is unstable, the protection function may not operate, causing IC breakdown. IC breakdown may cause injury, smoke or ignition.

[4] Do not insert devices in the wrong orientation or incorrectly.
Make sure that the positive and negative terminals of power supplies are connected properly.
Otherwise, the current or power consumption may exceed the absolute maximum rating, and exceeding the rating(s) may cause the device breakdown, damage or deterioration, and may result injury by explosion or combustion.
In addition, do not use any device that is applied the current with inserting in the wrong orientation or incorrectly even just one time.

Points to remember on handling of ICs

(1) Over current Detection Circuit

Over current detection circuits (referred to as current limiter circuits) do not necessarily protect ICs under all circumstances. If the over current detection circuits operate against the over current, clear the over current status immediately.

Depending on the method of use and usage conditions, such as exceeding absolute maximum ratings can cause the over current detection circuit to not operate properly or IC breakdown before operation. In addition, depending on the method of use and usage conditions, if over current continues to flow for a long time after operation, the IC may generate heat resulting in breakdown.

(2) Thermal Shutdown Circuit

Thermal shutdown circuits do not necessarily protect ICs under all circumstances. If the thermal shutdown circuits operate against the over temperature, clear the heat generation status immediately.

Depending on the method of use and usage conditions, such as exceeding absolute maximum ratings can cause the thermal shutdown circuit to not operate properly or IC breakdown before operation.

(3) Heat Radiation Design

In using an IC with large current flow such as power amp, regulator or driver, please design the device so that heat is appropriately radiated, not to exceed the specified junction temperature (T_j) at any time and condition. These ICs generate heat even during normal use. An inadequate IC heat radiation design can lead to decrease in IC life, deterioration of IC characteristics or IC breakdown. In addition, please design the device taking into considerate the effect of IC heat radiation with peripheral components.

(4) Back-EMF

When a motor rotates in the reverse direction, stops or slows down abruptly, a current flow back to the motor's power supply due to the effect of back-EMF. If the current sink capability of the power supply is small, the device's motor power supply and output pins might be exposed to conditions beyond absolute maximum ratings. To avoid this problem, take the effect of back-EMF into consideration in system design.

(5) Short-circuiting between outputs, air contamination faults, faults due to improper grounding, short-circuiting between contiguous pins

Utmost care is necessary in the design of the power supply lines, GND lines, and output lines since the IC may be destroyed by short-circuiting between outputs, air contamination faults, or faults due to improper grounding, or by short-circuiting between contiguous pins. They may destroy not only the IC but also peripheral parts and may contribute to injuries for users. Over current may continue to flow in the IC because of this destruction and cause smoke or ignition of the IC. Expect the volume of this over current and add an appropriate power supply fuse in order to minimize the effects of the over current. Capacity of the fuse, fusing time, and the inserting position in the circuit should be configured suitably.

RESTRICTIONS ON PRODUCT USE

- Toshiba Corporation, and its subsidiaries and affiliates (collectively "TOSHIBA"), reserve the right to make changes to the information in this document, and related hardware, software and systems (collectively "Product") without notice.

- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.

- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**

- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, devices related to electric power, and equipment used in finance-related fields. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative.

- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.

- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.

- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.

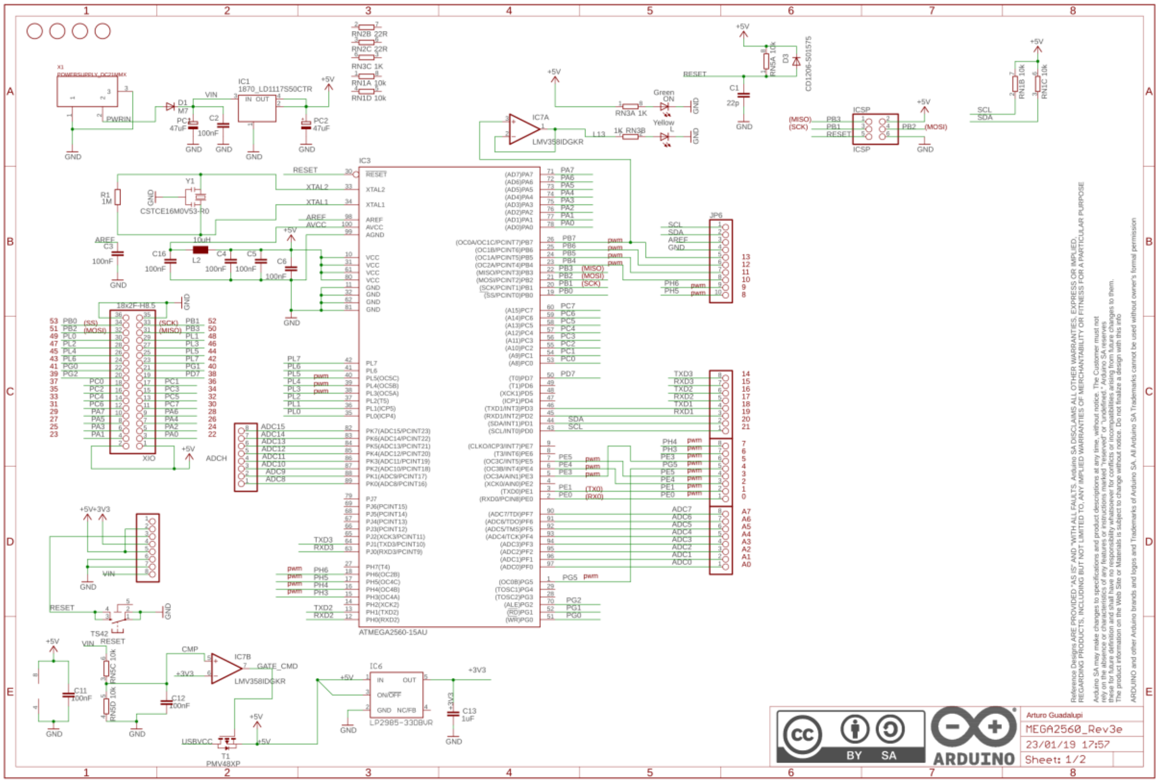
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**

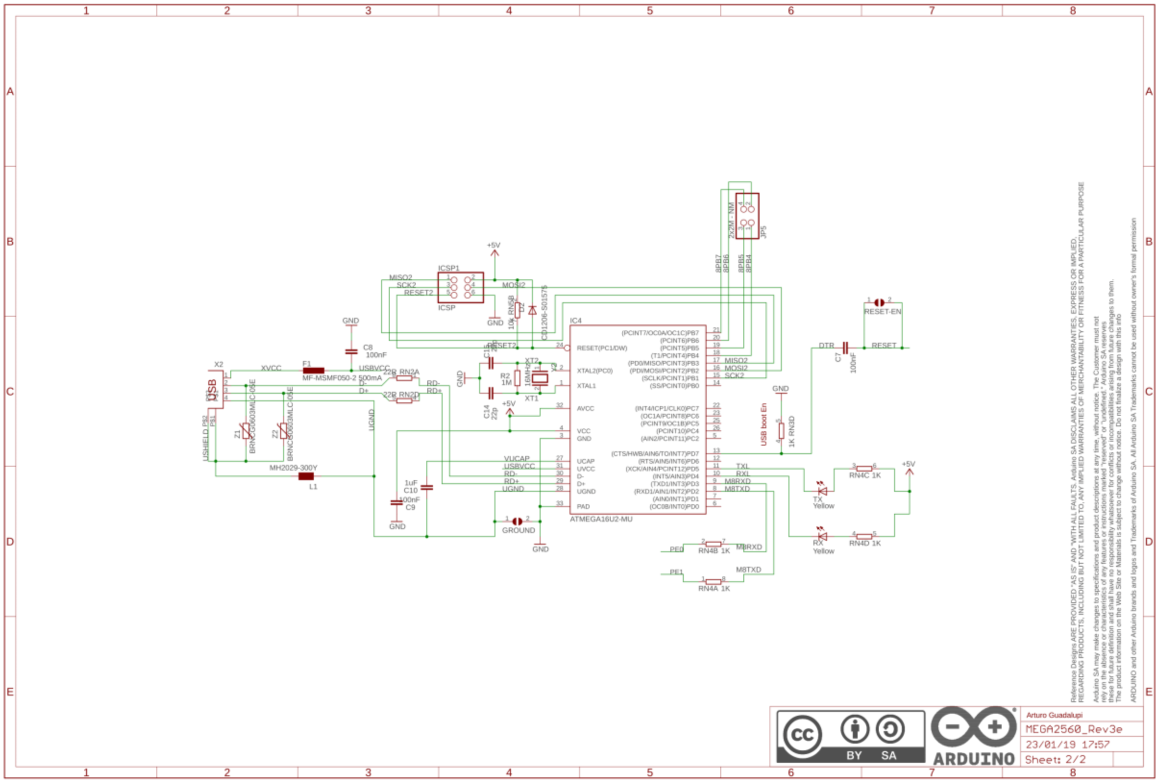
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.

- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**

A.3 Arduino Mega 2560 (ATmega2560) Microcontroller

A.3.1 Arduino Mega 2560 Schematic


ATMEGA2560-16AU
Arturo Guadalupi
MEGA2560_Rev3e
23/01/19 17:57
Sheet: 1/2
ARDUINO
BY
SA



ATMEGA16U2-MU
IC4
ICSP1
ICSP
GROUND
RESET-EN
USB boot En
MF-MSMF050-2 500mA
MH2029-300Y
Arturo Guadalupi
MEGA2560_Rev3e
23/01/19 17:57
Sheet: 2/2
ARDUINO

A.3.2 ATmega2560 Datasheet

Features

- High Performance, Low Power Atmel® AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 135 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 × 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16MHz
  - On-Chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 64K/128K/256KBytes of In-System Self-Programmable Flash
  - 4Kbytes EEPROM
  - 8Kbytes Internal SRAM
  - Write/Erase Cycles:10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
    - Endurance: Up to 64Kbytes Optional External Memory Space
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix® acquisition
  - Up to 64 sense channels
- JTAG (IEEE std. 1149.1 compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - Four 16-bit Timer/Counter with Separate Prescaler, Compare- and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Six/Twelve PWM Channels with Programmable Resolution from 2 to 16 Bits (ATmega1281/2561, ATmega640/1280/2560)
  - Output Compare Modulator
  - 8/16-channel, 10-bit ADC (ATmega1281/2561, ATmega640/1280/2560)
  - Two/Four Programmable Serial USART (ATmega1281/2561, ATmega640/1280/2560)
  - Master/Slave SPI Serial Interface
  - Byte Oriented 2-wire Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 54/86 Programmable I/O Lines (ATmega1281/2561, ATmega640/1280/2560)
  - 64-pad QFN/MLF, 64-lead TQFP (ATmega1281/2561)
  - 100-lead TQFP, 100-ball CBGA (ATmega640/1280/2560)
  - RoHS/Fully Green
- Temperature Range:
  - -40°C to 85°C Industrial
- Ultra-Low Power Consumption
  - Active Mode: 1MHz, 1.8V: 500µA
  - Power-down Mode: 0.1µA at 1.8V
- Speed Grade:
  - ATmega640V/ATmega1280V/ATmega1281V:
    - 0 - 4MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega2560V/ATmega2561V:
    - 0 - 2MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega640/ATmega1280/ATmega1281:
    - 0 - 8MHz @ 2.7V - 5.5V, 0 - 16MHz @ 4.5V - 5.5V
  - ATmega2560/ATmega2561:
    - 0 - 16MHz @ 4.5V - 5.5V

8-bit Atmel Microcontroller with 64K/128K/256K Bytes In-System Programmable Flash

ATmega640/V
ATmega1280/V
ATmega1281/V
ATmega2560/V
ATmega2561/V

Preliminary Summary

2549NS–AVR–05/11

1. Pin Configurations

Figure 1-1. TQFP-pinout ATmega640/1280/2560

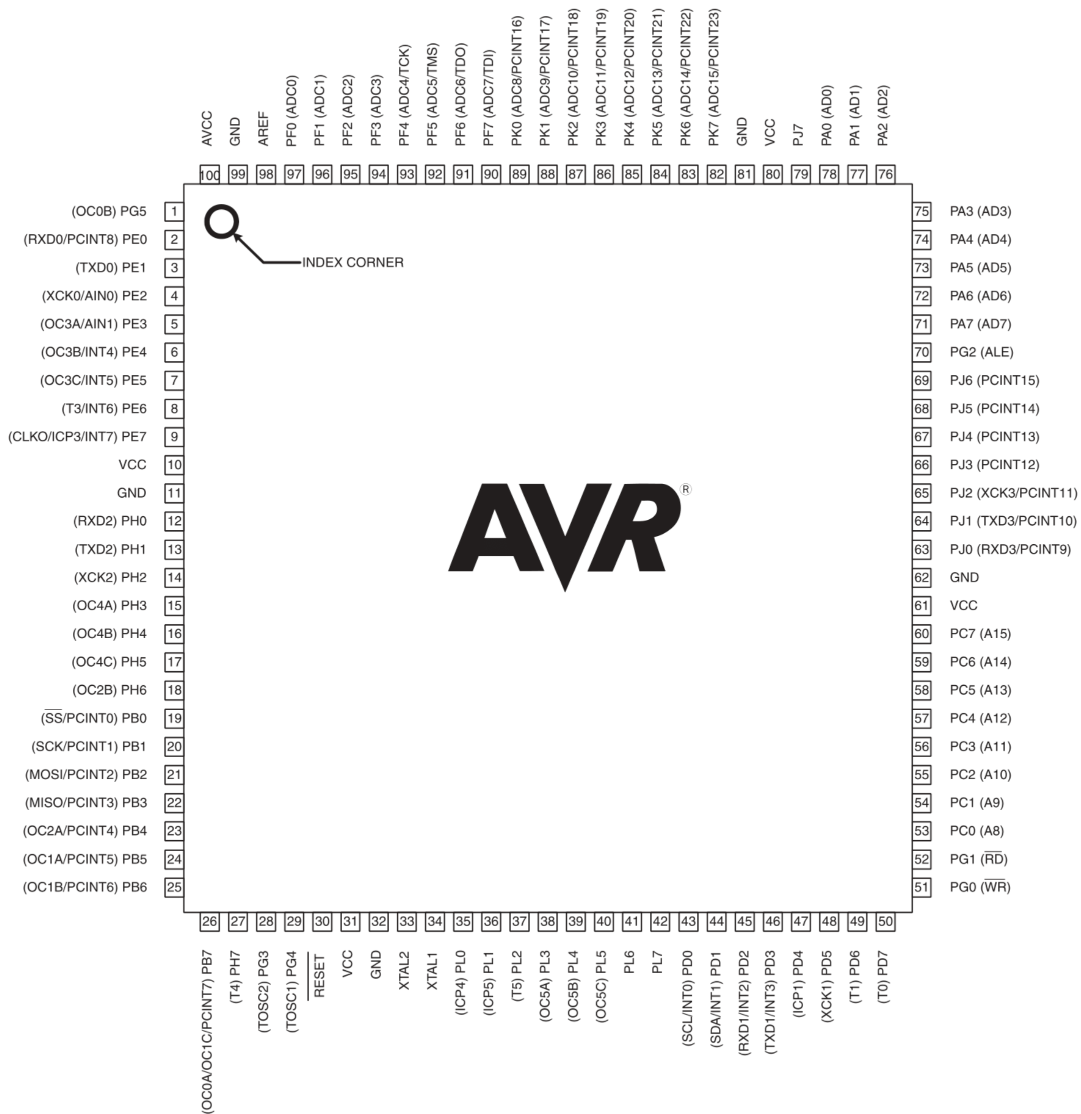


Figure 1-2. CBGA-pinout ATmega640/1280/2560

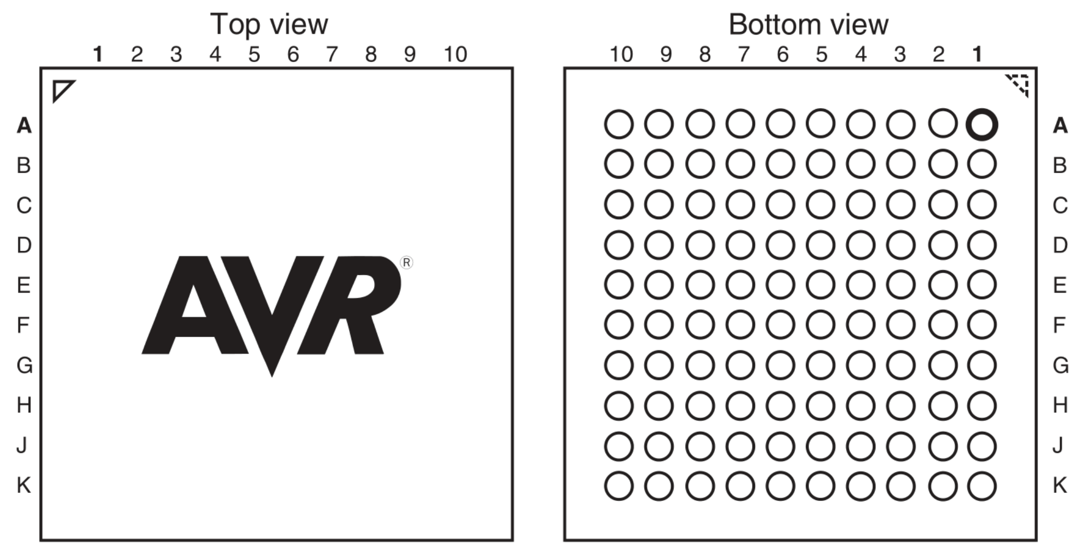
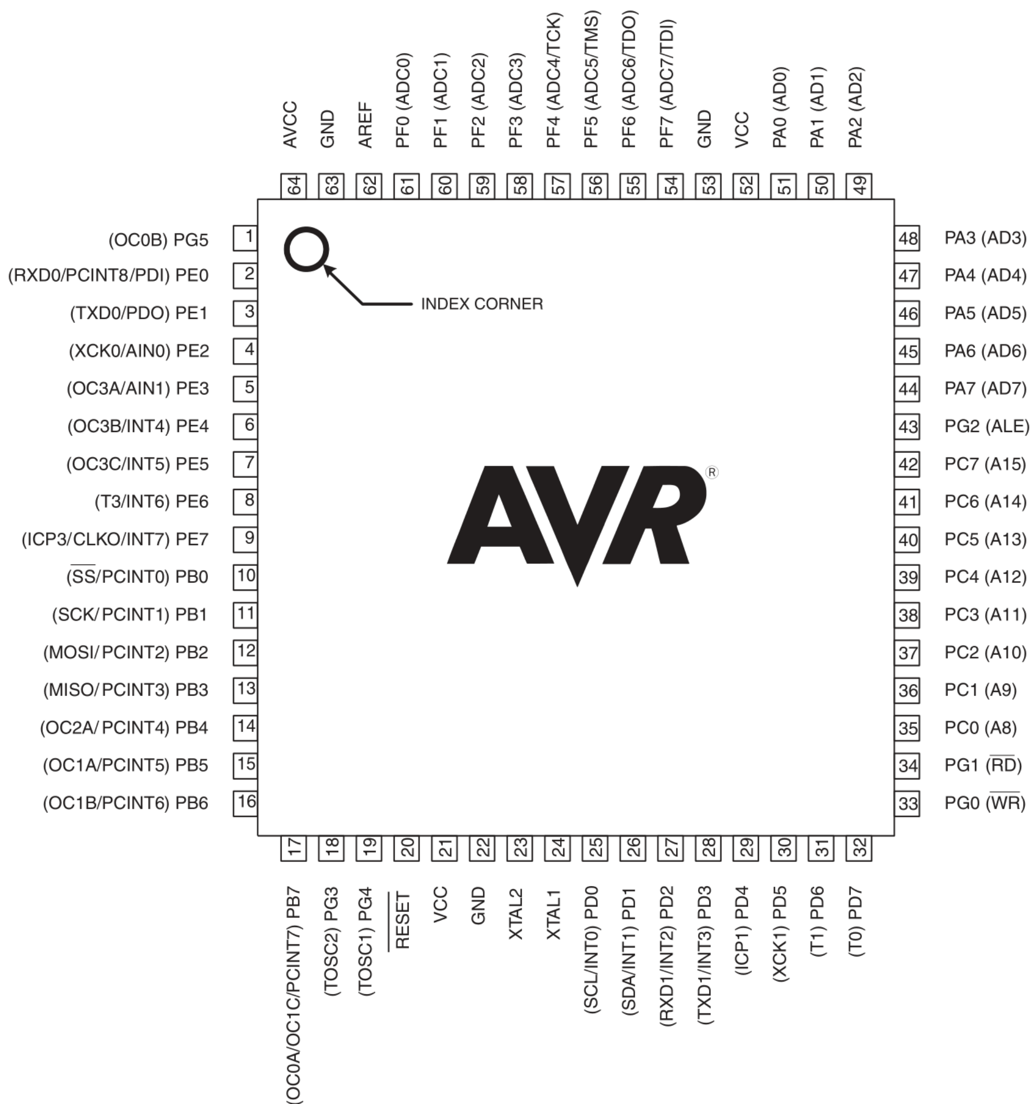


Table 1-1. CBGA-pinout ATmega640/1280/2560

	1	2	3	4	5	6	7	8	9	10
A	GND	AREF	PF0	PF2	PF5	PK0	PK3	PK6	GND	VCC
B	AVCC	PG5	PF1	PF3	PF6	PK1	PK4	PK7	PA0	PA2
C	PE2	PE0	PE1	PF4	PF7	PK2	PK5	PJ7	PA1	PA3
D	PE3	PE4	PE5	PE6	PH2	PA4	PA5	PA6	PA7	PG2
E	PE7	PH0	PH1	PH3	PH5	PJ6	PJ5	PJ4	PJ3	PJ2
F	VCC	PH4	PH6	PB0	PL4	PD1	PJ1	PJ0	PC7	GND
G	GND	PB1	PB2	PB5	PL2	PD0	PD5	PC5	PC6	VCC
H	PB3	PB4	RESET	PL1	PL3	PL7	PD4	PC4	PC3	PC2
J	PH7	PG3	PB6	PL0	XTAL2	PL6	PD3	PC1	PC0	PG1
K	PB7	PG4	VCC	GND	XTAL1	PL5	PD2	PD6	PD7	PG0

Note: The functions for each pin is the same as for the 100 pin packages shown in Figure 1-1 on page 2.

Figure 1-3. Pinout ATmega1281/2561



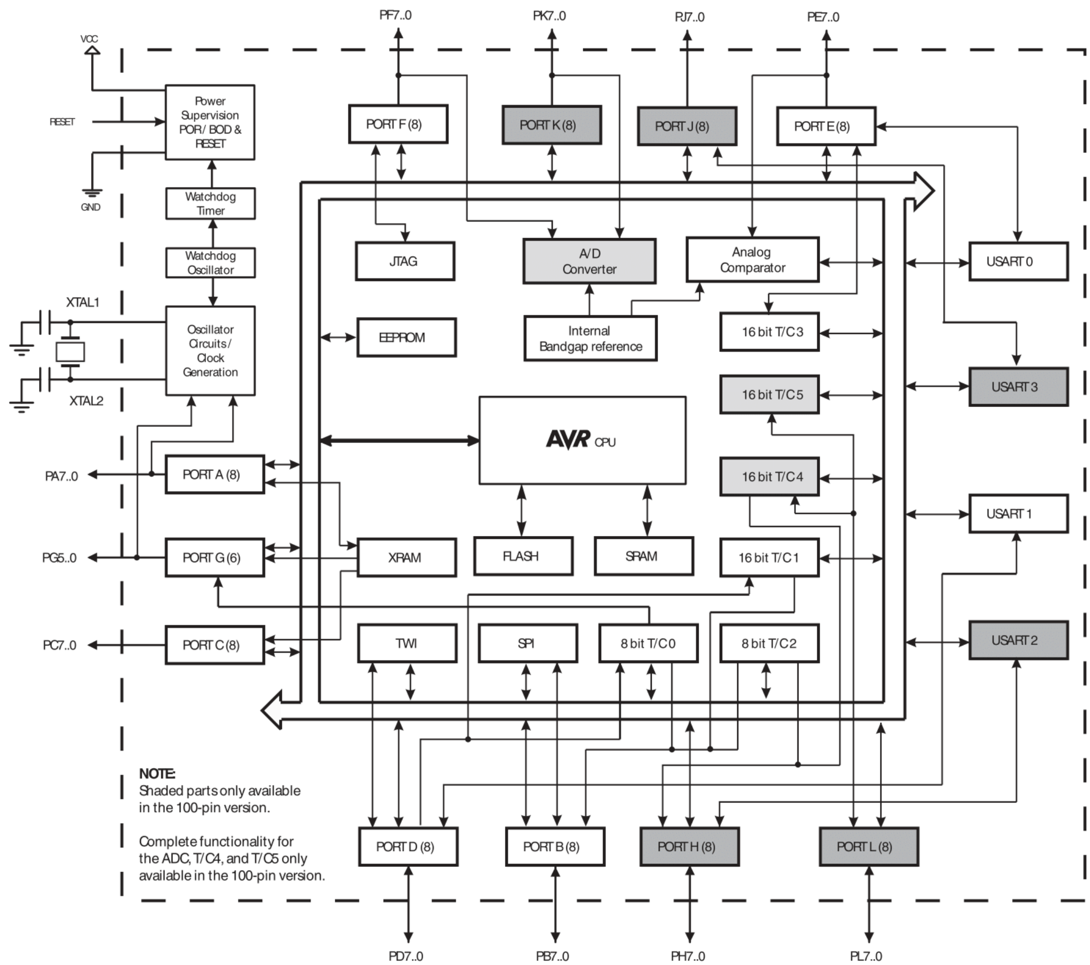
Note: The large center pad underneath the QFN/MLF package is made of metal and internally connected to GND. It should be soldered or glued to the board to ensure good mechanical stability. If the center pad is left unconnected, the package might loosen from the board.

2. Overview

The ATmega640/1280/1281/2560/2561 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega640/1280/1281/2560/2561 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The Atmel® AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega640/1280/1281/2560/2561 provides the following features: 64K/128K/256K bytes of In-System Programmable Flash with Read-While-Write capabilities, 4Kbytes EEPROM, 8 Kbytes SRAM, 54/86 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), six flexible Timer/Counters with compare modes and PWM, 4 USARTs, a byte oriented 2-wire Serial Interface, a 16-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable Watchdog Timer with Internal Oscillator, an SPI serial port, IEEE® std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheelsfunctionality into AVR microcontrollers. The patented charge-transfer signal acquisition offersrobust sensing and includes fully debounced reporting of touch keys and includes Adjacent KeySuppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega640/1280/1281/2560/2561 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega640/1280/1281/2560/2561 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

2.2 Comparison Between ATmega1281/2561 and ATmega640/1280/2560

Each device in the ATmega640/1280/1281/2560/2561 family differs only in memory size and number of pins. Table 2-1 summarizes the different configurations for the six devices.

Table 2-1. Configuration Summary

Device	Flash	EEPROM	RAM	General Purpose I/O pins	16 bits resolution PWM channels	Serial USARTs	ADC Channels
ATmega640	64KB	4KB	8KB	86	12	4	16
ATmega1280	128KB	4KB	8KB	86	12	4	16
ATmega1281	128KB	4KB	8KB	54	6	2	8
ATmega2560	256KB	4KB	8KB	86	12	4	16
ATmega2561	256KB	4KB	8KB	54	6	2	8

2.3 Pin Descriptions

2.3.1 VCC

Digital supply voltage.

2.3.2 GND

Ground.

2.3.3 Port A (PA7..PA0)

Port A is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 78.

2.3.4 Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B has better driving capabilities than the other ports.

Port B also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 79.

2.3.5 Port C (PC7..PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C also serves the functions of special features of the ATmega640/1280/1281/2560/2561 as listed on page 82.

2.3.6 Port D (PD7..PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 83.

2.3.7 Port E (PE7..PE0)

Port E is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port E output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated. The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port E also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 86.

2.3.8 Port F (PF7..PF0)

Port F serves as analog inputs to the A/D Converter.

Port F also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port F output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port F pins that are externally pulled low will source current if the pull-up resistors are activated. The Port F pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PF7(TDI), PF5(TMS), and PF4(TCK) will be activated even if a reset occurs.

Port F also serves the functions of the JTAG interface.

2.3.9 Port G (PG5..PG0)

Port G is a 6-bit I/O port with internal pull-up resistors (selected for each bit). The Port G output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port G pins that are externally pulled low will source current if the pull-up resistors are activated. The Port G pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port G also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 90.

2.3.10 Port H (PH7..PH0)

Port H is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port H output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port H pins that are externally pulled low will source current if the pull-up resistors are activated. The Port H pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port H also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 92.

2.3.11 Port J (PJ7..PJ0)

Port J is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port J output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port J pins that are externally pulled low will source current if the pull-up resistors are activated. The Port J pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port J also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 94.

2.3.12 Port K (PK7..PK0)

Port K serves as analog inputs to the A/D Converter.

Port K is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port K output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port K pins that are externally pulled low will source current if the pull-up resistors are activated. The Port K pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port K also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 96.

2.3.13 Port L (PL7..PL0)

Port L is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port L output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port L pins that are externally pulled low will source current if the pull-up resistors are activated. The Port L pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port L also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 98.

2.3.14 $\overline{\text{RESET}}$

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in "System and Reset Characteristics" on page 372. Shorter pulses are not guaranteed to generate a reset.

2.3.15 XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

2.3.16 XTAL2

Output from the inverting Oscillator amplifier.

2.3.17 AVCC

AVCC is the supply voltage pin for Port F and the A/D Converter. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

2.3.18 AREF

This is the analog reference pin for the A/D Converter.

3. Resources

A comprehensive set of development tools and application notes, and datasheets are available for download on http://www.atmel.com/avr.

4. About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

These code examples assume that the part specific header file is included before compilation. For I/O registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBRS", "SBRC", "SBR", and "CBR".

5. Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 ppm over 20 years at 85°C or 100 years at 25°C.

6. Capacitive touch sensing

The Atmel®QTouch® Library provides a simple to use solution to realize touch sensitive interfaces on most Atmel AVR® microcontrollers. The QTouch Library includes support for the QTouch and QMatrix® acquisition methods.

Touch sensing can be added to any application by linking the appropriate Atmel QTouch Library for the AVR Microcontroller. This is done by using a simple set of APIs to define the touch channels and sensors, and then calling the touch sensing API's to retrieve the channel information and determine the touch sensor states.

The QTouch Library is FREE and downloadable from the Atmel website at the following location: www.atmel.com/qtouchlibrary. For implementation details and other information, refer to the Atmel QTouch Library User Guide - also available for download from the Atmel website.

7. Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0x1FF)	Reserved	-	-	-	-	-	-	-	-	
...	Reserved	-	-	-	-	-	-	-	-	
(0x13F)	Reserved									
(0x13E)	Reserved									
(0x13D)	Reserved									
(0x13C)	Reserved									
(0x13B)	Reserved									
(0x13A)	Reserved									
(0x139)	Reserved									
(0x138)	Reserved									
(0x137)	Reserved									
(0x136)	UDR3	USART3 I/O Data Register								222
(0x135)	UBRR3H	-	-	-	-	USART3 Baud Rate Register High Byte				227
(0x134)	UBRR3L	USART3 Baud Rate Register Low Byte								227
(0x133)	Reserved	-	-	-	-	-	-	-	-	
(0x132)	UCSR3C	UMSEL31	UMSEL30	UPM31	UPM30	USBS3	UCSZ31	UCSZ30	UCPOL3	239
(0x131)	UCSR3B	RXCIE3	TXCIE3	UDRIE3	RXEN3	TXEN3	UCSZ32	RXB83	TXB83	238
(0x130)	UCSR3A	RXC3	TXC3	UDRE3	FE3	DOR3	UPE3	U2X3	MPCM3	238
(0x12F)	Reserved	-	-	-	-	-	-	-	-	
(0x12E)	Reserved	-	-	-	-	-	-	-	-	
(0x12D)	OCR5CH	Timer/Counter5 - Output Compare Register C High Byte								165
(0x12C)	OCR5CL	Timer/Counter5 - Output Compare Register C Low Byte								165
(0x12B)	OCR5BH	Timer/Counter5 - Output Compare Register B High Byte								165
(0x12A)	OCR5BL	Timer/Counter5 - Output Compare Register B Low Byte								165
(0x129)	OCR5AH	Timer/Counter5 - Output Compare Register A High Byte								164
(0x128)	OCR5AL	Timer/Counter5 - Output Compare Register A Low Byte								164
(0x127)	ICR5H	Timer/Counter5 - Input Capture Register High Byte								165
(0x126)	ICR5L	Timer/Counter5 - Input Capture Register Low Byte								165
(0x125)	TCNT5H	Timer/Counter5 - Counter Register High Byte								163
(0x124)	TCNT5L	Timer/Counter5 - Counter Register Low Byte								163
(0x123)	Reserved	-	-	-	-	-	-	-	-	
(0x122)	TCCR5C	FOC5A	FOC5B	FOC5C	-	-	-	-	-	162
(0x121)	TCCR5B	ICNC5	ICES5	-	WGM53	WGM52	CS52	CS51	CS50	160
(0x120)	TCCR5A	COM5A1	COM5A0	COM5B1	COM5B0	COM5C1	COM5C0	WGM51	WGM50	158
(0x11F)	Reserved	-	-	-	-	-	-	-	-	
(0x11E)	Reserved	-	-	-	-	-	-	-	-	
(0x11D)	Reserved	-	-	-	-	-	-	-	-	
(0x11C)	Reserved	-	-	-	-	-	-	-	-	
(0x11B)	Reserved	-	-	-	-	-	-	-	-	
(0x11A)	Reserved	-	-	-	-	-	-	-	-	
(0x119)	Reserved	-	-	-	-	-	-	-	-	
(0x118)	Reserved	-	-	-	-	-	-	-	-	
(0x117)	Reserved	-	-	-	-	-	-	-	-	
(0x116)	Reserved	-	-	-	-	-	-	-	-	
(0x115)	Reserved	-	-	-	-	-	-	-	-	
(0x114)	Reserved	-	-	-	-	-	-	-	-	
(0x113)	Reserved	-	-	-	-	-	-	-	-	
(0x112)	Reserved	-	-	-	-	-	-	-	-	
(0x111)	Reserved	-	-	-	-	-	-	-	-	
(0x110)	Reserved	-	-	-	-	-	-	-	-	
(0x10F)	Reserved	-	-	-	-	-	-	-	-	
(0x10E)	Reserved	-	-	-	-	-	-	-	-	
(0x10D)	Reserved	-	-	-	-	-	-	-	-	
(0x10C)	Reserved	-	-	-	-	-	-	-	-	
(0x10B)	PORTL	PORTL7	PORTL6	PORTL5	PORTL4	PORTL3	PORTL2	PORTL1	PORTL0	104
(0x10A)	DDRL	DDL7	DDL6	DDL5	DDL4	DDL3	DDL2	DDL1	DDL0	104
(0x109)	PINL	PINL7	PINL6	PINL5	PINL4	PINL3	PINL2	PINL1	PINL0	104
(0x108)	PORTK	PORTK7	PORTK6	PORTK5	PORTK4	PORTK3	PORTK2	PORTK1	PORTK0	103
(0x107)	DDRK	DDK7	DDK6	DDK5	DDK4	DDK3	DDK2	DDK1	DDK0	103
(0x106)	PINK	PINK7	PINK6	PINK5	PINK4	PINK3	PINK2	PINK1	PINK0	103
(0x105)	PORTJ	PORTJ7	PORTJ6	PORTJ5	PORTJ4	PORTJ3	PORTJ2	PORTJ1	PORTJ0	103
(0x104)	DDRJ	DDJ7	DDJ6	DDJ5	DDJ4	DDJ3	DDJ2	DDJ1	DDJ0	103
(0x103)	PINJ	PINJ7	PINJ6	PINJ5	PINJ4	PINJ3	PINJ2	PINJ1	PINJ0	103
(0x102)	PORTH	PORTH7	PORTH6	PORTH5	PORTH4	PORTH3	PORTH2	PORTH1	PORTH0	102
(0x101)	DDRH	DDH7	DDH6	DDH5	DDH4	DDH3	DDH2	DDH1	DDH0	103

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0x100)	PINH	PINH7	PINH6	PINH5	PINH4	PINH3	PINH2	PINH1	PINH0	103
(0xFF)	Reserved	-	-	-	-	-	-	-	-	
(0xFE)	Reserved	-	-	-	-	-	-	-	-	
(0xFD)	Reserved	-	-	-	-	-	-	-	-	
(0xFC)	Reserved	-	-	-	-	-	-	-	-	
(0xFB)	Reserved	-	-	-	-	-	-	-	-	
(0xFA)	Reserved	-	-	-	-	-	-	-	-	
(0xF9)	Reserved	-	-	-	-	-	-	-	-	
(0xF8)	Reserved	-	-	-	-	-	-	-	-	
(0xF7)	Reserved	-	-	-	-	-	-	-	-	
(0xF6)	Reserved	-	-	-	-	-	-	-	-	
(0xF5)	Reserved	-	-	-	-	-	-	-	-	
(0xF4)	Reserved	-	-	-	-	-	-	-	-	
(0xF3)	Reserved	-	-	-	-	-	-	-	-	
(0xF2)	Reserved	-	-	-	-	-	-	-	-	
(0xF1)	Reserved	-	-	-	-	-	-	-	-	
(0xF0)	Reserved	-	-	-	-	-	-	-	-	
(0xEF)	Reserved	-	-	-	-	-	-	-	-	
(0xEE)	Reserved	-	-	-	-	-	-	-	-	
(0xED)	Reserved	-	-	-	-	-	-	-	-	
(0xEC)	Reserved	-	-	-	-	-	-	-	-	
(0xEB)	Reserved	-	-	-	-		-	-	-	
(0xEA)	Reserved	-	-	-	-	-	-	-	-	
(0xE9)	Reserved	-	-	-	-	-	-	-	-	
(0xE8)	Reserved	-	-	-	-	-	-	-	-	
(0xE7)	Reserved	-	-	-	-		-	-	-	
(0xE6)	Reserved	-	-	-	-	-	-	-	-	
(0xE5)	Reserved	-	-	-	-	-	-	-	-	
(0xE4)	Reserved	-	-	-	-	-	-	-	-	
(0xE3)	Reserved	-	-	-	-		-	-	-	
(0xE2)	Reserved	-	-	-	-	-	-	-	-	
(0xE1)	Reserved	-	-	-	-		-	-	-	
(0xE0)	Reserved	-	-	-	-		-	-	-	
(0xDF)	Reserved	-	-	-	-	-	-	-	-	
(0xDE)	Reserved	-	-	-	-	-	-	-	-	
(0xDD)	Reserved	-	-	-	-		-	-	-	
(0xDC)	Reserved	-	-	-	-	-	-	-	-	
(0xDB)	Reserved	-	-	-	-	-	-	-	-	
(0xDA)	Reserved	-	-	-	-	-	-	-	-	
(0xD9)	Reserved	-	-	-	-		-	-	-	
(0xD8)	Reserved	-	-	-	-	-	-	-	-	
(0xD7)	Reserved	-	-	-	-	-	-	-	-	
(0xD6)	UDR2	USART2 I/O Data Register								222
(0xD5)	UBRR2H	-	-	-	-	USART2 Baud Rate Register High Byte				227
(0xD4)	UBRR2L	USART2 Baud Rate Register Low Byte								227
(0xD3)	Reserved	-	-	-	-	-	-	-	-	
(0xD2)	UCSR2C	UMSEL21	UMSEL20	UPM21	UPM20	USBS2	UCSZ21	UCSZ20	UCPOL2	239
(0xD1)	UCSR2B	RXCIE2	TXCIE2	UDRIE2	RXEN2	TXEN2	UCSZ22	RXB82	TXB82	238
(0xD0)	UCSR2A	RXC2	TXC2	UDRE2	FE2	DOR2	UPE2	U2X2	MPCM2	238
(0xCF)	Reserved	-	-	-	-	-	-	-	-	
(0xCE)	UDR1	USART1 I/O Data Register								222
(0xCD)	UBRR1H	-	-	-	-	USART1 Baud Rate Register High Byte				227
(0xCC)	UBRR1L	USART1 Baud Rate Register Low Byte								227
(0xCB)	Reserved	-	-	-	-	-	-	-	-	
(0xCA)	UCSR1C	UMSEL11	UMSEL10	UPM11	UPM10	USBS1	UCSZ11	UCSZ10	UCPOL1	239
(0xC9)	UCSR1B	RXCIE1	TXCIE1	UDRIE1	RXEN1	TXEN1	UCSZ12	RXB81	TXB81	238
(0xC8)	UCSR1A	RXC1	TXC1	UDRE1	FE1	DOR1	UPE1	U2X1	MPCM1	238
(0xC7)	Reserved	-	-	-	-	-	-	-	-	
(0xC6)	UDR0	USART0 I/O Data Register								222
(0xC5)	UBRR0H	-	-	-	-	USART0 Baud Rate Register High Byte				227
(0xC4)	UBRR0L	USART0 Baud Rate Register Low Byte								227
(0xC3)	Reserved	-	-	-	-	-	-	-	-	
(0xC2)	UCSR0C	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01	UCSZ00	UCPOL0	239
(0xC1)	UCSR0B	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80	238
(0xC0)	UCSR0A	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0	238
(0xBF)	Reserved	-	-	-	-	-	-	-	-	

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xBE)	Reserved	-	-	-	-	-	-	-	-	
(0xBD)	TWAMR	TWAM6	TWAM5	TWAM4	TWAM3	TWAM2	TWAM1	TWAM0	-	269
(0xBC)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE	266
(0xBB)	TWDR	2-wire Serial Interface Data Register								268
(0xBA)	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	269
(0xB9)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0	268
(0xB8)	TWBR	2-wire Serial Interface Bit Rate Register								266
(0xB7)	Reserved	-	-	-	-	-	-	-	-	
(0xB6)	ASSR	-	EXCLK	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB	184
(0xB5)	Reserved	-	-	-	-	-	-	-	-	
(0xB4)	OCR2B	Timer/Counter2 Output Compare Register B								191
(0xB3)	OCR2A	Timer/Counter2 Output Compare Register A								191
(0xB2)	TCNT2	Timer/Counter2 (8 Bit)								191
(0xB1)	TCCR2B	FOC2A	FOC2B	-	-	WGM22	CS22	CS21	CS20	190
(0xB0)	TCCR2A	COM2A1	COM2A0	COM2B1	COM2B0	-	-	WGM21	WGM20	191
(0xAF)	Reserved	-	-	-	-	-	-	-	-	
(0xAE)	Reserved	-	-	-	-	-	-	-	-	
(0xAD)	OCR4CH	Timer/Counter4 - Output Compare Register C High Byte								164
(0xAC)	OCR4CL	Timer/Counter4 - Output Compare Register C Low Byte								164
(0xAB)	OCR4BH	Timer/Counter4 - Output Compare Register B High Byte								164
(0xAA)	OCR4BL	Timer/Counter4 - Output Compare Register B Low Byte								164
(0xA9)	OCR4AH	Timer/Counter4 - Output Compare Register A High Byte								164
(0xA8)	OCR4AL	Timer/Counter4 - Output Compare Register A Low Byte								164
(0xA7)	ICR4H	Timer/Counter4 - Input Capture Register High Byte								165
(0xA6)	ICR4L	Timer/Counter4 - Input Capture Register Low Byte								165
(0xA5)	TCNT4H	Timer/Counter4 - Counter Register High Byte								163
(0xA4)	TCNT4L	Timer/Counter4 - Counter Register Low Byte								163
(0xA3)	Reserved	-	-	-	-	-	-	-	-	
(0xA2)	TCCR4C	FOC4A	FOC4B	FOC4C	-	-	-	-	-	162
(0xA1)	TCCR4B	ICNC4	ICES4	-	WGM43	WGM42	CS42	CS41	CS40	160
(0xA0)	TCCR4A	COM4A1	COM4A0	COM4B1	COM4B0	COM4C1	COM4C0	WGM41	WGM40	158
(0x9F)	Reserved	-	-	-	-	-	-	-	-	
(0x9E)	Reserved	-	-	-	-	-	-	-	-	
(0x9D)	OCR3CH	Timer/Counter3 - Output Compare Register C High Byte								164
(0x9C)	OCR3CL	Timer/Counter3 - Output Compare Register C Low Byte								164
(0x9B)	OCR3BH	Timer/Counter3 - Output Compare Register B High Byte								164
(0x9A)	OCR3BL	Timer/Counter3 - Output Compare Register B Low Byte								164
(0x99)	OCR3AH	Timer/Counter3 - Output Compare Register A High Byte								163
(0x98)	OCR3AL	Timer/Counter3 - Output Compare Register A Low Byte								163
(0x97)	ICR3H	Timer/Counter3 - Input Capture Register High Byte								165
(0x96)	ICR3L	Timer/Counter3 - Input Capture Register Low Byte								165
(0x95)	TCNT3H	Timer/Counter3 - Counter Register High Byte								162
(0x94)	TCNT3L	Timer/Counter3 - Counter Register Low Byte								162
(0x93)	Reserved	-	-	-	-	-	-	-	-	
(0x92)	TCCR3C	FOC3A	FOC3B	FOC3C	-	-	-	-	-	162
(0x91)	TCCR3B	ICNC3	ICES3	-	WGM33	WGM32	CS32	CS31	CS30	160
(0x90)	TCCR3A	COM3A1	COM3A0	COM3B1	COM3B0	COM3C1	COM3C0	WGM31	WGM30	158
(0x8F)	Reserved	-	-	-	-	-	-	-	-	
(0x8E)	Reserved	-	-	-	-	-	-	-	-	
(0x8D)	OCR1CH	Timer/Counter1 - Output Compare Register C High Byte								163
(0x8C)	OCR1CL	Timer/Counter1 - Output Compare Register C Low Byte								163
(0x8B)	OCR1BH	Timer/Counter1 - Output Compare Register B High Byte								163
(0x8A)	OCR1BL	Timer/Counter1 - Output Compare Register B Low Byte								163
(0x89)	OCR1AH	Timer/Counter1 - Output Compare Register A High Byte								163
(0x88)	OCR1AL	Timer/Counter1 - Output Compare Register A Low Byte								163
(0x87)	ICR1H	Timer/Counter1 - Input Capture Register High Byte								165
(0x86)	ICR1L	Timer/Counter1 - Input Capture Register Low Byte								165
(0x85)	TCNT1H	Timer/Counter1 - Counter Register High Byte								162
(0x84)	TCNT1L	Timer/Counter1 - Counter Register Low Byte								162
(0x83)	Reserved	-	-	-	-	-	-	-	-	
(0x82)	TCCR1C	FOC1A	FOC1B	FOC1C	-	-	-	-	-	161
(0x81)	TCCR1B	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	160
(0x80)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	COM1C1	COM1C0	WGM11	WGM10	158
(0x7F)	DIDR1	-	-	-	-	-	-	AIN1D	AIN0D	274
(0x7E)	DIDR0	ADC7D	ADC6D	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	295
(0x7D)	DIDR2	ADC15D	ADC14D	ADC13D	ADC12D	ADC11D	ADC10D	ADC9D	ADC8D	295

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0x7C)	ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	289
(0x7B)	ADCSRB	-	ACME	-	-	MUX5	ADTS2	ADTS1	ADTS0	272, 290, 294
(0x7A)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	292
(0x79)	ADCH	ADC Data Register High byte								294
(0x78)	ADCL	ADC Data Register Low byte								294
(0x77)	Reserved	-	-	-	-	-	-	-	-	
(0x76)	Reserved	-	-	-	-	-	-	-	-	
(0x75)	XMCRB	XMBK	-	-	-	-	XMM2	XMM1	XMM0	38
(0x74)	XMCRA	SRE	SRL2	SRL1	SRL0	SRW11	SRW10	SRW01	SRW00	37
(0x73)	TIMSK5	-	-	ICIE5	-	OCIE5C	OCIE5B	OCIE5A	TOIE5	166
(0x72)	TIMSK4	-	-	ICIE4	-	OCIE4C	OCIE4B	OCIE4A	TOIE4	166
(0x71)	TIMSK3	-	-	ICIE3	-	OCIE3C	OCIE3B	OCIE3A	TOIE3	166
(0x70)	TIMSK2	-	-	-	-	-	OCIE2B	OCIE2A	TOIE2	193
(0x6F)	TIMSK1	-	-	ICIE1	-	OCIE1C	OCIE1B	OCIE1A	TOIE1	166
(0x6E)	TIMSK0	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0	134
(0x6D)	PCMSK2	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16	116
(0x6C)	PCMSK1	PCINT15	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8	116
(0x6B)	PCMSK0	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	117
(0x6A)	EICRB	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40	114
(0x69)	EICRA	ISC31	ISC30	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00	113
(0x68)	PCICR	-	-	-	-	-	PCIE2	PCIE1	PCIE0	115
(0x67)	Reserved	-	-	-	-	-	-	-	-	
(0x66)	OSCCAL	Oscillator Calibration Register								50
(0x65)	PRR1	-	-	PRTIM5	PRTIM4	PRTIM3	PRUSART3	PRUSART2	PRUSART1	57
(0x64)	PRR0	PRTWI	PRTIM2	PRTIM0	-	PRTIM1	PRSPI	PRUSART0	PRADC	56
(0x63)	Reserved	-	-	-	-	-	-	-	-	
(0x62)	Reserved	-	-	-	-	-	-	-	-	
(0x61)	CLKPR	CLKPCE	-	-	-	CLKPS3	CLKPS2	CLKPS1	CLKPS0	50
(0x60)	WDTCSR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	67
0x3F (0x5F)	SREG	I	T	H	S	V	N	Z	C	14
0x3E (0x5E)	SPH	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	16
0x3D (0x5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	16
0x3C (0x5C)	EIND	-	-	-	-	-	-	-	EIND0	17
0x3B (0x5B)	RAMPZ	-	-	-	-	-	-	RAMPZ1	RAMPZ0	17
0x3A (0x5A)	Reserved	-	-	-	-	-	-	-	-	
0x39 (0x59)	Reserved	-	-	-	-	-	-	-	-	
0x38 (0x58)	Reserved	-	-	-	-	-	-	-	-	
0x37 (0x57)	SPMCSR	SPMIE	RWWSB	SIGRD	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	332
0x36 (0x56)	Reserved	-	-	-	-	-	-	-	-	
0x35 (0x55)	MCUCR	JTD	-	-	PUD	-	-	IVSEL	IVCE	67, 110, 100, 308
0x34 (0x54)	MCUSR	-	-	-	JTRF	WDRF	BORF	EXTRF	PORF	308
0x33 (0x53)	SMCR	-	-	-	-	SM2	SM1	SM0	SE	52
0x32 (0x52)	Reserved	-	-	-	-	-	-	-	-	
0x31 (0x51)	OCDR	OCDR7	OCDR6	OCDR5	OCDR4	OCDR3	OCDR2	OCDR1	OCDR0	301
0x30 (0x50)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	272
0x2F (0x4F)	Reserved	-	-	-	-	-	-	-	-	
0x2E (0x4E)	SPDR	SPI Data Register								204
0x2D (0x4D)	SPSR	SPIF	WCOL	-	-	-	-	-	SPI2X	203
0x2C (0x4C)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	202
0x2B (0x4B)	GPIOR2	General Purpose I/O Register 2								37
0x2A (0x4A)	GPIOR1	General Purpose I/O Register 1								37
0x29 (0x49)	Reserved	-	-	-	-	-	-	-	-	
0x28 (0x48)	OCR0B	Timer/Counter0 Output Compare Register B								133
0x27 (0x47)	OCR0A	Timer/Counter0 Output Compare Register A								133
0x26 (0x46)	TCNT0	Timer/Counter0 (8 Bit)								133
0x25 (0x45)	TCCR0B	FOC0A	FOC0B	-	-	WGM02	CS02	CS01	CS00	132
0x24 (0x44)	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	-	-	WGM01	WGM00	129
0x23 (0x43)	GTCCR	TSM	-	-	-	-	-	PSRASY	PSRSYNC	170, 194
0x22 (0x42)	EEARH	-	-	-	-	EEPROM Address Register High Byte				35
0x21 (0x41)	EEARL	EEPROM Address Register Low Byte								35
0x20 (0x40)	EEDR	EEPROM Data Register								35
0x1F (0x3F)	EECR	-	-	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE	35
0x1E (0x3E)	GPIOR0	General Purpose I/O Register 0								37
0x1D (0x3D)	EIMSK	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0	115
0x1C (0x3C)	EIFR	INTF7	INTF6	INTF5	INTF4	INTF3	INTF2	INTF1	INTF0	115
0x1B (0x3B)	PCIFR	-	-	-	-	-	PCIF2	PCIF1	PCIF0	116

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
0x1A (0x3A)	TIFR5	-	-	ICF5	-	OCF5C	OCF5B	OCF5A	TOV5	166
0x19 (0x39)	TIFR4	-	-	ICF4	-	OCF4C	OCF4B	OCF4A	TOV4	167
0x18 (0x38)	TIFR3	-	-	ICF3	-	OCF3C	OCF3B	OCF3A	TOV3	167
0x17 (0x37)	TIFR2	-	-	-	-	-	OCF2B	OCF2A	TOV2	193
0x16 (0x36)	TIFR1	-	-	ICF1	-	OCF1C	OCF1B	OCF1A	TOV1	167
0x15 (0x35)	TIFR0	-	-	-	-	-	OCF0B	OCF0A	TOV0	134
0x14 (0x34)	PORTG	-	-	PORTG5	PORTG4	PORTG3	PORTG2	PORTG1	PORTG0	102
0x13 (0x33)	DDRG	-	-	DDG5	DDG4	DDG3	DDG2	DDG1	DDG0	102
0x12 (0x32)	PING	-	-	PING5	PING4	PING3	PING2	PING1	PING0	102
0x11 (0x31)	PORTF	PORTF7	PORTF6	PORTF5	PORTF4	PORTF3	PORTF2	PORTF1	PORTF0	101
0x10 (0x30)	DDRF	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0	102
0x0F (0x2F)	PINF	PINF7	PINF6	PINF5	PINF4	PINF3	PINF2	PINF1	PINF0	102
0x0E (0x2E)	PORTE	PORTE7	PORTE6	PORTE5	PORTE4	PORTE3	PORTE2	PORTE1	PORTE0	101
0x0D (0x2D)	DDRE	DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0	101
0x0C (0x2C)	PINE	PINE7	PINE6	PINE5	PINE4	PINE3	PINE2	PINE1	PINE0	102
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	101
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	101
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	101
0x08 (0x28)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	101
0x07 (0x27)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	101
0x06 (0x26)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	101
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	100
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	100
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	100
0x02 (0x22)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	100
0x01 (0x21)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	100
0x00 (0x20)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	100

Notes:
1. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
2. I/O registers within the address range $00 - $1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions.
3. Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers 0x00 to 0x1F only.
4. When using the I/O specific commands IN and OUT, the I/O addresses $00 - $3F must be used. When addressing I/O registers as data space using LD and ST instructions, $20 must be added to these addresses. The ATmega640/1280/1281/2560/2561 is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from $60 - $1FF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

8. Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add two Registers	Rd ← Rd + Rr	Z, C, N, V, H	1
ADC	Rd, Rr	Add with Carry two Registers	Rd ← Rd + Rr + C	Z, C, N, V, H	1
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z, C, N, V, S	2
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd - Rr	Z, C, N, V, H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z, C, N, V, H	1
SBC	Rd, Rr	Subtract with Carry two Registers	Rd ← Rd - Rr - C	Z, C, N, V, H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd - K - C	Z, C, N, V, H	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z, C, N, V, S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z, N, V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z, N, V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z, N, V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z, N, V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z, N, V	1
COM	Rd	One's Complement	Rd ← 0xFF − Rd	Z, C, N, V	1
NEG	Rd	Two's Complement	Rd ← 0x00 − Rd	Z, C, N, V, H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z, N, V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • (0xFF - K)	Z, N, V	1
INC	Rd	Increment	Rd ← Rd + 1	Z, N, V	1
DEC	Rd	Decrement	Rd ← Rd − 1	Z, N, V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z, N, V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z, N, V	1
SER	Rd	Set Register	Rd ← 0xFF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z, C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z, C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z, C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z, C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z, C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z, C	2
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC ← Z	None	2
EIJMP		Extended Indirect Jump to (Z)	PC ←(EIND:Z)	None	2
JMP	k	Direct Jump	PC ← k	None	3
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	4
ICALL		Indirect Call to (Z)	PC ← Z	None	4
EICALL		Extended Indirect Call to (Z)	PC ←(EIND:Z)	None	4
CALL	k	Direct Subroutine Call	PC ← k	None	5
RET		Subroutine Return	PC ← STACK	None	5
RETI		Interrupt Return	PC ← STACK	I	5
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3
CP	Rd,Rr	Compare	Rd − Rr	Z, N, V, C, H	1
CPC	Rd,Rr	Compare with Carry	Rd − Rr − C	Z, N, V, C, H	1
CPI	Rd,K	Compare Register with Immediate	Rd − K	Z, N, V, C, H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC←PC+k + 1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC←PC+k + 1	None	1/2
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2

Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z, C, N, V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z, C, N, V	1
ROL	Rd	Rotate Left Through Carry	Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7)	Z, C, N, V	1
ROR	Rd	Rotate Right Through Carry	Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0)	Z, C, N, V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0..6	Z, C, N, V	1
SWAP	Rd	Swap Nibbles	Rd(3..0)←Rd(7..4),Rd(7..4)←Rd(3..0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
ELPM		Extended Load Program Memory	R0 ← (RAMPZ:Z)	None	3
ELPM	Rd, Z	Extended Load Program Memory	Rd ← (RAMPZ:Z)	None	3

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ELPM	Rd, Z+	Extended Load Program Memory	Rd ← (RAMPZ:Z), RAMPZ:Z ←RAMPZ:Z+1	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
MCU CONTROL INSTRUCTIONS					
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	1
WDR		Watchdog Reset	(see specific descr. for WDR/timer)	None	1
BREAK		Break	For On-chip Debug Only	None	N/A

Note: EICALL and EIJMP do not exist in ATmega640/1280/1281.
ELPM does not exist in ATmega640.

9. Ordering Information

9.1 ATmega640

Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][3]	Operation Range
8	1.8 - 5.5V	ATmega640V-8AU ATmega640V-8AUR[4] ATmega640V-8CU ATmega640V-8CUR[4]	100A 100A 100C1 100C1	Industrial (-40°C to 85°C)
16	2.7 - 5.5V	ATmega640-16AU ATmega640-16AUR[4] ATmega640-16CU ATmega640-16CUR[4]	100A 100A 100C1 100C1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. See “Speed Grades” on page 369.
3. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
4. Tape & Reel

Package Type	
100A	100-lead, Thin (1.0mm) Plastic Gull Wing Quad Flat Package (TQFP)
100C1	100-ball, Chip Ball Grid Array (CBGA)

9.2 ATmega1280

Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][3]	Operation Range
8	1.8V - 5.5V	ATmega1280V-8AU ATmega1280V-8AUR[4] ATmega1280V-8CU ATmega1280V-8CUR[4]	100A 100A 100C1 100C1	Industrial (-40°C to 85°C)
16	2.7V - 5.5V	ATmega1280-16AU ATmega1280-16AUR[4] ATmega1280-16CU ATmega1280-16CUR[4]	100A 100A 100C1 100C1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. See “Speed Grades” on page 369.
3. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
4. Tape & Reel

Package Type	
100A	100-lead, Thin (1.0mm) Plastic Gull Wing Quad Flat Package (TQFP)
100C1	100-ball, Chip Ball Grid Array (CBGA)

9.3 ATmega1281

Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][3]	Operation Range
8	1.8 - 5.5V	ATmega1281V-8AU ATmega1281V-8AUR[4] ATmega1281V-8MU ATmega1281V-8MUR[4]	64A 64A 64M2 64M2	Industrial (-40°C to 85°C)
16	2.7 - 5.5V	ATmega1281-16AU ATmega1281-16AUR[4] ATmega1281-16MU ATmega1281-16MUR[4]	64A 64A 64M2 64M2	

Notes: 1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

2. See "Speed Grades" on page 369.

3. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.

4. Tape & Reel

Package Type	
64A	64-lead, Thin (1.0mm) Plastic Gull Wing Quad Flat Package (TQFP)
64M2	64-pad, 9mm × 9mm × 1.0mm Body, Quad Flat No-lead/Micro Lead Frame Package (QFN/MLF)

9.4 ATmega2560

Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][3]	Operation Range
8	1.8V - 5.5V	ATmega2560V-8AU ATmega2560V-8AUR[4] ATmega2560V-8CU ATmega2560V-8CUR[4]	100A 100A 100C1 100C1	Industrial (-40°C to 85°C)
16	4.5V - 5.5V	ATmega2560-16AU ATmega2560-16AUR[4] ATmega2560-16CU ATmega2560-16CUR[4]	100A 100A 100C1 100C1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. See "Speed Grades" on page 369.
3. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
4. Tape & Reel

Package Type	
100A	100-lead, Thin (1.0mm) Plastic Gull Wing Quad Flat Package (TQFP)
100C1	100-ball, Chip Ball Grid Array (CBGA)

9.5 ATmega2561

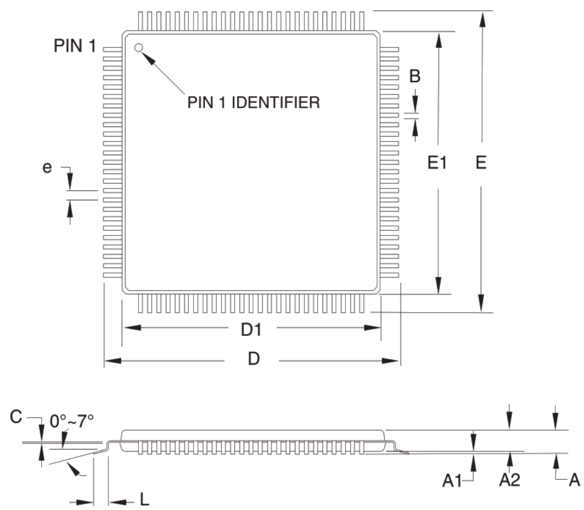
Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][3]	Operation Range
8	1.8V - 5.5V	ATmega1281V-8AU ATmega1281V-8AUR[4] ATmega1281V-8MU ATmega1281V-8MUR[4]	64A 64A 64M2 64M2	Industrial (-40°C to 85°C)
16	4.5V - 5.5V	ATmega1281-16AU ATmega1281-16AUR[4] ATmega1281-16MU ATmega1281-16MUR[4]	64A 64A 64M2 64M2	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. See "Speed Grades" on page 369.
3. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
4. Tape & Reel

Package Type	
64A	64-lead, Thin (1.0mm) Plastic Gull Wing Quad Flat Package (TQFP)
64M2	64-pad, 9mm × 9mm × 1.0mm Body, Quad Flat No-lead/Micro Lead Frame Package (QFN/MLF)

10. Packaging Information

10.1 100A



COMMON DIMENSIONS
(Unit of Measure = mm)

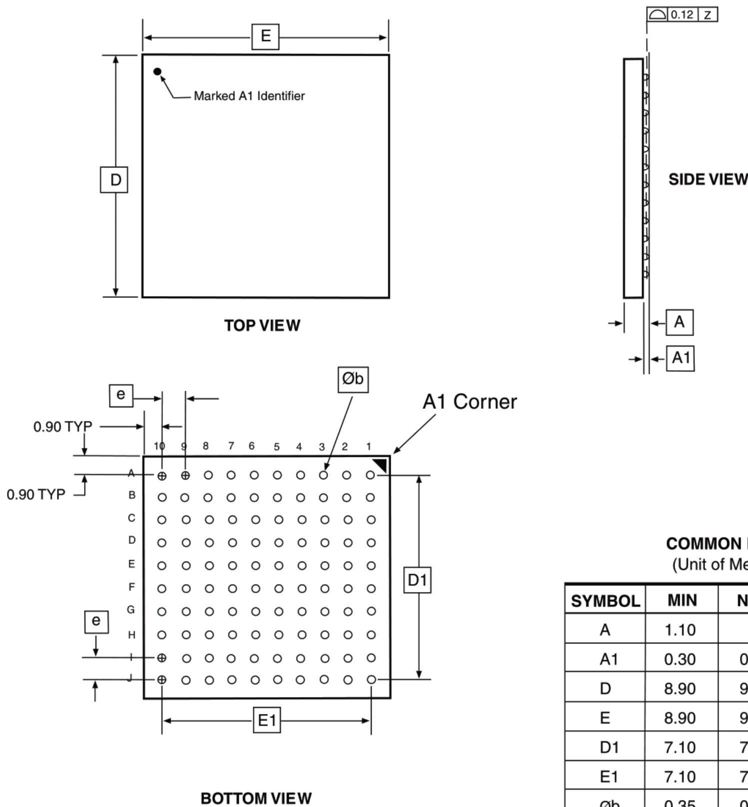
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	15.75	16.00	16.25	
D1	13.90	14.00	14.10	Note 2
E	15.75	16.00	16.25	
E1	13.90	14.00	14.10	Note 2
B	0.17	–	0.27	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.50 TYP			

Notes:
1. This package conforms to JEDEC reference MS-026, Variation AED.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.08 mm maximum.

2010-10-20

	TITLE	DRAWING NO.	REV.
ATMEL 2325 Orchard Parkway San Jose, CA 95131	**100A,** 100-lead, 14 x 14 mm Body Size, 1.0 mm Body Thickness, 0.5 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	100A	D

10.2 100C1



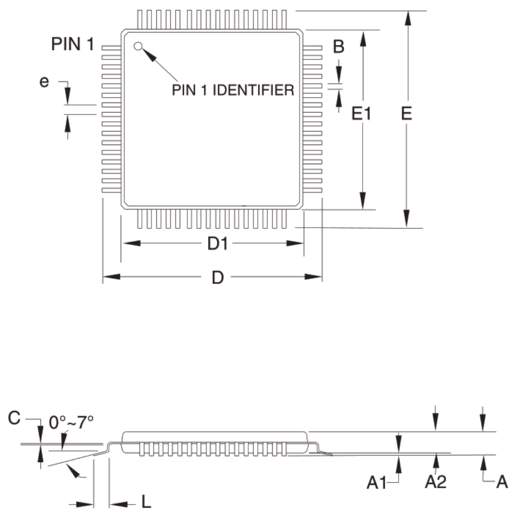
COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	1.10	–	1.20	
A1	0.30	0.35	0.40	
D	8.90	9.00	9.10	
E	8.90	9.00	9.10	
D1	7.10	7.20	7.30	
E1	7.10	7.20	7.30	
Øb	0.35	0.40	0.45	
e		0.80 TYP		

5/25/06

	TITLE	DRAWING NO.	REV.
ATMEL 2325 Orchard Parkway San Jose, CA 95131	**100C1**, 100-ball, 9 x 9 x 1.2 mm Body, Ball Pitch 0.80 mm Chip Array BGA Package (CBGA)	100C1	A

10.3 64A



COMMON DIMENSIONS
(Unit of Measure = mm)

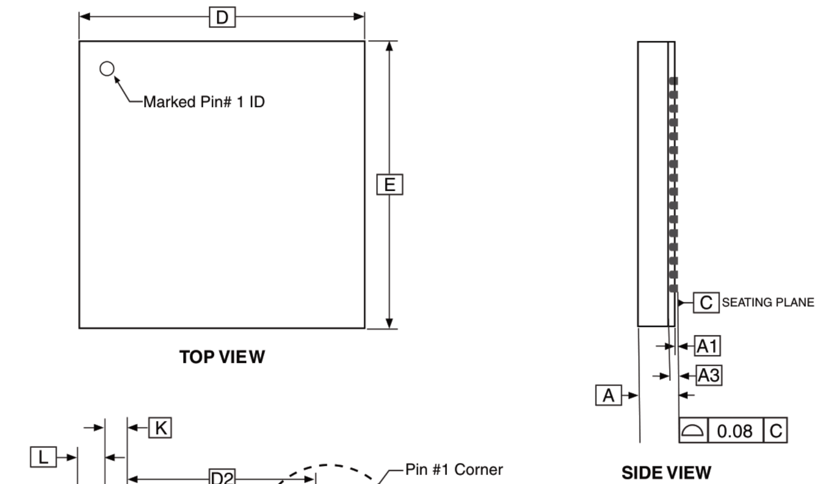
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	15.75	16.00	16.25	
D1	13.90	14.00	14.10	Note 2
E	15.75	16.00	16.25	
E1	13.90	14.00	14.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

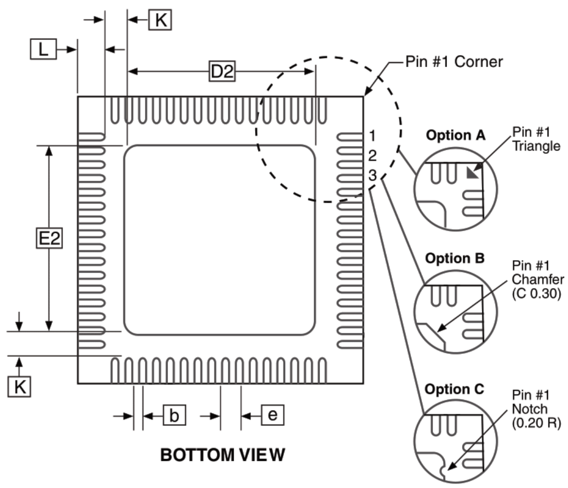
Notes:
1. This package conforms to JEDEC reference MS-026, Variation AEB.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

2010-10-20

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**64A,** 64-lead, 14 x 14 mm Body Size, 1.0 mm Body Thickness, 0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	64A	C

10.4 64M2





COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	–	0.02	0.05	
A3		0.20 REF		
b	0.18	0.25	0.30	
D	8.90	9.00	9.10	
D2	7.50	7.65	7.80	
E	8.90	9.00	9.10	
E2	7.50	7.65	7.80	
e		0.50 BSC		
L	0.35	0.40	0.45	
K	0.20	0.27	0.40	

Notes: 1. JEDEC Standard MO-220, (SAW Singulation) Fig. 1, VMMD.
2. Dimension and tolerance conform to ASMEY14.5M-1994.

2010-10-20

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**64M2**, 64-pad, 9 x 9 x 1.0 mm Body, Lead Pitch 0.50 mm, 7.65 mm Exposed Pad, Micro Lead Frame Package (MLF)	64M2	E

11. Errata

11.1 ATmega640 rev. B

- **Inaccurate ADC conversion in differential mode with 200× gain**
- **High current consumption in sleep mode**

1. Inaccurate ADC conversion in differential mode with 200× gain

With AVCC <3.6V, random conversions will be inaccurate. Typical absolute accuracy may reach 64 LSB.

Problem Fix/Workaround

None.

2. High current consumption in sleep mode

If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

Problem Fix/Workaround

Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.2 ATmega640 rev. A

- **Inaccurate ADC conversion in differential mode with 200× gain**
- **High current consumption in sleep mode**

1. Inaccurate ADC conversion in differential mode with 200× gain

With AVCC <3.6V, random conversions will be inaccurate. Typical absolute accuracy may reach 64 LSB.

Problem Fix/Workaround

None.

2. High current consumption in sleep mode

If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

Problem Fix/Workaround

Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.3 ATmega1280 rev. B

- **Inaccurate ADC conversion in differential mode with 200× gain**
- **High current consumption in sleep mode**

1. Inaccurate ADC conversion in differential mode with 200× gain

With AVCC <3.6V, random conversions will be inaccurate. Typical absolute accuracy may reach 64 LSB.

Problem Fix/Workaround

None.

2. **High current consumption in sleep mode**

   If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

   Problem Fix/Workaround

   Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.4 ATmega1280 rev. A

- **Inaccurate ADC conversion in differential mode with 200× gain**
- **High current consumption in sleep mode**

1. **Inaccurate ADC conversion in differential mode with 200× gain**

   With AVCC <3.6V, random conversions will be inaccurate. Typical absolute accuracy may reach 64 LSB.

   Problem Fix/Workaround

   None.

2. **High current consumption in sleep mode**

   If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

   Problem Fix/Workaround

   Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.5 ATmega1281 rev. B

- **Inaccurate ADC conversion in differential mode with 200× gain**
- **High current consumption in sleep mode**

1. **Inaccurate ADC conversion in differential mode with 200× gain**

   With AVCC <3.6V, random conversions will be inaccurate. Typical absolute accuracy may reach 64 LSB.

   Problem Fix/Workaround

   None.

2. **High current consumption in sleep mode**

   If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

   Problem Fix/Workaround

   Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.6 ATmega1281 rev. A

- **Inaccurate ADC conversion in differential mode with 200× gain**
- **High current consumption in sleep mode**

1. Inaccurate ADC conversion in differential mode with 200× gain

With AVCC <3.6V, random conversions will be inaccurate. Typical absolute accuracy may reach 64 LSB.

Problem Fix/Workaround

None.

2. High current consumption in sleep mode

If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

Problem Fix/Workaround

Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.7 ATmega2560 rev. F

Not sampled.

11.8 ATmega2560 rev. E

No known errata.

11.9 ATmega2560 rev. D

Not sampled.

11.10 ATmega2560 rev. C

- **High current consumption in sleep mode**

1. High current consumption in sleep mode

If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

Problem Fix/Workaround

Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.11 ATmega2560 rev. B

Not sampled.

11.12 ATmega2560 rev. A

- **Non-Read-While-Write area of flash not functional**
- **Part does not work under 2.4 volts**
- **Incorrect ADC reading in differential mode**
- **Internal ADC reference has too low value**
- **IN/OUT instructions may be executed twice when Stack is in external RAM**
- **EEPROM read from application code does not work in Lock Bit Mode 3**

1. Non-Read-While-Write area of flash not functional

The Non-Read-While-Write area of the flash is not working as expected. The problem is related to the speed of the part when reading the flash of this area.

Problem Fix/Workaround

- Only use the first 248K of the flash.

- If boot functionality is needed, run the code in the Non-Read-While-Write area at maximum 1/4th of the maximum frequency of the device at any given voltage. This is done by writing the CLKPR register before entering the boot section of the code.

2. Part does not work under 2.4 volts

The part does not execute code correctly below 2.4 volts.

Problem Fix/Workaround

Do not use the part at voltages below 2.4 volts.

3. Incorrect ADC reading in differential mode

The ADC has high noise in differential mode. It can give up to 7 LSB error.

Problem Fix/Workaround

Use only the 7 MSB of the result when using the ADC in differential mode.

4. Internal ADC reference has too low value

The internal ADC reference has a value lower than specified.

Problem Fix/Workaround

- Use AVCC or external reference.

- The actual value of the reference can be measured by applying a known voltage to the ADC when using the internal reference. The result when doing later conversions can then be calibrated.

5. IN/OUT instructions may be executed twice when Stack is in external RAM

If either an IN or an OUT instruction is executed directly before an interrupt occurs and the stack pointer is located in external ram, the instruction will be executed twice. In some cases this will cause a problem, for example:

- If reading SREG it will appear that the I-flag is cleared.

- If writing to the PIN registers, the port will toggle twice.

- If reading registers with interrupt flags, the flags will appear to be cleared.

Problem Fix/Workaround

There are two application work-arounds, where selecting one of them, will be omitting the issue:

- Replace IN and OUT with LD/LDS/LDD and ST/STS/STD instructions.

- Use internal RAM for stack pointer.

6. **EEPROM read from application code does not work in Lock Bit Mode 3**

   When the Memory Lock Bits LB2 and LB1 are programmed to mode 3, EEPROM read does not work from the application code.

   Problem Fix/Workaround

   Do not set Lock Bit Protection Mode 3 when the application code needs to read from EEPROM.

11.13 ATmega2561 rev. F

Not sampled.

11.14 ATmega2561 rev. E

No known errata.

11.15 ATmega2561 rev. D

Not sampled.

11.16 ATmega2561 rev. C

- **High current consumption in sleep mode.**

1. **High current consumption in sleep mode**

   If a pending interrupt cannot wake the part up from the selected sleep mode, the current consumption will increase during sleep when executing the SLEEP instruction directly after a SEI instruction.

   Problem Fix/Workaround

   Before entering sleep, interrupts not used to wake the part from the sleep mode should be disabled.

11.17 ATmega2561 rev. B

Not sampled.

11.18 ATmega2561 rev. A

- **Non-Read-While-Write area of flash not functional**
- **Part does not work under 2.4 Volts**
- **Incorrect ADC reading in differential mode**
- **Internal ADC reference has too low value**
- **IN/OUT instructions may be executed twice when Stack is in external RAM**
- **EEPROM read from application code does not work in Lock Bit Mode 3**

1. **Non-Read-While-Write area of flash not functional**
   The Non-Read-While-Write area of the flash is not working as expected. The problem is related to the speed of the part when reading the flash of this area.

   Problem Fix/Workaround
   - Only use the first 248K of the flash.

   - If boot functionality is needed, run the code in the Non-Read-While-Write area at maximum 1/4th of the maximum frequency of the device at any given voltage. This is done by writing the CLKPR register before entering the boot section of the code.

2. **Part does not work under 2.4 volts**
   The part does not execute code correctly below 2.4 volts.

   Problem Fix/Workaround
   Do not use the part at voltages below 2.4 volts.

3. **Incorrect ADC reading in differential mode**
   The ADC has high noise in differential mode. It can give up to 7 LSB error.

   Problem Fix/Workaround
   Use only the 7 MSB of the result when using the ADC in differential mode.

4. **Internal ADC reference has too low value**
   The internal ADC reference has a value lower than specified.

   Problem Fix/Workaround
   - Use AVCC or external reference.

   - The actual value of the reference can be measured by applying a known voltage to the ADC when using the internal reference. The result when doing later conversions can then be calibrated.

5. **IN/OUT instructions may be executed twice when Stack is in external RAM**
   If either an IN or an OUT instruction is executed directly before an interrupt occurs and the stack pointer is located in external ram, the instruction will be executed twice. In some cases this will cause a problem, for example:

   - If reading SREG it will appear that the I-flag is cleared.

   - If writing to the PIN registers, the port will toggle twice.

   - If reading registers with interrupt flags, the flags will appear to be cleared.

   Problem Fix/Workaround
   There are two application workarounds, where selecting one of them, will be omitting the issue:

   - Replace IN and OUT with LD/LDS/LDD and ST/STS/STD instructions.

   - Use internal RAM for stack pointer.

6. **EEPROM read from application code does not work in Lock Bit Mode 3**
   When the Memory Lock Bits LB2 and LB1 are programmed to mode 3, EEPROM read does not work from the application code.

Problem Fix/Workaround

Do not set Lock Bit Protection Mode 3 when the application code needs to read from EEPROM.

12. Datasheet Revision History

Please note that the referring page numbers in this section are referring to this document.The referring revision in this section are referring to the document revision.

12.1 Rev. 2549N-05/11

1. Added Atmel QTouch Library Support and QTouch Sensing Capablity Features
2. Updated Cross-reference in "Bit 5, 2:0 - WDP3:0: Watchdog Timer Prescaler 3, 2, 1 and 0" on page 68
3. Updated Assembly codes in section "USART Initialization" on page 210
4. Added "Standard Power-On Reset" on page 372.
5. Added "Enhanced Power-On Reset" on page 373.
6. Updated Figure 32-13 on page 393
7. Updated "Ordering Information" on page 20 to include Tape & Reel devices.

12.2 Rev. 2549M-09/10

1. Updated typos in Figure 26-9 on page 285 and in Figure 26-10 on page 285.
2. Note is added below Table 1-1 on page 3.
3. The values for "typical characteristics" in Table 31-9 on page 377 and Table 31-10 on page 378, has been rounded.
4. Units for tRST and tBOD in Table 31-3 on page 372 have been changed from "ns" to "µs".
5. The figure text for Table 31-2 on page 371 has been changed.
6. Text in first column in Table 30-3 on page 336 has been changed from "Fuse Low Byte" to "Extended Fuse Byte".
7. The text in "Power Reduction Register" on page 54 has been changed.
8. The value of the inductor in Figure 26-9 on page 285 and Figure 26-10 on page 285 has been changed to 10 µH.
9. "Port A" has been changed into "Port K" in the first paragraph of "Features" on page 275.
10. Minimum wait delay for tWD_EEPROM in Table 30-16 on page 351 has been changed from 9.0ms to 3.6ms
11. Dimension A3 is added in "64M2" on page 28.
12. Several cross-references are corrected.
13. "COM0A1:0" on page 130 is corrected to "COM0B1:0".
14. Corrected some Figure and Table numbering.
15. Updated Section 10.6 "Low Frequency Crystal Oscillator" on page 45.

12.3 Rev. 2549L-08/07

1. Updated note in Table 10-11 on page 47.
2. Updated Table 10-3 on page 43, Table 10-5 on page 44, Table 10-9 on page 47.
3. Updated typos in "DC Characteristics" on page 367
4. Updated "Clock Characteristics" on page 371
5. Updated "External Clock Drive" on page 371.
6. Added "System and Reset Characteristics" on page 372.

7. Updated "SPI Timing Characteristics" on page 375.
8. Updated "ADC Characteristics – Preliminary Data" on page 377.
9. Updated ordering code in "ATmega640" on page 20.

12.4 Rev. 2549K-01/07

1. Updated Table 1-1 on page 3.
2. Updated "Pin Descriptions" on page 7.
3. Updated "Stack Pointer" on page 16.
4. Updated "Bit 1 – EEPE: EEPROM Programming Enable" on page 36.
5. Updated Assembly code example in "Thus, when the BOD is not enabled, after setting the ACBG bit or enabling the ADC, the user must always allow the reference to start up before the output from the Analog Comparator or ADC is used. To reduce power consumption in Power-down mode, the user can avoid the three conditions above to ensure that the reference is turned off before entering Power-down mode." on page 63.
6. Updated "EIMSK – External Interrupt Mask Register" on page 115.
7. Updated Bit description in "PCIFR – Pin Change Interrupt Flag Register" on page 116.
8. Updated code example in "USART Initialization" on page 210.
9. Updated Figure 26-8 on page 284.
10. Updated "DC Characteristics" on page 367.

12.5 Rev. 2549J-09/06

1. Updated "" on page 46.
2. Updated code example in "Moving Interrupts Between Application and Boot Section" on page 109.
3. Updated "Timer/Counter Prescaler" on page 186.
4. Updated "Device Identification Register" on page 303.
5. Updated "Signature Bytes" on page 338.
6. Updated "Instruction Set Summary" on page 17.

12.6 Rev. 2549I-07/06

1. Added "Data Retention" on page 11.
2. Updated Table 16-3 on page 129, Table 16-6 on page 130, Table 16-8 on page 131, Table 17-2 on page 148, Table 17-4 on page 159, Table 17-5 on page 160, Table 20-3 on page 187, Table 20-6 on page 188 and Table 20-8 on page 189.
3. Updated "Fast PWM Mode" on page 150.

12.7 Rev. 2549H-06/06

1. Updated "" on page 46.
2. Updated "OSCCAL – Oscillator Calibration Register" on page 50.
3. Added Table 31-1 on page 371.

12.8 Rev. 2549G-06/06

1. Updated "Features" on page 1.
2. Added Figure 1-2 on page 3, Table 1-1 on page 3.
3. Updated "" on page 46.
4. Updated "Power Management and Sleep Modes" on page 52.
5. Updated note for Table 12-1 on page 68.
6. Updated Figure 26-9 on page 285 and Figure 26-10 on page 285.
7. Updated "Setting the Boot Loader Lock Bits by SPM" on page 324.
8. Updated "Ordering Information" on page 20.
9. Added Package information "100C1" on page 26.
10. Updated "Errata" on page 29.

12.9 Rev. 2549F-04/06

1. Updated Figure 9-3 on page 31, Figure 9-4 on page 31 and Figure 9-5 on page 32.
2. Updated Table 20-2 on page 187 and Table 20-3 on page 187.
3. Updated Features in "ADC – Analog to Digital Converter" on page 275.
4. Updated "Fuse Bits" on page 336.

12.10 Rev. 2549E-04/06

1. Updated "Features" on page 1.
2. Updated Table 12-1 on page 62.
3. Updated note for Table 12-1 on page 62.
4. Updated "Bit 6 – ACBG: Analog Comparator Bandgap Select" on page 273.
5. Updated "Prescaling and Conversion Timing" on page 278.
5. Updated "Maximum speed vs. V_{CC}" on page 373.
6. Updated "Ordering Information" on page 20.

12.11 Rev. 2549D-12/05

1. Advanced Information Status changed to Preliminary.
2. Changed number of I/O Ports from 51 to 54.
3. Updatet typos in "TCCR0A – Timer/Counter Control Register A" on page 129.
4. Updated Features in "ADC – Analog to Digital Converter" on page 275.
5. Updated Operation in"ADC – Analog to Digital Converter" on page 275
6. Updated Stabilizing Time in "Changing Channel or Reference Selection" on page 282.
7. Updated Figure 26-1 on page 276, Figure 26-9 on page 285, Figure 26-10 on page 285.
8. Updated Text in "ADCSRB – ADC Control and Status Register B" on page 290.
9. Updated Note for Table 4 on page 43, Table 13-15 on page 86, Table 26-3 on page 289 and Table 26-6 on page 295.
10. Updated Table 31-9 on page 377 and Table 31-10 on page 378.
11. Updated "Filling the Temporary Buffer (Page Loading)" on page 323.
12. Updated "Typical Characteristics" on page 385.
13. Updated "Packaging Information" on page 25.
14. Updated "Errata" on page 29.

12.12 Rev. 2549C-09/05

1. Updated Speed Grade in section "Features" on page 1.
2. Added "Resources" on page 11.
3. Updated "SPI – Serial Peripheral Interface" on page 195. In Slave mode, low and high period SPI clock must be larger than 2 CPU cycles.
4. Updated "Bit Rate Generator Unit" on page 247.
5. Updated "Maximum speed vs. V_{CC}" on page 373.
6. Updated "Ordering Information" on page 20.
7. Updated "Packaging Information" on page 25. Package 64M1 replaced by 64M2.
8. Updated "Errata" on page 29.

12.13 Rev. 2549B-05/05

1. JTAG ID/Signature for ATmega640 updated: 0x9608.
2. Updated Table 13-7 on page 81.
3. Updated "Serial Programming Instruction set" on page 352.
4. Updated "Errata" on page 29.

12.14 Rev. 2549A-03/05

1. Initial version.

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
JAPAN
Tel: (+81)(3) 3523-3551
Fax: (+81)(3) 3523-7581

2549NS–AVR–05/11

A.4 MCP23017-E/SP I/O Expander

MICROCHIP

MCP23017/MCP23S17

16-Bit I/O Expander with Serial Interface

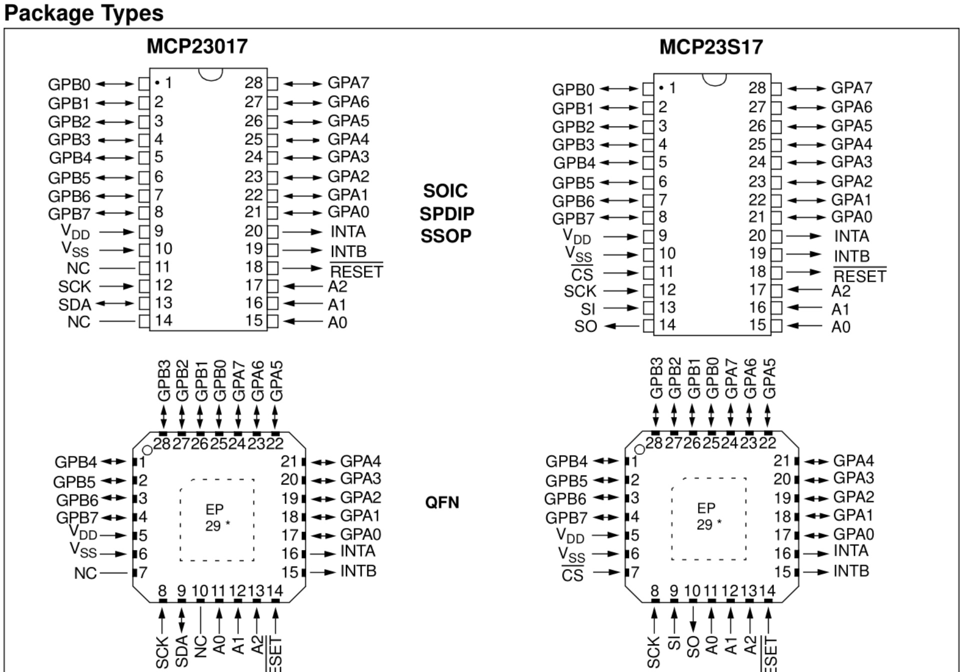
Features

- 16-Bit Remote Bidirectional I/O Port:
  - I/O pins default to input
- High-Speed I^{2}C Interface (**MCP23017**):
  - 100 kHz
  - 400 kHz
  - 1.7 MHz
- High-Speed SPI Interface (**MCP23S17**):
  - 10 MHz (maximum)
- Three Hardware Address Pins to Allow Up to Eight Devices On the Bus
- Configurable Interrupt Output Pins:
  - Configurable as active-high, active-low or open-drain
- INTA and INTB Can Be Configured to Operate Independently or Together
- Configurable Interrupt Source:
  - Interrupt-on-change from configured register defaults or pin changes
- Polarity Inversion Register to Configure the Polarity of the Input Port Data
- External Reset Input
- Low Standby Current: 1 µA (max.)
- Operating Voltage:
  - 1.8V to 5.5V @ -40°C to +85°C
  - 2.7V to 5.5V @ -40°C to +85°C
  - 4.5V to 5.5V @ -40°C to +125°C

Packages

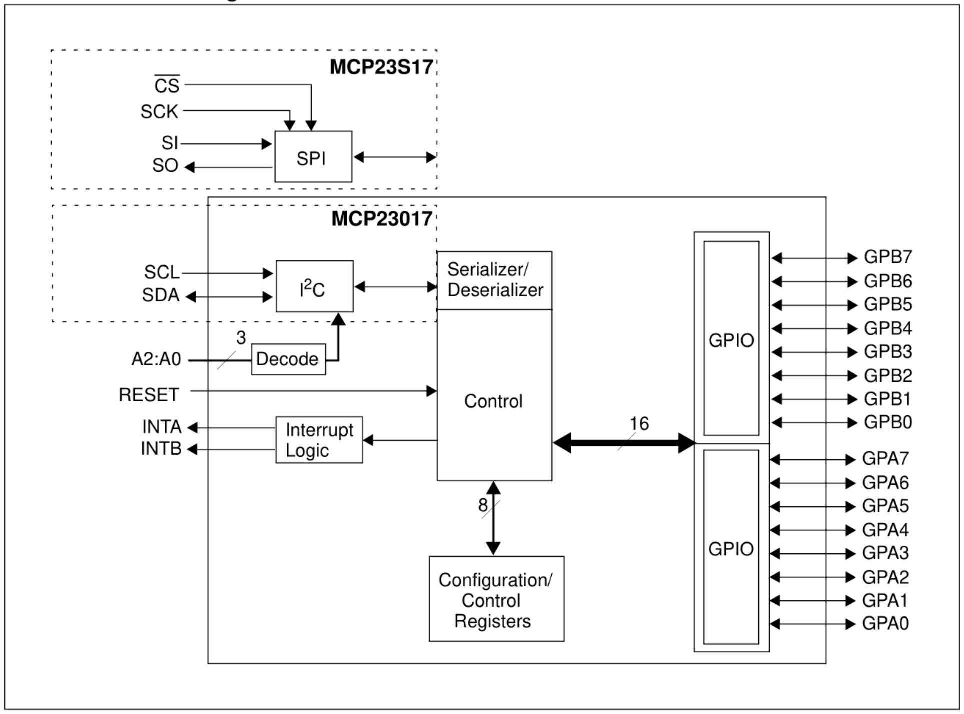
- 28-pin QFN, 6 x 6 mm Body
- 28-pin SOIC, Wide, 7.50 mm Body
- 28-pin SPDIP, 300 mil Body
- 28-pin SSOP, 5.30 mm Body

Package Types



* Includes Exposed Thermal Pad; see Table 2-1.

Functional Block Diagram



1.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

Ambient temperature under bias .... -40°C to +125°C
Storage temperature .... -65°C to +150°C
Voltage on V_{DD} with respect to V_{SS} .... -0.3V to +5.5V
Voltage on all other pins with respect to V_{SS} (except V_{DD}) .... -0.6V to (V_{DD} + 0.6V)
Total power dissipation .... 700 mW
Maximum current out of V_{SS} pin .... 150 mA
Maximum current into V_{DD} pin .... 125 mA
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) .... ±20 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{DD}) .... ±20 mA
Maximum output current sunk by any output pin .... 25 mA
Maximum output current sourced by any output pin .... 25 mA
ESD protection on all pins (HBM:MM) 4 kV:400V

† Notice: Stresses above those listed under “Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operational listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

1.1 DC Characteristics

TABLE 1-1: DC CHARACTERISTICS

Electrical Specifications: Unless otherwise noted, $1.8V \le V_{DD} \le 5.5V$ at $-40°C \le T_A \le +125°C$

Param. No.	Characteristic	Sym.	Min.	Typ.(1)	Max.	Units	Conditions
D001	Supply Voltage	V_{DD}	1.8	—	5.5	V	
D002	V_{DD} Start Voltage to ensure Power-on Reset	V_{POR}	—	V_{SS}	—	V	
D003	V_{DD} Rise Rate to ensure Power-on Reset	SV_{DD}	0.05	—	—	V/ms	Design guidance only. Not tested.
D004	Supply Current	I_{DD}	—	—	1	mA	SCL/SCK = 1 MHz
D005	Standby current	I_{DDS8}	—	—	1	µA	$-40°C \le T_A \le +85°C$
			—	—	3	µA	$4.5V \le V_{DD} \le 5.5V$ $+85°C \le T_A \le +125°C$ (**Note 1**)
Input Low Voltage							
D030	A0, A1, A2 (TTL buffer)	V_{IL}	V_{SS}	—	0.15 V_{DD}	V	
D031	$\overline{CS}$, GPIO, SCL/SCK, SDA, $\overline{RESET}$ (Schmitt Trigger)	V_{IL}	V_{SS}	—	0.2 V_{DD}	V	
Input High Voltage							
D040	A0, A1, A2 (TTL buffer)	V_{IH}	0.25 VDD + 0.8	—	V_{DD}	V	
D041	$\overline{CS}$, GPIO, SCL/SCK, SDA, $\overline{RESET}$ (Schmitt Trigger)	V_{IH}	0.8 VDD	—	V_{DD}	V	For entire V_{DD} range
Input Leakage Current							
D060	I/O port pins	I_{IL}	—	—	±1	µA	$V_{SS} \le V_{PIN} \le V_{DD}$
Output Leakage Current							
D065	I/O port pins	I_{LO}	—	—	±1	µA	$V_{SS} \le V_{PIN} \le V_{DD}$
D070	GPIO weak pull-up current	I_{PU}	40	75	115	µA	V_{DD} = 5V GP pins = V_{SS}
Output Low-Voltage							
D080	GPIO	V_{OL}	—	—	0.6	V	I_{OL} = 8.0 mA V_{DD} = 4.5V
	INT	V_{OL}	—	—	0.6	V	I_{OL} = 1.6 mA V_{DD} = 4.5V
	SO, SDA	V_{OL}	—	—	0.6	V	I_{OL} = 3.0 mA V_{DD} = 1.8V
	SDA	V_{OL}	—	—	0.8	V	I_{OL} = 3.0 mA V_{DD} = 4.5V
Output High-Voltage							
D090	GPIO, INT, SO	V_{OH}	V_{DD} – 0.7	—	—	V	I_{OH} = -3.0 mA V_{DD} = 4.5V
			V_{DD} – 0.7	—	—		I_{OH} = -400 µA V_{DD} = 1.8V
Capacitive Loading Specs on Output Pins							
D101	GPIO, SO, INT	C_{IO}	—	—	50	pF	
D102	SDA	C_B	—	—	400	pF	

Note 1: This parameter is characterized, not 100% tested.

1.2 AC Characteristics

FIGURE 1-1: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS

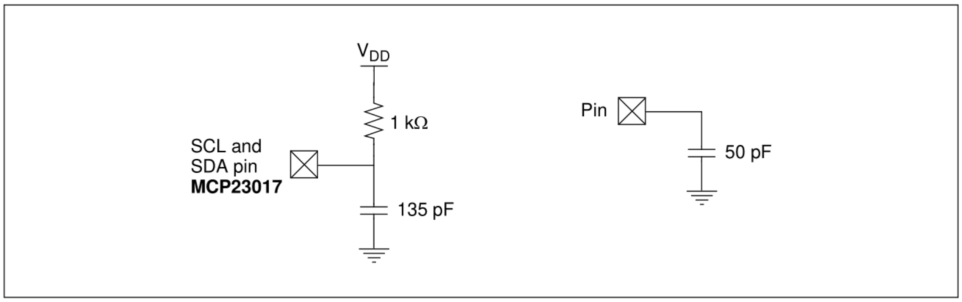


FIGURE 1-2: $\overline{\text{RESET}}$ AND DEVICE RESET TIMER TIMING

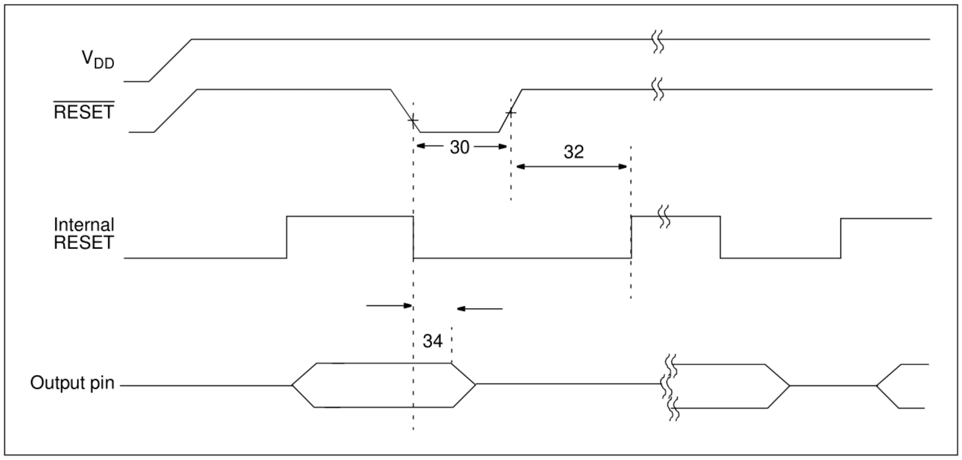


TABLE 1-2: DEVICE RESET SPECIFICATIONS

AC Characteristics: Unless otherwise noted, 1.8V ≤ V_{DD} ≤ 5.5V at -40°C ≤ T_A ≤ +125°C							
Param. No.	**Characteristic**	**Sym.**	**Min.**	**Typ. (1)**	**Max.**	**Units**	**Conditions**
30	$\overline{\text{RESET}}$ Pulse Width (Low)	T_{RSTL}	1	—	—	µs	
32	Device Active After Reset high	T_{HLD}	—	0	—	ns	V_{DD} = 5.0V
34	Output High-Impedance From $\overline{\text{RESET}}$ Low	T_{IOZ}	—	—	1	µs	

Note 1: This parameter is characterized, not 100% tested.

FIGURE 1-3: I²C BUS START/STOP BITS TIMING

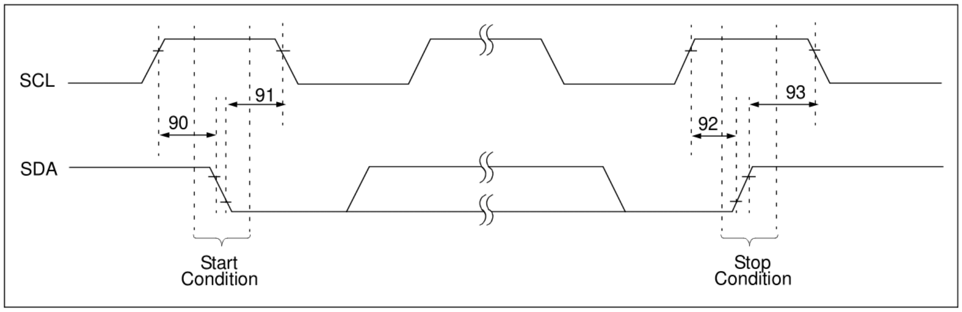


FIGURE 1-4: I²C BUS DATA TIMING

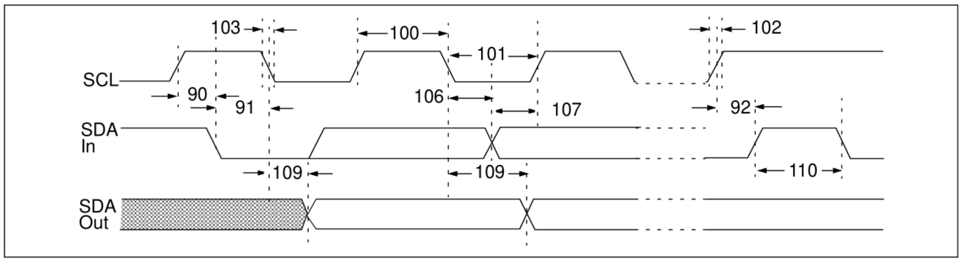


TABLE 1-3: I²C BUS DATA REQUIREMENTS

I²C Interface AC Characteristics: Unless otherwise noted, $1.8V \le V_{DD} \le 5.5V$ at $-40°C \le T_A \le +125°C$, R_{PU} (SCL, SDA) = 1 kΩ, C_L (SCL, SDA) = 135 pF

Param. No.	Characteristic	Sym.	Min.	Typ.	Max.	Units	Conditions
100	Clock High Time:	T_{HIGH}					
	100 kHz mode		4.0	—	—	µs	1.8V – 5.5V
	400 kHz mode		0.6	—	—	µs	2.7V – 5.5V
	1.7 MHz mode		0.12	—	—	µs	4.5V – 5.5V
101	Clock Low Time:	T_{LOW}					
	100 kHz mode		4.7	—	—	µs	1.8V – 5.5V
	400 kHz mode		1.3	—	—	µs	2.7V – 5.5V
	1.7 MHz mode		0.32	—	—	µs	4.5V – 5.5V
102	SDA and SCL Rise Time:	T_R (1)					
	100 kHz mode		—	—	1000	ns	1.8V – 5.5V
	400 kHz mode		20 + 0.1 C_B (2)	—	300	ns	2.7V – 5.5V
	1.7 MHz mode		20	—	160	ns	4.5V – 5.5V
103	SDA and SCL Fall Time:	T_F (1)					
	100 kHz mode		—	—	300	ns	1.8V – 5.5V
	400 kHz mode		20 + 0.1 C_B (2)	—	300	ns	2.7V – 5.5V
	1.7 MHz mode		20	—	80	ns	4.5V – 5.5V

Note 1: This parameter is characterized, not 100% tested.

2: C_B is specified to be from 10 to 400 pF.

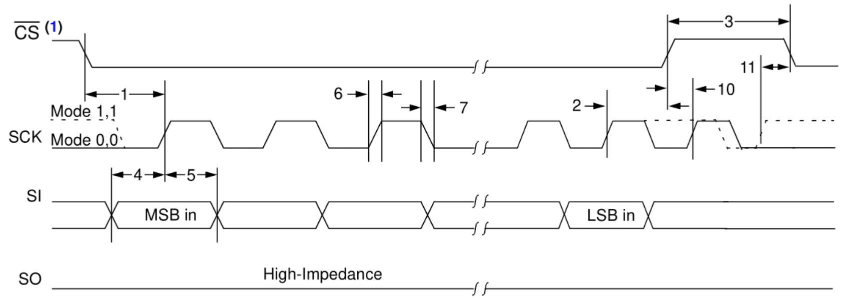
TABLE 1-3: I^2C BUS DATA REQUIREMENTS (CONTINUED)

I^2C Interface AC Characteristics: Unless otherwise noted, 1.8V ≤ V_{DD} ≤ 5.5V at -40°C ≤ T_A ≤ +125°C, R_{PU} (SCL, SDA) = 1 kΩ, C_L (SCL, SDA) = 135 pF

Param. No.	Characteristic	Sym.	Min.	Typ.	Max.	Units	Conditions
90	START Condition Setup Time:	$T_{SU:STA}$					
	100 kHz mode		4.7	—	—	µs	1.8V – 5.5V
	400 kHz mode		0.6	—	—	µs	2.7V – 5.5V
	1.7 MHz mode		0.16	—	—	µs	4.5V – 5.5V
91	START Condition Hold Time:	$T_{HD:STA}$					
	100 kHz mode		4.0	—	—	µs	1.8V – 5.5V
	400 kHz mode		0.6	—	—	µs	2.7V – 5.5V
	1.7 MHz mode		0.16	—	—	µs	4.5V – 5.5V
106	Data Input Hold Time:	$T_{HD:DAT}$					
	100 kHz mode		0	—	3.45	µs	1.8V – 5.5V
	400 kHz mode		0	—	0.9	µs	2.7V – 5.5V
	1.7 MHz mode		0	—	0.15	µs	4.5V – 5.5V
107	Data Input Setup Time:	$T_{SU:DAT}$					
	100 kHz mode		250	—	—	ns	1.8V – 5.5V
	400 kHz mode		100	—	—	ns	2.7V – 5.5V
	1.7 MHz mode		0.01	—	—	µs	4.5V – 5.5V
92	Stop Condition Setup Time:	$T_{SU:STO}$					
	100 kHz mode		4.0	—	—	µs	1.8V – 5.5V
	400 kHz mode		0.6	—	—	µs	2.7V – 5.5V
	1.7 MHz mode		0.16	—	—	µs	4.5V–5.5V
109	Output Valid From Clock:	T_{AA}					
	100 kHz mode		—	—	3.45	µs	1.8V – 5.5V
	400 kHz mode		—	—	0.9	µs	2.7V – 5.5V
	1.7 MHz mode		—	—	0.18	µs	4.5V – 5.5V
110	Bus Free Time:	T_{BUF}					
	100 kHz mode		4.7	—	—	µs	1.8V – 5.5V
	400 kHz mode		1.3	—	—	µs	2.7V – 5.5V
	1.7 MHz mode		N/A	—	N/A	µs	4.5V – 5.5V
111	Bus Capacitive Loading:	C_B					
	100 kHz and 400 kHz		—	—	400	pF	Note 1
	1.7 MHz		—	—	100	pF	Note 1
112	Input Filter Spike Suppression (SDA and SCL):	T_{SP}					
	100 kHz and 400 kHz		—	—	50	ns	
	1.7 MHz		—	—	10	ns	Spike suppression off

Note 1: This parameter is characterized, not 100% tested.
2: C_B is specified to be from 10 to 400 pF.

FIGURE 1-5: SPI INPUT TIMING



Note 1: When using SPI Mode 1,1 the $\overline{CS}$ pin needs to be toggled once before the first communication after power-up.

FIGURE 1-6: SPI OUTPUT TIMING

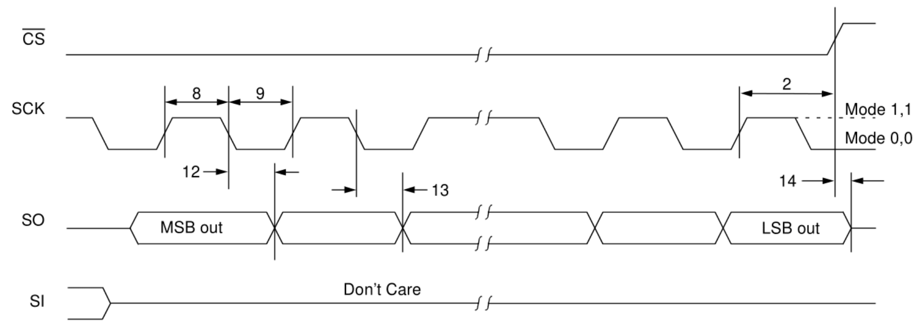


TABLE 1-4: SPI INTERFACE REQUIREMENTS

SPI Interface AC Characteristics: Unless otherwise noted, 1.8V ≤ V_{DD} ≤ 5.5V at -40°C ≤ T_A ≤ +125°C							
Param. No.	**Characteristic**	**Sym.**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
—	Clock Frequency	F_{CLK}	—	—	5	MHz	1.8V – 5.5V
			—	—	10	MHz	2.7V – 5.5V
			—	—	10	MHz	4.5V – 5.5V
1	$\overline{CS}$ Setup Time	T_{CSS}	50	—	—	ns	
2	$\overline{CS}$ Hold Time	T_{CSH}	100	—	—	ns	1.8V – 5.5V
			50	—	—	ns	2.7V – 5.5V
3	$\overline{CS}$ Disable Time	T_{CSD}	100	—	—	ns	1.8V – 5.5V
			50	—	—	ns	2.7V – 5.5V
4	Data Setup Time	T_{SU}	20	—	—	ns	1.8V – 5.5V
			10	—	—	ns	2.7V – 5.5V

Note 1: This parameter is characterized, not 100% tested.

TABLE 1-4: SPI INTERFACE REQUIREMENTS (CONTINUED)

SPI Interface AC Characteristics: Unless otherwise noted, 1.8V ≤ V_{DD} ≤ 5.5V at -40°C ≤ T_A ≤ +125°C							
Param. No.	**Characteristic**	**Sym.**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
5	Data Hold Time	T_{HD}	20	—	—	ns	1.8V – 5.5V
			10	—	—	ns	2.7V – 5.5V
6	CLK Rise Time	T_R	—	—	2	µs	Note 1
7	CLK Fall Time	T_F	—	—	2	µs	Note 1
8	Clock High Time	T_{HI}	90	—	—	ns	1.8V – 5.5V
			45	—	—	ns	2.7V – 5.5V
9	Clock Low Time	T_{LO}	90	—	—	ns	1.8V – 5.5V
			45	—	—	ns	2.7V – 5.5V
10	Clock Delay Time	T_{CLD}	50	—	—	ns	
11	Clock Enable Time	T_{CLE}	50	—	—	ns	
12	Output Valid from Clock Low	T_V	—	—	90	ns	1.8V – 5.5V
			—	—	45	ns	2.7V – 5.5V
13	Output Hold Time	T_{HO}	0	—	—	ns	
14	Output Disable Time	T_{DIS}	—	—	100	ns	

Note 1: This parameter is characterized, not 100% tested.

FIGURE 1-7: GPIO AND INT TIMING

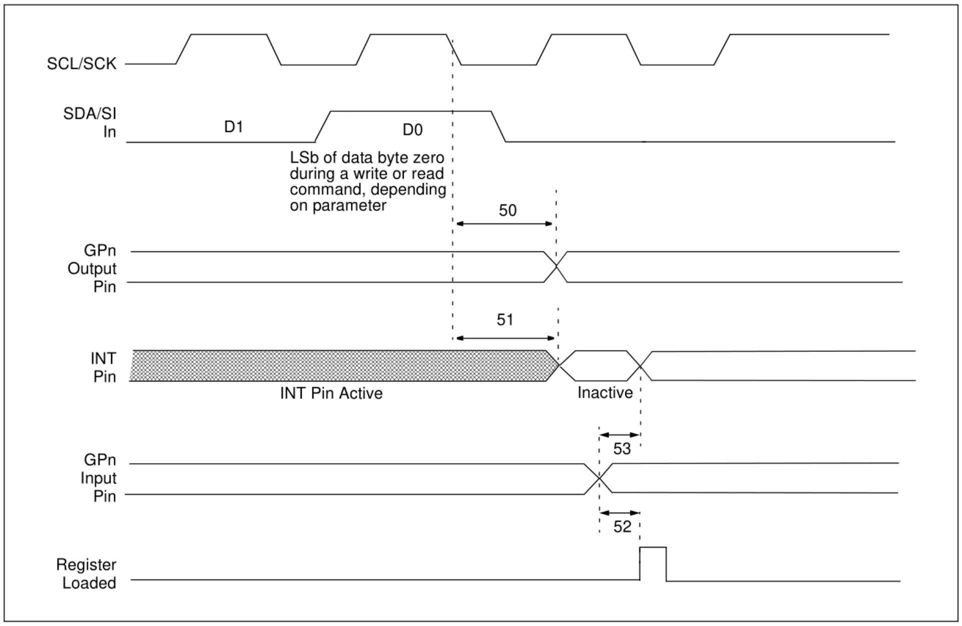


TABLE 1-5: GP AND INT PINS REQUIREMENTS

Param. No.	Characteristic	Sym.	Min.	Typ.	Max.	Units	Conditions
GP and INT Pins AC Characteristics: Unless otherwise noted, 1.8V ≤ V_{DD} ≤ 5.5V at -40°C ≤ T_A ≤ +125°C							
50	Serial Data to Output Valid	T_{GPOV}	—	—	500	ns	
51	Interrupt Pin Disable Time	T_{INTD}	—	—	600	ns	
52	GP Input Change to Register Valid	T_{GPIV}	—	—	450	ns	
53	IOC Event to INT Active	T_{GPINT}	—	—	600	ns	
	Glitch Filter on GP Pins	T_{GLITCH}	—	—	150	ns	Note 1

Note 1: This parameter is characterized, not 100% tested.

2.0 PIN DESCRIPTIONS

The descriptions of the pins are listed in Table 2-1.

TABLE 2-1: PINOUT DESCRIPTION

Pin Name	QFN	SOIC SPDIP SSOP	Pin Type	Function
GPB0	25	1	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPB1	26	2	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPB2	27	3	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPB3	28	4	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPB4	1	5	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPB5	2	6	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPB6	3	7	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPB7	4	8	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
V_{DD}	5	9	P	Power
V_{SS}	6	10	P	Ground
NC/$\overline{CS}$	7	11	I	NC (**MCP23017**)/Chip Select (**MCP23S17**)
SCK	8	12	I	Serial clock input
SDA/SI	9	13	I/O	Serial data I/O (**MCP23017**)/Serial data input (**MCP23S17**)
NC/SO	10	14	O	NC (**MCP23017**)/Serial data out (**MCP23S17**)
A0	11	15	I	Hardware address pin. Must be externally biased.
A1	12	16	I	Hardware address pin. Must be externally biased.
A2	13	17	I	Hardware address pin. Must be externally biased.
$\overline{RESET}$	14	18	I	Hardware reset. Must be externally biased.
INTB	15	19	O	Interrupt output for PORTB. Can be configured as active-high, active-low or open-drain.
INTA	16	20	O	Interrupt output for PORTA. Can be configured as active-high, active-low or open-drain.
GPA0	17	21	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPA1	18	22	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPA2	19	23	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPA3	20	24	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPA4	21	25	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPA5	22	26	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPA6	23	27	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
GPA7	24	28	I/O	Bidirectional I/O pin. Can be enabled for interrupt-on-change and/or internal weak pull-up resistor.
EP	29	—	—	Exposed Thermal Pad. Either connect to V_{SS}, or leave unconnected.

3.0 DEVICE OVERVIEW

The MCP23017/MCP23S17 (MCP23X17) device family provides 16-bit, general purpose parallel I/O expansion for I^2C bus or SPI applications. The two devices differ only in the serial interface:

- MCP23017 – I^2C interface
- MCP23S17 – SPI interface

The MCP23X17 consists of multiple 8-bit configuration registers for input, output and polarity selection. The system master can enable the I/Os as either inputs or outputs by writing the I/O configuration bits (IODIRA/B). The data for each input or output is kept in the corresponding input or output register. The polarity of the Input Port register can be inverted with the Polarity Inversion register. All registers can be read by the system master.

The 16-bit I/O port functionally consists of two 8-bit ports (PORTA and PORTB). The MCP23X17 can be configured to operate in the 8-bit or 16-bit modes via IOCON.BANK.

There are two interrupt pins, INTA and INTB, that can be associated with their respective ports, or can be logically OR'ed together so that both pins will activate if either port causes an interrupt.

The interrupt output can be configured to activate under two conditions (mutually exclusive):

1. When any input state differs from its corresponding Input Port register state. This is used to indicate to the system master that an input state has changed.
2. When an input state differs from a preconfigured register value (DEFVAL register).

The Interrupt Capture register captures port values at the time of the interrupt, thereby saving the condition that caused the interrupt.

The Power-on Reset (POR) sets the registers to their default values and initializes the device state machine.

The hardware address pins are used to determine the device address.

3.1 Power-on Reset (POR)

The on-chip POR circuit holds the device in reset until V_{DD} has reached a high enough voltage to deactivate the POR circuit (i.e., release the device from reset). The maximum V_{DD} rise time is specified in **Section 1.0 "Electrical Characteristics"**.

When the device exits the POR condition (releases reset), device operating parameters (i.e., voltage, temperature, serial bus frequency, etc.) must be met to ensure proper operation.

3.2 Serial Interface

This block handles the functionality of the I^2C (**MCP23017**) or SPI (**MCP23S17**) interface protocol. The MCP23X17 contains 22 individual registers (11 register pairs) that can be addressed through the Serial Interface block, as shown in Table 3-1.

TABLE 3-1: REGISTER ADDRESSES

Address IOCON.BANK = 1	Address IOCON.BANK = 0	Access to:
00h	00h	IODIRA
10h	01h	IODIRB
01h	02h	IPOLA
11h	03h	IPOLB
02h	04h	GPINTENA
12h	05h	GPINTENB
03h	06h	DEFVALA
13h	07h	DEFVALB
04h	08h	INTCONA
14h	09h	INTCONB
05h	0Ah	IOCON
15h	0Bh	IOCON
06h	0Ch	GPPUA
16h	0Dh	GPPUB
07h	0Eh	INTFA
17h	0Fh	INTFB
08h	10h	INTCAPA
18h	11h	INTCAPB
09h	12h	GPIOA
19h	13h	GPIOB
0Ah	14h	OLATA
1Ah	15h	OLATB

3.2.1 BYTE MODE AND SEQUENTIAL MODE

The MCP23X17 family has the ability to operate in Byte mode or Sequential mode (IOCON.SEQOP).

Byte mode disables automatic Address Pointer incrementing. When operating in Byte mode, the MCP23X17 family does not increment its internal address counter after each byte during the data transfer. This gives the ability to continually access the same address by providing extra clocks (without additional control bytes). This is useful for polling the GPIO register for data changes or for continually writing to the output latches.

A special mode **(Byte mode with IOCON.BANK = 0)** causes the address pointer to toggle between associated A/B register pairs. For example, if the BANK bit is cleared and the Address Pointer is initially set to address 12h (GPIOA) or 13h (GPIOB), the pointer will toggle between GPIOA and GPIOB. Note that the Address Pointer can initially point to either address in the register pair.

Sequential mode enables automatic address pointer incrementing. When operating in Sequential mode, the MCP23X17 family increments its address counter after each byte during the data transfer. The Address Pointer automatically rolls over to address 00h after accessing the last register.

These two modes are not to be confused with single writes/reads and continuous writes/reads that are serial protocol sequences. For example, the device may be configured for Byte mode and the master may perform a continuous read. In this case, the MCP23X17 would not increment the Address Pointer and would repeatedly drive data from the same location.

3.2.2 I^{2}C INTERFACE

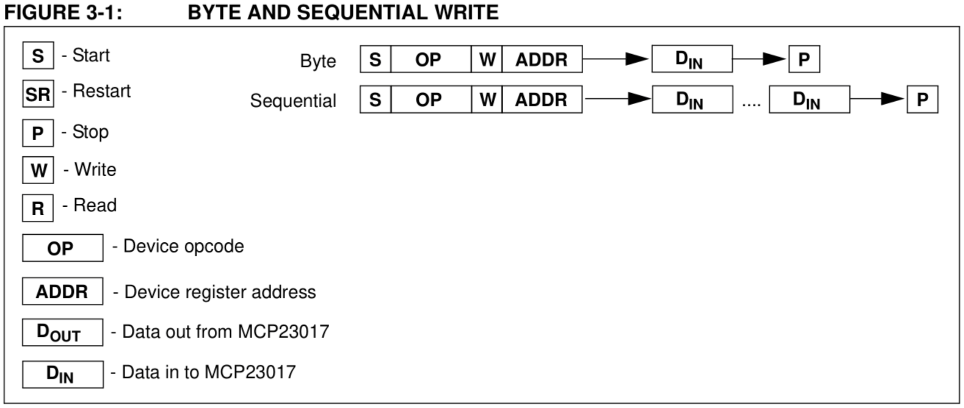
3.2.2.1 I^{2}C Write Operation

The I^{2}C write operation includes the control byte and register address sequence, as shown in Figure 3-1. This sequence is followed by eight bits of data from the master and an Acknowledge (ACK) from the MCP23017. The operation is ended with a Stop (P) or Restart (SR) condition being generated by the master.

Data is written to the MCP23017 after every byte transfer. If a Stop or Restart condition is generated during a data transfer, the data will not be written to the MCP23017.

Both “byte writes” and “sequential writes” are supported by the MCP23017. If Sequential mode is enabled (IOCON, SEQOP = 0) (default), the MCP23017 increments its address counter after each ACK during the data transfer.

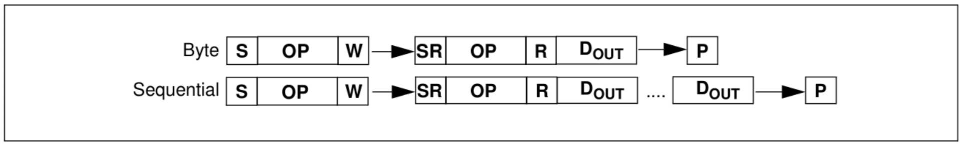
FIGURE 3-1: BYTE AND SEQUENTIAL WRITE



3.2.2.2 I²C Read Operation

I²C Read operations include the control byte sequence, as shown in Figure 3-2. This sequence is followed by another control byte (including the Start condition and ACK) with the R/W bit set (R/W = 1). The MCP23017 then transmits the data contained in the addressed register. The sequence is ended with the master generating a Stop or Restart condition.

FIGURE 3-2: BYTE AND SEQUENTIAL READ



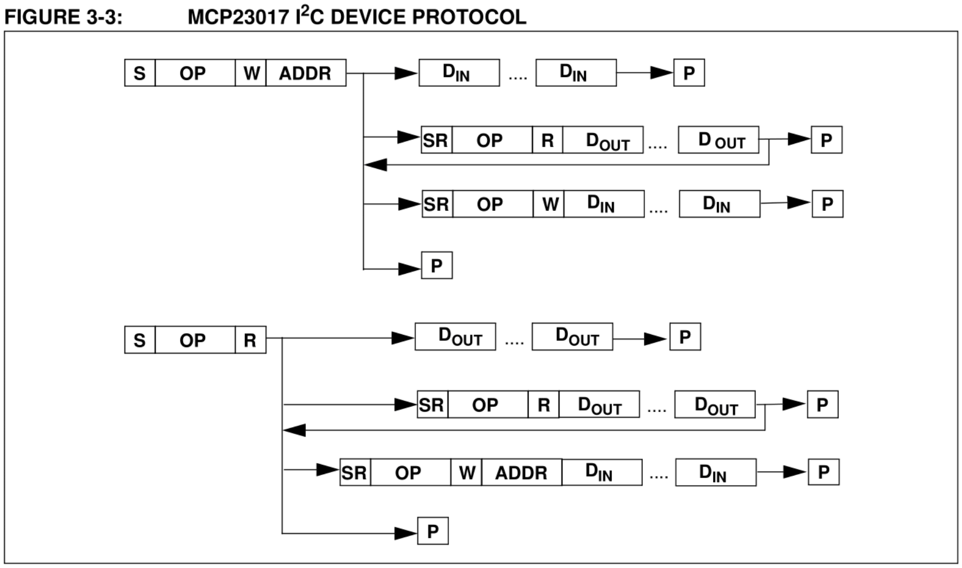
3.2.2.3 I²C Sequential Write/Read

For sequential operations (Write or Read), instead of transmitting a Stop or Restart condition after the data transfer, the master clocks the next byte pointed to by the address pointer (see **Section 3.2.1 "Byte Mode and Sequential Mode"** for details regarding sequential operation control).

The sequence ends with the master sending a Stop or Restart condition.

The MCP23017 Address Pointer will roll over to address zero after reaching the last register address. Refer to Figure 3-3.

FIGURE 3-3: MCP23017 I²C DEVICE PROTOCOL



3.2.3 SPI INTERFACE

3.2.3.1 SPI Write Operation

The SPI write operation is started by lowering $\overline{CS}$. The Write command (slave address with R/W bit cleared) is then clocked into the device. The opcode is followed by an address and at least one data byte.

3.2.3.2 SPI Read Operation

The SPI read operation is started by lowering $\overline{CS}$. The SPI read command (slave address with R/W bit set) is then clocked into the device. The opcode is followed by an address, with at least one data byte being clocked out of the device.

3.2.3.3 SPI Sequential Write/Read

For sequential operations, instead of deselecting the device by raising $\overline{CS}$, the master clocks the next byte pointed to by the Address Pointer. (see **Section 3.2.1 "Byte Mode and Sequential Mode"** for details regarding sequential operation control).

The sequence ends by the raising of $\overline{CS}$.

The MCP23S17 Address Pointer will roll over to address zero after reaching the last register address.

3.3 Hardware Address Decoder

The hardware address pins are used to determine the device address. To address a device, the corresponding address bits in the control byte must match the pin state. The pins must be biased externally.

3.3.1 ADDRESSING I^{2}C DEVICES (**MCP23017**)

The MCP23017 is a slave I^{2}C interface device that supports 7-bit slave addressing, with the read/write bit filling out the control byte. The slave address contains four fixed bits and three user-defined hardware address bits (pins A2, A1 and A0). Figure 3-4 shows the control byte format.

3.3.2 ADDRESSING SPI DEVICES (**MCP23S17**)

The MCP23S17 is a slave SPI device. The slave address contains four fixed bits and three user-defined hardware address bits (if enabled via IOCON.HAEN) (pins A2, A1 and A0) with the read/write bit filling out the control byte. Figure 3-5 shows the control byte format. The address pins should be externally biased even if disabled (IOCON.HAEN = 0).

FIGURE 3-4: I^{2}C CONTROL BYTE FORMAT

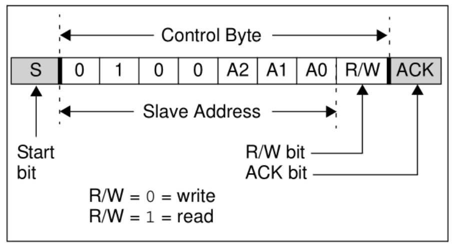


FIGURE 3-5: SPI CONTROL BYTE FORMAT

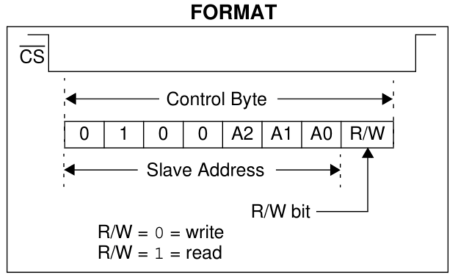


FIGURE 3-6: I^{2}C ADDRESSING REGISTERS

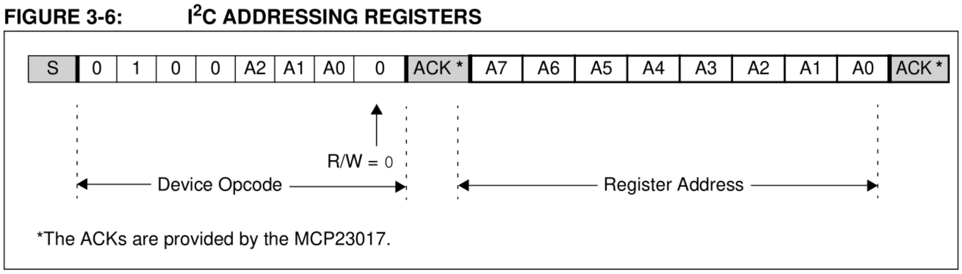
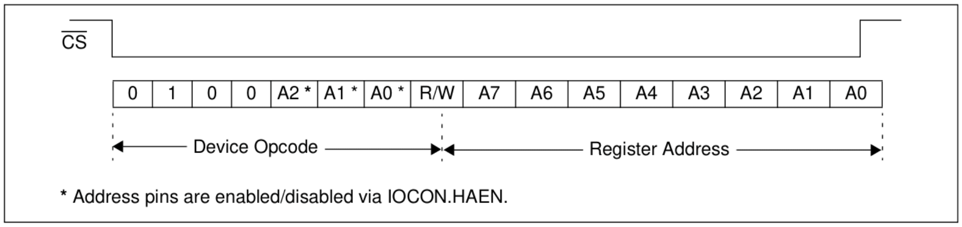


FIGURE 3-7: SPI ADDRESSING REGISTERS



3.4 GPIO Port

The GPIO module is a general purpose, 16-bit wide, bidirectional port that is functionally split into two 8-bit wide ports.

The GPIO module contains the data ports (GPIOn), internal pull-up resistors and the output latches (OLATn).

Reading the GPIOn register reads the value on the port. Reading the OLATn register only reads the latches, not the actual value on the port.

Writing to the GPIOn register actually causes a write to the latches (OLATn). Writing to the OLATn register forces the associated output drivers to drive to the level in OLATn. Pins configured as inputs turn off the associated output driver and put it in high-impedance.

TABLE 3-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE GPIO PORTS (BANK = 1)

Register Name	Address (hex)	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	POR/RST value
IODIRA	00	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IPOLA	01	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
GPINTENA	02	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
GPPUA	06	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000
GPIOA	09	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
OLATA	0A	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000
IODIRB	10	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IPOLB	11	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
GPINTENB	12	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
GPPUB	16	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000
GPIOB	19	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
OLATB	1A	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000

TABLE 3-3: SUMMARY OF REGISTERS ASSOCIATED WITH THE GPIO PORTS (BANK = 0)

Register Name	Address (hex)	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	POR/RST value
IODIRA	00	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IODIRB	01	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IPOLA	02	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
IPOLB	03	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
GPINTENA	04	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
GPINTENB	05	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
GPPUA	0C	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000
GPPUB	0D	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000
GPIOA	12	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
GPIOB	13	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
OLATA	14	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000
OLATB	15	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000

3.5 Configuration and Control Registers

There are 21 registers associated with the MCP23X17, as shown in Tables 3-4 and3-5. The two tables show the register mapping with the two BANK bit values. Ten registers are associated with PORTA and ten are associated with PORTB. One register (IOCON) is shared between the two ports. The PORTA registers are identical to the PORTB registers, therefore, they will be referred to without differentiating between the port designation (i.e., they will not have the “A” or “B” designator assigned) in the register tables.

TABLE 3-4: CONTROL REGISTER SUMMARY (IOCON.BANK = 1)

Register Name	Address (hex)	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	POR/RST value
IODIRA	00	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IPOLA	01	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
GPINTENA	02	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
DEFVALA	03	DEF7	DEF6	DEF5	DEF4	DEF3	DEF2	DEF1	DEF0	0000 0000
INTCONA	04	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0	0000 0000
IOCON	05	BANK	MIRROR	SEQOP	DISSLW	HAEN	ODR	INTPOL	—	0000 0000
GPPUA	06	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000
INTFA	07	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INTO	0000 0000
INTCAPA	08	ICP7	ICP6	ICP5	ICP4	ICP3	ICP2	ICP1	ICP0	0000 0000
GPIOA	09	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
OLATA	0A	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000
IODIRB	10	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IPOLB	11	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
GPINTENB	12	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
DEFVALB	13	DEF7	DEF6	DEF5	DEF4	DEF3	DEF2	DEF1	DEF0	0000 0000
INTCONB	14	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0	0000 0000
IOCON	15	BANK	MIRROR	SEQOP	DISSLW	HAEN	ODR	INTPOL	—	0000 0000
GPPUB	16	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000
INTFB	17	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INTO	0000 0000
INTCAPB	18	ICP7	ICP6	ICP5	ICP4	ICP3	ICP2	ICP1	ICP0	0000 0000
GPIOB	19	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
OLATB	1A	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000

TABLE 3-5: CONTROL REGISTER SUMMARY (IOCON.BANK = 0)

Register Name	Address (hex)	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	POR/RST value
IODIRA	00	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IODIRB	01	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0	1111 1111
IPOLA	02	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
IPOLB	03	IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0	0000 0000
GPINTENA	04	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
GPINTENB	05	GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0	0000 0000
DEFVALA	06	DEF7	DEF6	DEF5	DEF4	DEF3	DEF2	DEF1	DEF0	0000 0000
DEFVALB	07	DEF7	DEF6	DEF5	DEF4	DEF3	DEF2	DEF1	DEF0	0000 0000
INTCONA	08	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0	0000 0000
INTCONB	09	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0	0000 0000
IOCON	0A	BANK	MIRROR	SEQOP	DISSLW	HAEN	ODR	INTPOL	—	0000 0000
IOCON	0B	BANK	MIRROR	SEQOP	DISSLW	HAEN	ODR	INTPOL	—	0000 0000
GPPUA	0C	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000
GPPUB	0D	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0	0000 0000

TABLE 3-5: CONTROL REGISTER SUMMARY (IOCON.BANK = 0) (CONTINUED)

Register Name	Address (hex)	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	POR/RST value
INTFA	0E	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INTO	0000 0000
INTFB	0F	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INTO	0000 0000
INTCAPA	10	ICP7	ICP6	ICP5	ICP4	ICP3	ICP2	ICP1	ICP0	0000 0000
INTCAPB	11	ICP7	ICP6	ICP5	ICP4	ICP3	ICP2	ICP1	ICP0	0000 0000
GPIOA	12	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
GPIOB	13	GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0	0000 0000
OLATA	14	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000
OLATB	15	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0	0000 0000

3.5.1 I/O DIRECTION REGISTER

Controls the direction of the data I/O.

When a bit is set, the corresponding pin becomes an input. When a bit is clear, the corresponding pin becomes an output.

REGISTER 3-1: IODIR: I/O DIRECTION REGISTER (ADDR 0x00)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **IO<7:0>:** Controls the direction of data I/O <7:0>
1 = Pin is configured as an input.
0 = Pin is configured as an output.

3.5.2 INPUT POLARITY REGISTER

This register allows the user to configure the polarity on the corresponding GPIO port bits.

If a bit is set, the corresponding GPIO register bit will reflect the inverted value on the pin.

REGISTER 3-2: IPOL: INPUT POLARITY PORT REGISTER (ADDR 0x01)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IP7	IP6	IP5	IP4	IP3	IP2	IP1	IP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **IP<7:0>:** Controls the polarity inversion of the input pins <7:0>
1 = GPIO register bit reflects the opposite logic state of the input pin.
0 = GPIO register bit reflects the same logic state of the input pin.

3.5.3 INTERRUPT-ON-CHANGE CONTROL REGISTER

The GPINTEN register controls the interrupt-on-change feature for each pin.

If a bit is set, the corresponding pin is enabled for interrupt-on-change. The DEFVAL and INTCON registers must also be configured if any pins are enabled for interrupt-on-change.

REGISTER 3-3: GPINTEN: INTERRUPT-ON-CHANGE PINS (ADDR 0x02) (Note 1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GPINT7	GPINT6	GPINT5	GPINT4	GPINT3	GPINT2	GPINT1	GPINT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **GPINT<7:0>:** General purpose I/O interrupt-on-change bits <7:0>
1 = Enables GPIO input pin for interrupt-on-change event.
0 = Disables GPIO input pin for interrupt-on-change event.

Note 1: Refer to INTCON.

3.5.4 DEFAULT COMPARE REGISTER FOR INTERRUPT-ON-CHANGE

The default comparison value is configured in the DEFVAL register. If enabled (via GPINTEN and INTCON) to compare against the DEFVAL register, an opposite value on the associated pin will cause an interrupt to occur.

REGISTER 3-4: DEFVAL: DEFAULT VALUE REGISTER (ADDR 0x03)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DEF7	DEF6	DEF5	DEF4	DEF3	DEF2	DEF1	DEF0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **DEF<7:0>:** Sets the compare value for pins configured for interrupt-on-change from defaults <7:0> **(Note 1)**
If the associated pin level is the opposite from the register bit, an interrupt occurs. **(Note 2)**

Note 1: Refer to INTCON.
2: Refer to INTCON and GPINTEN.

3.5.5 INTERRUPT CONTROL REGISTER

The INTCON register controls how the associated pin value is compared for the interrupt-on-change feature. If a bit is set, the corresponding I/O pin is compared against the associated bit in the DEFVAL register. If a bit value is clear, the corresponding I/O pin is compared against the previous value.

REGISTER 3-5: INTCON: INTERRUPT-ON-CHANGE CONTROL REGISTER (ADDR 0x04) (Note 1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **IOC<7:0>:** Controls how the associated pin value is compared for interrupt-on-change <7:0>
1 = Pin value is compared against the associated bit in the DEFVAL register.
0 = Pin value is compared against the previous pin value.

Note 1: Refer to INTCON and GPINTEN.

3.5.6 CONFIGURATION REGISTER

The IOCON register contains several bits for configuring the device:

The BANK bit changes how the registers are mapped (see Tables 3-4 and3-5 for more details).

- If BANK = 1, the registers associated with each port are segregated. Registers associated with PORTA are mapped from address 00h - 0Ah and registers associated with PORTB are mapped from 10h - 1Ah.
- If BANK = 0, the A/B registers are paired. For example, IODIRA is mapped to address 00h and IODIRB is mapped to the next address (address 01h). The mapping for all registers is from 00h -15h.

It is important to take care when changing the BANK bit as the address mapping changes after the byte is clocked into the device. The address pointer may point to an invalid location after the bit is modified.

For example, if the device is configured to automatically increment its internal Address Pointer, the following scenario would occur:

- BANK = 0
- Write 80h to address 0Ah (IOCON) to set the BANK bit
- Once the write completes, the internal address now points to 0Bh which is an invalid address when the BANK bit is set.

For this reason, when changing the BANK bit, it is advised to only perform byte writes to this register.

The **MIRROR** bit controls how the INTA and INTB pins function with respect to each other.

- When MIRROR = 1, the INTn pins are functionally OR’ed so that an interrupt on either port will cause both pins to activate.
- When MIRROR = 0, the INT pins are separated. Interrupt conditions on a port will cause its respective INT pin to activate.

The Sequential Operation (**SEQOP**) controls the incrementing function of the Address Pointer. If the address pointer is disabled, the Address Pointer does not automatically increment after each byte is clocked during a serial transfer. This feature is useful when it is desired to continuously poll (read) or modify (write) a register.

The Slew Rate (**DISSLW**) bit controls the slew rate function on the SDA pin. If enabled, the SDA slew rate will be controlled when driving from a high to low.

The Hardware Address Enable (**HAEN**) bit enables/disables hardware addressing on the MCP23S17 only. The address pins (A2, A1 and A0) must be externally biased, regardless of the HAEN bit value.

If enabled (HAEN = 1), the device’s hardware address matches the address pins.

If disabled (HAEN = 0), the device’s hardware address is A2 = A1 = A0 = 0.

The Open-Drain (**ODR**) control bit enables/disables the INT pin for open-drain configuration. Setting this bit overrides the INTPOL bit.

The Interrupt Polarity (**INTPOL**) sets the polarity of the INT pin. This bit is functional only when the ODR bit is cleared, configuring the INT pin as active push-pull.

REGISTER 3-6: IOCON: I/O EXPANDER CONFIGURATION REGISTER (ADDR 0x05)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
BANK	MIRROR	SEQOP	DISSLW	HAEN	ODR	INTPOL	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **BANK:** Controls how the registers are addressed
1 = The registers associated with each port are separated into different banks.
0 = The registers are in the same bank (addresses are sequential).

bit 6 **MIRROR:** INT Pins Mirror bit
1 = The INT pins are internally connected
0 = The INT pins are not connected. INTA is associated with PORTA and INTB is associated with PORTB

bit 5 **SEQOP:** Sequential Operation mode bit
1 = Sequential operation disabled, address pointer does not increment.
0 = Sequential operation enabled, address pointer increments.

bit 4 **DISSLW:** Slew Rate control bit for SDA output
1 = Slew rate disabled
0 = Slew rate enabled

bit 3 **HAEN:** Hardware Address Enable bit (**MCP23S17** only) (**Note 1**)
1 = Enables the MCP23S17 address pins.
0 = Disables the MCP23S17 address pins.

bit 2 **ODR:** Configures the INT pin as an open-drain output
1 = Open-drain output (overrides the INTPOL bit.)
0 = Active driver output (INTPOL bit sets the polarity.)

bit 1 **INTPOL:** This bit sets the polarity of the INT output pin
1 = Active-high
0 = Active-low

bit 0 **Unimplemented:** Read as ‘0’

Note 1: Address pins are always enabled on the MCP23017.

3.5.7 PULL-UP RESISTOR CONFIGURATION REGISTER

The GPPU register controls the pull-up resistors for the port pins. If a bit is set and the corresponding pin is configured as an input, the corresponding port pin is internally pulled up with a 100 kΩ resistor.

REGISTER 3-7: GPPU: GPIO PULL-UP RESISTOR REGISTER (ADDR 0x06)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **PU<7:0>** Controls the weak pull-up resistors on each pin (when configured as an input)
1 = Pull-up enabled
0 = Pull-up disabled

3.5.8 INTERRUPT FLAG REGISTER

The INTF register reflects the interrupt condition on the port pins of any pin that is enabled for interrupts via the GPINTEN register. A set bit indicates that the associated pin caused the interrupt.

This register is read-only. Writes to this register will be ignored.

REGISTER 3-8: INTF: INTERRUPT FLAG REGISTER (ADDR 0x07)

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **INT<7:0>:** Reflects the interrupt condition on the port. It reflects the change only if interrupts are enabled per GPINTEN<7:0>.
1 = Pin caused interrupt.
0 = Interrupt not pending

3.5.9 INTERRUPT CAPTURED REGISTER

The INTCAP register captures the GPIO port value at the time the interrupt occurred. The register is read-only and is updated only when an interrupt occurs. The register remains unchanged until the interrupt is cleared via a read of INTCAP or GPIO.

REGISTER 3-9: INTCAP: INTERRUPT CAPTURED VALUE FOR PORT REGISTER (ADDR 0x08)

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ICP7	ICP6	ICP5	ICP4	ICP3	ICP2	ICP1	ICP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **ICP<7:0>:** Reflects the logic level on the port pins at the time of interrupt due to pin change <7:0>
1 = Logic-high
0 = Logic-low

3.5.10 PORT REGISTER

The GPIO register reflects the value on the port. Reading from this register reads the port. Writing to this register modifies the Output Latch (OLAT) register.

REGISTER 3-10: GPIO: GENERAL PURPOSE I/O PORT REGISTER (ADDR 0x09)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GP7	GP6	GP5	GP4	GP3	GP2	GP1	GP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **GP<7:0>:** Reflects the logic level on the pins <7:0>
1 = Logic-high
0 = Logic-low

3.5.11 OUTPUT LATCH REGISTER (OLAT)

The OLAT register provides access to the output latches. A read from this register results in a read of the OLAT and not the port itself. A write to this register modifies the output latches that modifies the pins configured as outputs.

REGISTER 3-11: OLAT: OUTPUT LATCH REGISTER 0 (ADDR 0x0A)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **OL<7:0>:** Reflects the logic level on the output latch <7:0>
`1` = Logic-high
`0` = Logic-low

3.6 Interrupt Logic

If enabled, the MCP23X17 activates the INTn interrupt output when one of the port pins changes state or when a pin does not match the preconfigured default. Each pin is individually configurable as follows:

- Enable/disable interrupt via GPINTEN
- Can interrupt on either pin change or change from default as configured in DEFVAL

Both conditions are referred to as Interrupt-on-Change (IOC).

The interrupt control module uses the following registers/bits:

- IOCON.MIRROR – controls if the two interrupt pins mirror each other
- GPINTEN – Interrupt enable register
- INTCON – controls the source for the IOC
- DEFVAL – contains the register default for IOC operation

3.6.1 INTA AND INTB

There are two interrupt pins: INTA and INTB. By default, INTA is associated with GPAn pins (PORTA) and INTB is associated with GPBn pins (PORTB). Each port has an independent signal which is cleared if its associated GPIO or INTCAP register is read.

3.6.1.1 Mirroring the INT pins

Additionally, the INTn pins can be configured to mirror each other so that any interrupt will cause both pins to go active. This is controlled via IOCON.MIRROR.

If IOCON.MIRROR = `0`, the internal signals are routed independently to the INTA and INTB pads.

If IOCON.MIRROR = `1`, the internal signals are OR’ed together and routed to the INTn pads. In this case, the interrupt will only be cleared if the associated GPIO or INTCAP is read (see Table 3-6).

TABLE 3-6: INTERRUPT OPERATION (IOCON.MIRROR = 1)

Interrupt Condition	Read PORTn [1]	Interrupt Result
GPIOA	PORTA	Clear
	PORTB	Unchanged
GPIOB	PORTA	Unchanged
	PORTB	Clear
GPIOA and GPIOB	PORTA	Unchanged
	PORTB	Unchanged
	Both PORTA and PORTB	Clear

Note 1: PORTn = GPIOn or INTCAPn

3.6.2 IOC FROM PIN CHANGE

If enabled, the MCP23X17 generates an interrupt if a mismatch condition exists between the current port value and the previous port value. Only IOC-enabled pins will be compared. Refer to Registers 3-3 and 3-5.

3.6.3 IOC FROM REGISTER DEFAULT

If enabled, the MCP23X17 generates an interrupt if a mismatch occurs between the DEFVAL register and the port. Only IOC enabled pins are compared. Refer to Registers 3-3, 3-4 and 3-5.

3.6.4 INTERRUPT OPERATION

The INTn interrupt output can be configured as active-low, active-high or open-drain via the IOCON register.

Only those pins that are configured as an input (IODIR register) with Interrupt-On-Change (IOC) enabled (IOINTEN register) can cause an interrupt. Pins defined as an output have no effect on the interrupt output pin.

Input change activity on a port input pin that is enabled for IOC generates an internal device interrupt and the device captures the value of the port and copies it into INTCAP. The interrupt remains active until the INTCAP or GPIO register is read. Writing to these registers does not affect the interrupt. The interrupt condition is cleared after the LSb of the data is clocked out during a read command of GPIO or INTCAP.

The first interrupt event causes the port contents to be copied into the INTCAP register. Subsequent interrupt conditions on the port will not cause an interrupt to occur as long as the interrupt is not cleared by a read of INTCAP or GPIO.

Note: The value in INTCAP can be lost if GPIO is read before INTCAP while another IOC is pending. After reading GPIO, the interrupt will clear and then set due to the pending IOC, causing the INTCAP register to update.

3.6.5 INTERRUPT CONDITIONS

There are two possible configurations that cause interrupts (configured via INTCON):

1. Pins configured for **interrupt-on-pin change** will cause an interrupt to occur if a pin changes to the opposite state. The default state is reset after an interrupt occurs and after clearing the interrupt condition (i.e., after reading GPIO or INTCAP). For example, an interrupt occurs by an input changing from '1' to '0'. The new initial state for the pin is a logic '0' after the interrupt is cleared.
2. Pins configured for **interrupt-on-change from register value** will cause an interrupt to occur if the corresponding input pin differs from the register bit. The interrupt condition will remain as long as the condition exists, regardless if the INTCAP or GPIO is read.

See Figures 3-8 and 3-9 for more information on interrupt operations.

FIGURE 3-8: INTERRUPT-ON-PIN CHANGE

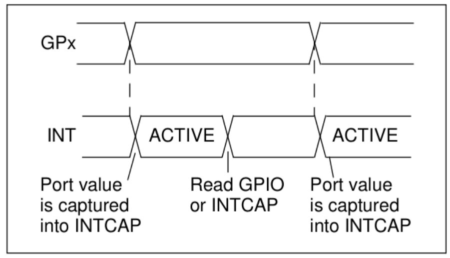
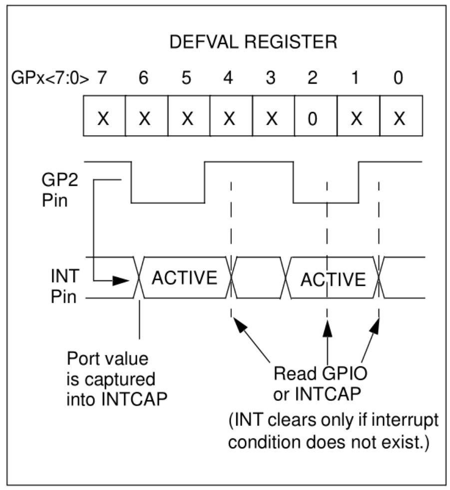


FIGURE 3-9: INTERRUPT-ON-CHANGE FROM REGISTER DEFAULT



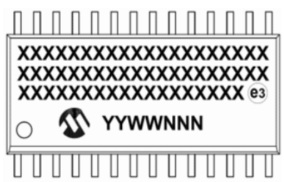
4.0 PACKAGING INFORMATION

4.1 Package Marking Information

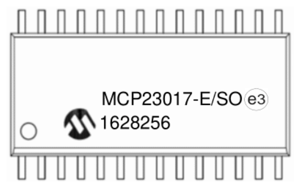
28-Lead QFN

Example:

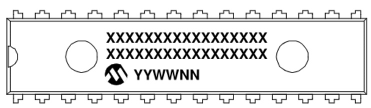
28-Lead SOIC



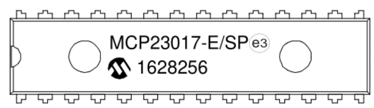
Example:



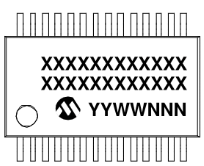
28-Lead SPDIP



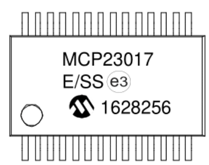
Example:



28-Lead SSOP



Example:

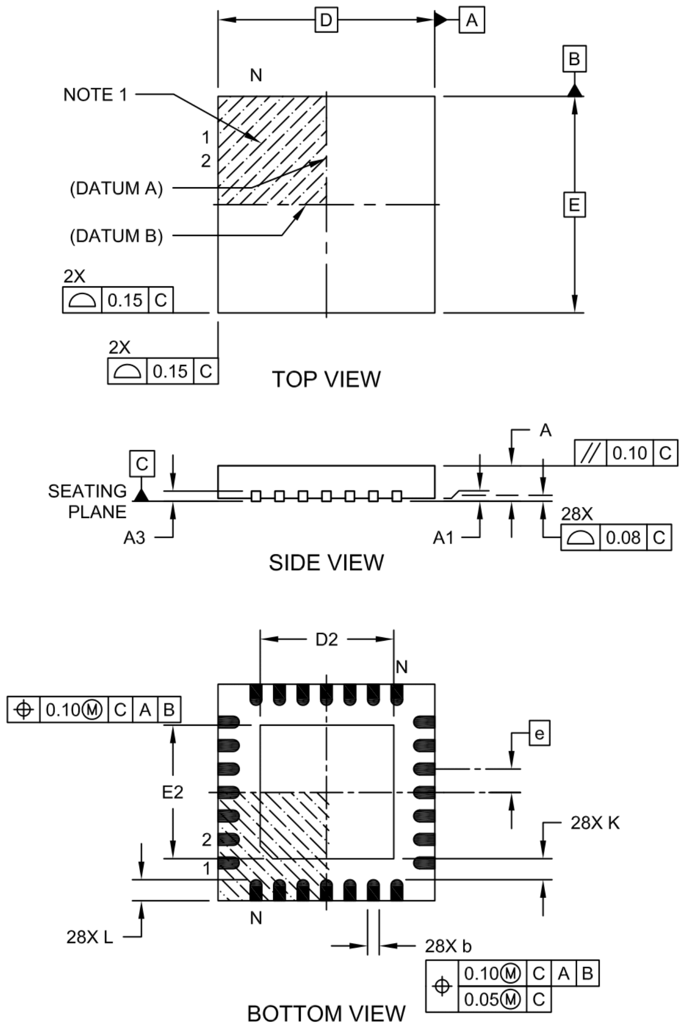


Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	e3	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

28-Lead Plastic Quad Flat, No Lead Package (ML) - 6x6 mm Body [QFN]
With 0.55 mm Terminal Length

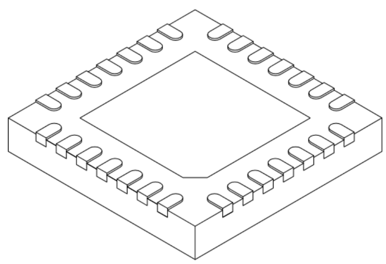
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-105C Sheet 1 of 2

28-Lead Plastic Quad Flat, No Lead Package (ML) - 6x6 mm Body [QFN] With 0.55 mm Terminal Length

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



	Units	MILLIMETERS		
Dimension	Limits	MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.20 REF	
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2	3.65	3.70	4.20
Terminal Width	b	0.23	0.30	0.35
Terminal Length	L	0.50	0.55	0.70
Terminal-to-Exposed Pad	K	0.20	-	-

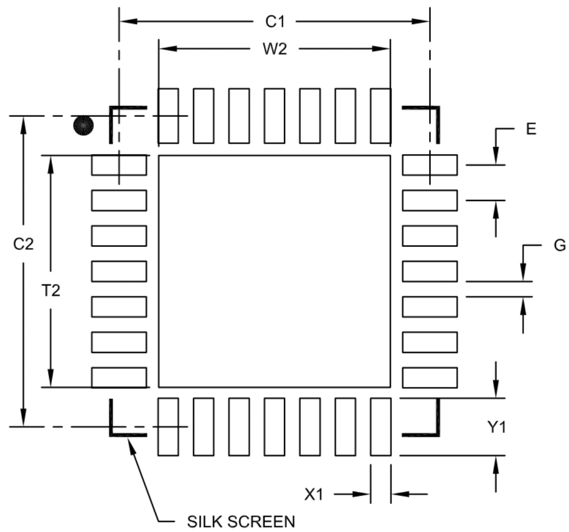
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-105C Sheet 2 of 2

28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
Distance Between Pads	G	0.20		

Notes:

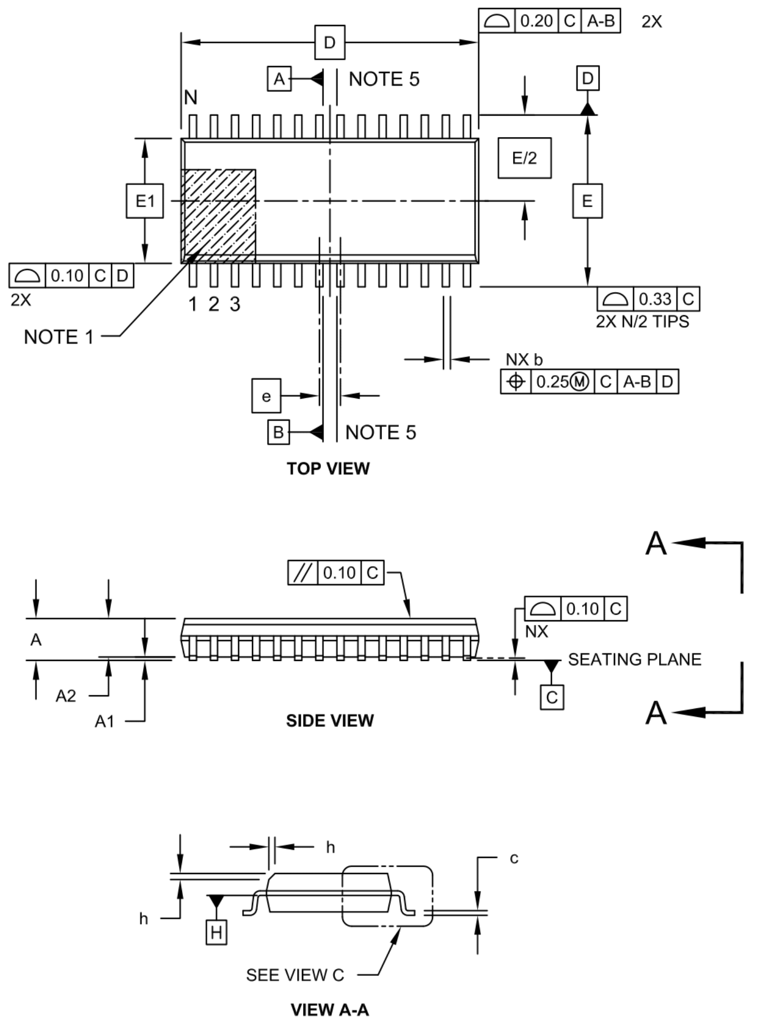
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

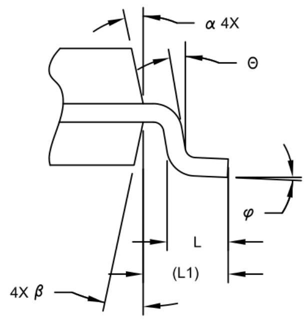
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

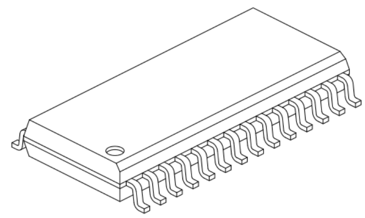


Microchip Technology Drawing C04-052C Sheet 1 of 2

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging





Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		1.27 BSC	
Overall Height	A	-	-	2.65
Molded Package Thickness	A2	2.05	-	-
Standoff §	A1	0.10	-	0.30
Overall Width	E		10.30 BSC	
Molded Package Width	E1		7.50 BSC	
Overall Length	D		17.90 BSC	
Chamfer (Optional)	h	0.25	-	0.75
Foot Length	L	0.40	-	1.27
Footprint	L1		1.40 REF	
Lead Angle	Θ	0°	-	-
Foot Angle	φ	0°	-	8°
Lead Thickness	c	0.18	-	0.33
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

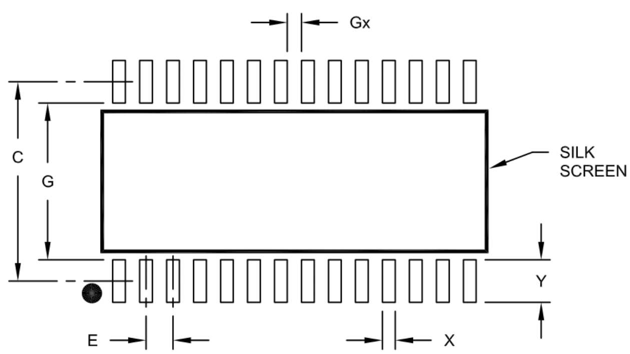
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.
5. Datums A & B to be determined at Datum H.

Microchip Technology Drawing C04-052C Sheet 2 of 2

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

		Units	MILLIMETERS		
		Dimension Limits	MIN	NOM	MAX
Contact Pitch	E		1.27 BSC		
Contact Pad Spacing	C			9.40	
Contact Pad Width (X28)	X				0.60
Contact Pad Length (X28)	Y				2.00
Distance Between Pads	Gx		0.67		
Distance Between Pads	G		7.40		

Notes:

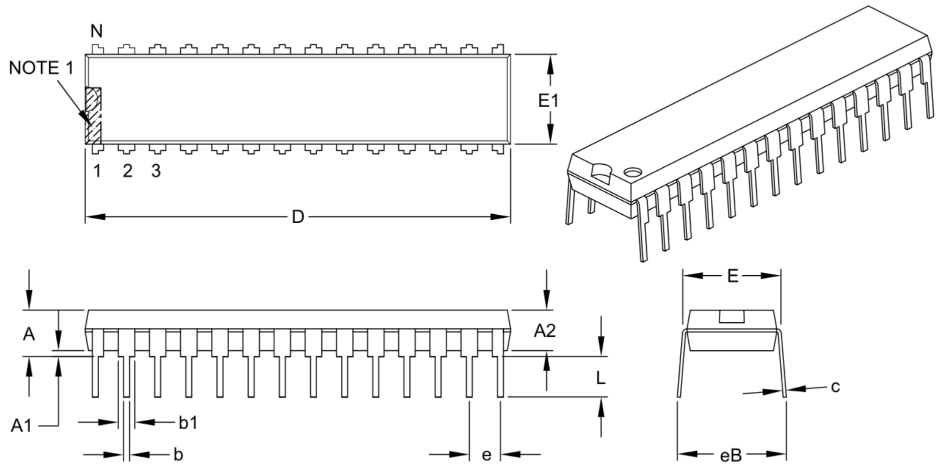
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2052A

28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



		Units	INCHES		
		Dimension Limits	MIN	NOM	MAX
Number of Pins	N			28	
Pitch	e			.100 BSC	
Top to Seating Plane	A		–	–	.200
Molded Package Thickness	A2		.120	.135	.150
Base to Seating Plane	A1		.015	–	–
Shoulder to Shoulder Width	E		.290	.310	.335
Molded Package Width	E1		.240	.285	.295
Overall Length	D		1.345	1.365	1.400
Tip to Seating Plane	L		.110	.130	.150
Lead Thickness	c		.008	.010	.015
Upper Lead Width	b1		.040	.050	.070
Lower Lead Width	b		.014	.018	.022
Overall Row Spacing §	eB		–	–	.430

Notes:

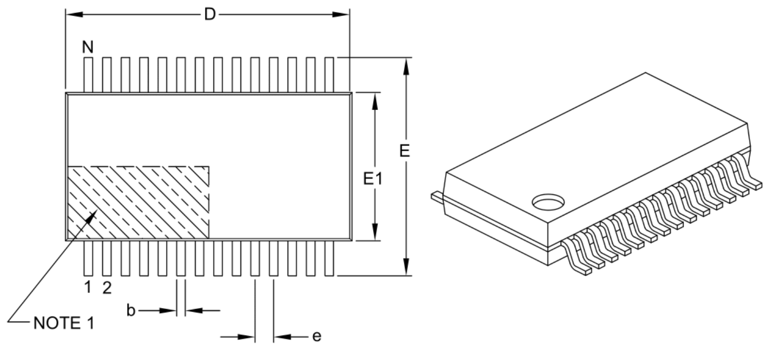
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

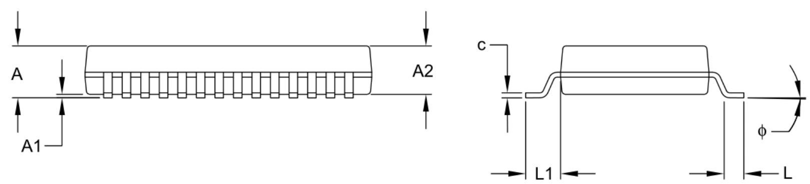
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

28-Lead Plastic Shrink Small Outline (SS) – 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging





	Units	MILLIMETERS		
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	0.65 BSC		
Overall Height	A	–	–	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	–	–
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1	1.25 REF		
Lead Thickness	c	0.09	–	0.25
Foot Angle	φ	0°	4°	8°
Lead Width	b	0.22	–	0.38

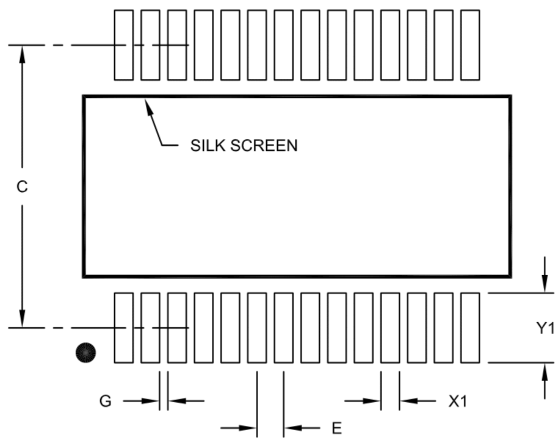
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Contact Pad Spacing	C		7.20	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.75
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2073A

NOTES:

APPENDIX A: REVISION HISTORY

Revision C (July 2016)

The following is the list of modifications:

1. Added ESD data to **Section 1.0, Electrical Characteristics**.
2. Updated Table 2-1.
3. Updated package outline drawings.
4. Minor typographical errors

Revision B (February 2007)

1. Changed Byte and Sequential Read in Figure 1-1 from “R” to “W”.
2. Table 2-4, Param No. 51 and 53: Changed from 450 to 600 and 500 to 600, respecively.
3. Added disclaimers to package outline drawings.
4. Updated package outline drawings.

Revision A (June 2005)

- Original release of this document.

NOTES:

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	– X	/XX	X (1)
Device	Temperature Range	Package	Tape and Reel Option

Device:	MCP23017:	16-Bit I/O Expander with I^2C Interface
	MCP23S17:	16-Bit I/O Expander with SPI Interface
Temperature Range:	E =	-40°C to +125°C (Extended)
Package:	ML =	Plastic Quad Flat, No Lead Package, 6x6 mm Body, QFN, 28-lead
	SO =	Plastic Small Outline, Wide, 7.50 mm Body, SOIC, 28-Lead
	SP =	Skinny Plastic Dual In-Line, 300 mil Body, SPDIP, 28-Lead
	SS =	Plastic Shrink Small Outline, 5.30 mm Body, SSOP, 28-Lead
Tape and Reel Option:	T =	Tape and Reel (1)
	Blank =	Tube

Examples:

a) MCP23017-E/ML: Extended temperature, 28LD QFN package
b) MCP23017T-E/ML: Extended temperature, 28LD QFN package, Tape and Reel
c) MCP23017-E/SP: Extended temperature , 28LD SPDIP package
d) MCP23017-E/SO: Extended temperature, 28LD SOIC package
e) MCP23017T-E/SO: Extended temperature, 28LD SOIC package, Tape and Reel
f) MCP23017-E/SS: Extended temperature, 28LD SSOP package
g) MCP23017T-E/SS: Extended temperature, 28LD SSOP package, Tape and Reel

a) MCP23S17-E/ML: Extended temperature, 28LD QFN package
b) MCP23S17T-E/ML: Extended temperature, 28LD QFN package, Tape and Reel
c) MCP23S17-E/SP: Extended temperature, 28LD SPDIP package
d) MCP23S17-E/SO: Extended temperature, 28LD SOIC package
e) MCP23S17T-E/SO: Extended temperature, 28LD SOIC package, Tape and Reel
f) MCP23S17-E/SS: Extended temperature, 28LD SSOP package
g) MCP23S17T-E/SS: Extended temperature, 28LD SSOP package Tape and Reel

Note 1: Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
== ISO/TS 16949 ==

Trademarks

ISBN: 978-1-5224-0755-3

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7828

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Dusseldorf
Tel: 49-2129-3766400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Venice
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Poland - Warsaw
Tel: 48-22-3325737

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

06/23/16

A.5 SE-600-12 Power Supply

MEAN WELL

600W Single Output Power Supply **SE-600 series**

■ Features :

- AC input active surge current limiting
- AC input range selected by switch
- Protections:Short circuit / Overload / Over voltage / Over temperature
- Forced air cooling by built-in DC fan
- High power density 4.9w/in^3
- Built-in remote sense function
- UL approved
- Low cost
- 2 years warranty

K60950-1 (for 12V,24V only) R33100 RoHS (Note.5)

SPECIFICATION

MODEL		SE-600-5	SE-600-12	SE-600-15	SE-600-24	SE-600-27	SE-600-36	SE-600-48
OUTPUT	DC VOLTAGE	5V	12V	15V	24V	27V	36V	48V
	RATED CURRENT	100A	50A	40A	25A	22.2A	16.6A	12.5A
	CURRENT RANGE	0 ~ 100A	0 ~ 50A	0 ~ 40A	0 ~ 25A	0 ~ 22.2A	0 ~ 16.6A	0 ~ 12.5A
	RATED POWER	500W	600W	600W	600W	599.4W	597.6W	600W
	RIPPLE & NOISE (max.) Note.2	150mVp-p	150mVp-p	150mVp-p	150mVp-p	150mVp-p	200mVp-p	200mVp-p
	VOLTAGE ADJ. RANGE	4.75 ~ 5.5V	10 ~ 13.5V	13.5 ~ 16.5V	22 ~ 26.4V	24 ~ 30V	32 ~ 40V	43 ~ 56V
	VOLTAGE TOLERANCE Note.3	±2.0%	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%
	LINE REGULATION	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%
	LOAD REGULATION	±2.0%	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%
	SETUP, RISE TIME	1000ms, 50ms/230VAC 1000ms, 50ms/115VAC at full load						
	HOLD UP TIME (Typ.)	20ms/230VAC 16ms/115VAC at full load						
INPUT	VOLTAGE RANGE	90 ~ 132VAC / 180 ~ 264VAC selected by switch 254 ~ 370VDC						
	FREQUENCY RANGE	47 ~ 63Hz						
	EFFICIENCY (Typ.)	78%	83%	84%	87%	87%	87%	88%
	AC CURRENT (Typ.)	12A/115VAC 7.5A/230VAC						
	INRUSH CURRENT (Typ.)	30A/115VAC 60A/230VAC						
	LEAKAGE CURRENT	<2.0mA / 240VAC						
PROTECTION	OVERLOAD	105 ~ 125% rated output power						
		Protection type : Shut down o/p voltage, re-power on to recover						
	OVER VOLTAGE	5.75 ~ 6.3V	13.8 ~ 16.2V	18 ~ 21V	27.6 ~ 32.4V	31 ~ 36.5V	42 ~ 50V	57.6 ~ 67.2V
		Protection type : Shut down o/p voltage, re-power on to recover						
	OVER TEMPERATURE	85℃±5℃ (TSW1) detect on heatsink of power transistor						
		Protection type : Shut down o/p voltage, recovers automatically after temperature goes down						
ENVIRONMENT	WORKING TEMP.	-20 ~ +60℃ (Refer to output load derating curve)						
	WORKING HUMIDITY	20 ~ 90% RH non-condensing						
	STORAGE TEMP., HUMIDITY	-40 ~ +85℃, 10 ~ 95% RH						
	TEMP. COEFFICIENT	±0.05%/℃ (0 ~ 50℃)						
	VIBRATION	10 ~ 500Hz, 2G 10min./1cycle, 60min. each along X, Y, Z axes						
SAFETY & EMC	SAFETY STANDARDS	UL60950-1,UL62368-1, EAC TP TC 004, BSMI CNS14336-1, KC K60950-1(for 12V,24V only) approved						
	WITHSTAND VOLTAGE	I/P-O/P:3KVAC I/P-FG:1.5KVAC O/P-FG:0.5KVAC						
	ISOLATION RESISTANCE	I/P-O/P, I/P-FG, O/P-FG:100M Ohms/500VDC						
OTHERS	MTBF	197K hrs min. MIL-HDBK-217F (25℃)						
	DIMENSION	247*127*63.5mm (L*W*H)						
	PACKING	2.1Kg; 6pcs/13.4Kg/1.03CUFT						

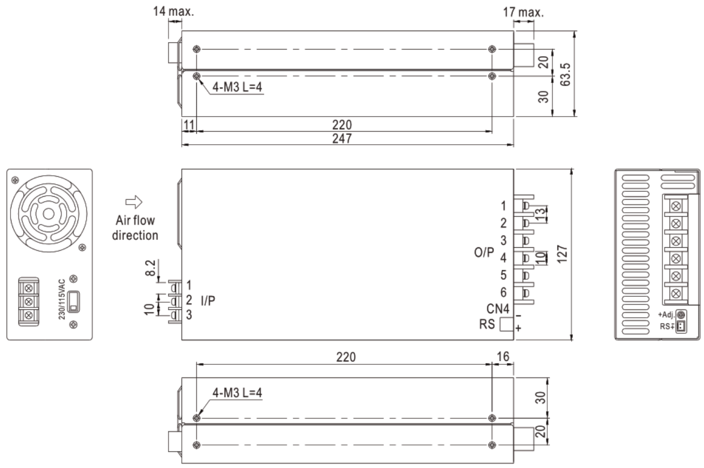
NOTE

1. All parameters NOT specially mentioned are measured at 230VAC input, rated load and 25℃ of ambient temperature.
2. Ripple & noise are measured at 20MHz of bandwidth by using a 12" twisted pair-wire terminated with a 0.1uf & 47uf parallel capacitor.
3. Tolerance : includes set up tolerance, line regulation and load regulation.
4. The ambient temperature derating of 3.5℃/1000m with fanless models and of 5℃/1000m with fan models for operating altitude higher than 2000m(6500ft).
5. This power supply does not meet the harmonic current requirements outlined by EN61000-3-2. Please do not use this power supply under the following conditions:
   a) the end-devices is used within the European Union, and
   b) the end-devices is connected to public mains supply with 220Vac or greater rated nominal voltage, and
   c) the power supply is:
   - installed in end-devices with average or continuous input power greater than 75W, or
   - belong to part of a lighting system
   Exception:
   Power supplies used within the following end-devices do not need to fulfill EN61000-3-2
   a) professional equipment with a total rated input power greater than 1000W;
   b) symmetrically controlled heating elements with a rated power less than or equal to 200W

File Name:SE-600-SPEC 2019-01-25

Mechanical Specification

Case No. 926A Unit:mm



AC input terminal :

Pin No.	Assignment
1	AC/L
2	AC/N
3	FG ⏚

DC Output terminal :

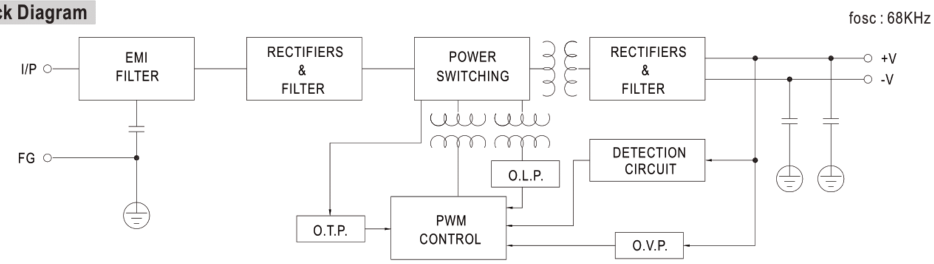
Pin No.	Assignment
1~3	+V
4~6	-V

RS Connector (CN4) : JST B-XH or equivalent

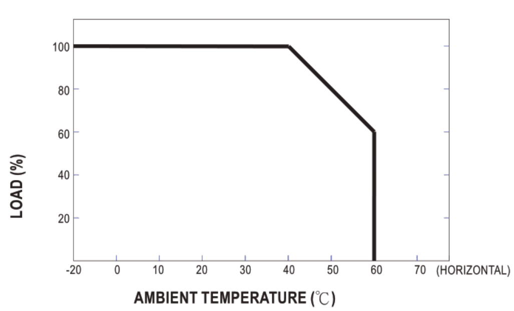
Pin No.	Assignment	Mating Housing	Terminal
1	+S	JST XHP or equivalent	JST SXH-001T or equivalent
2	-S		

Block Diagram

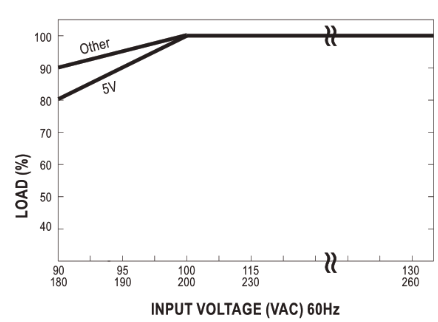
fosc : 68KHz



Derating Curve

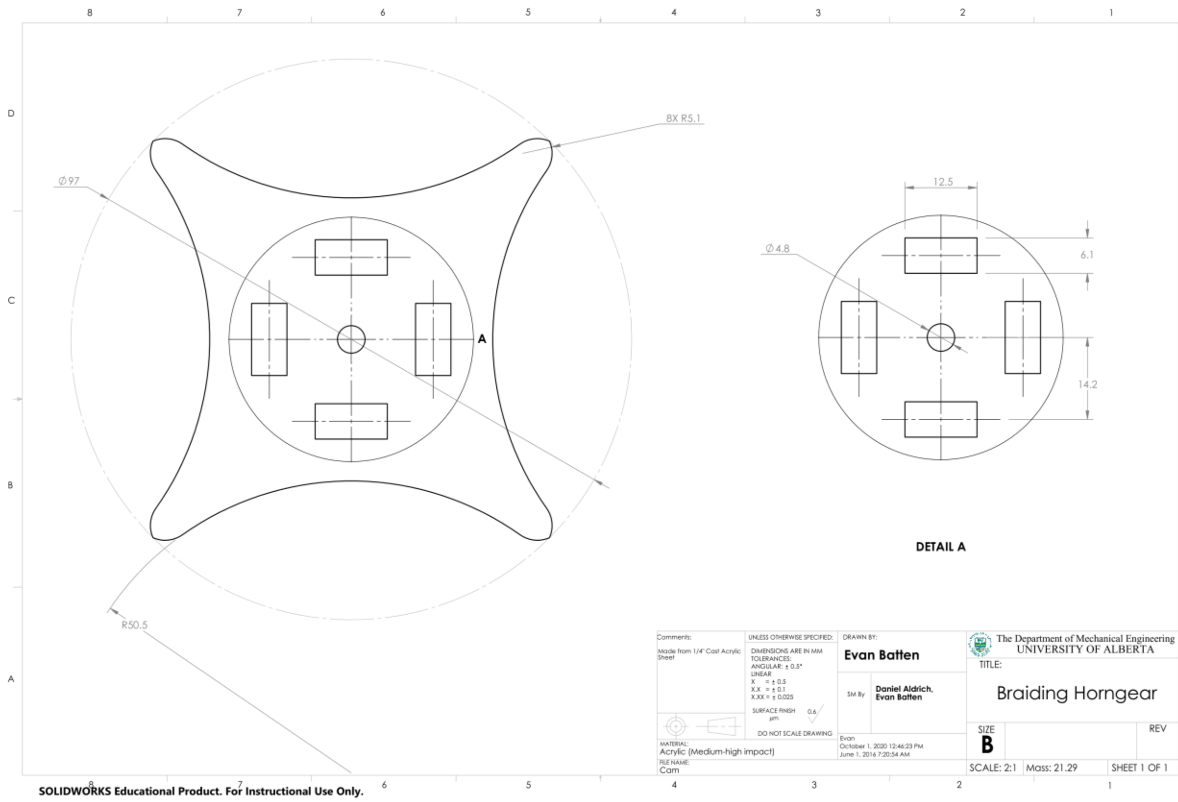


Static Characteristics



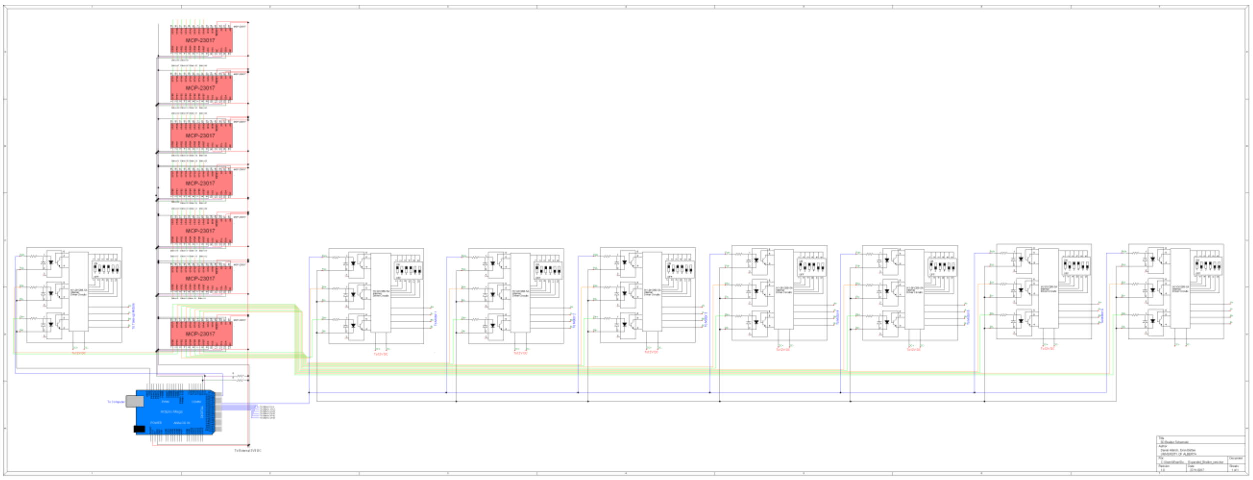
Appendix B Redesigned Component CAD Drawings

B.1 Redesigned Horngear Drawing

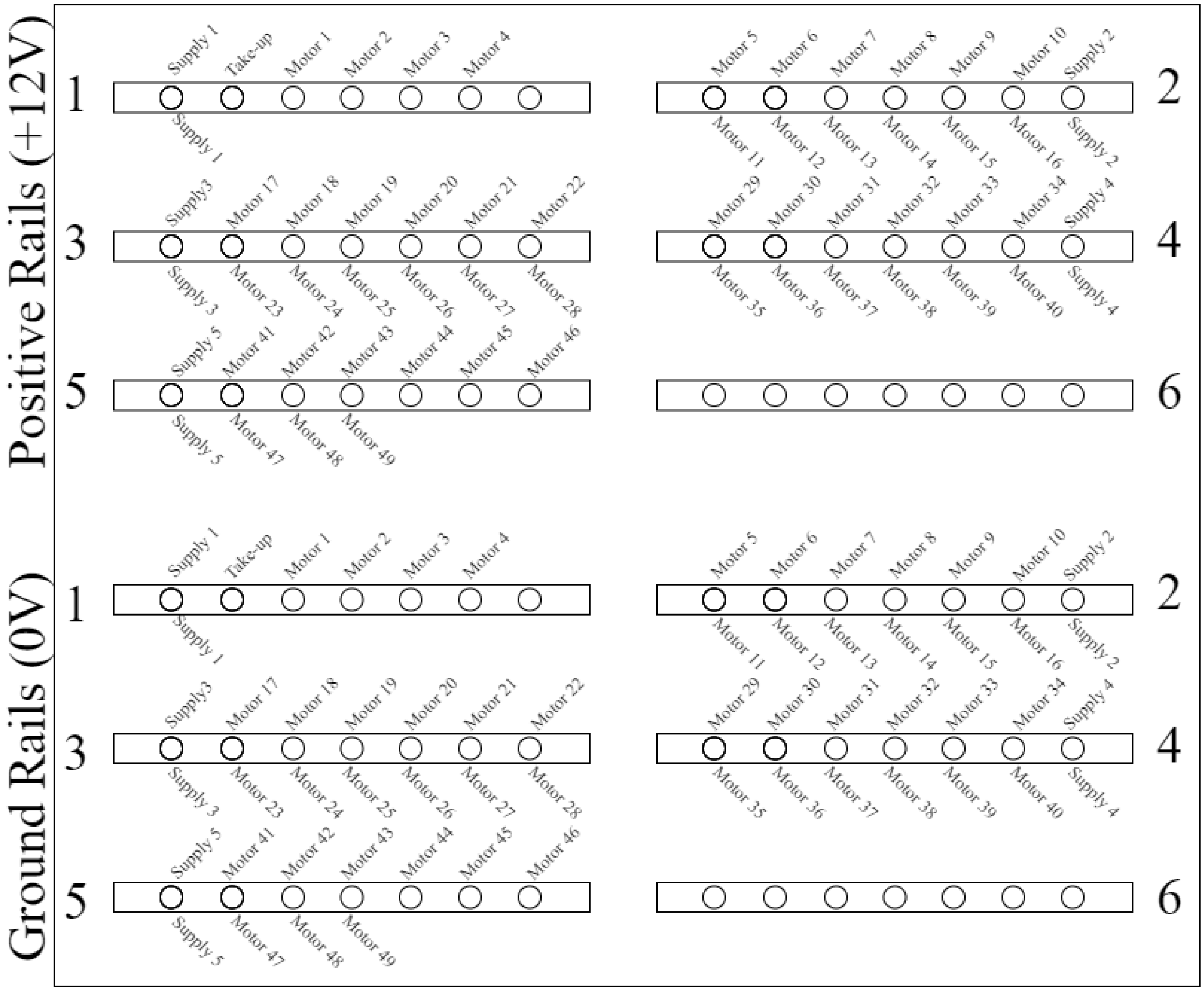


Appendix C Electrical/Electronic Schematics and Calculations

C.1 Control Schematic



C.2 Power Schematic



C.3 Power Calculations

Power Supply Sizing Calculations

Assumptions:

1. Worst-case scenario (all active motors in a circuit draw maximum current simultaneously)
2. Only half of the motors in a circuit are active at a given time (exception: take-up)

Motor Groupings:

1. Circuit 1: Take-Up, motors 1-4
2. Circuit 2: Motors 5-16
3. Circuit 3: Motors 17-28
4. Circuit 4: Motors 29-40
5. Circuit 5: Motors 41-49

Known Values:

- Supply Voltage: 12 V
- Motor Resistance: 1.65 Ω

$V_{sup} := 12\ \mathbf{V} \qquad R_{motor} := 1.65\ \mathbf{\Omega}$

$$I_{max} := \frac{V_{sup}}{R_{motor}} = 7.273\ \mathbf{A}$$

$$I_{max\_circuit} := \frac{12}{2} \cdot I_{max} = 43.636\ \mathbf{A}$$

Based on this calculation, a power supply capable of delivering up to 50 A at 12 V was selected.

Max current for each circuit:

1. Circuit 1: max 3 motors active, 21.818 A
2. Circuit 2: max 6 motors active, 43.636 A
3. Circuit 3: max 6 motors active, 43.636 A
4. Circuit 4: max 6 motors active, 43.636 A
5. Circuit 5: max 5 motors active, 36.364 A

Appendix D Computer Code

D.1 Modeling and Control Application Code

```
classdef braidGenerator3 < matlab.apps.AppBase


    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        TabGroup                        matlab.ui.container.TabGroup
        CarrierSetup                    matlab.ui.container.Tab
        CamMotorSetupPanel              matlab.ui.container.Panel
        PopulateCarriersButton          matlab.ui.control.Button
        cam_1                           matlab.ui.control.StateButton
        cam_2                           matlab.ui.control.StateButton
        cam_3                           matlab.ui.control.StateButton
        cam_4                           matlab.ui.control.StateButton
        cam_5                           matlab.ui.control.StateButton
        cam_6                           matlab.ui.control.StateButton
        cam_7                           matlab.ui.control.StateButton
        cam_8                           matlab.ui.control.StateButton
        cam_9                           matlab.ui.control.StateButton
        cam_10                          matlab.ui.control.StateButton
        cam_11                          matlab.ui.control.StateButton
        cam_12                          matlab.ui.control.StateButton
        cam_13                          matlab.ui.control.StateButton
        cam_14                          matlab.ui.control.StateButton
        cam_15                          matlab.ui.control.StateButton
        cam_16                          matlab.ui.control.StateButton
        cam_17                          matlab.ui.control.StateButton
        cam_18                          matlab.ui.control.StateButton
        cam_19                          matlab.ui.control.StateButton
        cam_20                          matlab.ui.control.StateButton
        cam_21                          matlab.ui.control.StateButton
        cam_22                          matlab.ui.control.StateButton
        cam_23                          matlab.ui.control.StateButton
        cam_24                          matlab.ui.control.StateButton
        cam_25                          matlab.ui.control.StateButton
        cam_26                          matlab.ui.control.StateButton
        cam_27                          matlab.ui.control.StateButton
        cam_28                          matlab.ui.control.StateButton
        cam_29                          matlab.ui.control.StateButton
        cam_30                          matlab.ui.control.StateButton
        cam_31                          matlab.ui.control.StateButton
        cam_32                          matlab.ui.control.StateButton
        cam_33                          matlab.ui.control.StateButton
        cam_34                          matlab.ui.control.StateButton
```

```
cam_35                          matlab.ui.control.StateButton
cam_36                          matlab.ui.control.StateButton
cam_37                          matlab.ui.control.StateButton
cam_38                          matlab.ui.control.StateButton
cam_39                          matlab.ui.control.StateButton
cam_40                          matlab.ui.control.StateButton
cam_41                          matlab.ui.control.StateButton
cam_42                          matlab.ui.control.StateButton
cam_43                          matlab.ui.control.StateButton
cam_44                          matlab.ui.control.StateButton
cam_45                          matlab.ui.control.StateButton
cam_46                          matlab.ui.control.StateButton
cam_47                          matlab.ui.control.StateButton
cam_48                          matlab.ui.control.StateButton
cam_49                          matlab.ui.control.StateButton
ActivateAllButton               matlab.ui.control.Button
DeactivateAllButton             matlab.ui.control.Button
AxialYarnsButton                matlab.ui.control.StateButton
SaveSelectionButton             matlab.ui.control.Button
CarrierSetupPanel               matlab.ui.container.Panel
Button_1                        matlab.ui.control.StateButton
Button_2                        matlab.ui.control.StateButton
Button_3                        matlab.ui.control.StateButton
Button_4                        matlab.ui.control.StateButton
Button_5                        matlab.ui.control.StateButton
Button_6                        matlab.ui.control.StateButton
Button_7                        matlab.ui.control.StateButton
Button_8                        matlab.ui.control.StateButton
Button_9                        matlab.ui.control.StateButton
Button_10                       matlab.ui.control.StateButton
Button_11                       matlab.ui.control.StateButton
Button_12                       matlab.ui.control.StateButton
Button_13                       matlab.ui.control.StateButton
Button_14                       matlab.ui.control.StateButton
Button_15                       matlab.ui.control.StateButton
Button_16                       matlab.ui.control.StateButton
Button_17                       matlab.ui.control.StateButton
Button_18                       matlab.ui.control.StateButton
Button_19                       matlab.ui.control.StateButton
Button_20                       matlab.ui.control.StateButton
Button_21                       matlab.ui.control.StateButton
Button_22                       matlab.ui.control.StateButton
Button_23                       matlab.ui.control.StateButton
Button_24                       matlab.ui.control.StateButton
Button_25                       matlab.ui.control.StateButton
Button_26                       matlab.ui.control.StateButton
Button_27                       matlab.ui.control.StateButton
Button_28                       matlab.ui.control.StateButton
Button_29                       matlab.ui.control.StateButton
Button_30                       matlab.ui.control.StateButton
```

```
Button_31                       matlab.ui.control.StateButton
Button_32                       matlab.ui.control.StateButton
Button_33                       matlab.ui.control.StateButton
Button_34                       matlab.ui.control.StateButton
Button_35                       matlab.ui.control.StateButton
Button_36                       matlab.ui.control.StateButton
Button_37                       matlab.ui.control.StateButton
Button_38                       matlab.ui.control.StateButton
Button_39                       matlab.ui.control.StateButton
Button_40                       matlab.ui.control.StateButton
Button_41                       matlab.ui.control.StateButton
Button_42                       matlab.ui.control.StateButton
Button_43                       matlab.ui.control.StateButton
Button_44                       matlab.ui.control.StateButton
Button_45                       matlab.ui.control.StateButton
Button_46                       matlab.ui.control.StateButton
Button_47                       matlab.ui.control.StateButton
Button_48                       matlab.ui.control.StateButton
Button_49                       matlab.ui.control.StateButton
Button_50                       matlab.ui.control.StateButton
Button_51                       matlab.ui.control.StateButton
Button_52                       matlab.ui.control.StateButton
Button_53                       matlab.ui.control.StateButton
Button_54                       matlab.ui.control.StateButton
Button_55                       matlab.ui.control.StateButton
Button_56                       matlab.ui.control.StateButton
Button_57                       matlab.ui.control.StateButton
Button_58                       matlab.ui.control.StateButton
Button_59                       matlab.ui.control.StateButton
Button_60                       matlab.ui.control.StateButton
Button_61                       matlab.ui.control.StateButton
Button_62                       matlab.ui.control.StateButton
Button_63                       matlab.ui.control.StateButton
Button_64                       matlab.ui.control.StateButton
Button_65                       matlab.ui.control.StateButton
Button_66                       matlab.ui.control.StateButton
Button_67                       matlab.ui.control.StateButton
Button_68                       matlab.ui.control.StateButton
Button_69                       matlab.ui.control.StateButton
Button_70                       matlab.ui.control.StateButton
Button_71                       matlab.ui.control.StateButton
Button_72                       matlab.ui.control.StateButton
Button_73                       matlab.ui.control.StateButton
Button_74                       matlab.ui.control.StateButton
Button_75                       matlab.ui.control.StateButton
Button_76                       matlab.ui.control.StateButton
Button_77                       matlab.ui.control.StateButton
Button_78                       matlab.ui.control.StateButton
Button_79                       matlab.ui.control.StateButton
Button_80                       matlab.ui.control.StateButton
```

Button_81	matlab.ui.control.StateButton
Button_82	matlab.ui.control.StateButton
Button_83	matlab.ui.control.StateButton
Button_84	matlab.ui.control.StateButton
Button_85	matlab.ui.control.StateButton
Button_86	matlab.ui.control.StateButton
Button_87	matlab.ui.control.StateButton
Button_88	matlab.ui.control.StateButton
Button_89	matlab.ui.control.StateButton
Button_90	matlab.ui.control.StateButton
Button_91	matlab.ui.control.StateButton
Button_92	matlab.ui.control.StateButton
Button_93	matlab.ui.control.StateButton
Button_94	matlab.ui.control.StateButton
Button_95	matlab.ui.control.StateButton
Button_96	matlab.ui.control.StateButton
Button_97	matlab.ui.control.StateButton
Button_98	matlab.ui.control.StateButton
Button_99	matlab.ui.control.StateButton
Button_100	matlab.ui.control.StateButton
Button_101	matlab.ui.control.StateButton
Button_102	matlab.ui.control.StateButton
Button_103	matlab.ui.control.StateButton
Button_104	matlab.ui.control.StateButton
Button_105	matlab.ui.control.StateButton
Button_106	matlab.ui.control.StateButton
Button_107	matlab.ui.control.StateButton
Button_108	matlab.ui.control.StateButton
Button_109	matlab.ui.control.StateButton
Button_110	matlab.ui.control.StateButton
Button_111	matlab.ui.control.StateButton
Button_112	matlab.ui.control.StateButton
Button_113	matlab.ui.control.StateButton
Button_114	matlab.ui.control.StateButton
Button_115	matlab.ui.control.StateButton
Button_116	matlab.ui.control.StateButton
Button_117	matlab.ui.control.StateButton
Button_118	matlab.ui.control.StateButton
Button_119	matlab.ui.control.StateButton
Button_120	matlab.ui.control.StateButton
Button_121	matlab.ui.control.StateButton
Button_122	matlab.ui.control.StateButton
Button_123	matlab.ui.control.StateButton
Button_124	matlab.ui.control.StateButton
Button_125	matlab.ui.control.StateButton
Button_126	matlab.ui.control.StateButton
Button_127	matlab.ui.control.StateButton
Button_128	matlab.ui.control.StateButton
Button_129	matlab.ui.control.StateButton
Button_130	matlab.ui.control.StateButton

Button_131	matlab.ui.control.StateButton
Button_132	matlab.ui.control.StateButton
Button_133	matlab.ui.control.StateButton
Button_134	matlab.ui.control.StateButton
Button_135	matlab.ui.control.StateButton
Button_136	matlab.ui.control.StateButton
Button_137	matlab.ui.control.StateButton
Button_138	matlab.ui.control.StateButton
Button_139	matlab.ui.control.StateButton
Button_140	matlab.ui.control.StateButton
Button_141	matlab.ui.control.StateButton
Button_142	matlab.ui.control.StateButton
Button_143	matlab.ui.control.StateButton
Button_144	matlab.ui.control.StateButton
Button_145	matlab.ui.control.StateButton
Button_146	matlab.ui.control.StateButton
Button_147	matlab.ui.control.StateButton
Button_148	matlab.ui.control.StateButton
Button_149	matlab.ui.control.StateButton
Button_150	matlab.ui.control.StateButton
Button_151	matlab.ui.control.StateButton
Button_152	matlab.ui.control.StateButton
Button_153	matlab.ui.control.StateButton
Button_154	matlab.ui.control.StateButton
Button_155	matlab.ui.control.StateButton
Button_156	matlab.ui.control.StateButton
Button_157	matlab.ui.control.StateButton
Button_158	matlab.ui.control.StateButton
Button_159	matlab.ui.control.StateButton
Button_160	matlab.ui.control.StateButton
Button_161	matlab.ui.control.StateButton
Button_162	matlab.ui.control.StateButton
Button_163	matlab.ui.control.StateButton
Button_164	matlab.ui.control.StateButton
Button_165	matlab.ui.control.StateButton
Button_166	matlab.ui.control.StateButton
Button_167	matlab.ui.control.StateButton
Button_168	matlab.ui.control.StateButton
Button_169	matlab.ui.control.StateButton
Button_170	matlab.ui.control.StateButton
Button_171	matlab.ui.control.StateButton
Button_172	matlab.ui.control.StateButton
Button_173	matlab.ui.control.StateButton
Button_174	matlab.ui.control.StateButton
Button_175	matlab.ui.control.StateButton
Button_176	matlab.ui.control.StateButton
Button_177	matlab.ui.control.StateButton
Button_178	matlab.ui.control.StateButton
Button_179	matlab.ui.control.StateButton
Button_180	matlab.ui.control.StateButton

```
Button_181                    matlab.ui.control.StateButton
Button_182                    matlab.ui.control.StateButton
Button_183                    matlab.ui.control.StateButton
Button_184                    matlab.ui.control.StateButton
Button_185                    matlab.ui.control.StateButton
Button_186                    matlab.ui.control.StateButton
Button_187                    matlab.ui.control.StateButton
Button_188                    matlab.ui.control.StateButton
Button_189                    matlab.ui.control.StateButton
Button_190                    matlab.ui.control.StateButton
Button_191                    matlab.ui.control.StateButton
Button_192                    matlab.ui.control.StateButton
Button_193                    matlab.ui.control.StateButton
Button_194                    matlab.ui.control.StateButton
Button_195                    matlab.ui.control.StateButton
Button_196                    matlab.ui.control.StateButton
Button_197                    matlab.ui.control.StateButton
Button_198                    matlab.ui.control.StateButton
Button_199                    matlab.ui.control.StateButton
Button_200                    matlab.ui.control.StateButton
Button_201                    matlab.ui.control.StateButton
Button_202                    matlab.ui.control.StateButton
Button_203                    matlab.ui.control.StateButton
Button_204                    matlab.ui.control.StateButton
Button_205                    matlab.ui.control.StateButton
Button_206                    matlab.ui.control.StateButton
Button_207                    matlab.ui.control.StateButton
Button_208                    matlab.ui.control.StateButton
Button_209                    matlab.ui.control.StateButton
Button_210                    matlab.ui.control.StateButton
Button_211                    matlab.ui.control.StateButton
Button_212                    matlab.ui.control.StateButton
Button_213                    matlab.ui.control.StateButton
Button_214                    matlab.ui.control.StateButton
Button_215                    matlab.ui.control.StateButton
Button_216                    matlab.ui.control.StateButton
Button_217                    matlab.ui.control.StateButton
Button_218                    matlab.ui.control.StateButton
Button_219                    matlab.ui.control.StateButton
Button_220                    matlab.ui.control.StateButton
Button_221                    matlab.ui.control.StateButton
Button_222                    matlab.ui.control.StateButton
Button_223                    matlab.ui.control.StateButton
Button_224                    matlab.ui.control.StateButton
Button_225                    matlab.ui.control.StateButton
BraidPatternSetup             matlab.ui.container.Tab
Motor1Panel                   matlab.ui.container.Panel
MotorStateButton_1            matlab.ui.control.StateButton
MotorDirectionButton_1        matlab.ui.control.StateButton
MotorRotationSpinner_1        matlab.ui.control.Spinner
```

```
AngleSpinnerLabel                matlab.ui.control.Label
Motor2Panel                      matlab.ui.container.Panel
MotorStateButton_2               matlab.ui.control.StateButton
MotorDirectionButton_2           matlab.ui.control.StateButton
MotorRotationSpinner_2           matlab.ui.control.Spinner
AngleSpinnerLabel_2              matlab.ui.control.Label
Motor3Panel                      matlab.ui.container.Panel
MotorStateButton_3               matlab.ui.control.StateButton
MotorDirectionButton_3           matlab.ui.control.StateButton
MotorRotationSpinner_3           matlab.ui.control.Spinner
AngleSpinnerLabel_3              matlab.ui.control.Label
Motor4Panel                      matlab.ui.container.Panel
MotorStateButton_4               matlab.ui.control.StateButton
MotorDirectionButton_4           matlab.ui.control.StateButton
MotorRotationSpinner_4           matlab.ui.control.Spinner
AngleSpinnerLabel_4              matlab.ui.control.Label
Motor5Panel                      matlab.ui.container.Panel
MotorStateButton_5               matlab.ui.control.StateButton
MotorDirectionButton_5           matlab.ui.control.StateButton
MotorRotationSpinner_5           matlab.ui.control.Spinner
AngleSpinnerLabel_5              matlab.ui.control.Label
Motor6Panel                      matlab.ui.container.Panel
MotorStateButton_6               matlab.ui.control.StateButton
MotorDirectionButton_6           matlab.ui.control.StateButton
MotorRotationSpinner_6           matlab.ui.control.Spinner
AngleSpinnerLabel_6              matlab.ui.control.Label
Motor7Panel                      matlab.ui.container.Panel
MotorStateButton_7               matlab.ui.control.StateButton
MotorDirectionButton_7           matlab.ui.control.StateButton
MotorRotationSpinner_7           matlab.ui.control.Spinner
AngleSpinnerLabel_7              matlab.ui.control.Label
Motor8Panel                      matlab.ui.container.Panel
MotorStateButton_8               matlab.ui.control.StateButton
MotorDirectionButton_8           matlab.ui.control.StateButton
MotorRotationSpinner_8           matlab.ui.control.Spinner
AngleSpinnerLabel_8              matlab.ui.control.Label
Motor9Panel                      matlab.ui.container.Panel
MotorStateButton_9               matlab.ui.control.StateButton
MotorDirectionButton_9           matlab.ui.control.StateButton
MotorRotationSpinner_9           matlab.ui.control.Spinner
AngleSpinnerLabel_9              matlab.ui.control.Label
Motor10Panel                     matlab.ui.container.Panel
MotorStateButton_10              matlab.ui.control.StateButton
MotorDirectionButton_10          matlab.ui.control.StateButton
MotorRotationSpinner_10          matlab.ui.control.Spinner
AngleSpinnerLabel_10             matlab.ui.control.Label
Motor11Panel                     matlab.ui.container.Panel
MotorStateButton_11              matlab.ui.control.StateButton
MotorDirectionButton_11          matlab.ui.control.StateButton
MotorRotationSpinner_11          matlab.ui.control.Spinner
```

```
AngleSpinnerLabel_11          matlab.ui.control.Label
Motor12Panel                  matlab.ui.container.Panel
MotorStateButton_12           matlab.ui.control.StateButton
MotorDirectionButton_12       matlab.ui.control.StateButton
MotorRotationSpinner_12       matlab.ui.control.Spinner
AngleSpinnerLabel_12          matlab.ui.control.Label
Motor13Panel                  matlab.ui.container.Panel
MotorStateButton_13           matlab.ui.control.StateButton
MotorDirectionButton_13       matlab.ui.control.StateButton
MotorRotationSpinner_13       matlab.ui.control.Spinner
AngleSpinnerLabel_13          matlab.ui.control.Label
Motor14Panel                  matlab.ui.container.Panel
MotorStateButton_14           matlab.ui.control.StateButton
MotorDirectionButton_14       matlab.ui.control.StateButton
MotorRotationSpinner_14       matlab.ui.control.Spinner
AngleSpinnerLabel_14          matlab.ui.control.Label
Motor15Panel                  matlab.ui.container.Panel
MotorStateButton_15           matlab.ui.control.StateButton
MotorDirectionButton_15       matlab.ui.control.StateButton
MotorRotationSpinner_15       matlab.ui.control.Spinner
AngleSpinnerLabel_15          matlab.ui.control.Label
Motor16Panel                  matlab.ui.container.Panel
MotorStateButton_16           matlab.ui.control.StateButton
MotorDirectionButton_16       matlab.ui.control.StateButton
MotorRotationSpinner_16       matlab.ui.control.Spinner
AngleSpinnerLabel_16          matlab.ui.control.Label
Motor17Panel                  matlab.ui.container.Panel
MotorStateButton_17           matlab.ui.control.StateButton
MotorDirectionButton_17       matlab.ui.control.StateButton
MotorRotationSpinner_17       matlab.ui.control.Spinner
AngleSpinnerLabel_17          matlab.ui.control.Label
Motor18Panel                  matlab.ui.container.Panel
MotorStateButton_18           matlab.ui.control.StateButton
MotorDirectionButton_18       matlab.ui.control.StateButton
MotorRotationSpinner_18       matlab.ui.control.Spinner
AngleSpinnerLabel_18          matlab.ui.control.Label
Motor19Panel                  matlab.ui.container.Panel
MotorStateButton_19           matlab.ui.control.StateButton
MotorDirectionButton_19       matlab.ui.control.StateButton
MotorRotationSpinner_19       matlab.ui.control.Spinner
AngleSpinnerLabel_19          matlab.ui.control.Label
Motor20Panel                  matlab.ui.container.Panel
MotorStateButton_20           matlab.ui.control.StateButton
MotorDirectionButton_20       matlab.ui.control.StateButton
MotorRotationSpinner_20       matlab.ui.control.Spinner
AngleSpinnerLabel_20          matlab.ui.control.Label
Motor21Panel                  matlab.ui.container.Panel
MotorStateButton_21           matlab.ui.control.StateButton
MotorDirectionButton_21       matlab.ui.control.StateButton
MotorRotationSpinner_21       matlab.ui.control.Spinner
```

```
AngleSpinnerLabel_21            matlab.ui.control.Label
Motor22Panel                    matlab.ui.container.Panel
MotorStateButton_22             matlab.ui.control.StateButton
MotorDirectionButton_22         matlab.ui.control.StateButton
MotorRotationSpinner_22         matlab.ui.control.Spinner
AngleSpinnerLabel_22            matlab.ui.control.Label
Motor23Panel                    matlab.ui.container.Panel
MotorStateButton_23             matlab.ui.control.StateButton
MotorDirectionButton_23         matlab.ui.control.StateButton
MotorRotationSpinner_23         matlab.ui.control.Spinner
AngleSpinnerLabel_23            matlab.ui.control.Label
Motor24Panel                    matlab.ui.container.Panel
MotorStateButton_24             matlab.ui.control.StateButton
MotorDirectionButton_24         matlab.ui.control.StateButton
MotorRotationSpinner_24         matlab.ui.control.Spinner
AngleSpinnerLabel_24            matlab.ui.control.Label
Motor25Panel                    matlab.ui.container.Panel
MotorStateButton_25             matlab.ui.control.StateButton
MotorDirectionButton_25         matlab.ui.control.StateButton
MotorRotationSpinner_25         matlab.ui.control.Spinner
AngleSpinnerLabel_25            matlab.ui.control.Label
Motor26Panel                    matlab.ui.container.Panel
MotorStateButton_26             matlab.ui.control.StateButton
MotorDirectionButton_26         matlab.ui.control.StateButton
MotorRotationSpinner_26         matlab.ui.control.Spinner
AngleSpinnerLabel_26            matlab.ui.control.Label
Motor27Panel                    matlab.ui.container.Panel
MotorStateButton_27             matlab.ui.control.StateButton
MotorDirectionButton_27         matlab.ui.control.StateButton
MotorRotationSpinner_27         matlab.ui.control.Spinner
AngleSpinnerLabel_27            matlab.ui.control.Label
Motor28Panel                    matlab.ui.container.Panel
MotorStateButton_28             matlab.ui.control.StateButton
MotorDirectionButton_28         matlab.ui.control.StateButton
MotorRotationSpinner_28         matlab.ui.control.Spinner
AngleSpinnerLabel_28            matlab.ui.control.Label
Motor29Panel                    matlab.ui.container.Panel
MotorStateButton_29             matlab.ui.control.StateButton
MotorDirectionButton_29         matlab.ui.control.StateButton
MotorRotationSpinner_29         matlab.ui.control.Spinner
AngleSpinnerLabel_29            matlab.ui.control.Label
Motor30Panel                    matlab.ui.container.Panel
MotorStateButton_30             matlab.ui.control.StateButton
MotorDirectionButton_30         matlab.ui.control.StateButton
MotorRotationSpinner_30         matlab.ui.control.Spinner
AngleSpinnerLabel_30            matlab.ui.control.Label
Motor31Panel                    matlab.ui.container.Panel
MotorStateButton_31             matlab.ui.control.StateButton
MotorDirectionButton_31         matlab.ui.control.StateButton
MotorRotationSpinner_31         matlab.ui.control.Spinner
```

```
AngleSpinnerLabel_31          matlab.ui.control.Label
Motor32Panel                  matlab.ui.container.Panel
MotorStateButton_32           matlab.ui.control.StateButton
MotorDirectionButton_32       matlab.ui.control.StateButton
MotorRotationSpinner_32       matlab.ui.control.Spinner
AngleSpinnerLabel_32          matlab.ui.control.Label
Motor33Panel                  matlab.ui.container.Panel
MotorStateButton_33           matlab.ui.control.StateButton
MotorDirectionButton_33       matlab.ui.control.StateButton
MotorRotationSpinner_33       matlab.ui.control.Spinner
AngleSpinnerLabel_33          matlab.ui.control.Label
Motor34Panel                  matlab.ui.container.Panel
MotorStateButton_34           matlab.ui.control.StateButton
MotorDirectionButton_34       matlab.ui.control.StateButton
MotorRotationSpinner_34       matlab.ui.control.Spinner
AngleSpinnerLabel_34          matlab.ui.control.Label
Motor35Panel                  matlab.ui.container.Panel
MotorStateButton_35           matlab.ui.control.StateButton
MotorDirectionButton_35       matlab.ui.control.StateButton
MotorRotationSpinner_35       matlab.ui.control.Spinner
AngleSpinnerLabel_35          matlab.ui.control.Label
Motor36Panel                  matlab.ui.container.Panel
MotorStateButton_36           matlab.ui.control.StateButton
MotorDirectionButton_36       matlab.ui.control.StateButton
MotorRotationSpinner_36       matlab.ui.control.Spinner
AngleSpinnerLabel_36          matlab.ui.control.Label
Motor37Panel                  matlab.ui.container.Panel
MotorStateButton_37           matlab.ui.control.StateButton
MotorDirectionButton_37       matlab.ui.control.StateButton
MotorRotationSpinner_37       matlab.ui.control.Spinner
AngleSpinnerLabel_37          matlab.ui.control.Label
Motor38Panel                  matlab.ui.container.Panel
MotorStateButton_38           matlab.ui.control.StateButton
MotorDirectionButton_38       matlab.ui.control.StateButton
MotorRotationSpinner_38       matlab.ui.control.Spinner
AngleSpinnerLabel_38          matlab.ui.control.Label
Motor39Panel                  matlab.ui.container.Panel
MotorStateButton_39           matlab.ui.control.StateButton
MotorDirectionButton_39       matlab.ui.control.StateButton
MotorRotationSpinner_39       matlab.ui.control.Spinner
AngleSpinnerLabel_39          matlab.ui.control.Label
Motor40Panel                  matlab.ui.container.Panel
MotorStateButton_40           matlab.ui.control.StateButton
MotorDirectionButton_40       matlab.ui.control.StateButton
MotorRotationSpinner_40       matlab.ui.control.Spinner
AngleSpinnerLabel_40          matlab.ui.control.Label
Motor41Panel                  matlab.ui.container.Panel
MotorStateButton_41           matlab.ui.control.StateButton
MotorDirectionButton_41       matlab.ui.control.StateButton
MotorRotationSpinner_41       matlab.ui.control.Spinner
```

```
AngleSpinnerLabel_41            matlab.ui.control.Label
Motor42Panel                    matlab.ui.container.Panel
MotorStateButton_42             matlab.ui.control.StateButton
MotorDirectionButton_42         matlab.ui.control.StateButton
MotorRotationSpinner_42         matlab.ui.control.Spinner
AngleSpinnerLabel_42            matlab.ui.control.Label
Motor43Panel                    matlab.ui.container.Panel
MotorStateButton_43             matlab.ui.control.StateButton
MotorDirectionButton_43         matlab.ui.control.StateButton
MotorRotationSpinner_43         matlab.ui.control.Spinner
AngleSpinnerLabel_43            matlab.ui.control.Label
Motor44Panel                    matlab.ui.container.Panel
MotorStateButton_44             matlab.ui.control.StateButton
MotorDirectionButton_44         matlab.ui.control.StateButton
MotorRotationSpinner_44         matlab.ui.control.Spinner
AngleSpinnerLabel_44            matlab.ui.control.Label
Motor45Panel                    matlab.ui.container.Panel
MotorStateButton_45             matlab.ui.control.StateButton
MotorDirectionButton_45         matlab.ui.control.StateButton
MotorRotationSpinner_45         matlab.ui.control.Spinner
AngleSpinnerLabel_45            matlab.ui.control.Label
Motor46Panel                    matlab.ui.container.Panel
MotorStateButton_46             matlab.ui.control.StateButton
MotorDirectionButton_46         matlab.ui.control.StateButton
MotorRotationSpinner_46         matlab.ui.control.Spinner
AngleSpinnerLabel_46            matlab.ui.control.Label
Motor47Panel                    matlab.ui.container.Panel
MotorStateButton_47             matlab.ui.control.StateButton
MotorDirectionButton_47         matlab.ui.control.StateButton
MotorRotationSpinner_47         matlab.ui.control.Spinner
AngleSpinnerLabel_47            matlab.ui.control.Label
Motor48Panel                    matlab.ui.container.Panel
MotorStateButton_48             matlab.ui.control.StateButton
MotorDirectionButton_48         matlab.ui.control.StateButton
MotorRotationSpinner_48         matlab.ui.control.Spinner
AngleSpinnerLabel_48            matlab.ui.control.Label
Motor49Panel                    matlab.ui.container.Panel
MotorStateButton_49             matlab.ui.control.StateButton
MotorDirectionButton_49         matlab.ui.control.StateButton
MotorRotationSpinner_49         matlab.ui.control.Spinner
AngleSpinnerLabel_49            matlab.ui.control.Label
MasterMotorRotationSpinner      matlab.ui.control.Spinner
MasterAngleControlLabel         matlab.ui.control.Label
MasterMotorDirectionButton      matlab.ui.control.StateButton
MasterDirectionControlLabel     matlab.ui.control.Label
SaveCurrentStepButton           matlab.ui.control.Button
SavedStepsDropDownLabel         matlab.ui.control.Label
SavedStepsDropDown              matlab.ui.control.DropDown
TakeUpControlButton             matlab.ui.control.StateButton
TakeUpControlLabel              matlab.ui.control.Label
```

```
TakeupSpeedmmsEditFieldLabel     matlab.ui.control.Label
TakeUpSpeedEditField             matlab.ui.control.NumericEditField
PresetsDropDownLabel             matlab.ui.control.Label
PresetsDropDown                  matlab.ui.control.DropDown
DeleteStepButton                 matlab.ui.control.Button
DeleteStepDropDown               matlab.ui.control.DropDown
DisableAllButton                 matlab.ui.control.Button
RepeatStepsEditField_min         matlab.ui.control.NumericEditField
RepeatStepstoLabel               matlab.ui.control.Label
RepeatStepsEditField_max         matlab.ui.control.NumericEditField
RepeatStepstimesLabel            matlab.ui.control.Label
RepeatStepsEditField_number      matlab.ui.control.NumericEditField
RepeatSavedStepsButton           matlab.ui.control.Button
ClearAllStepsButton              matlab.ui.control.Button
ReverseTakeupDirectionCheckBox   matlab.ui.control.CheckBox
CreateFullPathButton             matlab.ui.control.Button
Modeling                         matlab.ui.container.Tab
btGeneratePlots                  matlab.ui.control.Button
btGenerateModel                  matlab.ui.control.Button
cbSmoothing                      matlab.ui.control.CheckBox
cbTriAxial                       matlab.ui.control.CheckBox
btGeneratePaths                  matlab.ui.control.Button
btClosePlots                     matlab.ui.control.Button
cbType2Paths                     matlab.ui.control.CheckBox
cbDenier                         matlab.ui.control.CheckBox
LowerBoundPanel                  matlab.ui.container.Panel
tbAngleLamina                    matlab.ui.control.EditField
Label24                          matlab.ui.control.Label
etAngleLaminaEx                  matlab.ui.control.NumericEditField
Label25                          matlab.ui.control.Label
etAngleLaminavxy                 matlab.ui.control.NumericEditField
GxyLabel                         matlab.ui.control.Label
etAngleLaminaGxy                 matlab.ui.control.NumericEditField
Label27                          matlab.ui.control.Label
etAngleLaminaEy                  matlab.ui.control.NumericEditField
Label28                          matlab.ui.control.Label
etAngleLaminavxz                 matlab.ui.control.NumericEditField
GxzLabel                         matlab.ui.control.Label
etAngleLaminaGxz                 matlab.ui.control.NumericEditField
Label30                          matlab.ui.control.Label
etAngleLaminaEz                  matlab.ui.control.NumericEditField
Label31                          matlab.ui.control.Label
etAngleLaminavyz                 matlab.ui.control.NumericEditField
GyzLabel                         matlab.ui.control.Label
etAngleLaminaGyz                 matlab.ui.control.NumericEditField
UpperBoundPanel                  matlab.ui.container.Panel
tbLamina                         matlab.ui.control.EditField
Label15                          matlab.ui.control.Label
etLaminaE1                       matlab.ui.control.NumericEditField
Label16                          matlab.ui.control.Label
```

```
etLaminav12                         matlab.ui.control.NumericEditField
Label17                             matlab.ui.control.Label
etLaminaG12                         matlab.ui.control.NumericEditField
Label18                             matlab.ui.control.Label
etLaminaE2                          matlab.ui.control.NumericEditField
Label19                             matlab.ui.control.Label
etLaminav13                         matlab.ui.control.NumericEditField
Label20                             matlab.ui.control.Label
etLaminaG13                         matlab.ui.control.NumericEditField
Label21                             matlab.ui.control.Label
etLaminaE3                          matlab.ui.control.NumericEditField
Label22                             matlab.ui.control.Label
etLaminav23                         matlab.ui.control.NumericEditField
Label23                             matlab.ui.control.Label
etLaminaG23                         matlab.ui.control.NumericEditField
YarnPropertiesPanel                 matlab.ui.container.Panel
Label6                              matlab.ui.control.Label
etYarnE1                            matlab.ui.control.NumericEditField
Label7                              matlab.ui.control.Label
etYarnv12                           matlab.ui.control.NumericEditField
Label8                              matlab.ui.control.Label
etYarnG12                           matlab.ui.control.NumericEditField
Label9                              matlab.ui.control.Label
etYarnE2                            matlab.ui.control.NumericEditField
Label10                             matlab.ui.control.Label
etYarnv13                           matlab.ui.control.NumericEditField
Label11                             matlab.ui.control.Label
etYarnG13                           matlab.ui.control.NumericEditField
Label12                             matlab.ui.control.Label
etYarnE3                            matlab.ui.control.NumericEditField
Label13                             matlab.ui.control.Label
etYarnv23                           matlab.ui.control.NumericEditField
Label14                             matlab.ui.control.Label
etYarnG23                           matlab.ui.control.NumericEditField
YarnPackingFractionLabel            matlab.ui.control.Label
etYarnPackingFraction               matlab.ui.control.NumericEditField
YarnRadiusLabel                     matlab.ui.control.Label
etYarnRadius                        matlab.ui.control.NumericEditField
BraidProperties                     matlab.ui.container.Panel
BraidCrossSectionalDimsEditFieldLabel  matlab.ui.control.Label
BraidCrossSectionalDimsEditField  matlab.ui.control.EditField
UnitCellHeigthEditFieldLabel        matlab.ui.control.Label
UnitCellHeigthEditField             matlab.ui.control.EditField
AverageBraidingAngleEditFieldLabel  matlab.ui.control.Label
AverageBraidingAngleEditField       matlab.ui.control.EditField
BraidingTightnessEditFieldLabel  matlab.ui.control.Label
BraidingTightnessEditField          matlab.ui.control.EditField
FiberVolumeFractionEditFieldLabel  matlab.ui.control.Label
FiberVolumeFractionEditField        matlab.ui.control.EditField
NumberofYarnsEditFieldLabel         matlab.ui.control.Label
```

```
NumberofYarnsEditField           matlab.ui.control.EditField
YoungsModulusRangeEditFieldLabel  matlab.ui.control.Label
YoungsModulusRangeEditField      matlab.ui.control.EditField
InteriorBraidingAngleEditFieldLabel  matlab.ui.control.Label
InteriorBraidingAngleEditField  matlab.ui.control.EditField
SurfaceBraidingAngleEditFieldLabel  matlab.ui.control.Label
SurfaceBraidingAngleEditField   matlab.ui.control.EditField
FGMPanel                        matlab.ui.container.Panel
Label24_2                       matlab.ui.control.Label
etFGMEx                         matlab.ui.control.NumericEditField
Label25_2                       matlab.ui.control.Label
etFGMvxy                        matlab.ui.control.NumericEditField
GxyLabel_2                      matlab.ui.control.Label
etFGMGxy                        matlab.ui.control.NumericEditField
Label27_2                       matlab.ui.control.Label
etFGMEy                         matlab.ui.control.NumericEditField
Label28_2                       matlab.ui.control.Label
etFGMvxz                        matlab.ui.control.NumericEditField
GxzLabel_2                      matlab.ui.control.Label
etFGMGxz                        matlab.ui.control.NumericEditField
Label30_2                       matlab.ui.control.Label
etFGMEz                         matlab.ui.control.NumericEditField
Label31_2                       matlab.ui.control.Label
etFGMvyz                        matlab.ui.control.NumericEditField
GyzLabel_2                      matlab.ui.control.Label
etFGMGyz                        matlab.ui.control.NumericEditField
AnglesfromGeometryPanel         matlab.ui.container.Panel
ApproxAverageAngleLabel         matlab.ui.control.Label
etAverageAngle                  matlab.ui.control.NumericEditField
ApproxInteriorAngleLabel        matlab.ui.control.Label
etInteriorAngle                 matlab.ui.control.NumericEditField
ApproxSurfaceAngleLabel         matlab.ui.control.Label
etSurfaceAngle                  matlab.ui.control.NumericEditField
ButtonGroup                     matlab.ui.container.ButtonGroup
enablePitch                     matlab.ui.control.RadioButton
enableAngle                     matlab.ui.control.RadioButton
LabelNumericEditField2          matlab.ui.control.Label
etPathAngle                     matlab.ui.control.NumericEditField
LabelNumericEditField3          matlab.ui.control.Label
etBraidWidth                    matlab.ui.control.NumericEditField
LabelNumericEditField4          matlab.ui.control.Label
etKnotsinSpline                 matlab.ui.control.NumericEditField
LabelNumericEditField5          matlab.ui.control.Label
etBraidDepth                    matlab.ui.control.NumericEditField
LabelNumericEditField6          matlab.ui.control.Label
etSpacingFactor                 matlab.ui.control.NumericEditField
LabelNumericEditField7          matlab.ui.control.Label
etNumofUnitCells                matlab.ui.control.NumericEditField
LabelEditField                  matlab.ui.control.Label
etFolderName                    matlab.ui.control.EditField
```

```
        LabelDropDown                   matlab.ui.control.Label
        ddFiberSelect                   matlab.ui.control.DropDown
        LabelDropDown2                  matlab.ui.control.Label
        ddMatrixSelect                  matlab.ui.control.DropDown
        LabelNumericEditField9          matlab.ui.control.Label
        etFiberE1                       matlab.ui.control.NumericEditField
        Label                           matlab.ui.control.Label
        etFiberv                        matlab.ui.control.NumericEditField
        Label2                          matlab.ui.control.Label
        etFiberG                        matlab.ui.control.NumericEditField
        Label3                          matlab.ui.control.Label
        etMatrixE                       matlab.ui.control.NumericEditField
        Label4                          matlab.ui.control.Label
        etMatrixv                       matlab.ui.control.NumericEditField
        Label5                          matlab.ui.control.Label
        etMatrixG                       matlab.ui.control.NumericEditField
        Label33                         matlab.ui.control.Label
        etFiberE2                       matlab.ui.control.NumericEditField
        DenierEditFieldLabel            matlab.ui.control.Label
        etDenier                        matlab.ui.control.NumericEditField
        UnitCellHeightPitchLabel        matlab.ui.control.Label
        etZHeight                       matlab.ui.control.NumericEditField
        BraidAngleAverageLabel          matlab.ui.control.Label
        etTAngle                        matlab.ui.control.NumericEditField
        BraidControl                    matlab.ui.container.Tab
        CONNECTIONPanel                 matlab.ui.container.Panel
        CONNECTButton                   matlab.ui.control.StateButton
        COMPORTDropDownLabel            matlab.ui.control.Label
        COMPORTDropDown                 matlab.ui.control.DropDown
        CONNECTIONDisplay               matlab.ui.control.EditField
        StartBraidingButton             matlab.ui.control.Button
        PauseButton                     matlab.ui.control.StateButton
    end


    properties (Access = private)
        Knots = 10;
        Path = 0.0;
        Denier = 1500;
        Density = 1.32629;
        Radius = 0.2;
        braidLength = 3;
        braidWidth = 3;
        unitCells = 3;
        Spacing_Factor = 1;
        enableSmoothing = true;
        enableYarns = false
        folderName = 'default';
        Vf = 1;
```

```
        yarnAngle = 0;
        curDir
        Materials
        FGMData
        CamMatrix = zeros(2*7+1); %The cam matrix provides the initial setup of
cams and yarns before braiding begins
        Nyarn=24; %This is a count of the number of yarns active with a given cam
setup
        DefinePerimeterApp %vertex plot app
        Perimeter=[]; %Contains the vertices necessary to define the perimeter of
the total braid (yarn+matrix) for Solidworks implementation
        HolePerim=[]; %Same as perimeter except for the hole rather than outside
perimeter
        CenterX %Defines the center of the braid relative to the true center of
the braiding machine (Note x,y coords not row,col)
        CenterY %Defines the center of the braid relative to the true center of
the braiding machine (Note x,y coords not row,col)
        Area %The area of the polygon defined by perimeter above (equivalent to
total cross-sectional area of the braid)
        MotorStateMatrix = ones(1,49); %This is a temporary matrix used to
determine the which motor state button is pushed
        MotorDirectionMatrix = zeros(1,49); %This is a temporary matrix used to
determine the which motor direction button is pushed
        Step = struct; %This structure contains all steps of the current braid
design.
        % Subsections are motor (structure) and takeup (logical)
        motor = struct; %This structure contains the enable/disable and direction
for each motor in a given step
        StepList={} % This cell array contains the labels for the drop down list
"saved steps"

        %Arduino connection properties:
        serverObject
        comPort = 3;
        baudRate = 115000;
        errorCatch
        this = struct %Structure to contain arduino object
        StepCounter % Used to index current step being sent to arduino (allows
for pausing)
        Step4Arduino % See start braiding button pushed callback for description
        TotalSteps % Total number of braiding steps
    end

    methods (Access = private)

        function [] = updateLamina(app)

            Em = app.etMatrixE.Value;
            vm = app.etMatrixv.Value;
            Gm = app.etMatrixG.Value;
```

```
            E1 = app.etYarnE1.Value;
            E2 = app.etYarnE2.Value;
            v12 = app.etYarnv12.Value;
            G12 = app.etYarnG12.Value;

            theta = app.yarnAngle;
            Vm = 1-app.Vf;

            kf = E1/(2*(1+v12)*(1-2*v12));
            km = Em/(2*(1+vm)*(1-2*vm));
            k  = (km*(kf+Gm)*Vm + kf*(km+Gm)*app.Vf)/...
                ((kf+Gm)*Vm + (km+Gm)*app.Vf);

            E1 = app.Vf*E1 + Vm*Em;
            E2 = 1/(app.Vf/E2 + Vm/Em);
            E3 = E2;

            v12 = app.Vf*v12 + Vm*vm;
            v13 = v12;

            m = 1+4*k*v12^2/E1;

            v23 = @(G23) (k - m*G23)/(k + m*G23);

            G12 = 1/(app.Vf/G12 + Vm/Gm);
            G13 = G12;
            G23 = max(double(solve(@(G23) 2*(1-v23(G23))*G23 - E2))); %Check this
line 2018-11-19

            v23 = v23(G23);


            app.etLaminaE1.Value = E1;
            app.etLaminav12.Value = v12;
            app.etLaminaG12.Value = G12;
            app.etLaminaE2.Value = E2;
            app.etLaminav13.Value = v13;
            app.etLaminaG13.Value = G13;
            app.etLaminaE3.Value = E3;
            app.etLaminav23.Value = v23;
            app.etLaminaG23.Value = G23;

            E1 = app.etYarnE1.Value;
            E2 = app.etYarnE2.Value;
            v12 = app.etYarnv12.Value;
            G12 = app.etYarnG12.Value;
            app.Vf = app.Vf*cosd(theta);
            Vm = 1-app.Vf;
```

```
            kf = E1/(2*(1+v12)*(1-2*v12));
            km = Em/(2*(1+vm)*(1-2*vm));
            k  = (km*(kf+Gm)*Vm + kf*(km+Gm)*app.Vf)/...
                ((kf+Gm)*Vm + (km+Gm)*app.Vf);

            E1 = app.Vf*E1 + Vm*Em;
            E2 = 1/(app.Vf/E2 + Vm/Em);
            E3 = E2;

            v12 = app.Vf*v12 + Vm*vm;
            v13 = v12;

            m = 1+4*k*v12^2/E1;

            v23 = @(G23) (k - m*G23)/(k + m*G23);

            G12 = 1/(app.Vf/G12 + Vm/Gm);
            G13 = G12;
            G23 = max(double(solve(@(G23) 2*(1-v23(G23))*G23 - E2)));

            v23 = v23(G23);

            T = @(angle)[...
                cosd(angle)^2 sind(angle)^2 0 0 0 2*sind(angle)*cosd(angle)
                sind(angle)^2 cosd(angle)^2 0 0 0 -2*sind(angle)*cosd(angle)
                0 0 1 0 0 0
                0 0 0 cosd(angle) sind(angle) 0
                0 0 0 -sind(angle) cosd(angle) 0
                -sind(angle)*cosd(angle) sind(angle)*cosd(angle) 0 0 0
cosd(angle)^2-sind(angle)^2];

            R = diag([1 1 1 2 2 2]);

            S = [...
                1/E1   -v12/E1  -v13/E1    0      0      0
                -v12/E1    1/E2   -v23/E2    0      0      0
                -v13/E1  -v23/E2    1/E3     0      0      0
                0        0        0     1/G23    0      0
                0        0        0       0    1/G13    0
                0        0        0       0      0    1/G12];

            Sb = T(theta)'*S*T(theta);

            E.x = 1/Sb(1,1);
            E.y = 1/Sb(2,2);
            E.z = 1/Sb(3,3);
            v.xy = -Sb(1,2)*E.x;
            v.yx = -Sb(2,1)*E.y;
            v.xz = -Sb(1,3)*E.x;
            v.zx = -Sb(3,1)*E.z;
```

```
            v.yz = -Sb(2,3)*E.y;
            v.zy = -Sb(3,2)*E.z;
            G.xy = 1/Sb(6,6);
            G.xz = 1/Sb(5,5);
            G.yz = 1/Sb(4,4);

            app.etAngleLaminaEx.Value = E.x;
            app.etAngleLaminavxy.Value = v.xy;
            app.etAngleLaminaGxy.Value = G.xy;
            app.etAngleLaminaEy.Value = E.y;
            app.etAngleLaminavxz.Value = v.xz;
            app.etAngleLaminaGxz.Value = G.xz;
            app.etAngleLaminaEz.Value = E.z;
            app.etAngleLaminavyz.Value = v.yz;
            app.etAngleLaminaGyz.Value = G.yz;

            app.tbLamina.Value = sprintf('%s/%s %.2f (Vy: %0.3f, PF: %0.3f)',...
                app.ddFiberSelect.Value,...
                app.ddMatrixSelect.Value,...
                0,...
                app.Vf,...
                app.etYarnPackingFraction.Value);
            app.tbAngleLamina.Value = sprintf('%s/%s %.2f (Vy: %0.3f, PF:
%0.3f)',...
                app.ddFiberSelect.Value,...
                app.ddMatrixSelect.Value,...
                theta,...
                app.Vf,...
                app.etYarnPackingFraction.Value);
        end

        function [] = printResults(app,enablePlot)
            if nargin==1
                enablePlot = false;
            else
                enablePlot = true;
            end
            filename = [app.folderName,'\BRAID.PROPERTIES'];
            fID = fopen(filename,'W','native','UTF-8');
            Ef1= app.etFiberE1.Value;
            Ef2= app.etFiberE2.Value;
            vf = app.etFiberv.Value;
            Gf = app.etFiberG.Value;
            Em = app.etMatrixE.Value;
            vm = app.etMatrixv.Value;
            Gm = app.etMatrixG.Value;

            Ey1 = app.etYarnE1.Value;
            vy12 = app.etYarnv12.Value;
            Gy12 = app.etYarnG12.Value;
```

```
            Ey2 = app.etYarnE2.Value;
            vy13 = app.etYarnv13.Value;
            Gy13 = app.etYarnG13.Value;
            Ey3 = app.etYarnE3.Value;
            vy23 = app.etYarnv23.Value;
            Gy23 = app.etYarnG23.Value;

            E1 = app.etLaminaE1.Value;
            v12 = app.etLaminav12.Value;
            G12 = app.etLaminaG12.Value;
            E2 = app.etLaminaE2.Value;
            v13 = app.etLaminav13.Value;
            G13 = app.etLaminaG13.Value;
            E3 = app.etLaminaE3.Value;
            v23 = app.etLaminav23.Value;
            G23 = app.etLaminaG23.Value;

            E.x = app.etAngleLaminaEx.Value;
            v.xy = app.etAngleLaminavxy.Value;
            G.xy = app.etAngleLaminaGxy.Value;
            E.y = app.etAngleLaminaEy.Value;
            v.xz = app.etAngleLaminavxz.Value;
            G.xz = app.etAngleLaminaGxz.Value;
            E.z = app.etAngleLaminaEz.Value;
            v.yz = app.etAngleLaminavyz.Value;
            G.yz = app.etAngleLaminaGyz.Value;

            theta = app.yarnAngle;
            bn = app.braidWidth;
            bm = app.braidLength;
            SF = app.Spacing_Factor+app.enableYarns*(1/sind(45+app.Path)-
1.0+app.Path/180);
            Pf = app.etYarnPackingFraction.Value;
            Fiber  = upper(app.ddFiberSelect.Value);
            Matrix = upper(app.ddMatrixSelect.Value);
            yarnRadius = app.Radius;

            Sx = 2*yarnRadius*(2*bn*SF +1.5);
            Sy = 2*yarnRadius*(2*bm*SF +1.5);
            Ny=app.Nyarn;
            eta = Ny*pi*yarnRadius^2/(Sx*Sy);

            app.BraidCrossSectionalDimsEditField.Value = sprintf('%.3f X
%.3f',Sx,Sy);
            app.UnitCellHeigthEditField.Value = num2str(app.etZHeight.Value);
            app.AverageBraidingAngleEditField.Value = num2str(app.yarnAngle);
            app.InteriorBraidingAngleEditField.Value =...

num2str(atand(4*sqrt(2)*app.Radius*app.Spacing_Factor/app.etZHeight.Value));
            app.SurfaceBraidingAngleEditField.Value =...
```

```
num2str(atand(2*sqrt(2)*app.Radius*app.Spacing_Factor/app.etZHeight.Value));
            app.BraidingTightnessEditField.Value = num2str(eta);
            app.FiberVolumeFractionEditField.Value = num2str(app.Vf*Pf);
            app.NumberofYarnsEditField.Value = num2str(Ny);
            app.YoungsModulusRangeEditField.Value = sprintf('%.3f -
%.3f',E.x,E1);

            app.FGM(Ny, Sx*Sy)

            printHeader(fID)


printSummary(fID,Fiber,Matrix,Pf,app.Vf,theta,E.x,E1,app.etFGMEx.Value,bn,bm,SF,.
..

yarnRadius,app.Knots,app.enableSmoothing,app.enableYarns,app.etZHeight.Value)

            printFiberProps(fID,Ef1,Ef2,vf,Gf,['Fiber Properties: ',Fiber],...
                app.Density,app.Denier,yarnRadius)

            printIso(fID, Em,vm,Gm,['Matrix Properties: ' ,Matrix])

            printOrtho(fID, Ey1,Ey2,Ey3,vy12,vy13,vy23,Gy12,Gy13,Gy23,...
                sprintf('Yarn Properties: %s/%s (PF: %.3f)',Fiber,Matrix,Pf))

            printOrtho(fID, E1,E2,E3,v12,v13,v23,G12,G13,G23,...
                sprintf('%s/%s 0.00 deg (Vy: %.3f, PF:
%.3f)',Fiber,Matrix,app.Vf,Pf))

            printOrtho(fID,E.x,E.y,E.z,v.xy,v.xz,v.yz,G.xy,G.xz,G.yz,...
                sprintf('%s/%s %.2f deg (Vy: %.3f, PF: %.3f)',...
                Fiber,Matrix,theta,app.Vf,Pf))

            printOrtho(fID,...
                app.etFGMEx.Value,...
                app.etFGMEy.Value,...
                app.etFGMEz.Value,...
                app.etFGMvxy.Value,...
                app.etFGMvxz.Value,...
                app.etFGMvyz.Value,...
                app.etFGMGxy.Value,...
                app.etFGMGxz.Value,...
                app.etFGMGyz.Value,...
                'Fabric Geometery Model')

            fclose(fID);

            if enablePlot
                figure;
```

```
        fplot(Pf,[0,90]);
        hold on
        Sx = 2*yarnRadius*(2*bn*SF +1.5);
        Sy = 2*yarnRadius*(2*bm*SF +1.5);
        Ny = (2+app.enableYarns)*bn*bm+bn+bm;
        eta = Ny*pi*yarnRadius^2/(Sx*Sy);
        fplot(@(t)Pf.*eta./cosd(t),[0,acosd(sqrt(eta/(pi/2-eta)))]);
        fplot(@(t)pi./2.*Pf.*cosd(t)./(1+cosd(t).^2),[0,90]);

        plot(theta,Pf*eta/cosd(theta),'ok');
        plot([0,theta],[Pf*eta/cosd(theta),Pf*eta/cosd(theta)],'--k');
        plot([theta,theta],[0,Pf*eta/cosd(theta)],'--k');

        axis([0 90 0 1]);
        xlabel('Braiding Angle, \theta');
        ylabel('Fiber Volume Fraction, V_f');
        text(45,Pf+0.05,sprintf('Yarn Packing Fraction = %.2f',Pf),...
            'HorizontalAlignment','center')
    end

end

function [] = FGM(app,Ny, Area)
    E1 = app.etYarnE1.Value;
    v12 = app.etYarnv12.Value;
    G12 = app.etYarnG12.Value;
    E2 = app.etYarnE2.Value;
    v13 = app.etYarnv13.Value;
    G13 = app.etYarnG13.Value;
    E3 = app.etYarnE3.Value;
    v23 = app.etYarnv23.Value;
    G23 = app.etYarnG23.Value;

    Em = app.etMatrixE.Value;
    vm = app.etMatrixv.Value;
    Gm = app.etMatrixG.Value;

    S  = [...
        1/E1   -v12/E1  -v13/E1    0      0      0
        -v12/E1    1/E2   -v23/E2    0      0      0
        -v13/E1  -v23/E2    1/E3     0      0      0
        0        0        0      1/G23    0      0
        0        0        0        0    1/G13    0
        0        0        0        0      0    1/G12];
    Sm = [...
        1/Em    -vm/Em  -vm/Em    0      0      0
        -vm/Em     1/Em   -vm/Em    0      0      0
        -vm/Em    -vm/Em    1/Em    0      0      0
        0        0        0      1/Gm    0      0
        0        0        0        0    1/Gm    0
```

```
                0        0        0       0      0    1/Gm];

            A = @(theta,beta) [1 0 0; 0 cosd(beta) sind(beta); 0 -sind(beta)
cosd(beta)]*...
                [cosd(theta) -sind(theta) 0;sind(theta) cosd(theta) 0; 0 0 1];

            Ta = @(theta,beta)A(theta,beta).^2;
            Tb = 
@(theta,beta)2*circshift(A(theta,beta),1,2).*circshift(A(theta,beta),2,2);
            Tc = 
@(theta,beta)circshift(A(theta,beta),1,1).*circshift(A(theta,beta),2,1);
            Td = 
@(theta,beta)circshift(A(theta,beta),[1,1]).*circshift(A(theta,beta),[2,2]) +...
                circshift(A(theta,beta),[1,2]).*circshift(A(theta,beta),[2,1]);
            T  = @(theta,beta) [Ta(theta,beta),Tb(theta,beta);...
                Tc(theta,beta),Td(theta,beta)];

            for m = 1:2
                C = zeros(size(S));
                for n = 1:Ny
                    tFGM = app.FGMData(n+(m-1)*Ny,2);
                    bFGM = app.FGMData(n+(m-1)*Ny,1);
                    C = C + 
inv(T(tFGM,bFGM)'*S*T(tFGM,bFGM))*pi*app.Radius^2/cosd(tFGM)/Area;


                end
                C = C + inv(Sm)*(Area - sum(app.Radius./cosd(app.FGMData((m-
1)*Ny+1:m*Ny,2))))/Area;
                switch m
                    case 1
                        C1 = C;
                    case 2
                        C2 = C;
                end
            end
            Snew = inv(C1/2+C2/2);
            Snew(abs(Snew)<1e-5) = 0;

            app.etFGMEx.Value = 1/Snew(1,1);
            app.etFGMEy.Value   = 1/Snew(2,2);
            app.etFGMEz.Value   = 1/Snew(3,3);
            app.etFGMvxy.Value  = -Snew(1,2)*app.etFGMEx.Value;
            app.etFGMGxy.Value  = 1/Snew(6,6);
            app.etFGMvxz.Value  = -Snew(1,3)*app.etFGMEx.Value;
            app.etFGMGxz.Value  = 1/Snew(5,5);
            app.etFGMvyz.Value  = -Snew(2,3)*app.etFGMEy.Value;
            app.etFGMGyz.Value  = 1/Snew(4,4);
        end
```

```
%This function changes text and cosmetics within the GUI (tab 2)
%for motor setup
function [] = ToggleMotorState(app, MotorNum, Enable)

    %Defines the button names based on the button that was pushed
    StateName=['MotorStateButton_' num2str(MotorNum)];
    DirectionName=['MotorDirectionButton_' num2str(MotorNum)];
    RotationName=['MotorRotationSpinner_' num2str(MotorNum)];

    %Changes cosmetics of motor state button and enables/disables
    %direction and rotation button.
    %Note that "enabled" is given a value of 0 and "disabled" is
    %given a value of 1 to correspond with the values needed to
    %send to the motor driver
    if Enable==0
        app.(StateName).Value=0;
        app.(StateName).Text = 'Enabled';
        app.(StateName).BackgroundColor = [0 1 0];
        app.(DirectionName).Enable = 'on';
        app.(RotationName).Enable = 'on';
        app.MotorStateMatrix(MotorNum)=0;
    else
        app.(StateName).Value=1;
        app.(StateName).Text = 'Disabled';
        app.(StateName).BackgroundColor = [1 0 0];
        app.(DirectionName).Enable = 'off';
        app.(RotationName).Enable = 'off';
        app.MotorStateMatrix(MotorNum)=1;
    end


end

function []= ToggleMotorDirection(app,MotorNum,Direction)
    DirectionName=['MotorDirectionButton_' num2str(MotorNum)];
    if MotorNum==0
        DirectionName='MasterMotorDirectionButton';
    else
        app.MotorDirectionMatrix(MotorNum)=Direction;
    end

    app.(DirectionName).Value=Direction;

    %Changes cosmetics of motor direction button

    %Note that "CCW" is given a value of 0 and "CW" is
    %given a value of 1 to correspond with the values needed to
    %send to the motor driver
    if Direction
        app.(DirectionName).Text = 'CW';
```

```
                app.(DirectionName).BackgroundColor = [0 0 0];
                app.(DirectionName).FontColor = [1 1 1];
            else
                app.(DirectionName).Text = 'CCW';
                app.(DirectionName).BackgroundColor = [1 1 1];
                app.(DirectionName).FontColor = [0 0 0];
            end
        end

        function [] = ToggleTakeUpControl(app,value)
            if value
                app.TakeUpControlButton.Text = 'Active';
                app.TakeUpControlButton.BackgroundColor = [0 1 0];
                app.TakeUpControlButton.Value=1;
            else
                app.TakeUpControlButton.Text = 'Paused';
                app.TakeUpControlButton.BackgroundColor = [1 0 0];
                app.TakeUpControlButton.Value=0;
                app.TakeUpSpeedEditField.Value=0;
            end
        end

        function TogglePauseButton(app,enable,value)
            if enable
                app.PauseButton.Enable = true;
            else
                app.PauseButton.Enable = false;
            end

            if value
                app.PauseButton.Text = 'Resume';
                app.PauseButton.Value = 1;
            else
                app.PauseButton.Text = 'Pause';
                app.PauseButton.Value = 0;
            end
        end

        function attemptDisconnect(app, ~)

            fwrite(app.this.arduino,250,'uchar'); %stops takeup motors

            %Changes visual appearance and functionality of connect button
            app.CONNECTButton.ValueChangedFcn = createCallbackFcn(app,
@attemptConnection, true);
            app.CONNECTButton.BackgroundColor = [0,1,0];
            app.CONNECTButton.Text = 'CONNECT';

            %disable pause button
            TogglePauseButton(app,0,1);
```

```
            %delete serial connection with arduino
            instrreset;
        end

        function braidControl(app,~)
            while ~(app.PauseButton.Value) %Pauses if Pause Button pressed
                if app.this.arduino.BytesAvailable %Check if data has been sent
(1 byte)
                    load = fscanf( app.this.arduino, '%c'); %Reads data sent from
arduino
                    if strcmpi(load, 'Send Next Step Please') %Arduino sends this
line when looking for data
                        app.StepCounter = app.StepCounter+1;
                        if app.StepCounter <= app.TotalSteps
                            %Sends the current step in an array to Arduino.
                            %Arduino reads it as individual bytes
                            fwrite(
app.this.arduino,app.Step4Arduino(app.StepCounter,:),'uchar');
                        else
                            fprintf('Braid Completed \n')
                            attemptDisconnect(app);
                        end
                    else
                        fprintf('%c',load)
                    end
                end
                %Pauses loop for 10^-12 seconds to allow other
                %functions to be processed
                %This allows the pause button value to change if
                %pressed
                pause(10^-12)
            end

        end


    end


    methods (Access = public)

        function PlotPerimeter(app,Vertices,HoleVertices)

            %Ensure closed loop (important for area calculation)
            h1=size(Vertices,1);
            Vertices((h1+1),:)=Vertices(1,:);
            h1=size(Vertices,1);
```

```
            %Center perimeter coordinates around center of active cams
            %Note: the center of active cams is not necessarily the centroid of
the shape
            %   this is just for plotting and Solidworks purposes
            app.Perimeter=zeros(h1,2);
            app.Perimeter(1:h1,1)=-(Vertices(:,2)-app.CenterY);
            app.Perimeter(1:h1,2)=Vertices(:,3)-app.CenterX;

            %Plot perimeter to allow user to ensure shape is correct
            plot(app.Perimeter(:,2),app.Perimeter(:,1))

            %Convert for Solidworks implementation

            %This factor relates to the maximum size possible with a 7x7 setup
            %With 7x7 setup there are a max of 8 yarns and 7 spaces in any
direction
            %This factor takes into account that the center is 0,0
            %It is multiplied by 100 for display in Solidworks
            SWFactor=(1/7*(4+3.5+1.5*app.Spacing_Factor)*app.Radius)/10;

            %This step applies the above factor and adds a small amount to the
outer border

app.Perimeter=app.Perimeter*SWFactor+((app.Perimeter./abs(app.Perimeter))*1)/10;
%Doesn't work if vertex on zero axis


            %Calculate the area of the polygon defined by perimeter above
(equivalent to
            %total cross-sectional area of the braid)
            OutX=app.Perimeter(:,2)/100; %Divided by 100 to get true dimensions
rather than SW optimized values
            OutY=app.Perimeter(:,1)/100;


            app.Area=polyarea(OutX,OutY);

            %Computes values as above for a hole, if present
            if HoleVertices~=0
                h2=size(HoleVertices,1);
                HoleVertices((h2+1),:)=HoleVertices(1,:);
                h2=size(HoleVertices,1);

                app.HolePerim=zeros(h2,2);
                app.HolePerim(1:h2,1)=-(HoleVertices(:,2)-app.CenterY);
                app.HolePerim(1:h2,2)=HoleVertices(:,3)-app.CenterX;

                hold on
                plot(app.HolePerim(:,2),app.HolePerim(:,1))
                hold off
```

```
                app.HolePerim=app.HolePerim*SWFactor-
(app.HolePerim./abs(app.HolePerim))*50;


                HoleX=app.HolePerim(:,2)/100;
                HoleY=app.HolePerim(:,1)/100;

                app.Area=polyarea(OutX,OutY)-polyarea(HoleX,HoleY);
            end


        end

    end



    % Callbacks that handle component events
    methods (Access = private)


        % Code that executes after component creation
        function startupFcn(app)
            app.curDir = cd;

            app.Materials = xmlParse('Materials.lib');

            app.ddFiberSelect.Items = fieldnames(app.Materials.Fiber);

            app.ddMatrixSelect.Items = fieldnames(app.Materials.Matrix);
            value = app.ddFiberSelect.Value;
            E1 = app.Materials.Fiber.(value).E1;
            E2 = app.Materials.Fiber.(value).E2;
            v  = app.Materials.Fiber.(value).v;
            G  = app.Materials.Fiber.(value).G;
            app.Denier = app.Materials.Fiber.(value).Denier;
            app.etDenier.Value = app.Denier;
            app.Density = app.Materials.Fiber.(value).Density;
            app.Radius = sqrt(app.Denier/(9000*pi()*app.Density));
            app.etYarnRadius.Value = app.Radius;
            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = app.etZHeight.Value;
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
```

```
            app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) +
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
            app.etFiberE1.Value = E1;
            app.etFiberE2.Value = E2;
            app.etFiberv.Value = v;
            app.etFiberG.Value = G;
            value = app.ddMatrixSelect.Value;
            E = app.Materials.Matrix.(value).E;
            v = app.Materials.Matrix.(value).v;
            G = E/(2*(1+v));

            app.etMatrixE.Value = E;
            app.etMatrixv.Value = v;
            app.etMatrixG.Value = G;
            app.PF()

            %Activates cams on startup
            app.PopulateCarriers

            %Disables some buttons for carrier setup panel
            %These buttons are present for the purpose of creating a
            %matrix. Not physically possible to put carriers there
            for
i=[1:2:15,31:2:45,61:2:75,91:2:105,121:2:135,151:2:165,181:2:195,211:2:225]
                name=['Button_' num2str(i)];
                app.(name).Enable='off';
            end

        end


        % Value changed function: cbDenier
        function DenierChange(app, event)
            value = app.cbDenier.Value;

            if value
                app.etDenier.Editable = 'on';
            else
                app.etDenier.Editable = 'off';
                value = app.ddFiberSelect.Value;
                app.Denier  = app.Materials.Fiber.(value).Denier;
                app.etDenier.Value = app.Materials.Fiber.(value).Denier;
                app.Radius = sqrt(app.Denier/(9000*pi()*app.Density));
                app.etYarnRadius.Value = app.Radius;
                n = app.braidLength;
                m = app.braidWidth;
                R = app.Radius;
                SF = app.Spacing_Factor;
```

```
                z = app.etZHeight.Value;
                app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                    (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
                app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
                app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) +
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
                app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
            end

        end


        % Value changed function: etDenier
        function DenierValue(app, event)
            value = round(app.etDenier.Value);
            app.etDenier.Value = value;
            app.Denier = value;
            app.Radius = sqrt(app.Denier/(9000*pi()*app.Density));
            app.etYarnRadius.Value = app.Radius;
            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = app.etZHeight.Value;
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
            app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) +
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
        end


        % Value changed function: etYarnPackingFraction
        function PF(app, event)
            value = app.etYarnPackingFraction.Value;

            Ef1= app.etFiberE1.Value;
            Ef2= app.etFiberE2.Value;
            vf = app.etFiberv.Value;
            Gf = app.etFiberG.Value;
            Em = app.etMatrixE.Value;
            vm = app.etMatrixv.Value;
            Gm = app.etMatrixG.Value;

            Pf = value;
```

```
            Vm = 1-Pf;

            kf = Ef1/(2*(1+vf)*(1-2*vf));
            km = Em/(2*(1+vm)*(1-2*vm));
            k  = (km*(kf+Gm)*Vm + kf*(km+Gm)*Pf)/...
                ((kf+Gm)*Vm + (km+Gm)*Pf);

            E1 = Pf*Ef1 + Vm*Em;
            E2 = 1/(Pf/Ef2 + Vm/Em);
            E3 = E2;

            v12 = Pf*vf + Vm*vm;
            v13 = v12;

            m = 1+4*k*v12^2/E1;

            v23 = @(G23) (k - m*G23)/(k + m*G23);

            G12 = 1/(Pf/Gf + Vm/Gm);
            G13 = G12;
            G23 = max(double(solve(@(G23) 2*(1-v23(G23))*G23 - E2)));

            v23 = v23(G23);
            app.etYarnE1.Value = E1;
            app.etYarnv12.Value = v12;
            app.etYarnG12.Value = G12;
            app.etYarnE2.Value = E2;
            app.etYarnv13.Value = v13;
            app.etYarnG13.Value = G13;
            app.etYarnE3.Value = E3;
            app.etYarnv23.Value = v23;
            app.etYarnG23.Value = G23;
            app.updateLamina()
        end


        % Value changed function: etTAngle
        function TAngleChange(app, event)
            value = app.etTAngle.Value;
            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = 4*sqrt(2)*R*SF/tand(value);
            app.etZHeight.Value = z;
            app.UnitCellHeigthEditField.Value = num2str(app.etZHeight.Value);
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
```

```
            app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
        end


        % Value changed function: etBraidDepth
        function braidL(app, event)
            %Old function. Made obsolete by ActivateCams

            value = app.etBraidDepth.Value;
            app.braidLength = round(value);
            bm = app.braidLength;
            bn = app.braidWidth;
            SF = app.Spacing_Factor;
            Sx = 2*app.Radius*(2*bn*SF +1.5);
            Sy = 2*app.Radius*(2*bm*SF +1.5);
            app.BraidCrossSectionalDimsEditField.Value = sprintf('%.3f X
%.3f',Sx,Sy);
            app.NumberofYarnsEditField.Value =
num2str((2+app.enableYarns)*bn*bm+bn+bm);

            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = app.etZHeight.Value;
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
            app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) +
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
        end


        % Value changed function: etBraidWidth
        function braidW(app, event)
            %Old function. Made obsolete by ActivateCams

            value = app.etBraidWidth.Value;
            app.braidWidth = round(value);
            bm = app.braidLength;
            bn = app.braidWidth;
            SF = app.Spacing_Factor;
            Sx = 2*app.Radius*(2*bn*SF +1.5);
            Sy = 2*app.Radius*(2*bm*SF +1.5);
            app.BraidCrossSectionalDimsEditField.Value = sprintf('%.3f X
%.3f',Sx,Sy);
```

```
            app.NumberofYarnsEditField.Value =
num2str((2+app.enableYarns)*bn*bm+bn+bm);

            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = app.etZHeight.Value;
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
            app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) +
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
        end


        % Button pushed function: btClosePlots
        function closePlots(app, event)
            close all;
        end


        % Value changed function: cbSmoothing
        function enSmoothing(app, event)
            value = app.cbSmoothing.Value;
            app.enableSmoothing = value;
        end


        % Value changed function: cbTriAxial
        function enTriAxial(app, event)
            value = app.cbTriAxial.Value;
            app.enableYarns = value;
            bm = app.braidLength;
            bn = app.braidWidth;
            SF = app.Spacing_Factor;
            Sx = 2*app.Radius*(2*bn*SF +1.5);
            Sy = 2*app.Radius*(2*bm*SF +1.5);
            app.BraidCrossSectionalDimsEditField.Value = sprintf('%.3f X
%.3f',Sx,Sy);
            app.NumberofYarnsEditField.Value =
num2str((2+app.enableYarns)*bn*bm+bn+bm);
        end


        % Value changed function: ddFiberSelect
        function fiberChange(app, event)
```

```
            value = app.ddFiberSelect.Value;
            E1 = app.Materials.Fiber.(value).E1;
            E2 = app.Materials.Fiber.(value).E2;
            v  = app.Materials.Fiber.(value).v;
            G  = app.Materials.Fiber.(value).G;
            D  = app.Materials.Fiber.(value).Denier;
            app.etDenier.Value = D;
            rho= app.Materials.Fiber.(value).Density;
            app.etFiberE1.Value  = E1;
            app.etFiberE2.Value = E2;
            app.etFiberv.Value = v;
            app.etFiberG.Value = G;
            app.Density = rho;
            app.Denier = D;
            app.Radius = sqrt(app.Denier/(9000*pi()*app.Density));
            app.etYarnRadius.Value = app.Radius;
            bm = app.braidLength;
            bn = app.braidWidth;
            SF = app.Spacing_Factor;
            Sx = 2*app.Radius*(2*bn*SF +1.5);
            Sy = 2*app.Radius*(2*bm*SF +1.5);
            app.BraidCrossSectionalDimsEditField.Value = sprintf('%.3f X 
%.3f',Sx,Sy);
            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = app.etZHeight.Value;
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
            app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) + 
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
            app.PF()
        end


        % Value changed function: etFolderName
        function folder(app, event)
            value = app.etFolderName.Value;
            if isempty(value)
                app.etFolderName.Value = app.folderName;
            else
                app.folderName = value;
            end
        end
```

```
        % Button pushed function: btGenerateModel
        function genModel(app, event)
            if app.cbType2Paths.Value
                [~,app.Vf,app.yarnAngle,app.FGMData] =
generateBraidPaths(app.Knots,app.Path,app.braidLength,app.braidWidth,app.Radius,.
..

app.Spacing_Factor,app.unitCells,app.folderName,app.enableSmoothing,app.enableYar
ns,false,false,true,app.etZHeight.Value,...
                    app.CamMatrix,app.Area,app.Perimeter,app.HolePerim);
            else
                [~,app.Vf,app.yarnAngle,app.FGMData] =
generateBraidPaths3(app.Knots,app.Path,app.braidLength,app.braidWidth,app.Radius,
...

app.Spacing_Factor,app.unitCells,app.folderName,app.enableSmoothing,app.enableYar
ns,false,false,false,app.etZHeight.Value,...
                    app.CamMatrix,app.Step);
            end
            %app.updateLamina();
            app.printResults();
            system(['copy "',app.curDir,'\Import_Curves.swp"
"',app.curDir,'\',app.folderName,'\Import_Curves.swp"'])
            system(['"C:\Program Files\SOLIDWORKS Corp\SOLIDWORKS\SLDWORKS.exe" -
m "',...
                app.curDir,'\',app.folderName,'\Import_Curves.swp"'])
        end


        % Button pushed function: btGeneratePaths
        function genPaths(app, event)
            if app.cbType2Paths.Value
                [~,app.Vf,app.yarnAngle,app.FGMData] =
generateBraidPaths(app.Knots,app.Path,app.braidLength,app.braidWidth,app.Radius,.
..

app.Spacing_Factor,app.unitCells,app.folderName,app.enableSmoothing,app.enableYar
ns,false,false,true,app.etZHeight.Value,...
                    app.CamMatrix,app.Area,app.Perimeter,app.HolePerim);
            else
                [~,app.Vf,app.yarnAngle,app.FGMData] =
generateBraidPaths(app.Knots,app.Path,app.braidLength,app.braidWidth,app.Radius,.
..

app.Spacing_Factor,app.unitCells,app.folderName,app.enableSmoothing,app.enableYar
ns,false,false,false,app.etZHeight.Value,...
                    app.CamMatrix,app.Area,app.Perimeter,app.HolePerim);
            end
            app.updateLamina();
            app.printResults();
```

```
        end


        % Button pushed function: btGeneratePlots
        function genPlots(app, event)
            close all;
            if app.cbType2Paths.Value
                [~,app.Vf,app.yarnAngle,app.FGMData] =
generateBraidPaths(app.Knots,app.Path,app.braidLength,app.braidWidth,app.Radius,.
..

app.Spacing_Factor,app.unitCells,app.folderName,app.enableSmoothing,app.enableYar
ns,false,true,true,app.etZHeight.Value,...
                    app.CamMatrix,app.Area,app.Perimeter,app.HolePerim);
            else
                [~,app.Vf,app.yarnAngle,app.FGMData] =
generateBraidPaths3(app.Knots,app.Path,app.braidLength,app.braidWidth,app.Radius,
...

app.Spacing_Factor,app.unitCells,app.folderName,app.enableSmoothing,app.enableYar
ns,false,true,false,app.etZHeight.Value,...
                    app.CamMatrix,app.Step);
            end
%             app.updateLamina();
            app.printResults(1);
        end


        % Value changed function: ddMatrixSelect
        function matrixChange(app, event)
            value = app.ddMatrixSelect.Value;
            E = app.Materials.Matrix.(value).E;
            v = app.Materials.Matrix.(value).v;
            G = E/(2*(1+v));

            app.etMatrixE.Value = E;
            app.etMatrixv.Value = v;
            app.etMatrixG.Value = G;
            app.PF()
        end


        % Selection changed function: ButtonGroup
        function modeChange(app, event)
            selectedButton = app.ButtonGroup.SelectedObject;
            switch selectedButton
                case app.enablePitch
                    app.etZHeight.Editable = 'on';
                    app.etTAngle.Editable  = 'off';
                case app.enableAngle
```

```
                app.etZHeight.Editable = 'off';
                app.etTAngle.Editable  = 'on';
            end
        end


        % Value changed function: etPathAngle
        function pathAngle(app, event)
            value = app.etPathAngle.Value;
            app.Path = value;
        end


        % Value changed function: etSpacingFactor
        function sf(app, event)
            value = app.etSpacingFactor.Value;
            app.Spacing_Factor = value;
            bm = app.braidLength;
            bn = app.braidWidth;
            SF = app.Spacing_Factor;
            Sx = 2*app.Radius*(2*bn*SF +1.5);
            Sy = 2*app.Radius*(2*bm*SF +1.5);
            app.BraidCrossSectionalDimsEditField.Value = sprintf('%.3f X
%.3f',Sx,Sy);
            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = app.etZHeight.Value;
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etInteriorAngle.Value = atand((2*sqrt(2)*R*SF)/(0.5*z));
            app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) +
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
        end


        % Value changed function: etKnotsinSpline
        function splineKnots(app, event)
            value = app.etKnotsinSpline.Value;
            app.Knots = round(value);
        end


        % Value changed function: etNumofUnitCells
        function uc(app, event)
            value = app.etNumofUnitCells.Value;
```

```
            app.unitCells = round(value);
        end


        % Value changed function: etZHeight
        function zChange(app, event)
            value = app.etZHeight.Value;
            n = app.braidLength;
            m = app.braidWidth;
            R = app.Radius;
            SF = app.Spacing_Factor;
            z = app.etZHeight.Value;
            app.UnitCellHeigthEditField.Value = num2str(app.etZHeight.Value);
            app.etAverageAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(0.5*z))+...
                (2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etInteriorAngle.Value = atand((4*sqrt(2)*R*SF)/(z));
            app.etTAngle.Value = (2*n*m-n-
m)/(2*n*m+n+m)*atand((4*sqrt(2)*R*SF)/(z)) +
(2*(n+m))/(2*n*m+n+m)*atand((2*sqrt(2)*R*SF)/(z));
            app.etSurfaceAngle.Value  = atand((2*sqrt(2)*R*SF)/(z));
        end


        % Callback function: AxialYarnsButton, PopulateCarriersButton
        function PopulateCarriers(app, event)
           %Loop for turning off carriers that correspond to each cam
           for i=1:49
                CamName=['cam_' num2str(i)];
                if mod(i,7)~=0
                    cam_row=floor(i/7)+1;
                    cam_col=i-floor(i/7)*7;
                else
                    cam_row=floor(i/7);
                    cam_col=i-(floor(i/7)-1)*7;
                end

                top=cam_col*2+(cam_row-1)*30;
                left=15+cam_col*2-1+(cam_row-1)*30;
                right=left+2;
                bottom=top+30;
                axial=left+1;

                if app.(CamName).Value == false
                    for j=[top, right, left, bottom, axial]
                        CarrierName=['Button_' num2str(j)];
                        app.(CarrierName).Value = false;
                    end
%                  elseif app.(CamName).Value == false
%                      for j=[top, right, left, bottom]
```

```
%                          CarrierName=['Button_' num2str(j)];
%                          app.(CarrierName).Value = false;
%                      end
                 end
             end


             %Loop for turning on carriers that correspond to each cam
             %On goes after off due to overlap between cams
             for i=1:49
                 CamName=['cam_' num2str(i)];
                 if mod(i,7)~=0
                      cam_row=floor(i/7)+1;
                      cam_col=i-floor(i/7)*7;
                  else
                      cam_row=floor(i/7);
                      cam_col=i-(floor(i/7)-1)*7;
                 end

                 top=cam_col*2+(cam_row-1)*30;
                 left=15+cam_col*2-1+(cam_row-1)*30;
                 right=left+2;
                 bottom=top+30;
                 axial=left+1;

                 if app.(CamName).Value == true && app.AxialYarnsButton.Value ==
true
                     for j=[top, right, left, bottom, axial]
                          CarrierName=['Button_' num2str(j)];
                          app.(CarrierName).Value = true;
                     end
                 elseif app.(CamName).Value == true && app.AxialYarnsButton.Value
== false
                     for j=[top, right, left, bottom]
                          CarrierName=['Button_' num2str(j)];
                          app.(CarrierName).Value = true;
                     end
                     CarrierName=['Button_' num2str(axial)];
                     app.(CarrierName).Value = false;
                 elseif app.(CamName).Value == true
                     for j=[top, right, left, bottom]
                          CarrierName=['Button_' num2str(j)];
                          app.(CarrierName).Value = true;
                     end
                 end
             end



         end
```

```
% Button pushed function: ActivateAllButton
function ActivateAllButtonPushed(app, event)
    for i=1:49
        name=['cam_' num2str(i)];
        app.(name).Value=1;
    end
end


% Button pushed function: DeactivateAllButton
function DeactivateAllButtonPushed(app, event)
    for i=1:49
        name=['cam_' num2str(i)];
        app.(name).Value=0;
    end
end


% Close request function: UIFigure
function UIFigureCloseRequest(app, event)
    %Close all plots
    close all

    %Close app
    delete(app) %If you delete this line the app can't close

end


% Value changed function: MotorStateButton_1,
% MotorStateButton_10, MotorStateButton_11,
% MotorStateButton_12, MotorStateButton_13,
% MotorStateButton_14, MotorStateButton_15,
% MotorStateButton_16, MotorStateButton_17,
% MotorStateButton_18, MotorStateButton_19,
% MotorStateButton_2, MotorStateButton_20,
% MotorStateButton_21, MotorStateButton_22,
% MotorStateButton_23, MotorStateButton_24,
% MotorStateButton_25, MotorStateButton_26,
% MotorStateButton_27, MotorStateButton_28,
% MotorStateButton_29, MotorStateButton_3,
% MotorStateButton_30, MotorStateButton_31,
% MotorStateButton_32, MotorStateButton_33,
% MotorStateButton_34, MotorStateButton_35,
% MotorStateButton_36, MotorStateButton_37,
% MotorStateButton_38, MotorStateButton_39,
% MotorStateButton_4, MotorStateButton_40,
% MotorStateButton_41, MotorStateButton_42,
```

```
% MotorStateButton_43, MotorStateButton_44,
% MotorStateButton_45, MotorStateButton_46,
% MotorStateButton_47, MotorStateButton_48,
% MotorStateButton_49, MotorStateButton_5,
% MotorStateButton_6, MotorStateButton_7,
% MotorStateButton_8, MotorStateButton_9
function MotorStateButtonValueChanged(app, event)
    %Loop to determine which button was pushed
    NewMotorStateMatrix=zeros(1,49);
    for i=1:49
        name=['MotorStateButton_' num2str(i)];
        NewMotorStateMatrix(i)=app.(name).Value;
    end
    Change=app.MotorStateMatrix-NewMotorStateMatrix;
    app.MotorStateMatrix=NewMotorStateMatrix;
    Num=find(Change);

    %Defines the button name based on the button that was pushed
    StateName=['MotorStateButton_' num2str(Num)];
    value = app.(StateName).Value;
    ToggleMotorState(app,Num,value)


    %The next couple "if" statements are designed to disable all
    %orthogonally adjacent motors when a motor is enabled. This
    %will prevent the machine jamming when motors turn

    %Individual cases for the 4 corner motors

    %Motor 1 enabled (top left)
    if Num==1
        ToggleMotorState(app,2,1)
        ToggleMotorState(app,8,1)


    %Motor 7 enabled (top right)
    elseif Num==7
        ToggleMotorState(app,6,1)
        ToggleMotorState(app,14,1)


    %Motor 43 enabled (bottom left)
    elseif Num==43
        ToggleMotorState(app,36,1)
        ToggleMotorState(app,44,1)


    %Motor 49 enabled (bottom right)
    elseif Num==49
        ToggleMotorState(app,42,1)
```

```
    ToggleMotorState(app,48,1)


%Cases for the remainder of the outside motors

%Motor 2-6 enabled (top row)
elseif Num>=2 && Num <=6
   LeftNum=Num-1;
   RightNum=Num+1;
   BottomNum=Num+7;
   ToggleMotorState(app,LeftNum,1)
   ToggleMotorState(app,RightNum,1)
   ToggleMotorState(app,BottomNum,1)


%Motor 44-48 enabled (bottom row)
elseif Num>=44 && Num <=48
   LeftNum=Num-1;
   RightNum=Num+1;
   TopNum=Num-7;
   ToggleMotorState(app,LeftNum,1)
   ToggleMotorState(app,RightNum,1)
   ToggleMotorState(app,TopNum,1)


%Motor 8,15,22,29,36 enabled (left column)
elseif sum(Num==8:7:36)
   RightNum=Num+1;
   TopNum=Num-7;
   BottomNum=Num+7;
   ToggleMotorState(app,RightNum,1)
   ToggleMotorState(app,TopNum,1)
   ToggleMotorState(app,BottomNum,1)


%Motor 14,21,28,35,42 enabled (right column)
elseif sum(Num==14:7:42)
   LeftNum=Num-1;
   TopNum=Num-7;
   BottomNum=Num+7;
   ToggleMotorState(app,LeftNum,1)
   ToggleMotorState(app,TopNum,1)
   ToggleMotorState(app,BottomNum,1)


%Any other motors enabled
else
   LeftNum=Num-1;
   RightNum=Num+1;
   TopNum=Num-7;
```

```
            BottomNum=Num+7;
            ToggleMotorState(app,LeftNum,1)
            ToggleMotorState(app,RightNum,1)
            ToggleMotorState(app,TopNum,1)
            ToggleMotorState(app,BottomNum,1)
        end

    end


    % Value changed function: MotorDirectionButton_1,
    % MotorDirectionButton_10, MotorDirectionButton_11,
    % MotorDirectionButton_12, MotorDirectionButton_13,
    % MotorDirectionButton_14, MotorDirectionButton_15,
    % MotorDirectionButton_16, MotorDirectionButton_17,
    % MotorDirectionButton_18, MotorDirectionButton_19,
    % MotorDirectionButton_2, MotorDirectionButton_20,
    % MotorDirectionButton_21, MotorDirectionButton_22,
    % MotorDirectionButton_23, MotorDirectionButton_24,
    % MotorDirectionButton_25, MotorDirectionButton_26,
    % MotorDirectionButton_27, MotorDirectionButton_28,
    % MotorDirectionButton_29, MotorDirectionButton_3,
    % MotorDirectionButton_30, MotorDirectionButton_31,
    % MotorDirectionButton_32, MotorDirectionButton_33,
    % MotorDirectionButton_34, MotorDirectionButton_35,
    % MotorDirectionButton_36, MotorDirectionButton_37,
    % MotorDirectionButton_38, MotorDirectionButton_39,
    % MotorDirectionButton_4, MotorDirectionButton_40,
    % MotorDirectionButton_41, MotorDirectionButton_42,
    % MotorDirectionButton_43, MotorDirectionButton_44,
    % MotorDirectionButton_45, MotorDirectionButton_46,
    % MotorDirectionButton_47, MotorDirectionButton_48,
    % MotorDirectionButton_49, MotorDirectionButton_5,
    % MotorDirectionButton_6, MotorDirectionButton_7,
    % MotorDirectionButton_8, MotorDirectionButton_9
    function MotorDirectionButtonValueChanged(app, event)
        %Loop to determine which button was pushed
        NewMotorDirectionMatrix=zeros(1,49);
        for i=1:49
            name=['MotorDirectionButton_' num2str(i)];
            NewMotorDirectionMatrix(i)=app.(name).Value;
        end
        Change=app.MotorDirectionMatrix-NewMotorDirectionMatrix;
        app.MotorDirectionMatrix=NewMotorDirectionMatrix;
        Num=find(Change);

        %Defines the button name based on the button that was pushed
        DirectionName=['MotorDirectionButton_' num2str(Num)];
        value = app.(DirectionName).Value;
        ToggleMotorDirection(app,Num,value)
```

```
end


% Value changed function: MasterMotorRotationSpinner
function MasterMotorRotationSpinnerValueChanged(app, event)
    value = app.MasterMotorRotationSpinner.Value;
    for i=1:49
        StateName=['MotorStateButton_' num2str(i)];
        RotationName=['MotorRotationSpinner_' num2str(i)];
        if app.(StateName).Value==0
            app.(RotationName).Value=value;
        end
    end
end


% Value changed function: MasterMotorDirectionButton
function MasterMotorDirectionButtonValueChanged(app, event)
    value = app.MasterMotorDirectionButton.Value;
    ToggleMotorDirection(app,0,value)
    for i=1:49
        StateName=['MotorStateButton_' num2str(i)];
        if app.(StateName).Value==0
            ToggleMotorDirection(app,i,value)
        end
    end
end


% Button pushed function: SaveCurrentStepButton
function SaveCurrentStepButtonPushed(app, event)
    %This section adds the saved steps as options in the drop down
    %lists for the saved steps and the delete steps drop downs
    n=length(app.StepList)+1;
    app.StepList{n}=['Step ' num2str(n)];
    app.SavedStepsDropDown.Items = app.StepList;
    app.SavedStepsDropDown.Value = app.StepList{n};
    app.DeleteStepDropDown.Items = erase(app.StepList, 'Step ');

    %This loop saves the state of each motor for a given step
    for i=1:49
       StateName=['MotorStateButton_' num2str(i)];
       DirectionName=['MotorDirectionButton_' num2str(i)];
       RotationName=['MotorRotationSpinner_' num2str(i)];
       app.Step(n).motor(i).Enable=app.(StateName).Value;
       app.Step(n).motor(i).Direction=app.(DirectionName).Value;
       app.Step(n).motor(i).Rotation=app.(RotationName).Value;
```

```
            end
            app.Step(n).TakeUp.Enable=~(app.TakeUpControlButton.Value);
            app.Step(n).TakeUp.Speed=app.TakeUpSpeedEditField.Value;
            app.Step(n).TakeUp.Direction =
~(app.ReverseTakeupDirectionCheckBox.Value);

            %Enable delete and repeat buttons
            app.DeleteStepButton.Enable = 'on';
            app.ClearAllStepsButton.Enable = 'on';
            app.RepeatSavedStepsButton.Enable = 'on';
        end


        % Value changed function: TakeUpControlButton
        function TakeUpControlButtonValueChanged(app, event)
            value = app.TakeUpControlButton.Value;
            ToggleTakeUpControl(app,value)
        end


        % Value changed function: SavedStepsDropDown
        function SavedStepsDropDownValueChanged(app, event)
            n = str2num(erase(app.SavedStepsDropDown.Value, 'Step '));
            for i=1:49
               ToggleMotorState(app,i,app.Step(n).motor(i).Enable);
               ToggleMotorDirection(app,i,app.Step(n).motor(i).Direction)
               RotationName=['MotorRotationSpinner_' num2str(i)];
               app.(RotationName).Value=app.Step(n).motor(i).Rotation;
            end
            ToggleTakeUpControl(app,~(app.Step(n).TakeUp.Enable))
            app.TakeUpSpeedEditField.Value=app.Step(n).TakeUp.Speed;
        end


        % Value changed function: PresetsDropDown
        function PresetsDropDownValueChanged(app, event)
            value = app.PresetsDropDown.Value;
            switch value
                case 'None'
                    for i=1:49
                        ToggleMotorState(app,i,1)
                        ToggleMotorDirection(app,i,0)
                        RotationName=['MotorRotationSpinner_' num2str(i)];
                        app.(RotationName).Value=90;
                    end
                case 'Square 7x7 1'
                    for i=1:49
                        ToggleMotorState(app,i,1)
                    end
                    for i=1:2:49
```

```
                ToggleMotorState(app,i,0)
                ToggleMotorDirection(app,i,0)
                RotationName=['MotorRotationSpinner_' num2str(i)];
                app.(RotationName).Value=90;
            end
        case 'Square 7x7 2'
            for i=1:49
                ToggleMotorState(app,i,1)
            end
            for i=2:2:48
                ToggleMotorState(app,i,0)
                ToggleMotorDirection(app,i,1)
                RotationName=['MotorRotationSpinner_' num2str(i)];
                app.(RotationName).Value=90;
            end
    end
end


% Button pushed function: DisableAllButton
function DisableAllButtonPushed(app, event)
    for i=1:49
        ToggleMotorState(app,i,1)
    end
end


% Button pushed function: DeleteStepButton
function DeleteStepButtonPushed(app, event)
    if isempty(app.StepList)
        %Displays error message if there are no other steps to
        %delete and disables the button until more steps are saved
        f = uifigure;
        uialert(f,'No Step to Delete', 'Invalid Action')
        app.DeleteStepButton.Enable = 'off';
        app.ClearAllStepsButton.Enable = 'off';
        app.RepeatSavedStepsButton.Enable = 'off';
    else
        %This function deletes the step determined by the delete step
        %drop down value
        SteptoDelete=str2num(app.DeleteStepDropDown.Value);

        %Delete drop down options
        app.StepList(:,SteptoDelete)=[];

        %Re-order step lists
        for n=1:length(app.StepList)
            app.StepList{n}=['Step ' num2str(n)];

        end
```

```
                app.SavedStepsDropDown.Items = app.StepList;
                app.DeleteStepDropDown.Items = erase(app.StepList, 'Step ');


                %Delete step
                app.Step(SteptoDelete)=[];
            end


        end


        % Value changed function: RepeatStepsEditField_min
        function RepeatStepsEditField_minValueChanged(app, event)
            value = app.RepeatStepsEditField_min.Value;

            %Displays error message if the value selected exceeds the
            %number of steps saved
            if value>length(app.StepList)
                f = uifigure;
                message=sprintf('Value too large \n Step %d does not
exist',value);
                uialert(f,message, 'Invalid Action')
                app.RepeatStepsEditField_min.Value = 1;
            end

            %Displays an error message if the value selected exceeds the
            %max value defined by the box below
            if value>app.RepeatStepsEditField_max.Value
%                 f = uifigure;
%                 message=sprintf('Value too large \n Start value must be less
than or qual to end value');
%                 uialert(f,message, 'Invalid Action')
                app.RepeatStepsEditField_max.Value =
app.RepeatStepsEditField_min.Value;
            end

        end


        % Value changed function: RepeatStepsEditField_max
        function RepeatStepsEditField_maxValueChanged(app, event)
            value = app.RepeatStepsEditField_max.Value;

            %Displays error message if the value selected exceeds the
            %number of steps saved
            if value>length(app.StepList)
                f = uifigure;
                message=sprintf('Value too large \n Step %d does not
exist',value);
```

```
                uialert(f,message, 'Invalid Action')
                app.RepeatStepsEditField_max.Value = 
app.RepeatStepsEditField_min.Value;
            end
            
            %Displays an error message if the value selected is lower than
            %the min value defined by the box above
            if value<app.RepeatStepsEditField_min.Value
%                 f = uifigure;
%                 message=sprintf('Value too small \n End value must be greater 
than or equal to start value');
%                 uialert(f,message, 'Invalid Action')
                 app.RepeatStepsEditField_min.Value = 
app.RepeatStepsEditField_max.Value;
            end
        end


        % Button pushed function: RepeatSavedStepsButton
        function RepeatSavedStepsButtonPushed(app, event)
            %Define the start and end of the steps to repeat and the number
            %of times they are to be repeated
            StartStep=app.RepeatStepsEditField_min.Value;
            EndStep=app.RepeatStepsEditField_max.Value;
            Repeats=app.RepeatStepsEditField_number.Value;
            
            if EndStep<=length(app.StepList) && StartStep<=EndStep
                %Number of additional steps to repeat (i.e. if only one step is
                %being repeated this will return 0)
                NumStep=EndStep-StartStep;
                
                %Main loop for copying the desired steps on 'Step' and 'StepList'
                for i=1:Repeats
                    n=length(app.StepList)+1;
                    app.Step(n:n+NumStep)=app.Step(StartStep:EndStep);
                    for j=n:n+NumStep
                        app.StepList{j}=['Step ' num2str(j)];
                    end
                end
                app.SavedStepsDropDown.Items = app.StepList;
                app.DeleteStepDropDown.Items = erase(app.StepList, 'Step ');
            else
                f = uifigure;
                message=sprintf('Boundaries for steps to repeat are invalid');
                uialert(f,message, 'Invalid Action')
            end
            
        end
```

```
        % Button pushed function: ClearAllStepsButton
        function ClearAllStepsButtonPushed(app, event)
                %Delete all drop down options
                app.StepList={};

                app.SavedStepsDropDown.Items = app.StepList;
                app.DeleteStepDropDown.Items = erase(app.StepList, 'Step ');


                %Delete all steps
                app.Step=struct;

                %Disable delete and repeat buttons
                app.DeleteStepButton.Enable = 'off';
                app.ClearAllStepsButton.Enable = 'off';
                app.RepeatSavedStepsButton.Enable = 'off';
        end


        % Value changed function: CONNECTButton
        function attemptConnection(app, event)
            if ~strcmpi(app.COMPORTDropDown.Value,'-SELECT PORT-')
                if ~strcmpi(app.COMPORTDropDown.Value,'Bypass')
                    try
                        app.comPort = app.COMPORTDropDown.Value;
                        instrreset; %Clears any previous serial connections

                        %Set up arduino object
                        %Note: "this" is a structure. Code does not execute
                        %properly if arduino object is not stored in a
                        %structure
                        app.this.arduino =
serial(app.comPort,'Baudrate',115000,'Timeout',0.001);
                        app.this.arduino.BytesAvailableFcnCount = 1;
                        app.this.arduino.BytesAvailableFcnMode = 'byte';
                        fopen(app.this.arduino);
                        app.this.arduino.ReadAsyncMode = 'continuous';

                        %Change button appearance
                        app.CONNECTIONDisplay.Value = '*** CONNECTED ***';
                        app.CONNECTIONDisplay.FontColor = [0,1,0];
                        app.CONNECTButton.ValueChangedFcn =
createCallbackFcn(app, @attemptDisconnect, true);
                        app.CONNECTButton.BackgroundColor = [1,0,0];
                        app.CONNECTButton.Text = 'DISCONNECT';

                    catch ME
                        app.errorCatch = ME;
                        app.CONNECTButton.Value = false;
                        for n = 1:3
```

```
                            app.CONNECTIONDisplay.Value = '*** ERROR CONNECTING
***';
                            tic;while(toc<0.75);end
                            app.CONNECTIONDisplay.Value = '*** PLEASE TRY AGAIN
***';
                            tic;while(toc<0.75);end
                            app.CONNECTIONDisplay.Value = '*** NOT CONNECTED
***';
                        end
                    end

                else
                    app.CONNECTButton.ValueChangedFcn = createCallbackFcn(app,
@attemptDisconnect, true);
                    app.CONNECTButton.BackgroundColor = [1,0,0];
                    app.CONNECTButton.Text = 'DISCONNECT';
                end
            else
                app.CONNECTButton.Value = false;
            end

        end


        % Button pushed function: StartBraidingButton
        function StartBraidingButtonPushed(app, event)
            if ~isempty(app.StepList)
                TogglePauseButton(app,1,0); %Enable the pause button, set it to
unpressed state

                %Initialize steps as array. This function will send each
                %step as a single row of the array when asked nicely by
                %arduino. Each row must start with 254 and end with 255 or
                %arduino is confused

                %General format: [254, takeupSpeed, takeupEnable,
takeupDirection,
                %   Motor1Enable, Motor1Direction, ...Motor49 Direction,
                %   255]
                app.TotalSteps = length(app.StepList);
                app.Step4Arduino = ones(app.TotalSteps,103);
                row=1;
                for n=1:app.TotalSteps
                    app.Step4Arduino(row,2:4) = [app.Step(n).TakeUp.Speed,...
                        app.Step(n).TakeUp.Enable, app.Step(n).TakeUp.Direction];

                    incr=5;
                    maxrow=row;
                    for i=1:49
                        %incr stores the column position of the current
```

```
                        %motor
                        app.Step4Arduino(row,incr:incr+1) =
[app.Step(n).motor(i).Enable,...
                            app.Step(n).motor(i).Direction];

                        %Generate extra row(s) if rotation > 90. Arduino is
                        %set up for 90 degree turns so this allows arduino
                        %to read 180 degrees as 2 90 degree steps
                        if (app.Step(n).motor(i).Rotation > 90) &&
(app.Step(n).motor(i).Enable == 0)
                            factor = app.Step(n).motor(i).Rotation/90;

                            %maxrow stores the current max of added rows.
                            %If the current motor needs to add more rows,
                            %it adds 1's so that all other motors are
                            %disabled in the new row
                            if (row+(factor-1))>(maxrow)
                                app.Step4Arduino(maxrow+1:row+(factor-
1),5:102)=1;

                                %Takeup cloumns are copied as they don't
                                %change
                                for j=maxrow:row+(factor-1)
                                    app.Step4Arduino(j,2:4) = [...
                                        app.Step(n).TakeUp.Speed,...
                                        app.Step(n).TakeUp.Enable,
app.Step(n).TakeUp.Direction];
                                end
                                maxrow=row+(factor-1);

                                %Copy the current motor up to the new max
                                %row
                                for j=row:maxrow
                                    app.Step4Arduino(j,incr:incr+1) = [...
                                        app.Step(n).motor(i).Enable,...
                                        app.Step(n).motor(i).Direction];
                                end
                            else
                                %copy the current motor up to the factor
                                %amount (i.e copies 2x for 180, 3x for 270,
                                %etc)
                                for j=row:row+(factor-1)
                                    app.Step4Arduino(j,incr:incr+1) = [...
                                        app.Step(n).motor(i).Enable,...
                                        app.Step(n).motor(i).Direction];
                                end
                            end
                        end
                        incr=incr+2;
                    end
```

```
                row=maxrow+1;
            end
            %Add in the start and end bytes for Arduino
            app.Step4Arduino(:,1)=254;
            app.Step4Arduino(:,103)=255;

            %Re-determine max steps sent to Arduino
            app.TotalSteps = size(app.Step4Arduino,1);


            if strcmpi(app.CONNECTIONDisplay.Value, '*** CONNECTED ***')
                app.StepCounter=0;
                fwrite(app.this.arduino,251,'uchar'); %Starts takeup motors
                braidControl(app); %This function listens to arduino and
sends data if asked nicely
            else
                f = uifigure;
                message=sprintf('Need to connect to arduino before
braiding');
                uialert(f,message, 'Invalid Action')
            end
        else
            f = uifigure;
            message=sprintf('No steps stored \n Please go to "Braid Pattern
Setup" tab');
            uialert(f,message, 'Invalid Action')
        end
    end


    % Value changed function: PauseButton
    function PauseButtonValueChanged(app, event)
        value = app.PauseButton.Value;
        TogglePauseButton(app,1,value); %Toggle button appearance

        %The number 253 pauses/resumes motor turning in arduino. When
        %the pause button is first pressed, function is paused. When
        %pressed again, Matlab resumes listening to arduino per the
        %function braidControl
        if value
            fwrite(app.this.arduino,253,'uchar');
        else
            fwrite(app.this.arduino,252,'uchar');
            braidControl(app)
        end
    end


    % Button pushed function: CreateFullPathButton
    function CreateFullPathButtonPushed(app, event)
```

```
    %This function generates all the steps needed for a full path.
    %The full path is defined in fullPathPattern as the number of
    %steps required to replace all fibers into their original
    %position.
    if ~isempty(app.StepList)

        %Using a similar procedure to the machine emulation
        %function, this function determines the number of steps in
        %sa unit cell
        fullPathSteps=fullPathPattern(app.CamMatrix,app.Step);


        %Thiss loop copies the steps in a pattern until the number
        %of steps for the unit cell is reached
        for i=1:length(app.StepList):fullPathSteps
            for j=1:length(app.StepList)
                app.Step(i)=app.Step(j);
                i=i+1;
            end
        end


        %Thiss loop updates the list of availabel steps and related
        %menus
        for i=1:fullPathSteps
            app.StepList{i}=['Step ' num2str(i)];
        end
        app.SavedStepsDropDown.Items = app.StepList;
        app.DeleteStepDropDown.Items = erase(app.StepList, 'Step ');

    else
        %This throws an error message if there are no steps with
        %which to create a unit cell.
        f = uifigure;
        message=sprintf('Steps are needed to create path');
        uialert(f,message, 'Invalid Action')


    end
end


% Button pushed function: SaveSelectionButton
function SaveSelection(app, event)
    %Initialize matrix of fiber locations
    app.CamMatrix=zeros(15);
    for i=1:225
        name=['Button_' num2str(i)];
        if mod(i,15)~=0
            row=floor(i/15)+1;
```

```
                col=i-(row-1)*15;
            else
                row=floor(i/15);
                col=i-(row-1)*15;
            end
            app.CamMatrix(row,col)=app.(name).Value;
        end



        %Change 1's to integers. Assigned this way to ensure no repetition
        in=find(app.CamMatrix==1);
        app.CamMatrix(in)=1:length(in);
        app.Nyarn=length(in); %Defines total number of yarns

        %Turn axial yarns on/off (used to space yarns properly for no
        %interference
        app.enableYarns=app.AxialYarnsButton.Value;


    end
end


% Component initialization
methods (Access = private)


    % Create UIFigure and components
    function createComponents(app)


        % Create UIFigure and hide until all components are created
        app.UIFigure = uifigure('Visible', 'off');
        app.UIFigure.Position = [101 101 1405 868];
        app.UIFigure.Name = 'Braid Generator';
        app.UIFigure.CloseRequestFcn = createCallbackFcn(app,
@UIFigureCloseRequest, true);


        % Create TabGroup
        app.TabGroup = uitabgroup(app.UIFigure);
        app.TabGroup.Position = [1 0 1404 869];


        % Create CarrierSetup
        app.CarrierSetup = uitab(app.TabGroup);
        app.CarrierSetup.Title = 'Carrier Setup';
```

```
            % Create CamMotorSetupPanel
            app.CamMotorSetupPanel = uipanel(app.CarrierSetup);
            app.CamMotorSetupPanel.Title = 'Cam/Motor Setup';
            app.CamMotorSetupPanel.Position = [18 325 498 502];

            % Create PopulateCarriersButton
            app.PopulateCarriersButton = uibutton(app.CamMotorSetupPanel,
'push');
            app.PopulateCarriersButton.ButtonPushedFcn = createCallbackFcn(app,
@PopulateCarriers, true);
            app.PopulateCarriersButton.Position = [194 8 109 22];
            app.PopulateCarriersButton.Text = 'Populate Carriers';

            % Create cam_1
            app.cam_1 = uibutton(app.CamMotorSetupPanel, 'state');
            app.cam_1.Text = 'Cam 1';
            app.cam_1.FontSize = 11;
            app.cam_1.Position = [55 416 50 50];
            app.cam_1.Value = true;

            % Create cam_2
            app.cam_2 = uibutton(app.CamMotorSetupPanel, 'state');
            app.cam_2.Text = 'Cam 2';
            app.cam_2.FontSize = 11;
            app.cam_2.Position = [111 416 50 50];
            app.cam_2.Value = true;

            % Create cam_3
            app.cam_3 = uibutton(app.CamMotorSetupPanel, 'state');
            app.cam_3.Text = 'Cam 3';
            app.cam_3.FontSize = 11;
            app.cam_3.Position = [169 416 50 50];
            app.cam_3.Value = true;

            % Create cam_4
            app.cam_4 = uibutton(app.CamMotorSetupPanel, 'state');
            app.cam_4.Text = 'Cam 4';
            app.cam_4.FontSize = 11;
            app.cam_4.Position = [224 416 50 50];
            app.cam_4.Value = true;

            % Create cam_5
            app.cam_5 = uibutton(app.CamMotorSetupPanel, 'state');
            app.cam_5.Text = 'Cam 5';
```

```
app.cam_5.FontSize = 11;
app.cam_5.Position = [280 416 50 50];
app.cam_5.Value = true;


% Create cam_6
app.cam_6 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_6.Text = 'Cam 6';
app.cam_6.FontSize = 11;
app.cam_6.Position = [338 416 50 50];
app.cam_6.Value = true;


% Create cam_7
app.cam_7 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_7.Text = 'Cam 7';
app.cam_7.FontSize = 11;
app.cam_7.Position = [397 416 50 50];
app.cam_7.Value = true;


% Create cam_8
app.cam_8 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_8.Text = 'Cam 8';
app.cam_8.FontSize = 11;
app.cam_8.Position = [55 359 50 50];
app.cam_8.Value = true;


% Create cam_9
app.cam_9 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_9.Text = 'Cam 9';
app.cam_9.FontSize = 11;
app.cam_9.Position = [111 359 50 50];
app.cam_9.Value = true;


% Create cam_10
app.cam_10 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_10.Text = 'Cam 10';
app.cam_10.FontSize = 11;
app.cam_10.Position = [170 358 50 50];
app.cam_10.Value = true;


% Create cam_11
app.cam_11 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_11.Text = 'Cam 11';
app.cam_11.FontSize = 11;
app.cam_11.Position = [226 359 47 50];
```

```
app.cam_11.Value = true;


% Create cam_12
app.cam_12 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_12.Text = 'Cam 12';
app.cam_12.FontSize = 11;
app.cam_12.Position = [281 359 48 50];
app.cam_12.Value = true;


% Create cam_13
app.cam_13 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_13.Text = 'Cam 13';
app.cam_13.FontSize = 11;
app.cam_13.Position = [339 359 48 50];
app.cam_13.Value = true;


% Create cam_14
app.cam_14 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_14.Text = 'Cam 14';
app.cam_14.FontSize = 11;
app.cam_14.Position = [398 359 49 50];
app.cam_14.Value = true;


% Create cam_15
app.cam_15 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_15.Text = 'Cam 15';
app.cam_15.FontSize = 11;
app.cam_15.Position = [57 301 49 50];
app.cam_15.Value = true;


% Create cam_16
app.cam_16 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_16.Text = 'Cam 16';
app.cam_16.FontSize = 11;
app.cam_16.Position = [113 301 49 50];
app.cam_16.Value = true;


% Create cam_17
app.cam_17 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_17.Text = 'Cam 17';
app.cam_17.FontSize = 11;
app.cam_17.Position = [171 301 49 50];
app.cam_17.Value = true;
```

```
% Create cam_18
app.cam_18 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_18.Text = 'Cam 18';
app.cam_18.FontSize = 11;
app.cam_18.Position = [226 301 49 50];
app.cam_18.Value = true;


% Create cam_19
app.cam_19 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_19.Text = 'Cam 19';
app.cam_19.FontSize = 11;
app.cam_19.Position = [282 301 49 50];
app.cam_19.Value = true;


% Create cam_20
app.cam_20 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_20.Text = 'Cam 20';
app.cam_20.FontSize = 11;
app.cam_20.Position = [340 301 49 50];
app.cam_20.Value = true;


% Create cam_21
app.cam_21 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_21.Text = 'Cam 21';
app.cam_21.FontSize = 11;
app.cam_21.Position = [398 301 49 50];
app.cam_21.Value = true;


% Create cam_22
app.cam_22 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_22.Text = 'Cam 22';
app.cam_22.FontSize = 11;
app.cam_22.Position = [57 244 49 50];
app.cam_22.Value = true;


% Create cam_23
app.cam_23 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_23.Text = 'Cam 23';
app.cam_23.FontSize = 11;
app.cam_23.Position = [113 244 49 50];
app.cam_23.Value = true;


% Create cam_24
app.cam_24 = uibutton(app.CamMotorSetupPanel, 'state');
```

```
app.cam_24.Text = 'Cam 24';
app.cam_24.FontSize = 11;
app.cam_24.Position = [171 244 49 50];
app.cam_24.Value = true;

% Create cam_25
app.cam_25 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_25.Text = 'Cam 25';
app.cam_25.FontSize = 11;
app.cam_25.Position = [226 244 49 50];
app.cam_25.Value = true;

% Create cam_26
app.cam_26 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_26.Text = 'Cam 26';
app.cam_26.FontSize = 11;
app.cam_26.Position = [282 244 49 50];
app.cam_26.Value = true;

% Create cam_27
app.cam_27 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_27.Text = 'Cam 27';
app.cam_27.FontSize = 11;
app.cam_27.Position = [340 244 49 50];
app.cam_27.Value = true;

% Create cam_28
app.cam_28 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_28.Text = 'Cam 28';
app.cam_28.FontSize = 11;
app.cam_28.Position = [398 244 49 50];
app.cam_28.Value = true;

% Create cam_29
app.cam_29 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_29.Text = 'Cam 29';
app.cam_29.FontSize = 11;
app.cam_29.Position = [57 186 49 50];
app.cam_29.Value = true;

% Create cam_30
app.cam_30 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_30.Text = 'Cam 30';
app.cam_30.FontSize = 11;
```

```
app.cam_30.Position = [113 186 49 50];
app.cam_30.Value = true;

% Create cam_31
app.cam_31 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_31.Text = 'Cam 31';
app.cam_31.FontSize = 11;
app.cam_31.Position = [171 186 49 50];
app.cam_31.Value = true;

% Create cam_32
app.cam_32 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_32.Text = 'Cam 32';
app.cam_32.FontSize = 11;
app.cam_32.Position = [226 186 49 50];
app.cam_32.Value = true;

% Create cam_33
app.cam_33 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_33.Text = 'Cam 33';
app.cam_33.FontSize = 11;
app.cam_33.Position = [282 186 49 50];
app.cam_33.Value = true;

% Create cam_34
app.cam_34 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_34.Text = 'Cam 34';
app.cam_34.FontSize = 11;
app.cam_34.Position = [340 186 49 50];
app.cam_34.Value = true;

% Create cam_35
app.cam_35 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_35.Text = 'Cam 35';
app.cam_35.FontSize = 11;
app.cam_35.Position = [398 186 49 50];
app.cam_35.Value = true;

% Create cam_36
app.cam_36 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_36.Text = 'Cam 36';
app.cam_36.FontSize = 11;
app.cam_36.Position = [57 126 49 50];
app.cam_36.Value = true;
```

```
% Create cam_37
app.cam_37 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_37.Text = 'Cam 37';
app.cam_37.FontSize = 11;
app.cam_37.Position = [113 126 49 50];
app.cam_37.Value = true;


% Create cam_38
app.cam_38 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_38.Text = 'Cam 38';
app.cam_38.FontSize = 11;
app.cam_38.Position = [171 126 49 50];
app.cam_38.Value = true;


% Create cam_39
app.cam_39 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_39.Text = 'Cam 39';
app.cam_39.FontSize = 11;
app.cam_39.Position = [226 126 49 50];
app.cam_39.Value = true;


% Create cam_40
app.cam_40 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_40.Text = 'Cam 40';
app.cam_40.FontSize = 11;
app.cam_40.Position = [282 126 49 50];
app.cam_40.Value = true;


% Create cam_41
app.cam_41 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_41.Text = 'Cam 41';
app.cam_41.FontSize = 11;
app.cam_41.Position = [340 126 49 50];
app.cam_41.Value = true;


% Create cam_42
app.cam_42 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_42.Text = 'Cam 42';
app.cam_42.FontSize = 11;
app.cam_42.Position = [398 126 49 50];
app.cam_42.Value = true;
```

```
% Create cam_43
app.cam_43 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_43.Text = 'Cam 43';
app.cam_43.FontSize = 11;
app.cam_43.Position = [57 70 49 50];
app.cam_43.Value = true;


% Create cam_44
app.cam_44 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_44.Text = 'Cam 44';
app.cam_44.FontSize = 11;
app.cam_44.Position = [113 70 49 50];
app.cam_44.Value = true;


% Create cam_45
app.cam_45 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_45.Text = 'Cam 45';
app.cam_45.FontSize = 11;
app.cam_45.Position = [171 70 49 50];
app.cam_45.Value = true;


% Create cam_46
app.cam_46 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_46.Text = 'Cam 46';
app.cam_46.FontSize = 11;
app.cam_46.Position = [226 70 49 50];
app.cam_46.Value = true;


% Create cam_47
app.cam_47 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_47.Text = 'Cam 47';
app.cam_47.FontSize = 11;
app.cam_47.Position = [282 70 49 50];
app.cam_47.Value = true;


% Create cam_48
app.cam_48 = uibutton(app.CamMotorSetupPanel, 'state');
app.cam_48.Text = 'Cam 48';
app.cam_48.FontSize = 11;
app.cam_48.Position = [340 70 49 50];
app.cam_48.Value = true;


% Create cam_49
app.cam_49 = uibutton(app.CamMotorSetupPanel, 'state');
```

```
            app.cam_49.Text = 'Cam 49';
            app.cam_49.FontSize = 11;
            app.cam_49.Position = [398 70 49 50];
            app.cam_49.Value = true;

            % Create ActivateAllButton
            app.ActivateAllButton = uibutton(app.CamMotorSetupPanel, 'push');
            app.ActivateAllButton.ButtonPushedFcn = createCallbackFcn(app,
@ActivateAllButtonPushed, true);
            app.ActivateAllButton.Position = [150 39 100 22];
            app.ActivateAllButton.Text = 'Activate All';

            % Create DeactivateAllButton
            app.DeactivateAllButton = uibutton(app.CamMotorSetupPanel, 'push');
            app.DeactivateAllButton.ButtonPushedFcn = createCallbackFcn(app,
@DeactivateAllButtonPushed, true);
            app.DeactivateAllButton.Position = [257 39 100 22];
            app.DeactivateAllButton.Text = 'Deactivate All';

            % Create AxialYarnsButton
            app.AxialYarnsButton = uibutton(app.CamMotorSetupPanel, 'state');
            app.AxialYarnsButton.ValueChangedFcn = createCallbackFcn(app,
@PopulateCarriers, true);
            app.AxialYarnsButton.Text = 'Axial Yarns';
            app.AxialYarnsButton.Position = [363 39 100 22];

            % Create SaveSelectionButton
            app.SaveSelectionButton = uibutton(app.CamMotorSetupPanel, 'push');
            app.SaveSelectionButton.ButtonPushedFcn = createCallbackFcn(app,
@SaveSelection, true);
            app.SaveSelectionButton.Position = [42 39 100 22];
            app.SaveSelectionButton.Text = 'Save Selection';

            % Create CarrierSetupPanel
            app.CarrierSetupPanel = uipanel(app.CarrierSetup);
            app.CarrierSetupPanel.TitlePosition = 'centertop';
            app.CarrierSetupPanel.Title = 'Carrier Setup';
            app.CarrierSetupPanel.Position = [537 325 456 502];

            % Create Button_1
            app.Button_1 = uibutton(app.CarrierSetupPanel, 'state');
            app.Button_1.Text = '';
            app.Button_1.Position = [59 417 15 15];
```

```
% Create Button_2
app.Button_2 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_2.Text = '';
app.Button_2.Position = [82 417 15 15];

% Create Button_3
app.Button_3 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_3.Text = '';
app.Button_3.Position = [106 417 15 15];

% Create Button_4
app.Button_4 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_4.Text = '';
app.Button_4.Position = [128 417 15 15];

% Create Button_5
app.Button_5 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_5.Text = '';
app.Button_5.Position = [151 417 15 15];

% Create Button_6
app.Button_6 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_6.Text = '';
app.Button_6.Position = [175 417 15 15];

% Create Button_7
app.Button_7 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_7.Text = '';
app.Button_7.Position = [198 417 15 15];

% Create Button_8
app.Button_8 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_8.Text = '';
app.Button_8.Position = [221 417 15 15];

% Create Button_9
app.Button_9 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_9.Text = '';
app.Button_9.Position = [245 417 15 15];

% Create Button_10
app.Button_10 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_10.Text = '';
app.Button_10.Position = [267 417 15 15];

% Create Button_11
app.Button_11 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_11.Text = '';
app.Button_11.Position = [290 417 15 15];

% Create Button_12
app.Button_12 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_12.Text = '';
app.Button_12.Position = [314 417 15 15];

% Create Button_13
app.Button_13 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_13.Text = '';
app.Button_13.Position = [336 417 15 15];

% Create Button_14
app.Button_14 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_14.Text = '';
app.Button_14.Position = [359 417 15 15];

% Create Button_15
app.Button_15 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_15.Text = '';
app.Button_15.Position = [383 417 15 15];

% Create Button_16
app.Button_16 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_16.Text = '';
app.Button_16.Position = [59 394 15 15];

% Create Button_17
app.Button_17 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_17.Text = '';
app.Button_17.Position = [82 394 15 15];

% Create Button_18
app.Button_18 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_18.Text = '';
app.Button_18.Position = [106 394 15 15];
```

```
% Create Button_19
app.Button_19 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_19.Text = '';
app.Button_19.Position = [128 394 15 15];

% Create Button_20
app.Button_20 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_20.Text = '';
app.Button_20.Position = [151 394 15 15];

% Create Button_21
app.Button_21 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_21.Text = '';
app.Button_21.Position = [175 394 15 15];

% Create Button_22
app.Button_22 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_22.Text = '';
app.Button_22.Position = [198 394 15 15];

% Create Button_23
app.Button_23 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_23.Text = '';
app.Button_23.Position = [221 394 15 15];

% Create Button_24
app.Button_24 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_24.Text = '';
app.Button_24.Position = [245 394 15 15];

% Create Button_25
app.Button_25 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_25.Text = '';
app.Button_25.Position = [267 394 15 15];

% Create Button_26
app.Button_26 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_26.Text = '';
app.Button_26.Position = [290 394 15 15];
```

```
% Create Button_27
app.Button_27 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_27.Text = '';
app.Button_27.Position = [314 394 15 15];

% Create Button_28
app.Button_28 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_28.Text = '';
app.Button_28.Position = [336 394 15 15];

% Create Button_29
app.Button_29 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_29.Text = '';
app.Button_29.Position = [359 394 15 15];

% Create Button_30
app.Button_30 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_30.Text = '';
app.Button_30.Position = [383 394 15 15];

% Create Button_31
app.Button_31 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_31.Text = '';
app.Button_31.Position = [59 372 15 15];

% Create Button_32
app.Button_32 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_32.Text = '';
app.Button_32.Position = [82 372 15 15];

% Create Button_33
app.Button_33 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_33.Text = '';
app.Button_33.Position = [106 372 15 15];

% Create Button_34
app.Button_34 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_34.Text = '';
app.Button_34.Position = [128 372 15 15];

% Create Button_35
app.Button_35 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_35.Text = '';
app.Button_35.Position = [151 372 15 15];

% Create Button_36
app.Button_36 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_36.Text = '';
app.Button_36.Position = [175 372 15 15];

% Create Button_37
app.Button_37 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_37.Text = '';
app.Button_37.Position = [198 372 15 15];

% Create Button_38
app.Button_38 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_38.Text = '';
app.Button_38.Position = [221 372 15 15];

% Create Button_39
app.Button_39 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_39.Text = '';
app.Button_39.Position = [245 372 15 15];

% Create Button_40
app.Button_40 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_40.Text = '';
app.Button_40.Position = [267 372 15 15];

% Create Button_41
app.Button_41 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_41.Text = '';
app.Button_41.Position = [290 372 15 15];

% Create Button_42
app.Button_42 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_42.Text = '';
app.Button_42.Position = [314 372 15 15];

% Create Button_43
app.Button_43 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_43.Text = '';
app.Button_43.Position = [336 372 15 15];
```

```
% Create Button_44
app.Button_44 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_44.Text = '';
app.Button_44.Position = [359 372 15 15];

% Create Button_45
app.Button_45 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_45.Text = '';
app.Button_45.Position = [383 372 15 15];

% Create Button_46
app.Button_46 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_46.Text = '';
app.Button_46.Position = [59 349 15 15];

% Create Button_47
app.Button_47 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_47.Text = '';
app.Button_47.Position = [82 349 15 15];

% Create Button_48
app.Button_48 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_48.Text = '';
app.Button_48.Position = [106 349 15 15];

% Create Button_49
app.Button_49 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_49.Text = '';
app.Button_49.Position = [128 349 15 15];

% Create Button_50
app.Button_50 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_50.Text = '';
app.Button_50.Position = [151 349 15 15];

% Create Button_51
app.Button_51 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_51.Text = '';
app.Button_51.Position = [175 349 15 15];
```

```
% Create Button_52
app.Button_52 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_52.Text = '';
app.Button_52.Position = [198 349 15 15];

% Create Button_53
app.Button_53 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_53.Text = '';
app.Button_53.Position = [221 349 15 15];

% Create Button_54
app.Button_54 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_54.Text = '';
app.Button_54.Position = [245 349 15 15];

% Create Button_55
app.Button_55 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_55.Text = '';
app.Button_55.Position = [267 349 15 15];

% Create Button_56
app.Button_56 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_56.Text = '';
app.Button_56.Position = [290 349 15 15];

% Create Button_57
app.Button_57 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_57.Text = '';
app.Button_57.Position = [314 349 15 15];

% Create Button_58
app.Button_58 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_58.Text = '';
app.Button_58.Position = [336 349 15 15];

% Create Button_59
app.Button_59 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_59.Text = '';
app.Button_59.Position = [359 349 15 15];

% Create Button_60
app.Button_60 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_60.Text = '';
app.Button_60.Position = [383 349 15 15];

% Create Button_61
app.Button_61 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_61.Text = '';
app.Button_61.Position = [59 326 15 15];

% Create Button_62
app.Button_62 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_62.Text = '';
app.Button_62.Position = [82 326 15 15];

% Create Button_63
app.Button_63 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_63.Text = '';
app.Button_63.Position = [106 326 15 15];

% Create Button_64
app.Button_64 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_64.Text = '';
app.Button_64.Position = [128 326 15 15];

% Create Button_65
app.Button_65 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_65.Text = '';
app.Button_65.Position = [151 326 15 15];

% Create Button_66
app.Button_66 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_66.Text = '';
app.Button_66.Position = [175 326 15 15];

% Create Button_67
app.Button_67 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_67.Text = '';
app.Button_67.Position = [198 326 15 15];

% Create Button_68
app.Button_68 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_68.Text = '';
app.Button_68.Position = [221 326 15 15];
```

```
% Create Button_69
app.Button_69 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_69.Text = '';
app.Button_69.Position = [245 326 15 15];

% Create Button_70
app.Button_70 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_70.Text = '';
app.Button_70.Position = [267 326 15 15];

% Create Button_71
app.Button_71 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_71.Text = '';
app.Button_71.Position = [290 326 15 15];

% Create Button_72
app.Button_72 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_72.Text = '';
app.Button_72.Position = [314 326 15 15];

% Create Button_73
app.Button_73 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_73.Text = '';
app.Button_73.Position = [336 326 15 15];

% Create Button_74
app.Button_74 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_74.Text = '';
app.Button_74.Position = [359 326 15 15];

% Create Button_75
app.Button_75 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_75.Text = '';
app.Button_75.Position = [383 326 15 15];

% Create Button_76
app.Button_76 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_76.Text = '';
app.Button_76.Position = [59 304 15 15];
```

```
% Create Button_77
app.Button_77 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_77.Text = '';
app.Button_77.Position = [82 304 15 15];

% Create Button_78
app.Button_78 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_78.Text = '';
app.Button_78.Position = [106 304 15 15];

% Create Button_79
app.Button_79 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_79.Text = '';
app.Button_79.Position = [128 304 15 15];

% Create Button_80
app.Button_80 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_80.Text = '';
app.Button_80.Position = [151 304 15 15];

% Create Button_81
app.Button_81 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_81.Text = '';
app.Button_81.Position = [175 304 15 15];

% Create Button_82
app.Button_82 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_82.Text = '';
app.Button_82.Position = [198 304 15 15];

% Create Button_83
app.Button_83 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_83.Text = '';
app.Button_83.Position = [221 304 15 15];

% Create Button_84
app.Button_84 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_84.Text = '';
app.Button_84.Position = [245 304 15 15];

% Create Button_85
app.Button_85 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_85.Text = '';
app.Button_85.Position = [267 304 15 15];

% Create Button_86
app.Button_86 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_86.Text = '';
app.Button_86.Position = [290 304 15 15];

% Create Button_87
app.Button_87 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_87.Text = '';
app.Button_87.Position = [314 304 15 15];

% Create Button_88
app.Button_88 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_88.Text = '';
app.Button_88.Position = [336 304 15 15];

% Create Button_89
app.Button_89 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_89.Text = '';
app.Button_89.Position = [359 304 15 15];

% Create Button_90
app.Button_90 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_90.Text = '';
app.Button_90.Position = [383 304 15 15];

% Create Button_91
app.Button_91 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_91.Text = '';
app.Button_91.Position = [59 281 15 15];

% Create Button_92
app.Button_92 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_92.Text = '';
app.Button_92.Position = [82 281 15 15];

% Create Button_93
app.Button_93 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_93.Text = '';
app.Button_93.Position = [106 281 15 15];
```

```
% Create Button_94
app.Button_94 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_94.Text = '';
app.Button_94.Position = [128 281 15 15];


% Create Button_95
app.Button_95 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_95.Text = '';
app.Button_95.Position = [151 281 15 15];


% Create Button_96
app.Button_96 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_96.Text = '';
app.Button_96.Position = [175 281 15 15];


% Create Button_97
app.Button_97 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_97.Text = '';
app.Button_97.Position = [198 281 15 15];


% Create Button_98
app.Button_98 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_98.Text = '';
app.Button_98.Position = [221 281 15 15];


% Create Button_99
app.Button_99 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_99.Text = '';
app.Button_99.Position = [245 281 15 15];


% Create Button_100
app.Button_100 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_100.Text = '';
app.Button_100.Position = [267 281 15 15];


% Create Button_101
app.Button_101 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_101.Text = '';
app.Button_101.Position = [290 281 15 15];
```

```
% Create Button_102
app.Button_102 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_102.Text = '';
app.Button_102.Position = [314 281 15 15];

% Create Button_103
app.Button_103 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_103.Text = '';
app.Button_103.Position = [336 281 15 15];

% Create Button_104
app.Button_104 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_104.Text = '';
app.Button_104.Position = [359 281 15 15];

% Create Button_105
app.Button_105 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_105.Text = '';
app.Button_105.Position = [383 281 15 15];

% Create Button_106
app.Button_106 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_106.Text = '';
app.Button_106.Position = [59 258 15 15];

% Create Button_107
app.Button_107 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_107.Text = '';
app.Button_107.Position = [82 258 15 15];

% Create Button_108
app.Button_108 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_108.Text = '';
app.Button_108.Position = [106 258 15 15];

% Create Button_109
app.Button_109 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_109.Text = '';
app.Button_109.Position = [128 258 15 15];

% Create Button_110
app.Button_110 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_110.Text = '';
app.Button_110.Position = [151 258 15 15];

% Create Button_111
app.Button_111 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_111.Text = '';
app.Button_111.Position = [175 258 15 15];

% Create Button_112
app.Button_112 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_112.Text = '';
app.Button_112.Position = [198 258 15 15];

% Create Button_113
app.Button_113 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_113.Text = '';
app.Button_113.Position = [221 258 15 15];

% Create Button_114
app.Button_114 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_114.Text = '';
app.Button_114.Position = [245 258 15 15];

% Create Button_115
app.Button_115 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_115.Text = '';
app.Button_115.Position = [267 258 15 15];

% Create Button_116
app.Button_116 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_116.Text = '';
app.Button_116.Position = [290 258 15 15];

% Create Button_117
app.Button_117 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_117.Text = '';
app.Button_117.Position = [314 258 15 15];

% Create Button_118
app.Button_118 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_118.Text = '';
app.Button_118.Position = [336 258 15 15];
```

```
% Create Button_119
app.Button_119 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_119.Text = '';
app.Button_119.Position = [359 258 15 15];

% Create Button_120
app.Button_120 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_120.Text = '';
app.Button_120.Position = [383 258 15 15];

% Create Button_121
app.Button_121 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_121.Text = '';
app.Button_121.Position = [59 236 15 15];

% Create Button_122
app.Button_122 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_122.Text = '';
app.Button_122.Position = [82 236 15 15];

% Create Button_123
app.Button_123 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_123.Text = '';
app.Button_123.Position = [106 236 15 15];

% Create Button_124
app.Button_124 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_124.Text = '';
app.Button_124.Position = [128 236 15 15];

% Create Button_125
app.Button_125 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_125.Text = '';
app.Button_125.Position = [151 236 15 15];

% Create Button_126
app.Button_126 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_126.Text = '';
app.Button_126.Position = [175 236 15 15];
```

```
% Create Button_127
app.Button_127 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_127.Text = '';
app.Button_127.Position = [198 236 15 15];

% Create Button_128
app.Button_128 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_128.Text = '';
app.Button_128.Position = [221 236 15 15];

% Create Button_129
app.Button_129 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_129.Text = '';
app.Button_129.Position = [245 236 15 15];

% Create Button_130
app.Button_130 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_130.Text = '';
app.Button_130.Position = [267 236 15 15];

% Create Button_131
app.Button_131 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_131.Text = '';
app.Button_131.Position = [290 236 15 15];

% Create Button_132
app.Button_132 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_132.Text = '';
app.Button_132.Position = [314 236 15 15];

% Create Button_133
app.Button_133 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_133.Text = '';
app.Button_133.Position = [336 236 15 15];

% Create Button_134
app.Button_134 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_134.Text = '';
app.Button_134.Position = [359 236 15 15];

% Create Button_135
app.Button_135 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_135.Text = '';
app.Button_135.Position = [383 236 15 15];

% Create Button_136
app.Button_136 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_136.Text = '';
app.Button_136.Position = [59 214 15 15];

% Create Button_137
app.Button_137 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_137.Text = '';
app.Button_137.Position = [82 214 15 15];

% Create Button_138
app.Button_138 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_138.Text = '';
app.Button_138.Position = [106 214 15 15];

% Create Button_139
app.Button_139 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_139.Text = '';
app.Button_139.Position = [128 214 15 15];

% Create Button_140
app.Button_140 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_140.Text = '';
app.Button_140.Position = [151 214 15 15];

% Create Button_141
app.Button_141 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_141.Text = '';
app.Button_141.Position = [175 214 15 15];

% Create Button_142
app.Button_142 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_142.Text = '';
app.Button_142.Position = [198 214 15 15];

% Create Button_143
app.Button_143 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_143.Text = '';
app.Button_143.Position = [221 214 15 15];
```

```
% Create Button_144
app.Button_144 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_144.Text = '';
app.Button_144.Position = [245 214 15 15];

% Create Button_145
app.Button_145 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_145.Text = '';
app.Button_145.Position = [267 214 15 15];

% Create Button_146
app.Button_146 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_146.Text = '';
app.Button_146.Position = [290 214 15 15];

% Create Button_147
app.Button_147 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_147.Text = '';
app.Button_147.Position = [314 214 15 15];

% Create Button_148
app.Button_148 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_148.Text = '';
app.Button_148.Position = [336 214 15 15];

% Create Button_149
app.Button_149 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_149.Text = '';
app.Button_149.Position = [359 214 15 15];

% Create Button_150
app.Button_150 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_150.Text = '';
app.Button_150.Position = [383 214 15 15];

% Create Button_151
app.Button_151 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_151.Text = '';
app.Button_151.Position = [59 191 15 15];
```

```
% Create Button_152
app.Button_152 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_152.Text = '';
app.Button_152.Position = [82 191 15 15];

% Create Button_153
app.Button_153 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_153.Text = '';
app.Button_153.Position = [106 191 15 15];

% Create Button_154
app.Button_154 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_154.Text = '';
app.Button_154.Position = [128 191 15 15];

% Create Button_155
app.Button_155 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_155.Text = '';
app.Button_155.Position = [151 191 15 15];

% Create Button_156
app.Button_156 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_156.Text = '';
app.Button_156.Position = [175 191 15 15];

% Create Button_157
app.Button_157 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_157.Text = '';
app.Button_157.Position = [198 191 15 15];

% Create Button_158
app.Button_158 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_158.Text = '';
app.Button_158.Position = [221 191 15 15];

% Create Button_159
app.Button_159 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_159.Text = '';
app.Button_159.Position = [245 191 15 15];

% Create Button_160
app.Button_160 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_160.Text = '';
app.Button_160.Position = [267 191 15 15];


% Create Button_161
app.Button_161 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_161.Text = '';
app.Button_161.Position = [290 191 15 15];


% Create Button_162
app.Button_162 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_162.Text = '';
app.Button_162.Position = [314 191 15 15];


% Create Button_163
app.Button_163 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_163.Text = '';
app.Button_163.Position = [336 191 15 15];


% Create Button_164
app.Button_164 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_164.Text = '';
app.Button_164.Position = [359 191 15 15];


% Create Button_165
app.Button_165 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_165.Text = '';
app.Button_165.Position = [383 191 15 15];


% Create Button_166
app.Button_166 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_166.Text = '';
app.Button_166.Position = [59 169 15 15];


% Create Button_167
app.Button_167 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_167.Text = '';
app.Button_167.Position = [82 169 15 15];


% Create Button_168
app.Button_168 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_168.Text = '';
app.Button_168.Position = [106 169 15 15];
```

```
% Create Button_169
app.Button_169 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_169.Text = '';
app.Button_169.Position = [128 169 15 15];

% Create Button_170
app.Button_170 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_170.Text = '';
app.Button_170.Position = [151 169 15 15];

% Create Button_171
app.Button_171 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_171.Text = '';
app.Button_171.Position = [175 169 15 15];

% Create Button_172
app.Button_172 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_172.Text = '';
app.Button_172.Position = [198 169 15 15];

% Create Button_173
app.Button_173 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_173.Text = '';
app.Button_173.Position = [221 169 15 15];

% Create Button_174
app.Button_174 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_174.Text = '';
app.Button_174.Position = [245 169 15 15];

% Create Button_175
app.Button_175 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_175.Text = '';
app.Button_175.Position = [267 169 15 15];

% Create Button_176
app.Button_176 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_176.Text = '';
app.Button_176.Position = [290 169 15 15];
```

```
% Create Button_177
app.Button_177 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_177.Text = '';
app.Button_177.Position = [314 169 15 15];

% Create Button_178
app.Button_178 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_178.Text = '';
app.Button_178.Position = [336 169 15 15];

% Create Button_179
app.Button_179 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_179.Text = '';
app.Button_179.Position = [359 169 15 15];

% Create Button_180
app.Button_180 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_180.Text = '';
app.Button_180.Position = [383 169 15 15];

% Create Button_181
app.Button_181 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_181.Text = '';
app.Button_181.Position = [59 147 15 15];

% Create Button_182
app.Button_182 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_182.Text = '';
app.Button_182.Position = [82 147 15 15];

% Create Button_183
app.Button_183 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_183.Text = '';
app.Button_183.Position = [106 147 15 15];

% Create Button_184
app.Button_184 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_184.Text = '';
app.Button_184.Position = [128 147 15 15];

% Create Button_185
app.Button_185 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_185.Text = '';
app.Button_185.Position = [151 147 15 15];

% Create Button_186
app.Button_186 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_186.Text = '';
app.Button_186.Position = [175 147 15 15];

% Create Button_187
app.Button_187 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_187.Text = '';
app.Button_187.Position = [198 147 15 15];

% Create Button_188
app.Button_188 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_188.Text = '';
app.Button_188.Position = [221 147 15 15];

% Create Button_189
app.Button_189 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_189.Text = '';
app.Button_189.Position = [245 147 15 15];

% Create Button_190
app.Button_190 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_190.Text = '';
app.Button_190.Position = [267 147 15 15];

% Create Button_191
app.Button_191 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_191.Text = '';
app.Button_191.Position = [290 147 15 15];

% Create Button_192
app.Button_192 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_192.Text = '';
app.Button_192.Position = [314 147 15 15];

% Create Button_193
app.Button_193 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_193.Text = '';
app.Button_193.Position = [336 147 15 15];
```

```
% Create Button_194
app.Button_194 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_194.Text = '';
app.Button_194.Position = [359 147 15 15];

% Create Button_195
app.Button_195 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_195.Text = '';
app.Button_195.Position = [383 147 15 15];

% Create Button_196
app.Button_196 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_196.Text = '';
app.Button_196.Position = [59 124 15 15];

% Create Button_197
app.Button_197 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_197.Text = '';
app.Button_197.Position = [82 124 15 15];

% Create Button_198
app.Button_198 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_198.Text = '';
app.Button_198.Position = [106 124 15 15];

% Create Button_199
app.Button_199 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_199.Text = '';
app.Button_199.Position = [128 124 15 15];

% Create Button_200
app.Button_200 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_200.Text = '';
app.Button_200.Position = [151 124 15 15];

% Create Button_201
app.Button_201 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_201.Text = '';
app.Button_201.Position = [175 124 15 15];
```

```
% Create Button_202
app.Button_202 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_202.Text = '';
app.Button_202.Position = [198 124 15 15];

% Create Button_203
app.Button_203 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_203.Text = '';
app.Button_203.Position = [221 124 15 15];

% Create Button_204
app.Button_204 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_204.Text = '';
app.Button_204.Position = [245 124 15 15];

% Create Button_205
app.Button_205 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_205.Text = '';
app.Button_205.Position = [267 124 15 15];

% Create Button_206
app.Button_206 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_206.Text = '';
app.Button_206.Position = [290 124 15 15];

% Create Button_207
app.Button_207 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_207.Text = '';
app.Button_207.Position = [314 124 15 15];

% Create Button_208
app.Button_208 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_208.Text = '';
app.Button_208.Position = [336 124 15 15];

% Create Button_209
app.Button_209 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_209.Text = '';
app.Button_209.Position = [359 124 15 15];

% Create Button_210
app.Button_210 = uibutton(app.CarrierSetupPanel, 'state');
```

```
app.Button_210.Text = '';
app.Button_210.Position = [383 124 15 15];

% Create Button_211
app.Button_211 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_211.Text = '';
app.Button_211.Position = [59 102 15 15];

% Create Button_212
app.Button_212 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_212.Text = '';
app.Button_212.Position = [82 102 15 15];

% Create Button_213
app.Button_213 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_213.Text = '';
app.Button_213.Position = [106 102 15 15];

% Create Button_214
app.Button_214 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_214.Text = '';
app.Button_214.Position = [128 102 15 15];

% Create Button_215
app.Button_215 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_215.Text = '';
app.Button_215.Position = [151 102 15 15];

% Create Button_216
app.Button_216 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_216.Text = '';
app.Button_216.Position = [175 102 15 15];

% Create Button_217
app.Button_217 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_217.Text = '';
app.Button_217.Position = [198 102 15 15];

% Create Button_218
app.Button_218 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_218.Text = '';
app.Button_218.Position = [221 102 15 15];
```

```
% Create Button_219
app.Button_219 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_219.Text = '';
app.Button_219.Position = [245 102 15 15];

% Create Button_220
app.Button_220 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_220.Text = '';
app.Button_220.Position = [267 102 15 15];

% Create Button_221
app.Button_221 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_221.Text = '';
app.Button_221.Position = [290 102 15 15];

% Create Button_222
app.Button_222 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_222.Text = '';
app.Button_222.Position = [314 102 15 15];

% Create Button_223
app.Button_223 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_223.Text = '';
app.Button_223.Position = [336 102 15 15];

% Create Button_224
app.Button_224 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_224.Text = '';
app.Button_224.Position = [359 102 15 15];

% Create Button_225
app.Button_225 = uibutton(app.CarrierSetupPanel, 'state');
app.Button_225.Text = '';
app.Button_225.Position = [383 102 15 15];

% Create BraidPatternSetup
app.BraidPatternSetup = uitab(app.TabGroup);
app.BraidPatternSetup.Title = 'Braid Pattern Setup';

% Create Motor1Panel
```

```
            app.Motor1Panel = uipanel(app.BraidPatternSetup);
            app.Motor1Panel.TitlePosition = 'centertop';
            app.Motor1Panel.Title = 'Motor 1';
            app.Motor1Panel.Position = [19 731 100 102];

            % Create MotorStateButton_1
            app.MotorStateButton_1 = uibutton(app.Motor1Panel, 'state');
            app.MotorStateButton_1.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_1.Text = 'Disabled';
            app.MotorStateButton_1.BackgroundColor = [1 0 0];
            app.MotorStateButton_1.Position = [19 58 62 22];
            app.MotorStateButton_1.Value = true;

            % Create MotorDirectionButton_1
            app.MotorDirectionButton_1 = uibutton(app.Motor1Panel, 'state');
            app.MotorDirectionButton_1.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_1.Enable = 'off';
            app.MotorDirectionButton_1.Text = 'CCW';
            app.MotorDirectionButton_1.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_1.Position = [19 37 62 22];

            % Create MotorRotationSpinner_1
            app.MotorRotationSpinner_1 = uispinner(app.Motor1Panel);
            app.MotorRotationSpinner_1.Step = 90;
            app.MotorRotationSpinner_1.Limits = [90 360];
            app.MotorRotationSpinner_1.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_1.Enable = 'off';
            app.MotorRotationSpinner_1.Position = [39 8 56 22];
            app.MotorRotationSpinner_1.Value = 90;

            % Create AngleSpinnerLabel
            app.AngleSpinnerLabel = uilabel(app.Motor1Panel);
            app.AngleSpinnerLabel.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel.Position = [4 8 36 22];
            app.AngleSpinnerLabel.Text = 'Angle';

            % Create Motor2Panel
            app.Motor2Panel = uipanel(app.BraidPatternSetup);
            app.Motor2Panel.TitlePosition = 'centertop';
            app.Motor2Panel.Title = 'Motor 2';
            app.Motor2Panel.Position = [127 731 100 102];
```

```
            % Create MotorStateButton_2
            app.MotorStateButton_2 = uibutton(app.Motor2Panel, 'state');
            app.MotorStateButton_2.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_2.Text = 'Disabled';
            app.MotorStateButton_2.BackgroundColor = [1 0 0];
            app.MotorStateButton_2.Position = [19 58 62 22];
            app.MotorStateButton_2.Value = true;


            % Create MotorDirectionButton_2
            app.MotorDirectionButton_2 = uibutton(app.Motor2Panel, 'state');
            app.MotorDirectionButton_2.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_2.Enable = 'off';
            app.MotorDirectionButton_2.Text = 'CCW';
            app.MotorDirectionButton_2.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_2.Position = [19 37 62 22];


            % Create MotorRotationSpinner_2
            app.MotorRotationSpinner_2 = uispinner(app.Motor2Panel);
            app.MotorRotationSpinner_2.Step = 90;
            app.MotorRotationSpinner_2.Limits = [90 360];
            app.MotorRotationSpinner_2.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_2.Enable = 'off';
            app.MotorRotationSpinner_2.Position = [39 8 56 22];
            app.MotorRotationSpinner_2.Value = 90;


            % Create AngleSpinnerLabel_2
            app.AngleSpinnerLabel_2 = uilabel(app.Motor2Panel);
            app.AngleSpinnerLabel_2.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_2.Position = [4 8 36 22];
            app.AngleSpinnerLabel_2.Text = 'Angle';


            % Create Motor3Panel
            app.Motor3Panel = uipanel(app.BraidPatternSetup);
            app.Motor3Panel.TitlePosition = 'centertop';
            app.Motor3Panel.Title = 'Motor 3';
            app.Motor3Panel.Position = [235 731 100 102];


            % Create MotorStateButton_3
            app.MotorStateButton_3 = uibutton(app.Motor3Panel, 'state');
            app.MotorStateButton_3.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_3.Text = 'Disabled';
            app.MotorStateButton_3.BackgroundColor = [1 0 0];
```

```
            app.MotorStateButton_3.Position = [19 58 62 22];
            app.MotorStateButton_3.Value = true;

            % Create MotorDirectionButton_3
            app.MotorDirectionButton_3 = uibutton(app.Motor3Panel, 'state');
            app.MotorDirectionButton_3.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_3.Enable = 'off';
            app.MotorDirectionButton_3.Text = 'CCW';
            app.MotorDirectionButton_3.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_3.Position = [19 37 62 22];

            % Create MotorRotationSpinner_3
            app.MotorRotationSpinner_3 = uispinner(app.Motor3Panel);
            app.MotorRotationSpinner_3.Step = 90;
            app.MotorRotationSpinner_3.Limits = [90 360];
            app.MotorRotationSpinner_3.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_3.Enable = 'off';
            app.MotorRotationSpinner_3.Position = [39 8 56 22];
            app.MotorRotationSpinner_3.Value = 90;

            % Create AngleSpinnerLabel_3
            app.AngleSpinnerLabel_3 = uilabel(app.Motor3Panel);
            app.AngleSpinnerLabel_3.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_3.Position = [4 8 36 22];
            app.AngleSpinnerLabel_3.Text = 'Angle';

            % Create Motor4Panel
            app.Motor4Panel = uipanel(app.BraidPatternSetup);
            app.Motor4Panel.TitlePosition = 'centertop';
            app.Motor4Panel.Title = 'Motor 4';
            app.Motor4Panel.Position = [343 731 100 102];

            % Create MotorStateButton_4
            app.MotorStateButton_4 = uibutton(app.Motor4Panel, 'state');
            app.MotorStateButton_4.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_4.Text = 'Disabled';
            app.MotorStateButton_4.BackgroundColor = [1 0 0];
            app.MotorStateButton_4.Position = [19 58 62 22];
            app.MotorStateButton_4.Value = true;

            % Create MotorDirectionButton_4
            app.MotorDirectionButton_4 = uibutton(app.Motor4Panel, 'state');
```

```
            app.MotorDirectionButton_4.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_4.Enable = 'off';
            app.MotorDirectionButton_4.Text = 'CCW';
            app.MotorDirectionButton_4.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_4.Position = [19 37 62 22];

            % Create MotorRotationSpinner_4
            app.MotorRotationSpinner_4 = uispinner(app.Motor4Panel);
            app.MotorRotationSpinner_4.Step = 90;
            app.MotorRotationSpinner_4.Limits = [90 360];
            app.MotorRotationSpinner_4.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_4.Enable = 'off';
            app.MotorRotationSpinner_4.Position = [39 8 56 22];
            app.MotorRotationSpinner_4.Value = 90;

            % Create AngleSpinnerLabel_4
            app.AngleSpinnerLabel_4 = uilabel(app.Motor4Panel);
            app.AngleSpinnerLabel_4.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_4.Position = [4 8 36 22];
            app.AngleSpinnerLabel_4.Text = 'Angle';

            % Create Motor5Panel
            app.Motor5Panel = uipanel(app.BraidPatternSetup);
            app.Motor5Panel.TitlePosition = 'centertop';
            app.Motor5Panel.Title = 'Motor 5';
            app.Motor5Panel.Position = [451 731 100 102];

            % Create MotorStateButton_5
            app.MotorStateButton_5 = uibutton(app.Motor5Panel, 'state');
            app.MotorStateButton_5.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_5.Text = 'Disabled';
            app.MotorStateButton_5.BackgroundColor = [1 0 0];
            app.MotorStateButton_5.Position = [19 58 62 22];
            app.MotorStateButton_5.Value = true;

            % Create MotorDirectionButton_5
            app.MotorDirectionButton_5 = uibutton(app.Motor5Panel, 'state');
            app.MotorDirectionButton_5.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_5.Enable = 'off';
            app.MotorDirectionButton_5.Text = 'CCW';
            app.MotorDirectionButton_5.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_5.Position = [19 37 62 22];
```

```
            % Create MotorRotationSpinner_5
            app.MotorRotationSpinner_5 = uispinner(app.Motor5Panel);
            app.MotorRotationSpinner_5.Step = 90;
            app.MotorRotationSpinner_5.Limits = [90 360];
            app.MotorRotationSpinner_5.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_5.Enable = 'off';
            app.MotorRotationSpinner_5.Position = [39 8 56 22];
            app.MotorRotationSpinner_5.Value = 90;

            % Create AngleSpinnerLabel_5
            app.AngleSpinnerLabel_5 = uilabel(app.Motor5Panel);
            app.AngleSpinnerLabel_5.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_5.Position = [4 8 36 22];
            app.AngleSpinnerLabel_5.Text = 'Angle';

            % Create Motor6Panel
            app.Motor6Panel = uipanel(app.BraidPatternSetup);
            app.Motor6Panel.TitlePosition = 'centertop';
            app.Motor6Panel.Title = 'Motor 6';
            app.Motor6Panel.Position = [558 731 100 102];

            % Create MotorStateButton_6
            app.MotorStateButton_6 = uibutton(app.Motor6Panel, 'state');
            app.MotorStateButton_6.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_6.Text = 'Disabled';
            app.MotorStateButton_6.BackgroundColor = [1 0 0];
            app.MotorStateButton_6.Position = [19 58 62 22];
            app.MotorStateButton_6.Value = true;

            % Create MotorDirectionButton_6
            app.MotorDirectionButton_6 = uibutton(app.Motor6Panel, 'state');
            app.MotorDirectionButton_6.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_6.Enable = 'off';
            app.MotorDirectionButton_6.Text = 'CCW';
            app.MotorDirectionButton_6.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_6.Position = [19 37 62 22];

            % Create MotorRotationSpinner_6
            app.MotorRotationSpinner_6 = uispinner(app.Motor6Panel);
            app.MotorRotationSpinner_6.Step = 90;
            app.MotorRotationSpinner_6.Limits = [90 360];
```

```
            app.MotorRotationSpinner_6.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_6.Enable = 'off';
            app.MotorRotationSpinner_6.Position = [39 8 56 22];
            app.MotorRotationSpinner_6.Value = 90;

            % Create AngleSpinnerLabel_6
            app.AngleSpinnerLabel_6 = uilabel(app.Motor6Panel);
            app.AngleSpinnerLabel_6.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_6.Position = [4 8 36 22];
            app.AngleSpinnerLabel_6.Text = 'Angle';

            % Create Motor7Panel
            app.Motor7Panel = uipanel(app.BraidPatternSetup);
            app.Motor7Panel.TitlePosition = 'centertop';
            app.Motor7Panel.Title = 'Motor 7';
            app.Motor7Panel.Position = [665 731 100 102];

            % Create MotorStateButton_7
            app.MotorStateButton_7 = uibutton(app.Motor7Panel, 'state');
            app.MotorStateButton_7.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_7.Text = 'Disabled';
            app.MotorStateButton_7.BackgroundColor = [1 0 0];
            app.MotorStateButton_7.Position = [19 58 62 22];
            app.MotorStateButton_7.Value = true;

            % Create MotorDirectionButton_7
            app.MotorDirectionButton_7 = uibutton(app.Motor7Panel, 'state');
            app.MotorDirectionButton_7.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_7.Enable = 'off';
            app.MotorDirectionButton_7.Text = 'CCW';
            app.MotorDirectionButton_7.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_7.Position = [19 37 62 22];

            % Create MotorRotationSpinner_7
            app.MotorRotationSpinner_7 = uispinner(app.Motor7Panel);
            app.MotorRotationSpinner_7.Step = 90;
            app.MotorRotationSpinner_7.Limits = [90 360];
            app.MotorRotationSpinner_7.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_7.Enable = 'off';
            app.MotorRotationSpinner_7.Position = [39 8 56 22];
            app.MotorRotationSpinner_7.Value = 90;
```

```
            % Create AngleSpinnerLabel_7
            app.AngleSpinnerLabel_7 = uilabel(app.Motor7Panel);
            app.AngleSpinnerLabel_7.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_7.Position = [4 8 36 22];
            app.AngleSpinnerLabel_7.Text = 'Angle';


            % Create Motor8Panel
            app.Motor8Panel = uipanel(app.BraidPatternSetup);
            app.Motor8Panel.TitlePosition = 'centertop';
            app.Motor8Panel.Title = 'Motor 8';
            app.Motor8Panel.Position = [19 622 100 102];


            % Create MotorStateButton_8
            app.MotorStateButton_8 = uibutton(app.Motor8Panel, 'state');
            app.MotorStateButton_8.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_8.Text = 'Disabled';
            app.MotorStateButton_8.BackgroundColor = [1 0 0];
            app.MotorStateButton_8.Position = [19 58 62 22];
            app.MotorStateButton_8.Value = true;


            % Create MotorDirectionButton_8
            app.MotorDirectionButton_8 = uibutton(app.Motor8Panel, 'state');
            app.MotorDirectionButton_8.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_8.Enable = 'off';
            app.MotorDirectionButton_8.Text = 'CCW';
            app.MotorDirectionButton_8.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_8.Position = [19 37 62 22];


            % Create MotorRotationSpinner_8
            app.MotorRotationSpinner_8 = uispinner(app.Motor8Panel);
            app.MotorRotationSpinner_8.Step = 90;
            app.MotorRotationSpinner_8.Limits = [90 360];
            app.MotorRotationSpinner_8.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_8.Enable = 'off';
            app.MotorRotationSpinner_8.Position = [39 8 56 22];
            app.MotorRotationSpinner_8.Value = 90;


            % Create AngleSpinnerLabel_8
            app.AngleSpinnerLabel_8 = uilabel(app.Motor8Panel);
            app.AngleSpinnerLabel_8.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_8.Position = [4 8 36 22];
            app.AngleSpinnerLabel_8.Text = 'Angle';
```

```
            % Create Motor9Panel
            app.Motor9Panel = uipanel(app.BraidPatternSetup);
            app.Motor9Panel.TitlePosition = 'centertop';
            app.Motor9Panel.Title = 'Motor 9';
            app.Motor9Panel.Position = [127 622 100 102];


            % Create MotorStateButton_9
            app.MotorStateButton_9 = uibutton(app.Motor9Panel, 'state');
            app.MotorStateButton_9.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_9.Text = 'Disabled';
            app.MotorStateButton_9.BackgroundColor = [1 0 0];
            app.MotorStateButton_9.Position = [19 58 62 22];
            app.MotorStateButton_9.Value = true;


            % Create MotorDirectionButton_9
            app.MotorDirectionButton_9 = uibutton(app.Motor9Panel, 'state');
            app.MotorDirectionButton_9.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_9.Enable = 'off';
            app.MotorDirectionButton_9.Text = 'CCW';
            app.MotorDirectionButton_9.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_9.Position = [19 37 62 22];


            % Create MotorRotationSpinner_9
            app.MotorRotationSpinner_9 = uispinner(app.Motor9Panel);
            app.MotorRotationSpinner_9.Step = 90;
            app.MotorRotationSpinner_9.Limits = [90 360];
            app.MotorRotationSpinner_9.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_9.Enable = 'off';
            app.MotorRotationSpinner_9.Position = [36 8 56 22];
            app.MotorRotationSpinner_9.Value = 90;


            % Create AngleSpinnerLabel_9
            app.AngleSpinnerLabel_9 = uilabel(app.Motor9Panel);
            app.AngleSpinnerLabel_9.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_9.Position = [1 8 36 22];
            app.AngleSpinnerLabel_9.Text = 'Angle';


            % Create Motor10Panel
            app.Motor10Panel = uipanel(app.BraidPatternSetup);
            app.Motor10Panel.TitlePosition = 'centertop';
            app.Motor10Panel.Title = 'Motor 10';
            app.Motor10Panel.Position = [235 622 100 102];
```

```
            % Create MotorStateButton_10
            app.MotorStateButton_10 = uibutton(app.Motor10Panel, 'state');
            app.MotorStateButton_10.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_10.Text = 'Disabled';
            app.MotorStateButton_10.BackgroundColor = [1 0 0];
            app.MotorStateButton_10.Position = [19 58 62 22];
            app.MotorStateButton_10.Value = true;

            % Create MotorDirectionButton_10
            app.MotorDirectionButton_10 = uibutton(app.Motor10Panel, 'state');
            app.MotorDirectionButton_10.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_10.Enable = 'off';
            app.MotorDirectionButton_10.Text = 'CCW';
            app.MotorDirectionButton_10.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_10.Position = [19 37 62 22];

            % Create MotorRotationSpinner_10
            app.MotorRotationSpinner_10 = uispinner(app.Motor10Panel);
            app.MotorRotationSpinner_10.Step = 90;
            app.MotorRotationSpinner_10.Limits = [90 360];
            app.MotorRotationSpinner_10.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_10.Enable = 'off';
            app.MotorRotationSpinner_10.Position = [39 8 56 22];
            app.MotorRotationSpinner_10.Value = 90;

            % Create AngleSpinnerLabel_10
            app.AngleSpinnerLabel_10 = uilabel(app.Motor10Panel);
            app.AngleSpinnerLabel_10.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_10.Position = [4 8 36 22];
            app.AngleSpinnerLabel_10.Text = 'Angle';

            % Create Motor11Panel
            app.Motor11Panel = uipanel(app.BraidPatternSetup);
            app.Motor11Panel.TitlePosition = 'centertop';
            app.Motor11Panel.Title = 'Motor 11';
            app.Motor11Panel.Position = [343 622 100 102];

            % Create MotorStateButton_11
            app.MotorStateButton_11 = uibutton(app.Motor11Panel, 'state');
            app.MotorStateButton_11.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_11.Text = 'Disabled';
            app.MotorStateButton_11.BackgroundColor = [1 0 0];
```

```
            app.MotorStateButton_11.Position = [19 58 62 22];
            app.MotorStateButton_11.Value = true;

            % Create MotorDirectionButton_11
            app.MotorDirectionButton_11 = uibutton(app.Motor11Panel, 'state');
            app.MotorDirectionButton_11.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_11.Enable = 'off';
            app.MotorDirectionButton_11.Text = 'CCW';
            app.MotorDirectionButton_11.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_11.Position = [19 37 62 22];

            % Create MotorRotationSpinner_11
            app.MotorRotationSpinner_11 = uispinner(app.Motor11Panel);
            app.MotorRotationSpinner_11.Step = 90;
            app.MotorRotationSpinner_11.Limits = [90 360];
            app.MotorRotationSpinner_11.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_11.Enable = 'off';
            app.MotorRotationSpinner_11.Position = [39 8 56 22];
            app.MotorRotationSpinner_11.Value = 90;

            % Create AngleSpinnerLabel_11
            app.AngleSpinnerLabel_11 = uilabel(app.Motor11Panel);
            app.AngleSpinnerLabel_11.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_11.Position = [4 8 36 22];
            app.AngleSpinnerLabel_11.Text = 'Angle';

            % Create Motor12Panel
            app.Motor12Panel = uipanel(app.BraidPatternSetup);
            app.Motor12Panel.TitlePosition = 'centertop';
            app.Motor12Panel.Title = 'Motor 12';
            app.Motor12Panel.Position = [451 622 100 102];

            % Create MotorStateButton_12
            app.MotorStateButton_12 = uibutton(app.Motor12Panel, 'state');
            app.MotorStateButton_12.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_12.Text = 'Disabled';
            app.MotorStateButton_12.BackgroundColor = [1 0 0];
            app.MotorStateButton_12.Position = [19 58 62 22];
            app.MotorStateButton_12.Value = true;

            % Create MotorDirectionButton_12
            app.MotorDirectionButton_12 = uibutton(app.Motor12Panel, 'state');
```

```
            app.MotorDirectionButton_12.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_12.Enable = 'off';
            app.MotorDirectionButton_12.Text = 'CCW';
            app.MotorDirectionButton_12.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_12.Position = [19 37 62 22];

            % Create MotorRotationSpinner_12
            app.MotorRotationSpinner_12 = uispinner(app.Motor12Panel);
            app.MotorRotationSpinner_12.Step = 90;
            app.MotorRotationSpinner_12.Limits = [90 360];
            app.MotorRotationSpinner_12.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_12.Enable = 'off';
            app.MotorRotationSpinner_12.Position = [39 8 56 22];
            app.MotorRotationSpinner_12.Value = 90;

            % Create AngleSpinnerLabel_12
            app.AngleSpinnerLabel_12 = uilabel(app.Motor12Panel);
            app.AngleSpinnerLabel_12.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_12.Position = [4 8 36 22];
            app.AngleSpinnerLabel_12.Text = 'Angle';

            % Create Motor13Panel
            app.Motor13Panel = uipanel(app.BraidPatternSetup);
            app.Motor13Panel.TitlePosition = 'centertop';
            app.Motor13Panel.Title = 'Motor 13';
            app.Motor13Panel.Position = [558 622 100 102];

            % Create MotorStateButton_13
            app.MotorStateButton_13 = uibutton(app.Motor13Panel, 'state');
            app.MotorStateButton_13.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_13.Text = 'Disabled';
            app.MotorStateButton_13.BackgroundColor = [1 0 0];
            app.MotorStateButton_13.Position = [19 58 62 22];
            app.MotorStateButton_13.Value = true;

            % Create MotorDirectionButton_13
            app.MotorDirectionButton_13 = uibutton(app.Motor13Panel, 'state');
            app.MotorDirectionButton_13.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_13.Enable = 'off';
            app.MotorDirectionButton_13.Text = 'CCW';
            app.MotorDirectionButton_13.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_13.Position = [19 37 62 22];
```

```
            % Create MotorRotationSpinner_13
            app.MotorRotationSpinner_13 = uispinner(app.Motor13Panel);
            app.MotorRotationSpinner_13.Step = 90;
            app.MotorRotationSpinner_13.Limits = [90 360];
            app.MotorRotationSpinner_13.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_13.Enable = 'off';
            app.MotorRotationSpinner_13.Position = [39 8 56 22];
            app.MotorRotationSpinner_13.Value = 90;


            % Create AngleSpinnerLabel_13
            app.AngleSpinnerLabel_13 = uilabel(app.Motor13Panel);
            app.AngleSpinnerLabel_13.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_13.Position = [4 8 36 22];
            app.AngleSpinnerLabel_13.Text = 'Angle';


            % Create Motor14Panel
            app.Motor14Panel = uipanel(app.BraidPatternSetup);
            app.Motor14Panel.TitlePosition = 'centertop';
            app.Motor14Panel.Title = 'Motor 14';
            app.Motor14Panel.Position = [665 622 100 102];


            % Create MotorStateButton_14
            app.MotorStateButton_14 = uibutton(app.Motor14Panel, 'state');
            app.MotorStateButton_14.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_14.Text = 'Disabled';
            app.MotorStateButton_14.BackgroundColor = [1 0 0];
            app.MotorStateButton_14.Position = [19 58 62 22];
            app.MotorStateButton_14.Value = true;


            % Create MotorDirectionButton_14
            app.MotorDirectionButton_14 = uibutton(app.Motor14Panel, 'state');
            app.MotorDirectionButton_14.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_14.Enable = 'off';
            app.MotorDirectionButton_14.Text = 'CCW';
            app.MotorDirectionButton_14.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_14.Position = [19 37 62 22];


            % Create MotorRotationSpinner_14
            app.MotorRotationSpinner_14 = uispinner(app.Motor14Panel);
            app.MotorRotationSpinner_14.Step = 90;
            app.MotorRotationSpinner_14.Limits = [90 360];
```

```
            app.MotorRotationSpinner_14.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_14.Enable = 'off';
            app.MotorRotationSpinner_14.Position = [39 8 56 22];
            app.MotorRotationSpinner_14.Value = 90;

            % Create AngleSpinnerLabel_14
            app.AngleSpinnerLabel_14 = uilabel(app.Motor14Panel);
            app.AngleSpinnerLabel_14.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_14.Position = [4 8 36 22];
            app.AngleSpinnerLabel_14.Text = 'Angle';

            % Create Motor15Panel
            app.Motor15Panel = uipanel(app.BraidPatternSetup);
            app.Motor15Panel.TitlePosition = 'centertop';
            app.Motor15Panel.Title = 'Motor 15';
            app.Motor15Panel.Position = [19 513 100 102];

            % Create MotorStateButton_15
            app.MotorStateButton_15 = uibutton(app.Motor15Panel, 'state');
            app.MotorStateButton_15.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_15.Text = 'Disabled';
            app.MotorStateButton_15.BackgroundColor = [1 0 0];
            app.MotorStateButton_15.Position = [19 58 62 22];
            app.MotorStateButton_15.Value = true;

            % Create MotorDirectionButton_15
            app.MotorDirectionButton_15 = uibutton(app.Motor15Panel, 'state');
            app.MotorDirectionButton_15.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_15.Enable = 'off';
            app.MotorDirectionButton_15.Text = 'CCW';
            app.MotorDirectionButton_15.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_15.Position = [19 37 62 22];

            % Create MotorRotationSpinner_15
            app.MotorRotationSpinner_15 = uispinner(app.Motor15Panel);
            app.MotorRotationSpinner_15.Step = 90;
            app.MotorRotationSpinner_15.Limits = [90 360];
            app.MotorRotationSpinner_15.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_15.Enable = 'off';
            app.MotorRotationSpinner_15.Position = [39 8 56 22];
            app.MotorRotationSpinner_15.Value = 90;
```

```
            % Create AngleSpinnerLabel_15
            app.AngleSpinnerLabel_15 = uilabel(app.Motor15Panel);
            app.AngleSpinnerLabel_15.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_15.Position = [4 8 36 22];
            app.AngleSpinnerLabel_15.Text = 'Angle';

            % Create Motor16Panel
            app.Motor16Panel = uipanel(app.BraidPatternSetup);
            app.Motor16Panel.TitlePosition = 'centertop';
            app.Motor16Panel.Title = 'Motor 16';
            app.Motor16Panel.Position = [127 513 100 102];

            % Create MotorStateButton_16
            app.MotorStateButton_16 = uibutton(app.Motor16Panel, 'state');
            app.MotorStateButton_16.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_16.Text = 'Disabled';
            app.MotorStateButton_16.BackgroundColor = [1 0 0];
            app.MotorStateButton_16.Position = [19 58 62 22];
            app.MotorStateButton_16.Value = true;

            % Create MotorDirectionButton_16
            app.MotorDirectionButton_16 = uibutton(app.Motor16Panel, 'state');
            app.MotorDirectionButton_16.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_16.Enable = 'off';
            app.MotorDirectionButton_16.Text = 'CCW';
            app.MotorDirectionButton_16.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_16.Position = [19 37 62 22];

            % Create MotorRotationSpinner_16
            app.MotorRotationSpinner_16 = uispinner(app.Motor16Panel);
            app.MotorRotationSpinner_16.Step = 90;
            app.MotorRotationSpinner_16.Limits = [90 360];
            app.MotorRotationSpinner_16.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_16.Enable = 'off';
            app.MotorRotationSpinner_16.Position = [39 8 56 22];
            app.MotorRotationSpinner_16.Value = 90;

            % Create AngleSpinnerLabel_16
            app.AngleSpinnerLabel_16 = uilabel(app.Motor16Panel);
            app.AngleSpinnerLabel_16.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_16.Position = [4 8 36 22];
            app.AngleSpinnerLabel_16.Text = 'Angle';
```

```
            % Create Motor17Panel
            app.Motor17Panel = uipanel(app.BraidPatternSetup);
            app.Motor17Panel.TitlePosition = 'centertop';
            app.Motor17Panel.Title = 'Motor 17';
            app.Motor17Panel.Position = [235 513 100 102];


            % Create MotorStateButton_17
            app.MotorStateButton_17 = uibutton(app.Motor17Panel, 'state');
            app.MotorStateButton_17.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_17.Text = 'Disabled';
            app.MotorStateButton_17.BackgroundColor = [1 0 0];
            app.MotorStateButton_17.Position = [19 58 62 22];
            app.MotorStateButton_17.Value = true;


            % Create MotorDirectionButton_17
            app.MotorDirectionButton_17 = uibutton(app.Motor17Panel, 'state');
            app.MotorDirectionButton_17.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_17.Enable = 'off';
            app.MotorDirectionButton_17.Text = 'CCW';
            app.MotorDirectionButton_17.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_17.Position = [19 37 62 22];


            % Create MotorRotationSpinner_17
            app.MotorRotationSpinner_17 = uispinner(app.Motor17Panel);
            app.MotorRotationSpinner_17.Step = 90;
            app.MotorRotationSpinner_17.Limits = [90 360];
            app.MotorRotationSpinner_17.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_17.Enable = 'off';
            app.MotorRotationSpinner_17.Position = [39 8 56 22];
            app.MotorRotationSpinner_17.Value = 90;


            % Create AngleSpinnerLabel_17
            app.AngleSpinnerLabel_17 = uilabel(app.Motor17Panel);
            app.AngleSpinnerLabel_17.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_17.Position = [4 8 36 22];
            app.AngleSpinnerLabel_17.Text = 'Angle';


            % Create Motor18Panel
            app.Motor18Panel = uipanel(app.BraidPatternSetup);
            app.Motor18Panel.TitlePosition = 'centertop';
            app.Motor18Panel.Title = 'Motor 18';
            app.Motor18Panel.Position = [343 513 100 102];
```

```
            % Create MotorStateButton_18
            app.MotorStateButton_18 = uibutton(app.Motor18Panel, 'state');
            app.MotorStateButton_18.ValueChangedFcn = createCallbackFcn(app, 
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_18.Text = 'Disabled';
            app.MotorStateButton_18.BackgroundColor = [1 0 0];
            app.MotorStateButton_18.Position = [19 58 62 22];
            app.MotorStateButton_18.Value = true;


            % Create MotorDirectionButton_18
            app.MotorDirectionButton_18 = uibutton(app.Motor18Panel, 'state');
            app.MotorDirectionButton_18.ValueChangedFcn = createCallbackFcn(app, 
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_18.Enable = 'off';
            app.MotorDirectionButton_18.Text = 'CCW';
            app.MotorDirectionButton_18.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_18.Position = [19 37 62 22];


            % Create MotorRotationSpinner_18
            app.MotorRotationSpinner_18 = uispinner(app.Motor18Panel);
            app.MotorRotationSpinner_18.Step = 90;
            app.MotorRotationSpinner_18.Limits = [90 360];
            app.MotorRotationSpinner_18.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_18.Enable = 'off';
            app.MotorRotationSpinner_18.Position = [39 8 56 22];
            app.MotorRotationSpinner_18.Value = 90;


            % Create AngleSpinnerLabel_18
            app.AngleSpinnerLabel_18 = uilabel(app.Motor18Panel);
            app.AngleSpinnerLabel_18.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_18.Position = [4 8 36 22];
            app.AngleSpinnerLabel_18.Text = 'Angle';


            % Create Motor19Panel
            app.Motor19Panel = uipanel(app.BraidPatternSetup);
            app.Motor19Panel.TitlePosition = 'centertop';
            app.Motor19Panel.Title = 'Motor 19';
            app.Motor19Panel.Position = [451 513 100 102];


            % Create MotorStateButton_19
            app.MotorStateButton_19 = uibutton(app.Motor19Panel, 'state');
            app.MotorStateButton_19.ValueChangedFcn = createCallbackFcn(app, 
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_19.Text = 'Disabled';
            app.MotorStateButton_19.BackgroundColor = [1 0 0];
```

```
            app.MotorStateButton_19.Position = [19 58 62 22];
            app.MotorStateButton_19.Value = true;


            % Create MotorDirectionButton_19
            app.MotorDirectionButton_19 = uibutton(app.Motor19Panel, 'state');
            app.MotorDirectionButton_19.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_19.Enable = 'off';
            app.MotorDirectionButton_19.Text = 'CCW';
            app.MotorDirectionButton_19.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_19.Position = [19 37 62 22];


            % Create MotorRotationSpinner_19
            app.MotorRotationSpinner_19 = uispinner(app.Motor19Panel);
            app.MotorRotationSpinner_19.Step = 90;
            app.MotorRotationSpinner_19.Limits = [90 360];
            app.MotorRotationSpinner_19.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_19.Enable = 'off';
            app.MotorRotationSpinner_19.Position = [39 8 56 22];
            app.MotorRotationSpinner_19.Value = 90;


            % Create AngleSpinnerLabel_19
            app.AngleSpinnerLabel_19 = uilabel(app.Motor19Panel);
            app.AngleSpinnerLabel_19.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_19.Position = [4 8 36 22];
            app.AngleSpinnerLabel_19.Text = 'Angle';


            % Create Motor20Panel
            app.Motor20Panel = uipanel(app.BraidPatternSetup);
            app.Motor20Panel.TitlePosition = 'centertop';
            app.Motor20Panel.Title = 'Motor 20';
            app.Motor20Panel.Position = [558 513 100 102];


            % Create MotorStateButton_20
            app.MotorStateButton_20 = uibutton(app.Motor20Panel, 'state');
            app.MotorStateButton_20.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_20.Text = 'Disabled';
            app.MotorStateButton_20.BackgroundColor = [1 0 0];
            app.MotorStateButton_20.Position = [19 58 62 22];
            app.MotorStateButton_20.Value = true;


            % Create MotorDirectionButton_20
            app.MotorDirectionButton_20 = uibutton(app.Motor20Panel, 'state');
```

```
            app.MotorDirectionButton_20.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_20.Enable = 'off';
            app.MotorDirectionButton_20.Text = 'CCW';
            app.MotorDirectionButton_20.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_20.Position = [19 37 62 22];

            % Create MotorRotationSpinner_20
            app.MotorRotationSpinner_20 = uispinner(app.Motor20Panel);
            app.MotorRotationSpinner_20.Step = 90;
            app.MotorRotationSpinner_20.Limits = [90 360];
            app.MotorRotationSpinner_20.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_20.Enable = 'off';
            app.MotorRotationSpinner_20.Position = [39 8 56 22];
            app.MotorRotationSpinner_20.Value = 90;

            % Create AngleSpinnerLabel_20
            app.AngleSpinnerLabel_20 = uilabel(app.Motor20Panel);
            app.AngleSpinnerLabel_20.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_20.Position = [4 8 36 22];
            app.AngleSpinnerLabel_20.Text = 'Angle';

            % Create Motor21Panel
            app.Motor21Panel = uipanel(app.BraidPatternSetup);
            app.Motor21Panel.TitlePosition = 'centertop';
            app.Motor21Panel.Title = 'Motor 21';
            app.Motor21Panel.Position = [665 513 100 102];

            % Create MotorStateButton_21
            app.MotorStateButton_21 = uibutton(app.Motor21Panel, 'state');
            app.MotorStateButton_21.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_21.Text = 'Disabled';
            app.MotorStateButton_21.BackgroundColor = [1 0 0];
            app.MotorStateButton_21.Position = [19 58 62 22];
            app.MotorStateButton_21.Value = true;

            % Create MotorDirectionButton_21
            app.MotorDirectionButton_21 = uibutton(app.Motor21Panel, 'state');
            app.MotorDirectionButton_21.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_21.Enable = 'off';
            app.MotorDirectionButton_21.Text = 'CCW';
            app.MotorDirectionButton_21.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_21.Position = [19 37 62 22];
```

```
            % Create MotorRotationSpinner_21
            app.MotorRotationSpinner_21 = uispinner(app.Motor21Panel);
            app.MotorRotationSpinner_21.Step = 90;
            app.MotorRotationSpinner_21.Limits = [90 360];
            app.MotorRotationSpinner_21.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_21.Enable = 'off';
            app.MotorRotationSpinner_21.Position = [39 8 56 22];
            app.MotorRotationSpinner_21.Value = 90;

            % Create AngleSpinnerLabel_21
            app.AngleSpinnerLabel_21 = uilabel(app.Motor21Panel);
            app.AngleSpinnerLabel_21.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_21.Position = [4 8 36 22];
            app.AngleSpinnerLabel_21.Text = 'Angle';

            % Create Motor22Panel
            app.Motor22Panel = uipanel(app.BraidPatternSetup);
            app.Motor22Panel.TitlePosition = 'centertop';
            app.Motor22Panel.Title = 'Motor 22';
            app.Motor22Panel.Position = [19 403 100 102];

            % Create MotorStateButton_22
            app.MotorStateButton_22 = uibutton(app.Motor22Panel, 'state');
            app.MotorStateButton_22.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_22.Text = 'Disabled';
            app.MotorStateButton_22.BackgroundColor = [1 0 0];
            app.MotorStateButton_22.Position = [19 58 62 22];
            app.MotorStateButton_22.Value = true;

            % Create MotorDirectionButton_22
            app.MotorDirectionButton_22 = uibutton(app.Motor22Panel, 'state');
            app.MotorDirectionButton_22.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_22.Enable = 'off';
            app.MotorDirectionButton_22.Text = 'CCW';
            app.MotorDirectionButton_22.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_22.Position = [19 37 62 22];

            % Create MotorRotationSpinner_22
            app.MotorRotationSpinner_22 = uispinner(app.Motor22Panel);
            app.MotorRotationSpinner_22.Step = 90;
            app.MotorRotationSpinner_22.Limits = [90 360];
```

```
            app.MotorRotationSpinner_22.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_22.Enable = 'off';
            app.MotorRotationSpinner_22.Position = [39 8 56 22];
            app.MotorRotationSpinner_22.Value = 90;

            % Create AngleSpinnerLabel_22
            app.AngleSpinnerLabel_22 = uilabel(app.Motor22Panel);
            app.AngleSpinnerLabel_22.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_22.Position = [4 8 36 22];
            app.AngleSpinnerLabel_22.Text = 'Angle';

            % Create Motor23Panel
            app.Motor23Panel = uipanel(app.BraidPatternSetup);
            app.Motor23Panel.TitlePosition = 'centertop';
            app.Motor23Panel.Title = 'Motor 23';
            app.Motor23Panel.Position = [127 403 100 102];

            % Create MotorStateButton_23
            app.MotorStateButton_23 = uibutton(app.Motor23Panel, 'state');
            app.MotorStateButton_23.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_23.Text = 'Disabled';
            app.MotorStateButton_23.BackgroundColor = [1 0 0];
            app.MotorStateButton_23.Position = [19 58 62 22];
            app.MotorStateButton_23.Value = true;

            % Create MotorDirectionButton_23
            app.MotorDirectionButton_23 = uibutton(app.Motor23Panel, 'state');
            app.MotorDirectionButton_23.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_23.Enable = 'off';
            app.MotorDirectionButton_23.Text = 'CCW';
            app.MotorDirectionButton_23.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_23.Position = [19 37 62 22];

            % Create MotorRotationSpinner_23
            app.MotorRotationSpinner_23 = uispinner(app.Motor23Panel);
            app.MotorRotationSpinner_23.Step = 90;
            app.MotorRotationSpinner_23.Limits = [90 360];
            app.MotorRotationSpinner_23.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_23.Enable = 'off';
            app.MotorRotationSpinner_23.Position = [39 8 56 22];
            app.MotorRotationSpinner_23.Value = 90;
```

```
            % Create AngleSpinnerLabel_23
            app.AngleSpinnerLabel_23 = uilabel(app.Motor23Panel);
            app.AngleSpinnerLabel_23.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_23.Position = [4 8 36 22];
            app.AngleSpinnerLabel_23.Text = 'Angle';

            % Create Motor24Panel
            app.Motor24Panel = uipanel(app.BraidPatternSetup);
            app.Motor24Panel.TitlePosition = 'centertop';
            app.Motor24Panel.Title = 'Motor 24';
            app.Motor24Panel.Position = [235 403 100 102];

            % Create MotorStateButton_24
            app.MotorStateButton_24 = uibutton(app.Motor24Panel, 'state');
            app.MotorStateButton_24.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_24.Text = 'Disabled';
            app.MotorStateButton_24.BackgroundColor = [1 0 0];
            app.MotorStateButton_24.Position = [19 58 62 22];
            app.MotorStateButton_24.Value = true;

            % Create MotorDirectionButton_24
            app.MotorDirectionButton_24 = uibutton(app.Motor24Panel, 'state');
            app.MotorDirectionButton_24.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_24.Enable = 'off';
            app.MotorDirectionButton_24.Text = 'CCW';
            app.MotorDirectionButton_24.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_24.Position = [19 37 62 22];

            % Create MotorRotationSpinner_24
            app.MotorRotationSpinner_24 = uispinner(app.Motor24Panel);
            app.MotorRotationSpinner_24.Step = 90;
            app.MotorRotationSpinner_24.Limits = [90 360];
            app.MotorRotationSpinner_24.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_24.Enable = 'off';
            app.MotorRotationSpinner_24.Position = [39 8 56 22];
            app.MotorRotationSpinner_24.Value = 90;

            % Create AngleSpinnerLabel_24
            app.AngleSpinnerLabel_24 = uilabel(app.Motor24Panel);
            app.AngleSpinnerLabel_24.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_24.Position = [4 8 36 22];
            app.AngleSpinnerLabel_24.Text = 'Angle';
```

```
            % Create Motor25Panel
            app.Motor25Panel = uipanel(app.BraidPatternSetup);
            app.Motor25Panel.TitlePosition = 'centertop';
            app.Motor25Panel.Title = 'Motor 25';
            app.Motor25Panel.Position = [343 403 100 102];


            % Create MotorStateButton_25
            app.MotorStateButton_25 = uibutton(app.Motor25Panel, 'state');
            app.MotorStateButton_25.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_25.Text = 'Disabled';
            app.MotorStateButton_25.BackgroundColor = [1 0 0];
            app.MotorStateButton_25.Position = [19 58 62 22];
            app.MotorStateButton_25.Value = true;


            % Create MotorDirectionButton_25
            app.MotorDirectionButton_25 = uibutton(app.Motor25Panel, 'state');
            app.MotorDirectionButton_25.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_25.Enable = 'off';
            app.MotorDirectionButton_25.Text = 'CCW';
            app.MotorDirectionButton_25.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_25.Position = [19 37 62 22];


            % Create MotorRotationSpinner_25
            app.MotorRotationSpinner_25 = uispinner(app.Motor25Panel);
            app.MotorRotationSpinner_25.Step = 90;
            app.MotorRotationSpinner_25.Limits = [90 360];
            app.MotorRotationSpinner_25.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_25.Enable = 'off';
            app.MotorRotationSpinner_25.Position = [39 8 56 22];
            app.MotorRotationSpinner_25.Value = 90;


            % Create AngleSpinnerLabel_25
            app.AngleSpinnerLabel_25 = uilabel(app.Motor25Panel);
            app.AngleSpinnerLabel_25.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_25.Position = [4 8 36 22];
            app.AngleSpinnerLabel_25.Text = 'Angle';


            % Create Motor26Panel
            app.Motor26Panel = uipanel(app.BraidPatternSetup);
            app.Motor26Panel.TitlePosition = 'centertop';
            app.Motor26Panel.Title = 'Motor 26';
            app.Motor26Panel.Position = [451 403 100 102];
```

```
            % Create MotorStateButton_26
            app.MotorStateButton_26 = uibutton(app.Motor26Panel, 'state');
            app.MotorStateButton_26.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_26.Text = 'Disabled';
            app.MotorStateButton_26.BackgroundColor = [1 0 0];
            app.MotorStateButton_26.Position = [19 58 62 22];
            app.MotorStateButton_26.Value = true;


            % Create MotorDirectionButton_26
            app.MotorDirectionButton_26 = uibutton(app.Motor26Panel, 'state');
            app.MotorDirectionButton_26.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_26.Enable = 'off';
            app.MotorDirectionButton_26.Text = 'CCW';
            app.MotorDirectionButton_26.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_26.Position = [19 37 62 22];


            % Create MotorRotationSpinner_26
            app.MotorRotationSpinner_26 = uispinner(app.Motor26Panel);
            app.MotorRotationSpinner_26.Step = 90;
            app.MotorRotationSpinner_26.Limits = [90 360];
            app.MotorRotationSpinner_26.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_26.Enable = 'off';
            app.MotorRotationSpinner_26.Position = [39 8 56 22];
            app.MotorRotationSpinner_26.Value = 90;


            % Create AngleSpinnerLabel_26
            app.AngleSpinnerLabel_26 = uilabel(app.Motor26Panel);
            app.AngleSpinnerLabel_26.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_26.Position = [4 8 36 22];
            app.AngleSpinnerLabel_26.Text = 'Angle';


            % Create Motor27Panel
            app.Motor27Panel = uipanel(app.BraidPatternSetup);
            app.Motor27Panel.TitlePosition = 'centertop';
            app.Motor27Panel.Title = 'Motor 27';
            app.Motor27Panel.Position = [558 403 100 102];


            % Create MotorStateButton_27
            app.MotorStateButton_27 = uibutton(app.Motor27Panel, 'state');
            app.MotorStateButton_27.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_27.Text = 'Disabled';
            app.MotorStateButton_27.BackgroundColor = [1 0 0];
```

```
            app.MotorStateButton_27.Position = [19 58 62 22];
            app.MotorStateButton_27.Value = true;

            % Create MotorDirectionButton_27
            app.MotorDirectionButton_27 = uibutton(app.Motor27Panel, 'state');
            app.MotorDirectionButton_27.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_27.Enable = 'off';
            app.MotorDirectionButton_27.Text = 'CCW';
            app.MotorDirectionButton_27.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_27.Position = [19 37 62 22];

            % Create MotorRotationSpinner_27
            app.MotorRotationSpinner_27 = uispinner(app.Motor27Panel);
            app.MotorRotationSpinner_27.Step = 90;
            app.MotorRotationSpinner_27.Limits = [90 360];
            app.MotorRotationSpinner_27.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_27.Enable = 'off';
            app.MotorRotationSpinner_27.Position = [39 8 56 22];
            app.MotorRotationSpinner_27.Value = 90;

            % Create AngleSpinnerLabel_27
            app.AngleSpinnerLabel_27 = uilabel(app.Motor27Panel);
            app.AngleSpinnerLabel_27.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_27.Position = [4 8 36 22];
            app.AngleSpinnerLabel_27.Text = 'Angle';

            % Create Motor28Panel
            app.Motor28Panel = uipanel(app.BraidPatternSetup);
            app.Motor28Panel.TitlePosition = 'centertop';
            app.Motor28Panel.Title = 'Motor 28';
            app.Motor28Panel.Position = [665 403 100 102];

            % Create MotorStateButton_28
            app.MotorStateButton_28 = uibutton(app.Motor28Panel, 'state');
            app.MotorStateButton_28.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_28.Text = 'Disabled';
            app.MotorStateButton_28.BackgroundColor = [1 0 0];
            app.MotorStateButton_28.Position = [19 58 62 22];
            app.MotorStateButton_28.Value = true;

            % Create MotorDirectionButton_28
            app.MotorDirectionButton_28 = uibutton(app.Motor28Panel, 'state');
```

```
            app.MotorDirectionButton_28.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_28.Enable = 'off';
            app.MotorDirectionButton_28.Text = 'CCW';
            app.MotorDirectionButton_28.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_28.Position = [19 37 62 22];

            % Create MotorRotationSpinner_28
            app.MotorRotationSpinner_28 = uispinner(app.Motor28Panel);
            app.MotorRotationSpinner_28.Step = 90;
            app.MotorRotationSpinner_28.Limits = [90 360];
            app.MotorRotationSpinner_28.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_28.Enable = 'off';
            app.MotorRotationSpinner_28.Position = [39 8 56 22];
            app.MotorRotationSpinner_28.Value = 90;

            % Create AngleSpinnerLabel_28
            app.AngleSpinnerLabel_28 = uilabel(app.Motor28Panel);
            app.AngleSpinnerLabel_28.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_28.Position = [4 8 36 22];
            app.AngleSpinnerLabel_28.Text = 'Angle';

            % Create Motor29Panel
            app.Motor29Panel = uipanel(app.BraidPatternSetup);
            app.Motor29Panel.TitlePosition = 'centertop';
            app.Motor29Panel.Title = 'Motor 29';
            app.Motor29Panel.Position = [19 294 100 102];

            % Create MotorStateButton_29
            app.MotorStateButton_29 = uibutton(app.Motor29Panel, 'state');
            app.MotorStateButton_29.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_29.Text = 'Disabled';
            app.MotorStateButton_29.BackgroundColor = [1 0 0];
            app.MotorStateButton_29.Position = [19 58 62 22];
            app.MotorStateButton_29.Value = true;

            % Create MotorDirectionButton_29
            app.MotorDirectionButton_29 = uibutton(app.Motor29Panel, 'state');
            app.MotorDirectionButton_29.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_29.Enable = 'off';
            app.MotorDirectionButton_29.Text = 'CCW';
            app.MotorDirectionButton_29.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_29.Position = [19 37 62 22];
```

```
            % Create MotorRotationSpinner_29
            app.MotorRotationSpinner_29 = uispinner(app.Motor29Panel);
            app.MotorRotationSpinner_29.Step = 90;
            app.MotorRotationSpinner_29.Limits = [90 360];
            app.MotorRotationSpinner_29.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_29.Enable = 'off';
            app.MotorRotationSpinner_29.Position = [39 8 56 22];
            app.MotorRotationSpinner_29.Value = 90;

            % Create AngleSpinnerLabel_29
            app.AngleSpinnerLabel_29 = uilabel(app.Motor29Panel);
            app.AngleSpinnerLabel_29.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_29.Position = [4 8 36 22];
            app.AngleSpinnerLabel_29.Text = 'Angle';

            % Create Motor30Panel
            app.Motor30Panel = uipanel(app.BraidPatternSetup);
            app.Motor30Panel.TitlePosition = 'centertop';
            app.Motor30Panel.Title = 'Motor 30';
            app.Motor30Panel.Position = [127 294 100 102];

            % Create MotorStateButton_30
            app.MotorStateButton_30 = uibutton(app.Motor30Panel, 'state');
            app.MotorStateButton_30.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_30.Text = 'Disabled';
            app.MotorStateButton_30.BackgroundColor = [1 0 0];
            app.MotorStateButton_30.Position = [19 58 62 22];
            app.MotorStateButton_30.Value = true;

            % Create MotorDirectionButton_30
            app.MotorDirectionButton_30 = uibutton(app.Motor30Panel, 'state');
            app.MotorDirectionButton_30.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_30.Enable = 'off';
            app.MotorDirectionButton_30.Text = 'CCW';
            app.MotorDirectionButton_30.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_30.Position = [19 37 62 22];

            % Create MotorRotationSpinner_30
            app.MotorRotationSpinner_30 = uispinner(app.Motor30Panel);
            app.MotorRotationSpinner_30.Step = 90;
            app.MotorRotationSpinner_30.Limits = [90 360];
```

```
            app.MotorRotationSpinner_30.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_30.Enable = 'off';
            app.MotorRotationSpinner_30.Position = [39 8 56 22];
            app.MotorRotationSpinner_30.Value = 90;

            % Create AngleSpinnerLabel_30
            app.AngleSpinnerLabel_30 = uilabel(app.Motor30Panel);
            app.AngleSpinnerLabel_30.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_30.Position = [4 8 36 22];
            app.AngleSpinnerLabel_30.Text = 'Angle';

            % Create Motor31Panel
            app.Motor31Panel = uipanel(app.BraidPatternSetup);
            app.Motor31Panel.TitlePosition = 'centertop';
            app.Motor31Panel.Title = 'Motor 31';
            app.Motor31Panel.Position = [235 294 100 102];

            % Create MotorStateButton_31
            app.MotorStateButton_31 = uibutton(app.Motor31Panel, 'state');
            app.MotorStateButton_31.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_31.Text = 'Disabled';
            app.MotorStateButton_31.BackgroundColor = [1 0 0];
            app.MotorStateButton_31.Position = [19 58 62 22];
            app.MotorStateButton_31.Value = true;

            % Create MotorDirectionButton_31
            app.MotorDirectionButton_31 = uibutton(app.Motor31Panel, 'state');
            app.MotorDirectionButton_31.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_31.Enable = 'off';
            app.MotorDirectionButton_31.Text = 'CCW';
            app.MotorDirectionButton_31.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_31.Position = [19 37 62 22];

            % Create MotorRotationSpinner_31
            app.MotorRotationSpinner_31 = uispinner(app.Motor31Panel);
            app.MotorRotationSpinner_31.Step = 90;
            app.MotorRotationSpinner_31.Limits = [90 360];
            app.MotorRotationSpinner_31.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_31.Enable = 'off';
            app.MotorRotationSpinner_31.Position = [39 8 56 22];
            app.MotorRotationSpinner_31.Value = 90;
```

```
            % Create AngleSpinnerLabel_31
            app.AngleSpinnerLabel_31 = uilabel(app.Motor31Panel);
            app.AngleSpinnerLabel_31.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_31.Position = [4 8 36 22];
            app.AngleSpinnerLabel_31.Text = 'Angle';

            % Create Motor32Panel
            app.Motor32Panel = uipanel(app.BraidPatternSetup);
            app.Motor32Panel.TitlePosition = 'centertop';
            app.Motor32Panel.Title = 'Motor 32';
            app.Motor32Panel.Position = [343 294 100 102];

            % Create MotorStateButton_32
            app.MotorStateButton_32 = uibutton(app.Motor32Panel, 'state');
            app.MotorStateButton_32.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_32.Text = 'Disabled';
            app.MotorStateButton_32.BackgroundColor = [1 0 0];
            app.MotorStateButton_32.Position = [19 58 62 22];
            app.MotorStateButton_32.Value = true;

            % Create MotorDirectionButton_32
            app.MotorDirectionButton_32 = uibutton(app.Motor32Panel, 'state');
            app.MotorDirectionButton_32.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_32.Enable = 'off';
            app.MotorDirectionButton_32.Text = 'CCW';
            app.MotorDirectionButton_32.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_32.Position = [19 37 62 22];

            % Create MotorRotationSpinner_32
            app.MotorRotationSpinner_32 = uispinner(app.Motor32Panel);
            app.MotorRotationSpinner_32.Step = 90;
            app.MotorRotationSpinner_32.Limits = [90 360];
            app.MotorRotationSpinner_32.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_32.Enable = 'off';
            app.MotorRotationSpinner_32.Position = [39 8 56 22];
            app.MotorRotationSpinner_32.Value = 90;

            % Create AngleSpinnerLabel_32
            app.AngleSpinnerLabel_32 = uilabel(app.Motor32Panel);
            app.AngleSpinnerLabel_32.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_32.Position = [4 8 36 22];
            app.AngleSpinnerLabel_32.Text = 'Angle';
```

```
            % Create Motor33Panel
            app.Motor33Panel = uipanel(app.BraidPatternSetup);
            app.Motor33Panel.TitlePosition = 'centertop';
            app.Motor33Panel.Title = 'Motor 33';
            app.Motor33Panel.Position = [451 294 100 102];


            % Create MotorStateButton_33
            app.MotorStateButton_33 = uibutton(app.Motor33Panel, 'state');
            app.MotorStateButton_33.ValueChangedFcn = createCallbackFcn(app, 
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_33.Text = 'Disabled';
            app.MotorStateButton_33.BackgroundColor = [1 0 0];
            app.MotorStateButton_33.Position = [19 58 62 22];
            app.MotorStateButton_33.Value = true;


            % Create MotorDirectionButton_33
            app.MotorDirectionButton_33 = uibutton(app.Motor33Panel, 'state');
            app.MotorDirectionButton_33.ValueChangedFcn = createCallbackFcn(app, 
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_33.Enable = 'off';
            app.MotorDirectionButton_33.Text = 'CCW';
            app.MotorDirectionButton_33.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_33.Position = [19 37 62 22];


            % Create MotorRotationSpinner_33
            app.MotorRotationSpinner_33 = uispinner(app.Motor33Panel);
            app.MotorRotationSpinner_33.Step = 90;
            app.MotorRotationSpinner_33.Limits = [90 360];
            app.MotorRotationSpinner_33.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_33.Enable = 'off';
            app.MotorRotationSpinner_33.Position = [39 8 56 22];
            app.MotorRotationSpinner_33.Value = 90;


            % Create AngleSpinnerLabel_33
            app.AngleSpinnerLabel_33 = uilabel(app.Motor33Panel);
            app.AngleSpinnerLabel_33.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_33.Position = [4 8 36 22];
            app.AngleSpinnerLabel_33.Text = 'Angle';


            % Create Motor34Panel
            app.Motor34Panel = uipanel(app.BraidPatternSetup);
            app.Motor34Panel.TitlePosition = 'centertop';
            app.Motor34Panel.Title = 'Motor 34';
            app.Motor34Panel.Position = [558 294 100 102];
```

```
            % Create MotorStateButton_34
            app.MotorStateButton_34 = uibutton(app.Motor34Panel, 'state');
            app.MotorStateButton_34.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_34.Text = 'Disabled';
            app.MotorStateButton_34.BackgroundColor = [1 0 0];
            app.MotorStateButton_34.Position = [19 58 62 22];
            app.MotorStateButton_34.Value = true;

            % Create MotorDirectionButton_34
            app.MotorDirectionButton_34 = uibutton(app.Motor34Panel, 'state');
            app.MotorDirectionButton_34.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_34.Enable = 'off';
            app.MotorDirectionButton_34.Text = 'CCW';
            app.MotorDirectionButton_34.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_34.Position = [19 37 62 22];

            % Create MotorRotationSpinner_34
            app.MotorRotationSpinner_34 = uispinner(app.Motor34Panel);
            app.MotorRotationSpinner_34.Step = 90;
            app.MotorRotationSpinner_34.Limits = [90 360];
            app.MotorRotationSpinner_34.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_34.Enable = 'off';
            app.MotorRotationSpinner_34.Position = [39 8 56 22];
            app.MotorRotationSpinner_34.Value = 90;

            % Create AngleSpinnerLabel_34
            app.AngleSpinnerLabel_34 = uilabel(app.Motor34Panel);
            app.AngleSpinnerLabel_34.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_34.Position = [4 8 36 22];
            app.AngleSpinnerLabel_34.Text = 'Angle';

            % Create Motor35Panel
            app.Motor35Panel = uipanel(app.BraidPatternSetup);
            app.Motor35Panel.TitlePosition = 'centertop';
            app.Motor35Panel.Title = 'Motor 35';
            app.Motor35Panel.Position = [665 294 100 102];

            % Create MotorStateButton_35
            app.MotorStateButton_35 = uibutton(app.Motor35Panel, 'state');
            app.MotorStateButton_35.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_35.Text = 'Disabled';
            app.MotorStateButton_35.BackgroundColor = [1 0 0];
```

```
            app.MotorStateButton_35.Position = [19 58 62 22];
            app.MotorStateButton_35.Value = true;


            % Create MotorDirectionButton_35
            app.MotorDirectionButton_35 = uibutton(app.Motor35Panel, 'state');
            app.MotorDirectionButton_35.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_35.Enable = 'off';
            app.MotorDirectionButton_35.Text = 'CCW';
            app.MotorDirectionButton_35.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_35.Position = [19 37 62 22];


            % Create MotorRotationSpinner_35
            app.MotorRotationSpinner_35 = uispinner(app.Motor35Panel);
            app.MotorRotationSpinner_35.Step = 90;
            app.MotorRotationSpinner_35.Limits = [90 360];
            app.MotorRotationSpinner_35.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_35.Enable = 'off';
            app.MotorRotationSpinner_35.Position = [39 8 56 22];
            app.MotorRotationSpinner_35.Value = 90;


            % Create AngleSpinnerLabel_35
            app.AngleSpinnerLabel_35 = uilabel(app.Motor35Panel);
            app.AngleSpinnerLabel_35.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_35.Position = [4 8 36 22];
            app.AngleSpinnerLabel_35.Text = 'Angle';


            % Create Motor36Panel
            app.Motor36Panel = uipanel(app.BraidPatternSetup);
            app.Motor36Panel.TitlePosition = 'centertop';
            app.Motor36Panel.Title = 'Motor 36';
            app.Motor36Panel.Position = [19 185 100 102];


            % Create MotorStateButton_36
            app.MotorStateButton_36 = uibutton(app.Motor36Panel, 'state');
            app.MotorStateButton_36.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_36.Text = 'Disabled';
            app.MotorStateButton_36.BackgroundColor = [1 0 0];
            app.MotorStateButton_36.Position = [19 58 62 22];
            app.MotorStateButton_36.Value = true;


            % Create MotorDirectionButton_36
            app.MotorDirectionButton_36 = uibutton(app.Motor36Panel, 'state');
```

```
            app.MotorDirectionButton_36.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_36.Enable = 'off';
            app.MotorDirectionButton_36.Text = 'CCW';
            app.MotorDirectionButton_36.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_36.Position = [19 37 62 22];


            % Create MotorRotationSpinner_36
            app.MotorRotationSpinner_36 = uispinner(app.Motor36Panel);
            app.MotorRotationSpinner_36.Step = 90;
            app.MotorRotationSpinner_36.Limits = [90 360];
            app.MotorRotationSpinner_36.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_36.Enable = 'off';
            app.MotorRotationSpinner_36.Position = [39 8 56 22];
            app.MotorRotationSpinner_36.Value = 90;


            % Create AngleSpinnerLabel_36
            app.AngleSpinnerLabel_36 = uilabel(app.Motor36Panel);
            app.AngleSpinnerLabel_36.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_36.Position = [4 8 36 22];
            app.AngleSpinnerLabel_36.Text = 'Angle';


            % Create Motor37Panel
            app.Motor37Panel = uipanel(app.BraidPatternSetup);
            app.Motor37Panel.TitlePosition = 'centertop';
            app.Motor37Panel.Title = 'Motor 37';
            app.Motor37Panel.Position = [127 185 100 102];


            % Create MotorStateButton_37
            app.MotorStateButton_37 = uibutton(app.Motor37Panel, 'state');
            app.MotorStateButton_37.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_37.Text = 'Disabled';
            app.MotorStateButton_37.BackgroundColor = [1 0 0];
            app.MotorStateButton_37.Position = [19 58 62 22];
            app.MotorStateButton_37.Value = true;


            % Create MotorDirectionButton_37
            app.MotorDirectionButton_37 = uibutton(app.Motor37Panel, 'state');
            app.MotorDirectionButton_37.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_37.Enable = 'off';
            app.MotorDirectionButton_37.Text = 'CCW';
            app.MotorDirectionButton_37.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_37.Position = [19 37 62 22];
```

```
            % Create MotorRotationSpinner_37
            app.MotorRotationSpinner_37 = uispinner(app.Motor37Panel);
            app.MotorRotationSpinner_37.Step = 90;
            app.MotorRotationSpinner_37.Limits = [90 360];
            app.MotorRotationSpinner_37.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_37.Enable = 'off';
            app.MotorRotationSpinner_37.Position = [39 8 56 22];
            app.MotorRotationSpinner_37.Value = 90;

            % Create AngleSpinnerLabel_37
            app.AngleSpinnerLabel_37 = uilabel(app.Motor37Panel);
            app.AngleSpinnerLabel_37.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_37.Position = [4 8 36 22];
            app.AngleSpinnerLabel_37.Text = 'Angle';

            % Create Motor38Panel
            app.Motor38Panel = uipanel(app.BraidPatternSetup);
            app.Motor38Panel.TitlePosition = 'centertop';
            app.Motor38Panel.Title = 'Motor 38';
            app.Motor38Panel.Position = [235 185 100 102];

            % Create MotorStateButton_38
            app.MotorStateButton_38 = uibutton(app.Motor38Panel, 'state');
            app.MotorStateButton_38.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_38.Text = 'Disabled';
            app.MotorStateButton_38.BackgroundColor = [1 0 0];
            app.MotorStateButton_38.Position = [19 58 62 22];
            app.MotorStateButton_38.Value = true;

            % Create MotorDirectionButton_38
            app.MotorDirectionButton_38 = uibutton(app.Motor38Panel, 'state');
            app.MotorDirectionButton_38.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_38.Enable = 'off';
            app.MotorDirectionButton_38.Text = 'CCW';
            app.MotorDirectionButton_38.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_38.Position = [19 37 62 22];

            % Create MotorRotationSpinner_38
            app.MotorRotationSpinner_38 = uispinner(app.Motor38Panel);
            app.MotorRotationSpinner_38.Step = 90;
            app.MotorRotationSpinner_38.Limits = [90 360];
```

```
            app.MotorRotationSpinner_38.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_38.Enable = 'off';
            app.MotorRotationSpinner_38.Position = [39 8 56 22];
            app.MotorRotationSpinner_38.Value = 90;

            % Create AngleSpinnerLabel_38
            app.AngleSpinnerLabel_38 = uilabel(app.Motor38Panel);
            app.AngleSpinnerLabel_38.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_38.Position = [4 8 36 22];
            app.AngleSpinnerLabel_38.Text = 'Angle';

            % Create Motor39Panel
            app.Motor39Panel = uipanel(app.BraidPatternSetup);
            app.Motor39Panel.TitlePosition = 'centertop';
            app.Motor39Panel.Title = 'Motor 39';
            app.Motor39Panel.Position = [343 185 100 102];

            % Create MotorStateButton_39
            app.MotorStateButton_39 = uibutton(app.Motor39Panel, 'state');
            app.MotorStateButton_39.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_39.Text = 'Disabled';
            app.MotorStateButton_39.BackgroundColor = [1 0 0];
            app.MotorStateButton_39.Position = [19 58 62 22];
            app.MotorStateButton_39.Value = true;

            % Create MotorDirectionButton_39
            app.MotorDirectionButton_39 = uibutton(app.Motor39Panel, 'state');
            app.MotorDirectionButton_39.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_39.Enable = 'off';
            app.MotorDirectionButton_39.Text = 'CCW';
            app.MotorDirectionButton_39.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_39.Position = [19 37 62 22];

            % Create MotorRotationSpinner_39
            app.MotorRotationSpinner_39 = uispinner(app.Motor39Panel);
            app.MotorRotationSpinner_39.Step = 90;
            app.MotorRotationSpinner_39.Limits = [90 360];
            app.MotorRotationSpinner_39.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_39.Enable = 'off';
            app.MotorRotationSpinner_39.Position = [39 8 56 22];
            app.MotorRotationSpinner_39.Value = 90;
```

```
            % Create AngleSpinnerLabel_39
            app.AngleSpinnerLabel_39 = uilabel(app.Motor39Panel);
            app.AngleSpinnerLabel_39.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_39.Position = [4 8 36 22];
            app.AngleSpinnerLabel_39.Text = 'Angle';


            % Create Motor40Panel
            app.Motor40Panel = uipanel(app.BraidPatternSetup);
            app.Motor40Panel.TitlePosition = 'centertop';
            app.Motor40Panel.Title = 'Motor 40';
            app.Motor40Panel.Position = [451 185 100 102];


            % Create MotorStateButton_40
            app.MotorStateButton_40 = uibutton(app.Motor40Panel, 'state');
            app.MotorStateButton_40.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_40.Text = 'Disabled';
            app.MotorStateButton_40.BackgroundColor = [1 0 0];
            app.MotorStateButton_40.Position = [19 58 62 22];
            app.MotorStateButton_40.Value = true;


            % Create MotorDirectionButton_40
            app.MotorDirectionButton_40 = uibutton(app.Motor40Panel, 'state');
            app.MotorDirectionButton_40.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_40.Enable = 'off';
            app.MotorDirectionButton_40.Text = 'CCW';
            app.MotorDirectionButton_40.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_40.Position = [19 37 62 22];


            % Create MotorRotationSpinner_40
            app.MotorRotationSpinner_40 = uispinner(app.Motor40Panel);
            app.MotorRotationSpinner_40.Step = 90;
            app.MotorRotationSpinner_40.Limits = [90 360];
            app.MotorRotationSpinner_40.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_40.Enable = 'off';
            app.MotorRotationSpinner_40.Position = [36 8 56 22];
            app.MotorRotationSpinner_40.Value = 90;


            % Create AngleSpinnerLabel_40
            app.AngleSpinnerLabel_40 = uilabel(app.Motor40Panel);
            app.AngleSpinnerLabel_40.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_40.Position = [1 8 36 22];
            app.AngleSpinnerLabel_40.Text = 'Angle';
```

```
            % Create Motor41Panel
            app.Motor41Panel = uipanel(app.BraidPatternSetup);
            app.Motor41Panel.TitlePosition = 'centertop';
            app.Motor41Panel.Title = 'Motor 41';
            app.Motor41Panel.Position = [558 185 100 102];

            % Create MotorStateButton_41
            app.MotorStateButton_41 = uibutton(app.Motor41Panel, 'state');
            app.MotorStateButton_41.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_41.Text = 'Disabled';
            app.MotorStateButton_41.BackgroundColor = [1 0 0];
            app.MotorStateButton_41.Position = [19 58 62 22];
            app.MotorStateButton_41.Value = true;

            % Create MotorDirectionButton_41
            app.MotorDirectionButton_41 = uibutton(app.Motor41Panel, 'state');
            app.MotorDirectionButton_41.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_41.Enable = 'off';
            app.MotorDirectionButton_41.Text = 'CCW';
            app.MotorDirectionButton_41.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_41.Position = [19 37 62 22];

            % Create MotorRotationSpinner_41
            app.MotorRotationSpinner_41 = uispinner(app.Motor41Panel);
            app.MotorRotationSpinner_41.Step = 90;
            app.MotorRotationSpinner_41.Limits = [90 360];
            app.MotorRotationSpinner_41.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_41.Enable = 'off';
            app.MotorRotationSpinner_41.Position = [39 8 56 22];
            app.MotorRotationSpinner_41.Value = 90;

            % Create AngleSpinnerLabel_41
            app.AngleSpinnerLabel_41 = uilabel(app.Motor41Panel);
            app.AngleSpinnerLabel_41.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_41.Position = [4 8 36 22];
            app.AngleSpinnerLabel_41.Text = 'Angle';

            % Create Motor42Panel
            app.Motor42Panel = uipanel(app.BraidPatternSetup);
            app.Motor42Panel.TitlePosition = 'centertop';
            app.Motor42Panel.Title = 'Motor 42';
            app.Motor42Panel.Position = [665 185 100 102];
```

```
            % Create MotorStateButton_42
            app.MotorStateButton_42 = uibutton(app.Motor42Panel, 'state');
            app.MotorStateButton_42.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_42.Text = 'Disabled';
            app.MotorStateButton_42.BackgroundColor = [1 0 0];
            app.MotorStateButton_42.Position = [19 58 62 22];
            app.MotorStateButton_42.Value = true;

            % Create MotorDirectionButton_42
            app.MotorDirectionButton_42 = uibutton(app.Motor42Panel, 'state');
            app.MotorDirectionButton_42.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_42.Enable = 'off';
            app.MotorDirectionButton_42.Text = 'CCW';
            app.MotorDirectionButton_42.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_42.Position = [19 37 62 22];

            % Create MotorRotationSpinner_42
            app.MotorRotationSpinner_42 = uispinner(app.Motor42Panel);
            app.MotorRotationSpinner_42.Step = 90;
            app.MotorRotationSpinner_42.Limits = [90 360];
            app.MotorRotationSpinner_42.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_42.Enable = 'off';
            app.MotorRotationSpinner_42.Position = [39 8 56 22];
            app.MotorRotationSpinner_42.Value = 90;

            % Create AngleSpinnerLabel_42
            app.AngleSpinnerLabel_42 = uilabel(app.Motor42Panel);
            app.AngleSpinnerLabel_42.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_42.Position = [4 8 36 22];
            app.AngleSpinnerLabel_42.Text = 'Angle';

            % Create Motor43Panel
            app.Motor43Panel = uipanel(app.BraidPatternSetup);
            app.Motor43Panel.TitlePosition = 'centertop';
            app.Motor43Panel.Title = 'Motor 43';
            app.Motor43Panel.Position = [19 76 100 102];

            % Create MotorStateButton_43
            app.MotorStateButton_43 = uibutton(app.Motor43Panel, 'state');
            app.MotorStateButton_43.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_43.Text = 'Disabled';
            app.MotorStateButton_43.BackgroundColor = [1 0 0];
```

```
            app.MotorStateButton_43.Position = [19 58 62 22];
            app.MotorStateButton_43.Value = true;

            % Create MotorDirectionButton_43
            app.MotorDirectionButton_43 = uibutton(app.Motor43Panel, 'state');
            app.MotorDirectionButton_43.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_43.Enable = 'off';
            app.MotorDirectionButton_43.Text = 'CCW';
            app.MotorDirectionButton_43.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_43.Position = [19 37 62 22];

            % Create MotorRotationSpinner_43
            app.MotorRotationSpinner_43 = uispinner(app.Motor43Panel);
            app.MotorRotationSpinner_43.Step = 90;
            app.MotorRotationSpinner_43.Limits = [90 360];
            app.MotorRotationSpinner_43.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_43.Enable = 'off';
            app.MotorRotationSpinner_43.Position = [39 8 56 22];
            app.MotorRotationSpinner_43.Value = 90;

            % Create AngleSpinnerLabel_43
            app.AngleSpinnerLabel_43 = uilabel(app.Motor43Panel);
            app.AngleSpinnerLabel_43.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_43.Position = [4 8 36 22];
            app.AngleSpinnerLabel_43.Text = 'Angle';

            % Create Motor44Panel
            app.Motor44Panel = uipanel(app.BraidPatternSetup);
            app.Motor44Panel.TitlePosition = 'centertop';
            app.Motor44Panel.Title = 'Motor 44';
            app.Motor44Panel.Position = [127 76 100 102];

            % Create MotorStateButton_44
            app.MotorStateButton_44 = uibutton(app.Motor44Panel, 'state');
            app.MotorStateButton_44.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_44.Text = 'Disabled';
            app.MotorStateButton_44.BackgroundColor = [1 0 0];
            app.MotorStateButton_44.Position = [19 58 62 22];
            app.MotorStateButton_44.Value = true;

            % Create MotorDirectionButton_44
            app.MotorDirectionButton_44 = uibutton(app.Motor44Panel, 'state');
```

```
            app.MotorDirectionButton_44.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_44.Enable = 'off';
            app.MotorDirectionButton_44.Text = 'CCW';
            app.MotorDirectionButton_44.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_44.Position = [19 37 62 22];


            % Create MotorRotationSpinner_44
            app.MotorRotationSpinner_44 = uispinner(app.Motor44Panel);
            app.MotorRotationSpinner_44.Step = 90;
            app.MotorRotationSpinner_44.Limits = [90 360];
            app.MotorRotationSpinner_44.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_44.Enable = 'off';
            app.MotorRotationSpinner_44.Position = [39 8 56 22];
            app.MotorRotationSpinner_44.Value = 90;


            % Create AngleSpinnerLabel_44
            app.AngleSpinnerLabel_44 = uilabel(app.Motor44Panel);
            app.AngleSpinnerLabel_44.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_44.Position = [4 8 36 22];
            app.AngleSpinnerLabel_44.Text = 'Angle';


            % Create Motor45Panel
            app.Motor45Panel = uipanel(app.BraidPatternSetup);
            app.Motor45Panel.TitlePosition = 'centertop';
            app.Motor45Panel.Title = 'Motor 45';
            app.Motor45Panel.Position = [235 76 100 102];


            % Create MotorStateButton_45
            app.MotorStateButton_45 = uibutton(app.Motor45Panel, 'state');
            app.MotorStateButton_45.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_45.Text = 'Disabled';
            app.MotorStateButton_45.BackgroundColor = [1 0 0];
            app.MotorStateButton_45.Position = [19 58 62 22];
            app.MotorStateButton_45.Value = true;


            % Create MotorDirectionButton_45
            app.MotorDirectionButton_45 = uibutton(app.Motor45Panel, 'state');
            app.MotorDirectionButton_45.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_45.Enable = 'off';
            app.MotorDirectionButton_45.Text = 'CCW';
            app.MotorDirectionButton_45.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_45.Position = [19 37 62 22];
```

```
            % Create MotorRotationSpinner_45
            app.MotorRotationSpinner_45 = uispinner(app.Motor45Panel);
            app.MotorRotationSpinner_45.Step = 90;
            app.MotorRotationSpinner_45.Limits = [90 360];
            app.MotorRotationSpinner_45.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_45.Enable = 'off';
            app.MotorRotationSpinner_45.Position = [39 8 56 22];
            app.MotorRotationSpinner_45.Value = 90;

            % Create AngleSpinnerLabel_45
            app.AngleSpinnerLabel_45 = uilabel(app.Motor45Panel);
            app.AngleSpinnerLabel_45.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_45.Position = [4 8 36 22];
            app.AngleSpinnerLabel_45.Text = 'Angle';

            % Create Motor46Panel
            app.Motor46Panel = uipanel(app.BraidPatternSetup);
            app.Motor46Panel.TitlePosition = 'centertop';
            app.Motor46Panel.Title = 'Motor 46';
            app.Motor46Panel.Position = [343 76 100 102];

            % Create MotorStateButton_46
            app.MotorStateButton_46 = uibutton(app.Motor46Panel, 'state');
            app.MotorStateButton_46.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_46.Text = 'Disabled';
            app.MotorStateButton_46.BackgroundColor = [1 0 0];
            app.MotorStateButton_46.Position = [19 58 62 22];
            app.MotorStateButton_46.Value = true;

            % Create MotorDirectionButton_46
            app.MotorDirectionButton_46 = uibutton(app.Motor46Panel, 'state');
            app.MotorDirectionButton_46.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_46.Enable = 'off';
            app.MotorDirectionButton_46.Text = 'CCW';
            app.MotorDirectionButton_46.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_46.Position = [19 37 62 22];

            % Create MotorRotationSpinner_46
            app.MotorRotationSpinner_46 = uispinner(app.Motor46Panel);
            app.MotorRotationSpinner_46.Step = 90;
            app.MotorRotationSpinner_46.Limits = [90 360];
```

```
            app.MotorRotationSpinner_46.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_46.Enable = 'off';
            app.MotorRotationSpinner_46.Position = [39 8 56 22];
            app.MotorRotationSpinner_46.Value = 90;

            % Create AngleSpinnerLabel_46
            app.AngleSpinnerLabel_46 = uilabel(app.Motor46Panel);
            app.AngleSpinnerLabel_46.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_46.Position = [4 8 36 22];
            app.AngleSpinnerLabel_46.Text = 'Angle';

            % Create Motor47Panel
            app.Motor47Panel = uipanel(app.BraidPatternSetup);
            app.Motor47Panel.TitlePosition = 'centertop';
            app.Motor47Panel.Title = 'Motor 47';
            app.Motor47Panel.Position = [451 76 100 102];

            % Create MotorStateButton_47
            app.MotorStateButton_47 = uibutton(app.Motor47Panel, 'state');
            app.MotorStateButton_47.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_47.Text = 'Disabled';
            app.MotorStateButton_47.BackgroundColor = [1 0 0];
            app.MotorStateButton_47.Position = [19 58 62 22];
            app.MotorStateButton_47.Value = true;

            % Create MotorDirectionButton_47
            app.MotorDirectionButton_47 = uibutton(app.Motor47Panel, 'state');
            app.MotorDirectionButton_47.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_47.Enable = 'off';
            app.MotorDirectionButton_47.Text = 'CCW';
            app.MotorDirectionButton_47.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_47.Position = [19 37 62 22];

            % Create MotorRotationSpinner_47
            app.MotorRotationSpinner_47 = uispinner(app.Motor47Panel);
            app.MotorRotationSpinner_47.Step = 90;
            app.MotorRotationSpinner_47.Limits = [90 360];
            app.MotorRotationSpinner_47.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_47.Enable = 'off';
            app.MotorRotationSpinner_47.Position = [39 8 56 22];
            app.MotorRotationSpinner_47.Value = 90;
```

```
            % Create AngleSpinnerLabel_47
            app.AngleSpinnerLabel_47 = uilabel(app.Motor47Panel);
            app.AngleSpinnerLabel_47.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_47.Position = [4 8 36 22];
            app.AngleSpinnerLabel_47.Text = 'Angle';


            % Create Motor48Panel
            app.Motor48Panel = uipanel(app.BraidPatternSetup);
            app.Motor48Panel.TitlePosition = 'centertop';
            app.Motor48Panel.Title = 'Motor 48';
            app.Motor48Panel.Position = [558 76 100 102];


            % Create MotorStateButton_48
            app.MotorStateButton_48 = uibutton(app.Motor48Panel, 'state');
            app.MotorStateButton_48.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_48.Text = 'Disabled';
            app.MotorStateButton_48.BackgroundColor = [1 0 0];
            app.MotorStateButton_48.Position = [19 58 62 22];
            app.MotorStateButton_48.Value = true;


            % Create MotorDirectionButton_48
            app.MotorDirectionButton_48 = uibutton(app.Motor48Panel, 'state');
            app.MotorDirectionButton_48.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_48.Enable = 'off';
            app.MotorDirectionButton_48.Text = 'CCW';
            app.MotorDirectionButton_48.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_48.Position = [19 37 62 22];


            % Create MotorRotationSpinner_48
            app.MotorRotationSpinner_48 = uispinner(app.Motor48Panel);
            app.MotorRotationSpinner_48.Step = 90;
            app.MotorRotationSpinner_48.Limits = [90 360];
            app.MotorRotationSpinner_48.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_48.Enable = 'off';
            app.MotorRotationSpinner_48.Position = [39 8 56 22];
            app.MotorRotationSpinner_48.Value = 90;


            % Create AngleSpinnerLabel_48
            app.AngleSpinnerLabel_48 = uilabel(app.Motor48Panel);
            app.AngleSpinnerLabel_48.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_48.Position = [4 8 36 22];
            app.AngleSpinnerLabel_48.Text = 'Angle';
```

```
            % Create Motor49Panel
            app.Motor49Panel = uipanel(app.BraidPatternSetup);
            app.Motor49Panel.TitlePosition = 'centertop';
            app.Motor49Panel.Title = 'Motor 49';
            app.Motor49Panel.Position = [665 76 100 102];


            % Create MotorStateButton_49
            app.MotorStateButton_49 = uibutton(app.Motor49Panel, 'state');
            app.MotorStateButton_49.ValueChangedFcn = createCallbackFcn(app,
@MotorStateButtonValueChanged, true);
            app.MotorStateButton_49.Text = 'Disabled';
            app.MotorStateButton_49.BackgroundColor = [1 0 0];
            app.MotorStateButton_49.Position = [19 58 62 22];
            app.MotorStateButton_49.Value = true;


            % Create MotorDirectionButton_49
            app.MotorDirectionButton_49 = uibutton(app.Motor49Panel, 'state');
            app.MotorDirectionButton_49.ValueChangedFcn = createCallbackFcn(app,
@MotorDirectionButtonValueChanged, true);
            app.MotorDirectionButton_49.Enable = 'off';
            app.MotorDirectionButton_49.Text = 'CCW';
            app.MotorDirectionButton_49.BackgroundColor = [1 1 1];
            app.MotorDirectionButton_49.Position = [19 37 62 22];


            % Create MotorRotationSpinner_49
            app.MotorRotationSpinner_49 = uispinner(app.Motor49Panel);
            app.MotorRotationSpinner_49.Step = 90;
            app.MotorRotationSpinner_49.Limits = [90 360];
            app.MotorRotationSpinner_49.HorizontalAlignment = 'left';
            app.MotorRotationSpinner_49.Enable = 'off';
            app.MotorRotationSpinner_49.Position = [39 8 56 22];
            app.MotorRotationSpinner_49.Value = 90;


            % Create AngleSpinnerLabel_49
            app.AngleSpinnerLabel_49 = uilabel(app.Motor49Panel);
            app.AngleSpinnerLabel_49.HorizontalAlignment = 'center';
            app.AngleSpinnerLabel_49.Position = [4 8 36 22];
            app.AngleSpinnerLabel_49.Text = 'Angle';


            % Create MasterMotorRotationSpinner
            app.MasterMotorRotationSpinner = uispinner(app.BraidPatternSetup);
            app.MasterMotorRotationSpinner.Step = 90;
            app.MasterMotorRotationSpinner.Limits = [90 360];
            app.MasterMotorRotationSpinner.ValueChangedFcn =
createCallbackFcn(app, @MasterMotorRotationSpinnerValueChanged, true);
```

```
            app.MasterMotorRotationSpinner.HorizontalAlignment = 'left';
            app.MasterMotorRotationSpinner.Position = [898 811 75 22];
            app.MasterMotorRotationSpinner.Value = 90;


            % Create MasterAngleControlLabel
            app.MasterAngleControlLabel = uilabel(app.BraidPatternSetup);
            app.MasterAngleControlLabel.HorizontalAlignment = 'center';
            app.MasterAngleControlLabel.Position = [771 810 113 22];
            app.MasterAngleControlLabel.Text = 'Master Angle Control';


            % Create MasterMotorDirectionButton
            app.MasterMotorDirectionButton = uibutton(app.BraidPatternSetup,
'state');
            app.MasterMotorDirectionButton.ValueChangedFcn =
createCallbackFcn(app, @MasterMotorDirectionButtonValueChanged, true);
            app.MasterMotorDirectionButton.Text = 'CCW';
            app.MasterMotorDirectionButton.BackgroundColor = [1 1 1];
            app.MasterMotorDirectionButton.Position = [911 781 62 22];


            % Create MasterDirectionControlLabel
            app.MasterDirectionControlLabel = uilabel(app.BraidPatternSetup);
            app.MasterDirectionControlLabel.Position = [771 781 135 22];
            app.MasterDirectionControlLabel.Text = 'Master Direction Control';


            % Create SaveCurrentStepButton
            app.SaveCurrentStepButton = uibutton(app.BraidPatternSetup, 'push');
            app.SaveCurrentStepButton.ButtonPushedFcn = createCallbackFcn(app,
@SaveCurrentStepButtonPushed, true);
            app.SaveCurrentStepButton.BackgroundColor = [0.0745 0.6235 1];
            app.SaveCurrentStepButton.Position = [1065 760 114 43];
            app.SaveCurrentStepButton.Text = 'Save Current Step';


            % Create SavedStepsDropDownLabel
            app.SavedStepsDropDownLabel = uilabel(app.BraidPatternSetup);
            app.SavedStepsDropDownLabel.HorizontalAlignment = 'right';
            app.SavedStepsDropDownLabel.Position = [1027 810 74 22];
            app.SavedStepsDropDownLabel.Text = 'Saved Steps';


            % Create SavedStepsDropDown
            app.SavedStepsDropDown = uidropdown(app.BraidPatternSetup);
            app.SavedStepsDropDown.Items = {'None'};
            app.SavedStepsDropDown.ValueChangedFcn = createCallbackFcn(app,
@SavedStepsDropDownValueChanged, true);
            app.SavedStepsDropDown.Position = [1116 810 100 22];
```

```
            app.SavedStepsDropDown.Value = 'None';


            % Create TakeUpControlButton
            app.TakeUpControlButton = uibutton(app.BraidPatternSetup, 'state');
            app.TakeUpControlButton.ValueChangedFcn = createCallbackFcn(app,
@TakeUpControlButtonValueChanged, true);
            app.TakeUpControlButton.Text = 'Active';
            app.TakeUpControlButton.BackgroundColor = [0 1 0];
            app.TakeUpControlButton.Position = [905 752 68 22];
            app.TakeUpControlButton.Value = true;


            % Create TakeUpControlLabel
            app.TakeUpControlLabel = uilabel(app.BraidPatternSetup);
            app.TakeUpControlLabel.Position = [771 752 90 22];
            app.TakeUpControlLabel.Text = 'Take-up Control';


            % Create TakeupSpeedmmsEditFieldLabel
            app.TakeupSpeedmmsEditFieldLabel = uilabel(app.BraidPatternSetup);
            app.TakeupSpeedmmsEditFieldLabel.Position = [771 723 127 22];
            app.TakeupSpeedmmsEditFieldLabel.Text = 'Take-up Speed (mm/s)';


            % Create TakeUpSpeedEditField
            app.TakeUpSpeedEditField = uieditfield(app.BraidPatternSetup,
'numeric');
            app.TakeUpSpeedEditField.Limits = [0 255];
            app.TakeUpSpeedEditField.RoundFractionalValues = 'on';
            app.TakeUpSpeedEditField.Position = [911 723 62 22];
            app.TakeUpSpeedEditField.Value = 1;


            % Create PresetsDropDownLabel
            app.PresetsDropDownLabel = uilabel(app.BraidPatternSetup);
            app.PresetsDropDownLabel.HorizontalAlignment = 'right';
            app.PresetsDropDownLabel.Position = [1027 722 46 22];
            app.PresetsDropDownLabel.Text = 'Presets';


            % Create PresetsDropDown
            app.PresetsDropDown = uidropdown(app.BraidPatternSetup);
            app.PresetsDropDown.Items = {'None', 'Square 7x7 1', 'Square 7x7 2'};
            app.PresetsDropDown.ValueChangedFcn = createCallbackFcn(app,
@PresetsDropDownValueChanged, true);
            app.PresetsDropDown.Position = [1106 722 110 22];
            app.PresetsDropDown.Value = 'None';
```

```
            % Create DeleteStepButton
            app.DeleteStepButton = uibutton(app.BraidPatternSetup, 'push');
            app.DeleteStepButton.ButtonPushedFcn = createCallbackFcn(app,
@DeleteStepButtonPushed, true);
            app.DeleteStepButton.BackgroundColor = [1 0 0];
            app.DeleteStepButton.Enable = 'off';
            app.DeleteStepButton.Position = [1240 810 79 22];
            app.DeleteStepButton.Text = 'Delete Step';

            % Create DeleteStepDropDown
            app.DeleteStepDropDown = uidropdown(app.BraidPatternSetup);
            app.DeleteStepDropDown.Items = {'N/A', ''};
            app.DeleteStepDropDown.Position = [1328 810 53 22];
            app.DeleteStepDropDown.Value = 'N/A';

            % Create DisableAllButton
            app.DisableAllButton = uibutton(app.BraidPatternSetup, 'push');
            app.DisableAllButton.ButtonPushedFcn = createCallbackFcn(app,
@DisableAllButtonPushed, true);
            app.DisableAllButton.BackgroundColor = [1 0 0];
            app.DisableAllButton.Position = [765 635 101 43];
            app.DisableAllButton.Text = 'Disable All';

            % Create RepeatStepsEditField_min
            app.RepeatStepsEditField_min = uieditfield(app.BraidPatternSetup,
'numeric');
            app.RepeatStepsEditField_min.Limits = [1 Inf];
            app.RepeatStepsEditField_min.ValueChangedFcn = createCallbackFcn(app,
@RepeatStepsEditField_minValueChanged, true);
            app.RepeatStepsEditField_min.Position = [1118 680 36 22];
            app.RepeatStepsEditField_min.Value = 1;

            % Create RepeatStepstoLabel
            app.RepeatStepstoLabel = uilabel(app.BraidPatternSetup);
            app.RepeatStepstoLabel.HorizontalAlignment = 'right';
            app.RepeatStepstoLabel.Position = [1027 660 78 42];
            app.RepeatStepstoLabel.Text = {'Repeat Steps'; ''; 'to'};

            % Create RepeatStepsEditField_max
            app.RepeatStepsEditField_max = uieditfield(app.BraidPatternSetup,
'numeric');
            app.RepeatStepsEditField_max.Limits = [1 Inf];
            app.RepeatStepsEditField_max.ValueChangedFcn = createCallbackFcn(app,
@RepeatStepsEditField_maxValueChanged, true);
            app.RepeatStepsEditField_max.Position = [1118 659 36 22];
```

```
            app.RepeatStepsEditField_max.Value = 1;

            % Create RepeatStepstimesLabel
            app.RepeatStepstimesLabel = uilabel(app.BraidPatternSetup);
            app.RepeatStepstimesLabel.Position = [1033 614 164 22];
            app.RepeatStepstimesLabel.Text = 'Repeat Steps
times';

            % Create RepeatStepsEditField_number
            app.RepeatStepsEditField_number = uieditfield(app.BraidPatternSetup,
'numeric');
            app.RepeatStepsEditField_number.Limits = [1 Inf];
            app.RepeatStepsEditField_number.Position = [1118 614 36 22];
            app.RepeatStepsEditField_number.Value = 1;

            % Create RepeatSavedStepsButton
            app.RepeatSavedStepsButton = uibutton(app.BraidPatternSetup, 'push');
            app.RepeatSavedStepsButton.ButtonPushedFcn = createCallbackFcn(app,
@RepeatSavedStepsButtonPushed, true);
            app.RepeatSavedStepsButton.BackgroundColor = [0.0745 0.6235 1];
            app.RepeatSavedStepsButton.Enable = 'off';
            app.RepeatSavedStepsButton.Position = [1057 563 130 44];
            app.RepeatSavedStepsButton.Text = 'Repeat Saved Steps';

            % Create ClearAllStepsButton
            app.ClearAllStepsButton = uibutton(app.BraidPatternSetup, 'push');
            app.ClearAllStepsButton.ButtonPushedFcn = createCallbackFcn(app,
@ClearAllStepsButtonPushed, true);
            app.ClearAllStepsButton.BackgroundColor = [0.6353 0.0784 0.1843];
            app.ClearAllStepsButton.Enable = 'off';
            app.ClearAllStepsButton.Position = [1240 781 95 22];
            app.ClearAllStepsButton.Text = 'Clear All Steps';

            % Create ReverseTakeupDirectionCheckBox
            app.ReverseTakeupDirectionCheckBox =
uicheckbox(app.BraidPatternSetup);
            app.ReverseTakeupDirectionCheckBox.Text = 'Reverse Take-up
Direction';
            app.ReverseTakeupDirectionCheckBox.Position = [771 692 163 22];

            % Create CreateFullPathButton
            app.CreateFullPathButton = uibutton(app.BraidPatternSetup, 'push');
            app.CreateFullPathButton.ButtonPushedFcn = createCallbackFcn(app,
@CreateFullPathButtonPushed, true);
```

```
            app.CreateFullPathButton.BackgroundColor = [1 1 0.0667];
            app.CreateFullPathButton.Position = [1239 722 102 22];
            app.CreateFullPathButton.Text = 'Create Full Path';


            % Create Modeling
            app.Modeling = uitab(app.TabGroup);
            app.Modeling.Title = 'Modeling';


            % Create btGeneratePlots
            app.btGeneratePlots = uibutton(app.Modeling, 'push');
            app.btGeneratePlots.ButtonPushedFcn = createCallbackFcn(app,
@genPlots, true);
            app.btGeneratePlots.Position = [164 436 120 42];
            app.btGeneratePlots.Text = 'Generate Plots';


            % Create btGenerateModel
            app.btGenerateModel = uibutton(app.Modeling, 'push');
            app.btGenerateModel.ButtonPushedFcn = createCallbackFcn(app,
@genModel, true);
            app.btGenerateModel.Position = [308 436 120 42];
            app.btGenerateModel.Text = 'Generate Model';


            % Create cbSmoothing
            app.cbSmoothing = uicheckbox(app.Modeling);
            app.cbSmoothing.ValueChangedFcn = createCallbackFcn(app,
@enSmoothing, true);
            app.cbSmoothing.Text = 'Enable Smoothing';
            app.cbSmoothing.Position = [393 788 120 16];
            app.cbSmoothing.Value = true;


            % Create cbTriAxial
            app.cbTriAxial = uicheckbox(app.Modeling);
            app.cbTriAxial.ValueChangedFcn = createCallbackFcn(app, @enTriAxial,
true);
            app.cbTriAxial.Text = 'Enable Tri-Axial Yarns ';
            app.cbTriAxial.Position = [393 748 144 16];


            % Create btGeneratePaths
            app.btGeneratePaths = uibutton(app.Modeling, 'push');
            app.btGeneratePaths.ButtonPushedFcn = createCallbackFcn(app,
@genPaths, true);
            app.btGeneratePaths.Position = [19 436 120 42];
            app.btGeneratePaths.Text = 'Generate Paths';
```

```
            % Create btClosePlots
            app.btClosePlots = uibutton(app.Modeling, 'push');
            app.btClosePlots.ButtonPushedFcn = createCallbackFcn(app,
@closePlots, true);
            app.btClosePlots.Position = [452 436 120 42];
            app.btClosePlots.Text = 'Close Plots';


            % Create cbType2Paths
            app.cbType2Paths = uicheckbox(app.Modeling);
            app.cbType2Paths.Enable = 'off';
            app.cbType2Paths.Text = 'Enable Type II';
            app.cbType2Paths.Position = [393 706 97 16];


            % Create cbDenier
            app.cbDenier = uicheckbox(app.Modeling);
            app.cbDenier.ValueChangedFcn = createCallbackFcn(app, @DenierChange,
true);
            app.cbDenier.Text = 'Override Denier';
            app.cbDenier.Position = [651 472 109 15];


            % Create LowerBoundPanel
            app.LowerBoundPanel = uipanel(app.Modeling);
            app.LowerBoundPanel.Title = 'Lower Bound';
            app.LowerBoundPanel.Position = [368 53 291 325];


            % Create tbAngleLamina
            app.tbAngleLamina = uieditfield(app.LowerBoundPanel, 'text');
            app.tbAngleLamina.Editable = 'off';
            app.tbAngleLamina.HorizontalAlignment = 'center';
            app.tbAngleLamina.BackgroundColor = [0.9373 0.9373 0.9373];
            app.tbAngleLamina.Position = [9 257 276 35];


            % Create Label24
            app.Label24 = uilabel(app.LowerBoundPanel);
            app.Label24.HorizontalAlignment = 'center';
            app.Label24.Position = [47 224 20 15];
            app.Label24.Text = 'Ex';


            % Create etAngleLaminaEx
            app.etAngleLaminaEx = uieditfield(app.LowerBoundPanel, 'numeric');
            app.etAngleLaminaEx.Editable = 'off';
            app.etAngleLaminaEx.HorizontalAlignment = 'center';
            app.etAngleLaminaEx.Position = [25 192 64 22];
```

```
% Create Label25
app.Label25 = uilabel(app.LowerBoundPanel);
app.Label25.HorizontalAlignment = 'center';
app.Label25.VerticalAlignment = 'top';
app.Label25.Position = [47 141 20 15];
app.Label25.Text = 'vxy';


% Create etAngleLaminavxy
app.etAngleLaminavxy = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminavxy.Editable = 'off';
app.etAngleLaminavxy.HorizontalAlignment = 'center';
app.etAngleLaminavxy.Position = [25 109 64 22];


% Create GxyLabel
app.GxyLabel = uilabel(app.LowerBoundPanel);
app.GxyLabel.HorizontalAlignment = 'center';
app.GxyLabel.VerticalAlignment = 'top';
app.GxyLabel.Position = [43 47 27 22];
app.GxyLabel.Text = 'Gxy';


% Create etAngleLaminaGxy
app.etAngleLaminaGxy = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminaGxy.Editable = 'off';
app.etAngleLaminaGxy.HorizontalAlignment = 'center';
app.etAngleLaminaGxy.Position = [25 22 64 22];


% Create Label27
app.Label27 = uilabel(app.LowerBoundPanel);
app.Label27.HorizontalAlignment = 'center';
app.Label27.Position = [135 223 20 15];
app.Label27.Text = 'Ey';


% Create etAngleLaminaEy
app.etAngleLaminaEy = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminaEy.Editable = 'off';
app.etAngleLaminaEy.HorizontalAlignment = 'center';
app.etAngleLaminaEy.Position = [113 191 64 22];


% Create Label28
app.Label28 = uilabel(app.LowerBoundPanel);
app.Label28.HorizontalAlignment = 'center';
app.Label28.VerticalAlignment = 'top';
app.Label28.Position = [135 140 20 15];
app.Label28.Text = 'vxz';
```

```
% Create etAngleLaminavxz
app.etAngleLaminavxz = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminavxz.Editable = 'off';
app.etAngleLaminavxz.HorizontalAlignment = 'center';
app.etAngleLaminavxz.Position = [113 108 64 22];


% Create GxzLabel
app.GxzLabel = uilabel(app.LowerBoundPanel);
app.GxzLabel.HorizontalAlignment = 'center';
app.GxzLabel.VerticalAlignment = 'top';
app.GxzLabel.Position = [131 46 27 22];
app.GxzLabel.Text = 'Gxz';


% Create etAngleLaminaGxz
app.etAngleLaminaGxz = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminaGxz.Editable = 'off';
app.etAngleLaminaGxz.HorizontalAlignment = 'center';
app.etAngleLaminaGxz.Position = [113 21 64 22];


% Create Label30
app.Label30 = uilabel(app.LowerBoundPanel);
app.Label30.HorizontalAlignment = 'center';
app.Label30.Position = [223 223 20 15];
app.Label30.Text = 'Ez';


% Create etAngleLaminaEz
app.etAngleLaminaEz = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminaEz.Editable = 'off';
app.etAngleLaminaEz.HorizontalAlignment = 'center';
app.etAngleLaminaEz.Position = [201 191 64 22];


% Create Label31
app.Label31 = uilabel(app.LowerBoundPanel);
app.Label31.HorizontalAlignment = 'center';
app.Label31.VerticalAlignment = 'top';
app.Label31.Position = [223 140 20 15];
app.Label31.Text = 'vyz';


% Create etAngleLaminavyz
app.etAngleLaminavyz = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminavyz.Editable = 'off';
app.etAngleLaminavyz.HorizontalAlignment = 'center';
```

```
app.etAngleLaminavyz.Position = [201 108 64 22];

% Create GyzLabel
app.GyzLabel = uilabel(app.LowerBoundPanel);
app.GyzLabel.HorizontalAlignment = 'center';
app.GyzLabel.VerticalAlignment = 'top';
app.GyzLabel.Position = [219 46 27 22];
app.GyzLabel.Text = 'Gyz';

% Create etAngleLaminaGyz
app.etAngleLaminaGyz = uieditfield(app.LowerBoundPanel, 'numeric');
app.etAngleLaminaGyz.Editable = 'off';
app.etAngleLaminaGyz.HorizontalAlignment = 'center';
app.etAngleLaminaGyz.Position = [201 21 64 22];

% Create UpperBoundPanel
app.UpperBoundPanel = uipanel(app.Modeling);
app.UpperBoundPanel.Title = 'Upper Bound';
app.UpperBoundPanel.Position = [27 53 297 325];

% Create tbLamina
app.tbLamina = uieditfield(app.UpperBoundPanel, 'text');
app.tbLamina.Editable = 'off';
app.tbLamina.HorizontalAlignment = 'center';
app.tbLamina.BackgroundColor = [0.9373 0.9373 0.9373];
app.tbLamina.Position = [11 260 280 35];

% Create Label15
app.Label15 = uilabel(app.UpperBoundPanel);
app.Label15.HorizontalAlignment = 'center';
app.Label15.Position = [48 224 20 15];
app.Label15.Text = 'E1';

% Create etLaminaE1
app.etLaminaE1 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminaE1.Editable = 'off';
app.etLaminaE1.HorizontalAlignment = 'center';
app.etLaminaE1.Position = [26 192 64 22];

% Create Label16
app.Label16 = uilabel(app.UpperBoundPanel);
app.Label16.HorizontalAlignment = 'center';
app.Label16.VerticalAlignment = 'top';
```

```
app.Label16.Position = [48 141 20 15];
app.Label16.Text = 'v12';


% Create etLaminav12
app.etLaminav12 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminav12.Editable = 'off';
app.etLaminav12.HorizontalAlignment = 'center';
app.etLaminav12.Position = [26 109 64 22];


% Create Label17
app.Label17 = uilabel(app.UpperBoundPanel);
app.Label17.HorizontalAlignment = 'center';
app.Label17.VerticalAlignment = 'top';
app.Label17.Position = [46 54 23 15];
app.Label17.Text = 'G12';


% Create etLaminaG12
app.etLaminaG12 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminaG12.Editable = 'off';
app.etLaminaG12.HorizontalAlignment = 'center';
app.etLaminaG12.Position = [26 22 64 22];


% Create Label18
app.Label18 = uilabel(app.UpperBoundPanel);
app.Label18.HorizontalAlignment = 'center';
app.Label18.Position = [139 224 20 15];
app.Label18.Text = 'E2';


% Create etLaminaE2
app.etLaminaE2 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminaE2.Editable = 'off';
app.etLaminaE2.HorizontalAlignment = 'center';
app.etLaminaE2.Position = [117 192 64 22];


% Create Label19
app.Label19 = uilabel(app.UpperBoundPanel);
app.Label19.HorizontalAlignment = 'center';
app.Label19.VerticalAlignment = 'top';
app.Label19.Position = [139 141 20 15];
app.Label19.Text = 'v13';


% Create etLaminav13
app.etLaminav13 = uieditfield(app.UpperBoundPanel, 'numeric');
```

```
app.etLaminav13.Editable = 'off';
app.etLaminav13.HorizontalAlignment = 'center';
app.etLaminav13.Position = [117 109 64 22];

% Create Label20
app.Label20 = uilabel(app.UpperBoundPanel);
app.Label20.HorizontalAlignment = 'center';
app.Label20.VerticalAlignment = 'top';
app.Label20.Position = [137 54 23 15];
app.Label20.Text = 'G13';

% Create etLaminaG13
app.etLaminaG13 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminaG13.Editable = 'off';
app.etLaminaG13.HorizontalAlignment = 'center';
app.etLaminaG13.Position = [117 22 64 22];

% Create Label21
app.Label21 = uilabel(app.UpperBoundPanel);
app.Label21.HorizontalAlignment = 'center';
app.Label21.Position = [230 223 20 15];
app.Label21.Text = 'E3';

% Create etLaminaE3
app.etLaminaE3 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminaE3.Editable = 'off';
app.etLaminaE3.HorizontalAlignment = 'center';
app.etLaminaE3.Position = [208 191 64 22];

% Create Label22
app.Label22 = uilabel(app.UpperBoundPanel);
app.Label22.HorizontalAlignment = 'center';
app.Label22.VerticalAlignment = 'top';
app.Label22.Position = [230 140 20 15];
app.Label22.Text = 'v23';

% Create etLaminav23
app.etLaminav23 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminav23.Editable = 'off';
app.etLaminav23.HorizontalAlignment = 'center';
app.etLaminav23.Position = [208 108 64 22];

% Create Label23
```

```
app.Label23 = uilabel(app.UpperBoundPanel);
app.Label23.HorizontalAlignment = 'center';
app.Label23.VerticalAlignment = 'top';
app.Label23.Position = [228 53 23 15];
app.Label23.Text = 'G23';


% Create etLaminaG23
app.etLaminaG23 = uieditfield(app.UpperBoundPanel, 'numeric');
app.etLaminaG23.Editable = 'off';
app.etLaminaG23.HorizontalAlignment = 'center';
app.etLaminaG23.Position = [208 21 64 22];


% Create YarnPropertiesPanel
app.YarnPropertiesPanel = uipanel(app.Modeling);
app.YarnPropertiesPanel.Title = 'Yarn Properties';
app.YarnPropertiesPanel.Position = [967 505 403 325];


% Create Label6
app.Label6 = uilabel(app.YarnPropertiesPanel);
app.Label6.HorizontalAlignment = 'center';
app.Label6.Position = [66 218 20 15];
app.Label6.Text = 'E1';


% Create etYarnE1
app.etYarnE1 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnE1.Editable = 'off';
app.etYarnE1.HorizontalAlignment = 'center';
app.etYarnE1.Position = [26 186 100 22];


% Create Label7
app.Label7 = uilabel(app.YarnPropertiesPanel);
app.Label7.HorizontalAlignment = 'center';
app.Label7.VerticalAlignment = 'top';
app.Label7.Position = [66 135 20 15];
app.Label7.Text = 'v12';


% Create etYarnv12
app.etYarnv12 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnv12.Editable = 'off';
app.etYarnv12.HorizontalAlignment = 'center';
app.etYarnv12.Position = [26 103 100 22];


% Create Label8
```

```
app.Label8 = uilabel(app.YarnPropertiesPanel);
app.Label8.HorizontalAlignment = 'center';
app.Label8.VerticalAlignment = 'top';
app.Label8.Position = [64 48 23 15];
app.Label8.Text = 'G12';


% Create etYarnG12
app.etYarnG12 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnG12.Editable = 'off';
app.etYarnG12.HorizontalAlignment = 'center';
app.etYarnG12.Position = [26 16 100 22];


% Create Label9
app.Label9 = uilabel(app.YarnPropertiesPanel);
app.Label9.HorizontalAlignment = 'center';
app.Label9.Position = [192 218 20 15];
app.Label9.Text = 'E2';


% Create etYarnE2
app.etYarnE2 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnE2.Editable = 'off';
app.etYarnE2.HorizontalAlignment = 'center';
app.etYarnE2.Position = [152 186 100 22];


% Create Label10
app.Label10 = uilabel(app.YarnPropertiesPanel);
app.Label10.HorizontalAlignment = 'center';
app.Label10.VerticalAlignment = 'top';
app.Label10.Position = [192 135 20 15];
app.Label10.Text = 'v13';


% Create etYarnv13
app.etYarnv13 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnv13.Editable = 'off';
app.etYarnv13.HorizontalAlignment = 'center';
app.etYarnv13.Position = [152 103 100 22];


% Create Label11
app.Label11 = uilabel(app.YarnPropertiesPanel);
app.Label11.HorizontalAlignment = 'center';
app.Label11.VerticalAlignment = 'top';
app.Label11.Position = [190 48 23 15];
app.Label11.Text = 'G13';
```

```
% Create etYarnG13
app.etYarnG13 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnG13.Editable = 'off';
app.etYarnG13.HorizontalAlignment = 'center';
app.etYarnG13.Position = [152 16 100 22];


% Create Label12
app.Label12 = uilabel(app.YarnPropertiesPanel);
app.Label12.HorizontalAlignment = 'center';
app.Label12.Position = [318 218 20 15];
app.Label12.Text = 'E3';


% Create etYarnE3
app.etYarnE3 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnE3.Editable = 'off';
app.etYarnE3.HorizontalAlignment = 'center';
app.etYarnE3.Position = [278 186 100 22];


% Create Label13
app.Label13 = uilabel(app.YarnPropertiesPanel);
app.Label13.HorizontalAlignment = 'center';
app.Label13.VerticalAlignment = 'top';
app.Label13.Position = [318 135 20 15];
app.Label13.Text = 'v23';


% Create etYarnv23
app.etYarnv23 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnv23.Editable = 'off';
app.etYarnv23.HorizontalAlignment = 'center';
app.etYarnv23.Position = [278 103 100 22];


% Create Label14
app.Label14 = uilabel(app.YarnPropertiesPanel);
app.Label14.HorizontalAlignment = 'center';
app.Label14.VerticalAlignment = 'top';
app.Label14.Position = [316 48 23 15];
app.Label14.Text = 'G23';


% Create etYarnG23
app.etYarnG23 = uieditfield(app.YarnPropertiesPanel, 'numeric');
app.etYarnG23.Editable = 'off';
app.etYarnG23.HorizontalAlignment = 'center';
app.etYarnG23.Position = [278 16 100 22];
```

```
            % Create YarnPackingFractionLabel
            app.YarnPackingFractionLabel = uilabel(app.YarnPropertiesPanel);
            app.YarnPackingFractionLabel.HorizontalAlignment = 'right';
            app.YarnPackingFractionLabel.VerticalAlignment = 'top';
            app.YarnPackingFractionLabel.Position = [82 275 125 15];
            app.YarnPackingFractionLabel.Text = 'Yarn Packing Fraction';


            % Create etYarnPackingFraction
            app.etYarnPackingFraction = uieditfield(app.YarnPropertiesPanel,
'numeric');
            app.etYarnPackingFraction.Limits = [0.05 1];
            app.etYarnPackingFraction.ValueDisplayFormat = '%.3f';
            app.etYarnPackingFraction.ValueChangedFcn = createCallbackFcn(app,
@PF, true);
            app.etYarnPackingFraction.HorizontalAlignment = 'center';
            app.etYarnPackingFraction.Position = [222 271 100 22];
            app.etYarnPackingFraction.Value = 0.75;


            % Create YarnRadiusLabel
            app.YarnRadiusLabel = uilabel(app.YarnPropertiesPanel);
            app.YarnRadiusLabel.HorizontalAlignment = 'right';
            app.YarnRadiusLabel.VerticalAlignment = 'top';
            app.YarnRadiusLabel.Position = [134 242 73 15];
            app.YarnRadiusLabel.Text = 'Yarn Radius';


            % Create etYarnRadius
            app.etYarnRadius = uieditfield(app.YarnPropertiesPanel, 'numeric');
            app.etYarnRadius.Limits = [1e-05 1001];
            app.etYarnRadius.ValueDisplayFormat = '%.3f';
            app.etYarnRadius.Editable = 'off';
            app.etYarnRadius.HorizontalAlignment = 'center';
            app.etYarnRadius.Position = [222 238 100 22];
            app.etYarnRadius.Value = 0.2;


            % Create BraidProperties
            app.BraidProperties = uipanel(app.Modeling);
            app.BraidProperties.Title = 'Braid Properties';
            app.BraidProperties.Position = [1086 53 296 325];


            % Create BraidCrossSectionalDimsEditFieldLabel
            app.BraidCrossSectionalDimsEditFieldLabel =
uilabel(app.BraidProperties);
            app.BraidCrossSectionalDimsEditFieldLabel.HorizontalAlignment =
'right';
            app.BraidCrossSectionalDimsEditFieldLabel.VerticalAlignment = 'top';
```

```
            app.BraidCrossSectionalDimsEditFieldLabel.Position = [6 243 158 15];
            app.BraidCrossSectionalDimsEditFieldLabel.Text = 'Braid Cross-
Sectional Dims.';

            % Create BraidCrossSectionalDimsEditField
            app.BraidCrossSectionalDimsEditField =
uieditfield(app.BraidProperties, 'text');
            app.BraidCrossSectionalDimsEditField.Editable = 'off';
            app.BraidCrossSectionalDimsEditField.HorizontalAlignment = 'center';
            app.BraidCrossSectionalDimsEditField.Position = [179 239 100 22];

            % Create UnitCellHeigthEditFieldLabel
            app.UnitCellHeigthEditFieldLabel = uilabel(app.BraidProperties);
            app.UnitCellHeigthEditFieldLabel.HorizontalAlignment = 'right';
            app.UnitCellHeigthEditFieldLabel.VerticalAlignment = 'top';
            app.UnitCellHeigthEditFieldLabel.Position = [73 210 91 15];
            app.UnitCellHeigthEditFieldLabel.Text = 'Unit Cell Heigth';

            % Create UnitCellHeigthEditField
            app.UnitCellHeigthEditField = uieditfield(app.BraidProperties,
'text');
            app.UnitCellHeigthEditField.Editable = 'off';
            app.UnitCellHeigthEditField.HorizontalAlignment = 'center';
            app.UnitCellHeigthEditField.Position = [179 206 100 22];

            % Create AverageBraidingAngleEditFieldLabel
            app.AverageBraidingAngleEditFieldLabel =
uilabel(app.BraidProperties);
            app.AverageBraidingAngleEditFieldLabel.HorizontalAlignment = 'right';
            app.AverageBraidingAngleEditFieldLabel.VerticalAlignment = 'top';
            app.AverageBraidingAngleEditFieldLabel.Position = [29 177 135 15];
            app.AverageBraidingAngleEditFieldLabel.Text = 'Average Braiding
Angle';

            % Create AverageBraidingAngleEditField
            app.AverageBraidingAngleEditField = uieditfield(app.BraidProperties,
'text');
            app.AverageBraidingAngleEditField.Editable = 'off';
            app.AverageBraidingAngleEditField.HorizontalAlignment = 'center';
            app.AverageBraidingAngleEditField.Position = [179 173 100 22];

            % Create BraidingTightnessEditFieldLabel
            app.BraidingTightnessEditFieldLabel = uilabel(app.BraidProperties);
            app.BraidingTightnessEditFieldLabel.HorizontalAlignment = 'right';
```

```
            app.BraidingTightnessEditFieldLabel.VerticalAlignment = 'top';
            app.BraidingTightnessEditFieldLabel.Position = [57 78 107 15];
            app.BraidingTightnessEditFieldLabel.Text = 'Braiding Tightness';

            % Create BraidingTightnessEditField
            app.BraidingTightnessEditField = uieditfield(app.BraidProperties,
'text');
            app.BraidingTightnessEditField.Editable = 'off';
            app.BraidingTightnessEditField.HorizontalAlignment = 'center';
            app.BraidingTightnessEditField.Position = [179 74 100 22];

            % Create FiberVolumeFractionEditFieldLabel
            app.FiberVolumeFractionEditFieldLabel = uilabel(app.BraidProperties);
            app.FiberVolumeFractionEditFieldLabel.HorizontalAlignment = 'right';
            app.FiberVolumeFractionEditFieldLabel.VerticalAlignment = 'top';
            app.FiberVolumeFractionEditFieldLabel.Position = [39 45 125 15];
            app.FiberVolumeFractionEditFieldLabel.Text = 'Fiber Volume Fraction';

            % Create FiberVolumeFractionEditField
            app.FiberVolumeFractionEditField = uieditfield(app.BraidProperties,
'text');
            app.FiberVolumeFractionEditField.Editable = 'off';
            app.FiberVolumeFractionEditField.HorizontalAlignment = 'center';
            app.FiberVolumeFractionEditField.Position = [179 41 100 22];

            % Create NumberofYarnsEditFieldLabel
            app.NumberofYarnsEditFieldLabel = uilabel(app.BraidProperties);
            app.NumberofYarnsEditFieldLabel.HorizontalAlignment = 'right';
            app.NumberofYarnsEditFieldLabel.VerticalAlignment = 'top';
            app.NumberofYarnsEditFieldLabel.Position = [67 12 97 15];
            app.NumberofYarnsEditFieldLabel.Text = 'Number of Yarns';

            % Create NumberofYarnsEditField
            app.NumberofYarnsEditField = uieditfield(app.BraidProperties,
'text');
            app.NumberofYarnsEditField.Editable = 'off';
            app.NumberofYarnsEditField.HorizontalAlignment = 'center';
            app.NumberofYarnsEditField.Position = [179 8 100 22];

            % Create YoungsModulusRangeEditFieldLabel
            app.YoungsModulusRangeEditFieldLabel = uilabel(app.BraidProperties);
            app.YoungsModulusRangeEditFieldLabel.HorizontalAlignment = 'right';
            app.YoungsModulusRangeEditFieldLabel.VerticalAlignment = 'top';
            app.YoungsModulusRangeEditFieldLabel.Position = [26 277 138 15];
```

```
            app.YoungsModulusRangeEditFieldLabel.Text = 'Young''s Modulus Range';

            % Create YoungsModulusRangeEditField
            app.YoungsModulusRangeEditField = uieditfield(app.BraidProperties,
'text');
            app.YoungsModulusRangeEditField.Editable = 'off';
            app.YoungsModulusRangeEditField.HorizontalAlignment = 'center';
            app.YoungsModulusRangeEditField.Position = [179 273 100 22];

            % Create InteriorBraidingAngleEditFieldLabel
            app.InteriorBraidingAngleEditFieldLabel =
uilabel(app.BraidProperties);
            app.InteriorBraidingAngleEditFieldLabel.HorizontalAlignment =
'right';
            app.InteriorBraidingAngleEditFieldLabel.VerticalAlignment = 'top';
            app.InteriorBraidingAngleEditFieldLabel.Position = [37 144 127 15];
            app.InteriorBraidingAngleEditFieldLabel.Text = 'Interior Braiding
Angle';

            % Create InteriorBraidingAngleEditField
            app.InteriorBraidingAngleEditField = uieditfield(app.BraidProperties,
'text');
            app.InteriorBraidingAngleEditField.Editable = 'off';
            app.InteriorBraidingAngleEditField.HorizontalAlignment = 'center';
            app.InteriorBraidingAngleEditField.Position = [179 140 100 22];

            % Create SurfaceBraidingAngleEditFieldLabel
            app.SurfaceBraidingAngleEditFieldLabel =
uilabel(app.BraidProperties);
            app.SurfaceBraidingAngleEditFieldLabel.HorizontalAlignment = 'right';
            app.SurfaceBraidingAngleEditFieldLabel.VerticalAlignment = 'top';
            app.SurfaceBraidingAngleEditFieldLabel.Position = [32 111 131 15];
            app.SurfaceBraidingAngleEditFieldLabel.Text = 'Surface Braiding
Angle';

            % Create SurfaceBraidingAngleEditField
            app.SurfaceBraidingAngleEditField = uieditfield(app.BraidProperties,
'text');
            app.SurfaceBraidingAngleEditField.Editable = 'off';
            app.SurfaceBraidingAngleEditField.HorizontalAlignment = 'center';
            app.SurfaceBraidingAngleEditField.Position = [178 107 100 22];

            % Create FGMPanel
            app.FGMPanel = uipanel(app.Modeling);
```

```
app.FGMPanel.Title = 'FGM';
app.FGMPanel.Position = [708 53 336 325];


% Create Label24_2
app.Label24_2 = uilabel(app.FGMPanel);
app.Label24_2.HorizontalAlignment = 'center';
app.Label24_2.Position = [69 229 20 15];
app.Label24_2.Text = 'Ex';


% Create etFGMEx
app.etFGMEx = uieditfield(app.FGMPanel, 'numeric');
app.etFGMEx.Editable = 'off';
app.etFGMEx.HorizontalAlignment = 'center';
app.etFGMEx.Position = [47 197 64 22];


% Create Label25_2
app.Label25_2 = uilabel(app.FGMPanel);
app.Label25_2.HorizontalAlignment = 'center';
app.Label25_2.VerticalAlignment = 'top';
app.Label25_2.Position = [69 146 20 15];
app.Label25_2.Text = 'vxy';


% Create etFGMvxy
app.etFGMvxy = uieditfield(app.FGMPanel, 'numeric');
app.etFGMvxy.Editable = 'off';
app.etFGMvxy.HorizontalAlignment = 'center';
app.etFGMvxy.Position = [47 114 64 22];


% Create GxyLabel_2
app.GxyLabel_2 = uilabel(app.FGMPanel);
app.GxyLabel_2.HorizontalAlignment = 'center';
app.GxyLabel_2.VerticalAlignment = 'top';
app.GxyLabel_2.Position = [65 52 27 22];
app.GxyLabel_2.Text = 'Gxy';


% Create etFGMGxy
app.etFGMGxy = uieditfield(app.FGMPanel, 'numeric');
app.etFGMGxy.Editable = 'off';
app.etFGMGxy.HorizontalAlignment = 'center';
app.etFGMGxy.Position = [47 27 64 22];


% Create Label27_2
app.Label27_2 = uilabel(app.FGMPanel);
```

```
app.Label27_2.HorizontalAlignment = 'center';
app.Label27_2.Position = [157 228 20 15];
app.Label27_2.Text = 'Ey';

% Create etFGMEy
app.etFGMEy = uieditfield(app.FGMPanel, 'numeric');
app.etFGMEy.Editable = 'off';
app.etFGMEy.HorizontalAlignment = 'center';
app.etFGMEy.Position = [135 196 64 22];

% Create Label28_2
app.Label28_2 = uilabel(app.FGMPanel);
app.Label28_2.HorizontalAlignment = 'center';
app.Label28_2.VerticalAlignment = 'top';
app.Label28_2.Position = [157 145 20 15];
app.Label28_2.Text = 'vxz';

% Create etFGMvxz
app.etFGMvxz = uieditfield(app.FGMPanel, 'numeric');
app.etFGMvxz.Editable = 'off';
app.etFGMvxz.HorizontalAlignment = 'center';
app.etFGMvxz.Position = [135 113 64 22];

% Create GxzLabel_2
app.GxzLabel_2 = uilabel(app.FGMPanel);
app.GxzLabel_2.HorizontalAlignment = 'center';
app.GxzLabel_2.VerticalAlignment = 'top';
app.GxzLabel_2.Position = [153 51 27 22];
app.GxzLabel_2.Text = 'Gxz';

% Create etFGMGxz
app.etFGMGxz = uieditfield(app.FGMPanel, 'numeric');
app.etFGMGxz.Editable = 'off';
app.etFGMGxz.HorizontalAlignment = 'center';
app.etFGMGxz.Position = [135 26 64 22];

% Create Label30_2
app.Label30_2 = uilabel(app.FGMPanel);
app.Label30_2.HorizontalAlignment = 'center';
app.Label30_2.Position = [245 228 20 15];
app.Label30_2.Text = 'Ez';

% Create etFGMEz
```

```
app.etFGMEz = uieditfield(app.FGMPanel, 'numeric');
app.etFGMEz.Editable = 'off';
app.etFGMEz.HorizontalAlignment = 'center';
app.etFGMEz.Position = [223 196 64 22];


% Create Label31_2
app.Label31_2 = uilabel(app.FGMPanel);
app.Label31_2.HorizontalAlignment = 'center';
app.Label31_2.VerticalAlignment = 'top';
app.Label31_2.Position = [245 145 20 15];
app.Label31_2.Text = 'vyz';


% Create etFGMvyz
app.etFGMvyz = uieditfield(app.FGMPanel, 'numeric');
app.etFGMvyz.Editable = 'off';
app.etFGMvyz.HorizontalAlignment = 'center';
app.etFGMvyz.Position = [223 113 64 22];


% Create GyzLabel_2
app.GyzLabel_2 = uilabel(app.FGMPanel);
app.GyzLabel_2.HorizontalAlignment = 'center';
app.GyzLabel_2.VerticalAlignment = 'top';
app.GyzLabel_2.Position = [241 51 27 22];
app.GyzLabel_2.Text = 'Gyz';


% Create etFGMGyz
app.etFGMGyz = uieditfield(app.FGMPanel, 'numeric');
app.etFGMGyz.Editable = 'off';
app.etFGMGyz.HorizontalAlignment = 'center';
app.etFGMGyz.Position = [223 26 64 22];


% Create AnglesfromGeometryPanel
app.AnglesfromGeometryPanel = uipanel(app.Modeling);
app.AnglesfromGeometryPanel.TitlePosition = 'centertop';
app.AnglesfromGeometryPanel.Title = 'Angles from Geometry';
app.AnglesfromGeometryPanel.Position = [313 496 279 138];


% Create ApproxAverageAngleLabel
app.ApproxAverageAngleLabel = uilabel(app.AnglesfromGeometryPanel);
app.ApproxAverageAngleLabel.HorizontalAlignment = 'right';
app.ApproxAverageAngleLabel.VerticalAlignment = 'top';
app.ApproxAverageAngleLabel.Position = [11 89 131 15];
app.ApproxAverageAngleLabel.Text = 'Approx. Average Angle';
```

```
            % Create etAverageAngle
            app.etAverageAngle = uieditfield(app.AnglesfromGeometryPanel,
'numeric');
            app.etAverageAngle.Editable = 'off';
            app.etAverageAngle.HorizontalAlignment = 'center';
            app.etAverageAngle.Position = [157 85 100 22];

            % Create ApproxInteriorAngleLabel
            app.ApproxInteriorAngleLabel = uilabel(app.AnglesfromGeometryPanel);
            app.ApproxInteriorAngleLabel.HorizontalAlignment = 'right';
            app.ApproxInteriorAngleLabel.VerticalAlignment = 'top';
            app.ApproxInteriorAngleLabel.Position = [19 53 123 15];
            app.ApproxInteriorAngleLabel.Text = 'Approx. Interior Angle';

            % Create etInteriorAngle
            app.etInteriorAngle = uieditfield(app.AnglesfromGeometryPanel,
'numeric');
            app.etInteriorAngle.Editable = 'off';
            app.etInteriorAngle.HorizontalAlignment = 'center';
            app.etInteriorAngle.Position = [157 49 100 22];

            % Create ApproxSurfaceAngleLabel
            app.ApproxSurfaceAngleLabel = uilabel(app.AnglesfromGeometryPanel);
            app.ApproxSurfaceAngleLabel.HorizontalAlignment = 'right';
            app.ApproxSurfaceAngleLabel.VerticalAlignment = 'top';
            app.ApproxSurfaceAngleLabel.Position = [15 15 127 15];
            app.ApproxSurfaceAngleLabel.Text = 'Approx. Surface Angle';

            % Create etSurfaceAngle
            app.etSurfaceAngle = uieditfield(app.AnglesfromGeometryPanel,
'numeric');
            app.etSurfaceAngle.Editable = 'off';
            app.etSurfaceAngle.HorizontalAlignment = 'center';
            app.etSurfaceAngle.Position = [157 11 100 22];

            % Create ButtonGroup
            app.ButtonGroup = uibuttongroup(app.Modeling);
            app.ButtonGroup.SelectionChangedFcn = createCallbackFcn(app,
@modeChange, true);
            app.ButtonGroup.BorderType = 'none';
            app.ButtonGroup.Position = [10 506 21 72];

            % Create enablePitch
            app.enablePitch = uiradiobutton(app.ButtonGroup);
```

```
            app.enablePitch.Text = '';
            app.enablePitch.Position = [3 48 25 15];
            app.enablePitch.Value = true;


            % Create enableAngle
            app.enableAngle = uiradiobutton(app.ButtonGroup);
            app.enableAngle.Text = '';
            app.enableAngle.Position = [3 6 25 15];


            % Create LabelNumericEditField2
            app.LabelNumericEditField2 = uilabel(app.Modeling);
            app.LabelNumericEditField2.HorizontalAlignment = 'right';
            app.LabelNumericEditField2.VerticalAlignment = 'top';
            app.LabelNumericEditField2.Position = [96 765 60 15];
            app.LabelNumericEditField2.Text = 'Path Angle';


            % Create etPathAngle
            app.etPathAngle = uieditfield(app.Modeling, 'numeric');
            app.etPathAngle.Limits = [0 45];
            app.etPathAngle.ValueDisplayFormat = '%.1f';
            app.etPathAngle.ValueChangedFcn = createCallbackFcn(app, @pathAngle,
true);

            app.etPathAngle.HorizontalAlignment = 'center';
            app.etPathAngle.Position = [171 761 100 22];


            % Create LabelNumericEditField3
            app.LabelNumericEditField3 = uilabel(app.Modeling);
            app.LabelNumericEditField3.HorizontalAlignment = 'right';
            app.LabelNumericEditField3.VerticalAlignment = 'top';
            app.LabelNumericEditField3.Position = [93 722 63 15];
            app.LabelNumericEditField3.Text = 'Braid Width';


            % Create etBraidWidth
            app.etBraidWidth = uieditfield(app.Modeling, 'numeric');
            app.etBraidWidth.Limits = [1 20];
            app.etBraidWidth.ValueDisplayFormat = '%.0f';
            app.etBraidWidth.ValueChangedFcn = createCallbackFcn(app, @braidW,
true);

            app.etBraidWidth.Editable = 'off';
            app.etBraidWidth.HorizontalAlignment = 'center';
            app.etBraidWidth.Enable = 'off';
            app.etBraidWidth.Position = [171 718 100 22];
            app.etBraidWidth.Value = 3;
```

```
            % Create LabelNumericEditField4
            app.LabelNumericEditField4 = uilabel(app.Modeling);
            app.LabelNumericEditField4.HorizontalAlignment = 'right';
            app.LabelNumericEditField4.VerticalAlignment = 'top';
            app.LabelNumericEditField4.Position = [74 808 82 15];
            app.LabelNumericEditField4.Text = 'Knots in Spline';


            % Create etKnotsinSpline
            app.etKnotsinSpline = uieditfield(app.Modeling, 'numeric');
            app.etKnotsinSpline.Limits = [5 1000];
            app.etKnotsinSpline.ValueDisplayFormat = '%.0f';
            app.etKnotsinSpline.ValueChangedFcn = createCallbackFcn(app,
@splineKnots, true);
            app.etKnotsinSpline.HorizontalAlignment = 'center';
            app.etKnotsinSpline.Position = [171 804 100 22];
            app.etKnotsinSpline.Value = 10;


            % Create LabelNumericEditField5
            app.LabelNumericEditField5 = uilabel(app.Modeling);
            app.LabelNumericEditField5.HorizontalAlignment = 'right';
            app.LabelNumericEditField5.VerticalAlignment = 'top';
            app.LabelNumericEditField5.Position = [91 679 65 15];
            app.LabelNumericEditField5.Text = 'Braid Depth';


            % Create etBraidDepth
            app.etBraidDepth = uieditfield(app.Modeling, 'numeric');
            app.etBraidDepth.Limits = [1 20];
            app.etBraidDepth.ValueDisplayFormat = '%.0f';
            app.etBraidDepth.ValueChangedFcn = createCallbackFcn(app, @braidL,
true);
            app.etBraidDepth.Editable = 'off';
            app.etBraidDepth.HorizontalAlignment = 'center';
            app.etBraidDepth.Enable = 'off';
            app.etBraidDepth.Position = [171 675 100 22];
            app.etBraidDepth.Value = 3;


            % Create LabelNumericEditField6
            app.LabelNumericEditField6 = uilabel(app.Modeling);
            app.LabelNumericEditField6.HorizontalAlignment = 'right';
            app.LabelNumericEditField6.VerticalAlignment = 'top';
            app.LabelNumericEditField6.Position = [74 637 82 15];
            app.LabelNumericEditField6.Text = 'Spacing Factor';


            % Create etSpacingFactor
            app.etSpacingFactor = uieditfield(app.Modeling, 'numeric');
```

```
            app.etSpacingFactor.Limits = [0.5 50];
            app.etSpacingFactor.ValueDisplayFormat = '%.2f';
            app.etSpacingFactor.ValueChangedFcn = createCallbackFcn(app, @sf,
true);
            app.etSpacingFactor.HorizontalAlignment = 'center';
            app.etSpacingFactor.Position = [171 633 100 22];
            app.etSpacingFactor.Value = 1;


            % Create LabelNumericEditField7
            app.LabelNumericEditField7 = uilabel(app.Modeling);
            app.LabelNumericEditField7.HorizontalAlignment = 'right';
            app.LabelNumericEditField7.VerticalAlignment = 'top';
            app.LabelNumericEditField7.Position = [43 595 113 15];
            app.LabelNumericEditField7.Text = 'Number of Unit Cells';


            % Create etNumofUnitCells
            app.etNumofUnitCells = uieditfield(app.Modeling, 'numeric');
            app.etNumofUnitCells.Limits = [1 20];
            app.etNumofUnitCells.ValueDisplayFormat = '%.0f';
            app.etNumofUnitCells.ValueChangedFcn = createCallbackFcn(app, @uc,
true);
            app.etNumofUnitCells.HorizontalAlignment = 'center';
            app.etNumofUnitCells.Position = [171 591 100 22];
            app.etNumofUnitCells.Value = 3;


            % Create LabelEditField
            app.LabelEditField = uilabel(app.Modeling);
            app.LabelEditField.HorizontalAlignment = 'right';
            app.LabelEditField.VerticalAlignment = 'top';
            app.LabelEditField.Position = [358 654 74 15];
            app.LabelEditField.Text = 'Folder Output';


            % Create etFolderName
            app.etFolderName = uieditfield(app.Modeling, 'text');
            app.etFolderName.ValueChangedFcn = createCallbackFcn(app, @folder,
true);
            app.etFolderName.HorizontalAlignment = 'center';
            app.etFolderName.Position = [447 650 100 22];
            app.etFolderName.Value = 'test';


            % Create LabelDropDown
            app.LabelDropDown = uilabel(app.Modeling);
            app.LabelDropDown.VerticalAlignment = 'top';
            app.LabelDropDown.Position = [613 808 28 15];
            app.LabelDropDown.Text = 'Fiber';
```

```
            % Create ddFiberSelect
            app.ddFiberSelect = uidropdown(app.Modeling);
            app.ddFiberSelect.Items = {'Option 1'};
            app.ddFiberSelect.ValueChangedFcn = createCallbackFcn(app,
@fiberChange, true);
            app.ddFiberSelect.Position = [656 804 100 22];


            % Create LabelDropDown2
            app.LabelDropDown2 = uilabel(app.Modeling);
            app.LabelDropDown2.HorizontalAlignment = 'right';
            app.LabelDropDown2.VerticalAlignment = 'top';
            app.LabelDropDown2.Position = [789 809 33 15];
            app.LabelDropDown2.Text = 'Matrix';


            % Create ddMatrixSelect
            app.ddMatrixSelect = uidropdown(app.Modeling);
            app.ddMatrixSelect.Items = {'Option 1'};
            app.ddMatrixSelect.ValueChangedFcn = createCallbackFcn(app,
@matrixChange, true);
            app.ddMatrixSelect.Position = [837 805 100 22];


            % Create LabelNumericEditField9
            app.LabelNumericEditField9 = uilabel(app.Modeling);
            app.LabelNumericEditField9.HorizontalAlignment = 'center';
            app.LabelNumericEditField9.VerticalAlignment = 'top';
            app.LabelNumericEditField9.Position = [651 758 111 15];
            app.LabelNumericEditField9.Text = 'Logitudinal Modulus';


            % Create etFiberE1
            app.etFiberE1 = uieditfield(app.Modeling, 'numeric');
            app.etFiberE1.Editable = 'off';
            app.etFiberE1.HorizontalAlignment = 'center';
            app.etFiberE1.Position = [656 726 100 22];


            % Create Label
            app.Label = uilabel(app.Modeling);
            app.Label.HorizontalAlignment = 'right';
            app.Label.VerticalAlignment = 'top';
            app.Label.Position = [664 646 84 15];
            app.Label.Text = 'Poisson''s Ratio';


            % Create etFiberv
```

```
app.etFiberv = uieditfield(app.Modeling, 'numeric');
app.etFiberv.Editable = 'off';
app.etFiberv.HorizontalAlignment = 'center';
app.etFiberv.Position = [656 614 100 22];


% Create Label2
app.Label2 = uilabel(app.Modeling);
app.Label2.HorizontalAlignment = 'right';
app.Label2.VerticalAlignment = 'top';
app.Label2.Position = [664 588 83 15];
app.Label2.Text = 'Shear Modulus';


% Create etFiberG
app.etFiberG = uieditfield(app.Modeling, 'numeric');
app.etFiberG.Editable = 'off';
app.etFiberG.HorizontalAlignment = 'center';
app.etFiberG.Position = [656 556 100 22];


% Create Label3
app.Label3 = uilabel(app.Modeling);
app.Label3.HorizontalAlignment = 'right';
app.Label3.VerticalAlignment = 'top';
app.Label3.Position = [840 759 94 15];
app.Label3.Text = 'Young''s Modulus';


% Create etMatrixE
app.etMatrixE = uieditfield(app.Modeling, 'numeric');
app.etMatrixE.Editable = 'off';
app.etMatrixE.HorizontalAlignment = 'center';
app.etMatrixE.Position = [837 727 100 22];


% Create Label4
app.Label4 = uilabel(app.Modeling);
app.Label4.HorizontalAlignment = 'right';
app.Label4.VerticalAlignment = 'top';
app.Label4.Position = [845 676 84 15];
app.Label4.Text = 'Poisson''s Ratio';


% Create etMatrixv
app.etMatrixv = uieditfield(app.Modeling, 'numeric');
app.etMatrixv.Editable = 'off';
app.etMatrixv.HorizontalAlignment = 'center';
app.etMatrixv.Position = [837 644 100 22];
```

```
% Create Label5
app.Label5 = uilabel(app.Modeling);
app.Label5.HorizontalAlignment = 'right';
app.Label5.VerticalAlignment = 'top';
app.Label5.Position = [845 589 83 15];
app.Label5.Text = 'Shear Modulus';


% Create etMatrixG
app.etMatrixG = uieditfield(app.Modeling, 'numeric');
app.etMatrixG.Editable = 'off';
app.etMatrixG.HorizontalAlignment = 'center';
app.etMatrixG.Position = [837 557 100 22];


% Create Label33
app.Label33 = uilabel(app.Modeling);
app.Label33.HorizontalAlignment = 'center';
app.Label33.VerticalAlignment = 'top';
app.Label33.Position = [651 700 111 15];
app.Label33.Text = 'Transverse Modulus';


% Create etFiberE2
app.etFiberE2 = uieditfield(app.Modeling, 'numeric');
app.etFiberE2.Editable = 'off';
app.etFiberE2.HorizontalAlignment = 'center';
app.etFiberE2.Position = [656 668 100 22];


% Create DenierEditFieldLabel
app.DenierEditFieldLabel = uilabel(app.Modeling);
app.DenierEditFieldLabel.HorizontalAlignment = 'center';
app.DenierEditFieldLabel.VerticalAlignment = 'top';
app.DenierEditFieldLabel.Position = [685 532 42 15];
app.DenierEditFieldLabel.Text = 'Denier';


% Create etDenier
app.etDenier = uieditfield(app.Modeling, 'numeric');
app.etDenier.Limits = [1 Inf];
app.etDenier.ValueChangedFcn = createCallbackFcn(app, @DenierValue,
true);

app.etDenier.Editable = 'off';
app.etDenier.HorizontalAlignment = 'center';
app.etDenier.Position = [656 500 100 22];
app.etDenier.Value = 1000;


% Create UnitCellHeightPitchLabel
```

```
            app.UnitCellHeightPitchLabel = uilabel(app.Modeling);
            app.UnitCellHeightPitchLabel.HorizontalAlignment = 'right';
            app.UnitCellHeightPitchLabel.VerticalAlignment = 'top';
            app.UnitCellHeightPitchLabel.Position = [27 553 129 15];
            app.UnitCellHeightPitchLabel.Text = 'Unit Cell Height (Pitch)';


            % Create etZHeight
            app.etZHeight = uieditfield(app.Modeling, 'numeric');
            app.etZHeight.Limits = [1e-06 1000];
            app.etZHeight.ValueChangedFcn = createCallbackFcn(app, @zChange,
true);
            app.etZHeight.HorizontalAlignment = 'center';
            app.etZHeight.Position = [171 549 100 22];
            app.etZHeight.Value = 1.21775;


            % Create BraidAngleAverageLabel
            app.BraidAngleAverageLabel = uilabel(app.Modeling);
            app.BraidAngleAverageLabel.HorizontalAlignment = 'right';
            app.BraidAngleAverageLabel.VerticalAlignment = 'top';
            app.BraidAngleAverageLabel.Position = [30 511 126 15];
            app.BraidAngleAverageLabel.Text = 'Braid Angle (Average)';


            % Create etTAngle
            app.etTAngle = uieditfield(app.Modeling, 'numeric');
            app.etTAngle.Limits = [1e-15 90];
            app.etTAngle.ValueChangedFcn = createCallbackFcn(app, @TAngleChange,
true);
            app.etTAngle.Editable = 'off';
            app.etTAngle.HorizontalAlignment = 'center';
            app.etTAngle.Position = [171 507 100 22];
            app.etTAngle.Value = 2;


            % Create BraidControl
            app.BraidControl = uitab(app.TabGroup);
            app.BraidControl.Title = 'Braid Control';


            % Create CONNECTIONPanel
            app.CONNECTIONPanel = uipanel(app.BraidControl);
            app.CONNECTIONPanel.TitlePosition = 'centertop';
            app.CONNECTIONPanel.Title = 'CONNECTION';
            app.CONNECTIONPanel.FontName = 'Consolas';
            app.CONNECTIONPanel.Position = [52 661 260 146];


            % Create CONNECTButton
```

```
            app.CONNECTButton = uibutton(app.CONNECTIONPanel, 'state');
            app.CONNECTButton.ValueChangedFcn = createCallbackFcn(app,
@attemptConnection, true);
            app.CONNECTButton.Text = 'CONNECT';
            app.CONNECTButton.BackgroundColor = [0 1 0];
            app.CONNECTButton.FontName = 'Consolas';
            app.CONNECTButton.Position = [81 13 100 22];

            % Create COMPORTDropDownLabel
            app.COMPORTDropDownLabel = uilabel(app.CONNECTIONPanel);
            app.COMPORTDropDownLabel.HorizontalAlignment = 'right';
            app.COMPORTDropDownLabel.VerticalAlignment = 'top';
            app.COMPORTDropDownLabel.FontName = 'Consolas';
            app.COMPORTDropDownLabel.Position = [30 60 58 15];
            app.COMPORTDropDownLabel.Text = 'COM PORT';

            % Create COMPORTDropDown
            app.COMPORTDropDown = uidropdown(app.CONNECTIONPanel);
            app.COMPORTDropDown.Items = {'-SELECT PORT-', 'COM0', 'COM1', 'COM2',
'COM3', 'COM4', 'COM5', 'COM6', 'COM7', 'COM8', 'COM9', 'Bypass'};
            app.COMPORTDropDown.FontName = 'Consolas';
            app.COMPORTDropDown.Position = [103 56 130 22];
            app.COMPORTDropDown.Value = '-SELECT PORT-';

            % Create CONNECTIONDisplay
            app.CONNECTIONDisplay = uieditfield(app.CONNECTIONPanel, 'text');
            app.CONNECTIONDisplay.Editable = 'off';
            app.CONNECTIONDisplay.HorizontalAlignment = 'center';
            app.CONNECTIONDisplay.FontName = 'Consolas';
            app.CONNECTIONDisplay.FontColor = [1 0 0];
            app.CONNECTIONDisplay.Position = [35 96 191 22];
            app.CONNECTIONDisplay.Value = '*** NOT CONNECTED ***';

            % Create StartBraidingButton
            app.StartBraidingButton = uibutton(app.BraidControl, 'push');
            app.StartBraidingButton.ButtonPushedFcn = createCallbackFcn(app,
@StartBraidingButtonPushed, true);
            app.StartBraidingButton.Position = [52 619 100 22];
            app.StartBraidingButton.Text = 'Start Braiding';

            % Create PauseButton
            app.PauseButton = uibutton(app.BraidControl, 'state');
            app.PauseButton.ValueChangedFcn = createCallbackFcn(app,
@PauseButtonValueChanged, true);
            app.PauseButton.Enable = 'off';
```

```
            app.PauseButton.Text = 'Resume';
            app.PauseButton.Position = [212 619 100 22];
            app.PauseButton.Value = true;

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = braidGenerator3

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            % Execute the startup function
            runStartupFcn(app, @startupFcn)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)
            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

D.1.1 Braid Generation Sub-function

```
%Acknowledgements: edited from code developed by Daniel Aldrich

%This function provides the main functionality for the modeling tab of the
%braid generating application. Through this function (and sub-functions)
%the paths of each yarn are fully described and can be plotted or modelled
%as desired

%The inputs are as follows:
%Knot: The number of knots in the spline describing the yarn paths between
%each braid step (typically 10)
%
%Path: The path angle (typically 0)
%
%braidLength/Width: The length and width of the braid in terms of number of
%horngears. Only useful for rectangular braids
%
%Radius: Yarn radius based on denier
%
%Spacing Factor: Spacing factor based on geometry to eliminate overlapping
%yarns
%
%unitCells: The number of unit cells to be examined - currently this is
%superceded by StepStruct
%
%folderName: The desired folder name for saving data
%
%enableSmoothing: Logical data to turn on/off zero phase filter to smooth
%yarn paths
%
%enableYarns: Logical data to turn on/off axial yarns - currently
%superceded by CamMatrix
%
%enableTimer: Logical data to turn on/off speed analysis
%
%enablePlots: Logical data to turn on/off plotting function
%
%enableTypeII: Logical data to turn on/off type II path plotting -
%currently superceded through updated Plot_Path function
%
%z: defined pitch from user entry field in braidGenerator application
%
%CamMatrix: Matrix (15x15) of data with locations for all included yarns
%before braiding begins
%
%StepStruct: Structure data including all information of motors in each
%step (motor enable disable, direction, rotation, takeup speed)


function [time, Vf, yarnAngle,FGM] = generateBraidPaths3(Knot,Path,...
    braidLength,braidWidth,Radius,Spacing_Factor,unitCells,folderName,...
    enableSmoothing,enableYarns,enableTimer,enablePlots,enableTypeII,z,...
    CamMatrix,StepStruct)
```

```
system(['mkdir ',folderName]);
res = 15;
Spacing_Factor = Spacing_Factor+enableYarns*(1/sind(45+Path)-1.0+Path/180);
zScale = z/2; %pitch divided by 2 to get the length of a single step
braidingSteps = unitCells*2;
smoothingFactor = floor(Knot/3)*2+1;
repeats = 1;
replicates = 1;
if enableTimer
    repeats = 3;
    replicates = 100;
end
for m = 1:repeats
    tic
    for c = 1:replicates
        clear x y z M
        Path = pi()*Path/180;
        Pattern=Machine_Emulation_3(CamMatrix,StepStruct);
        Size = max(max(Pattern{1}));
        Deadzone = 2;
        number_n = 24;

        Middlex = (braidWidth*2+2)/2;
        Middley = (braidLength*2+2)/2;

        for n = 1:Size
            if enableTypeII
                [x(n,:),y(n,:),z(n,:)] =...
                    Plot_Path_t(Pattern(:),n,1,Knot,Path);
            else
                [x(n,:),y(n,:),z(n,:)] =...
                    Plot_Path2(StepStruct,Pattern(:),n,1,Knot,Path);
                [xFGM(n,:),yFGM(n,:),zFGM(n,:)] = ...
                    Plot_Path2(StepStruct,Pattern(1:3),n,1,1,0);
            end
        end


        Z_n = z;
        X_n = x-Middlex;
        Y_n = y-Middley;

        xFGM = diff((xFGM-Middlex)*Radius*2*Spacing_Factor,1,2);
        yFGM = diff((yFGM-Middley)*Radius*2*Spacing_Factor,1,2);
        mFGM = sqrt(xFGM.^2 + yFGM.^2);
        zFGM = diff(zFGM*zScale,1,2);

        logiFGM = asind(yFGM./mFGM)>=0;

        bFGM = logiFGM.*(acosd(xFGM./mFGM)) +...
            ~logiFGM.*(360 - acosd(xFGM./mFGM));
        bFGM(isnan(bFGM)) = 0;
        tFGM = atand(abs(mFGM./zFGM));

        FGM = [bFGM(:), tFGM(:)];
```

```
        %This loop uses a modified form of Plot_Path2 to force intersecting
        %yarns to rotate around one another
        checkInterfere=1; % for troubleshooting
        if checkInterfere
            new_cols=0;
            for n=1:(Size-1) %for each yarn
                for m=1:(Knot-1):(size(z,2)-1) %for each z
                    delta_x=abs(x(n,m)-round(x(n,m+1)));
                    delta_y=abs(y(n,m)-round(y(n,m+1)));
                    if (delta_x>=1) && (delta_y>=1)
                        disp('we got one!') %for troubleshooting
                        disp(n) %for troubleshooting
                        disp(m)%for troubleshooting


                        points=Knot;

                        mid_x=max([x(n,m-(points/2-1)),x(n,m+(points/2-1))]);
                        mid_y=max([y(n,m-(points/2-1)),y(n,m+(points/2-1))]);
                        %Find the position of the yarn relative to the
                        %midpoint as an angle. "Below" the motor corresponds
to 0, "right"
                        %corresponds to pi/2, "left" corresponds to -pi/2,
and "top"
                        %corresponds to pi
                        [t,~] = cart2pol(x(n,m)-mid_x,y(n,m)-mid_y);

                        Rotation=pi/2;
                        %Creates evenly spaced points (both for angles and z
movement)
                        %between the start and end points of the motor
rotation. The
                        %value of theta is changed if rotation is > 90
degrees
                        Ztheta = linspace(z(n,m-(points/2-
1)),z(n,m+(points/2-1)),points);
                        angles = linspace(0,Rotation,points+1);
                        angles = angles(2:end-1);

                        %Change values depending on the direction of rotation
                        if y(n,m)<y(n,m+1) %if y increase CW rotation
                            [Xtheta,Ytheta] = pol2cart(t-angles,1);
                        else
                            [Xtheta,Ytheta] = pol2cart(t+angles,1);
                        end


                        %Finds the x and y coordinates that are allowable
with the
                        %selected max angle
                        maxAngle=0;

                        temp = Xtheta((angles >= maxAngle) & (angles
<=(Rotation-maxAngle)));
```

```
                        if length(temp)>1
                            x1 = temp(1);
                            x2 = temp(end);

                            temp =
Ytheta((angles>=maxAngle)&(angles<=(Rotation-maxAngle)));

                            y1 = temp(1);
                            y2 = temp(end);
                        end
                        Xtheta = Xtheta + mid_x;
                        Ytheta = Ytheta + mid_y;

                        if length(temp)>1
                            %Find slope and intercept of line from start to
end of
                            %rotation
                            slope = (x2 - x1)/(y2 - y1);
                            b = y2 - slope*x2;

                            %Anonymous function "r" used to define points of
the
                            %line described by the above slope and intercept
                            r = @(t) (b)./(-cos(t)/slope+sin(t));

                            %finds the allowables angles and indices of those
                            %angles
                            ang = angles(angles>=maxAngle &
angles<=(Rotation-maxAngle));
                            ind = find(angles>=maxAngle & angles<=(Rotation-
maxAngle));

                            %Finds the cartesian coordinate equivalent, again
dependent
                            %on direction
                            if y(n,m)<y(n,m+1) %if y increases CW rotation
                                [tempx, tempy] = pol2cart(t-ang,r(t-
ang)+Radius);
                            else %0=CCW
                                [tempx, tempy] =
pol2cart(t+ang,r(t+ang)+2*Radius);
                            end

                            %Finds the difference between consecutive values
in
                            %tempx and tempy (i.e. dx(1)=tempx(2)-tempx(1))
                            dx = diff(tempx);
                            dy = diff(tempy);

                            nlen = sqrt(dx.^2 + dy.^2)/norm([y2 - y1,x2 -
x1]);

                            zlen = Ztheta(ind(end)+1)-Ztheta(ind(1)+1);
                            tempz = cumsum(zlen*nlen);

                            Ztheta(ind+1) = [0,tempz]+Ztheta(ind(1)+1);
```

```
                              %Moves line Xtheta-Ytheta to prevent overlap,
                              %again dependent on direction
                              if y(n,m)<y(n,m+1) %if y increases CW rotation
                                  Xtheta(ind) = tempx + mid_x+4*Radius;
                                  Ytheta(ind) = tempy + mid_y+4*Radius;
                              else %0=CCW
                                  Xtheta(ind) = tempx + mid_x-...
                                      10*Radius;
                                  Ytheta(ind) = tempy + mid_y-...
                                      10*Radius;
                              end
%                               Xtheta(ind) = tempx + mid_x;
%                               Ytheta(ind) = tempy + mid_y;
                          end

                          new_cols=new_cols+1;
                          %define new column for x, y, z
                          m_store(new_cols)=m;
                          n_store(new_cols)=n;
%                           Ztheta_store(new_cols,:)=Ztheta(2:end-1);
                          Xtheta_store(new_cols,:)=Xtheta;
                          Ytheta_store(new_cols,:)=Ytheta;

                      end
                  end
              end

              if (new_cols)~=0
                  previous_m=0;
                  for i=1:new_cols
                      zpoints=length(z);
                      m=m_store(i);
                      n=n_store(i);
                      ins_x_row=Xtheta_store(i,:);
                      ins_y_row=Ytheta_store(i,:);


                      x(n,m-(Knot/2-1):m+(Knot/2-1))=ins_x_row;
                      y(n,m-(Knot/2-1):m+(Knot/2-1))=ins_y_row;


                      previous_m=m;
                  end

              end
          end

          Z_n = z;
          X_n = x-Middlex;
          Y_n = y-Middley;

          for n = 1:Size
              x = X_n(n,:);
              y = Y_n(n,:);
```

```
            z = Z_n(n,:);

            x = [x(1)*[ones(1,20)],x(:)',x(end)*[ones(1,20)]]...
                *Radius*2*Spacing_Factor;
            y = [y(1)*[ones(1,20)],y(:)',y(end)*[ones(1,20)]]...
                *Radius*2*Spacing_Factor;
            if enableSmoothing
                x = filtfilt(ones(1,smoothingFactor)/smoothingFactor,1,x);
                y = filtfilt(ones(1,smoothingFactor)/smoothingFactor,1,y);
            end
            z = [Deadzone/20*[20:-1:1],...
                z(:)'*zScale,...
                -Deadzone/20*[1:20]+z(end)*zScale]...
                -Deadzone;

            if enablePlots
                if n == 1
                    fig1 = figure('Name','Braid Paths',...
                        'units','normalized',...
                        'position',[0,0.55,0.5,0.35]);
                    hold on;
                    fig2 = figure('Name','Braid Geometry Approximation',...
                        'units','normalized',...
                        'position',[0.5,0.55,0.5,0.35]);
                    view([1,1,1])
                    hold on;
                end
                figure(fig1)
                if sum(n == number_n)
                    plot3(x,y,z,'-o','Color','r','LineWidth',2)
                else
                    plot3(x,y,z,':','Color',[0.5 0.5 0.5],'LineWidth',1.8)
                end
                axis equal
                grid on
                figure(fig2)
                tubeplot([x(:),y(:),z(:)]',Radius,res);
                axis equal
                grid on
            end

            M(:,:,n) = [x(:)*100,z(:)*100,y(:)*100];
            filename = sprintf('\\Yarn%i.txt',n);
            csvwrite([folderName,filename],M(:,:,n))
        end

        csvwrite([folderName,'\Data.txt'],...
            [Size, Radius/10, max(abs(z))/10,...
            braidWidth, braidLength, Spacing_Factor]);

        j=1;
    end
    time(m) = toc/100;
end
```

```
M = M/100;
a(size(M,1)-1,size(M,3)) = 0;
dM = diff(M);
yarnArea(size(M,1)-1) = 0;
yarnAngle(size(M,1)-1) = 0;
yAngle(size(M,1)-1,size(M,3)) = 0;
yArea(size(M,1)-1,size(M,3)) = 0;
Vf(size(M,1)-1) = 0;


Area = ((4 * Spacing_Factor * braidWidth + 3) * Radius)...
    *((4 * Spacing_Factor * braidLength + 3) * Radius);

c = 0;
for m = 1:size(M,1)-1
    for n = 1:size(M,3)
        c = c + 1;
        dx = dM(m,1,n);
        dz = dM(m,2,n);
        dy = dM(m,3,n);
        a(m,n) = Radius/(abs(dz)/norm([dx,dy,dz]));
        yAngle(m,n) = atand(norm([dx,dy])/abs(dz));
        yArea(m,n) = pi()*Radius*a(m,n);
        yarnAngle(m) = yarnAngle(m) + atand(norm([dx,dy])/abs(dz));
        yarnArea(m) = yarnArea(m) + pi()*Radius*a(m,n);
    end
    yarnAngle(m) = yarnAngle(m)/n;
    Vf(m) = yarnArea(m)/Area;
end
if enablePlots
    figure('Name','Volume Fraction along Braid Length',...
        'units','normalized',...
        'position',[0,0.05,1/3,0.35])
    plot(-M(1:end-1,2,1),Vf)
end
ind = -M(1:end-1,2,1)>2 & -M(1:end-1,2,1)<-M(end,2,1)-2;
Vf = (mean(Vf(ind)));
if enablePlots
    figure('Name','Average Yarn Angle along Braid Length',...
        'units','normalized',...
        'position',[1/3,0.05,1/3,0.35])
    plot(-M(1:end-1,2,1),yarnAngle)
end
yarnAngle = (mean(yarnAngle(ind)));
if enablePlots
    figure('Name','Yarn Angle along Braid Length',...
        'units','normalized',...
        'position',[2/3,0.05,1/3,0.35])
    for n = 1:size(M,3)
        plot3(-M(1:end-1,2,n),ones(1,size(M,1)-1)*n,yAngle(:,n))
        hold on
    end
end
if enablePlots
    if n == 1
```

```
        fig1 = figure('Name','Braid Paths',...
            'units','normalized',...
            'position',[0,0.55,0.5,0.35]);
        hold on;
        fig2 = figure('Name','Braid Geometry Approximation',...
            'units','normalized',...
            'position',[0.5,0.55,0.5,0.35]);
        view([1,1,1])
        hold on;
    end
    figure(fig1)
    if sum(n == number_n)
        plot3(x,y,z,'-o','Color','r','LineWidth',2)
    else
        plot3(x,y,z,':','Color',[0.5 0.5 0.5],'LineWidth',1.8)
    end
    axis equal
    grid on
    figure(fig2)
    tubeplot([x(:),y(:),z(:)]',Radius,res);
    axis equal
    grid on
end
```

D.1.2 Machine Emulation Sub-function

```
function [Pattern] = Machine_Emulation_3(CamMatrix,StepStruct)
%Acknowledgements: edited from code developed by Daniel Aldrich

%This function outputs the total pattern of the braid in "Pattern"
%The pattern of the braid is the position of the yarns in a 15x15
% matrix after each rotation step

%The inputs are as follows:
%CamMatrix: a 15x15 matrix of the original yarn positions as defined in the
% "braidGenerator" app. Should similarly represent the physical positions
% of the yarns in the braiding machine.
%
%StepStruct: a structure that contains the subsections "motor" and
%"takeup". "motor" contains the subsections "Enable", "Direction", and
%"Rotation" which give the information of the main 49 motors in a given
%step. "takeup" contains the subsections "Enable" and "Speed" which give
%similar information for the take-up (vertical motion) motors.
%
%Repeats: a number that defines the amount of times the steps in
%"StepStruct" repeat.

%The output is a cell matrix 'Pattern' which contains the positions of each
%yarn after each step. The first cell is the position before any motors
%turn
    Braid_start = CamMatrix; %Starts with the original setup of cams/carriers
    StepNum=length(StepStruct); %Determine number of steps to be repeated
    Pattern=cell(1,StepNum+1); %Repeats steps defined in step struct n number
of times
```

```
    %Initialize variables for loop
    n = 1;
    Pattern{1}=Braid_start;
    Braid=Braid_start;

    %Main loop that performs rotation
    while n<StepNum+1
        for i=1:49
            if StepStruct(n).motor(i).Enable==0
               Braid = Rotate(Braid,i,StepStruct(n).motor(i).Direction,...
                   StepStruct(n).motor(i).Rotation);
            end
        end
        n=n+1;
        Pattern{n} = Braid;
    end
end

function [Braid] = Rotate(Braid,MotorNum,Direction,Rotation)
    %First determine the position (row,col) of the center of a given
    %motor (MotorNum)within the 15x15 matrix "Braid"

    %Center of motor 1 is (2,2), center of motor 49 is (14,14)

    if mod(MotorNum,7)==0
        row=(floor(MotorNum/7))*2;
        col=(MotorNum-(row/2-1)*7)*2;
    else
        row=(floor(MotorNum/7)+1)*2;
        col=(MotorNum-(row/2-1)*7)*2;
    end

    %Define the 3x3 section of the "braid" matrix that is centered around
    %the given motor
    BraidSection = Braid(row-1:row+1,col-1:col+1);

    %Now rotate based on direction and rotation angle
    if Direction
        %Direction==1 means CW
        for i=1:Rotation/90
           BraidSection([4,6]) = BraidSection([6,4]);
           BraidSection = BraidSection';
           Braid(row-1:row+1,col-1:col+1) = BraidSection;
        end
    else
        %Direction==0 means CCW
        for i=1:Rotation/90
           BraidSection([2,8]) = BraidSection([8,2]);
           BraidSection = BraidSection';
           Braid(row-1:row+1,col-1:col+1) = BraidSection;
        end
    end
end
```

D.1.3 Plot Path Sub-function

```
%Acknowledgements: edited from code developed by Daniel Aldrich

%This function determines the x,y,z coordinates of the yarn 'n' throughout
%the length of the braid.
%StepStruct and Pattern are as defined in Machine_Emulation_3

%theta is the number of 'knots in spline' i.e. the number of points
%found between the start and end of motor rotation

%maxAngle is the value of 'Path Angle' in the braidGenerator app
function [x,y,z] = Plot_Path2(StepStruct,Pattern,n,res,theta,maxAngle)
patternLength = length(Pattern);
x(patternLength) = 0;
y(patternLength) = 0;
z(patternLength) = 0;

YarnNum=n; %Used to find the position of current yarn
%This loop finds the coordinates for the given yarn (YarnNum) in each level
%of Pattern
for i = 1:patternLength
    temp = Pattern{i};

    [x(i),y(i)] = find(temp==YarnNum);

    if i == 1
        z(1) = 0;
    elseif StepStruct(i-1).TakeUp.Enable==1
        z(i) = z(i-1);
    else
        z(i) = z(i-1) - 1*StepStruct(i-1).TakeUp.Speed/8;
        %TakeUp.Speed/8 normalizes the speed relative to 1 mm/min

    end
end


%Defines the center of each motor i in row,col coordinates
for i=1:49
    if mod(i,7)==0
        row=(floor(i/7))*2;
        col=(i-(row/2-1)*7)*2;
    else
        row=(floor(i/7)+1)*2;
        col=(i-(row/2-1)*7)*2;
    end
    motorx(i)=row;
    motory(i)=col;
end
```

```
%Theta is the number of knots in the spline
if theta > 2
    X = x(1);
    Y = y(1);
    Z = z(1);

    for n = 2:patternLength
        if ~(StepStruct(n-1).TakeUp.Enable)
            %Checks whether the yarn moved between Pattern{n} and Pattern{n-
1}
            if x(n)~=x(n-1) && y(n)~=y(n-1)

                %First find closest(Enabled) motor to yarn 'YarnNum' on
                %previous step

                %Find distance from any motor to the current yarn position
                dist = (motorx-x(n-1)).^2 + (motory-y(n-1)).^2;
                %Find which motor(s) are closest
                [~,ind] = find(dist==min(dist));
                for i=1:length(ind)
                    motorNum=ind(i);
                    %Check if each motor was enabled on previous step. Only
                    %possible for one of them to be enabled. Enabled=0
                    if StepStruct(n-1).motor(motorNum).Enable==0
                        index=motorNum;
                    end
                end
                Direction=StepStruct(n-1).motor(index).Direction;
                Rotation=deg2rad(StepStruct(n-1).motor(index).Rotation);

                %Find the position of the yarn relative to the center of the
                %motor as an angle. "Below" the motor corresponds to 0,
"right"
                %corresponds to pi/2, "left" corresponds to -pi/2, and "top"
                %corresponds to pi
                [t,~] = cart2pol(x(n-1)-motorx(index),y(n-1)-motory(index));

                %Creates evenly spaced points (both for angles and z
movement)
                %between the start and end points of the motor rotation. The
                %value of theta is changed if rotation is > 90 degrees
                theta=theta*Rotation/(pi/2);
                Ztheta = linspace(z(n-1),z(n),theta);
                angles = linspace(0,Rotation,theta);
                angles = angles(2:end-1);

                %Change values depending on the direction of rotation
                if Direction %1=CW
                    [Xtheta,Ytheta] = pol2cart(t-angles,1);
                else %0=CCW
                    [Xtheta,Ytheta] = pol2cart(t+angles,1);
                end
```

```
            %Finds the x and y coordinates that are allowable with the
            %selected max angle (maxAngle=path=path angle)

            temp = Xtheta((angles >= maxAngle) & (angles <=(Rotation-
maxAngle)));
            if length(temp)>1
                x1 = temp(1);
                x2 = temp(end);

                temp = Ytheta((angles>=maxAngle)&(angles<=(Rotation-
maxAngle)));

                y1 = temp(1);
                y2 = temp(end);
            end
                Xtheta = Xtheta + motorx(index);
                Ytheta = Ytheta + motory(index);

            if length(temp)>1
                %Find slope and intercept of line from start to end of
                %rotation
                m = (x2 - x1)/(y2 - y1);
                b = y2 - m*x2;

                %Anonymous function "r" used to define points of the
                %line described by the above slope and intercept
                r = @(t) (b)./(-cos(t)/m+sin(t));

                %finds the allowables angles and indices of those
                %angles
                ang = angles(angles>=maxAngle & angles<=(Rotation-
maxAngle));
                ind = find(angles>=maxAngle & angles<=(Rotation-
maxAngle));

                %Finds the cartesian coordinate equivalent, again
dependent
                %on direction
                if Direction %1=CW
                    [tempx, tempy] = pol2cart(t-ang,r(t-ang));
                else %0=CCW
                    [tempx, tempy] = pol2cart(t+ang,r(t+ang));
                end

                %Finds the difference between consecutive values in
                %tempx and tempy (i.e. dx(1)=tempx(2)-tempx(1))
                dx = diff(tempx);
                dy = diff(tempy);

                nlen = sqrt(dx.^2 + dy.^2)/norm([y2 - y1,x2 - x1]);
                zlen = Ztheta(ind(end)+1)-Ztheta(ind(1)+1);
                tempz = cumsum(zlen*nlen);

                Ztheta(ind+1) = [0,tempz]+Ztheta(ind(1)+1);
```

```
                    Xtheta(ind) = tempx + motorx(index);
                    Ytheta(ind) = tempy + motory(index);
                end
                X = [X,Xtheta,x(n)];
                Y = [Y,Ytheta,y(n)];
                Z = [Z,Ztheta(2:end)];
            else
                X = [X,ones(1,theta-1)*x(n)];
                Y = [Y,ones(1,theta-1)*y(n)];
                Z = [Z,-(1:theta-1)/(theta-1)*abs(z(n)-z(n-1))+z(n-1)];
            end
        else

        end
    end
else
    X = x;
    Y = y;
    Z = z;
end
z = Z;
if res>1
    Z = z(1):(z(2) - z(1))/res:z(end);
    x = spline(z,X,Z);
    y = spline(z,Y,Z);
else
   x = X(:);
   y = Y(:);
end
z = Z(:);
x = x(:);
y = y(:);
x = x';
y = y';
z = z(:)';
```

D.2 Microcontroller Code

```
/* This sketch is designed to communicate with the braid ganeration matlab app to control the 3D
braider
 * The setup function initializes the IO expanders and clock pin
 * The loop function alternates between receiving data from matlab (for the state of each motor's
enable and
 * direction pins) and controlling the motors based on the data recieved
 * The process is: (receive data for single step) -> (Perform single step) -> (receive more data) ->
etc.
 *
 * For more detail on initializing/use of IO expanders see tutorial at:
 * http://tronixstuff.com/2011/08/26/tutorial-maximising-your-arduinos-io-ports
 *
 * For more information on the receipt of data check the ArduinoPC sketch found:
 * https://forum.arduino.cc/index.php?topic=225329.msg1810764#msg1810764
 *
 * For more information on the use of time rather than delay() to pulse the clock lines
 * refer to the BlinkWithoutDelay sketch found:
 * https://www.arduino.cc/en/Tutorial/BlinkWithoutDelay
 */

#include "Wire.h" //Standard arduino library that enables communication with IO expanders

#define startMarker 254 //When the number 254 is sent from Matlab, arduino starts storing data
#define endMarker 255 //When the number 255 is sent from Matlab, arduino stops storing data
#define pauseMarker 253 //If the number 253 is sent from Matlab, arduino pauses turning the
motors
#define resumeMarker 252 //If the number 252 is sent from Matlab, arduino resumes turning the
motors
#define beginMarker 251 //When the number 251 is sent from Matlab, arduino begins braiding (i.e
takeup motors)
#define finishMarker 250 //When the number 250 is sent from Matlab, arduino stops braiding (i.e
takeup motors)
#define stepsFor90 7681 //The number of steps required to turn 90 degrees
//The number of steps for 90 equals the gear ratio (3591/187) * microsteps for 360 / 4
//(800/4 if dip switches set to 4 or 1/4 on motor driver) * 2 due to the variable microStep
//incrementing every half step (i.e. pin needs to write high/low before a microStep is completed)

const int clock1Pin = 52; //Pin number of clock pin for horizontal plane motors
const int clock2Pin = 23; //Pin number of clock pin for takeup (vertical) motors
const int clock1Interval = 1; //Interval at which to pulse clock for horizontal plane motors
(milliseconds). Constant
const int clock2IntervalReference = 225; //Reference interval for 1/8 mm/min (chosen to avoid
fraction) takeupSpeed
```

```
int StepCount = 0; //Used to count number of steps completed (step defined as a complete 90 degree turn of the motor)
int microStep = 0; //Used to count number of microsteps completed (microstep defined as a high/low pulse sent to the motor driver)
unsigned long fracStep = 0; //stepsFor90 is a rounded number. This number stores the decimal points to enable greater precision
                            //stored as unsigned long rather than float because float greatly decreases speed
boolean extraStep = 0; //When fracStep (i.e. decimal value of stepsFor90) has accumulated enough, this adds an extra step

int clock1State = LOW; //State of clock pin for horizontal plane motors
byte clock1Byte = B00000000; //Used to set multiple pins high/low simultaneously for clock 1
int clock2State = LOW; //State of clock pin for takeup (vertical) motors
byte takeupSpeed = 1; //Used for modifying the speed of the takeup (vertical) motors. Approximately mm/s
int clock2Interval = 225; //Interval at which to pulse clock for takeup (vertical) motors (milliseconds). Variable depending on takeupSpeed
unsigned long previousMillis1 = 0; // will store last time clock 1 was updated
unsigned long previousMillis2 = 0; // will store last time clock 2 was updated

byte bytesRecvd = 0; //Used to count # of numbers recieved including start and end numbers
//byte expectedBytes = 103; //Used for error checking to ensure enough numbers are received

byte tempBuffer[103]; //Array of all numbers received including start and end markers
boolean StepParams[100]; //Array of all enable/direction values for motors (start and end markers removed)

boolean inProgress = false; //Used to determine if important data is being received (turned on by startMarker)
boolean allReceived = false; //Used to determine if all important data has been received (turned on by endMarker)
boolean motorsReady = false; //Used to determine if activateMotors has been run with current step data from Matlab
boolean paused = false; //Allows for Matlab to temporarily pause motors turning
boolean takeup = false; //Takeup is true when the start braiding button is pushed in Matlab and false when Matlab disconnects


/////////////////////////////////////////////////////////////////////////

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115000); //Setup serial port wih 115000 baud rate

  //Setup all IO expanders to function as desired
  //General note: if notch on IO expander is up, then port A is on the right and port B is on the left
```

```
  Wire.begin(); //Starts up I2C bus on arduino

//  // I/O expander 0 (pin 15-17 to GND)
//  // **CURRENTLY NOT IN USE**
//  // I2C bus address is 0x20
//  // All I/O pins as outputs
//
//  Wire.beginTransmission(0x20); //Talking to I/O expander 0
//  Wire.write(0x00); // IODIRA register (i.e. talking to port A)
//  Wire.write(0x00); // set all port A to output
//  Wire.endTransmission();
//
//  Wire.beginTransmission(0x20); //Talking to I/O expander 0
//  Wire.write(0x01); // IODIRB register (i.e. talking to port B)
//  Wire.write(0x00); // set all port B to output
//  Wire.endTransmission();

  // I/O expander 1 (pin 15 to 5v, 16 & 17 to GND)
  // I2C bus address is 0x21
  // All I/O pins as outputs

  Wire.beginTransmission(0x21); //Talking to I/O expander 1
  Wire.write(0x00); // IODIRA register (i.e. talking to port A)
  Wire.write(0x00); // set all port A to output
  Wire.endTransmission();

  Wire.beginTransmission(0x21); //Talking to I/O expander 1
  Wire.write(0x01); // IODIRB register (i.e. talking to port B)
  Wire.write(0x00); // set all port B to output
  Wire.endTransmission();

  // I/O expander 2 (pin 15 & 17 to GND, 16 to 5v)
  // I2C bus address is 0x22
  // All I/O pins as outputs

  Wire.beginTransmission(0x22); //Talking to I/O expander 2
  Wire.write(0x00); // IODIRA register (i.e. talking to port A)
  Wire.write(0x00); // set all port A to output
  Wire.endTransmission();

  Wire.beginTransmission(0x22); //Talking to I/O expander 2
  Wire.write(0x01); // IODIRB register (i.e. talking to port B)
  Wire.write(0x00); // set all port B to output
  Wire.endTransmission();

  // I/O expander 3 (pin 15 & 16 to 5v, 17 to GND)
```

```
// I2C bus address is 0x23
// All I/O pins as outputs

Wire.beginTransmission(0x23); //Talking to I/O expander 3
Wire.write(0x00); // IODIRA register (i.e. talking to port A)
Wire.write(0x00); // set all port A to output
Wire.endTransmission();

Wire.beginTransmission(0x23); //Talking to I/O expander 3
Wire.write(0x01); // IODIRB register (i.e. talking to port B)
Wire.write(0x00); // set all port B to output
Wire.endTransmission();

// I/O expander 4 (pin 15 & 16 to GND, 17 to 5v)
// I2C bus address is 0x24
// All I/O pins as outputs

Wire.beginTransmission(0x24); //Talking to I/O expander 4
Wire.write(0x00); // IODIRA register (i.e. talking to port A)
Wire.write(0x00); // set all port A to output
Wire.endTransmission();

Wire.beginTransmission(0x24); //Talking to I/O expander 4
Wire.write(0x01); // IODIRB register (i.e. talking to port B)
Wire.write(0x00); // set all port B to output
Wire.endTransmission();

// I/O expander 5 (pin 16 to GND, 15 & 17 to 5v)
// I2C bus address is 0x25
// All I/O pins as outputs

Wire.beginTransmission(0x25); //Talking to I/O expander 5
Wire.write(0x00); // IODIRA register (i.e. talking to port A)
Wire.write(0x00); // set all port A to output
Wire.endTransmission();

Wire.beginTransmission(0x25); //Talking to I/O expander 5
Wire.write(0x01); // IODIRB register (i.e. talking to port B)
Wire.write(0x00); // set all port B to output
Wire.endTransmission();

// I/O expander 6 (pin 15 to GND, 16 & 17 to 5v)
// I2C bus address is 0x26
// All I/O pins as outputs

Wire.beginTransmission(0x26); //Talking to I/O expander 6
```

```
  Wire.write(0x00); // IODIRA register (i.e. talking to port A)
  Wire.write(0x00); // set all port A to output
  Wire.endTransmission();

  Wire.beginTransmission(0x26); //Talking to I/O expander 6
  Wire.write(0x01); // IODIRB register (i.e. talking to port B)
  Wire.write(0x00); // set all port B to output
  Wire.endTransmission();

  // I/O expander 7 (pin 15-17 to 5v)
  // I2C bus address is 0x27
  // All I/O pins as outputs

  Wire.beginTransmission(0x27); //Talking to I/O expander 7
  Wire.write(0x00); // IODIRA register (i.e. talking to port A)
  Wire.write(0x00); // set all port A to output
  Wire.endTransmission();

  Wire.beginTransmission(0x27); //Talking to I/O expander 7
  Wire.write(0x01); // IODIRB register (i.e. talking to port B)
  Wire.write(0x00); // set all port B to output
  Wire.endTransmission();
  //NOTE: port B currently not in use. Does not need to be output

  //pinMode(clock1Pin,OUTPUT); //Set clock 1 pin on arduino board to output
  //DDRC = B11111111; //Set PORT C (digital 37-30) to output (used for horizontal plane motor
clock lines)
  pinMode(clock2Pin,OUTPUT); //Set clock 2 pin on arduino board to output

  //Set pins for horizontal plane motors to output
  pinMode(30,OUTPUT);
  pinMode(31,OUTPUT);
  pinMode(32,OUTPUT);
  pinMode(33,OUTPUT);
  pinMode(34,OUTPUT);
  pinMode(35,OUTPUT);
  pinMode(36,OUTPUT);

  Serial.println("Arduino ready for motorControl"); //This will display in Matlab to confirm that
setup
                                                //is complete and the proper sketch is loaded


  Serial.print("Send Next Step Please"); //This line instructs Matlab to send the data for the first
step
}
```

```
//////////////////////////////////////////////////////////////////

void loop() {
  // put your main code here, to run repeatedly:

  getSerialData(); //Receive data from Matlab

  if ((takeup) && (!paused)) {
    verticalStep(); //Perform step of takeup (vertical) motors
  }

  if (allReceived) {
    activateMotors(); //Run activateMotors, which sets enable/direction pins
  }

  if ((motorsReady) && (!paused)) {
  motorStep(); //Perform step of horizontal plane motors per current StepParams
  }

}

void getSerialData() {
  if(Serial.available() > 0) { //Check to ensure serial data is being sent
    byte x = Serial.read(); //read serial data
    if (x == beginMarker) {//Determine if Matlab wants arduino to begin takeup
      takeup = true;
    }
    if (x == finishMarker) {//Determine if Matlab wants arduino to stop takeup
      takeup = false;
    }
    if (x == pauseMarker) {//Determine if Matlab wants arduino to pause
      paused = true;
//      Serial.println("Pause");
    }
    if (x == resumeMarker) {//Determine if Matlab wants arduino to resume
      paused = false;
//      Serial.println("Resume");
    }
    if (x == startMarker) { //data starts being stored if x = 254
      bytesRecvd = 0;
      inProgress = true;
    }
    if(inProgress) {
      tempBuffer[bytesRecvd] = x; //Stores current byte (number) in tempBuffer
      bytesRecvd ++;
```

```
    }
    if (x == endMarker) { //data stops being stored if x = 255
      inProgress = false;
      allReceived = true;

//      if (bytesRecvd != expectedBytes){
//        Serial.println("Missing some motor data")
//        Serial.println(bytesRecvd);
//      }


    }
  }
}

//////////////////////////////////////////////////////////////////

void motorStep() {
  //Pulses clock 1 line "stepsFor90" times to turn the motors 90 degrees (unless paused)

  unsigned long currentMillis = millis(); //stores the current time the program has been running

  if (microStep < stepsFor90 + extraStep) {
    //This section pulses the clock 1 line based on time
    //By not using delay, other functions can be performed while waiting
    if (currentMillis - previousMillis1 >= clock1Interval){
      //save the last time clock 1 was pulsed
      previousMillis1 = currentMillis;

//      Serial.println("Check clock 1 pulse");

      //Toggle clock 1 state
      if (clock1State == LOW) {
        clock1State = HIGH;
        clock1Byte = B11111111;
      } else {
        clock1State = LOW;
        clock1Byte = B00000000;
      }


      //Write data to clock 1 port
      //digitalWrite(clock1Pin,clock1State);
      //PORTC = clock1Byte;

      //Write horizontal plane motor clock pins to clock1State
```

```
      //***NOTE*** The port manipulation method above would be preferred as this method causes
a
      // very slight delay between the motors grouped on different pins. As of 2019-09-11, the port
      // manipulation method has not been able to function as it sets all pins to high/low rather
      // than just the single port
      digitalWrite(30,clock1State);
      digitalWrite(31,clock1State);
      digitalWrite(32,clock1State);
      digitalWrite(33,clock1State);
      digitalWrite(34,clock1State);
      digitalWrite(35,clock1State);
      digitalWrite(36,clock1State);

      microStep++;

//      Serial.println(microStep);
    }
  }
  else {

    fracStep = fracStep + 283422459; //Adds 9 digits after the decimal of stepsFor90 to increase
precision
    if (fracStep > 1000000000){
      fracStep = fracStep - 1000000000;
      extraStep = 1;
    } else {
      extraStep = 0;
    }

//    Serial.println(extraStep);

    motorsReady = false; //Arduino will not run motorStep again until activateMotors has been run
with new data
    microStep = 0; //Zero out microStep for next Step
    StepCount++;

    //Prints a line in Matlab saying 'Step x Completed' for progress tracking
    String stringOne = "Step ";
    String stringTwo = " Completed";
    String stringThree = stringOne + StepCount + stringTwo;
    Serial.println(stringThree);

    Serial.print("Send Next Step Please");//Instructs Matlab to send the data for the next step
  }

}
```

```
//////////////////////////////////////////////////////////////////////

void verticalStep() {
  //Pulses the clock 2 line based on the interval defined by takeupSpeed

  unsigned long currentMillis = millis(); //stores the current time the program has been running

  if (currentMillis - previousMillis2 >= clock2Interval){
      //save the last time clock 1 was pulsed
      previousMillis2 = currentMillis;

      //Toggle clock 2 state
      if (clock2State == LOW) {
        clock2State = HIGH;
      } else {
        clock2State = LOW;
      }

      //Write data to clock 2 pin
      digitalWrite(clock2Pin,clock2State);
  }

}


//////////////////////////////////////////////////////////////////////

void activateMotors() {

  // Remove start and end markers
  for (int i = 0; i < (bytesRecvd-2); i++) {
    StepParams[i]=tempBuffer[i+2];
  }

  //Determine speed for takeup (vertical) motors from data sent by Matlab
  takeupSpeed = tempBuffer[1];

  //Calculate interval at which to pulse clock 2 based on reference for 1 mm/s takeupSpeed
  if (StepParams[1]){
    clock2Interval = clock2IntervalReference / takeupSpeed ;
  } else {
    clock2Interval = clock2IntervalReference / (takeupSpeed*100);
    if (clock2Interval<1) {
      clock2Interval = 1;
    }
```

```
    //Serial.println(clock2Interval);
  }

//  int ParamLength = sizeof(StepParams)/sizeof(StepParams[0]);
//  if (ParamLength != 100) {
//    Serial.println("Issue with step parameters");
//    Serial.println(ParamLength);
//  }

//  for (int i = 0; i<BufferLength; i++) {
//   Serial.println(StepParams[i]);
//  }

//First decode StepParams into individual motor parameters.
//This step just allows for easier reading of the next step

  //Vertical motor parameters
  boolean EnableV = StepParams[0];
  boolean DirectionV = StepParams[1];

  //Motor 1 parameters
  boolean Enable1 = StepParams[2];
  boolean Direction1 = StepParams[3];

  //Motor 2 parameters
  boolean Enable2 = StepParams[4];
  boolean Direction2 = StepParams[5];

  //Motor 3 parameters
  boolean Enable3 = StepParams[6];
  boolean Direction3 = StepParams[7];

  //Motor 4 parameters
  boolean Enable4 = StepParams[8];
  boolean Direction4 = StepParams[9];

  //Motor 5 parameters
  boolean Enable5 = StepParams[10];
  boolean Direction5 = StepParams[11];

  //Motor 6 parameters
  boolean Enable6 = StepParams[12];
  boolean Direction6 = StepParams[13];

  //Motor 7 parameters
  boolean Enable7 = StepParams[14];
```

```
boolean Direction7 = StepParams[15];

//Motor 8 parameters
boolean Enable8 = StepParams[16];
boolean Direction8 = StepParams[17];

//Motor 9 parameters
boolean Enable9 = StepParams[18];
boolean Direction9 = StepParams[19];

//Motor 10 parameters
boolean Enable10 = StepParams[20];
boolean Direction10 = StepParams[21];

//Motor 11 parameters
boolean Enable11 = StepParams[22];
boolean Direction11 = StepParams[23];

//Motor 12 parameters
boolean Enable12 = StepParams[24];
boolean Direction12 = StepParams[25];

//Motor 13 parameters
boolean Enable13 = StepParams[26];
boolean Direction13 = StepParams[27];

//Motor 14 parameters
boolean Enable14 = StepParams[28];
boolean Direction14 = StepParams[29];

//Motor 15 parameters
boolean Enable15 = StepParams[30];
boolean Direction15 = StepParams[31];

//Motor 16 parameters
boolean Enable16 = StepParams[32];
boolean Direction16 = StepParams[33];

//Motor 17 parameters
boolean Enable17 = StepParams[34];
boolean Direction17 = StepParams[35];

//Motor 18 parameters
boolean Enable18 = StepParams[36];
boolean Direction18 = StepParams[37];
```

```
//Motor 19 parameters
boolean Enable19 = StepParams[38];
boolean Direction19 = StepParams[39];

//Motor 20 parameters
boolean Enable20 = StepParams[40];
boolean Direction20 = StepParams[41];

//Motor 21 parameters
boolean Enable21 = StepParams[42];
boolean Direction21 = StepParams[43];

//Motor 22 parameters
boolean Enable22 = StepParams[44];
boolean Direction22 = StepParams[45];

//Motor 23 parameters
boolean Enable23 = StepParams[46];
boolean Direction23 = StepParams[47];

//Motor 24 parameters
boolean Enable24 = StepParams[48];
boolean Direction24 = StepParams[49];

//Motor 25 parameters
boolean Enable25 = StepParams[50];
boolean Direction25 = StepParams[51];

//Motor 26 parameters
boolean Enable26 = StepParams[52];
boolean Direction26 = StepParams[53];

//Motor 27 parameters
boolean Enable27 = StepParams[54];
boolean Direction27 = StepParams[55];

//Motor 28 parameters
boolean Enable28 = StepParams[56];
boolean Direction28 = StepParams[57];

//Motor 29 parameters
boolean Enable29 = StepParams[58];
boolean Direction29 = StepParams[59];

//Motor 30 parameters
boolean Enable30 = StepParams[60];
```

```
boolean Direction30 = StepParams[61];

//Motor 31 parameters
boolean Enable31 = StepParams[62];
boolean Direction31 = StepParams[63];

//Motor 32 parameters
boolean Enable32 = StepParams[64];
boolean Direction32 = StepParams[65];

//Motor 33 parameters
boolean Enable33 = StepParams[66];
boolean Direction33 = StepParams[67];

//Motor 34 parameters
boolean Enable34 = StepParams[68];
boolean Direction34 = StepParams[69];

//Motor 35 parameters
boolean Enable35 = StepParams[70];
boolean Direction35 = StepParams[71];

//Motor 36 parameters
boolean Enable36 = StepParams[72];
boolean Direction36 = StepParams[73];

//Motor 37 parameters
boolean Enable37 = StepParams[74];
boolean Direction37 = StepParams[75];

//Motor 38 parameters
boolean Enable38 = StepParams[76];
boolean Direction38 = StepParams[77];

//Motor 39 parameters
boolean Enable39 = StepParams[78];
boolean Direction39 = StepParams[79];

//Motor 40 parameters
boolean Enable40 = StepParams[80];
boolean Direction40 = StepParams[81];

//Motor 41 parameters
boolean Enable41 = StepParams[82];
boolean Direction41 = StepParams[83];
```

```
  //Motor 42 parameters
  boolean Enable42 = StepParams[84];
  boolean Direction42 = StepParams[85];

  //Motor 43 parameters
  boolean Enable43 = StepParams[86];
  boolean Direction43 = StepParams[87];

  //Motor 44 parameters
  boolean Enable44 = StepParams[88];
  boolean Direction44 = StepParams[89];

  //Motor 45 parameters
  boolean Enable45 = StepParams[90];
  boolean Direction45 = StepParams[91];

  //Motor 46 parameters
  boolean Enable46 = StepParams[91];
  boolean Direction46 = StepParams[92];

  //Motor 47 parameters
  boolean Enable47 = StepParams[93];
  boolean Direction47 = StepParams[94];

  //Motor 48 parameters
  boolean Enable48 = StepParams[96];
  boolean Direction48 = StepParams[97];

  //Motor 49 parameters
  boolean Enable49 = StepParams[98];
  boolean Direction49 = StepParams[99];

//Next compute the bytes to sent to the IO expanders
//These bytes are formed assuming the lowest number motor is connected to the lowest number pin
//Note that pin numbers run counter clockwise from the top left of the IO expander

  //Byte for IO Expander 1, port A (contains motors V, 1-3)
  byte IOExp1AByte = 1*EnableV+2*DirectionV+4*Enable1+8*Direction1+16*Enable2+32*Direction2+64*Enable3+128*Direction3;

  //Byte for IO Expander 1, port B (contains motors 4-7)
  byte IOExp1BByte = 1*Enable4+2*Direction4+4*Enable5+8*Direction5+16*Enable6+32*Direction6+64*Enable7+128*Direction7;
```

```
  //Byte for IO Expander 2, port A (contains motors 8-11)
  byte                                        IOExp2AByte                                        =
1*Enable8+2*Direction8+4*Enable9+8*Direction9+16*Enable10+32*Direction10+64*Enable1
1+128*Direction11;

  //Byte for IO Expander 2, port B (contains motors 12-15)
  byte                                        IOExp2BByte                                        =
1*Enable12+2*Direction12+4*Enable13+8*Direction13+16*Enable14+32*Direction14+64*En
able15+128*Direction15;

  //Byte for IO Expander 3, port A (contains motors 16-19)
  byte                                        IOExp3AByte                                        =
1*Enable16+2*Direction16+4*Enable17+8*Direction17+16*Enable18+32*Direction18+64*En
able19+128*Direction19;

  //Byte for IO Expander 3, port B (contains motors 20-23)
  byte                                        IOExp3BByte                                        =
1*Enable20+2*Direction20+4*Enable21+8*Direction21+16*Enable22+32*Direction22+64*En
able23+128*Direction23;

  //Byte for IO Expander 4, port A (contains motors 24-27)
  byte                                        IOExp4AByte                                        =
1*Enable24+2*Direction24+4*Enable25+8*Direction25+16*Enable26+32*Direction26+64*En
able27+128*Direction27;

  //Byte for IO Expander 4, port B (contains motors 28-31)
  byte                                        IOExp4BByte                                        =
1*Enable28+2*Direction28+4*Enable29+8*Direction29+16*Enable30+32*Direction30+64*En
able31+128*Direction31;

  //Byte for IO Expander 5, port A (contains motors 32-35)
  byte                                        IOExp5AByte                                        =
1*Enable32+2*Direction32+4*Enable33+8*Direction33+16*Enable34+32*Direction34+64*En
able35+128*Direction35;

  //Byte for IO Expander 5, port B (contains motors 36-39)
  byte                                        IOExp5BByte                                        =
1*Enable36+2*Direction36+4*Enable37+8*Direction37+16*Enable38+32*Direction38+64*En
able39+128*Direction39;

  //Byte for IO Expander 6, port A (contains motors 40-43)
  byte                                        IOExp6AByte                                        =
128*Enable40+2*Direction40+4*Enable41+8*Direction41+16*Enable42+32*Direction42+64*
Enable43+128*Direction43;
```

```
  //Byte for IO Expander 6, port B (contains motors 44-47)
  byte IOExp6BByte = 1*Enable44+2*Direction44+4*Enable45+8*Direction45+16*Enable46+32*Direction46+64*Enable47+128*Direction47;

  //Byte for IO Expander 7, port A (contains motors 48-49, w/4 blank pins)
  byte IOExp7AByte = 1*Enable48+2*Direction48+4*Enable49+8*Direction49;//+16*Enable42+32*Direction42+64*Enable43+128*Direction43;

//Finally, send the bytes where they need to go

  // IO Expander 1, port A
  Wire.beginTransmission(0x21);
  Wire.write(0x12);  //GPIO A
  Wire.write(IOExp1AByte);
  Wire.endTransmission();

  // IO Expander 1, port B
  Wire.beginTransmission(0x21);
  Wire.write(0x13);  //GPIO B
  Wire.write(IOExp1BByte);
  Wire.endTransmission();

  // IO Expander 2, port A
  Wire.beginTransmission(0x22);
  Wire.write(0x12);  //GPIO A
  Wire.write(IOExp2AByte);
  Wire.endTransmission();

  // IO Expander 2, port B
  Wire.beginTransmission(0x22);
  Wire.write(0x13);  //GPIO B
  Wire.write(IOExp2BByte);
  Wire.endTransmission();

  // IO Expander 3, port A
  Wire.beginTransmission(0x23);
  Wire.write(0x12);  //GPIO A
  Wire.write(IOExp3AByte);
  Wire.endTransmission();

  // IO Expander 3, port B
  Wire.beginTransmission(0x23);
  Wire.write(0x13);  //GPIO B
  Wire.write(IOExp3BByte);
```

```
Wire.endTransmission();

// IO Expander 4, port A
Wire.beginTransmission(0x24);
Wire.write(0x12);  //GPIO A
Wire.write(IOExp4AByte);
Wire.endTransmission();

// IO Expander 4, port B
Wire.beginTransmission(0x24);
Wire.write(0x13);  //GPIO B
Wire.write(IOExp4BByte);
Wire.endTransmission();

// IO Expander 5, port A
Wire.beginTransmission(0x25);
Wire.write(0x12);  //GPIO A
Wire.write(IOExp5AByte);
Wire.endTransmission();

// IO Expander 5, port B
Wire.beginTransmission(0x25);
Wire.write(0x13);  //GPIO B
Wire.write(IOExp5BByte);
Wire.endTransmission();

// IO Expander 6, port A
Wire.beginTransmission(0x26);
Wire.write(0x12);  //GPIO A
Wire.write(IOExp6AByte);
Wire.endTransmission();

// IO Expander 6, port B
Wire.beginTransmission(0x26);
Wire.write(0x13);  //GPIO B
Wire.write(IOExp6BByte);
Wire.endTransmission();

// IO Expander 7, port A
Wire.beginTransmission(0x27);
Wire.write(0x12);  //GPIO A
Wire.write(IOExp7AByte);
Wire.endTransmission();

allReceived = false; //Arduino does not run activateMotors again until new data is received
motorsReady = true; //Arduino will run motorStep while motorsReady is true
```

```
}

////////////////////////////////////////////////////////////////////
```

D.3 Micro-CT Analysis Code

D.3.1 Braid Recreation Script

```
% =========================================================================
% Code written by : Ali Gholami                                            |
%                   PhD student                                            |
%                   Department of Mechanical Engineering                   |
%                   Lassonde School of Engineering, York University        |
%                   Canada                                                 |
%                   E-mail : gholami.ali90@gmail.com
%--------------------------------------------------------------------------
%|
% This will load the stack of images and will detect the boundaries at each
% image. The report text file will include center, area, major and minor
% diameter
%
%==========================================================================
%% This code is to Analyze images including just single yarn of braid
% This code reads all the images in the given address and analyzes them
% one by one.

%% Defining Address of Images
clc, clear all, close all

D = 'E:\Gholami-20200210-3D tracing yarn-second test\3d braid_Rec'; %address
of the Images
S = dir(fullfile(D,'*.bmp')); % pattern to match filenames.
Number_Images = numel(S);
resolution = 4.999E-3; %Resolution of the uCT images

for n=1:20-1
    n*100/Number_Images
    F = fullfile(D,S(n).name); %Loading image n
    images = imread(F); % Reading Image n
    I = images; % Showing Image n
    I = im2bw(I,0.4); %Binarizing Image n
    I = bwareaopen(I, 50); %Filtering unwanted boundaries. Boundaries with
areas below 50 will be ignored

%     imshow(I)
%     hold on
    I = imclose(I > 0, true(3));
    [B,L] = bwboundaries(I,'noholes'); %Getting boudaries in the Image n
    stats =
regionprops(L,'Area','Centroid','orientation','MajorAxisLength','MinorAxisLen
gth'); %Getting properties of the detected boundary

    Area(n,1) = stats.Area*resolution^2;
    Centre(n,1) = stats.Centroid(1)*resolution;
    Centre(n,2) = stats.Centroid(2);
    Diameter(n,1) = stats.MajorAxisLength*resolution;
    Diameter(n,2) = stats.MinorAxisLength*resolution;
```

```
%     boundary = B{1};
%     plot(boundary(:,2),boundary(:,1),'blue','LineWidth',2)
%
% %     display the results
%     metric_stringA = sprintf('%2.2f',Area(n,1));
%     metric_stringD = sprintf('%2.3f',Diameter(n,1));
%
%     text(boundary(1,2)-35,boundary(1,1)+13,metric_stringA,'Color','red',...
%            'FontSize',11,'FontWeight','bold')
%
%     plot(Centre(n,1),Centre(n,2),'green*')


end
h = [Centre,(1:n)']*resolution;
Prop = [h, Area, Diameter];
fileID = fopen('Properties of Tracing 3D braided Yarn_uct.txt','w');
% fprintf(fileID,'%6s %12s %18s\n','X','Y','Z');
fprintf(fileID,'%6.8f %12.8f %12.8f %12.8f %12.8f %12.8f\n',Prop');
fclose(fileID);

figure(2)
plot3(h(1:1782,1),h(1:1782,2),h(1:1782,3),'r') %Plotting the tracing yarn.
One unit cell is plotted
```

D.3.2 Angle and Pitch Analysis Script

```
clc; clear; close all

YarnFileID=fopen('Yarn1.txt');
C=textscan(YarnFileID,'%f, %f, %f');
fclose(YarnFileID);
xYarn=C{1};
zYarn=C{2};
yYarn=C{3};

MCT_FileID=fopen('XYZ_3D braid uct.txt');
C2=textscan(MCT_FileID,'%f %f %f');
fclose(MCT_FileID);
xMCT=C2{1};
yMCT=C2{2};
zMCT=C2{3};

%Process data to remove deadzone (20 values if Deadzone=2)
xYarn(1:20)=[];
yYarn(1:20)=[];
zYarn(1:20)=[];

xYarn(end-19:end)=[];
yYarn(end-19:end)=[];
```

```
zYarn(end-19:end)=[];

%Process z data
zYarn=(zYarn*-1)-200;


%Start data for both at 0,0
xYarn=xYarn-xYarn(1,1);
yYarn=yYarn-yYarn(1,1);

xMCT=xMCT-xMCT(1,1);
yMCT=yMCT-yMCT(1,1);

%Rotate micro-CT data to align with model
theta=255;
tempx=xMCT;
tempy=yMCT;
xMCT=tempx.*cosd(theta)-tempy.*sind(theta);
yMCT=tempy.*cosd(theta)+tempx.*sind(theta);


XYZ_Store=zeros(length(zYarn),4);
count=1;
XYZ_Store(1,1)=xYarn(1,1);
XYZ_Store(1,2)=yYarn(1,1);
XYZ_Store(1,3)=zYarn(1,1);

XYZ_Store(1,4)=xMCT(1,1);
XYZ_Store(1,5)=yMCT(1,1);
XYZ_Store(1,6)=zMCT(1,1);

for i=2:length(zYarn)
    for j=2:length(zMCT)
        if abs(zYarn(i,1)-zMCT(j,1))<0.5
            count=count+1;
            if (zMCT(j,1)-XYZ_Store(count-1,6))>=0.5
                index=count;
                XYZ_Store(index,1)=xYarn(i,1);
                XYZ_Store(index,2)=yYarn(i,1);
                XYZ_Store(index,3)=zYarn(i,1);

                XYZ_Store(index,4)=xMCT(j,1);
                XYZ_Store(index,5)=yMCT(j,1);
                XYZ_Store(index,6)=zMCT(j,1);
            else
                count=count-1;
            end
        end
    end
end

XYZ_Store=XYZ_Store/100;

for i=2:(size(XYZ_Store,1)-1)
```

```
    XY_length1=((XYZ_Store(i+1,1)-XYZ_Store(i,1))^2 ...
        +(XYZ_Store(i+1,2)-XYZ_Store(i,2))^2)^(1/2);
    XY_length2=((XYZ_Store(i,1)-XYZ_Store(i-1,1))^2 ...
        +(XYZ_Store(i,2)-XYZ_Store(i-1,2))^2)^(1/2);
    ForwardAngle=atand(XY_length1/(XYZ_Store(i+1,3)-XYZ_Store(i,3)));
    BackwardAngle=atand(XY_length2/(XYZ_Store(i,3)-XYZ_Store(i-1,3)));
    AveAngle=(ForwardAngle+BackwardAngle)/2;
    AngleYarn(i,1:2)=[XYZ_Store(i,3), AveAngle];
end

for i=2:(size(XYZ_Store,1)-1)
    XY_length1=((XYZ_Store(i+1,4)-XYZ_Store(i,4))^2 ...
        +(XYZ_Store(i+1,5)-XYZ_Store(i,5))^2)^(1/2);
    XY_length2=((XYZ_Store(i,4)-XYZ_Store(i-1,4))^2 ...
        +(XYZ_Store(i,5)-XYZ_Store(i-1,5))^2)^(1/2);
    ForwardAngle=atand(XY_length1/(XYZ_Store(i+1,6)-XYZ_Store(i,6)));
    BackwardAngle=atand(XY_length2/(XYZ_Store(i,6)-XYZ_Store(i-1,6)));
    AveAngle=(ForwardAngle+BackwardAngle)/2;
    AngleMCT(i,1:2)=[XYZ_Store(i,6), AveAngle];
end


%YARN ANGLE ANALYSIS -----------------------------------------------------

%This loop finds the constant-angle sections for the model by finding
%points where another point 5 z-steps ahead or behind is still within 0.5
%degrees
count=0;
for i=6:size(AngleYarn,1)-5
    if (abs(AngleYarn(i,2)-AngleYarn(i+5,2))<3) ...
            || (abs(AngleYarn(i-5,2)-AngleYarn(i,2))<3)
        count=count+1;
        ConstAngleYarn(count,1:2)=[AngleYarn(i,1) AngleYarn(i,2)];
    end
end

%This loop averages the individual constant-angle sections to approximate
%the braid angle
lastRef=1;
count=0;
for i=1:size(ConstAngleYarn,1)-1
    if abs(ConstAngleYarn(i+1,2)-ConstAngleYarn(i,2))>3
        count=count+1;
        AveAngleYarn(count)=mean(ConstAngleYarn(lastRef:i,2));
        AveZYarn(count)=mean(ConstAngleYarn(lastRef:i,1));
        lastRef=i+1;
    end
end
count=count+1;
AveAngleYarn(count)=mean(ConstAngleYarn(lastRef:end,2))
AveZYarn(count)=mean(ConstAngleYarn(lastRef:end,1));

%This loop finds the constant-angle sections for the micro-CT data by
%finding points where another point 5 z-steps ahead or behind is still
%within 0.5degrees
count=0;
```

```
for i=6:size(AngleMCT,1)-5
    if (abs(AngleMCT(i,2)-AngleMCT(i+5,2))<1) ...
            || (abs(AngleMCT(i-5,2)-AngleMCT(i,2))<1)
        count=count+1;
        ConstAngleMCT(count,1:2)=[AngleMCT(i,1) AngleMCT(i,2)];
    end
end

%This loop averages the individual constant-angle sections to approximate
%the braid angle
lastRef=1;
count=0;
for i=1:size(ConstAngleMCT,1)-1
    if abs(ConstAngleMCT(i+1,1)-ConstAngleMCT(i,1))>0.3
        count=count+1;
        AveAngleMCT(count)=mean(ConstAngleMCT(lastRef:i,2));
        AveZMCT(count)=mean(ConstAngleMCT(lastRef:i,1));
        lastRef=i+1;
    end
end
count=count+1;
AveAngleMCT(count)=mean(ConstAngleMCT(lastRef:end,2))
AveZMCT(count)=mean(ConstAngleMCT(lastRef:end,1));


%PITCH LENGTH ANALYSIS---------------------------------------------------

%Pitch for Yarn data determined by average z of constant angle 1 and
%constant angle 6
fullPathLength_Yarn=AveZYarn(8)-AveZYarn(4);
YarnPitch=fullPathLength_Yarn/4
YarnPitchPlot=[AveZYarn(4), AveAngleYarn(4)...
    ;AveZYarn(8), AveAngleYarn(8)];

%Pitch for MCT data determined by average z of constant angle 1 and
%constant angle 6
fullPathLength_MCT=AveZMCT(5)-AveZMCT(2);
MCTPitch=fullPathLength_MCT/4
MCTPitchPlot=[AveZMCT(2), (AveAngleMCT(2)+AveAngleMCT(5))/2 ...
    ;AveZMCT(5), (AveAngleMCT(2)+AveAngleMCT(5))/2];

%Plotting ----------------------------------------------------------------
figure(1)
hold on
scatter3(XYZ_Store(:,1),XYZ_Store(:,2),XYZ_Store(:,3))
scatter3(XYZ_Store(:,4),XYZ_Store(:,5),XYZ_Store(:,6))
ax = gca;
ax.FontSize=28;
hold off

figure(2)
plot(AngleYarn(2:end-2,1),AngleYarn(2:end-2,2),'k-.', 'LineWidth', 5)
xlabel('Distance along Z-axis (mm)')
ylabel('Average Angle relative to Z-axis (deg)')
ax = gca;
```

```
ax.FontSize=28;

figure(3)
hold on
plot(AngleYarn(2:end-2,1),AngleYarn(2:end-2,2),'k-.', 'LineWidth', 5)
plot(AngleMCT(2:end-2,1),AngleMCT(2:end-2,2),'k-', 'LineWidth', 5)
xlabel('Distance along Z-axis (mm)')
ylabel('Average Angle relative to Z-axis (deg)')
legend({'Model','Micro-CT Data'},'Location','northeast')
ax = gca;
ax.FontSize=28;
hold off

figure(4)
hold on
plot(AngleYarn(2:end-2,1),AngleYarn(2:end-2,2),'k-.', 'LineWidth', 5)
plot(ConstAngleYarn(:,1),ConstAngleYarn(:,2),'ro', ...
    'MarkerSize', 10, 'MarkerFaceColor', 'r')
plot(YarnPitchPlot(:,1),YarnPitchPlot(:,2),'r-X', 'LineWidth', 3, ...
    'MarkerSize', 10)

xlabel('Distance along Z-axis (mm)')
ylabel('Average Angle relative to Z-axis (deg)')
legend({'Model','Model Const. Angle',...
    'Model Pitch'},'Location','northeast')
ax = gca;
ax.FontSize=28;
hold off

figure(5)
hold on
plot(AngleMCT(2:end-2,1),AngleMCT(2:end-2,2),'k-', 'LineWidth', 5)
plot(ConstAngleMCT(:,1),ConstAngleMCT(:,2),'rs', ...
    'MarkerSize', 10, 'MarkerFaceColor', 'r')
plot(MCTPitchPlot(:,1),MCTPitchPlot(:,2),'r-X', 'LineWidth', 3, ...
    'MarkerSize', 10)

xlabel('Distance along Z-axis (mm)')
ylabel('Average Angle relative to Z-axis (deg)')
legend({'Micro-CT Data','M-CT Const. Angle',...
     'M-CT Pitch'},'Location','northeast')
ax = gca;
ax.FontSize=28;
hold off
```

Appendix E Sub-unit Cell Distribution Diagrams

E.1 Orthogonal Shapes

E.1.1 Square (6x6)

C	E	E	E	E	C
E	M	M	M	M	E
E	M	M	M	M	E
E	M	M	M	M	E
E	M	M	M	M	E
C	E	E	E	E	C

E.1.2 I-Shape (6x6x2)

C	E	E	E	E	C
C	E	M	M	E	C
		E	E		
		E	E		
C	E	M	M	E	C
C	E	E	E	E	C

E.1.3 C-shape (6x6x2)

C	E	E	E	E	C
E	M	E	E	E	C
E	E				
E	E				
E	M	E	E	E	C
C	E	E	E	E	C

E.1.4 T-Shape (6x6x2)

C	E	E	E	E	C
C	E	M	M	E	C
		E	E		
		E	E		
		E	E		
		C	C		

E.1.5 H-Shape (6x6x2)

C	C			C	C
E	E			E	E
E	M	E	E	M	E
E	M	E	E	M	E
E	E			E	E
C	C			C	C

E.1.6 L-Shape (6x6x2)

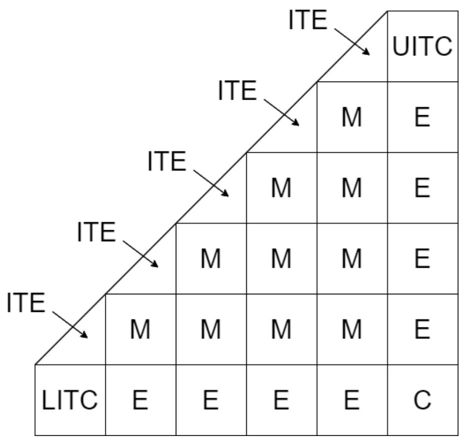
C	C				
E	E				
E	E				
E	E				
E	M	E	E	E	C
C	E	E	E	E	C

E.1.7 Box-Shape (6x6x2)

C	E	E	E	E	C
E	M	E	E	M	E
E	E			E	E
E	E			E	E
E	M	E	E	M	E
C	E	E	E	E	C

E.2 Non-orthogonal Shapes

E.2.1 Isosceles Triangle (6x6)



E.2.2 Right Triangle (4x7x2)