INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

UNIVERSITY OF ALBERTA

Ultraviolet Laser Pulse Amplification in Ce:LLF

by

MICHAEL DAVID BUHR ©

A Thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment
of the requirements for the degree of Master of Science

Department of Electrical and Computer Engineering
Edmonton, Alberta
Fall 2001

0-612-69496-8

University of Alberta

Library Release Form

Name of Author: Michael David Buhr

Title of Thesis: Ultraviolet Laser Pulse Amplification in Ce:LLF

Degree: Master of Science

Year this Degree Granted: 2001

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Michael D. Buhr

716 – Knottwod Road South NW
Edmonton, Alberta, Canada
T6K 1W5

8 August 2001

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled 'Ultraviolet Laser Pulse Amplification in Ce:LLF' submitted by Michael David Buhr in partial fulfillment of the requirements for the degree of Master of Science

Robert Fedosejevs
Dr. R. Fedosejevs

Dr. H. J. J. Seguin

Dr. L. Li

July 20, 2001
Date

Abstract

A number of properties of the laser crystal cerium doped lithium lutetium fluoride (Ce:LLF) were measured. The spontaneous lifetime was found to be 37.6 ± 1.0 ns. The gain cross-section at 308 nm (pi polarization) was determined to be $(3.1 \pm 1.2) \times 10^{-18}$ cm^2. The absorption cross-sections at 248 nm for sigma and pi polarizations were found to be $(7.3 \pm 3.2) \times 10^{-18}$ cm^2 and $(1.3 \pm 0.7) \times 10^{-17}$ cm^2. A sigma polarized transverse pump flux of 1.3 J/cm^2 at 248 nm gave a single pass gain coefficient of 4.7 cm^{-1} at 308 nm (pi polarization) in a 7.5 mm long crystal. A 3 stage amplifier system was simulated, indicating that a 60 fs 20 μJ seed pulse could be amplified to an output energy of 798 mJ, resulting in a peak power of 13.3 TW using 5.1 J of KrF pump energy.

Table of Contents

List of Tables

List of Figures

Chapter1: Introduction

1.1 Motivation

High-intensity ultrashort laser pulses have applications in a number of fields. Specific examples include micromachining, surgery, laser-plasma interaction and x-ray generation. Tunable lasers are useful in applications such as Lidar[1] of gas species in the atmosphere. The generation of such ultrashort and tunable laser pulses both require a gain medium with a broad fluorescence spectrum. Titanium doped sapphire (Ti^{3+}:Al_2O_3) [2] has a bandwidth of 100 THz [3] and is a common laser medium for generating ultrashort laser pulses in the infrared portion of the spectrum. However, quite often, shorter wavelength pulses are desirable, particularly in the ultraviolet part of the spectrum. Nonlinear optical crystals can be used to convert visible or infrared laser pulses into the ultraviolet spectrum. These nonlinear crystals require critical alignment and are subject to damage at high pulse energies, and thus the generation of high energy ultrashort ultraviolet laser pulses would require a gain medium with broad fluorescence in the ultraviolet spectrum.

The broad fluorescence spectrum typical of the 5d-4f transition in the Ce^{3+} ion makes it useful as another crystal dopant for ultrashort and tunable ultraviolet lasers. Common host crystals for Ce^{3+} include the fluorides $LiCaAlF_6$[4], $LiSrAlF_6$[5], $YLiF_4$[6] and $LiLuF_4$[7] [8]. The bandwidth of Ce^{3+}:$LiLuF_4$ is ~98 THz, based on the full fluorescence spectra determined by Sarukura and Dubinskii[7]. If only the emission band centred at 325 nm is considered, the effective bandwidth would be 50 THz (based on an average of sigma and pi polarizations). A schematic diagram of the energy levels for the Ce^{3+} ion[9] is shown in Figure 1.1-1, and a summary of the Ce:LLF fluorescence lines is given in

Table 1.1-1. The basic structure of the energy levels (5d-4f) is common to all of the Ce^{3+} doped fluoride crystals, with different hosts yielding different wavelengths for the absorption and emission bands. The approximate wavelengths of the absorption and fluorescence peaks in Figure 1.1-1 are shown for Ce^{3+}:$LiLuF_4$. The 243 nm absorption band of Ce^{3+}:$LiLuF_4$ has been used for pumping with a KrF excimer laser[7], and the 295 nm absorption band has been used for pumping with a Ce^{3+}:$LiSrAlF_6$ laser[8]. Other possible pumping schemes include pumping the 295 nm absorption band with the second harmonic of a Rhodamine 590 dye laser, pumping the 193 nm absorption band with an ArF laser and pumping the 212 nm absorption band with the fifth harmonic[16] of a Nd:YAG laser. The combination of pumping at 248 nm along with lasing at either 312 nm or 325 nm is expected to be close to an ideal four level laser system. Detailed energy level calculations for Ce:LLF have been performed by various groups[10,11,12]. Drawbacks of the Ce^{3+} doped fluoride crystals include solarization and excited state absorption.

Computer simulations are an effective tool for assisting in the design of and predicting the performance of laser systems. Design parameters can be easily varied in simulation, saving time and money that would otherwise be spent on the preparation of laser crystals and carrying out many parametric studies.

This thesis presents measurements of parameters for a heavily-doped Ce^{3+}:$LiLuF_4$ crystal along with numerical simulations of a design for an ultrashort pulse laser amplifier system, based on KrF laser pumping.

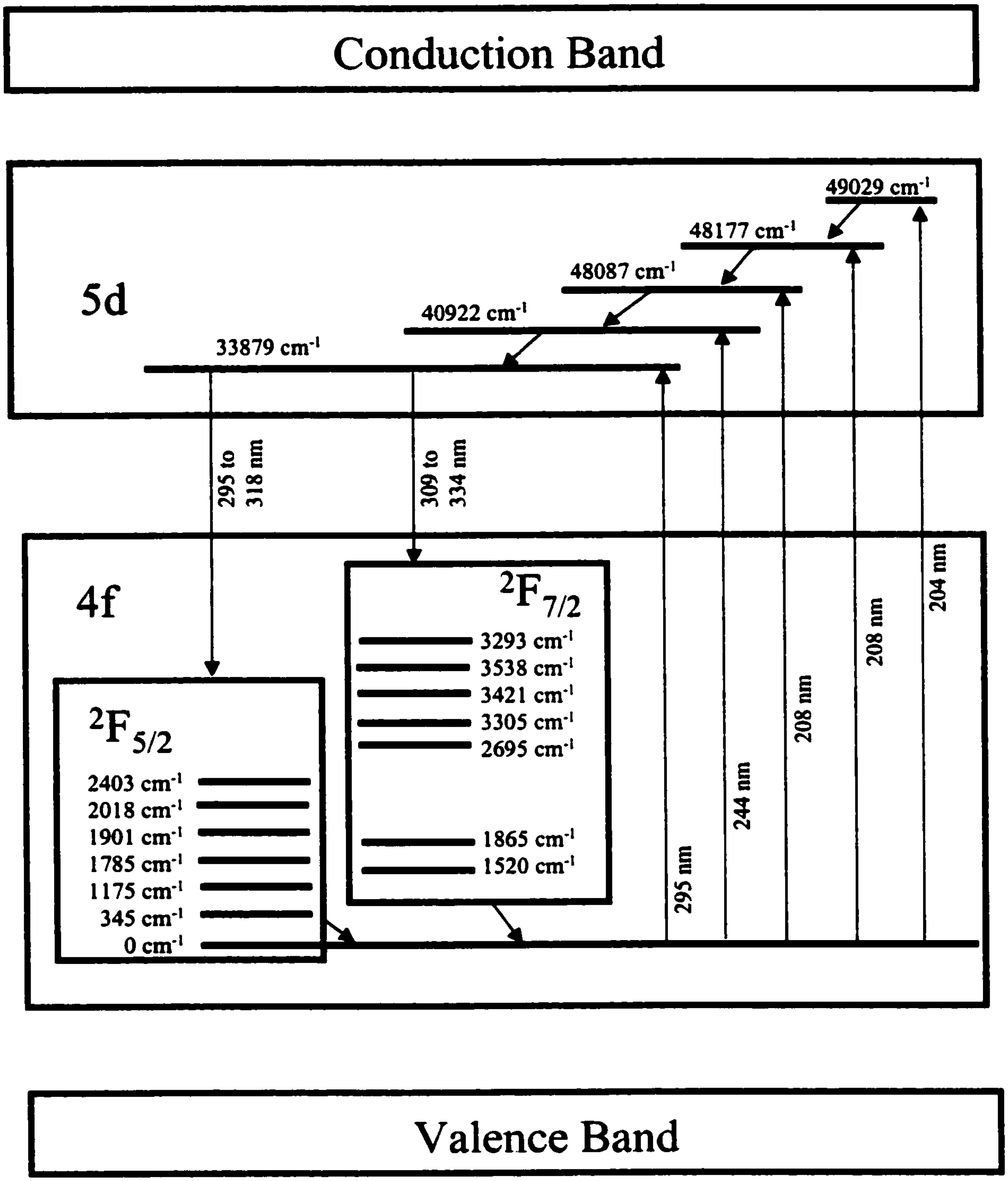

Figure 1.1-1: Energy levels of Ce^{3+}in $LiLuF_4$ host.

		Emission from lowest 5d Level 33879 cm^{-1}		
4f State	Energy Level [cm^{-1}]	Wave Number [cm^{-1}]	Wavelength [nm]	Energy Difference [eV]
2F (5/2)	0	33879	295.2	4.21
2F (5/2)	345	33534	298.2	4.17
2F (5/2)	1175	32704	305.8	4.07
2F (5/2)	1785	32094	311.6	3.99
2F (5/2)	1901	31978	312.7	3.98
2F (5/2)	2018	31861	313.9	3.96
2F (5/2)	2403	31476	317.7	3.91
2F (7/2)	1520	32359	309.0	4.02
2F (7/2)	1865	32014	312.4	3.98
2F (7/2)	2695	31184	320.7	3.88
2F (7/2)	3305	30574	327.1	3.80
2F (7/2)	3421	30458	328.3	3.79
2F (7/2)	3538	30341	329.6	3.77
2F (7/2)	3923	29956	333.8	3.72

Table 1.1-1: Emissions in Ce:LLF

1.2 Previous Experimental work with Ce:LLF

Ce:LLF has been investigated for its laser characteristics by various different groups. Sarukura and Dubinskii[7] measured the absorption and fluorescence spectra of Ce:LLF. Single pass gains of up to 4.3 were observed for transverse KrF pumping of a 1 cm long crystal at a pump flux of 600 mJ/cm^2 and a pump energy of 120 mJ. The gain saturation fluence at 325 nm was determined to be ~50 mJ/cm^2, according to data fit to the Franz-Nodvik equations. The corresponding gain cross-section was found to be ~1×10^{-17} cm^2. The radiative lifetime of the 5d state was found to be 40 ± 3 ns[13].

A simple laser cavity was used to investigate the lasing characteristics of Ce:LLF and Ce:YLF [7,13]. The cavity consisted of two dielectric multilayer mirrors separated by 30 cm, with the laser crystal being placed 5 cm from the output coupler. The rear reflector had a radius of curvature = 0.5 m and the output coupler was planar, and had a transmission of roughly 1%. The mirrors were optimized for lasing at the short wavelength peak (311 nm for Ce:LLF and 309 nm for Ce:YLF). KrF pump pulses with 8 ns duration and energies up to 30 mJ were focused onto the side window of the laser crystal. Laser output energy of ~2.5 mJ was observed for a KrF pump flux of 187 mJ/cm^2 and a pump energy of 24 mJ. Under similar pumping conditions, the Ce:YLF laser had a peak output of ~1.4 mJ. The lasing pump threshold of Ce:LLF in this experiment was found to be 41 mJ/cm^2, while Ce:YLF had a lasing threshold of 70 mJ/cm^2.

A confocal 4-pass Ce:LLF amplifier[7] was subsequently developed with transverse KrF pumping. The gain was 10^4 for ps duration pulses with an input energy of ~1 μJ and a wavelength of 325 nm. The laser crystal had a length of 1 cm and a diameter of 5 mm.

Pump energies of 100 mJ (at a pump flux of 500 mJ/cm^2) at a repetition rate of 0.5 Hz were used.

Sarukura and Dubinksii also developed a passive self-injection seeded pulse train laser[7]. This involves a pulse seeding cavity within a larger feedback laser cavity, and has no requirement for timing control or Pockels cells. Pulses are generated in the seeding cavity and then injected into the feedback laser cavity, after which they are amplified regneratively until the gain is quenched. The feedback laser cavity consisted of a flat 100% reflector and a flat output coupler (with transmission of 30%) separated by ~1.1 m. The pulse seeding cavity was formed by the output coupler, and reflection from the end face of the laser crystal, (with a total distance between reflective surfaces of 3 cm). A confocal pair of lenses with 10 cm focal length are used to improve the mode matching between the feedback and seeding cavities. Adjusting the length of the feedback cavity can be used to control the pulse separation. The 2.5 cm long laser crystal had a diameter of 5 mm and was pumped transversely by 10 ns KrF pulses. Such a laser system produced output pulses of near nanosecond duration having an energy of 650 μJ at a transverse KrF pump flux of 200 mJ/cm^2 and a pump energy of 72 mJ. A similar passive self-injection seeded pulse train laser system was developed with Ce:LiCAF. The Ce:LiCAF system used a shorter gain length and a shorter crystal and thus was able to produce shorter pulses (~600 ps).

Sarukura and Dubinskii have compared Ce:LLF with a similar laser crystal, Ce:YLF[7,13]. The laser characteristics of Ce:YLF were initially investigated by Ehrlich et al in 1979[6]. Ehrlich determined the peak gain cross section for Ce:YLF (at a wavelength of 325 nm) to be 7.6×10^{-18} cm^2. The formation of colour centres in Ce:YLF was studied by Lim and Hamilton[14]. These colour centers were found to produce absorption between 300 and 600 nm. Absorption due to colour centers (induced by pumping with KrF) was found to be less prominent in Ce:LLF [7,13]. The peak of absorption due to colour centres was near 330 nm, and the absorbance of Ce:LLF was roughly 65% of the absorbance of Ce:YLF at 330 nm. Colour centres in Ce:LLF were also studies by Hooker[15], who found that gain was reduced for pump fluxes greater than 40 mJ/cm^2. The spontaneous lifetime of Ce:YLF was reported to be 40 ± 3 ns [6,13], which is equivalent to what was reported for Ce:LLF. Ce:YLF and Ce:LLF exhibit similar fluorescence spectra [7,13]. Ce:YLF was found to have a quantum yield of 0.45 ± 0.05, whereas the quantum yield for Ce:LLF was found to be 0.88 ± 0.08 [7,13]. As described previously, the laser characteristics of Ce:YLF and Ce:LLF were compared, and Ce:LLF generally performed better [7,13].

Rambaldi[8] measured the absorption and fluorescence spectra of Ce:LLF, and also investigated an end-pumped laser, using a Ce:LiSAF laser as a pump (290 nm). This laser produced up to 2.1 mJ of output at a pump flux of ~ 3J/cm^2 and a pump energy of 6 mJ. The output coupler producing the best performance had a transmission of 60%. The laser cavity consisted of two flat mirrors separated by a distance of 25 cm and a 3.65 mm long laser crystal. This laser was tunable from 307.6nm to 313.5nm and from 324 to 328.5nm. Pumping at fluxes up to 4 J/cm^2 was possible without any noticeable solarization or damage to the crystal.

Sarukura and Dubunskii[16] have also developed a tunable Ce:LLF laser, pumped at 213 nm by the fifth harmonic of a Nd:YAG laser. Pump pulses had a duration of 5 ns. The transverse pumping geometry involved a lateral tilt angle as shown below in Figure 1.2-1. Tunability was achieved from 309.5nm to 312.3nm and from 324.5nm to

327.7nm. Pulse durations were as short as 0.88 ns. The peak output of this laser system was 55 μJ at 140 mJ/cm^2 pump flux and a pump energy of 25 mJ. Pumping at 213 nm for several hours at a 10 Hz repetition rate was found to cause no noticeable solarization.

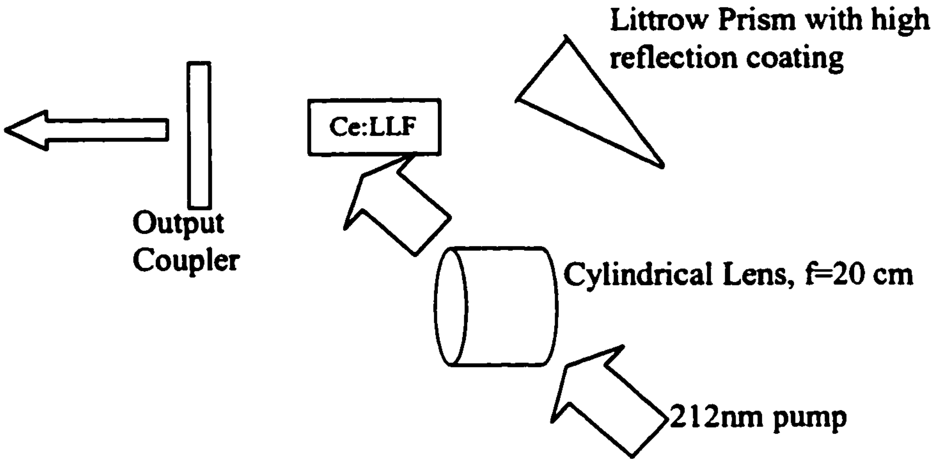

Figure 1.2-1: Slanted pumping scheme used by Sarukura et al. The angle of incidence is approximately 70 degrees with respect to the side window's normal.

McGonigle has investigated the tunability range of Ce:LLF lasers. Initial attempts[17] yielded tunability for only 305.5nm to 316nm and 323nm to 331nm. McGonigle's initial laser configuration consisted of a 2 mm long crystal in a 65 mm long cavity having a 100% reflector with radius of curvature equal to 250 mm and a flat output coupler. Output couplers with reflectivities between 45% and 90% were used. Tunability was achieved with a prism and the rotation of the 100% reflector. The laser was axially pumped by the second harmonic of the yellow output of a copper vapour laser (289nm), which produced kHz repetition rate output. A maximum output power of 300 mW was obtained when pumping at 1.05 W, and making use of an output coupler with a reflectivity of 80%. The slope efficiency of this laser was 38%. Absorption due to colour centres was found to be mostly alleviated by cooling the crystal to –3 °C.

McGonigle's later experiment[18]. achieved tunability over the range 305 nm to 330 nm. A similar cavity configuration was used, with a 2 mm long higher quality crystal. As before, the pump was the frequency doubled output of a copper vapour laser (289 nm), which operated at a repetition rate of 10 kHz. The polarization of the laser output was pi for the short wavelength region (below 327 nm) and sigma polarized near 327nm. Slightly higher output power (except around 327 nm) was obtained with the crystal being cooled to –2°C. The peak output from this laser was 360 mW for pumping at 920 mW, with a slope efficiency of 51%.

High laser output pulse energies have been obtained by Liu[19], using a large Ce:LLF crystal (18 mm diameter, 10 mm length). The laser cavity consisted of two flat mirrors: a 100% reflector and an output coupler with 45% transmission. The laser was transversely pumped with a randomly polarized KrF laser having a repetition rate of 1 Hz. The peak output energy obtained was 27 mJ for a transverse pump flux of 600 mJ/cm^2 and a pump energy of 230 mJ. This laser was tunable over the two separate bands: near 309nm and near 325 nm. The 309nm output was pi polarized, while the 325nm output was sigma polarized.

Chapter2: Theoretical Background

The absorption or emission of photons in a two-level transition of an atomic system is described by the photon transport equation [20]:

$$\frac{\partial \phi}{\partial t} + c \cdot \left(\frac{\partial \phi}{\partial z} \right) = \sigma \cdot c \cdot \phi \cdot (N_2 - N_1) \tag{2-1}$$

where ϕ is the photon flux density, c is the speed of light, σ is the interaction cross-section (absorption or amplification) and N_2 and N_1 are the population densities of the upper and lower states, respectively. The associated equations describing the populations of the two states are:

$$\frac{\partial N_1}{\partial t} = \sigma \cdot c \cdot \phi \cdot (N_2 - N_1) \tag{2-2}$$

$$\frac{\partial N_2}{\partial t} = -\sigma \cdot c \cdot \phi \cdot (N_2 - N_1) \tag{2-3}$$

Solutions to the above set of equations (2-1, 2-2 and 2-3) have been developed by Franz and Nodvik[21]. These solutions are valid for time durations which are much smaller than the upper state lifetime and much greater than the phase relaxation time of the gain medium. The solution for absorption of pump energy is

$$W_P(z,t) = W_{SP} \cdot \ln\left[1 - \exp(\sigma_A \cdot \Delta N_{30} \cdot z) \cdot \left[1 - \exp\left(\frac{W_{P0}(t)}{W_{SP}}\right)\right]\right] \tag{2-4}$$

where $W_{P0}(t)$ is the input pump energy flux and σ_A is the absorption cross-section.

For an ideal 4 level laser system (as shown in Figure 2-1), ΔN_{30} is the difference between population densities of states 3 and 0 (N_3-N_0). W_{SP} is the pump saturation energy density, and is given by

$$W_{SP} = \frac{h \cdot c}{\sigma_A \cdot \lambda} \tag{2-5}$$

The solution for amplification of the laser flux is:

$$W(z,t) = W_S \cdot \ln\left[1 - \exp(\sigma_G \cdot \Delta N_{21} \cdot z) \cdot \left[1 - \exp\left(\frac{W_0(t)}{W_S}\right)\right]\right] \tag{2-6}$$

where $W_0(t)$ is the input energy density, σ_G is the gain cross-section, ΔN_{21} is the difference between population densities of states 2 and 1 (N_2-N_1). W_S is the gain saturation energy density, and is given by:

$$W_S = \frac{h \cdot c}{\sigma_G \cdot \lambda} \tag{2-7}$$

for an ideal 4 level system, or

$$W_S = \frac{h \cdot c}{2 \cdot \sigma_G \cdot \lambda} \tag{2-8}$$

for an ideal 3 level system.

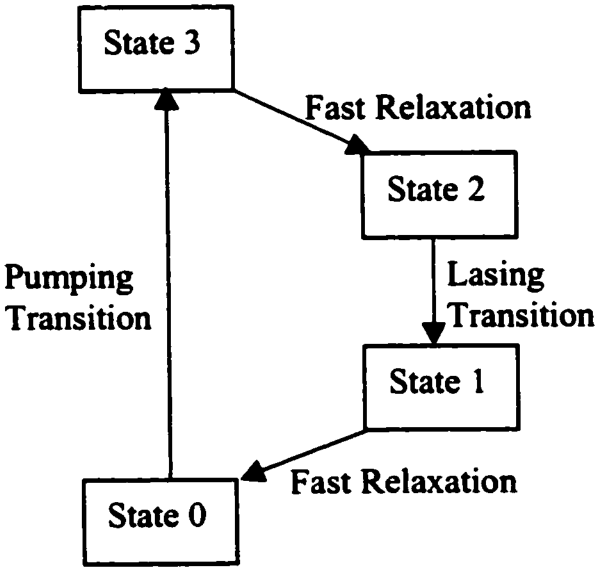

Figure 2-1: An Ideal 4 level laser system.

The spontaneous decay of the upper state (state 2) is described by equation (2-9).

$$\frac{\partial N_2}{\partial t} = -\frac{N_2}{T_1} \tag{2-9}$$

where T_1 is the upper state (state 2 in Figure 2-1) lifetime, which can be alternately expressed as the inverse of the Einstein coefficient A_{21}. It is assumed that every atom which decays from state 2 to state 1 will emit one photon. Thus, the generation of new spontaneous emission is described by equation (2-10).

$$\frac{\partial W_{ASE}}{\partial t} = \left(\frac{h \cdot c}{\lambda_0}\right) \cdot \frac{N_2}{T_1} \tag{2-10}$$

where W_{ASE} is the ASE energy density, and λ_0 is the vacuum wavelength of the spontaneously emitted radiation. Newly generated ASE is emitted over 4π steradians of solid angle.

Equations (2-2) and (2-3) can also be solved for ΔN_{21} due to absorption of pump radiation without any spontaneous decay:

$$\Delta N_{21}(z) = N_T \cdot \left[1 - \frac{1}{1 - \exp(-\sigma_A \cdot N_T \cdot z) \cdot \left[1 - \exp\left(\frac{W_{PO}}{W_{SP}}\right)\right]} \right] \tag{2-11}$$

where N_T is the total population density and W_{P0} is the input pump flux. Solutions of this equation for various values of $sp = W_{P0}/W_{SP}$ are shown in Figure 2-2. With sufficiently high values of sp, it is possible to achieve a full inversion for the initial pumped region.

In the above analysis, no other competing de-excitation mechanisms have been considered. This corresponds to quantum yield of one, meaning that for every pump photon absorbed, one laser photon is emitted. Deviations from this ideal case can be treated by assigning an appropriate branching ratio for the transition of interest.

Transverse pumping of a laser gain medium is often used when there is a large absorption coefficient at the pumping wavelength. This will in general lead to a nonuniform population inversion profile, causing the edge(s) of the amplified laser beam to experience more gain than the center. This effect can to a certain extent be alleviated by pumping from both sides of the gain medium.

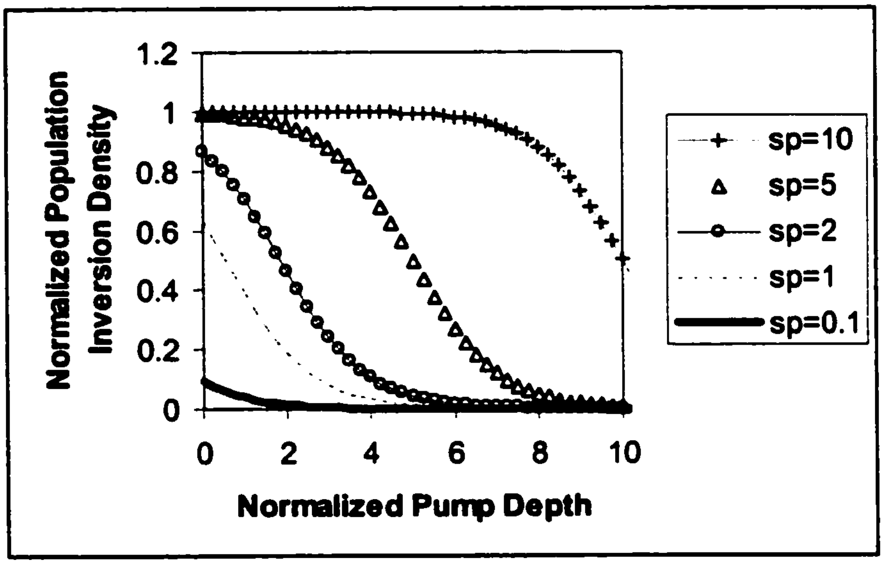

Figure 2-2: Population Inversion Profiles for different values of $sp=W_{P0}/W_{SP}$
Population Inversion Density is normalized with respect to total population density.
Pump depth is normalized with respect to the small signal absorption length
$L_A=1/(N_T*\sigma_A)$.

Amplified Spontaneous Emission (ASE), while necessary for the formation of a laser mode, is undesirable when amplifying a pre-existing laser pulse. In the case of a laser amplifier, ASE depletes the available population inversion and reduces the gain available for the pulse being amplified. ASE will also appear as noise on any detection device being used to record the laser output. The amount of ASE produced will decrease exponentially with time, according to equations (2-9) and (2-10).

While analytic solutions for the coupled gain equations exist for simple cases, the design of a laser system requires accounting for the effects of ASE and spatial and temporal variations in population inversion and laser beams. Thus, it is beneficial to use a computer simulation which accounts for these processes.

Chapter 3: Simulation Algorithm

Two separate simulation programs (compiled in Fortran) have been developed: one for cylindrical geometry (to be used for axial pumping) and one for Cartesian geometry which can be used for transverse or axial pumping. The cylindrical simulation was based on a similar simulation of excimer lasers developed by R. Fedosejevs. Modifications were made to account for optical pumping. The cylindrical simulation was initially tested with published laser results[22]. The geometries of these simulations are shown in Figures 3-1, 3-2 and 3-3. In both cases the cell length in the gain region differs from that in air due to different group velocities.

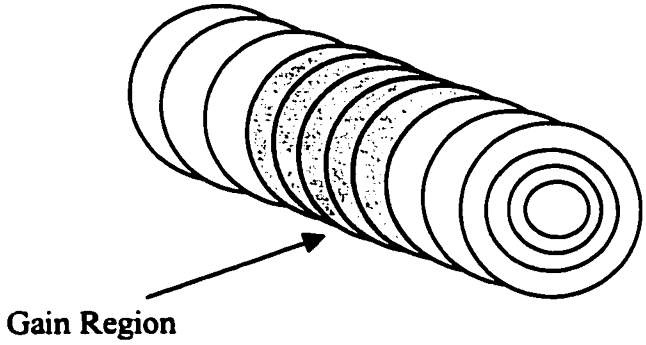

Figure 3-1: Geometry of Cylindrical Laser Simulation

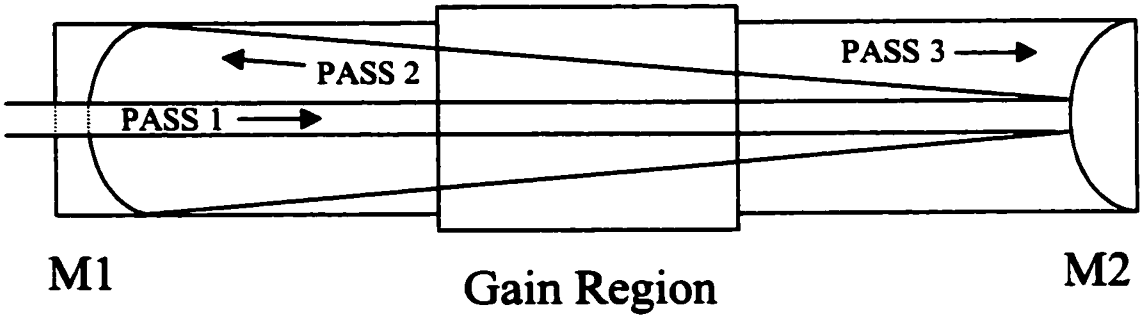

Figure 3-2: An example of divergent Beam Geometry in Cartesian or Cylindrical Simulation.

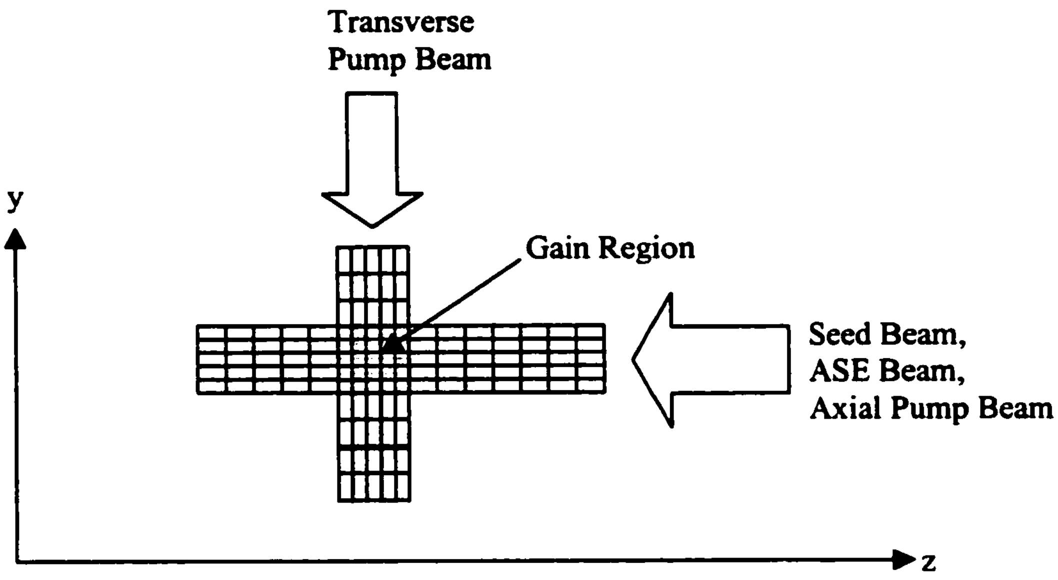

Figure 3-3: Geometry of Cartesian Laser Simulation

As shown in Figure 3-2, divergent beams can be specified for both the cylindrical and Cartesian simulations.

A number of assumptions are made in order to simplify the algorithm. Laser beams are assumed to be monochromatic, and sufficiently large in cross-sectional area to warrant the use of the ray optics assumption (requiring beam diameters to be much larger than the wavelength). The simulation time steps are assumed to be within the range previously specified for validity of the Franz-Nodvik equations (2-4 and 2-6). Finally, the laser system being simulated is assumed to have no non-radiative decay channels. Thus, all atoms promoted to state 3 relax to state 2 and result in the emission of a photon (see Figure 2-1). This corresponds to a quantum yield of unity.

The cylindrical simulation divides the gain region and laser beams into a series of annular rings. For the Cartesian simulation, the gain region is divided up into cubes and the laser beams are divided up into parallelepipeds. In both cases, divergent beams and multiple passes are allowed for. All calculations are based on the total amount of energy contained within a cell of the gain region. This requires the calculation of overlap fractions between laser beam cells and gain region cells. Such a calculation is relatively straightforward for cylindrical geometry, but becomes more involved for the case of divergent beams in a Cartesian geometry. The general shape of an overlap in the Cartesian geometry is that of a prismatoid, as depicted in Figure 3-4. The volume of such a shape is given by equation 3-1, which also applies to parallelepipeds and pyramids[23].

$$V = \frac{\Delta L \cdot (B_1 + 4 \cdot M + B_2)}{6} \tag{3-1}$$

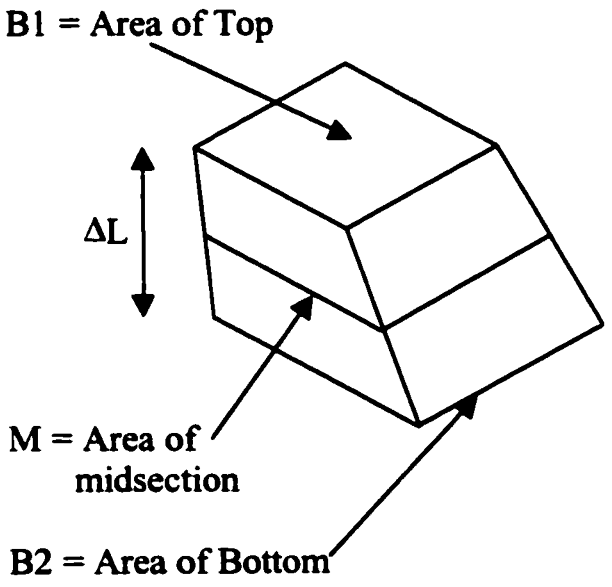

Figure 3-4: A prismatoid

The computer simulation accounts for three processes in the amplification and/or generation of a laser pulse: absorption of optical pump radiation, spontaneous emission, and amplification of spontaneous and stimulated emissions. The simulation can be run for an ideal 3 level system, or an ideal 4 level system (see Figure 2), which is the model used for Ti:Sapphire and Ce^{3+}:$LiLuF_4$. Absorption of pump radiation and amplification of laser radiation are calculated according to the Franz-Nodvik equations (2-4) and (2-6) for each cell. The spontaneous decay of the upper state (state 2) is calculated according to a finite difference version of (2-9):

$$N_2(t_{n+1}) - N_2(t_n) = -\frac{N_2 \cdot \Delta t}{T_1} \tag{3-2}$$

where Δt is the time step of the simulation. A similar finite difference equation is used to calculate the generation of new spontaneous emission:

$$W_{ASE}(t_{n+1}) - W_{ASE}(t_n) = \left(\frac{h \cdot c}{\lambda_0}\right) \cdot \frac{N_2 \cdot \Delta t}{T_1} \tag{3-3}$$

where W_{ASE} is the energy density of the ASE and λ_0 is the vacuum wavelength of the spontaneously emitted radiation. The beams are all propagated forward one cell $\Delta z=(c_o/n)\cdot\Delta t$ either in the axial or the transverse directions before the calculation of the interaction for the next time step.

One of the more complicated aspects of simulating ASE (which is emitted over 4π steradians of solid angle) is the geometry. The simulation programs track the ASE emitted into a given solid angle specified by the user. This could represent the portion of ASE that propagates with the main laser beam or a larger cone angle which will propagate once through the cavity leading to gain depletion. The solid acceptance angle for the ASE is thus usually defined as either large angle or small angle. The large angle ASE is that portion of the ASE which propagates within the gain medium, but does not reflect back for a second pass. Small angle ASE is the portion of ASE which propagates to an end mirror and is reflected multiple times. The small angle ASE is used to simulate the buildup of laser oscillation without an injected seed pulse.

A flowchart of the simulation algorithm is shown in Figure 3-5 with full Fortran code listings (for the Cartesian simulation) in Appendix B. The simulation parameters are specified by the user in a text file (an example parameter file is shown in Appendix A). After the geometrical calculations of beam sizes, gain cells and overlap fractions are finished, the main loop is begun. Input energies are calculated separately for each time step, in order to allow for an unlimited number of time steps. Maximum array sizes are specified within a .inc file and can be changed if necessary when compiling the code. Temporal distributions of beam energies and inversion densities are written to a data file, with a resolution set by the user. Spatial distributions of the beams and inversion density are also written to data files for times specified by the user. A final summary of the simulation is written to a separate file. Detailed results of intermediate calculations can also be written to data files, but this is not recommended for large simulations.

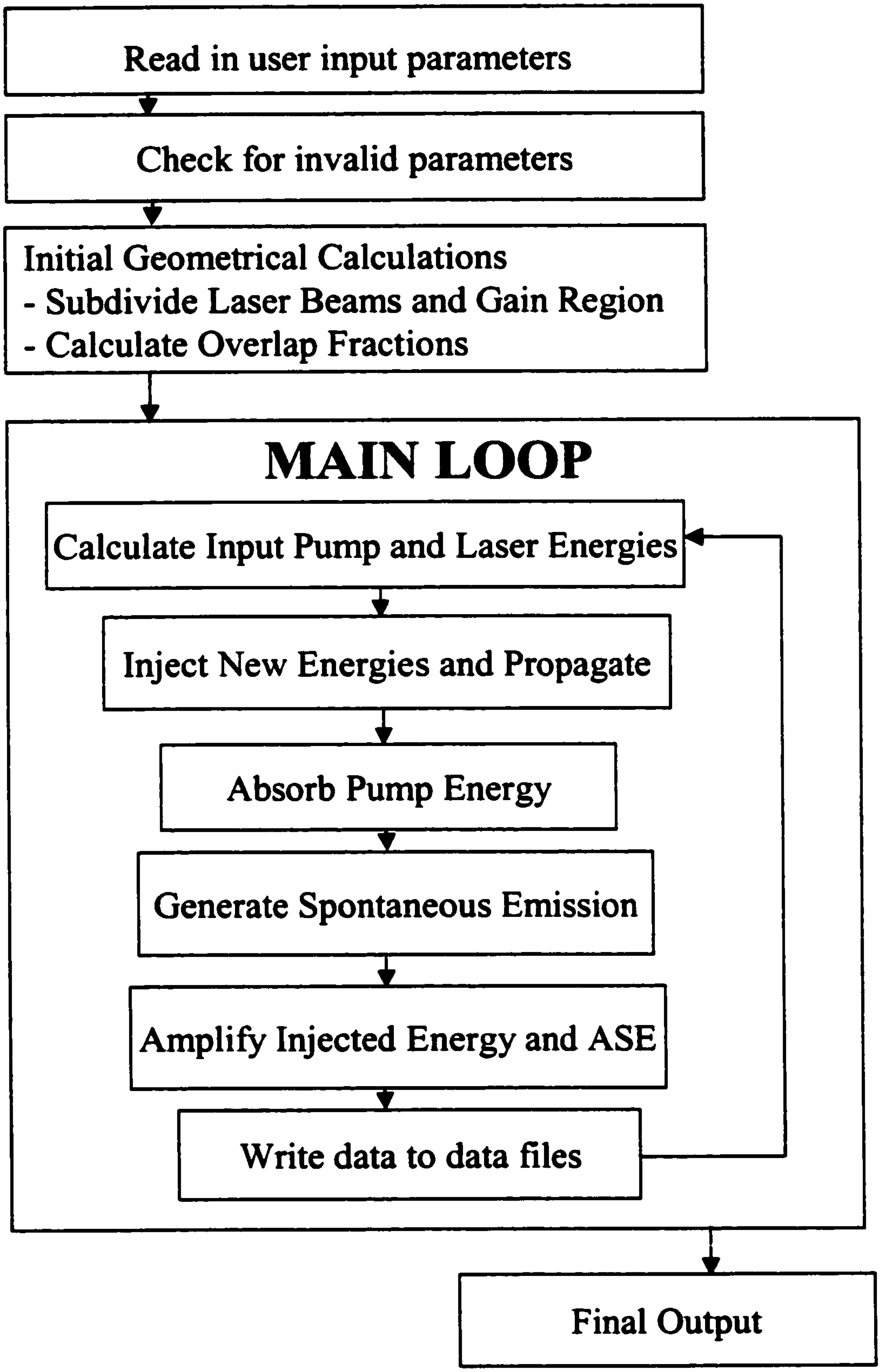

Figure 3-5: Simulation Algorithm

Chapter 4: Experimental Results

4.1 Overview of Experiments

A cylindrical Ce:LLF crystal was obtained from Kazan State University in Kazan, Russia. The cerium concentration was specified as 0.5% in the melt. Based on the specified crystal C axis alignment, a side face for transverse pumping was polished parallel to the labeled c axis to allow for transverse laser pumping, as shown below in Figure 4.1-1. The initial orientation of side face (parallel to the labeled C axis) was chosen to allow for the possibility of pumping with either the sigma or pi polarization. A notch was placed in one end for orientation purposes.

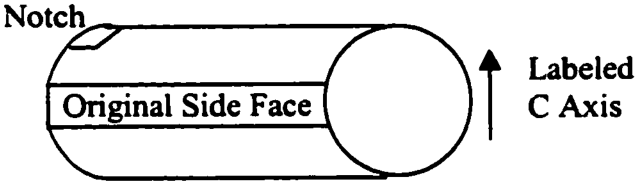

Figure 4.1-1: Original Side Face on Ce:LLF Crystal

A series of measurements were performed on the Ce:LLF crystal in order to obtain parameters which could be used in simulation calculations. The measured results were also used to compare with published results for Ce:LLF[7,8]. Due to certain unforeseen irregularities in labeling of the axes of the Ce:LLF crystal which became apparent during initial measurements, it was necessary to experimentally ascertain the true orientation of the crystal's C axis. A new side face was polished to account for the true crystal axis orientation, and the new face was used for all subsequent experiments. Additional experiments were performed for determining the fluorescence spectra, spontaneous lifetime, absorption coefficients at 248 nm and the gain at 308 nm. The data from the absorption coefficient and gain measurements was used to estimate values for absorption cross-section, gain cross-section and number density of the activator ion (Ce^{3+}) in the crystal.

The pump laser used for the measurement of fluorescence spectra, spontaneous lifetime, absorption coefficient and amplification was a discharge pumped KrF [24,25] laser developed at the University of Alberta. A discharge pumped XeCl laser (also developed at the University of Alberta) was used to provide seed pulses for the amplification measurements. Both of these discharge pumped excimer lasers were initially developed as part of a dye laser system for investigating x-ray generation from plasmas[24].

4.2 Determination of crystal axis.

LiLuF$_4$ is a positive uniaxial crystal, having a tetragonal lattice, C^6_{4h} point group and unit cell dimensions a=b=5.124 Å and c=10.54 Å [26]. In order to determine the exact orientation of the C axis, measurements were made of the birefringence of the crystal for beams propagating in different directions. The refractive indices for standard visible light are n_O=1.468 and n_E=1.494[27,28].

The general case of light propagating in an arbitrary direction through a uniaxial crystal can be described with the assistance of an index ellipsoid[3,29], shown below in Figure 4.2-1. The equation of this ellipsoid is given by:

$$\frac{A^2}{n_A^2}+\frac{B^2}{n_B^2}+\frac{C^2}{n_C^2}=1 \tag{4.2-1}$$

For a uniaxial crystal, n_A=n_B=n_O (the ordinary index) and n_C=n_E (the extraordinary index).

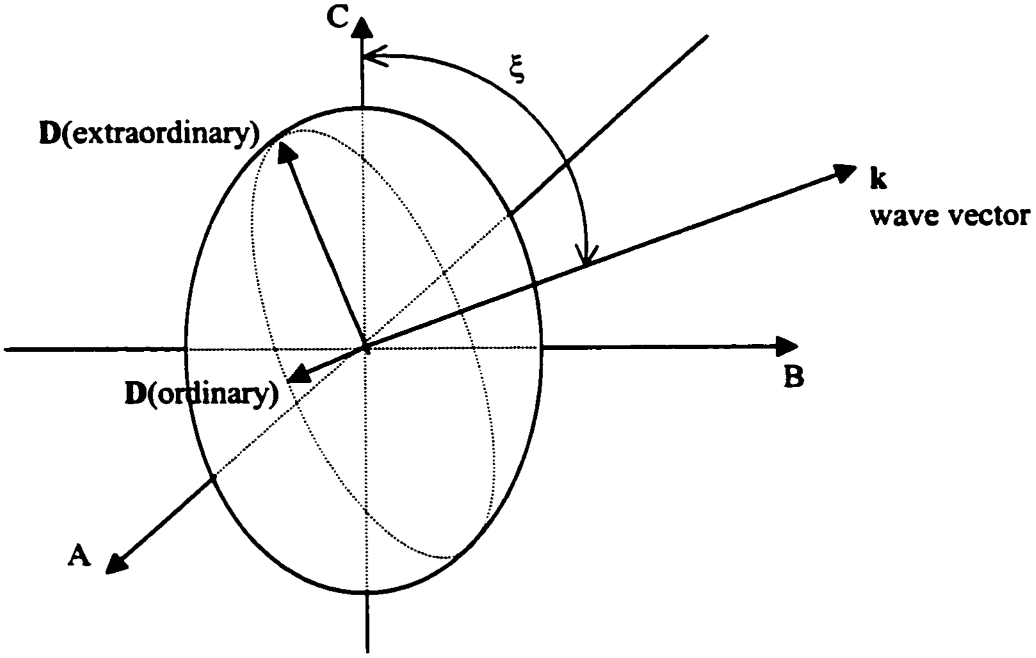

Figure 4.2-1: Index Ellipsoid for a uniaxial crystal.
A,B,C=Crystallographic Axes. D=Electric Displacement Vector.
ξ=Angle between C and **k**.

In general, light will be separated into two orthogonally polarized modes (ordinary and extraordinary) which will have different phase velocities (due to the different refractive indices). The polarizations of the electric displacement vectors of the two modes can be determined by constructing a plane which is normal to the wave vector **k**, and passes through the origin of the ellipsoid. This plane intersects with the surface of the ellipsoid to form an ellipse, which is shown in Figure 4.2-2 with a dashed outline. The major (longer) axis of this ellipse is the direction of the extraordinary mode, while the minor (shorter) axis is the direction of the ordinary mode.

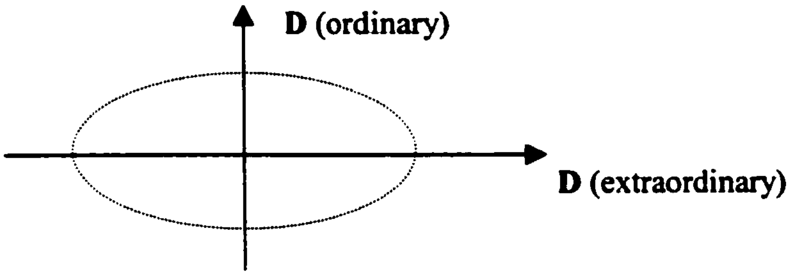

Figure 4.2-2: Polarizations of ordinary and extraordinary modes in a uniaxial crystal. The wave vector **k** is directed out of the page.

The index of the ordinary mode is equal to n_O, while the index of the extraordinary mode depends on the direction of the wave vector **k**, and can be determined by rearranging the equation of the ellipsoid:

$$n(\xi)=\sqrt{\frac{1}{\left(\frac{\cos(\xi)^2}{n_O^2}\right)+\left(\frac{\sin(\xi)^2}{n_E^2}\right)}} \tag{4.2-2}$$

where ξ is the angle between the wave vector and the C axis. The polarization of the extraordinary mode can also be visualized by taking a 2 dimensional projection of the index ellipsoid onto the plane defined by the C axis and the wave vector **k**.

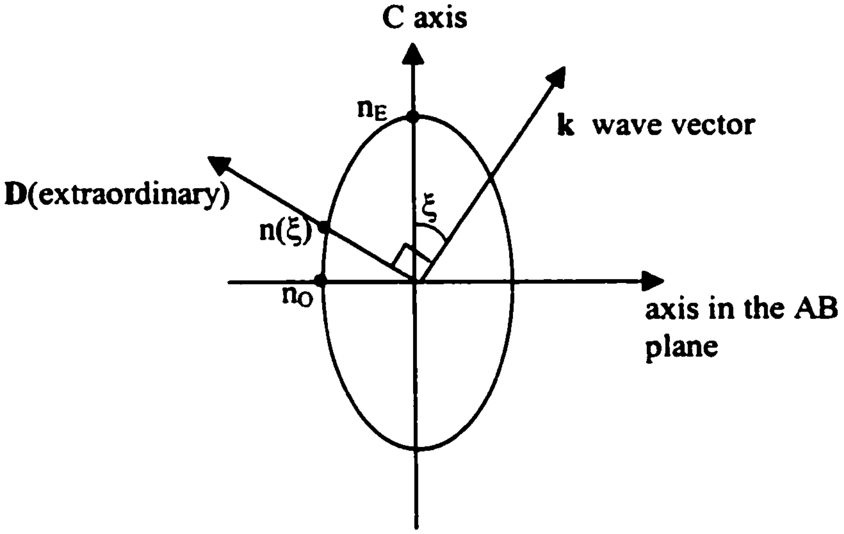

Figure 4.2-3: Angular dependence of index of extraordinary mode.

As shown in Figure 4.2-3, it can be seen that in a uniaxial crystal, the crystal's C axis is found to lie within the plane defined by the wave vector **k** and the electric displacement vector **D** (direction of polarization) of the extraordinary mode. Using this principle, a set of planes can be deduced from measurements of double refraction. The intersection of these planes is the C axis.

The measurements for determining crystal axis were conducted with respect to a co-ordinate system based on the original side face, as shown in Figures 4.2-4 and 4.2-5.

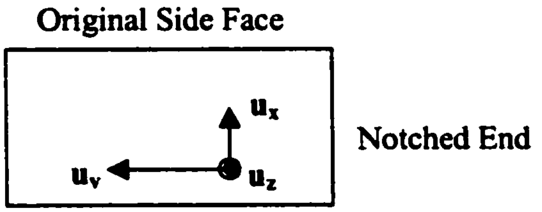

Figure 4.2-4: Unit vectors for determining crystal axis.
$\mathbf{u}_z$ is directed out of the page.

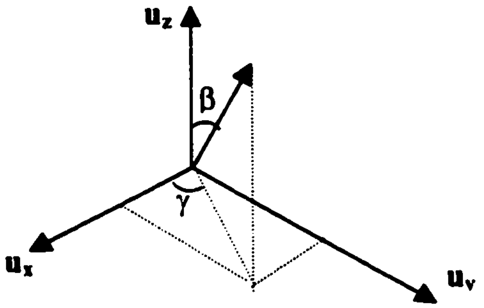

Figure 4.2-5: Unit vectors with corresponding polar co-ordinates.

Double refraction was observed for different incident angles using a prism output coupler in the experimental setup shown in Figure 4-2.6.

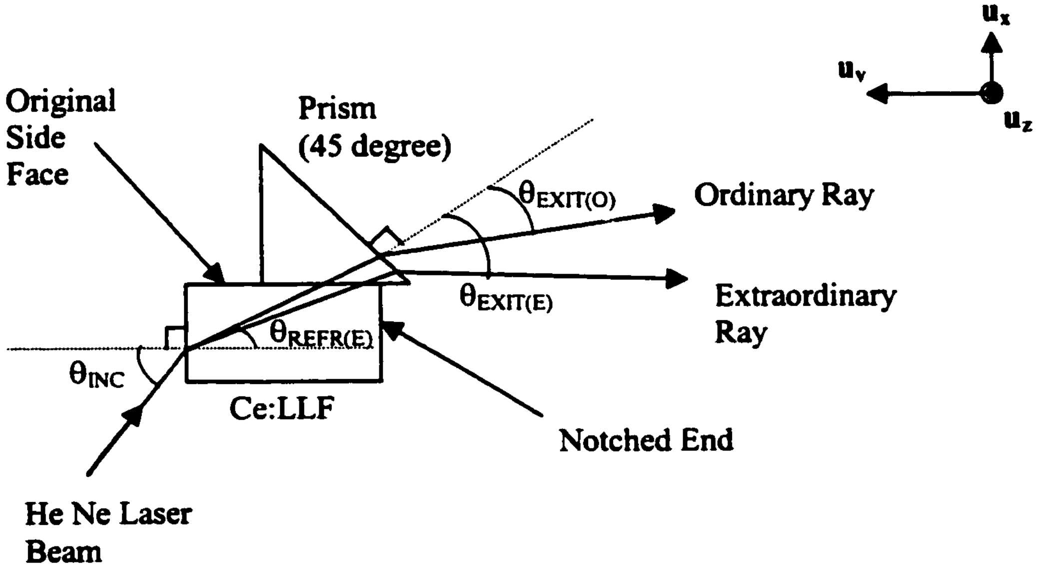

Figure 4.2-6: Experimental setup for double refraction.
$\mathbf{u}_z$ is directed out of the page.

Glycerin (n=1.473) was used for index matching between the Ce:LLF crystal (n_o=1.468; n_e=1.494) and the prism (n=1.4656), thus avoiding total internal reflection at the interface.

The results of measurements using the above setup are summarized in Table 4.2-1. The polarization angle of the exiting extraordinary ray ($\theta_{POL(E)}$) is measured from the vertical (positive $\mathbf{u}_z$ direction), and is inclined towards the positive $\mathbf{u}_x$ direction, as shown in Figure 4.2-7. The direction of the extraordinary wave vector (inside the crystal) is given by the angle $\theta_{REFR(E)}$ as shown below in Figure 4.2-6. The polarization of the extraordinary mode while inside the crystal is determined from the polarization angle of the exiting extraordinary ray. The slight difference in these polarization angles (shown in Table 4.2-1) is calculated from the Fresnel equations. The polarizatrion angle of the exiting extraordinary ray is measured by mounting a polarizer in the beam path. The polarizer is rotated to obtain maximum transmission, and this angle is measured with a rotatable level. All other angles in Table 4.2-1 are measured as shown in Figure 4.2-6.

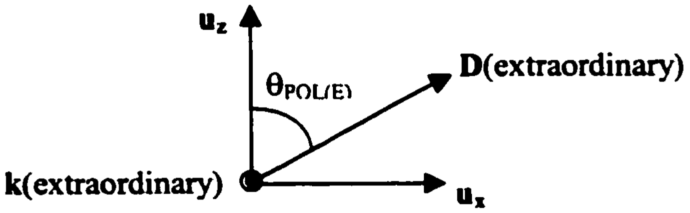

Figure 4.2-7: Measurement of polarization angle of extraordinary ray. The wave vector of the extraordinary ray is directed out of the page.

Plane	Incident Face	θ_{INC} Angle of Incidence	$\theta_{EXIT(O)}$ Exit Angle - Ordinary	$\theta_{EXIT(E)}$ Exit Angle - Extraordinary	$\theta_{REFR(E)}$ Angle of Refraction - Extraordinary	$\theta_{POL(E)}$ Polarization Angle - Extraordinary	Internal Polarization Angle - Extraordinary
A	notched	33.5	40	44	21.5	57	56.0
B	unnotched	60	15.5	18	35.1	55.5	55.4
C	notched	40	31	34.5	25.4	56	55.4
D	notched	58.5	17	18.5	34.6	54	53.9
E	unnotched	61	15	17.5	35.5	55.5	55.4
F	unnotched	69	10	13	38.3	54	53.9

Table 4.2-1: Double Refraction data. All angles are in degrees.

For any of the different incident angles investigated, the plane defined by the extraordinary wave vector (represented by the extraordinary angle of refraction) and the electric displacement vector (represented by the polarization angle) will contain the C axis. Thus, the intersection of any two of these planes should represent the C axis. The intersection of two planes (having normal vectors $\mathbf{n}_{p1}$ and $\mathbf{n}_{p2}$) can be determined with a vector cross product:

$$\mathbf{v}_{p1p2} = \pm(\mathbf{n}_{p1} \mathrm{X} \mathbf{n}_{p2}) \tag{4.2-3}$$

where $\mathbf{v}_{p1p2}$ is a vector which defines the intersection of the plans p1 and p2. The normal vectors $\mathbf{n}_{p1}$ and $\mathbf{n}_{p2}$ are also determined by vector cross products:

$$\mathbf{n}_{p1} = \pm(\mathbf{v}_{POL(1)} \mathrm{X} \mathbf{v}_{REFR(1)}) \tag{4.2-4}$$

$$\mathbf{n}_{p2} = \pm(\mathbf{v}_{POL(2)} \mathrm{X} \mathbf{v}_{REFR(2)}) \tag{4.2-5}$$

where $\mathbf{v}_{POL(1)}$ is a vector parallel to the electric displacement vector (the direction of polarization) of the extraordinary mode and $\mathbf{v}_{REFR(1)}$ is a vector parallel to the wave vector of the extraordinary mode inside the crystal. In all of the above cross product formulas (4.2-3, 4.2-4, 4.2-5), the positive and negative solutions represent vectors in opposite directions. Either the positive or negative solution can be used.

Table 4.2-2 shows unit vectors representing the intersections of selected pairs of planes, which are estimates of the orientation of the crystal's c axis (in the co-ordinate system of the original side face). These unit vectors are represented in terms of the crystal's orginal side face, as shown previously in Figure 4.2-4 and 4.2-5. The C axis orientation determined from the birefringence measurements in shown in Figure 4.2-8.

Planes	u_x	u_y	u_z	β [degrees]	γ [degrees]
A,B	0.8531	0.0826	-0.5151	121.0	5.53
C,E	0.856	0.0745	-0.5116	120.8	4.97
D,F	0.8619	0.0294	-0.5061	120.4	1.95

Table 4.2-2: Estimates of axis orientation.

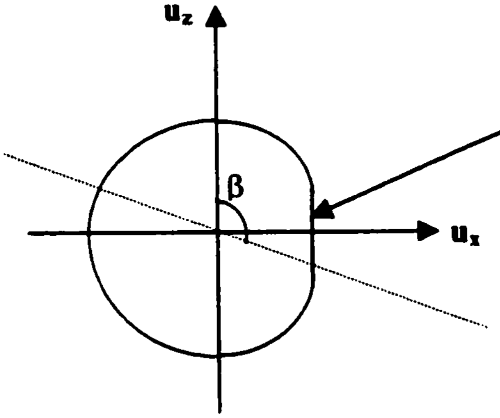

Figure 4.2-8: Axis orientation deduced from Birefringence Measurement. The crystal is viewed from the notched end.

Insight into the crystal's axial orientation can also be gained from observing transmitted light through the crossed polarizer setup shown in Figure 4.2-9. The transmitted power is measured with a power meter. The orientation of crossed polarizers which produces null transmission represents two planes, one of which contains the crystal's c axis. These planes are shown with dashed lines in Figure 4.2-10. The angles of these planes with respect to the $\mathbf{u_z}$ axis are $\beta_1=33°$ and $\beta_2=123°$. Plane #2 is close to the results of the birefringence experiment.

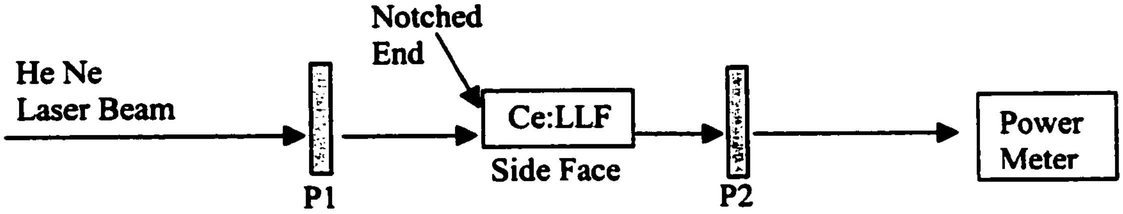

Figure 4.2-9: Crossed Polarizers. P2 is oriented at an angle of 90° with respect to P1.

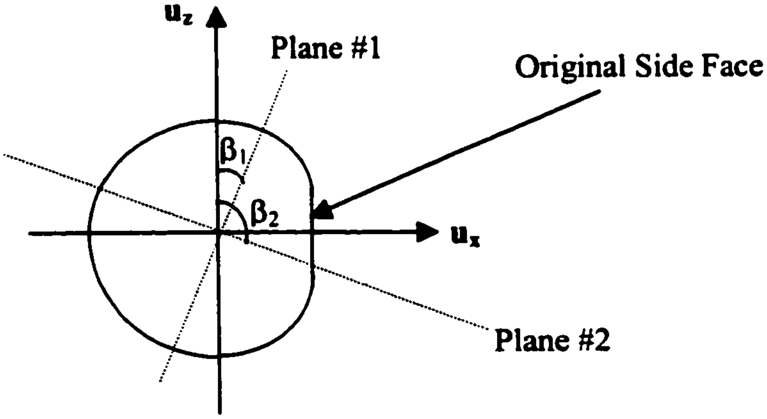

Figure 4.2-10: Possible axis orientations deduced from crossed polarizer measurement. The crystal is viewed from the notched end.

The estimates of axis orientation deduced from double refraction measurements are consistently pointed downwards (negative $\mathbf{u_z}$ direction) at about 59 degrees from the vertical ($\beta \cong 121°$). This is somewhat different from the estimate produced by the crossed polarizer measurement. The source of this 2° discrepancy may be due to the limited accuracy of the angular measurements (± 1°).

An X-Ray diffraction (XRD) measurement was performed on the crystal's original side face in order to obtain an approximation to the orientation of the face. The X-Ray system used was a Rigaku geigerflex vertical goniometer in the Department of Earth and Atmospheric Sciences at the University of Alberta. A summary of $LiLuF_4$ crystal planes detectable with XRD is shown in Table 4.2-3. The angle 2θ is determined from the Bragg formula:

$$\sin(\theta) = \frac{\lambda}{2 \cdot d_{hkl}} \qquad (4.2\text{-}6)$$

where λ is the x-ray wavelength and d_{hkl} is the perpendicular distance between equivalent (hkl) planes.

d_{hkl} [Angstroms]	h	k	l	2 Theta [degrees]
4.620	1	0	1	22.327
2.990	1	1	2	34.814
2.900	1	0	3	35.931
2.630	0	0	4	39.767
2.560	2	0	0	40.902
2.241	2	1	1	47.049
2.135	1	1	4	49.538
1.951	1	0	5	54.577
1.920	2	1	3	55.534
1.840	2	0	4	58.173
1.806	2	2	0	59.377
1.685	3	0	1	64.126
1.581	1	1	6	68.912
1.552	2	1	5	70.387
1.537	3	0	3	71.178
1.317	0	0	8	85.559
1.257	2	1	7	90.731

Table 4.2-3: X-Ray diffraction reference data for Lithium Lutetium Fluoride. The assumed wavelength is 1.79 Angstroms. Data is taken from a reference sheet prepared by the International Centre for Diffraction Data.

The result of the XRD analysis is shown in Figure 4.2-11. The relatively strong peak near $2\theta=90^\circ$ indicates that the original side face has an orientation approximated by the Miller Indices (217). A definitive determination of the crystal axis from X-Ray diffraction would require rotating the sample, which was not possible. While this was not feasible in this case, the stationary X-Ray Diffraction can nevertheless provide an approximation of the crystal's orientation. The (217) face orientation can be interpreted as in Figure 4.2-12. The intercepts (in Angstroms) of the (217) plane are determined by

$$\begin{aligned} Intercept &= \frac{unit\_cell\_length}{Miller\_Index} \\ Intercept(A) &= \frac{1}{2}\cdot(5.12) = 2.56 \\ Intercept(B) &= \frac{1}{1}\cdot(5.12) = 5.12 \\ Intercept(C) &= \frac{1}{7}\cdot(10.54) = 1.5057 \end{aligned} \tag{4.2-7}$$

The normal vector to the (217) plane (expressed in terms of the crystallographic axes A,B and C) is thus $0.4914\mathbf{u}_A + 0.2457\mathbf{u}_B + 0.8355\mathbf{u}_C$. The angle between the (217) plane normal and the c axis can be calculated from equation 4.2-8.

$$\theta_{c217} = \tan^{-1}\left[\sqrt{\left(0.588^2+0.294^2\right)}\right] = 33^\circ \tag{4.2-8}$$

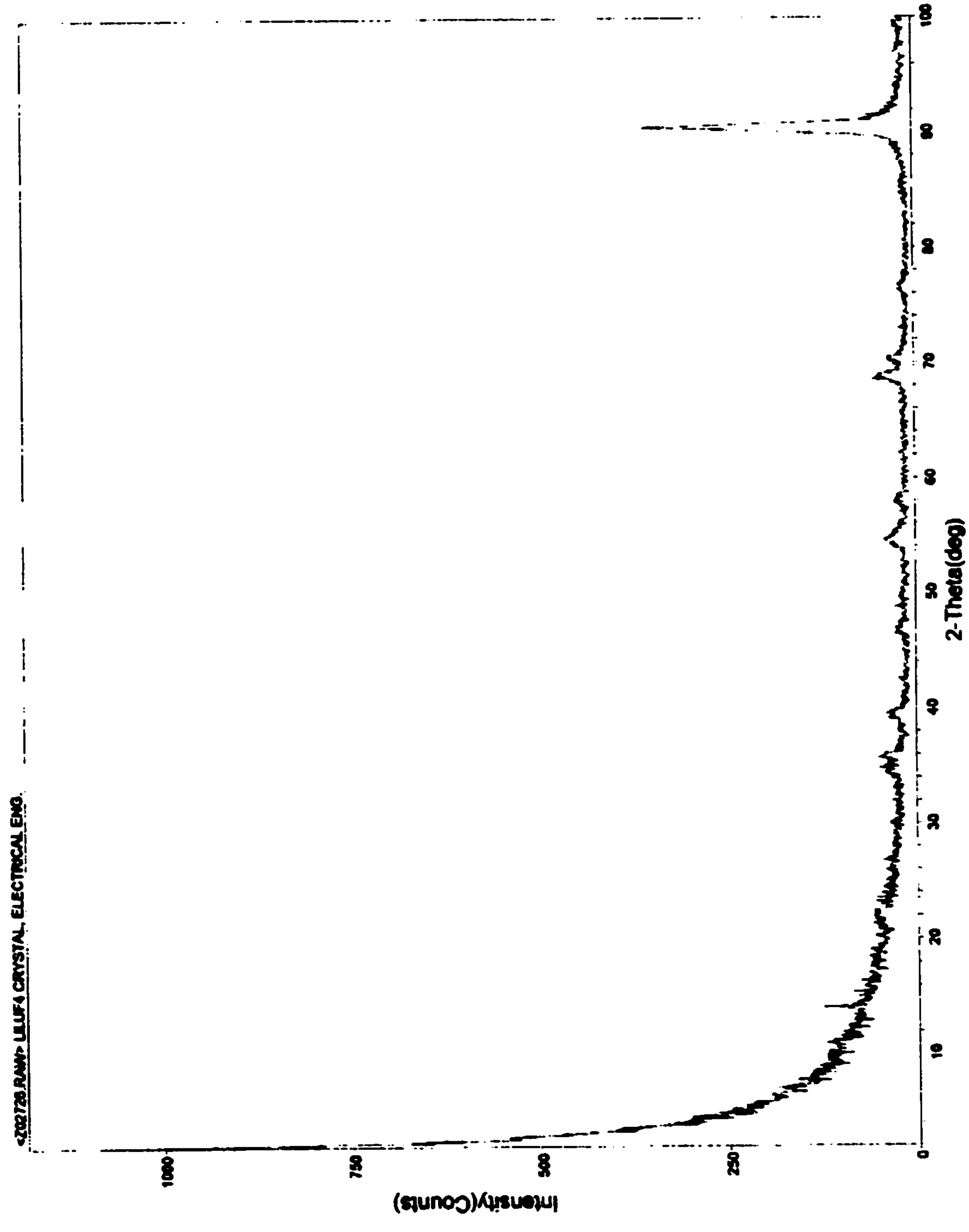

Figure 4.2-11: X-Ray Diffraction Pattern for original side face of Ce:LLF crystal

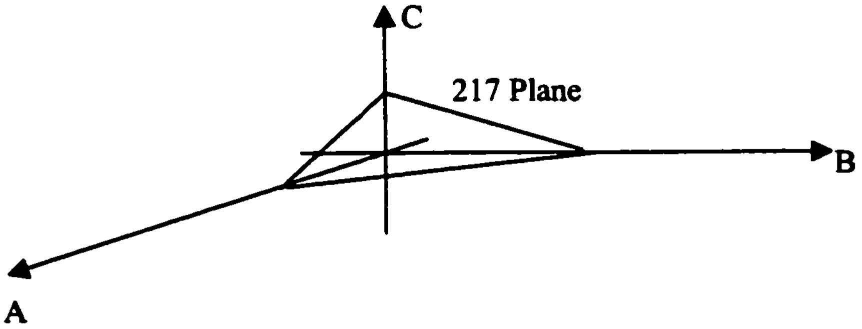

Figure 4.2-12: (217) Plane in the Lithium Lutetium Fluoride crystal.

Thus, the XRD analysis indicated a C axis orientation of 33 degrees from the $\mathbf{u}_x$ direction (normal to the original side face). This represents a cone of possible orientations for the C axis. If the C axis is assumed to lie within the xz plane, there are two possible solutions, corresponding to $\beta_1=57^\circ$ and $\beta_2=123^\circ$, as shown in Figure 4.2-13. This result is only the closest matching crystal plane and is not necessarily an exact indication of the crystal axis.

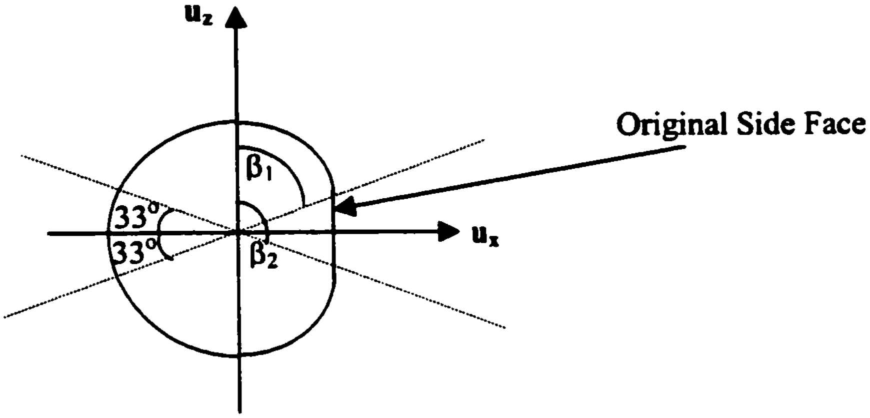

Figure 4.2-13: Possible axis orientations deduced from XRD Analysis. The crystal is viewed from the notched end.

Based on the above analysis, a new side face was polished on the crystal, such that the new crystal face would be normal to the correct C axis. By this time, there were indications of a very high absorption coefficient at 248 nm, which would make pumping with the pi polarization impractical. Thus, by polishing a new face normal to the correct C axis, it was ensured that all pumping on this new face would be sigma polarized. All subsequent experiments with the Ce:LLF crystal were performed with the new face oriented vertically with its surface normal in the horizontal plane. The new face was also used as the pumping face for amplification experiments.

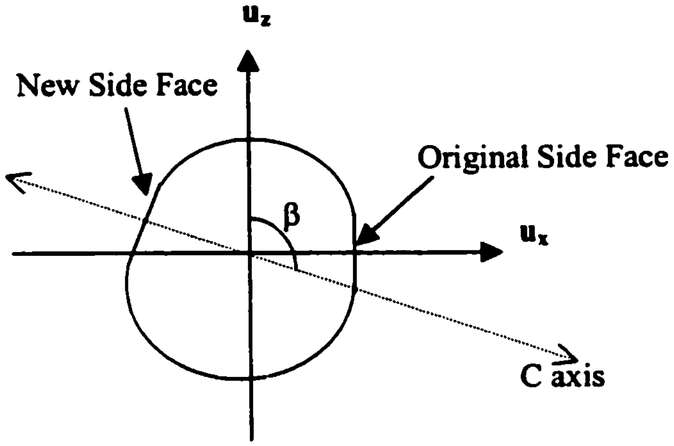

Figure 4.2-14: Cross Sectional view of Ce:LLF crystal showing C axis. The crystal is viewed from the notched end.

Figure 4.2-14 shows the C axis with respect to the original co-ordinate system shown in Figures 4.2-4 and 4.2-5. The birefringence measurements yield β=121° , while the X-Ray diffraction and crossed polarizer measurements yield β=123°. The new side face was cut using the value of β=123°.

4.3 Fluorsecence Spectra of Ce^{3+}:$LiLuF_4$.

The polarized fluorescence spectra of the Ce:LLF crystal were investigated with a miniature fibre coupled digital Spectrograph (Ocean Optics SD2000), using the setup shown in Figure 4.3-1. A Wollaston prism was used to separate the fluorescence into orthogonal polarizations. A biconvex lens was used to produce magnified images of the polarized fluorescence. An optical fibre was used to convey the fluorescence to the spectrometer. Pump fluxes in the range of 45 to 70 mJ/cm^2 were employed using a discharge-pumped KrF laser. The length of the illumination region was limited to approximately 2 mm in order to avoid nonlinear amplification of the fluorescence spectrum.

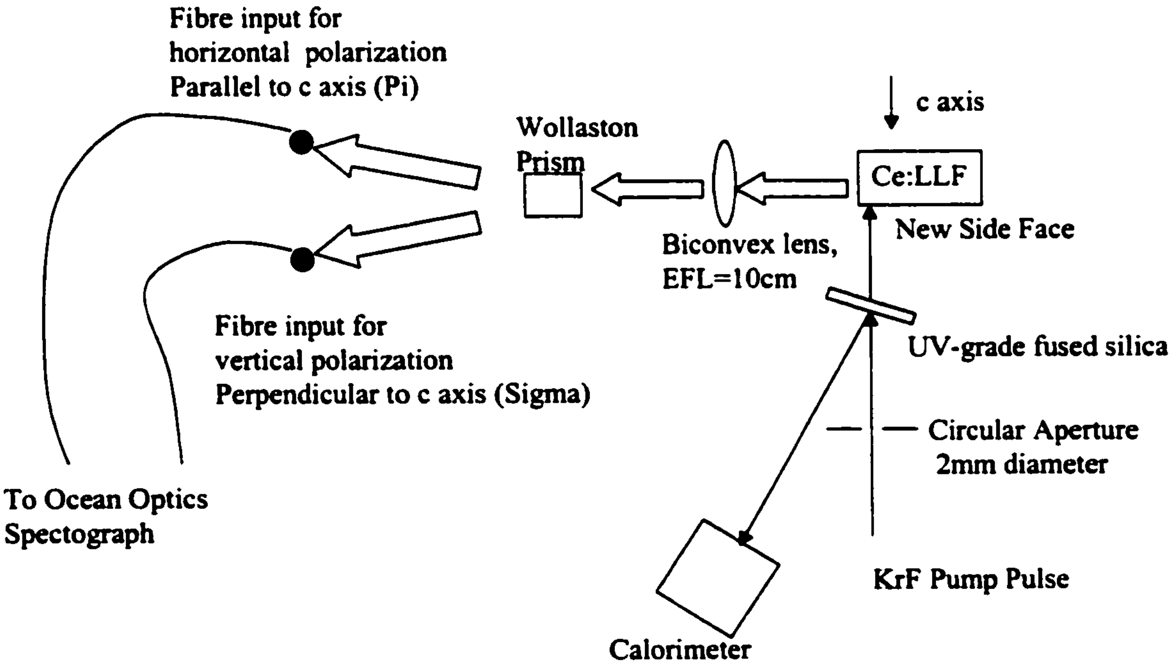

Figure 4.3-1: Experimental setup for measuring fluorescence spectra of Ce:LLF

The fluorescence data was averaged over 10 shots, and corrected for the responsivity of the spectrometer (determined by observing the spectrum of a Tungsten calibration lamp). The measured responsivity of the spectrograph in the region of the Ce:LLF fluorescence output is shown in Figure 4.3-2. A 200 µm slit and a 600 lines per mm grating (blazed at 400 nm) were used in the spectrograph giving a spectral resolution of 2.7 nm at 320 nm.

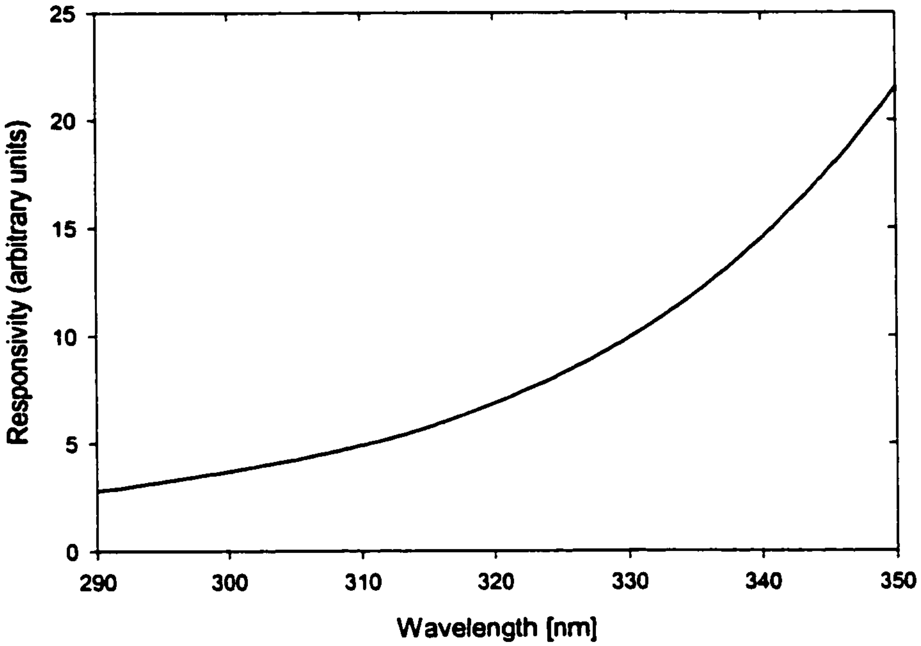

Figure 4.3-2: Responsivity Curve for Ocean Optics Spectrograph

The resulting measured fluorescence curves are shown in Figures 4.3-3 and 4.3-4. Particularly strong pi fluorescence around 311 nm and strong sigma fluorescence around 328 nm was observed. These results are similar to those published by Rambaldi[8], but dissimilar to the results of Sarukura, Dubinskii et al.[7] In the latter measurements it was observed that the Pi fluorescence is strongest at 325 nm which is different from our results presented in Figure 4.3-3. It is possible that different crystal doping concentrations may lead to different amounts of re-absorption of fluorescence, producing different observed curves.

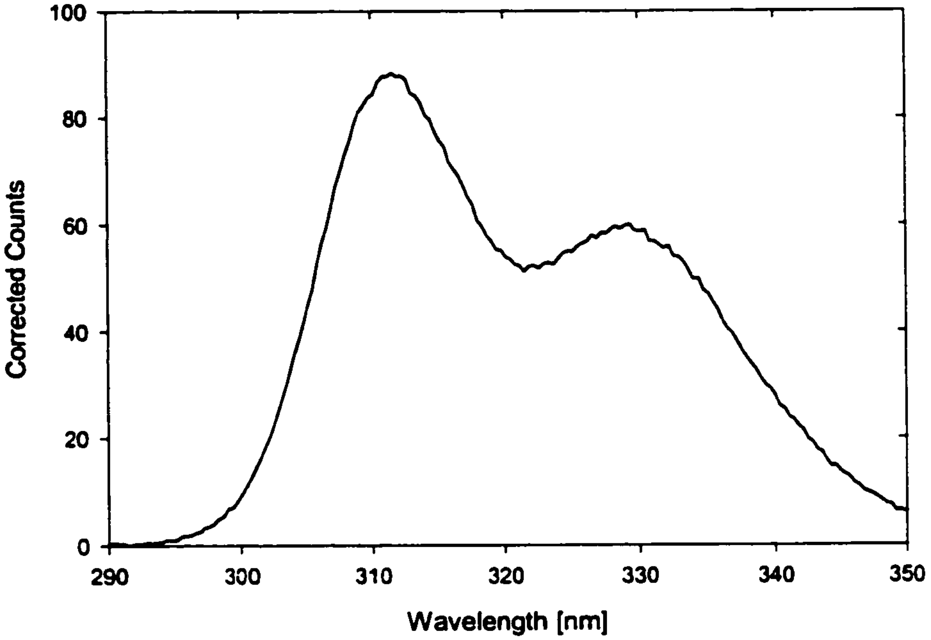

Figure 4.3-3: Ce:LLF Pi Fluorescence Spectrum.

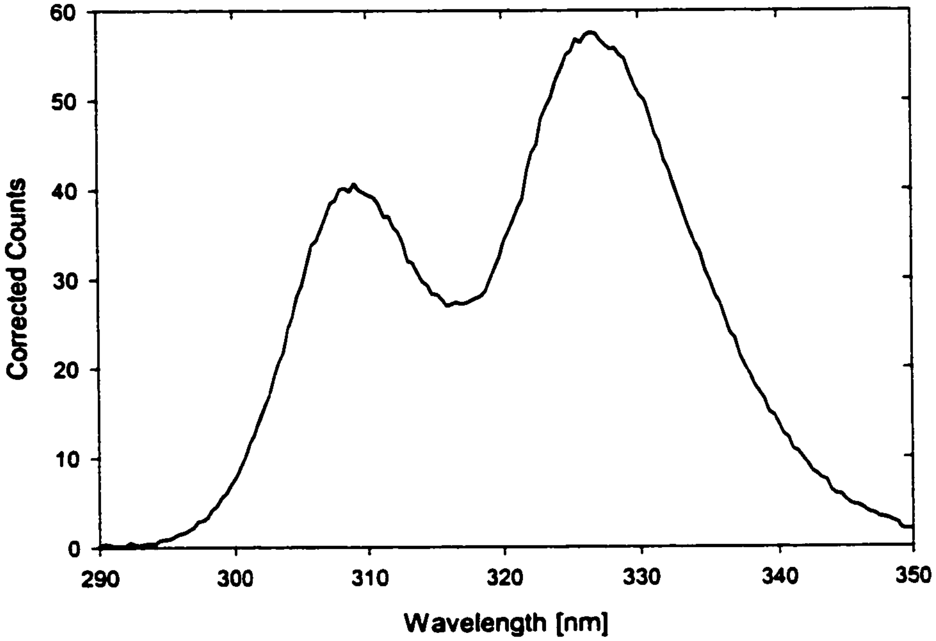

Figure 4.3-4: Ce:LLF sigma fluorescence spectrum.

The pi polarized fluorescence band near 325 nm has a full width half max bandwidth of 55.5 THz (assuming a symmetric fluorescence line). The sigma polarized fluorescence band near 325 nm has a full width half max bandwidth of 44.9 THz. The average of these two values would be 50.2 THz. These values are comparable with bandwidths determined from published fluorescence curves for Ce:LLF, as shown in Table 4.3-1.

Group	Polarization	FWHM [THz]
Ranieri	Unpolarized	64.4
Sarukura	Sigma	56.5
Sarukura	Pi	61.8
Rambaldi	Sigma	45.2
Rambaldi	Pi	50.5

Table 4.3-1: Comparison of bandwidths for Ce:LLF fluorescence band at 325 nm. Data is taken from Ranieri[30], Sarukura[7] and Rambaldi[8].

4.4 Spontaneous Lifetime of Ce^{3+}:$LiLuF_4$

A KrF laser was used as a pump source for the excitation of fluorescence in Ce:LLF. The spontaneous lifetime of the Ce:LLF laser crystal was determined by taking a photodiode measurement of the fluorescence, and comparing this with a convolution of the KrF pump pulse with various fluorescence decay curves. Curve fitting analysis was performed in Matlab to obtain the solution of minimum RMS error for normalized versions of the experimental and simulated data using the fluorescence lifetime as a free parameter. In order to obtain a curve-fit comparison, the data was first low-pass filtered, and then pseudodeconvolved[31,32,33] to remove temporal broadening due to the limited bandwidth of the photodiode/oscilliscope measurement system. Low pass filtering (with a Hamming filter) was implemented in Matlab at a cutoff frequency of 200 MHz (equal to the oscilliscope bandwidth) and was able to reduce the amount of noise present in the data files. The overall impulse response of the photodiode, cables and oscilliscope was measured using ultraviolet picosecond pulses. The impulse response had a full width half max of 1.7 ns, and was employed in the pseudodeconvolution routine.

A diagram of the experimental setup for measuring the spontaneous lifetime is shown below in Figure 4.4-1. The experiment was set up to minimize the amount of gain experienced by the amplified spontaneous emission (ASE) in the pumped region by restricting the lateral size of the pumped region.

The algorithm for determining the spontaneous lifetime is shown in Figure 4.4-2. The data sequences for the KrF and fluorescence pulses are first low pass filtered and then pseudodeconvolved to remove instrument broadening. The resulting filtered and deconvolved fluorescence pulse is compared with successive theoretical fluorescence curves, based on convolving the pump pulse (after filtering and deconvolution) with an exponential decay function for fluorescence having a time constant between 32 and 42 ns. For each data set, the theoretical fluorescence curve which has the lowest RMS error is taken as the best fit. A typical KrF pump pulse is shown in Figure 4.4-3. An example of the best fit fluorescence curve is shown in Figure 4.4-4.

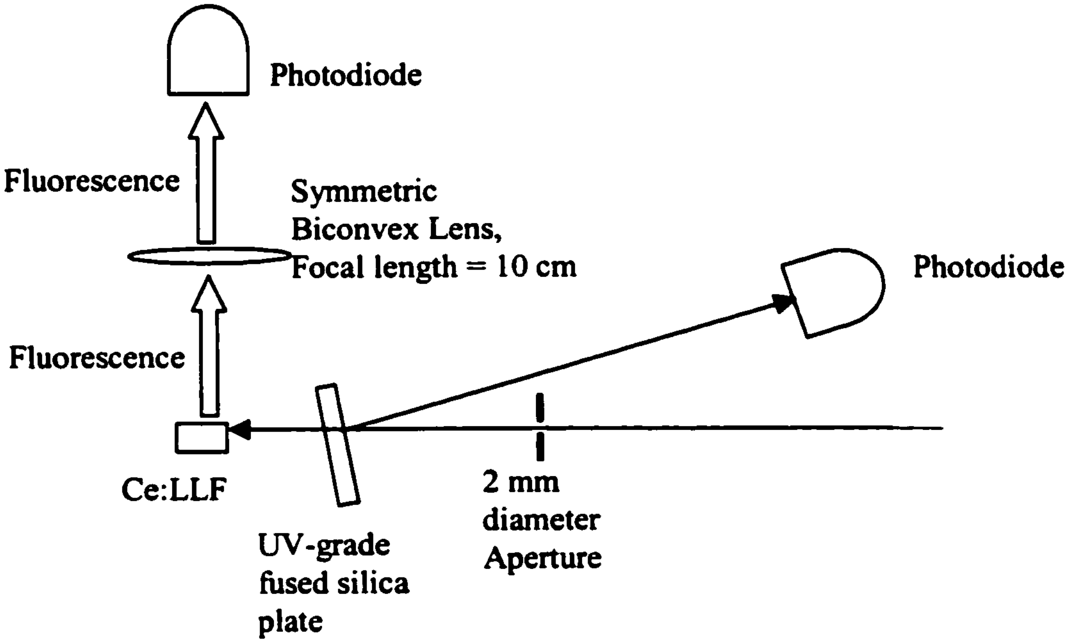

Figure 4.4-1: Experimental setup for measuring spontaneous lifetime of Ce:LLF

A summary of the experimental results is given in Table 4.4-1. After weighting each case by the inverse of its variance, the weighted average spontaneous lifetime was determined to be 37.6 with a standard error calculated from the scattering of the data of ± 0.4 ns. When other potential systematic errors are included, the accuracy of the measurement is estimated to be ± 1.0 ns. The results are quite close to the reported value of 40 ± 3 ns, determined by Sarukura and Dubinksii [7,13].

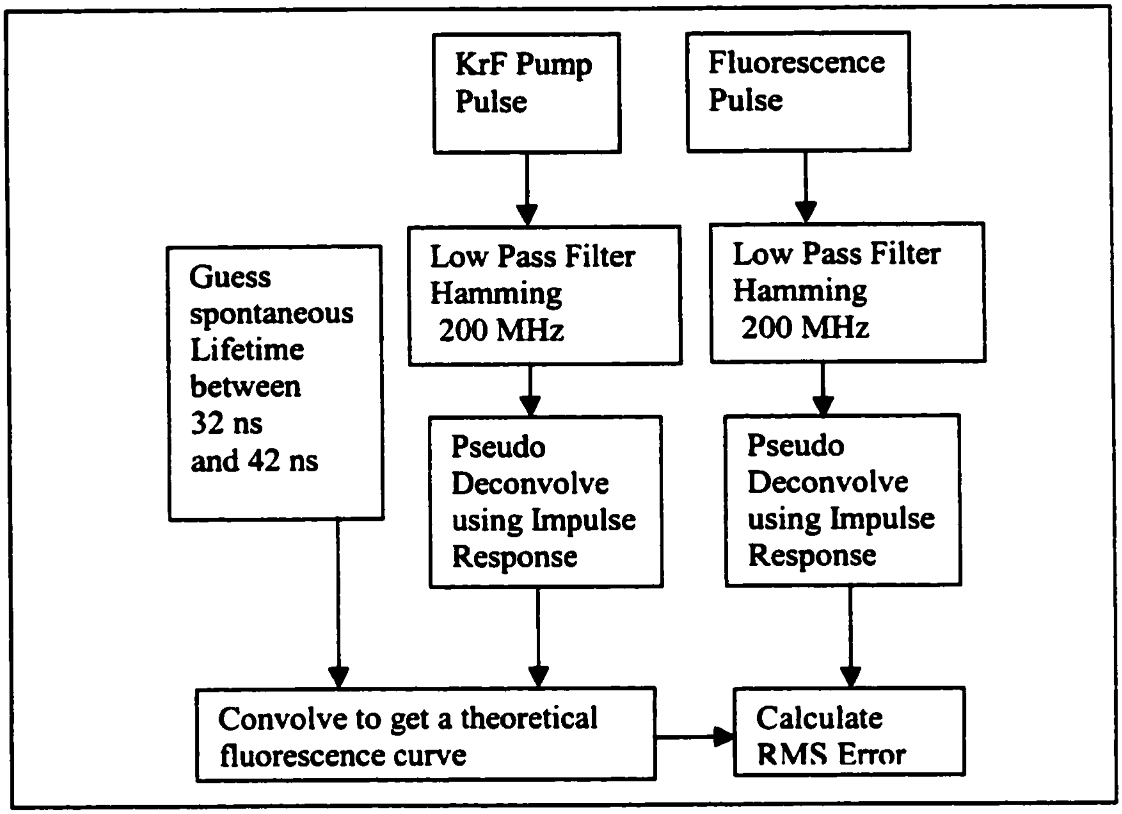

Figure 4.4-2: Algorithm for determining fluorescence lifetime, as carried out in MATLAB.

KrF Pump Flux [mJ/cm^2]	KrF Pump Duration FWHM [ns]	Spontaneous Lifetime best fit [ns]	RMS Error of best fit
9.80	18.5	37.4	0.0353
6.55	17.5	39.3	0.044
10.11	19	34.6	0.0462
8.59	18	40.4	0.0495
9.16	18.5	40.4	0.0364
8.82	18.5	36.8	0.0376
8.52	18	37.6	0.0408
8.47	18	39.4	0.0408
6.29	17	36.1	0.0521
5.70	16.5	37.5	0.0588
6.08	18	36.9	0.0593
5.85	16.5	34.6	0.0535
5.75	17.5	38.2	0.0434
5.55	16	38.8	0.0384
5.32	16.5	38.1	0.0313
5.39	18	39	0.0314
4.95	16.5	36.4	0.039

Table 4.4-1: Experimental Data for spontaneous lifetime of Ce:LLF

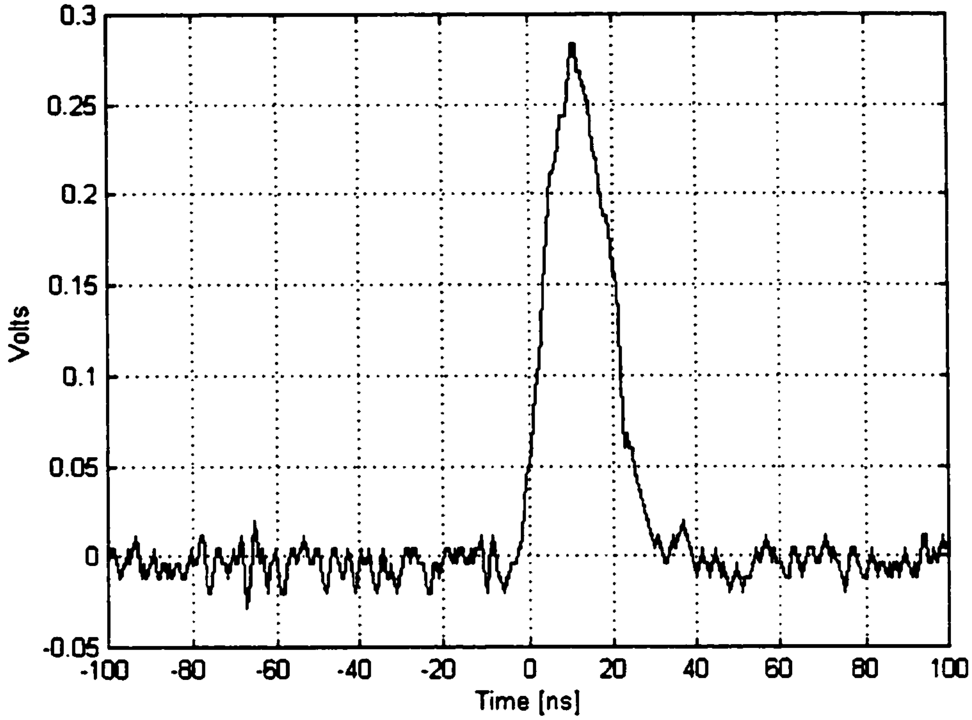

Figure 4.4-3: KrF Pump Pulse

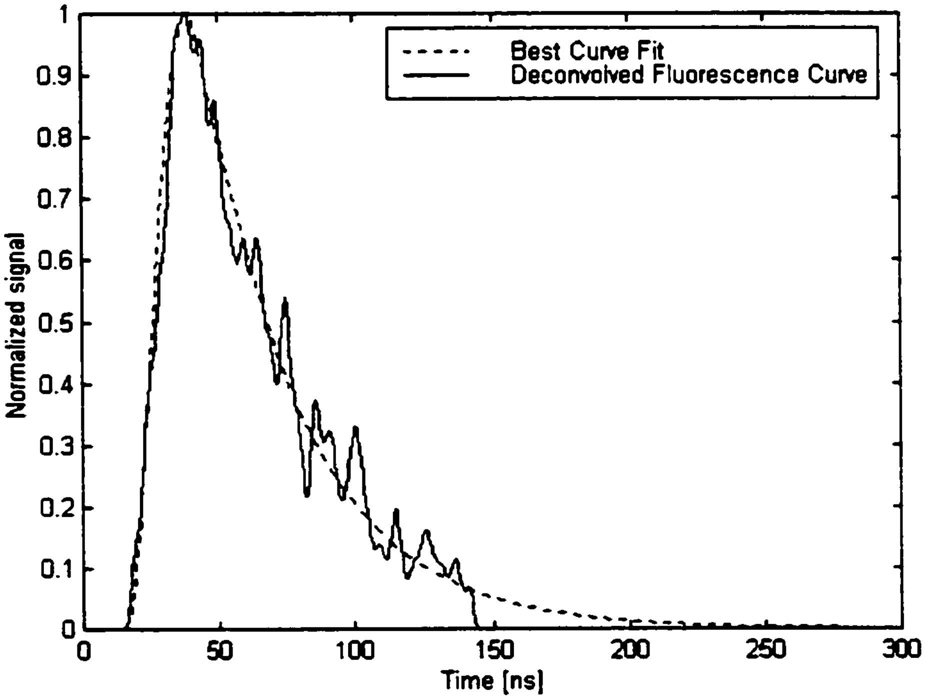

Figure 4.4-4: Measured Fluorescence compared with a calculated response for a lifetime of 36.1 ns.

4.5 Absorption Coefficient of Ce^{3+}:LiLuF$_4$

Initial measurements of crystal absorption were performed with a Cary 2415 spectrophotometer. An unpolarized absorption spectrum through the 7.5mm length of the crystal is shown in Figure 4.5-1. The absorption peaks at 205 nm, 245 nm and 295 nm appear flat due to the saturation of the spectrophotometer, which does not have enough range to measure the true absorbance at these wavelengths. Absorbances at these wavelengths are expected to be significantly higher than indicated on the graph, indicative of a heavy doping concentration.

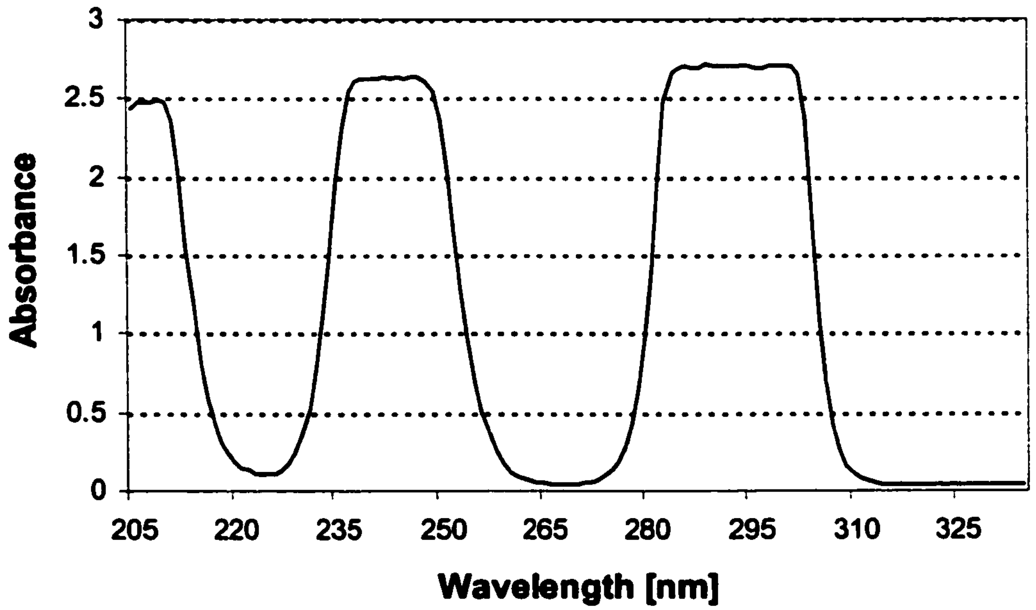

Figure 4.5-1: Unpolarized Absorption Spectrum of Ce:LLF.

The heavy doping concentration of the laser crystal being used required an imaged fluorescence experiment (see Figure 4.5-2) for the measurement of the absorption coefficient at 248 nm. In this experiment the exponential decrease in absorbed pump flux as a distance of pump depth can be measured from the exponential decrease of fluorescence as a function of depth. KrF pump pulses of 9 to 19 ns full-width half max duration are used to excite the fluorescence spectra. Multiple shots were taken, with pump fluxes ranging from 4 to 40 mJ/cm^2 . These pump fluxes are assumed to be unsaturated. With sufficiently higher pump fluxes in the saturation regime, it would be possible to estimate the absorption cross-section from the imaged fluorescence data. In our case, where we have used unsaturated pump flux intensities, we can only estimate a lower bound of the pump saturation flux which would still be consistent with the data.

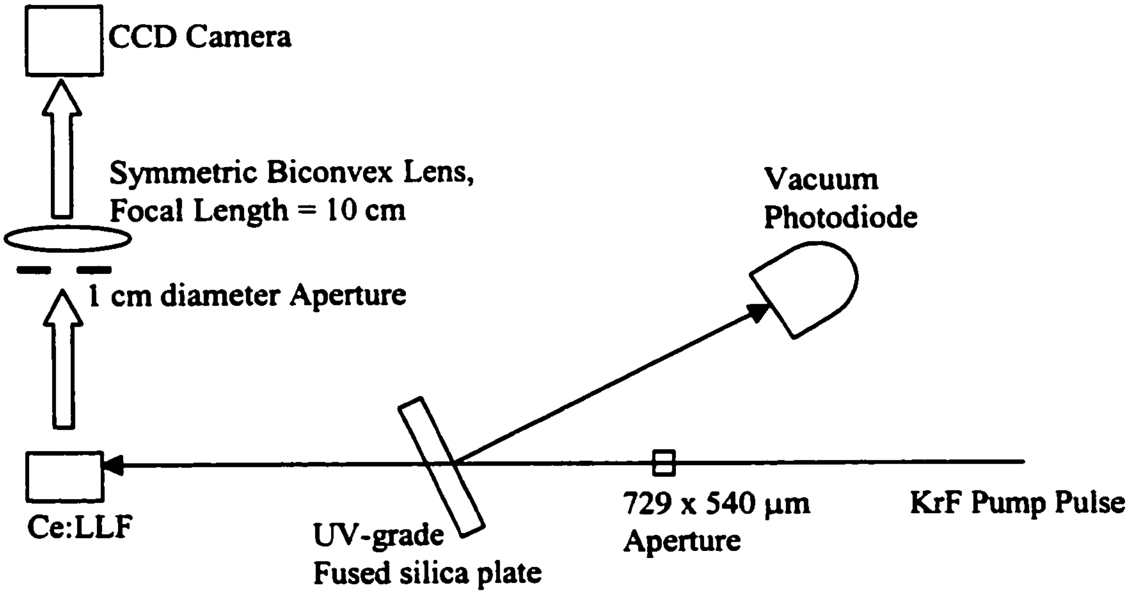

Figure 4.5-2: Setup for imaged fluorescence measurement

The biconvex lens is placed at twice the focal distance from the Ce:LLF crystal, and an equal distance from the CCD camera in order to give 1:1 imaging of the crystal face on the CCD camera. A 1 cm diameter aperture is added in front of the lens to reduce spherical aberrations. A ray tracing analysis of the imaging system was performed using a simple ray tracing code (Beam4). This analysis produced an estimate of the blur spot size (see Figure 4.5-3), which has a full width half-max of 20 µm. This is much smaller than the order of distances encountered in the fluorescence images (see Figures 4.5-6 and 4.5-7), although the blur spot has large wings out to ~ 45µm. These blur spots were considered small enough that the imaging system was considered to be perfect in the subsequent analysis.

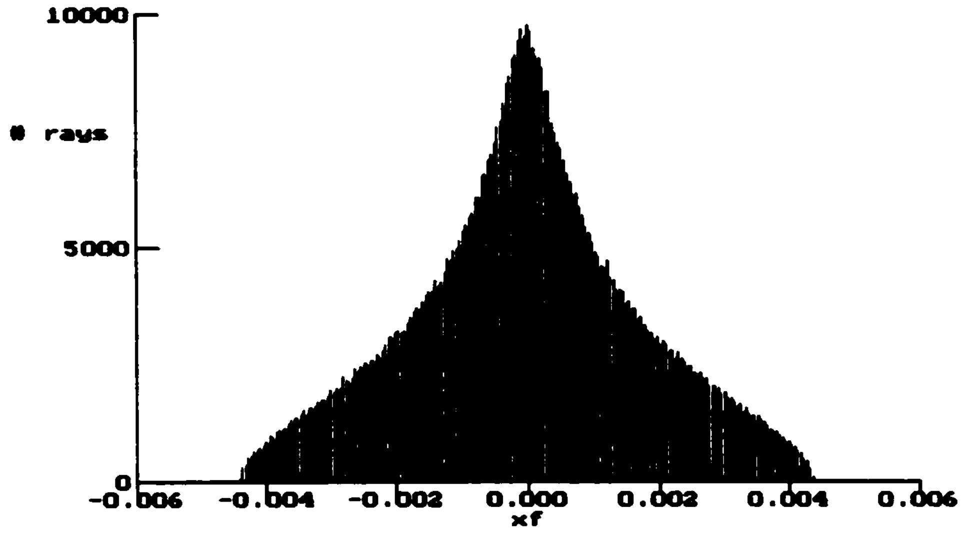

Figure 4.5-3: Blur analysis of imaging system used for the fluorescence measurement . (The horizontal scale is in cm)

The imaged fluorescence data is first averaged over vertical bands (over a height given by the width of the pump beam), and then sampled at every third pixel (in the horizontal direction). The CCD camera employed had three output amplifiers which were used successively for output columns giving minor (a few percent) variations in gain and offset for three neighbouring columns of data. By analyzing every third pixel column, these variations could be avoided, as shown below in Figure 4.5-4. The camera pixel sizes were 11.5 μm (horizontal) by 27 μm (vertical).

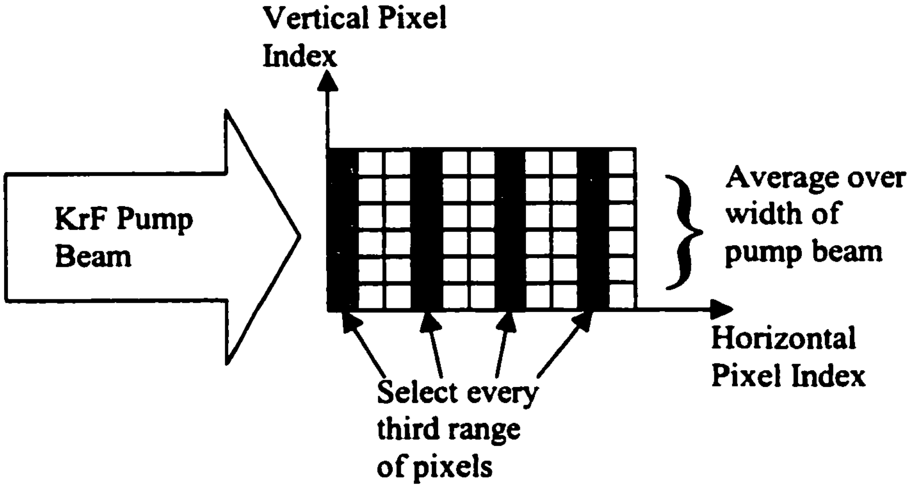

Figure 4.5-4: Analysis of imaged fluorescence data from CCD camera output.

The resulting curve is then normalized and compared with a normalized analytical solution for the spatial distribution of population inversion density. The analytical solution, given by equation 2-11 (repeated below as equation 4.5-1) is derived from Franz and Nodvik's[21] equations.

$$\Delta N(z) = N_T \cdot \left[1 - \frac{1}{1 - \exp(-\sigma_A \cdot N_T \cdot z) \cdot \left[1 - \exp\left(\frac{W_{P0}}{W_{SP}} \right) \right]} \right] \qquad (4.5\text{-}1)$$

where $\Delta N(z)$ is the population inversion density at pump depth z, N_T is the total dopant density, σ_A is the absorption cross section, W_{P0} is the input pump flux and W_{SP} is the saturation pump flux. In the limit of low fluxes and the absence of stimulated emission the fluorescence intensity is proportional to the population inversion density.

Two sets of data were taken (on different days); the results of both sets are summarized in Table 4.5-1. Both the sigma and pi polarizations were investigated. The sigma polarization, being the one used for amplification experiments, was investigated more thoroughly than the pi polarization. The absorption coefficient for sigma polarization (electric field of pump radiation perpendicular to the c axis of the crystal) was found to be between 10.4 and 14.35 cm^{-1}, with a weighted average value of 12.4 ± 0.4 cm^{-1}. A typical curve for the sigma polarization is shown in Figure 4.5-6. The pi polarized absorption coefficient (electric field of pump radiation parallel to the c axis of the crystal) was found to be in the range 21.25 to 22.25 cm^{-1} (see Figure 4.4-6) with a weighted average value of 21.9 ± 0.4 cm^{-1}. A typical curve fit for the pi polarization is shown in Figure 4.5-7. The above averages were calculated with a weighting factor equal to the inverse of the variance from the curve fit. This trend of a larger absorption coefficient for pi polarization at 248 nm is consistent with the results of Sarukura[7] and Rambaldi[8]. The sigma polarization, which allows for a deeper pumping depth at 248 nm is thus desirable for amplification.

All of the data presented in Table 4.5-1 were found to be consistent with unsaturated absorption curves. This corresponds to $W_{P0} < W_{SP}$ in equation 4.5-1. From such a condition, it is expected that the pump saturation flux is greater than 41 mJ/cm^2, the highest pump flux used in this experiment.

There is some discrepancy between the two data sets. The second set of data at pump fluxes of 36 to 41 mJ/cm^2 was taken at a different pumping spot on the crystal, and after a number of high pump flux amplification measurements had been made. These results are slightly different from the first set. The discrepancy is possibly due to spatial variations in the doping concentration or to solarization caused by high pump fluxes.

KrF Pump Flux [mJ/cm^2]	KrF FWHM Duration [ns]	Crystal Face Pumped	KrF Polarization	Alpha from best fit [cm^{-1}]	RMS Error
26.47	16.20	side	H (sigma)	11.65	0.0067
25.66	12.40	side	H (sigma)	11.65	0.0055
25.20	12.80	side	H (sigma)	11.40	0.0059
4.07	9.60	side	V (sigma)	11.25	0.0088
4.00	12.00	side	V (sigma)	10.45	0.0152
4.06	9.40	side	V (sigma)	10.70	0.0110
6.14	16.60	end	H (pi)	22.10	0.0093
6.27	16.40	end	H (pi)	22.25	0.0091
5.94	16.20	end	H (pi)	21.25	0.0086
41.44	17.6	side	H (sigma)	14.05	0.0029
40.09	14.6	side	H (sigma)	14.35	0.0039
40.42	15.6	side	H (sigma)	14.20	0.004
38.97	12.8	side	H (sigma)	14.20	0.0051
39.05	15.2	side	H (sigma)	13.45	0.0042
38.49	16.2	side	H (sigma)	13.25	0.0039
38.30	14.6	side	H (sigma)	13.70	0.0039
36.19	15.2	side	H (sigma)	13.25	0.0042
39.20	14.6	side	H (sigma)	13.60	0.0028
39.39	18.4	side	H (sigma)	14.00	0.0039

Table 4.5-1: Results of curve-fitting Imaged fluorescence Data

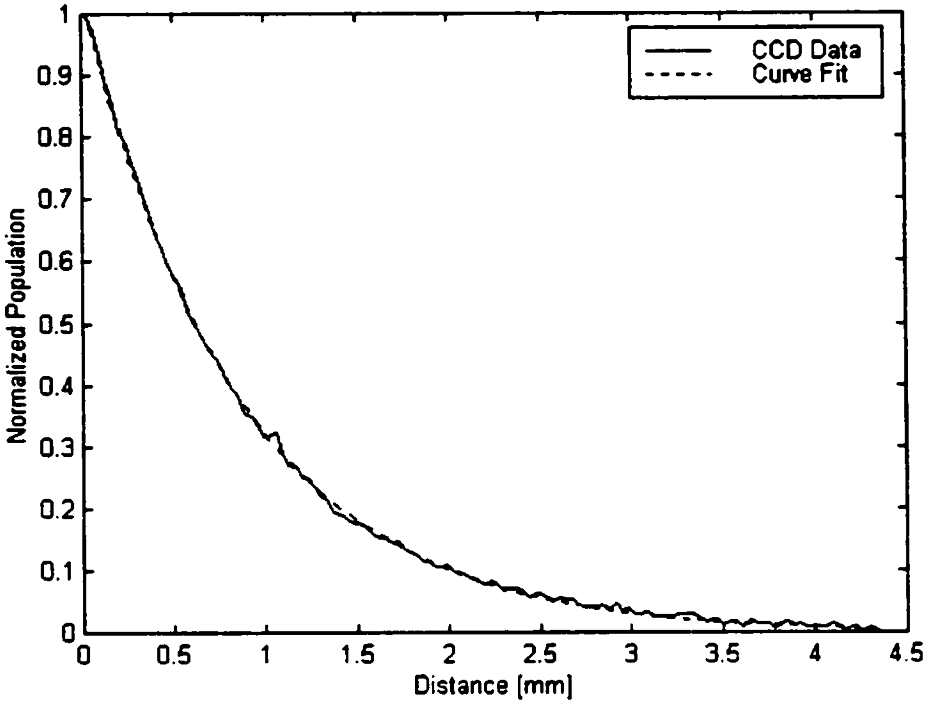

Figure 4.5-6: Sigma polarized absorption profile of Ce:LLF at 248 nm.

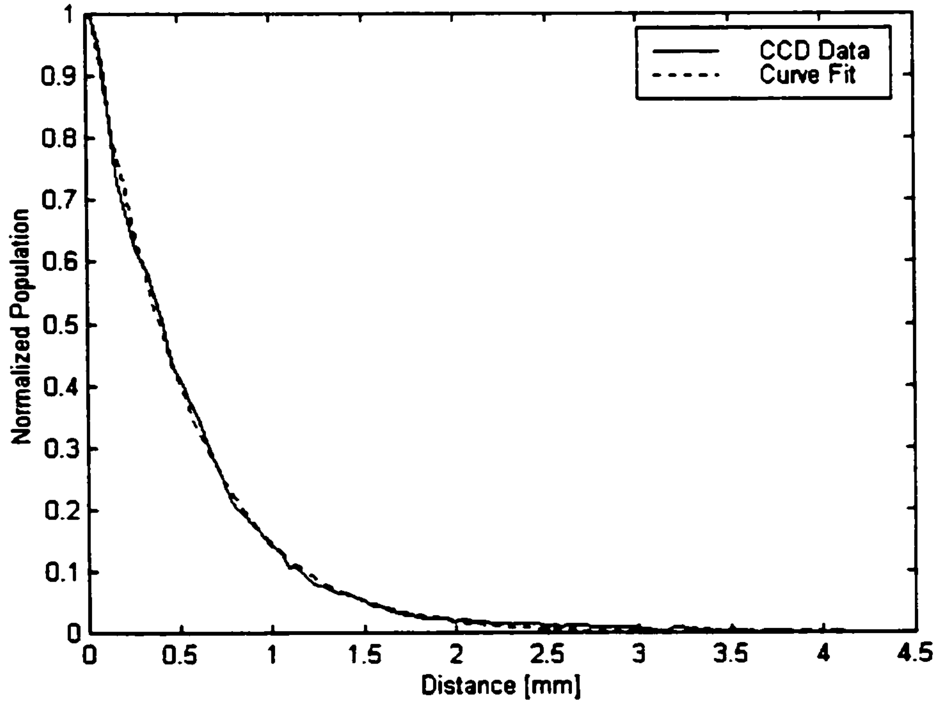

Figure 4.5-7: Pi polarized absorption profile of Ce:LLF at 248 nm.

4.6 Amplification in Ce^{3+}:$LiLuF_4$

A XeCl laser was used to investigate the gain of Ce:LLF at 308 nm, which has not been investigated very much to this point in the reported literature. The experimental layout for these measurements is shown in Figure 4.6-1.

The XeCl laser and the KrF pump laser (both of which were discharge pumped) were triggered from a common high voltage trigger generator. An optical delay system was used to delay the XeCl pulse with respect to the KrF pulse. Two different configurations of the optical delay system were tested. Timing jitter (due to the operation of spark gaps in the KrF and XeCl lasers) was typically in the range of 10 ns for a given day. This jitter caused a significant variation in the gains observed, due to injecting at different times along the excitation and decay of the population inversion.

Gain was investigated for various pump fluxes between approximately 70 and 1300 mJ/cm^2. A cylindrical lens was used to focus the pump beam to different line focal widths (0.3 mm, 0.6 mm and 1.7 mm) based on different distances from the crystal. The spatial profile of the seed beam was characterized by replacing the Ce:LLF crystal with a CCD camera and recording an image. This yielded a vertical FWHM of 270μm and a horizontal FWHM of 397 μm. Pump flux variation also occurred due to the decay of energy with shot number in a single gas fill of the KrF laser. The pump energy was measured with a pyroelectric calorimeter (PY19) connected to a HP 1727A oscilliscope (275 MHz bandwidth).

The KrF pump laser was polarized horizontally (perpendicular to the crystal's C axis) for all gain measurements, in order to ensure the largest possible absorption depth for 248 nm. The KrF laser was polarized with Brewster plates placed within the resonator (2 at each end). The XeCl pulses were polarized (after going through the optical delay) with a Wollaston Prism. The Ce:LLF laser crystal was inclined at an angle of 2^0 to the direction of propagation of the XeCl laser beam, in order to avoid feedback reflections from the end faces and reduce the possibility of lasing.

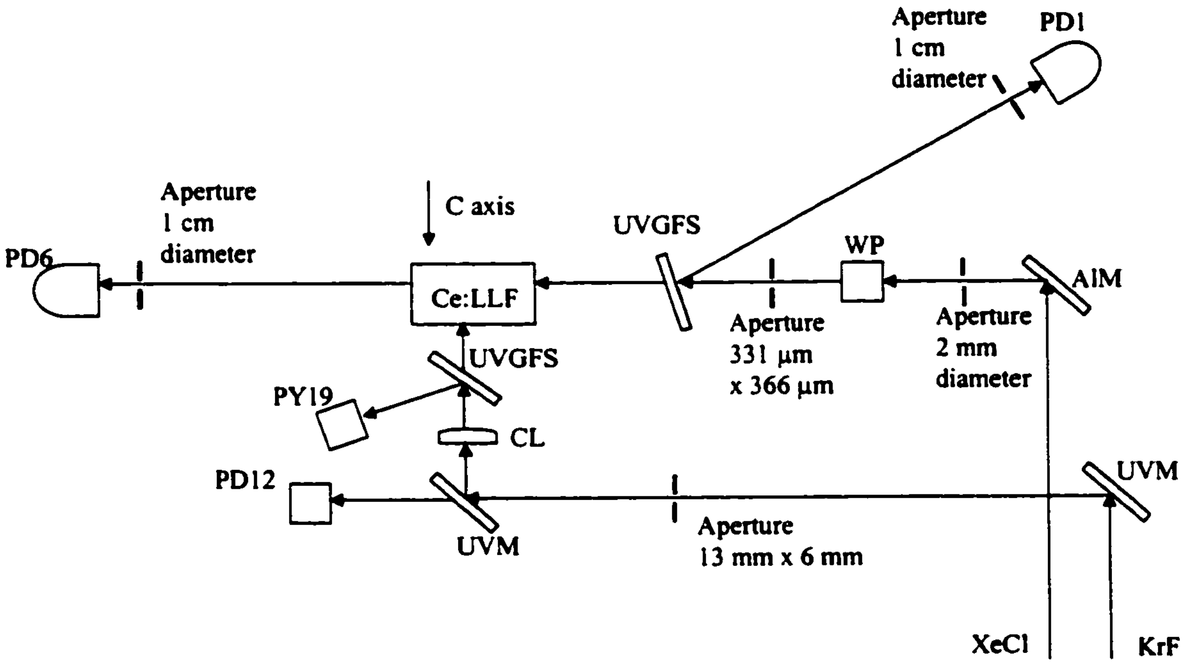

Figure 4.6-1: Amplification setup.
UVGFS = Ultraviolet Grade Fused Silica Plate; CL = Cylindrical Lens; UVM = UV Mirror (High Reflectance @ 248 nm); AlM = Aluminum Mirror; WP = Wollaston Polarizer; PY19 = Pyroelectric Calorimeter; PD1,PD6 = Vacuum Photodiodes; PD12 = Silicon Photodiode

Energy measurements of XeCl pulses were taken with Hamamatsu vacuum photodiodes. These photodiodes were calibrated for centred normal incidence beams to ensure a repeatable responsivity. Both photodiodes were fitted with UG5 filters (which remained in place for all experiments) to remove any stray visible light. The signals from

PD1 and PD6 were recorded on a Tektronix TDS360 oscilliscope, which was triggered externally by the rising edge of the pump pulse, as recorded on PD12 (in Figure 4.6-1). Each gain measurement was based on three laser shots (as shown in Figure 4.6-2): seed pulse only, pump pulse only and both pulses together.

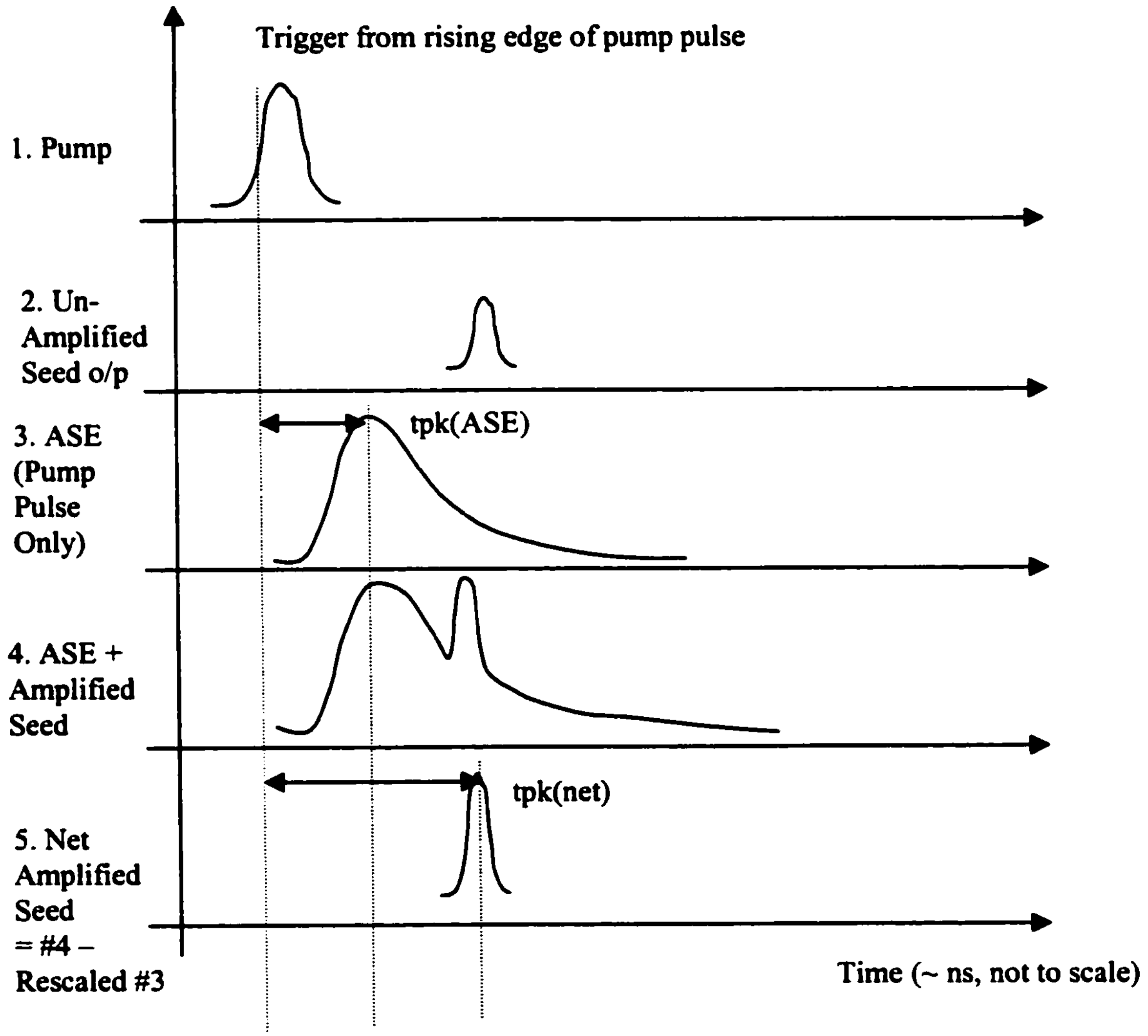

Figure 4.6-2: Laser Shots used for gain measurements.

The gain is calculated as the ratio of pumped output pulse energy to unpumped output seed energy, normalized to the input energies, as shown in equation 4.6-1. All pulse energy calculations from photodiode data were based on integrating the area in the photodiode pulse. For the amplified pulse (pumped output), it is necessary to subtract the ASE energy from the photodiode curve. The pump-only laser shot is used to generate an ASE curve which can be subtracted from the pumped output curve.

$$Gain = \frac{\left(\frac{Pumped\_OP\_Energy}{Pumped\_IP\_Energy}\right)}{\left(\frac{Unpumped\_OP\_Energy}{Unpumped\_IP\_Energy}\right)} \qquad (4.6\text{-}1)$$

While all times in the experiment are measured with respect to the trigger point, it is useful to calculate the delay from the time of maximum ASE, which in general corresponds to the time of maximum inversion density. This delay factor has the physical significance of indicating the degree of population decay. The delay from maximum ASE is calculated as the time of the peak of the seed pulse peak (after subtracting background ASE) minus the time of the ASE peak, as shown in equation 4.6-2.

$$Delay = tpk(net) - tpk(ASE) \qquad (4.6\text{-}2)$$

In this notation, a negative delay corresponds to injection of the seed pulse before the ASE peak. All further references to 'delay' (unless otherwise noted) refer to the quantity defined in equation 4.6-2. Noise spikes near the pulse peaks will tend to distort the delay calculation.

The initial configuration of the optical delay system produced injection times in the range of 0.6 to 29.6 ns delay past the ASE peak. This configuration was tested for

XeCl pulses polarized parallel and perpendicular to the crystal's c axis. The parallel polarization produced gains of 1.59 to 4.2. A summary of this gain data is shown in Table 4.6-1. No significant amount of gain was observed for the perpendicular polarized XeCl pulses. This is consistent with trends observed in various Ce:LLF laser tunability studies[18,17,19], where the perpendicular polarization has less gain than the parallel polarization at 308 nm.

Experiment	Pump Flux [mJ/cm^2]	Delay [ns]	Experimental Gain	Experimental Gain Coefficient [cm-1]
Jul 13 #1	226.7	0.6	4.2	1.9
Jul 13 #2	215.4	5.8	3.8	1.8
Jul 13 #3	204.0	8.0	3.9	1.8
Jul 15 #1	85.0	15.4	1.5	0.55
Jul 15 #2	73.7	9.4	1.8	0.78
Jul 15 #3	607.7	27.6	2.4	1.2
Jul 15 #4	575.7	29.6	2.1	1.0
Jul 15 #5	463.8	20.0	3.3	1.6
Jul 15 #6	447.8	16.6	3.8	1.8
Jul 15 #7	415.8	16.4	4.2	1.9
Jul 17 #1	191.9	12.8	2.8	1.4
Jul 17 #2	159.9	3.4	3.7	1.7
Jul 17 #3	127.9	5.0	2.8	1.4
Jul 17 #4	559.7	25.0	1.7	0.68
Jul 17 #5	543.7	24.8	1.7	0.71
Jul 17 #6	511.8	23.8	2.2	1.1
Jul 17 #7	511.8	22.6	2.2	1.0
Jul 18 #1	239.9	15.8	2.7	1.3
Jul 18 #2	239.9	15.8	2.4	1.2
Jul 18 #3	223.9	11.0	3.7	1.7
Jul 18 #4	223.9	15.6	2.7	1.3

Table 4.6-1: Initial set of gain data.

A second set of amplification data was taken with a revised configuration of the optical delay system, in which the injection time ranged from 14.2 ns in advance of the

ASE peak to 3 ns in delay of the ASE peak. A summary of these gains is shown in Table 4.6-2. Due to time constraints, this configuration was only tested for parallel polarized seed pulses. Pump fluxes of 97 to 1380 mJ/cm^2 produced gains of 1.4 to 35. The gains for both data sets are plotted in Figures 4.6-3 and 4.6-4. High flux pumping (1380 mJ/cm^2) with near-zero delay gives indications in the results of onset of absorption saturation. This effect is manifested by the flattening of the gain curve at high pump fluxes as show in Figure 4.6-4.

Experiment	Pump Flux [mJ/cm^2]	Delay [ns]	Experimental Gain	Experimental Gain Coefficient [cm-1]
Sep 12 #1	97.9	-14.2	1.2	0.27
Sep 12 #2	407.7	0.4	7.4	2.7
Sep 12 #3	399.6	-1.2	6.7	2.5
Sep 12 #4	391.4	-5.2	10.5	3.1
Sep 12 #5	375.1	-5.2	9.5	3.0
Sep 12 #6	277.3	1.4	7.9	2.8
Sep 14 #1	260.9	1.0	7.7	2.7
Sep 14 #2	203.9	-3.6	6.4	2.5
Sep 14 #3	575.2	-1.4	18.0	3.9
Sep 14 #4	552.2	-6.2	10.9	3.2
Sep 14 #5	1380.5	-3.8	33.3	4.7
Sep 14 #6	1311.5	-4.8	34.5	4.7
Sep 14 #7	1265.5	-6.6	30.9	4.6
Sep 15 #1	276.1	-7.8	3.8	1.8
Sep 15 #2	851.3	-3.2	29.2	4.5
Sep 15 #3	782.3	3.0	23.6	4.2
Sep 15 #4	690.3	-1.2	24.3	4.3

Table 4.6-2: Second set of gain data.

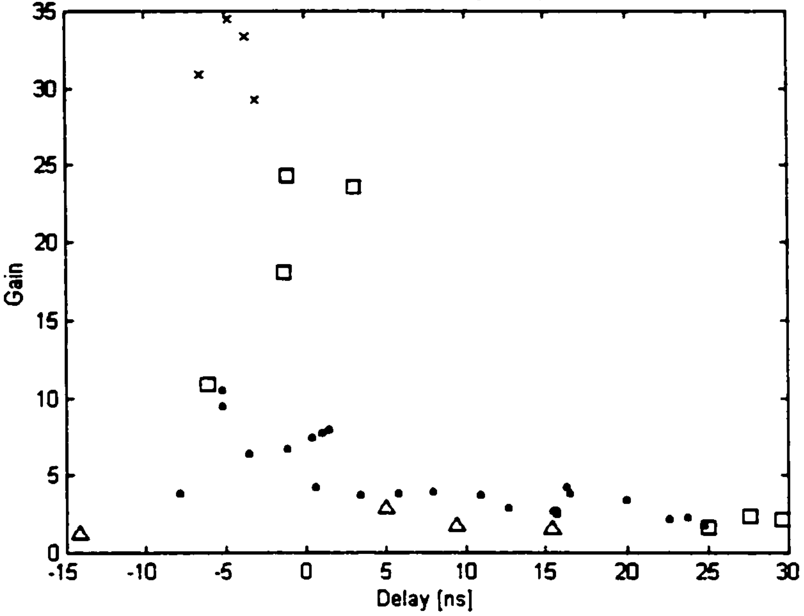

Figure 4.6-3: Ce:LLF Gain vs Delay, clustered by Pump Flux
Pump Flux Clusters: Δ <150 mJ/cm²; • 150 to 550 mJ/cm²;
□ 550 to 800 mJ/cm²; x >800J/cm²

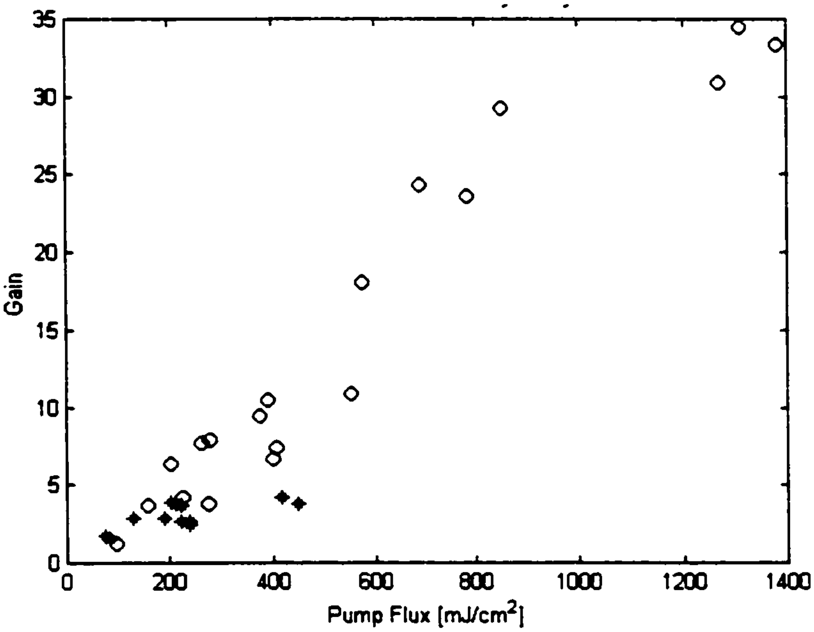

Figure 4.6-4: Ce:LLF Gain vs Pump Flux, clustered by Delay
Delay Clusters: * >5 ns; o < 5 ns

In all of the above gain measurements, a significant factor is the crystal's partial absorption of the seed beam at 308 nm. This absorption is likely in part due to the tail end of the absorption peak at 295 nm, and may also be partially due to excited state absorption or crystal defects. The Ce:LLF crystal absorbance in the vicinity of 308 nm is shown in Figure 4.6-5.

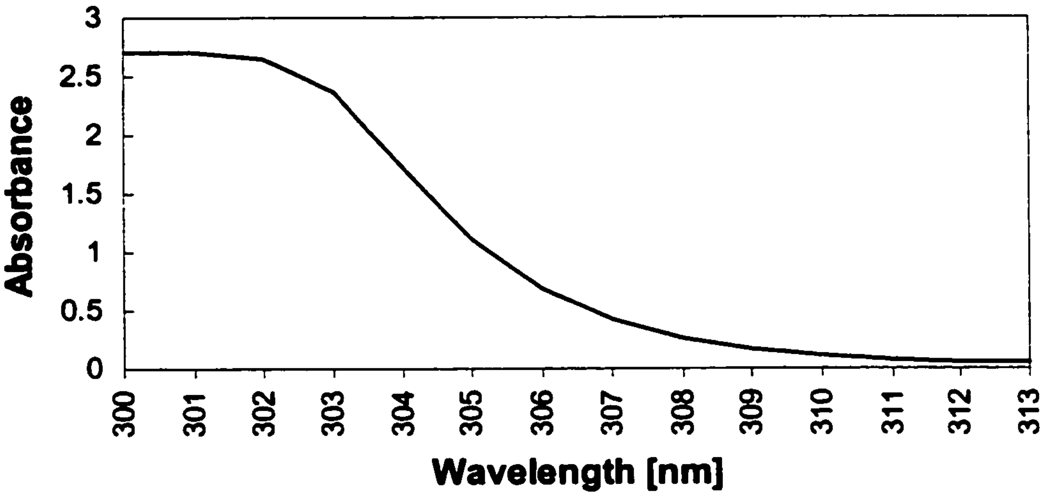

Figure 4.6-5: Ce:LLF unpolarized absorbance.

Chapter 5: Comparison of Gain Data with Simulation

As the time dependent experimental data for Ce:LLF is not readily described by analytical equations, simulations (using Cartesian geometry) were used to obtain estimates for the total population density, and thus also the gain and absorption cross-sections. The spontaneous lifetime (37.6 ns) and absorption coefficient (12.4 cm^{-1}) used for the simulation were based on weighted averages of the data presented in sections 4.4 and 4.5 Energies of the injected seed pulses were between 10 and 40 nJ in a beam cross-section area of $1.07 \times 10^{-3}\ \mathrm{cm}^2$, based on measurements in section 4.6.

A simple geometrical setup was used to model the gain experiments. The length of the crystal was 0.75 cm, which corresponds to the gain length (corresponding to the Z direction in Figure 5-1). The crystal is modeled as a single row of 19 cubic cells (see Figure 5-1), each having an edge length corresponding to the seed beam width of 397 µm. Each gain cell has a uniform population inversion density, which represents the average inversion density over the initial 397 µm of pumping depth.

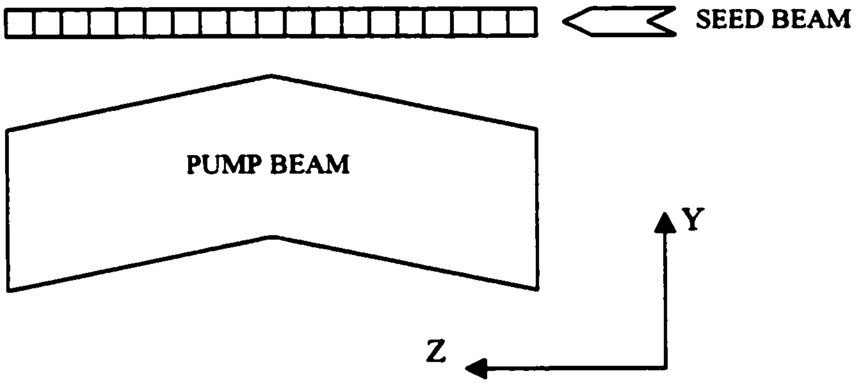

Figure 5-1: Geometrical Model for Ce:LLF Amplification Experiments

The time step Δt per cell required to obtain a cell edge Δz of 397 μm is determined by:

$$\Delta t = \frac{\Delta z}{v_{ph}} = \frac{\Delta z}{(c/n)} \tag{5-1}$$

where c is the vacuum speed of light, v_{ph} is the phase velocity in the crystal and n is the refractive index. In this case, the average of the ordinary (1.468) and extraordinary (1.494) indices was used as the refractive index (n_{AVG}=1.481). This yields a value of Δt=1.96 ps.

The solid acceptance angle for ASE is defined by the proportion of ASE generated in a given cell which propagates to the end of gain region. The simulation code is setup for a single acceptance angle, so this must be an average or effective value representing all gain cells. The solid acceptance angle for ASE acceptance is determined by the geometry of the pumped volume, as shown in Figure 5-2.

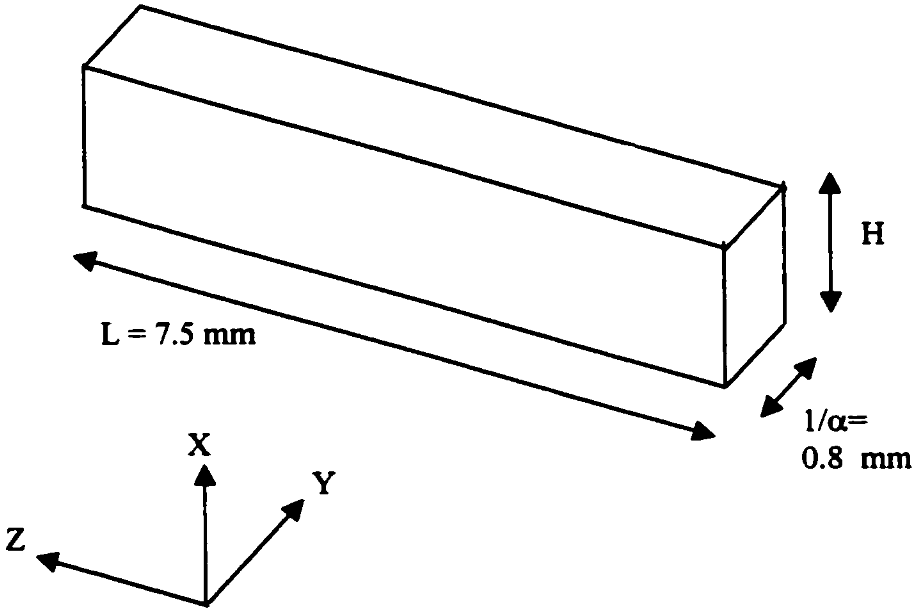

Figure 5-2: Geometry for determining solid angle of ASE acceptance.
H = 1.7mm, 0.6 mm or 0.3 mm.

The value of $1/\alpha$ = 0.8 mm corresponds to the effective pumping depth for sigma polarization, (for the complete sigma absorption curve, refer to Figure 4.5-6). The solid angle Ω is calculated according to equation 5-2.

$$\Omega = \frac{H \cdot (1/\alpha)}{(L)^2} \tag{5-2}$$

Three different values for the pump beam thickness H (corresponding to different focal conditions of the pump beam) were used: 1.7 mm, 0.6 mm and 0.3 mm. The corresponding solid angles are 0.024 sr, 0.0087 sr and 0.0043 sr. The same average solid angle applies in both directions for the simulation. The exact value of the solid angle does not make a significant difference in these simulation results. Multiplying the ASE solid angle by a factor of 10 or 0.1 resulted in at most a 2.5% change in simulated gain for the data from Sep 14 #1 and Sep 14 #7.

The absorption coefficient at 308 nm used in the simulation is taken from fig. 4.5-2, which shows a base-10 absorbance of 0.25 at 308 nm. This corresponds to an absorption coefficient of 0.77 cm^{-1} based on a crystal length of 0.75 cm.

The simulation parameters determined above are used to estimate the gain and absorption cross-sections and the dopant number density. A set of experimental gain results which have little noise and have a minimal amount of delay was used to estimate the cross-sections and number density. These numbers are then used to compare with the other experimental gain data.

Determination of the gain cross section requires a few iteration steps, as shown in Figure 5-3. Different values of the pump saturation flux (in steps of 5 mJ/cm^2) are tested. From the pump satruration flux and absorption coefficient (determined previously), the absorption cross section is calculated. Two data points (with high and low pump flux, representing high and low amounts of pump saturation) are used to estimate values for the gain cross section.

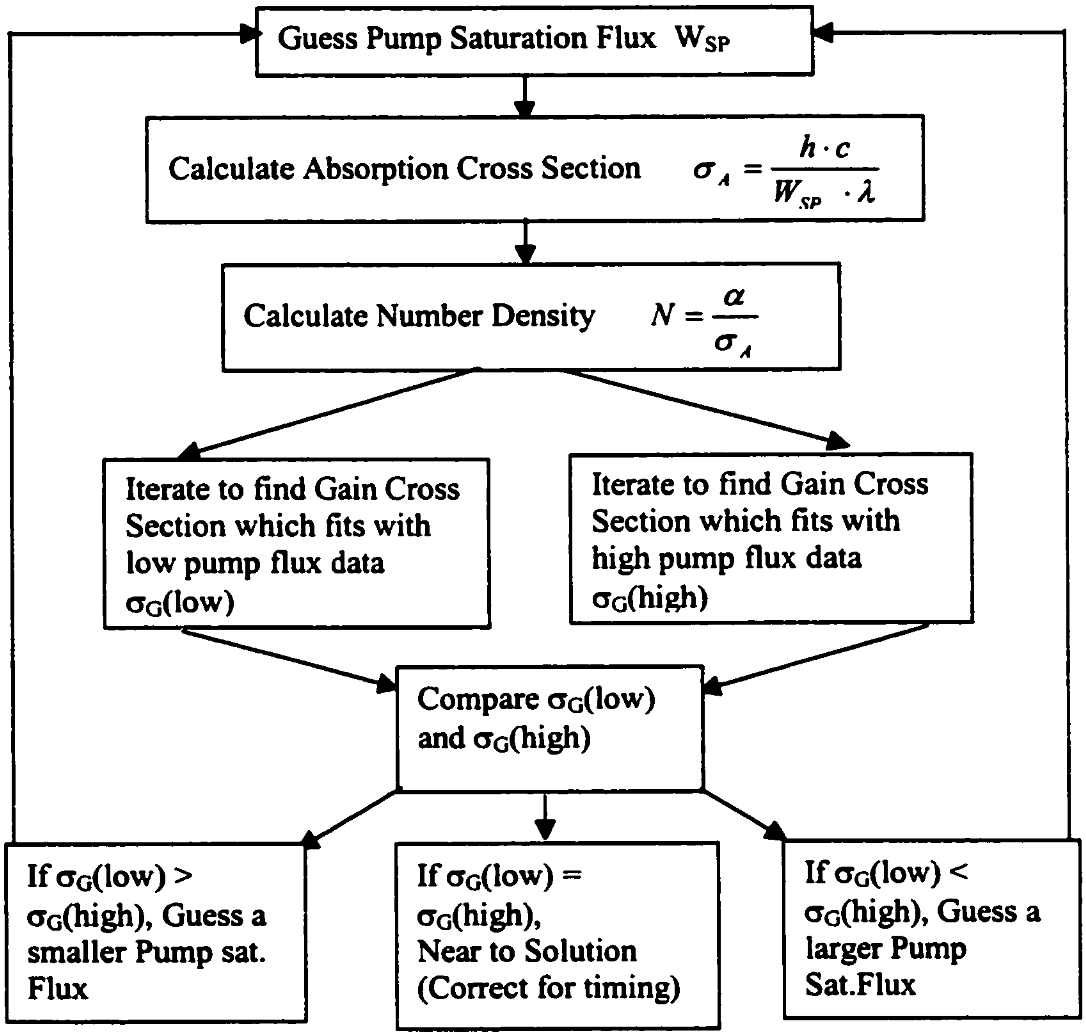

Figure 5-3: Iteration Procedure for determining laser simulation parameters.

The simulations are setup to match the delay factors (between seed injection and inversion peak) as calculated in equation 4.6-2. Because the time of maximum inversion density changes with pump flux (as shown in Figure 5-4), it is necessary to check that the injection time of the seed pulse is chosen to meet this condition for each set of parameters. This is determined by running a simulation with pump only.

The parameters determined from fitting five different pairs of the experimental gain data are summarized in Table 5-1. The median pump saturation flux from these five data pairs is used to determine a final parameter set (shown in Table 5-2), which coincides with the results for the Sep 12 #6 and Sep 14 #5 comparison. Error bars of the final parameter set are based doubling the discrepancies of the extreme values from the set of 5 data pairs.

A summary of reported cross sections and saturation fluxes for Cerium doped fluoride crystals (having 5d-4f transitions) is shown in Table 5-3. The gain cross section determined from the gain data at 308 nm is lower than Sarukura's reported value at 325 nm (determined from saturating the gain). Since the peak gain occurs at 325 nm and 308 nm has a gain coefficient considerably less than the peak value this is not surprising. In addition, the resulting low gain cross section may represent the effect of a branching ratio (quantum yield) below unity combined with a somewhat higher gain cross section. Sarukura and Dubinksii[7,13] have determined the quantum yield of Ce:LLF to be 0.88. This means that only 88% of the atoms excited to the first 5d energy level are available for the desired stimulated emission transition, reducing the effective population inversion density N_{EFF} in equation 5-3. Thus, to account for this effect and give the same experimentally measured gain, the gain cross section σ_G should be divided by 0.88, resulting in a value of 3.5×10^{-18} cm^2.

$$Gain = \exp(\sigma_G \cdot N_{EFF} \cdot z) \qquad (5\text{-}3)$$

It is also possible that excited state absorption to higher 5d states (not accounted for in the simulation) may counteract the effect of a higher gain cross section. A larger gain cross section may exist for slightly longer wavelengths (e.g. 312 nm). Absorption cross sections for pump laser radiation for Ce:LLF are not well characterized in the literature. The absorption cross section determined from the gain data is similar to those reported for Ce:LiSAF and Ce:LiCAF.

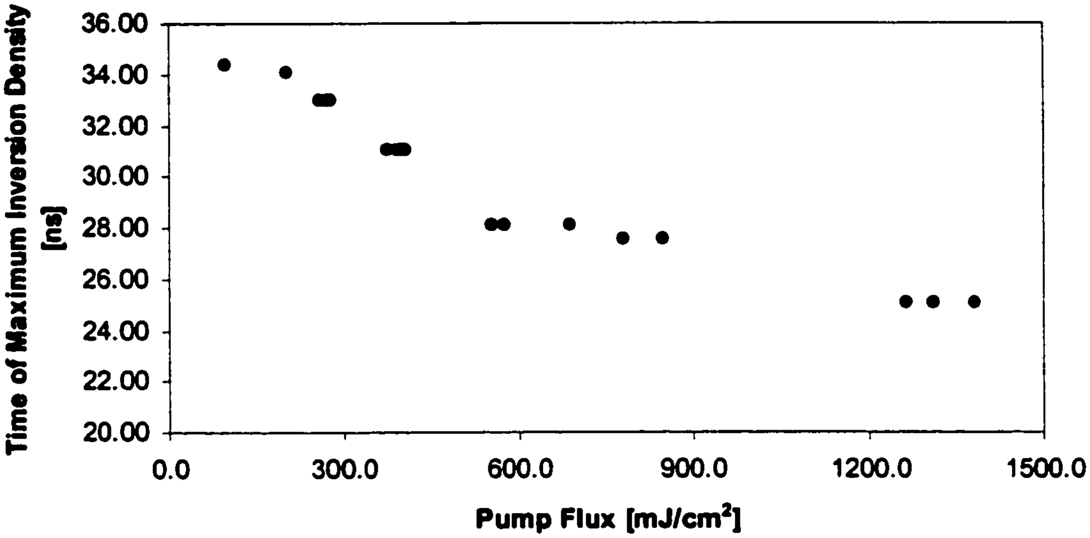

Figure 5-4: Variation of time of inversion peak (relative to the start of pump pulse) with pump flux. The time at the peak of the pump pulse is 24 ns.

Data Pair	Pump Saturation Flux [mJ/cm^2]	Absorption Cross Section [cm^2]	Number Density [cm^{-3}]	Gain Cross Section [cm^2]
Sep 12 #6, Sep 14 #7	125	6.4×10^{-18}	$1.9 \times 10^{+18}$	3.0×10^{-18}
Sep 14 #2, Sep 14 #5	90	8.9×10^{-18}	$1.4 \times 10^{+18}$	3.7×10^{-18}
Sep 14 #1, Sep 14 #6	115	7.0×10^{-18}	$1.8 \times 10^{+18}$	3.1×10^{-18}
Sep 12 #6, Sep 14 #5	110	7.3×10^{-18}	$1.7 \times 10^{+18}$	3.1×10^{-18}
Sep 14 #2, Sep 14 #7	95	8.4×10^{-18}	$1.5 \times 10^{+18}$	3.6×10^{-18}

Table 5-1: Best fit parameters for different data pairs.

Gain Cross Section [cm^2] @ 308 nm, Pi Polarization	(3.1 ± 1.2) x 10^{-18}
Absorption Cross Section [cm^2] @ 248 nm, Sigma Polarization	(7.3 ± 3.2) x 10^{-18}
Total Ground State Starting Number Density [cm^{-3}]	(1.7 ± 0.6) x 10^{18}
Pump Saturation Flux [mJ/cm^2] @ 248 nm, Sigma Polarization	110 ± 40
Gain Saturation Flux [mJ/cm^2] @ 308 nm, Pi Polarization	208 ± 131

Table 5-2: Laser simulation parameters for Ce:LLF

Laser Crystal	Pump Wavelength [nm]	Absorption Cross Section [cm^2]	Pump Saturation Flux [mJ/cm^2]	Laser Wavelength [nm]	Gain Cross Section [cm^2]	Gain Saturation Flux [mJ/cm^2]	Source
Ce:LiSAF	266	7.3E-18	102	290	9.5E-18	72	Marshall
Ce:LiCAF	266	7.5E-18	100	290	9.6E-18	71	Marshall
Ce:YLF				325	7.6E-18	81	Ehrlich
Ce:LLF				325	1.E-17	50	Sarukura

Table 5-3: Comparison of reported cross sections and saturation fluxes for Cerium-doped fluoride crystals. Data is taken from Marshall[5], Ehrlich[6] and Sarukura[7].

The number density of active Ce^{3+} ions determined from the gain data can be compared to an estimate based on the specified atomic melt concentration of C_{Melt} = 0.5%, as shown in equation 5-4 (N_{Av} is Avogadro's Number).

$$ND_{Melt} = \frac{C_{Melt} \cdot \rho \cdot N_{Av}}{M} \tag{5-4}$$

Using the density for $LiLuF_4$ of $\rho = 6.19$ g/cm^3 and the molar mass of M = 257.941 g/mol, the resulting number density in the melt is $ND_{Melt} = 7.23 \times 10^{19}$ cm^{-3}. To estimate the crystal's number density, the melt number density must be multiplied by the segregation coefficient, which is the ratio of crystal concentration to melt concentration. A segregation coefficient of 0.066 was reported by Ranieri[30] et al for Ce:LLF. Thus, based on a melt concentration of 0.5%, a crystal number density of 4.77 x 10^{18} cm^{-3} can be calculated. This is roughly 2.8 times the value determined from the gain data. There are a number of possible explanations for this apparent discrepancy. It is possible that nonuniformities exist in the crystal growth process. It is also possible that some of the Cerium ions present in the crystal do not contribute to the gain process.

Based on the number density shown in Table 5-2, and the absorption coefficient for pi polarized light at 248 nm (21.9 cm^{-1} determined in section 4.5) the pi polarized absorption cross section at 248 nm would be $(1.3 \pm 0.7) \times 10^{-17}$ cm^2.

Comparisons of simulation with experimental data are shown in Tables 5-4a and 5-4b. A scatter plot with these comparisons (for both the July and September data) is shown in Figure 5-5

Experiment	Pump Flux [mJ/cm^2]	Delay [ns]	Experimental Gain Coefficient [cm^{-1}]	Simulated Gain Coeffciient [cm^{-1}]
Sep 12 #1	97.9	-14.2	0.27	0.76
Sep 12 #2	407.7	0.4	2.7	3.5
Sep 12 #3	399.6	-1.2	2.5	3.4
Sep 12 #4	391.4	-5.2	3.1	3.2
Sep 12 #5	375.1	-5.2	3.0	3.1
Sep 12 #6	277.3	1.4	2.8	2.8
Sep 14 #1	260.9	1	2.7	2.7
Sep 14 #2	203.9	-3.6	2.5	2.2
Sep 14 #3	575.2	-1.4	3.9	4.0
Sep 14 #4	552.2	-6.2	3.2	3.5
Sep 14 #5	1380.5	-3.8	4.7	4.7
Sep 14 #6	1311.5	-4.8	4.7	4.6
Sep 14 #7	1265.5	-6.6	4.6	4.3
Sep 15 #1	276.1	-7.8	1.8	2.5
Sep 15 #2	851.3	-3.2	4.5	4.4
Sep 15 #3	782.3	3	4.2	4.4
Sep 15 #4	690.3	-1.2	4.3	4.2

Table 5-4a: Comparison of experimental and simulated gain for September data

For the September data (which have near zero delays), the simulations are generally close to the experimental results. The July data (with larger delays) has somewhat more deviation from the simulations, especially for pump fluxes greater than 400 mJ/cm^2. There are a number of possible explanations for the discrepancies. Experimental error in the form of noise pickup by the oscilliscope will distort both the gain and delay determined from the data. Slightly different alignments of pump and seed beams on different days would change the overlap of the seed beam with the inverted volume, leading to slightly different gain characteristics on different days.

Experiment	Pump Flux [mJ/cm^2]	Delay [ns]	Experimental Gain Coefficient [cm^{-1}]	Simulated Gain Coeffciient [cm^{-1}]
Jul 13 #1	226.7	0.6	1.9	2.4
Jul 13 #2	215.4	5.8	1.8	2.1
Jul 13 #3	204.0	8	1.8	2.0
Jul 15 #1	85.0	15.4	0.55	0.77
Jul 15 #2	73.7	9.4	0.78	0.79
Jul 15 #3	607.7	27.6	1.2	2.4
Jul 15 #4	575.7	29.6	1.0	2.3
Jul 15 #5	463.8	20	1.6	2.5
Jul 15 #6	447.8	16.6	1.8	2.7
Jul 15 #7	415.8	16.4	1.9	2.6
Jul 17 #1	191.9	12.8	1.4	1.7
Jul 17 #2	159.9	3.4	1.7	1.8
Jul 17 #3	127.9	5	1.4	1.5
Jul 17 #4	559.7	25	0.68	2.5
Jul 17 #5	543.7	24.8	0.71	2.4
Jul 17 #6	511.8	23.8	1.1	2.4
Jul 17 #7	511.8	22.6	1.0	2.5
Jul 18 #1	239.9	15.8	1.3	1.8
Jul 18 #2	239.9	15.8	1.2	1.8
Jul 18 #3	223.9	11	1.7	1.9
Jul 18 #4	223.9	15.6	1.3	1.7

Table 5-4b: Comparison of experimental and simulated gain for July data

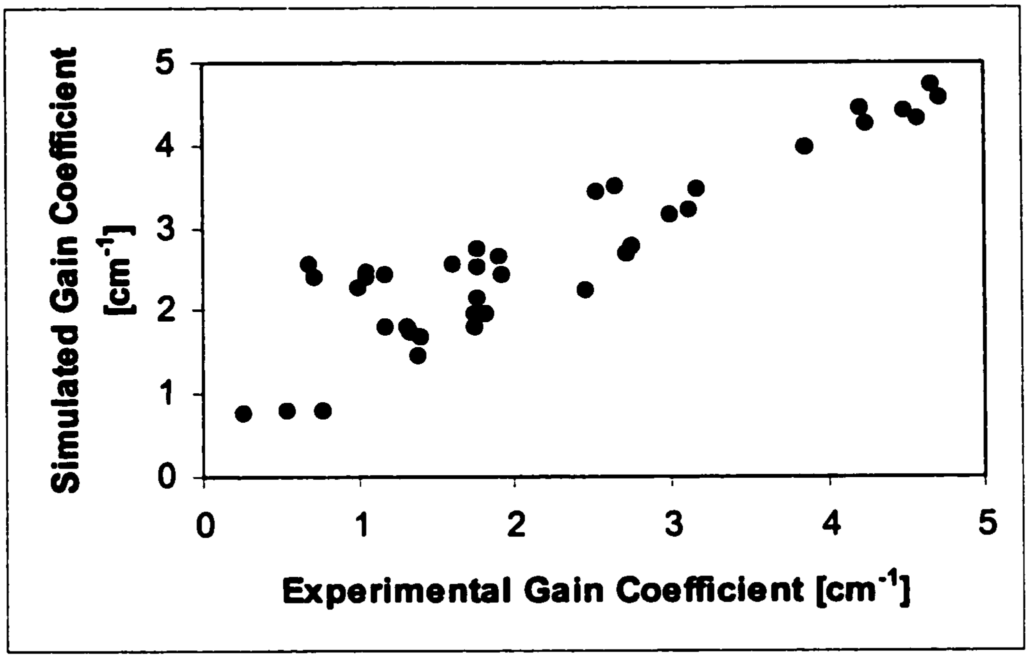

Figure 5-5: Comparison of simulated and experimental gain coefficients.

Chapter 6: Design Simulations for a short pulse amplifier system.

The amplification of femtosecond-duration laser pulses is typically accomplished with Chirped Pulse Amplification (CPA). This process initially stretches the pulse by a factor of 1000 to 10,000 which results in a pulse duration of 10's to 100's of picoseconds. The stretching of the pulse will reduce the peak intensity, and thus reduce the possibility of optical nonlinear effects (which could produce damage) in the amplifier crystal. After amplification, the pulse is recompressed to near its original duration. CPA is commonly used in Ti:Sapphire amplifiers. It has also recently been demonstrated for ultraviolet pulses with a wavelength of 290 nm using a Ce:LiCAF laser crystal[34]. In this Chapter, a design for a short pulse ultraviolet amplifier system based on Ce:LLF is proposed and the predicted performance calculated numerically.

Splitting a laser amplifier into multiple stages allows for ASE from each stage to be reduced significantly by spatial filters and saturable absorbers. A multiple stage amplifier also allows for different crystal concentrations and sizes to produce different gain characteristics. The final stages (power amplifiers) of the system often make use of larger beam cross sections to reduce the input laser flux and reduce gain saturation.

Design simulations were performed for a three stage Ce:LLF amplifier, with each stage being pumped by a portion of the output of a multi stage KrF laser system[35] (developed at the University of Alberta), having a Gaussian full width half max duration of 20 ns and a wavelength of 248 nm. The injected pulse is assumed to be at the wavelength of 325 nm and has a Gaussian full width half max duration of 250 ps (after going through a pulse stretcher). The wavelength of 325 nm is selected in order to minimize any effects of the 295 nm absorption band. Input pulses at 325 nm could be obtained from a Ce:LLF laser, third harmonic of a Ti:Sapphire laser, second harmonic of a dye laser, or a frequency mixed combination of lasers. It was assumed that this choice of wavelength, along with a crystal of sufficiently good quality would result in no absorption of the seed energy. The gain cross-section at 325 nm wavelength is taken to be 1 x 10^{-17} cm^2, the value reported by Sarukura and Dubinskii[7]. The maximum limiting amplifier efficiency for this pump and seed wavelength is 76%. The absorption cross section at 248 nm is taken to be 7.3 x 10^{-18} cm^2, the value determined in Chapter 5. The spontaneous lifetime is taken as 37.6 ns, as determined in section 4.4. An ideal 4 level laser system is assumed in the simulations.

All laser beams are assumed to propagate without divergence and have normal incidence on crystal faces. In the simulations, the laser beams are assumed to be uniformly distributed and have square or rectangular cross-sectional areas which correspond exactly to the crystal surface which they are incident upon.

Two different types of ASE are considered in the simulations. Large angle ASE depletes the inversion density in the laser crystal, but does not reflect back for a second pass. Small angle ASE represents the portion of ASE that propagates with the main laser beam and will be reflected as many times as the main laser beam. Part of the small angle ASE from one stage is injected into the following stage where it is amplified and is combined with newly generated small angle ASE. The large solid acceptance angle for ASE is determined in a manner similar to that used in Chapter 5. For stage 1, this is calculated as the crystal's cross sectional area divided by length squared, resulting in 0.25 sr.

Stages 2 and 3 employ laser crystals with large cross sectional areas (4 cm^2 and 16 cm^2 respectively). This geometry could potentially lead to lasing in the transverse direction. A number of techniques can be used to alleviate this possibility. Segmentation of the laser crystal into cubes of 1 cm x 1 cm x 1 cm, would limit the possible transverse gain length to 1 cm. Based on this segmentation scheme, the large solid acceptance angle for stages 2 and 3 is assumed to be 1.0 sr. The amplifier segments could be separated with absorbing strips made of black anodized aluminum. Along with this segmentation, it is important to consider the small signal gain that could be present in the gain medium. The small signal gain should be kept small to reduce the possibility of lasing. This will limit the pump energy that can be used in the amplifier systems. In each of the 3 amplifier stages considered, the simulations were also run to test for the small signal gain. The small signal gain simulations were run for an injected signal energy of 1 nJ, with the results summarized in Table 6-1. These simulations were run with and without the presence of ASE. Lasing within the laser crystals can also be alleviated by using antireflective coatings on the end faces. The small signal gains calculated for the amplifier stages are in the range of 7.8 to 12.1, which should be low enough to prevent lasing off the end faces (even if end face reflection is as high as 4%).

The small solid acceptance angle for ASE is determined by considering the double-pass distance between the laser crystal and the end mirror. In these designs, the distance between the laser crystal and the end mirror is assumed to be 50 cm. The small solid angle for ASE is calculated from the beam cross sectional area divided by the square of the double-pass distance between the laser crystal and the end mirror (100 cm). The small solid angles assumed in the design simulations were 2.5×10^{-5} sr for stage 1,

4.0×10^{-4} sr for stage 2 and 1.6×10^{-3} sr for stage 3. A portion of the small angle ASE produced in one stage is injected into the next stage. The temporal distribution of the injected ASE is approximated as a Gaussian, with FWHM durations of 14 ns for stage 2 and 10 ns for stage 3. These durations are based on the output characteristics of the previous stage. The injected ASE pulses are assumed to reach their peaks at the same time as the signal pulses.

A diagram of the proposed implementation of the three stages is shown in Figure 6-1. Each amplifier stage is pumped from two sides to increase inversion uniformity. Relay imaging and telescope optics can be used to couple the pulse from one stage to the next. The number of passes in each stage is limited to two in order to limit the buildup of small angle ASE.

A summary of the simulation parameters for the three amplifier stages is shown in Table 6-1. The total pump energy of 5.1 J is divided among the three stages in a 2%:20%:78% ratio. The energy extraction efficiency shown in Table 6-1 is calculated as the ratio of extracted energy to total input pump energy. The energy extraction efficiency increases with each successive amplifier stage, corresponding to increasing gain saturation. The three stage system extracts roughly 22% of the total pump energy, slightly less than one third of the quantum limit.

It is assumed that the use of saturable absorbers between amplifier stages will result in a transmission of 70% of the signal energy and only 10% of the small angle ASE energy. Similar performance has been observed for the dye BBQ[36] when used as a saturable absorber at a wavelength of 308 nm. The BBQ dye has an abosorption saturation flux of 3 mJ/cm^2 and a sufficiently broad absorption band that it could be considered for use as a saturable absorber at wavelengths of 310 nm to 325 nm. Other dyes exist with similar characteristics. The amplifier layout shown in fgure 6-1 assumes that the ASE levels after the third amplifier stage are low enough that no saturable absorber is needed after the third stage.

Spatial filters would be inserted after every amplifier stage to reduce the amount of ASE being coupled from one stage to the next. The spatial filter after the first stage is assumed to pass an angle corresponding to twice the main beam's divergence angle (1.3×10^{-4} rad). This is based on the assumption of a diffraction main lobe corresponding to the main beam's divergence angle, with the first diffraction sidelobe (corresponding to twice the main beam's divergence angle) being transmitted as well. For the small angle ASE, the transmittance of the first spatial filter is calculated from the ratio of solid angle passed to full solid angle of the beam. The solid angle passed can be calculated by taking the square of the plane angle passed (2.6×10^{-4} rad), resulting in 6.8×10^{-8} sr. Dividing this by the full solid angle of the small angle ASE beam (2.5×10^{-5} sr) results in a transmittance of 2.7×10^{-4}. When combined with a saturable absorber (having a transmittance of 10% for small angle ASE), an overall transmittance of 2.7×10^{-5} results for the small angle ASE being coupled from stage 1 to stage 2.

It is assumed that the small angle ASE after stages 2 and 3 is mostly confined to the dimensions of the main beam. Thus, all of the small angle ASE after stages 2 and 3 is assumed to be passed by spatial filters (which are still needed to block the propagation of large angle ASE). Electrooptic shutters could also be used to remove the portion of ASE which precedes or lags the main pulse by more than one nanosecond.

	Stage 1	Stage 2	Stage 3
Number Density of Ce^{3+}	5×10^{17} cm^{-3}	1×10^{18} cm^{-3}	1×10^{18} cm^{-3}
Input Seed Energy	20 μJ	1.4 mJ	57 mJ
Seed Beam Area	0.5 cm x 0.5 cm	2 cm x 2 cm	4 cm x 4 cm
Input ASE energy	0	454 pJ	23 μJ
Input Pump Energy Per Side	50 mJ	500 mJ	2 J
Pump Geometry	Transverse	Longitudinal	Longitudinal
Pump Beam Area	1 cm x 0.5 cm	2 cm x 2 cm	4 cm x 4 cm
Gain Length	1 cm	1 cm	1 cm
% Absorbed Pump Energy	68%	96%	96%
Pass 1 Signal Energy	217 μJ	12 mJ	353 mJ
Pass1 ASE Energy	222 nJ	42 μJ	511 μJ
Pass 2 Signal Energy	2.1 mJ	82 mJ	1.1 J
Pass 2 ASE Energy	1.7 μJ	234 μJ	2.1 mJ
Energy Extraction Efficiency	2.1%	8.1%	27%
Signal to ASE Pk. Power Ratio [For stretched pulse]	8.5×10^4	1.9×10^4	1.7×10^4
Number of Gain Cells (x)	5	8	16
Number of Gain Cells (y)	5	8	16
Number of Gain Cells (z)	10	4	4
Small Signal Gain (without ASE)	11.7	12.1	9.2
Small Signal Gain (with ASE)	10.9	9.1	7.8

Table 6-1: Summary of Amplifier Stages. Small signal gains are calculated for a single pass through the 1 cm gain length.

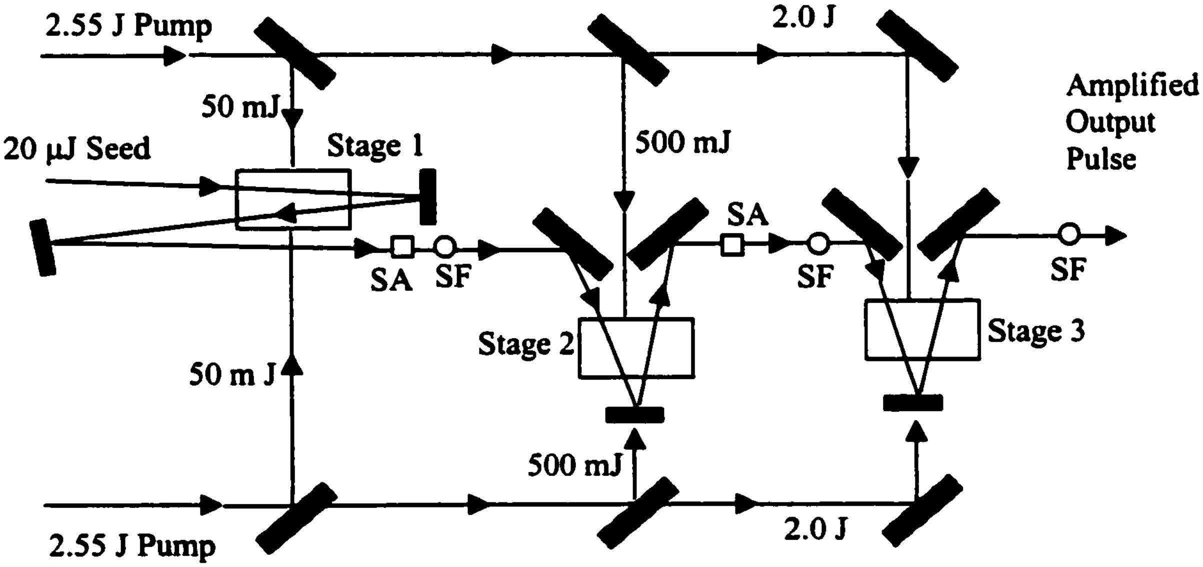

Figure 6-1: Three Stage Amplifier System. SF=Spatial Filter. SA=Saturable Absorber. The input seed pulse is assumed to come from a pulse stretcher and the output pulse is assumed to be sent through a pulse compressor.

A diagram of the layout for the first amplifier stage is shown in Figure 6-2. The crystal used is 1 cm in length and 0.5 cm x 0.5 cm in cross-section. A concentration of active Ce^{3+} ions of 5.0×10^{17} cm^{-3} is chosen to achieve a near-uniform inversion density. This corresponds to a small signal absorption coefficient of 3.7 cm^{-1} and a 1/e pump absorption length of 0.27 cm. The pump flux per side is 100 mJ/cm^2, close to the pump saturation flux. The overall double pass gain was 105, producing 2.1 mJ of output energy for an input energy of 20 μJ. The temporal profile of the population inversion density for stage 1 is shown in Figure 6-3, while the temporal profiles of the pump, ASE and signal are shown in Figure 6-4. The two passes of the injected signal experience gains near to the small signal value, although the second pass is slightly more saturated and causes a slight depletion of the inversion.

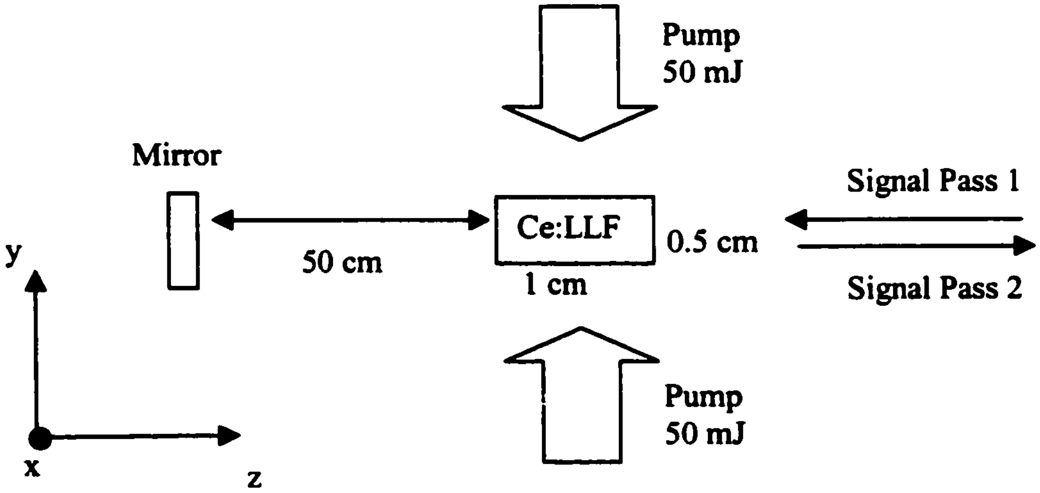

Figure 6-2: Ce:LLF Stage 1 Amplifier

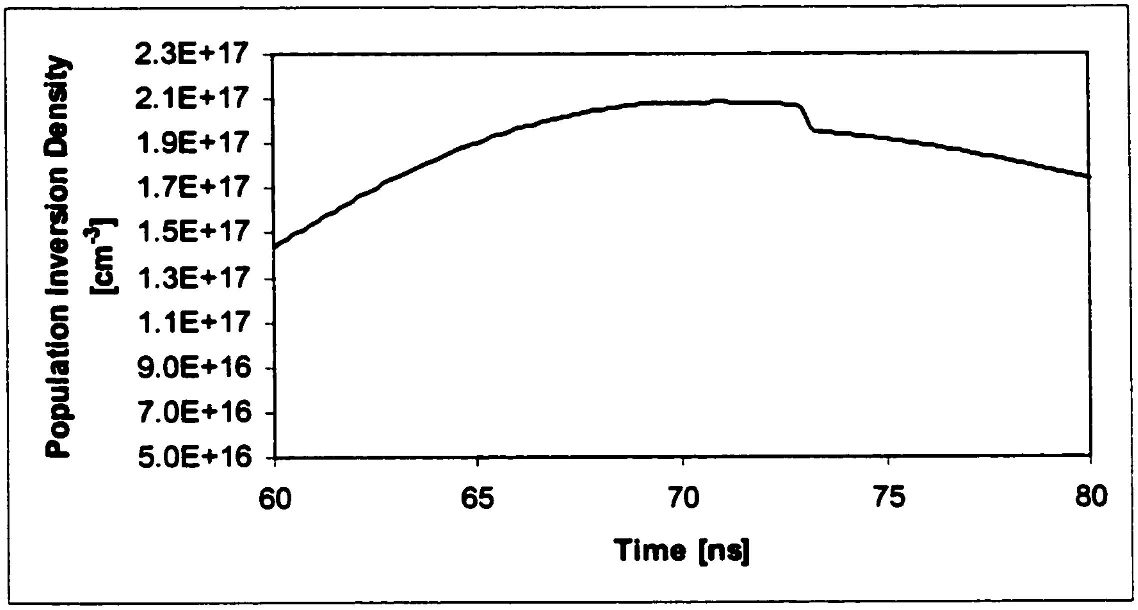

Figure 6-3: Population Inversion Density for Stage 1, calculated for the center of the laser crystal.

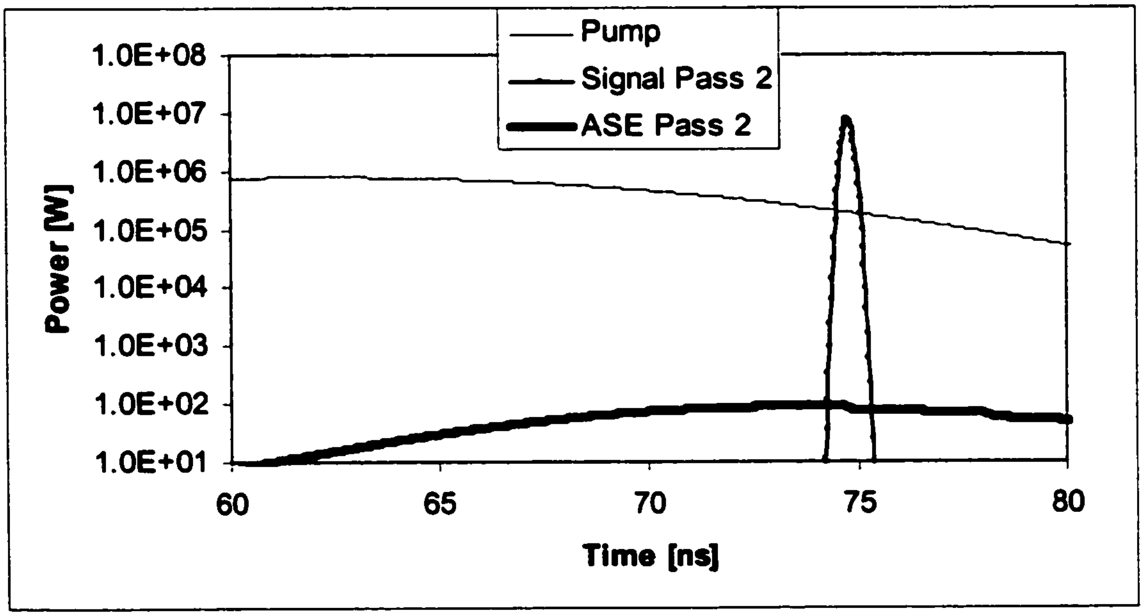

Figure 6-4: Beam Powers for Stage 1.
The injected pump pulse reaches its peak at 60 ns, while the injected signal and ASE are delayed by 8 ns. All beams are plotted for the exit of the second pass.

One disadvantage of transverse pumping is the possibility of a nonuniform population inversion. This can distort the spatial profile of the laser beam. Generally, a lower doping concentration will lead to a more uniform inversion profile. Saturating the absorption of pump energy will also tend to produce a more uniform inversion. An example of the transverse inversion profile for the stage 1 amplifier is shown in Figure 6-5.

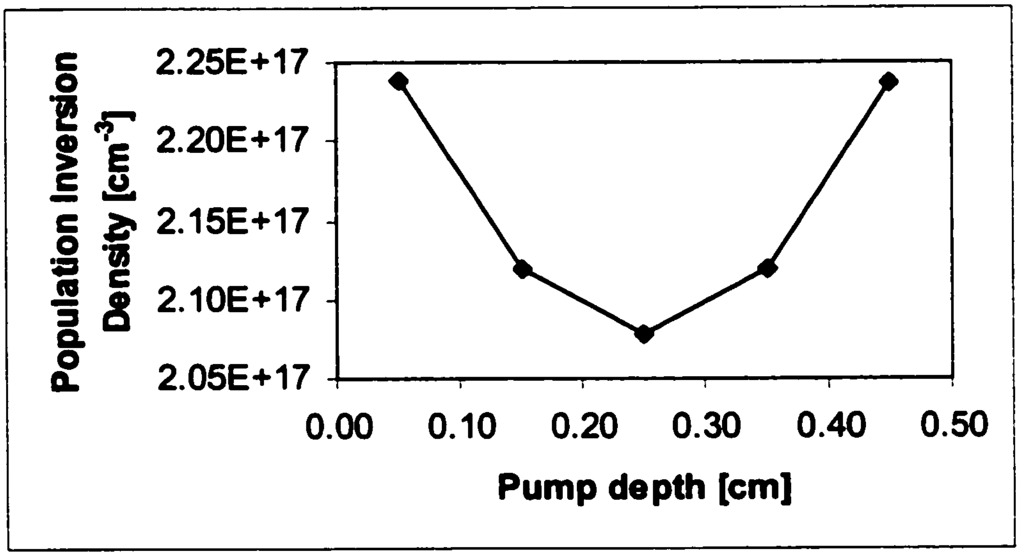

Figure 6-5: Spatial profile of population inversion density for stage 1. The inversion density is plotted along the pump direction, halfway along the gain length for a time of 69 ns (near the peak).

Stages two and three employ laser crystals having large cross-sectional areas (4 cm^2 and 16 cm^2 respectively) to accommodate the increasing energy of the pulse at each stage while working in a moderately saturated gain regime. Laser crystals with smaller cross section areas could be used for Ti:Sapphire, which has a much larger gain saturation fluence (800 mJ/cm^2) than does Ce:LLF. The geometry of stages 2 and 3 is shown in Figure 6-6. Both amplifiers are set up for a double pass geometry, with double sided axial pumping. The pump flux for stages 2 and 3 is 125 mJ/cm^2 per side, slightly higher than the pump saturation flux. Stage 2 has a two pass gain of 56, which results in an output energy of 78.6 mJ for an input of 1.4 mJ. Stage 3 has a two pass gain of 20.6, which results in an output energy of 1.14 J for an input of 55 mJ.

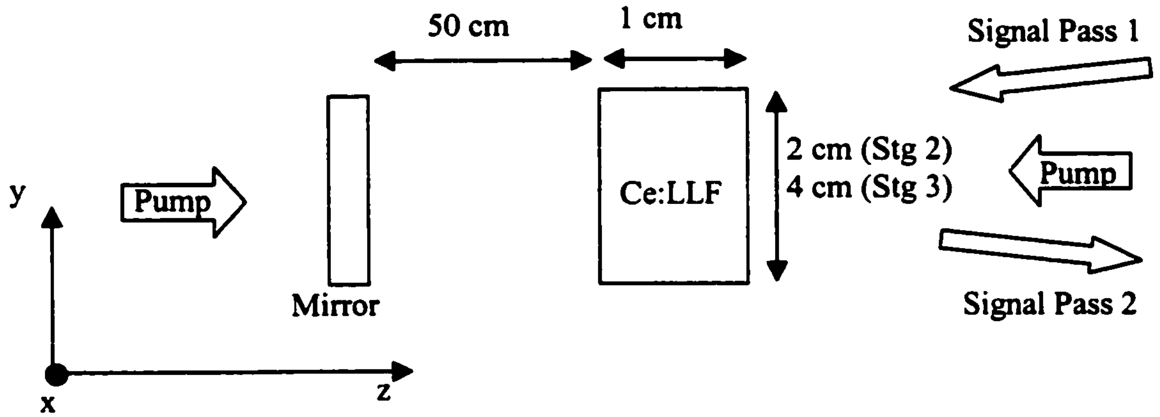

Figure 6-6: Geometry of Amplifier Stages 2 and 3. The pump and signal beams are assumed to have equal heights and widths, and fit exactly within the crystal face.

The temporal evolution of the population inversion density for stage 2 is shown in Figure 6-7. The first pass of the signal pulse experiences a gain near to the small signal value and does not significantly deplete the inversion. The second pass of the signal pulse (shown in Figure 6-8) experiences a somewhat saturated gain and causes a slight depletion of the population inversion.

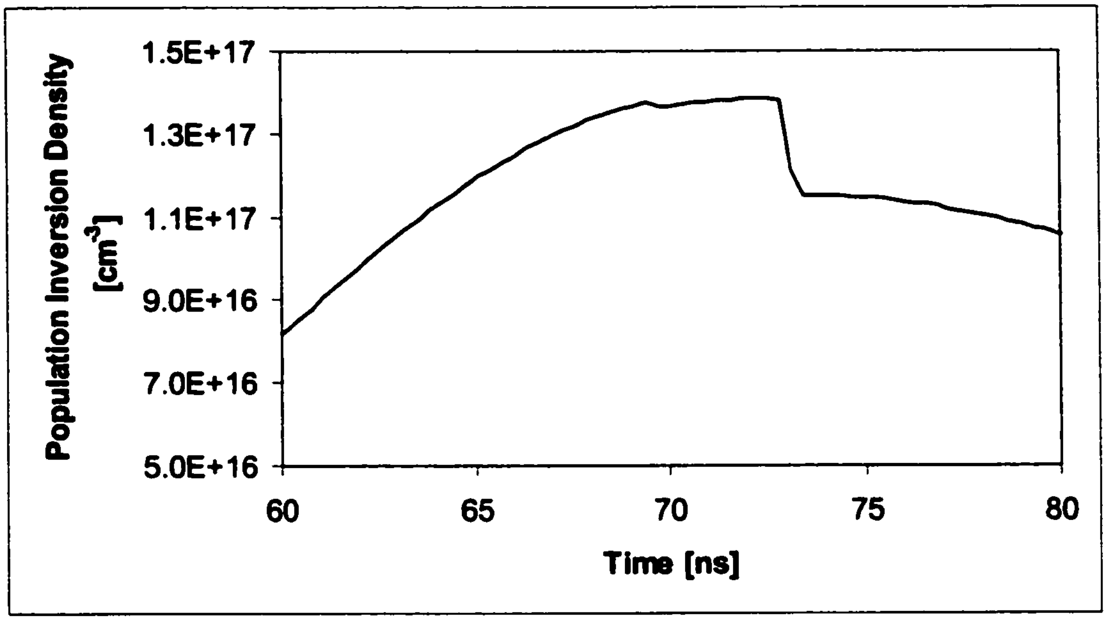

Figure 6-7: Population Inversion Density for Stage 2, plotted for the center of the laser crystal.

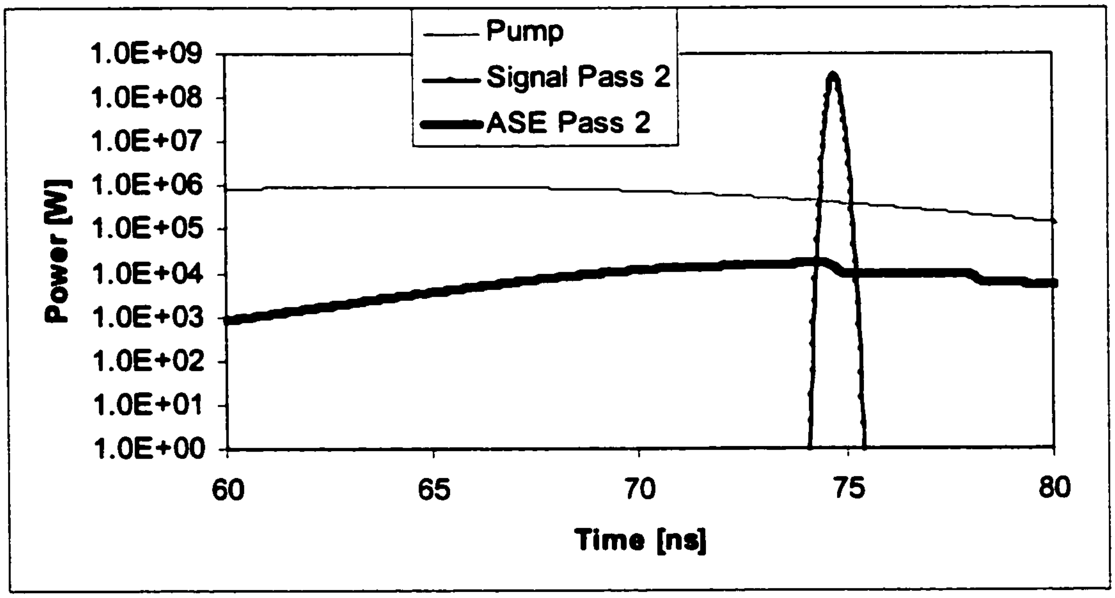

Figure 6-8: Beam Powers for Stage 2. The injected pump pulse reaches its peak at 60 ns, while the injected signal and ASE pulses are delayed by 8 ns. All powers are plotted for the exit of the second pass.

In stage 3, both passes of the signal pulse experience the effects of gain saturation. This effect is more pronounced with the second pass (shown in Figure 6-10), which produces an output signal flux of 71 mJ/cm^2, slightly higher than the gain saturation flux of 61 mJ/cm^2. Both of these passes cause significant population inversion depletions (shown in Figure 6-9), with this effect increasing with gain saturation. One advantage of gain saturation in the final amplifier stage is that the output signal energy will be relatively stable with respect to fluctuations in the input energy. Saturated gain also allows for a higher energy extraction ratio. A drawback of saturated gain is that the small angle ASE will experience a gain higher than that of the signal pulse, decreasing the signal to noise ratio. Thus, it is important to minimize the number of saturated beam passes through the gain medium, and generally avoid gain saturation in the earlier amplifier stages.

In both stage 2 and 3, it appears that additional gain could be achieved with the addition of extra passes of the seed pulse through the gain medium. The drawback of this approach is that small angle ASE propagating with the main signal beam would also be significantly amplified, which could potentially lead to an undesirable signal to noise ratio.

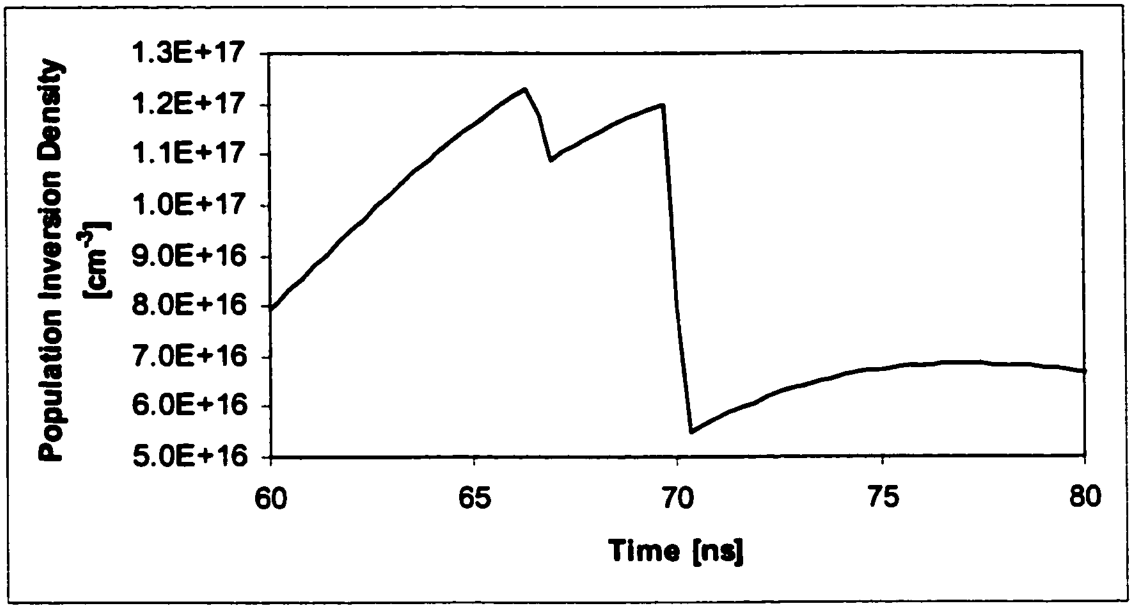

Figure 6-9: Population Inversion Density for Stage 3, plotted for the centre of the laser crystal.

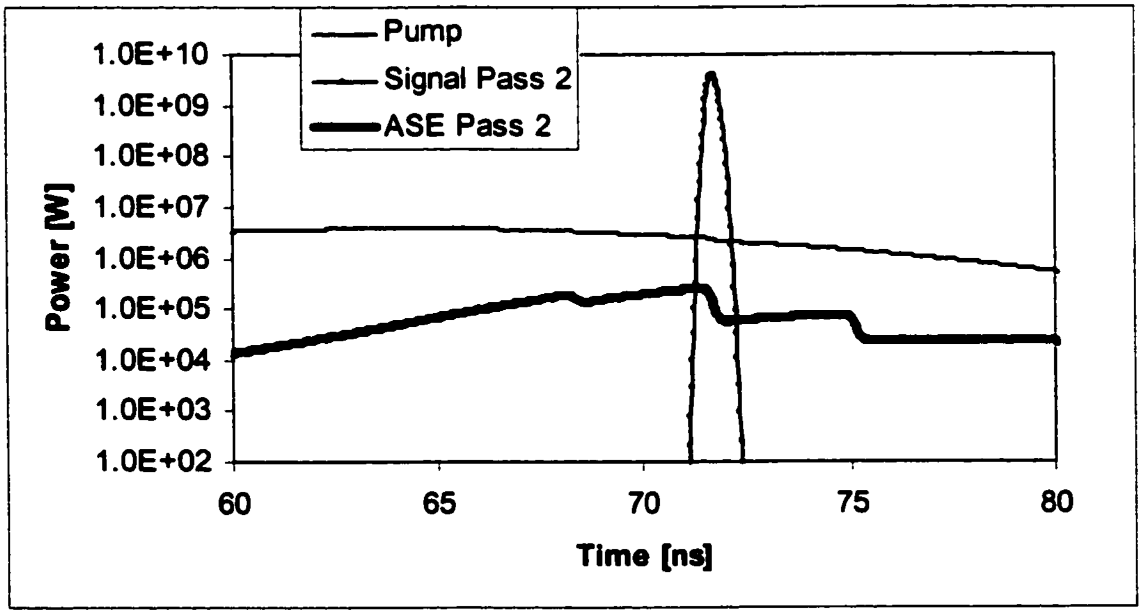

Figure 6-10: Beam Powers for Stage 3. The injected pump pulse reaches its peak at 60 ns, while the injected signal and ASE pulses are delayed by 5 ns. All powers are plotted for the exit of the second pass.

If the small angle ASE can be attenuated according to previously stated assumptions, a signal to noise (ASE) power ratio of 1.7×10^4 could be achieved for the stretched pulse output of stage 3, as seen in Figure 6-9. Given a gain bandwidth of 50 THz, a transform-limited Gaussian pulse of 8.8 fs duration could be amplified in a Ce:LLF amplifier system (based on a pulse bandwidth product of 0.441). Based on experience with Ti:Sapphire amplifier systems where 30 fs pulses have been amplified using the CPA technique to the multiterawatt level, we assume that the pulse could be compressed from 250 ps to 60 fs (assuming that Ce:LLF has a bandwidth roughly half that of Ti:Sapphire). In this case the peak signal to noise ratio would become 6.9×10^7. This would likely be an acceptable level of ASE for many short pulse experiments. Assuming 70% energy transmittance through the compression stage (from 250 ps to 60 fs), a compressed output energy of 798 mJ and a peak ouptut power of 13.3 TW would be achieved. If ASE cannot be suppressed as well as predicted in the simulations, it would be necessary to operate the amplifiers at a lower pump energies, yielding lower gains and output energies.

Chapter 7: Conclusion

Ce:LLF has the potential to provide tunable ultrashort laser pulses in the ultraviolet spectrum. The simulations show that high power 13.3 TW, 60 fs pulses could be achieved with a multistage chirped pulse amplification scheme pumped by 5.1 J from an electron-beam pumped KrF laser. Such a design would lead to peak laser powers in the ultraviolet wavelength range of interest for the study of relativistic laser plasma interactions.

The measured fluorescence spectra (for sigma and polarizations) have bandwidths comparable with published results[7,8,30]. The spontaneous lifetime of 37.6 ± 1.0 ns is close to the result reported by Sarukura (40 ± 3 ns)[7,13]. The absorption coefficient at 248 nm was found to be higher for pi polarization than for sigma polarization, which is consistent with published results[7,8].

The parameters determined from the gain data represent the best fit of the gain data with the simulation model. The absorption cross section of $(7.3 \pm 3.2) \times 10^{-18}$ cm^2 is comparable with other Cerium doped fluoride crystals. The 308 nm gain cross section of $(3.1 \pm 1.2) \times 10^{-18}$ cm^2 is smaller than the value determined by Sarukura at 325 nm[7], as expected since it is far from the peak gain wavelength. The crystal number density of $(1.7 \pm 0.6) \times 10^{18}$ cm^{-3} is smaller than what would be expected from the 0.5% doping concentration in the melt. This discrepancy is probably not significant, as crystal growth does not necessarily produce the same segregation coefficient and number density under different growing conditions. The above analysis assumes a unity branching ratio of the pump energy to the upper lasing state. If this is not the case, the gain cross section would be larger than the value determined in Chapter 5. If one makes the assumption of a quantum yield equal to 0.88 for Ce:LLF (determined by Sarukura and Dubinksii[7,13]), the gain cross section determined in Chapter 5 would need to be divided by 0.88, resulting in a gain cross section of 3.5×10^{-18} cm^2.

Future improvements to the simulation model could include a better representation of ASE, the inclusion of a branching ratio coefficient from the pumped state to the upper laser level and the representation of spectral (wavelength-dependent) effects. It would also be useful to account for the effects of excited state absorption and solarization.

Future experimental work with Ce:LLF would be useful. Specifically, imaged fluorescence experiments at higher pump fluxes would allow for a more accurate determination of the pump saturation flux (and thus also the absorption cross section). The measurement of saturated gain data would similarly allow for a more accurate determination of the gain cross section. The accurate measurement of both absorption and emission cross sections would allow a calculation of the effective pump quantum yield. A more detailed spectroscopic analysis of the energy level structure of Ce:LLF and the effects of excited state absorption and solarization would be useful for predicting the performance of Ce:LLF amplifiers.

Comparison of the amplifier simulations in Chapter 6 with actual experimental measurements would be an ideal extension of this work, allowing for further testing of the simulation program, characterization of Ce:LLF and the use of tunable ultrashort UV laser pulses for various high peak power applications. The optimum implementation of the amplifier designs would require the selection of an appropriate saturable absorber for ASE suppression between amplifier stages.

Appendix A: Simulation Parameter File

What follows is an example parameter file used with the Fortran simulation program Lascrt01. The parameters are split into four namelists, corresponding to laser crystal parameters, seed beam parameters, pump beam parameters and ASE parameters. Explanations of the parameters are given in [square brackets] on the right side of the pages. These explanations are not part of the actual parameter file.

The geometry of the laser beams is as follows: Seed and ASE beams propagate in the (positive or negative) z direction. There can be an unlimited number of passes for the seed and ASE beams, with different geometries and beam sizes for the different passes as required. Typically, small angle ASE will have the same number of passes as the injected seed pulse, while large angle ASE will have two passes (in the positive and negative z directions).

There are 4 pump beams allowed at present. Pump beams 1 and 2 propagate in the (positive and negative) z direction. Pump beams 3 and 4 propagate in the (positive or negative) y direction. Future modifications to the code could allow for additional pump beams 5 and 6, which would propagate in the (positive or negative x direction). Thus, pump beams 1 and 2 are used to specify axial pumping, while pump beams 3 and 4 are used to specify transverse pumping.

Each of the pump, seed and ASE beams can have a nonzero divergence and/or propagation angle. The angles can be specified as being different for the two transverse directions. For example, a seed beam propagating in the z direction could be specified as having an x divergence angle of 1 radian, and a y divergence angle of 2 radians.

Unless otherwise specified all energy units are in μJ, all spatial units are in cm and all temporal units are in ps.

Temporal and spatial profiles of beams are specified with a series of switch variables. Temporal profiles can be Gaussian, uniform, or read from a data file. Spatial profiles can be uniform or read from a data file.

A series of output files are used to collect and summarize various output data from the simulation. Most of the output files are in .csv format, which can be opened with Excel, Sigmaplot or Matlab. The output files are as follows:

UNIT	FILE	DESCRIPTION
12	SUMMARY.OUT	SUMAMRY DATA
13	DUMP.OUT	DIAGNOSTIC FILE FOR DEBUGGING
20	SLABOUT.CSV	OUTPUT BEAM ENERGIES AS A FCN. OF TIME
21	POPXY.CSV	XY DISTRIBUTION OF UPPER STATE POPULATION
22	POPYZ.CSV	YZ DISTRIBUTION OF UPPER STATE POPULATION
23	POPZX.CSV	ZX DISTRIBUTION OF UPPER STATE POPULATION
24	LASSPAT.CSV	SPATIAL DISTRIBUTION OF BEAMS
25	PUMPCALC.CSV	PUMP ABSORPTION CALC'S AS A FCN OF TIME
26	SEEDCALC.CSV	SEED GAIN CALC'S AS A FCN. OF TIME
27	GEOMCALC.CSV	GEOMETRICAL CALCULATIONS

```
&XTALPAR                  [namelist for crystal parameters]
COMMENT1=' COMPARE WITH EXP RESULTS OF 308NM GAIN '
COMMENT2=' KFSP1401.PAR '
COMMENT3= ' END MAR 2001 '
SLABOUTSW=90              [switch to control temporal resolution of slabout.csv
                           in this case, data for every 90th time step will be printed]
SYSTEMLEVELS=4            [4 level system]
DT=1.96D0                 [ time step in ps]
NSTEPS=90000              [number of time steps]
DUMPOU=0                  [switch to control which set of information is sent to
                           dump.out]
MIRREFL=0.D0 0.D0         [mirror reflectances; ]
N1X=0                     [number of x cells in initial airspace]
N2X=1                     [number of x cells in gain region]
N3X=0                     [number of x cells in final airspace]
N1Y=0                     [number of y cells in initial airspace]
N2Y=1                     [number of y cells in gain region]
N3Y=0                     [number of y cells in final airspace]
N1Z=0                     [number of z cells in initial airspace]
N2Z=19                    [number of z cells in gain region]
N3Z=0                     [number of z cells in final airspace]
DENSNUM=1.87777D18        [number density in cm-3]
SIGG=2.98D-18             [gain cross section in cm2]
SIGABS=6.6835D-18         [absorption cross section in cm2]
TGLIFE=37.57D3            [spontaneous lifetime in ps]
ALPHAG=0.7675D0           [absorption coefficient in cm-1 at 308 nm]
JT1=10000                 [1st time index for writing data to spatial o/p files]
JT2=20000                 [2nd time index for writing data to spatial o/p files]
JT3=50000                 [3rd time index for writing data to spatial o/p files]
POPXYSW=0                 [switch to control creation of popxy.csv;
                           0 = no output; 1 = full output]
LASSPATSW=0               [switch to control creation of lasspat.csv
                           0 = no output;  n>1: every nth data point]
PUMPCALCSW=0              [switch to control creation of pumpcalc.csv;
                           0 = no output; n>1: every nth data point]
SEEDCALCSW=0              [switch to control creation of seedcalc.csv;
                           0 = no output; n>1: every nth data point]
GEOMCALCSW=1              [switch to control creation of geomcalc.csv
                           0: no output; 1: data for first spatial step; >1 all data]
REFRINDEX=1.481D0         [refractive index]

/
```

```
&SEEDPAR                             [namelist for seed beam parameters]
INJPT=1                              [pass for injection of seed pulse]
THETAPROPX=0.D0 0.D0                 [x prop. angle in rad. for seed beams]
THETAPROPY=0.D0 0.D0                 [y prop. angle in rad. for seed beams]
THETADIVGX=0.D0 0.D0                 [x divg. angle in rad. for seed beams]
THETADIVGY=0.D0 0.D0                 [y divg. angle in rad. for seed beams]
XBEAMCENTRE=135.D-4 135.D-4          [x centre of seed beams; cm]
XBEAMWIDTH=270.D-4 270.D-4           [x width of seed beams; cm]
YBEAMCENTRE=198.5D-4 198.5D-4        [y centre of seed beams; cm]
YBEAMWIDTH=397.D-4 397.D-4           [y width of seed beams; cm]
RSEEDSW=2                            [switch to control spatial distribution of
                                      seed beam;
                                     0: data file 1: Gaussian (n/a) 2:Uniform]
TSEEDSW=0                            [switch to control temporal distribution of
                                      seed beam;
                                     0: data file 1: Gaussian  2:Uniform]
SEEDENERGY=3.3319D-2                 [input seed energy in µJ]
SEEDDURN=300.D0                      [duration of seed pulse in ps]
TSEED=25163.D0                       [time of seed pulse injection  ps]
SEEDRADE=1.D0                        [1/e radius of Gaussian seed beam]
LAMBDASEED=308.D-7                   [wavelength of seed pulse in cm]
SEEDPER=5.D23                        [repetiton period in ps
                                      for seed pulses]
NBX=1                                [number of seed beam divisions,
                                      x direction]
NBY=1                                [number of seed beam divisions,
                                      y direction]
NPASS=2                              [number of passes]
FILE50='jy18tk5.txt'                 [data file for temporal distribution
                                      of seed beam]
FILE51='FOO2.FOO'                    [data file for spatial distribution
                                      of seed beam]
NPT50=96                             [# of data points in FILE50]
NPT51=96                             [# of data points in FILE51]
DT50=200.D0                          [time step in ps between data points
                                      in FILE50]

/
```

```
&PUMPPAR                          [namelist for pump beam  parameters]
RPUMPSW=2                         [switch to control spatial distribution of
                                   pump beam;
                                  0: data file 1: Gaussian (n/a)  2:Uniform]
TPUMPSW=0                         [switch to control temporal distribution of
                                   pump beam;
                                  0: data file 1: Gaussian  2:Uniform]
PUMPENERGY(3)=5.314D3             [pass 3 input pump energy in μJ]
PUMPRADE=10.D-1                   [1/e radius for Gaussian
                                   Pump Beam]
LAMBDAPUMP=248.D-7                [wavelength of pump pulse in cm]
PUMPPER=23.D7                     [repetition period in ps
                                   for pump pulses]
PUMPDURN=9.96D0                   [duration of pump pulse in ps]
TPUMP=0.D0                        [time of pump pulse injection in ps]
RP12=0.D0                         [reflectance from pump pass 1 to 2]
XTHPRPGPY(3)=0.D0                 [x prop. angle in rad. for pump pass 3]
ZTHPRPGPY(3)=0.D0                 [z prop. angle in rad. for pump pass 3]
XTHDIVGPY(3)=0.D0                 [x divg. angle in rad. for pump pass 3]
ZTHDIVGPY(3)=0.D0                 [z divg. angle in rad. for pump pass 3]
XBEAMWIDTHPY(3)=270.D-4           [x width of pump beam pass 3; cm]
ZBEAMWIDTHPY(3)=7.543D-1          [z width of pump beam pass 3; cm]
XBEAMCENTREPY(3)=135.D-4          [x centre of pump beam pass 3; cm]
ZBEAMCENTREPY(3)=3.7715D-1        [z centre of pump beam pass 3; cm]
XTHPRPGPY(4)=0.0D0                [x prop. angle in rad. for pump pass 4]
ZTHPRPGPY(4)=0.0D0                [z prop. angle in rad. for pump pass 4]
XTHDIVGPY(4)=0.0D0                [x divg. angle in rad. for pump pass 4]
ZTHDIVGPY(4)=0.0D0                [z divg. angle in rad. for pump pass 4]
XBEAMWIDTHPY(4)=270.D-4           [x width of pump beam pass 4; cm]
ZBEAMWIDTHPY(4)=7.543D-1          [z width of pump beam pass 4; cm]
XBEAMCENTREPY(4)=135.D-4          [x centre of pump beam pass 4; cm]
ZBEAMCENTREPY(4)=3.7715D-1        [z centre of pump beam pass 4; cm]
MPX=1                             [# of pump beam x-divisions for passes 3,4]
MPZ=1                             [# of pump beam z-divisions for passes 3,4]
FILE54='jy11tk38.txt'             [data file for temporal distribution
                                   of pump beam]
FILE55='FOO55.FOO'                [data file for spatial distribution
                                   of pump beam]
NPT54=75                          [# of data points in FILE54]
NPT55=12                          [# of data points in FILE55]
DT54=500.D0                       [time step in ps for FILE54]
PUMPENERGY(1)=0.D3                [pass 1 input pump energy in μJ]
PUMPENERGY(2)=0.D3                [pass 2 input pump energy in μJ]
RP34=0.D0                         [reflectance from pump pass 3 to 4]
RP56=0.D0                         [reflectance from pump pass 5 to 6]
XTHPRPGPZ(1)=0.D0                 [x prop. angle in rad. for pump pass 1]
```

```
YTHPRPGPZ(1)=0.D0                [y prop. angle in rad. for pump pass 1]
XTHDIVGPZ(1)=0.D0                [x divg. angle in rad. for pump pass 1]
YTHDIVGPZ(1)=0.D0                [y divg. angle in rad. for pump pass 1]
XTHPRPGPZ(2)=0.D0                [x prop. angle in rad. for pump pass 2]
YTHPRPGPZ(2)=0.D0                [y prop. angle in rad. for pump pass 2]
XTHDIVGPZ(2)=0.D0                [x divg. angle in rad. for pump pass 2]
YTHDIVGPZ(2)=0.D0                [y divg. angle in rad. for pump pass 2]
XBEAMCENTREPZ(1)=135.D-4         [x centre of pump beam pass 1; cm]
XBEAMWIDTHPZ(1)=270.D-4          [x width of  pump beam pass 1; cm]
YBEAMCENTREPZ(1)=198.5D-4        [y centre of pump beam pass 1; cm]
YBEAMWIDTHPZ(1)=397.D-4          [y width of  pump beam pass 1; cm]
XBEAMCENTREPZ(2)=135.D-4         [x centre of  pump beam pass 2; cm]
XBEAMWIDTHPZ(2)=270.D-4          [x width of  pump beam pass 2; cm]
YBEAMCENTREPZ(2)=198.5D-4        [y centre of  pump beam pass 2; cm]
YBEAMWIDTHPZ(2)=397.D-4          [y width of  pump beam pass 2; cm]
NPX=1                            [# of pump beam x-divisions for passes 1,2]
NPY=1                            [# of pump beam y-divisions for passes 1,2]

/

&ASEPAR                          [namelist for ASE parameters]
ASESR=1.39D-2 1.39D-2            [solid angle of ASE acceptance in sr]
ASEENERGY=0.D0                   [energy of input ASE pulse in μJ]
ASEDURN=500.D0                   [duration of input ASE pulse in ps]
TASE=500.D0                      [time of injection of input ASE pulse  ps]
ASERADE=0.D0                     [1/e radius in cm for Gaussian Beam]
ASEREFL=0.D0 0.D0                [ASE reflectances]
TASESW=2                         [switch to specify temporal distribution
                                  of input ASE pulse;
                                  0: data file 1: Gaussian  2: Uniform ]
RASESW=2                         [switch to specify spatial distribution of
                                 of input ASE pulse;
                                 0: data file 1: Gaussian (n/a)  2: Uniform ]
FILE52='FOO52.FOO'               [data file for temporal distribution
                                  of seed beam]
FILE53='FOO53.FOO'               [data file for spatial distribution
                                  of seed beam]
NPT52=17                         [# of data points in FILE52]
NPT53=8                          [# of data points in FILE53]
DT52=2.78D3                      [time step in ps for FILE52]

/
```

Appendix B: Laser Simulation Code for 3D Cartesian Geometry

```
C LASCRT01.F
C 3- OR 4- LEVEL LASER SIMULATION USING CARTESIAN CO-ORDIANTES
C (USED FOR TRANSVERSE PUMPING SIMULATION)
C
C COMPILER: DEC VISUAL FORTRAN 5.0C
C PROJECT TYPE: QUICKWIN APPLICATION
C
C

C INPUT AND OUTPUT FILES ----------------------------------------

C [UNIT 11] INP.PAR          INPUT PARAMETERS SPECIFIED BY USER
C [UNIT 12] SUMMARY.OUT      SUMMARY OF OUTPUT CALCULATIONS
C [UNIT 13] DUMP.OUT         FILE USED FOR DEBUGGING [SEE DUMPOU]

C [UNIT 20] SLABOUT.CSV      OUTPUT SLAB ENERGIES AS A FUNCTION OF TIME
C [UNIT 21] POPXY.CSV        XY DISTRIBUTION OF UPPER STATE POPULATION
C [UNIT 22] POPYZ.CSV        YZ DISTRIBUTION OF UPPER STATE POPULATION
C [UNIT 23] POPZX.CSV        ZX DISTRIBUTION OF UPPER STATE POPULATION
C [UNIT 24] LASSPAT.CSV      SPATIAL DISTRIBUTION OF PUMP,SEED & ASE BEAMS
C [UNIT 25] PUMPCALC.CSV     PUMP CALCULATIONS AS A FUNCTION OF TIME
C [UNIT 26] SEEDCALC.CSV     SEED CALCULATIONS AS A FUNCTION OF TIME
C [UNIT 27] GEOMCALC.CSV     GEOMETRICAL CALCULATIONS

C FILENAMES FOR UNITS 50 TO 55 ARE SPECIFIED BY THE USER IN INP.PAR
C
C [UNIT 50]                  SEED TEMPORAL DISTRIBUTION
C [UNIT 51]                  SEED SPATIAL DISTRIBUTION
C [UNIT 52]                  INPUT ASE TEMPORAL DISTRIBUTION
C [UNIT 53]                  INPUT ASE SPATIAL DISTRIBUTION
C [UNIT 54]                  PUMP TEMPORAL DISTRIBUTION
C [UNIT 55]                  PUMP SPATIAL DISTRIBUTION
C
C ****** ALL TEMPORAL DISTRIBUTION FILES MUST BE SINGLE-COLUMN *********
C

C FILES FOR THIS PROJECT
C       ABSCRT01.F           ABSORPTION SUBROUTINE
C       AMPCRT01.F           AMPLIFICATION SUBROTUINE
C       CALCVOL.F            VOLUME CALCULATION FOR OVERLAP FRACTIONS
C       GEOCRT02.F           GEOMETRIC CALCULATION SUBROUTINE
C       INICRT01.F           INITIALIZATION SUBROUTINE
C       LASCRT01.F           MAIN PROGRAM
C       LPICRT01.F           LOOP INITIALIZATION SUBROUTINE
C       OUTCRT01.F           IN-LOOP PRINTING SUBROUTINE
C       PMICRT02.F           PUMP INITIALIZATION SUBROTUINE
                             FOR Y PROPAGATION
C       PMZCRT01.FOR         PUMP INITIALIZATION SUBROUTINE
                             FOR Z PROPAGATION
C       PRFCRT01.F           FINAL PRINTING SUBROUTINE
C       PRPCRT01.F           PROPAGATION SUBROUTINE
C       SDICRT02.F           SEED INITILAZATION SUBROTUINE
```

```
C       SPOCRT01.F          SPONTANEOUS EMISSION SUBROTUINE
C       LASCRT01.INC        COMMON DECLARATION FILE
C       LASCRT01.DSP        PROJECT FILE
C       LASCRT01.DSW        WORKSPACE FILE
C

C SEED VARIABLES ---------------------------------------------

C **** ALL USER-SPECIFIED ANGLES ARE FOR PROPAGATION IN AIR ****
C
C FILE50                    DATA FILE FOR TEMPORAL DISTRIBUTION OF SEED
C FILE51                    DATA FILE FOR SPATIAL DISTRIBUTION OF SEED
C NPT50                     NUMBER OF DATA POINTS IN FILE50
C NPT51                     NUMBER OF DATA POINTS IN FILE51
C DT50                      TIME STEP FOR FILE50
C
C NBX                       NUMBER OF SEED BEMALETS IN THE X DIRECTION
C NBY                       NUMBER OF SEED BEAMLETS IN THE Y DIRECTION
C
C MIRREFL(IP)               MIRROR REFLECTIVITY TO PASS IP.
C IPSOURCE(IP)              'SOURCE PASS' FOR REFELCTION TO PASS IP.
C                                   E.G. TO SPECIFY A REFLECTANCE OF 0.5 FROM
C                                           PASS 4 TO PASS 5, SET IPSOURCE(5)=4
C                                           AND MIRREFL(5)=0.5
C
C IBX1S(IP,IXG,IZG)         FIRST (SMALLEST X VALUE) SEED BEAMLET
C                           OVERLAPPING THE (IXG,IZG) RANGE OF GAIN CELLS
C IBX2S(IP,IXG,IZG)         LAST (LARGEST X VALUE) SEED BEAMLET
C                           OVERALPPING THE (IXG,IZG) RANGE OF GAIN CELLS
C IBY1S(IP,IYG,IZG)         FIRST (SMALLEST Y VALUE) SEED BEAMLET
C                           OVERLAPPING THE (IYG,IZG) RANGE OF GAIN CELLS
C IBY2S(IP,IYG,IZG)         FIRST (SMALLEST Y VALUE) SEED BEAMLET
C                           OVERLAPPING THE (IYG,IZG) RANGE OF GAIN CELLS
C

C THETAPROPX(IP)            SEED BEAM PROPAGATION DIRECTION IN XZ PLANE [rad]
C THETAPROPY(IP)            SEED BEAM PROPAGATION DIRECTION IN YZ PLANE [rad]
C THETADIVGX(IP)            SEED BEAM FULL DIVERGENCE ANGLE IN XZ PLANE [rad]
C THETADIVGY(IP)            SEED BEAM FULL DIVERGENCE ANGLE IN YZ PLANE [rad]
C
C XBEAMCENTRE(IP)           SEED BEAM CENTRE CO-ORDINATE IN XZ PLANE [cm]
C                                   (START OF PASS)
C               ALL 'BEAMCENTRE' VARIABLES ARE IN TERMS OF ABSOLUTE CO-ORD'S
C
C XBEAMWIDTH(IP)            SEED BEAM WIDTH IN XZ PLANE [cm]
C                                   (START OF PASS)
C YBEAMCENTRE(IP)           SEED BEAM CENTRE CO-ORDINATE IN YZ PLANE [cm]
C                                   (START OF PASS)
C YBEAMWIDTH(IP)            SEED BEAM WIDTH IN YZ PLANE [cm]
C                                   (START OF PASS)
C

C THETABEAMX(IP,0:IBX)      DIRECTION ANGLE OF SEED BEAMLET EDGE
C                                           IN XZ PLANE [rad]
C THETABEAMY(IP,0:IBX)      DIRECTION ANGLE OF SEED BEAMLET EDGE
C                                           IN YZ PLANE [rad]
```

```
C
C *** GENERAL NOTE FOR DIRECTION ANGLES:
C A POSITIVE ANGLE IS IN THE DIRECTION OF INCREASING BOTH CO-ORD'S.
C
C XVS(IP,0:IBX,0:IZ)          X CO-ORDINATE OF VERTEX
C                             FOR A RANGE OF SEED BEAMLETS [cm]
C YVS(IP,0:IBY,0:IZ)          Y CO-ORDINATE OF VERTEX
C                             FOR A RANGE OF SEED BEAMLETS [cm]
C FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)
C                             VOLUME OVERLAP FRACTION OF      SEED
C                             BEAMLET (IBX,IBY,IZG) WITH GAIN CELL (IXG,IYG,IZG)
C                             = OVERLAP VOLUME / BEAMLET VOLUME
C

C ESEEDINP(IP,IBX,IBY)        INPUT SEED ENERGY FOR
C                             THE BEAMLET (IBX,IBY) [uJ]
C ESEED(IP,IBX,IBY,IZ)        SEED ENERGY CONTAINED IN
C                             THE BEAMLET (IBX,IBY,IZ) [uJ]
C ESEEDSLAB(IP)               TOTAL SEED ENERGY CONTAINED IN
C                             THE FINAL Z STEP OF PASS IP [uJ]
C ESEEDTOTALOUT(IP)           TOTAL OUTPUT SEED ENERGY FROM PASS IP [uJ]
C
C RSEEDSW                     SWITCH TO SPECIFY RADIAL DISTRIBUTION OF SEED BEAM
C                             =0 READ FROM DATA FILE
C                             =1 NOT AVAILABLE (FUTURE GAUSSIAN?)
C                             =2 UNIFORM
C TSEEEDSW                    SWITCH TO SPECIFY TEMPORAL DISTRIBUTION
                              OF SEED BEAM
C                             =0 READ FROM DATA FILE
C                             =1 GAUSSIAN
C                             =2 UNIFORM
C
C SEEDENERGY                  TOTAL INPUT SEED ENERGY [uJ]
C SEEDDURN                    SEED PULSE DURATION [uJ]
C                             (FWHM IF GAUSSIAN)
C TSEED                       CENTRE TIME OF SEED PULSE [ps]
C SEEDRADE                    1/e^2 RADIUS OF SEED BEAM [cm]
C                             (ONLY USED FOR GAUSSIAN BEAM)
C LAMBDASEED                  SEED BEAM WAVELENGTH [cm]
C SEEDPER                     REPETITION PERIOD FOR SEED PULSES [ps]
C                             (IF SEEDPER IS ZERO OR NEGATIVE,
C                             THERE WILL BE NO REPETITION)

C ASE VARIABLES ----------------------------------------
C
C FILE52                      DATA FILE FOR TEMPORAL DISTRIBUTION OF INPUT ASE
C FILE53                      DATA FILE FOR SPATIAL DISTRIBUTION OF INPUT ASE
C NPT52                       NUMBER OF DATA POINTS IN FILE52
C NPT53                       NUMBER OF DATA POINTS IN FILE 53
C DT52                        TIME STEP FOR FILE52
C
C ASESR(IP)                   SOLID ANGLE OF ACCEPTED ASE ENERGY FOR
C                             PASS IP [sterad]
C                ** ASE IS NOT NORMALIZED W.R.T. NPASS
C
```

```
C
C ASEREFL(IP)              ASE REFLECTANCE TO PASS IP
C
C EASEINP(IP,IBX,IBY)      INPUT ASE ENERGY FOR
C                          THE BEAMLET (IBX,IBY) [uJ]
C EASE(IP,IBX,IBY,IZ)      ASE ENERGY CONTAINED IN
C                          THE BEAMLET (IBX,IBY,IZ) [uJ]
C EASESLAB(IP              TOTAL ASE ENERGY CONTAINED IN
C                          THE FINAL Z STEP OF PASS IP [uJ]
C EASETOTALOUT(IP)         TOTAL OUTPUT ASE ENERGY FROM PASS IP [uJ]
C
C ASEENERGY                TOTAL INPUT ASE ENERGY (OPTIONAL) [uJ]
C ASEDURN                  DURATION OF INPUT ASE PULSE [ps]
C                          (FWHM IF GAUSSIAN)
C TASE                     CENTRE TIME OF ASE INPUT PULSE [ps]
C ASERADE                  1/e^2 RADIUS OF ASE INPUT PULSE [cm]
C                          (ONLY IF GAUSSIAN BEAM)
C
C
C RASESW                   SWITCH TO SPECIFY RADIAL DISTRIBUTION OF
                           INPUT ASE BEAM
C                          =0 READ FROM DATA FILE
C                          =1 NOT AVAILABLE (FUTURE GAUSSIAN?)
C                          =2 UNIFORM
C TASESW                   SWITCH TO SPECIFY TEMPORAL DISTRIBUTION OF
                           INPUT ASE
C                          =0 READ FROM DATA FILE
C                          =1 GAUSSIAN
C                          =2 UNIFORM
C

C PUMP PASS NOTATION: ----------------------------------------------
C
C PASS #      DIRECTION    COUNTER VARIABLES      TOTAL VARIABLES
C ------ --------- ---------------- ---------------
C  1          POS. Z       I                          N
C  2          NEG. Z       I                          N
C  3          POS. Y       J                          M
C  4          NEG. Y       J                          M
C  5          POS. X       K                          L
C  6          NEG. X       K                          L
C

C PUMP VARIABLES ---------------------------------------------------
C
C **** ALL USER-SPECIFIED ANGLES ARE FOR PROPAGATION IN AIR ****
C
C FILE54                   DATA FILE FOR TEMPORAL DISTRIBUTION OF PUMP
C                          (MUST BE SINGLE-COLUMN)
C FILE55                   DATA FILE FOR SPATIAL DISTRIBUTION OF PUMP
C NPT54                    NUMBER OF DATA POINTS IN FILE54
C NPT55                    NUMBER OF DATA POINTS IN FILE 55
C DT54                     TIME STEP FOR FILE54
C
C
```

```
C NPX                   NUMBER OF (Z PROPAG.) PUMP BEAMLETS IN THE X
                        DIRECTION
C NPY                   NUMBER OF (Z PROPAG.) PUMP BEAMLETS IN THE Y
                        DIRECTION
C MPX                   NUMBER OF (Y PROPAG.) PUMP BEAMLETS IN THE X
                        DIRECTION
C MPZ                   NUMBER OF (Y PROPAG.) PUMP BEAMLETS IN THE Z
                        DIRECTION
C LPY                   NUMBER OF (X PROPAG.) PUMP BEAMLETS IN THE Y
                        DIRECTION
C LPZ                   NUMBER OF (X PROPAG.) PUMP BEAMLETS IN THE Z
                        DIRECTION
C
C
C RP12                  PUMP REFLECTANCE FROM PASS 1 TO PASS 2
C RP34                  PUMP REFLECTANCE FROM PASS 3 TO PASS 4
C RP56                  PUMP REFLECTANCE FROM PASS 5 TO PASS 6
C
C IBX1PZ(IP,IXG,IZG)    FIRST (SMALLEST X VALUE) PUMP BEAMLET (Z PROPAG.)
C                       OVERLAPPING THE (IXG,IZG) RANGE OF GAIN CELLS
C IBX2PZ(IP,IXG,IZG)    LAST (LARGEST X VALUE) PUMP BEAMLET (Z PROPAG.)
C                       OVERLAPPING THE (IXG,IZG) RANGE OF GAIN CELLS
C IBY1PZ(IP,IYG,IZG)    FIRST (SMALLEST Y VALUE) PUMP BEAMLET (Z PROPAG.)
C                       OVERLAPPING THE (IYG,IZG) RANGE OF GAIN CELLS
C IBY2PZ(IP,IYG,IZG)    LAST (LARGEST Y VALUE) PUMP BEAMLET (Z PROPAG.)
C                       OVERLAPPING THE (IYG,IZG) RANGE OF GAIN CELLS
C
C JBX1PY(JP,JXG,JYG)    FIRST (SMALLEST X VALUE) PUMP BEAMLET (Y PROPAG.)
C                       OVERLAPPING THE (JXG,JYG) RANGE OF GAIN CELLS
C JBX2PY(JP,JXG,JYG)    LAST (LARGEST X VALUE) PUMP BEAMLET (Y PROPAG.)
C                       OVERLAPPING THE (JXG,JYG) RANGE OF GAIN CELLS
C JBZ1PY(JP,JZG,JYG)    FIRST (SMALLEST Z VALUE) PUMP BEAMLET (Y PROPAG.)
C                       OVERLAPPING THE (JZG,JYG) RANGE OF GAIN CELLS
C JBZ2PY(JP,JZG,JYG)    LAST (LARGEST Z VALUE) PUMP BEAMLET (Y PROPAG.)
C                       OVERLAPPING THE (JZG,JYG) RANGE OF GAIN CELLS

C
C KBY1PX(KP,KYG,KXG)    FIRST (SMALLEST Y VALUE) PUMP BEAMLET (X PROPAG.)
C                       OVERLAPPING THE (KYG,KXG) RANGE OF GAIN CELLS
C KBY2PX(KP,KYG,KXG)    LAST (LARGEST Y VALUE) PUMP BEAMLET (X PROPAG.)
C                       OVERLAPPING THE (KYG,KXG) RANGE OF GAIN CELLS
C KBZ1PX(KP,KZG,KXG)    FIRST (SMALLEST Z VALUE) PUMP BEAMLET (X PROPAG.)
C                       OVERLAPPING THE (KZG,KXG) RANGE OF GAIN CELLS
C KBZ2PX(KP,KZG,KXG)    LAST (LARGEST Z VALUE) PUMP BEAMLET (X PROPAG.)
C                       OVERLAPPING THE (KZG,KXG) RANGE OF GAIN CELLS

C

C YTHPRPGPX(KP)         (X PROPAG.) PUMP BEAM PROPAGATION
C                       DIRECTION IN XY PLANE [rad]
C ZTHPRPGPX(KP)         (X PROPAG.) PUMP BEAM PROPAGATION
C                       DIRECTION IN XZ PLANE [rad]
C YTHDIVGPX(KP)         (X PROPAG.) PUMP BEAM FULL DIVERGENCE ANGLE
C                       IN XY PLANE [rad]
C ZTHDIVGPX(KP)         (X PROPAG.) PUMP BEAM FULL DIVERGENCE ANGLE
```

```
C                          IN XZ PLANE [rad]
C
C XTHPRPGPY(JP)            (Y PROPAG.) PUMP BEAM PROPAGATION
C                          DIRECTION IN XY PLANE [rad]
C ZTHPRPGPY(JP)            (Y PROPAG.) PUMP BEAM PROPAGATION
C                          DIRECTION IN ZY PLANE [rad]
C XTHDIVGPY(JP)            (Y PROPAG.) PUMP BEAM FULL DIVERGENCE ANGLE
C                          IN XY PLANE [rad]
C ZTHDIVGPY(JP)            (Y PROPAG.) PUMP BEAM FULL DIVERGENCE ANGLE
C                          IN ZY PLANE [rad]
C
C XTHPRPGPZ(IP)            (Z PROPAG.) PUMP BEAM PROPAGATION
C                          DIRECTION IN XZ PLANE [rad]
C YTHPRPGPZ(IP)            (Z PROPAG.) PUMP BEAM PROPAGATION
C                          DIRECTION IN YZ PLANE [rad]
C XTHDIVGPZ(IP)            (Z PROPAG.) PUMP BEAM FULL DIVERGENCE ANGLE
C                          IN XZ PLANE [rad]
C YTHDIVGPZ(IP)            (Z PROPAG.) PUMP BEAM FULL DIVERGENCE ANGLE
C                          IN YZ PLANE [rad]
C

C YTHBEAMPX(KP,0:KBY)      (X PROPAG.) DIRECTION ANGLE OF PUMP
C                          BEAMLET EDGE IN XY PLANE [rad]
C ZTHBEAMPX(KP,0:KBZ)      (X PROPAG.) DIRECTION ANGLE OF PUMP
C                          BEAMLET EDGE IN XZ PLANE [rad]
C ZTHBEAMPY(JP,0:JBZ)      (Y PROPAG.) DIRECTION ANGLE OF PUMP
C                          BEAMLET EDGE IN ZY PLANE [rad]
C XTHBEAMPY(JP,0:JBX)      (Y PROPAG.) DIRECTION ANGLE OF PUMP
C                          BEAMLET EDGE IN XY PLANE [rad]
C XTHBEAMPZ(IP,0:IBX)      (Z PROPAG.) DIRECTION ANGLE OF PUMP
C                          BEAMLET EDGE IN XZ PLANE [rad]
C YTHBEAMPZ(IP.0:IBY)      (Z PROPAG.) DIRECTION ANGLE OF PUMP
C                          BEAMLET EDGE IN ZY PLANE [rad]

C
C XBEAMWIDTHPZ(IP)         (Z PROPAG.) PUMP BEAM WIDTH IN XZ PLANE [cm]
C                          (START OF PASS)
C YBEAMWIDTHPZ(IP)         (Z PROPAG.) PUMP BEAM WIDTH IN XY PLANE [cm]
C                          (START OF PASS)
C XBEAMWIDTHPY(JP)         (Y PROPAG.) PUMP BEAM WIDTH IN XY PLANE [cm]
C                          (START OF PASS)
C ZBEAMWIDTHPY(JP)         (Y PROPAG.) PUMP BEAM WIDTH IN YZ PLANE [cm]
C                          (START OF PASS)
C YBEAMWIDTHPX(KP)         (X PROPAG.) PUMP BEAM WIDTH IN XY PLANE [cm]
C                          (START OF PASS)
C ZBEAMWIDTHPX(KP)         (X PROPAG.) PUMP BEAM WIDTH IN XZ PLANE [cm]
C                          (START OF PASS)
C
C XBEAMCENTREPZ(IP)        (Z PROPAG.) PUMP BEAM CENTRE CO-ORDINATE
C                          IN XZ PLANE [cm] (START OF PASS)
C YBEAMCENTREPZ(IP)        (Z PROPAG.) PUMP BEAM CENTRE CO-ORDINATE
C                          IN YZ PLANE [cm] (START OF PASS)
C XBEAMCENTREPY(JP)        (Y PROPAG.) PUMP BEAM CENTRE CO-ORDINATE
C                          IN XY PLANE [cm] (START OF PASS)
C ZBEAMCENTREPY(JP)        (Y PROPAG.) PUMP BEAM CENTRE CO-ORDINATE
C                          IN YZ PLANE [cm] (START OF PASS)
```

```
C YBEAMCENTREPX(KP)          (X PROPAG.) PUMP BEAM CENTRE CO-ORDINATE
C                            IN XY PLANE [cm] (START OF PASS)
C ZBEAMCENTREPX(KP)          (X PROPAG.) PUMP BEAM CENTRE CO-ORDINATE
C                            IN XZ PLANE [cm] (START OF PASS)
C

C XVPZ(IP,0:IBX,0:IZ)        X CO-ORDINATE OF VERTEX
C                            FOR A RANGE OF (Z PROPAG.) PUMP BEAMLETS [cm]
C YVPZ(IP,0:IBY,0:IZ)        Y CO-ORDINATE OF VERTEX
C                            FOR A RANGE OF (Z PROPAG.) PUMP BEAMLETS [cm]
C XVPY(JP,0:JBX,0:JY)        X CO-ORDINATE OF VERTEX
C                            FOR A RANGE OF (Y PROPAG.) PUMP BEAMLETS [cm]
C ZVPY(JP,0:JBZ,0:JY)        Z CO-ORDINATE OF VERTEX
C                            FOR A RANGE OF (Y PROPAG.) PUMP BEAMLETS [cm]
C YVPX(KP,0:KBY,0:KX)        Y CO-ORDINATE OF VERTEX
C                            FOR A RANGE OF (X PROPAG.) PUMP BEAMLETS [cm]
C ZVPX(KP,0:KBZ,0:KX)        Z CO-ORDINATE OF VERTEX
C                            FOR A RANGE OF (X PROPAG.) PUMP BEAMLETS [cm]
C
C FVOLPX(KP,KBY,KBZ,KXG,KYG,KZG)
C                            VOLUME OVERLAP FRACTION OF      (X PROPAG.) PUMP
C                            BEAMLET (KBY,KBZ,KXG) WITH GAIN CELL (KXG,KYG,KZG)
C FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG)
C                            VOLUME OVERLAP FRACTION OF      (Y PROPAG.) PUMP
C                            BEAMLET (JBX,JBZ,JYG) WITH GAIN CELL (JXG,JYG,JZG)
C FVOLPZ(IP,IBX,IBY,IXG,IYG,IZG)
C                            VOLUME OVERLAP FRACTION OF      (Z PROPAG.) PUMP
C                            BEAMLET (IBX,IBY,IZG) WITH GAIN CELL (IXG,IYG,IZG)
C
C       (GENERAL OVERLPA FRACTION)= OVERLAP VOLUME / BEAMLET VOLUME


C
C
C EPUMPZ(IP,IBX,IBY,IZ)      (Z PROPAG.) PUMP ENERGY CONTAINED
C                            IN ONE BEAMLET [uJ]
C EPUMPINPZ(IP,IBX,IBY)      (Z PROPAG.) INPUT PUMP ENERGY
C                            FOR A BEAMLET [uJ]
C EPUMPY(JP,JBX,JBZ,JY)      (Y PROPAG.) PUMP ENERGY CONTAINED
C                            IN ONE BEAMLET [uJ]
C EPUMPINPY(JP,JBX,JBZ)      (Y PROPAG.) INPUT PUMP ENERGY
C                            FOR A BEAMLET [uJ]
C EPUMPX(KP,KBY,KBZ,KX)      (X PROPAG.) PUMP ENERGY CONTAINED
C                            IN ONE BEAMLET [uJ]
C EPUMPINPX(KP,KBY,KBZ)      (X PROPAG.) INPUT PUMP ENERGY
C                            FOR A BEAMLET [uJ]
C
C EPUMPSLAB(IP)              TOTAL PUMP ENERGY CONTAINED IN THE FINAL
C                            PROPAGATION STEP OF PASS IP. [uJ]
C EPUMPTOTALOUT(IP)          TOTAL PUMP ENERGY IN PASS IP [uJ]
C

C RPUMPSW                    SWITCH TO SPECIFY RADIAL DISTRIBUTION OF PUMP BEAM
C                            =0 READ FROM DATA FILE
C                            =1 (NOT AVAILABLE - FUTURE GAUSSIAN?)
C                            =2 UNIFORM
```

```
C
C TPUMPSW                      SWITCH TO SPECIFY TEMPORAL DISTRIBUTION OF PUMP
C                              PULSE
C                              =0 READ FROM DATA FILE
C                              =1 GAUSSIAN
C                              =2 UNIFORM
C
C PUMPENERGY(JP)               TOTAL INPUT PUMP ENERGY FOR A PASS [uJ]
C PUMPDURN                     PUMP PULSE DURATION (FWHM IF GAUSSIAN) [ps]
C TPUMP                        CENTRE TIME OF PUMP PULSE [ps]
C PUMPRADE                     1/e^2 RADIUS OF PUMP BEAM (IF GAUSSIAN) [cm]
C LAMBDAPUMP                   PUMP WAVELENGTH [cm]
C PUMPPER                      REPETITION PERIOD OF PUMP PULSES [ps]
C


C
C OTHER COMMON VARIABLES, ARRAYS --------------------------------
C

C
C ******* ARRAY MAXIMUM DIMENSIONS - DEFINED IN LASCRT01.INC **********
C NBMAX                        MAXIMUM NUMBER OF BEAMLET DIVISIONS FOR A BEAM
C NXMAX                        MAXIMUM NUMBER OF X CELLS
                               (N1X + N2X + N3X .LE. NXMAX)
C NYMAX                        MAXIMUM NUMBER OF Y CELLS
                               (N1Y + N2Y + N3Y .LE. NYMAX)
C NZMAX                        MAXIMUM NUMBER OF Z CELLS
                               (N1Z + N2Z + N3Z .LE. NZMAX)
C NPMAX                        MAXIMUM UMBER OF SEED PASSES
C N2XMAX                       MAXIMUM VALUE FOR N2X
C N2YMAX                       MAXIMUM VALUE FOR N2Y
C N2ZMAX                       MAXIMUM VALUE FOR N2Z

C SYSTEMLEVELS                 =3 -----> 3 LEVEL SYSTEM WITH COMMON UPPER STATE
C                              (UPPER STATE IS SHARED BY PUMP AND
                               LASE TRANSITIONS)
C                              =-3 -----> 3 LEVEL SYSTEM WITH COMMON
                               GROUND STATE
C                              (GROUND STATE IS SHARED BY PUMP AND
                               LASE TRANSITIONS)
C                              =4 -----> 4 LEVEL SYSTEM
C
C GPOP(IX,IY,IZ)               UPPER STATE POPULATION DENSITY [cm^-3]
C APOP(IX,IY,IZ)               GROUND STATE POPULATION DENSITY [cm^-3]
C
C REFREINDEX =                 REFRACTIVE INDEX OF THE GAIN MEDIUM [dq]
C DWAIR =                      LENGTH OF AXIAL SPATIAL STEP IN AIR SPACES [cm]
C
C
C IQUIT                        DIAGNOSTIC VARIABLE
C                              =1 ----> PROBLEM ----> TERMINATE PROGRAM
C                              =0 ----> NO PROBLEM ----> NORMAL EXECUTION
                                                         OF PROGRAM
C
```

```
C DUMPOU                CONTROL PRINTING TO DUMPOUT FILE
C                       =5 PRINT SEED GEOMETRICAL CALC'S
C                       =6 PRINT PUMP GEOMETRICAL CALC'S
C                       =7 PRINT SEED GAIN CALC'S
C                       =8 PRINT PUMP ABS. CALC'S
C                       =9 PRINT ASE CALC'S
C
C SLABOUTSW             CONTROL RESOLUTION OF SLABOUT.CSV
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .GT. 0 -----> WRITE EVERY (SLABOUTSW)TH DATA POINT
C
C POPXYSW               CONTROL RESOLUTION OF POPXY.CSV
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .GT. 0 -----> WRITE EVERY DATA POINT

C POPYZSW               CONTROL RESOLUTION OF POPYZ.CSV
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .GT. 0 -----> WRITE EVERY DATA POINT

C POPZXSW               CONTROL RESOLUTION OF POPZX.CSV
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .GT. 0 -----> WRITE EVERY DATA POINT

C
C LASSPATSW             CONTROL RESOLUTION OF LASSPAT.CSV
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .GT. 0 -----> WRITE EVERY (LASSPATSW)TH DATA POINT
C
C PUMPCALCSW            CONTROL RESOLUTION OF PUMPCALC.CSV
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .GT. 0 -----> WRITE EVERY (PUMPCALCSW)TH DATA POINT
C
C SEEDCALCSW            CONTROL RESOLUTION OF SEEDCALC.CSV
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .GT. 0 -----> WRITE EVERY (SEEDCALCSW)TH DATA POINT
C
C GEOMCALCSW            CONTROL OUPTUT OF GEOMETRICAL DATA
C                       .LE. 0 -----> DO NOT WRITE DATA TO FILE
C                       .EQ. 1 -----> WRITE DATA FOR FIRST SPATIAL STEP
C                       .EQ. 2 -----> WRITE ALL DATA
C

C JT1,JT2,JT3           TIME IDEXES FOR WHICH SPATIAL INFORMATION
C                       WILL BE PLOTTED.
C
C NPPARITY              EQUALS ZERO IF NPASS IS EVEN, EQUALS ONE OTHERWISE
C NPFINALEVEN           FINAL EVEN PASS
C NPFINALODD            FINAL ODD PASS
C
C NSTEPS                NUMBER OF TIME STEPS IN THE SIMULATION
C                       NSTEPS SHOULD BE 10 OR GREATER TO
                        AVOID DIV BY ZERO.
C N1X                   NUMBER OF CELLS IN INITIAL AIRSPACE (X DIRECTION)
C N2X                   NUMBER OF CELLS IN GAIN MEDIUM (X DIRECTION)
C N3X                   NUMBER OF CELLS IN FINAL AIRSPACE (X DIRECTION)
C N1Y                   NUMBER OF CELLS IN INITIAL AIRSPACE (Y DIRECTION)
```

```
C N2Y                     NUMBER OF CELLS IN GAIN MEDIUM (Y DIRECTION)
C N3Y                     NUMBER OF CELLS IN FINAL AIRSPACE (Y DIRECTION)
C N1Z                     NUMBER OF CELLS IN INITIAL AIRSPACE (Z DIRECTION)
C N2Z                     NUMBER OF CELLS IN GAIN MEDIUM (Z DIRECTION)
C N3Z                     NUMBER OF CELLS IN FINAL AIRSPACE (Z DIRECTION)
C
C NX                      N1X + N2X + N3X
C NY                      N1Y + N2Y + N3Y
C NZ                      N1Z + N2Z + N3Z
C

C IXGMID                  CENTRAL X CO-ORDINATE OF GAIN REGION
C IYGMID                  CENTRAL Y CO-ORDINATE OF GAIN REGION
C IZGMID                  CENTRAL Z CO-ORDINATE OF GAIN REGION
C
C TIME                    TIME CORRESPONDING TO THE END OF A TIME STEP [ps]
C DENSNUM                 NUMBER DENSITY OF THE ACTIVATOR [cm^-3]
C SIGG                    GAIN CROSS-SECTION AT LASING WAVELENGTH [cm^2]
C SIGABS                  ABSORPTION CORSS-SECTION AT
                          PUMP WAVELEGNTH [cm^2]
C TGLIFE                  UPPER STATE LIFETIME [ps]
C ALPHAG                  LINEAR ABSORPTION COEFFICIENT @
                          LASING WAVELENGTH [cm^-1]

C
C DX,DY,DZ                LENGTH OR WIDTH OF A UNIT GAIN CELL [cm]
C DA                      SIDE FACE ARFEA OF A UNIT GAIN CELL [cm^2]
C DV                      VOLUME OF A UNIT GAIN CELL [cm^3]
C DT                      TIME STEP [ps]
C

C COMMENT1, COMMENT2, COMMENT3
C                         CHARACTER STRING ARRAYS - UP TO 70 CHAR'S
C                         COMMENTS FOR DESCRIBING THE SIMULATION
C
C
C
C ------------------------------------------------------------
C


C "USE DFLIB" IS NEEDED IN ORDER TO USE A QUICKWIN FUNCTION (SETEXITQQ).
      USE DFLIB

C 'GLOBAL' DECLARATIONS
   INCLUDE 'Lascrt01.inc'

C LOCAL DECLARATIONS
      INTEGER LOOPFRAC
      DOUBLE PRECISION LOOPPCT

C QUICKWIN FUNCTION FOR AUTOMATIC TERMINATION OF QUICKWIN APPLICATION
C WHEN FINISHED.  THIS IS NEEDED FOR BATCH FILE OPERATION.  ULTIMATELY,
C THIS COULD BE CONTROLLED BY AN INPUT PARAMETER.

      INTEGER(4) RESULT
```

```
        RESULT=SETEXITQQ(QWIN$EXITNOPERSIST)

C INITIALIZATION -----------------------------------------------

        CALL INICRT01

        IF (IQUIT.EQ.1) THEN

                PRINT '(A30)','PROGRAM TERMINATED'
                GOTO 9999

        END IF

        PRINT '(A30)','INICRT01 FINISHED'

        LOOPFRAC=INT(DBLE(NSTEPS)/(10.D0))
        PRINT '(A10,I10)','NSTEPS=',NSTEPS
        PRINT '(A10,I10)','LOOPFRAC=',LOOPFRAC


C MAIN LOOP ----------------------------------------------------

        DO JT=1,NSTEPS

                TIME=DT*(DBLE(JT))

                IF ((DUMPOU.EQ.7).OR.(DUMPOU.EQ.8).OR.(DUMPOU.EQ.666) ) THEN
                        WRITE(13,'(A20)'),'*-*-*-*-*-*-*-*-*'
                        WRITE(13,'(A5,I3,A6,D13.5)'),'JT= ',JT,'TIME=',TIME
                END IF


                CALL LPICRT01
                CALL PRPCRT01
                CALL ABSCRT01

                IF (TGLIFE.GT.(0.D0)) THEN
                        CALL SPOCRT01
                END IF

                CALL AMPCRT01
                CALL OUTCRT01

                IF (IQUIT.EQ.1) THEN
                        PRINT '(A30)','PROGRAM TERMINATED'
                        GOTO 9999
                END IF

C PRINT THE % COMPLETION FACTOR TO THE SCREEN (IF IT IS A MULTIPLE
C       OF TEN).

                IF (MOD(JT,LOOPFRAC).EQ.0) THEN
                        LOOPPCT=100*DBLE(JT)/DBLE(NSTEPS)
                        PRINT '(A10,I10,A15,F6.2,A10)','JT=',JT,' MAIN LOOP=',
     *                  LOOPPCT,'% COMPLETE'
                END IF
```

```
      END DO

C POST-LOOP PRINTING ---------------------------------------------

      CALL PRFCRT01


C OPEN FINISHED.OUT - THIS ALLOWS FOR A BATCH FILE TO DETECT WHEN ----
C THE RUN IS FINISHED.

      OPEN(99,FILE='finis.out')

C CLOSE THE DEFAULT WINDOW. ----------------------------------------

      CLOSE(UNIT=6,STATUS='DELETE')



 9999 STOP
      END
```

……………………………………………………………………………………………

```
      SUBROUTINE ABSCRT01

C ABSORPTION OF PUMP RADIATION
C
C


      INCLUDE 'Lascrt01.inc'

C LOCAL DECLARATIONS ------------------------------------------------

      DOUBLE PRECISION APOPINV,GPOPTEMP,APOPTEMP
      DOUBLE PRECISION EPUMPTOTIN,CPMP,SATFRC,PARENTH,LNARG,EABS
      DOUBLE PRECISION EPUMPTOTOUT,DELTAGPOP,EABSPU
      DOUBLE PRECISION ESATPUMPLOCAL,XPHIL,ZPHIL

C LOOP THROUGH THE Y-ZONES IN THE GAIN MEDIUM ----------------------

      DO JY=N1Y+1,N1Y+N2Y

            JYG=JY-N1Y

            DO JX=N1X+1,N1X+N2X
                  JXG=JX-N1X
                  DO JZ=N1Z+1,N1Z+N2Z
                        JZG=JZ-N1Z

                        IF (DUMPOU.EQ.8) THEN
                              WRITE(13,'(A5,I4,A5,I4,A5,I4,A16)'),
     *                        'JY=',JY,'JX=',JX,
     *                        'JZ=',JZ,'================'
                              WRITE(13,'(A5,I4,A5,I4,A5,I4)'),
     *                        'JYG=',JYG,'JXG=',JXG,
     *                        'JZG=',JZG
                        END IF


C CALCULATE THE TOTAL PUMP ENERGY ENTERING A CUBIC GAIN ZONE.
C ONLY LOOP THROUGH THE OVERLAPPING BEAMLETS.

                        EPUMPTOTIN=0.D0

                        DO JP=3,4

                              DO JBX=JBX1PY(JP,JXG,JYG),JBX2PY(JP,JXG,JYG)
                               DO JBZ=JBZ1PY(JP,JZG,JYG),JBZ2PY(JP,JZG,JYG)

                                          EPUMPTOTIN=EPUMPTOTIN+
     *                                    ( FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG)*
     *                                    EPUMPY(JP,JBX,JBZ,JY) )

                                    IF (DUMPOU.EQ.8) THEN

                                           WRITE(13,'(A5,I4,A5,I4,A5,I4,A11)'),
     *                                          'JP=',JP,'JBX=',JBX,
     *                                          'JBZ=',JBZ,'----------'
```

```
                                WRITE(13,'(3(A14,D15.7))'),
     *                          'FVOLPY()=',
     *
                                FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG),
     *
                                'EPUMPY()=',EPUMPY(JP,JBX,JBZ,JY),
     *                          'EPUMPTOTIN=',EPUMPTOTIN
                          END IF

                         END DO
                  END DO
            END DO

C           ADD IN THE ENERGY FROM THE PUMP Z BEAMS


            DO IP=1,2

             DO IBX=IBX1PZ(IP,JXG,JZG),IBX2PZ(IP,JXG,JZG)
              DO IBY=IBY1PZ(IP,JYG,JZG),IBY2PZ(IP,JYG,JZG)

                   EPUMPTOTIN=EPUMPTOTIN+
     *             ( FVOLPZ(IP,IBX,IBY,JXG,JYG,JZG)*
     *             EPUMPZ(IP,IBX,IBY,JZ) )

                   IF (DUMPOU.EQ.8) THEN

                          WRITE(13,'(A5,I4,A5,I4,A5,I4,A11)'),
     *                    'IP=',IP,'IBX=',IBX,
     *                    'IBY=',IBY,'---------'
                          WRITE(13,'(3(A14,D15.7))'),
     *                    'FVOLPZ()=',
     *
                          FVOLPZ(IP,IBX,IBY,JXG,JYG,JZG),
     *
                          'EPUMPZ()=',EPUMPZ(IP,IBX,IBY,JZ),
     *                    'EPUMPTOTIN=',EPUMPTOTIN
                   END IF

              END DO
             END DO
            END DO


C           CALCULATE APOPINV. THIS IS THE DIFFERENCE BETWEEN
C           GROUND STATE POPULATION DESNITY AND 'TARGET'
C           STATE POPULATION DENSITY

            APOP(JXG,JYG,JZG)=DENSNUM-GPOP(JXG,JYG,JZG)

            IF (SYSTEMLEVELS.EQ.3) THEN
                   APOPINV=APOP(JXG,JYG,JZG)-GPOP(JXG,JYG,JZG)
            ELSE
                   APOPINV=APOP(JXG,JYG,JZG)
            END IF
```

```
                IF (APOPINV.LE.(0.D0)) THEN
                        EPUMPTOTOUT=EPUMPTOTIN
                        GOTO 7373
                END IF

C               CALCULATE ESATPUMPLOCAL ----------------------------
C               DIFFERENT FROM ESATPUMP FOR INCOMPLETELY FILLED
C               CELLS
C               THIS ACCOUNTS FOR XBEAMWIDTHPY.LT.DX
C                       AND/OR ZBEAMWDITHPY.LT.DZ
C               THIS PART NOT SET UP FOR Z PUMPING

                ZPHIL=1.D0
                XPHIL=1.D0
                ESATPUMPLOCAL=ESATPUMP
                IF (XBEAMWIDTHPY(3).LT.DX) THEN
                        XPHIL=(XBEAMWIDTHPY(3)/DX)
                END IF
                IF (ZBEAMWIDTHPY(3).LT.DZ) THEN
                        ZPHIL=(ZBEAMWIDTHPY(3)/DZ)
                END IF
                ESATPUMPLOCAL=XPHIL*ZPHIL*ESATPUMP

C               CALCULATE THE ABSORPTION ---------------------------

                CPMP=-1.D0*SIGABS*DY*APOPINV
                SATFRC=EPUMPTOTIN/ESATPUMPLOCAL

                IF (SATFRC.LT.(1.D-4)) THEN
C                       UNSATURATED CALCULATION
                        EPUMPTOTOUT=EPUMPTOTIN*(DEXP(CPMP))
                        PARENTH=0.D0
                        LNARG=0.D0

                ELSE
C                       SATURTATED CALCULATION. USE EQN 3.56, D/R
                        PARENTH=1.D0-DEXP(SATFRC)
                        LNARG=1.D0-(DEXP(CPMP))*PARENTH
                        EPUMPTOTOUT=ESATPUMPLOCAL*DLOG(LNARG)
                END IF

 7373           CONTINUE

C CHECK IF ABSORBED ENERGY EXCEEDS INPUT PUMP ENERGY -----------------

                IF (EPUMPTOTOUT.GT.EPUMPTOTIN) THEN
                        EABS=EPUMPTOTIN
                        EPUMPTOTOUT=0.D0
                ELSE
                        EABS=EPUMPTOTIN-EPUMPTOTOUT
                END IF

                IF (DUMPOU.EQ.8) THEN
                        WRITE(13,'(A20,D25.17)'),'IP APOPINV=',APOPINV
                        WRITE(13,'(A20,D25.17)'),'IP GPOP()=',
     *                  GPOP(JXG,JYG,JZG)
```

```
                                   WRITE(13,'(A20,D25.17)'),'IP APOP()=',
     *                             APOP(JXG,JYG,JZG)
                                   WRITE(13,'(A20,D15.7)'),
     *                             'ESATPUMPLOCAL=',ESATPUMPLOCAL
                                   WRITE(13,'(A20,D15.7)'),'CPMP=',CPMP
                                   WRITE(13,'(A20,D15.7)'),'SATFRC=',SATFRC

                                   IF (SATFRC.LT.(1.D-4)) THEN
                                        WRITE(13,'(A20)'),'**SM. SIGNAL ABS'
                                   ELSE
                                        WRITE(13,'(A20)'),'**LG. SIGNAL ABS'
                                   END IF

                                   WRITE(13,'(A20,D15.7)'),'PARENTH=',PARENTH
                                   WRITE(13,'(A20,D15.7)'),'LNARG=',LNARG

         WRITE(13,'(A20,D15.7)'),'EPUMPTOTOUT=',EPUMPTOTOUT
                                   WRITE(13,'(A20,D15.7)'),'EABS=',EABS
                              END IF



C WRITE DATA TO PUMPCALC.CSV ------------------------------------------

                              IF ( (PUMPCALCSW.GT.0).AND.
     *                        (MOD(JT,PUMPCALCSW).EQ.0).AND.(JXG.EQ.IXGMID)
     *                        .AND.(JYG.EQ.IYGMID).AND.(JZG.EQ.IZGMID) ) THEN

                                   WRITE(25,'(9(A3,E13.5))') ,',', TIME ,',',
     *                             APOPINV ,',',
     *                             EPUMPTOTIN ,',', CPMP ,',', SATFRC ,',',
     *              PARENTH ,',', LNARG ,',', EABS ,',',
     *                             EPUMPTOTOUT

                              END IF


C CHECK IF ABSORBED ENERGY WOULD CAUSE GPOP TO EXCEED DENSNUM --------
C ALSO UPDATE THE POPULATIONS HERE.

                              APOPTEMP=APOP(JXG,JYG,JZG)-(EABS*KPUMP/DA)
                              GPOPTEMP=GPOP(JXG,JYG,JZG)+(EABS*KPUMP/DA)

                              IF (    (APOPTEMP.LE.(0.D0)    )
     *                        .OR.(GPOPTEMP.GE.DENSNUM    ) ) THEN

                                   DELTAGPOP=DENSNUM-GPOP(JXG,JYG,JZG)
                                   EABS=DELTAGPOP*DA/KPUMP
                                   GPOP(JXG,JYG,JZG)=DENSNUM
                                   APOP(JXG,JYG,JZG)=0.D0

                                   IF (DUMPOU.EQ.8) THEN
                                        WRITE(13,'(A23)'),'POP UPDATE OVERRIDDEN'
                                   END IF

                              ELSE
```

```
                              GPOP(JXG,JYG,JZG)=GPOPTEMP
                              APOP(JXG,JYG,JZG)=DENSNUM-GPOP(JXG,JYG,JZG)


                      END IF


C                     IF ( ( GPOP(JXG,JYG,JZG)+(EABS*KPUMP/DA) )
C    *                .GT.DENSNUM) THEN
C                             DELTAGPOP=DENSNUM-GPOP(JXG,JYG,JZG)
C                             EABS=DELTAGPOP*DA/KPUMP
C                             GPOP(JXG,JYG,JZG)=DENSNUM
C                             APOP(JXG,JYG,JZG)=0.D0
C                             GOTO 3232
C                     END IF

C UPDATE POPULATIONS -----------------------------------------------

C                     GPOP(JXG,JYG,JZG)=GPOP(JXG,JYG,JZG)+(EABS*KPUMP/DA)
C                     APOP(JXG,JYG,JZG)=DENSNUM-GPOP(JXG,JYG,JZG)

C 3232                CONTINUE


                      IF (DUMPOU.EQ.8) THEN
                              WRITE(13,'(A23)'),'UPDATED POPULATIONS:'
C                             WRITE(13,'(A5,I4,A5,I4,A5,I4,A11)'),
C    *                                'JP=',JP,'JBX=',JBX,
C    *                                'JBZ=',JBZ,'---------'
                              WRITE(13,'(A20,D25.17)'),'APOP(JXG,JYG,JZG)=',
   *                          APOP(JXG,JYG,JZG)
                              WRITE(13,'(A20,D25.17)'),'GPOP(JXG,JYG,JZG)='
   *                          ,GPOP(JXG,JYG,JZG)

                      END IF



C SPLIT TOTAL ABSORBED ENERGY INTO PUMP BEAMLETS --------------------
C                     PUMP Y BEAMLETS
                      DO JP=3,4
                       IF      ( (JBX1PY(JP,JXG,JYG).GT.0).AND.
   *                             (JBZ1PY(JP,JZG,JYG).GT.0).AND.
   *                             (EABS.GT.0.D-50) ) THEN
                        DO JBX=JBX1PY(JP,JXG,JYG),JBX2PY(JP,JXG,JYG)
                         DO JBZ=JBZ1PY(JP,JZG,JYG),JBZ2PY(JP,JZG,JYG)


                              EABSPU=(EPUMPY(JP,JBX,JBZ,JY)/EPUMPTOTIN)
   *                                  *EABS*FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG)
                              EPUMPY(JP,JBX,JBZ,JY)=
   *                                          (EPUMPY(JP,JBX,JBZ,JY)-EABSPU)
```

```
                            IF (DUMPOU.EQ.8) THEN
                             WRITE(13,'(A5,I4,A5,I4,A5,I4,A11)'),
  *                               'JP=',JP,'JBX=',JBX,
  *                               'JBZ=',JBZ,'----------'
                             WRITE(13,'(A20,D15.7)'),'EABSPU=',
  *                               EABSPU
                             WRITE(13,'(A20,D15.7)'),'EPUMPY()='
  *                               ,EPUMPY(JP,JBX,JBZ,JY)
                             END IF

                         END DO
                        END DO

                       END IF
                      END DO

C                     PUMP Z BEAMLETS
                      DO IP=1,2
                       IF    ( (IBX1PZ(IP,JXG,JZG).GT.0).AND.
  *                           (IBY1PZ(IP,JYG,JZG).GT.0).AND.
  *                           (EABS.GT.0.D-50) ) THEN
                        DO IBX=IBX1PZ(IP,JXG,JZG),IBX2PZ(IP,JXG,JZG)
                         DO IBY=IBY1PZ(IP,JYG,JZG),IBY2PZ(IP,JYG,JZG)


                             EABSPU=(EPUMPZ(IP,IBX,IBY,JZ)/EPUMPTOTIN)
  *                               *EABS*FVOLPZ(IP,IBX,IBY,JXG,JYG,JZG)
                                        EPUMPZ(IP,IBX,IBY,JZ)=
  *                                     (EPUMPZ(IP,IBX,IBY,JZ)-EABSPU)

                                        IF (DUMPOU.EQ.8) THEN
                                         WRITE(13,'(A5,I4,A5,I4,A5,I4,A11)'),
  *                                           'IP=',IP,'IBX=',IBX,
  *                                           'IBY=',IBY,'----------'
                                         WRITE(13,'(A20,D15.7)'),'EABSPU=',
  *                                            EABSPU
                                         WRITE(13,'(A20,D15.7)'),'EPUMPZ()='
  *                                           ,EPUMPZ(IP,IBX,IBY,JZ)
                                        END IF

                         END DO
                        END DO
                       END IF
                      END DO

                 END DO
            END DO

C END OF JY(JYG) LOOP
      END DO

      RETURN
      END
```

……

```
      SUBROUTINE AMPCRT01

      INCLUDE 'Lascrt01.inc'

C LOCAL DECLARATIONS ---------------------------------------------

      DOUBLE PRECISION GPOPINV,APOPTEMP,GPOPTEMP
      DOUBLE PRECISION ESEEDTOTIN,CSEED,SATFRC,PARENTH,LNARG,EGAIN
      DOUBLE PRECISION ESEEDTOTOUT,DELTAGPOP,EGAINPU
      DOUBLE PRECISION ESATSEEDLOCAL,XFILL,YFILL

C LOOP THROUGH THE Z-ZONES IN THE GAIN MEDIUM -----------------------

      DO IZ=N1Z+1,N1Z+N2Z

         IZG=IZ-N1Z

         DO IX=N1X+1,N1X+N2X
            IXG=IX-N1X
            DO IY=N1Y+1,N1Y+N2Y
               IYG=IY-N1Y

C CALCULATE THE TOTAL SEED AND ASE ENERGY ENTERING A CUBIC GAIN ZONE.

               ESEEDTOTIN=0.D0

               DO IP=1,NPASS
                  DO IBX=IBX1S(IP,IXG,IZG),IBX2S(IP,IXG,IZG)
                     DO IBY=IBY1S(IP,IYG,IZG),IBY2S(IP,IYG,IZG)

                        ESEEDTOTIN=ESEEDTOTIN+
     *                  FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)*
     *                  ( ESEED(IP,IBX,IBY,IZ) +
     *                    EASE(IP,IBX,IBY,IZ) )
                     END DO
                  END DO
               END DO



C              CALCULATE GPOPINV. THIS IS THE DIFFERENCE BETWEEN
C              UPPER STATE POPULATION DESNITY AND 'TARGET' STATE
C              POPULATION DENSITY

               APOP(IXG,IYG,IZG)=DENSNUM-GPOP(IXG,IYG,IZG)

               IF (SYSTEMLEVELS.EQ.-3) THEN

                  IF ( GPOP(IXG,IYG,IZG) .GT.
     *             APOP(IXG,IYG,IZG) ) THEN
                     GPOPINV=GPOP(IXG,IYG,IZG) -
     *               APOP(IXG,IYG,IZG)
                  ELSE
                     GPOPINV=0.D0
                  END IF
```

```
                ELSE
                        GPOPINV=GPOP(IXG,IYG,IZG)
                END IF

                IF (GPOPINV.LE.(0.D0)) THEN
                        ESEEDTOTOUT=ESEEDTOTIN
                        GOTO 3737
                END IF

C               CALCULATE ESATSEEDLOCAL------------------------------
C               DIFFERENT FROM ESATSEED FOR INCOMPLETELY
C               FILLED CELLS
C               THIS ACCOUNTS FOR XBEAMWIDTH.LT.DX
C                       AND/OR YBEAMWIDTH.LT.DY

                XFILL=1.D0
                YFILL=1.D0
                ESATSEEDLOCAL=ESATSEED
                IF (XBEAMWIDTH(1).LT.DX) THEN
                        XFILL=((XBEAMWIDTH(1))/DX)
                END IF
                IF (YBEAMWIDTH(1).LT.DY) THEN
                        YFILL=((YBEAMWIDTH(1))/DY)
                END IF
                ESATSEEDLOCAL=XFILL*YFILL*ESATSEED


C               CALCULATE THE GAIN --------------------------------

                CSEED=SIGG*DZ*GPOPINV
                SATFRC=ESEEDTOTIN/ESATSEEDLOCAL

                IF (SATFRC.LT.(1.D-4)) THEN
C                       UNSATURATED CALCULATION
                        ESEEDTOTOUT=ESEEDTOTIN*(DEXP(CSEED))
                        PARENTH=0.D0
                        LNARG=0.D0

                ELSE
C                       SATURTATED CALCULATION. USE EQN 3.56, D/R
                        PARENTH=1.D0-DEXP(SATFRC)
                        LNARG=1.D0-(DEXP(CSEED))*PARENTH
                        ESEEDTOTOUT=ESATSEEDLOCAL*DLOG(LNARG)
                END IF

 3737           CONTINUE


                EGAIN=ESEEDTOTOUT-ESEEDTOTIN

C WRITE DATA TO SEEDCALC.CSV ----------------------------------------

                IF ( (SEEDCALCSW.GT.0).AND.
     *            (MOD(JT,SEEDCALCSW).EQ.0).AND.(IXG.EQ.IXGMID)
     *            .AND.(IYG.EQ.IYGMID).AND.(IZG.EQ.IZGMID) ) THEN
```

```
                         WRITE(26,'(9(A3,E13.5))') ,',', TIME ,',',
     *                   GPOPINV ,',',
     *                   ESEEDTOTIN ,',', CSEED ,',', SATFRC ,',',
     *       PARENTH ,',', LNARG ,',', EGAIN ,',',
     *                   ESEEDTOTOUT

                    END IF

C WRITE CALC'S TO DUMP.OUT IF DUMPOU EQ 7 ---------------------------

                    IF (DUMPOU.EQ.7) THEN
                         WRITE(13,'(A10)'),'--------'
                         WRITE(13,'(3(A6,I3))'),'IXG= ',IXG,'IYG= ',IYG
     *                   ,'IZG= ',IZG
                         WRITE(13,'(A14,D13.5)'),'ESEEDTOTIN=',ESEEDTOTIN
                         WRITE(13,'(A14,D25.17)'),'IP GPOPINV=',GPOPINV
                         WRITE(13,'(A20,D25.17)'),'IP GPOP()=',
     *                   GPOP(IXG,IYG,IZG)
                         WRITE(13,'(A20,D25.17)'),'IP APOP()=',
     *                   APOP(IXG,IYG,IZG)

                         WRITE(13,'(A20,D13.5)'),'DX=',DX
                         WRITE(13,'(A20,D13.5)'),'DY=',DY
                         WRITE(13,'(A20,D13.5)'),'XBEAMWIDTH(1)=',
     *                   XBEAMWIDTH(1)
                         WRITE(13,'(A20,D13.5)'),'YBEAMWIDTH(1)=',
     *                   YBEAMWIDTH(1)



                         WRITE(13,'(A20,D13.5)'),'XFILL=',XFILL
                         WRITE(13,'(A20,D13.5)'),'YFILL=',YFILL


                         WRITE(13,'(A20,D13.5)'),'ESATSEEDLOCAL='
     *                   ,ESATSEEDLOCAL

                         WRITE(13,'(A14,D13.5)'),'CSEED=',CSEED
                         WRITE(13,'(A14,D13.5)'),'SATFRC=',SATFRC

                         IF (SATFRC.LT.(1.D-4)) THEN
                              WRITE(13,'(A20)'),'**SM. SIGNAL GAIN'
                         ELSE
                              WRITE(13,'(A20)'),'**LG. SIGNAL GAIN'
                         END IF

                         WRITE(13,'(A14,D13.5)'),'PARENTH=',PARENTH
                         WRITE(13,'(A14,D13.5)'),'LNARG=',LNARG

                    WRITE(13,'(A14,D13.5)'),'ESEEDTOTOUT=',ESEEDTOTOUT
                         WRITE(13,'(A14,D13.5)'),'EGAIN=',EGAIN
                         WRITE(13,'(A14,D13.5)'),'ESEEDTOTIN=',ESEEDTOTIN
                    END IF

C CHECK IF GAINED ENERGY WOULD CAUSE GPOP TO BE DEPLETED COMPLETELY ---
```

```
C ALSO UPDATE POPULATIONS ----------------------------------------



C BOTH APOPTEMP AND GPOPTEMP ARE CHECKED IN CASE OF A DP ERROR.
                  APOPTEMP=APOP(IXG,IYG,IZG)+(EGAIN*KSEED/DA)
                  GPOPTEMP=GPOP(IXG,IYG,IZG)-(EGAIN*KSEED/DA)

                  IF (   (APOPTEMP.GE.DENSNUM )
     *            .OR.(GPOPTEMP.LE.(0.D0)   ) ) THEN

                        DELTAGPOP=GPOP(IXG,IYG,IZG)
                        EGAIN=DELTAGPOP*DA/KSEED
                        GPOP(IXG,IYG,IZG)=0.D0
                        APOP(IXG,IYG,IZG)=DENSNUM

                        IF (DUMPOU.EQ.7) THEN
                              WRITE(13,'(A23)'),'POP UPDATE OVERRIDDEN'
                        END IF

                  ELSE
C                       GPOP(IXG,IYG,IZG)=GPOP(IXG,IYG,IZG)
C    *                  -(EGAIN*KSEED/DA)
C                       APOP(IXG,IYG,IZG)=DENSNUM-GPOP(IXG,IYG,IZG)

                        APOP(IXG,IYG,IZG)=APOPTEMP
                        GPOP(IXG,IYG,IZG)=DENSNUM-APOP(IXG,IYG,IZG)

                  END IF


                  IF (DUMPOU.EQ.7) THEN
                        WRITE(13,'(A25,D13.5)'),'EGAIN AFTER LIN. ABS.=',
     *                  EGAIN
                        WRITE(13,'(A14,D25.17)'),'GPOP()='
     *                  ,GPOP(IXG,IYG,IZG)
                        WRITE(13,'(A14,D25.17)'),'APOP()='
     *                  ,APOP(IXG,IYG,IZG)
                  END IF

C SPLIT TOTAL ABSORBED ENERGY INTO PUMP BEAMLETS --------------------

                  DO IP=1,NPASS
                        IF      ( (IBX1S(IP,IXG,IZG).GT.0).AND.
     *                   (IBY1S(IP,IYG,IZG).GT.0)
     *                  .AND.(EGAIN.GT.0.D-50) ) THEN


                              DO IBX=IBX1S(IP,IXG,IZG),IBX2S(IP,IXG,IZG)
                               DO IBY=IBY1S(IP,IYG,IZG),IBY2S(IP,IYG,IZG)


        EGAINPU=(ESEED(IP,IBX,IBY,IZ)/ESEEDTOTIN)
```

```
*
     *EGAIN*FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)

                                                 ESEED(IP,IBX,IBY,IZ)=
*                                                ESEED(IP,IBX,IBY,IZ)+EGAINPU

                           IF (DUMPOU.EQ.7) THEN
                                  WRITE(13,'(A5,I4,A5,I4,A5,I4,A11)'),
*                                        'IP=',IP,'IBX=',IBX,
*                                        'IBY=',IBY,'---------'
                                  WRITE(13,'(A20,D15.7)'),'EGAINPU=',EGAINPU
                                  WRITE(13,'(A20,D15.7)'),'ESEED()='
*                                 ,ESEED(IP,IBX,IBY,IZ)

                           END IF



     EGAINPU=(EASE(IP,IBX,IBY,IZ)/ESEEDTOTIN)
*
     *EGAIN*FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)
                                                 EASE(IP,IBX,IBY,IZ)=
*                                                EASE(IP,IBX,IBY,IZ)+EGAINPU

                           IF (DUMPOU.EQ.7) THEN
                                  WRITE(13,'(A20,D15.7)'),'EGAINPU=',EGAINPU
                                  WRITE(13,'(A20,D15.7)'),'EASE()='
*                                 ,EASE(IP,IBX,IBY,IZ)

                           END IF




                                            END DO
                                           END DO

                                  END IF
                           END DO

C REDUCE THE SEED ENERGY DUE TO ALPHAG ------------------------------
C ALPHAG DOES NOT CHANGE THE ASE ENERGY

                           IF (ALPHAG.GT.(0.D0)) THEN

                            DO IP=1,NPASS
                             IF     ( (IBX1S(IP,IXG,IZG).GT.0).AND.
*                             (IBY1S(IP,IYG,IZG).GT.0) ) THEN

                                  DO IBX=IBX1S(IP,IXG,IZG),IBX2S(IP,IXG,IZG)
                                   DO IBY=IBY1S(IP,IYG,IZG),IBY2S(IP,IYG,IZG)

                                            ESEED(IP,IBX,IBY,IZ)=ESEED(IP,IBX,IBY,IZ)*
*                                          DEXP(-1.D0*DZ*ALPHAG)

                                           END DO
```

```
                                    END DO

                                   END IF
                               END DO

                              END IF


                          END DO
                     END DO

C END OF IZ(IZG) LOOP
      END DO


      RETURN
      END
.........................................................................................................................
```

```
        DOUBLE PRECISION FUNCTION
CALCVOL(DELTAW,UCMAX,UCMIN,VCMAX,VCMIN,
   *
        UBMAX,UBMIN,VBMAX,VBMIN,DUMPOU)


        DOUBLE PRECISION DELTAW,UCMAX,UCMIN,VCMAX,VCMIN
        DOUBLE PRECISION UBMAX(2),UBMIN(2),VBMAX(2),VBMIN(2)
        DOUBLE PRECISION BASE1,MIDSECT,BASE2,LU,LV
        DOUBLE PRECISION UMIDMAX,UMIDMIN,VMIDMAX,VMIDMIN
        INTEGER DUMPOU

C
C THIS FUNCTION CALCULATES THE VOLUME OF A PRISMATOID HAVING
C PARALLEL BASES IN THE UW PLANE AND A THICKNESS DELTAW.
C
C DUMMY CO-ORDINATE SYSTEM:
C W = DIRECTION OF PROPAGATION
C U,V = TRANSVERSE CO-ORDINATES
C

C
C VARIABLES USED IN THE FUNCTION CALCVOL.F
C
C DELTAW =     THICKNESS OF PRISMATOID IN W DIRECTION
C                  (W = DIRECTION OF PROPAGATION)
C UCMAX,UCMIN = CELL LIMITS IN THE U DIRECTION
C VCMAX,VCMIN = CELL LIMITS IN THE V DIRECTION
C UBMAX,UBMIN = BEAMLET LIMITS IN THE U DIRECTION
C VBMAX,VBMIN = BEAMLET LIMITS IN THE V DIRECTION
C
C NOTE: FOR BEAMLET LIMIT ARRAYS, THE (1) ENTRY CORRESPONDS TO THE
C     FRONT FACE (SMALLER W), AND THE (2) ENTRY CORRESPONDS TO
C     THE BACK FACE (LARGER W).
C
C



C       AREA OF BASE1 (FRONT)

        LU=DMIN1( UCMAX,UBMAX(1) ) -
     * DMAX1( UCMIN,UBMIN(1) )
        LV=DMIN1( VCMAX,VBMAX(1) ) -
     * DMAX1( VCMIN,VBMIN(1) )

        BASE1=LU*LV


        IF (DUMPOU.EQ.2) THEN
                WRITE(13,'(A40)'),'=====FCN.CALCVOL====='
                WRITE(13,'(A40,D15.5)'),'LU=',LU
                WRITE(13,'(A40,D15.5)'),'LV=',LV
                WRITE(13,'(A40,D15.5)'),'BASE1=',BASE1
        END IF
```

```
C    CALCULATE MID-POINTS.

       UMIDMAX=0.5D0*( UBMAX(1) + UBMAX(2) )
       UMIDMIN=0.5D0*( UBMIN(1) + UBMIN(2) )
       VMIDMAX=0.5D0*( VBMAX(1) + VBMAX(2) )
       VMIDMIN=0.5D0*( VBMIN(1) + VBMIN(2) )


C      AREA OF MID-SECTION

       LU=DMIN1( UCMAX,UMIDMAX ) -
     * DMAX1( UCMIN,UMIDMIN )
       LV=DMIN1( VCMAX,VMIDMAX ) -
     * DMAX1( VCMIN,VMIDMIN )

       MIDSECT=LU*LV

       IF (DUMPOU.EQ.2) THEN
              WRITE(13,'(A40,D15.5)'),'UMIDMAX=',UMIDMAX
              WRITE(13,'(A40,D15.5)'),'UMIDMIN=',UMIDMIN
              WRITE(13,'(A40,D15.5)'),'VMIDMAX=',VMIDMAX
              WRITE(13,'(A40,D15.5)'),'VMIDMIN=',VMIDMIN
              WRITE(13,'(A40,D15.5)'),'LU=',LU
              WRITE(13,'(A40,D15.5)'),'LV=',LV
              WRITE(13,'(A40,D15.5)'),'MIDSECT=',MIDSECT
       END IF


C      AREA OF BASE2 (BACK)

       LU=DMIN1( UCMAX,UBMAX(2) ) -
     * DMAX1( UCMIN,UBMIN(2) )
       LV=DMIN1( VCMAX,VBMAX(2) ) -
     * DMAX1( VCMIN,VBMIN(2) )

       BASE2=LU*LV

       IF (DUMPOU.EQ.2) THEN
              WRITE(13,'(A40,D15.5)'),'LU=',LU
              WRITE(13,'(A40,D15.5)'),'LV=',LV
              WRITE(13,'(A40,D15.5)'),'BASE2=',BASE2
              WRITE(13,'(A40)'),'===================='
       END IF


C    CHECK FOR 'NEGATIVE AREA' (NO OVERLAP)

       IF ( (BASE1.LT.(0.D0)).OR.(BASE2.LT.(0.D0)).OR.
     *     (MIDSECT.LT.(0.D0)) ) THEN

              CALCVOL=0.D0
```

```
        ELSE

                CALCVOL=(DELTAW/6.D0)*(BASE1+(4.D0*MIDSECT)+BASE2)

        END IF


        RETURN
        END
```

....

```
      SUBROUTINE GEOCRT02(PASSNUMBER,PASSTYPE,
     *  NBU,NBV,WDIRN,DUMPPRN
     *  ,DW,UPROPANGLE,VPROPANGLE,UDIVGANGLE,VDIVGANGLE,UCENTRE
     *  ,VCENTRE,UWIDTH,VWIDTH,N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W)


      INCLUDE 'Lascrt01.inc'

C
C     THIS IS THE INPUT/OUTPUT DUMMY ARGUMENT LIST
C
C     (PASSNUMBER,PASSTYPE,NBU,NBV,WDIRN,DUMPPRN
C     ,DW,UPROPANGLE,VPROPANGLE,UDIVGANGLE,VDIVGANGLE,UCENTRE,VCENTRE
C     ,UWIDTH,VWIDTH,N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W)


C THIS 'SLAVE' SUBROUTINE PERFORMS THE REQUIRED GEOMETRICAL
C CALCULATIONS FOR THE VARIOUS PUMP AND SEED PASSES.
C
C THIS SUBROUTINE IS TO BE CALLED SEPARATELY FOR EACH PASS
C
C


C DUMMY CO-ORDINATE SYSTEM:
C W = DIRECTION OF PROPAGATION
C U,V = TRANSVERSE CO-ORDINATES
C
C MAKE SURE THAT DUMMY ARGUMENT NAMES ARE NOT REPEATED ANYWHERE ELSE!
C
C INPUT DUMMY ARGUMENTS: ------------------------------------------
C
C DW=               DX,DY OR DZ
C WINIT=            W INITIAL
C WDIRN=            W DIRECTION (POSITIVE OR NEGATIVE)
C                   WDIRN >= 0 ---> PROPAGATION IN POSITIVE DIRECTION
C                   WDIRN < 0 ----> PROPAGATION IN NEGATIVE DIRECTION
C


C N1W,N2W,N3W=      NUMBER OF W CELLS (AIR,GAIN,AIR)
C N1U,N2U,N3U=      NUMBER OF U CELLS (AIR,GAIN,AIR)
C N1V,N2V,N3V=      NUMBER OF V CELLS (AIR,GAIN,AIR)
C
C UPROPANGLE=       U PROPAGATION ANGLE
C VPROPANGLE=       V PROPAGATION ANGLE
C
C UDIVGANGLE=       U DIVERGENCE ANGLE
C VDIVGANGLE=       V DIVERGENCE ANGLE
C
C NBU=         NUMBER OF U BEAMLETS
C NBV=         NUMBER OF V BEAMLETS
C
C UCENTRE=     U BEAM CENTRE (START OF PASS)
C VCENTRE=     V BEAM CENTRE (START OF PASS)
C
C UWIDTH=           U BEAM WIDTH (START OF PASS)
```

```
C VWIDTH=               V BEAM WIDTH (START OF PASS)
C
C
C OUTPUT DUMMY ARGUMENTS: ----------------------------------------------
C
C UBEAMANGLE(0:IBU)=U BEAMLET EDGE ANGLES
C                                      (E.G. THETABEAMX,THETABEAMY, ETC.)
C VBEAMANGLE(0:IBV)=V BEAMLET EDGE ANGLES
C UVERTEX(0:IBU,0:IW)=U VERTEXES (E.G. XVS,YVS)
C VVERTEX(0:IBV,0:IW)=V VERTEXES
C IBU1(IUG,IWG)=        FIRST U BEAMLET TO OVERLAP A CELL (E.G. IBX1S, ETC.)
C IBU2(IUG,IWG)=        LAST U BEAMLET CELL TO OVERLAP A CELL
C IBV1(IVG,IWG)=        FIRST V BEAMLET CELL TO OVERLAP A CELL
C IBV2(IVG,IWG)=        LAST V BEAMLET TO OVERLAP A CELL
C FVOLUVW(IBU,IBV,IUG,IVG,IWG)=
C                                 VOLUME OVERLAP FRACTIONS (E.G. FVOLZ ETC.)
C BEAMLETVOL(IBU,IBV,IWG) = VOLUME OF BEAMLET
C


C VARIABLE NAME CONVERSION TABLE:  (FOR PROPAGATION IN THE Z DIRECTION)
C
C   X ----->U
C   Y ----->V
C   Z ----->W
C


C NOTE:IBU1,FVOL, ETC. ONLY NEED TO BE DONE FOR THE GAIN CELLS, NOT
C THE AIR CELLS.

C REMEMEBR ARRAY DIMENSIONING!

C DECLARATION OF INPUT AND OUTPUT VARIABLES ----------------------------

        INTEGER PASSNUMBER
        CHARACTER*4 PASSTYPE

        INTEGER N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W
        INTEGER NBU,NBV,WDIRN,DUMPPRN
        DOUBLE PRECISION DW,UPROPANGLE,VPROPANGLE
        DOUBLE PRECISION UDIVGANGLE,VDIVGANGLE,UCENTRE,VCENTRE
        DOUBLE PRECISION UWIDTH,VWIDTH

        INTEGER IBU1(N2U,N2W),IBU2(N2U,N2W)
        INTEGER IBV1(N2V,N2W),IBV2(N2V,N2W)
        DOUBLE PRECISION UBEAMANGLE(0:NBMAX),VBEAMANGLE(0:NBMAX)
        DOUBLE PRECISION UREFRANGLE(0:NBMAX),VREFRANGLE(0:NBMAX)

        DOUBLE PRECISION UVERTEX(0:NBMAX,0:NZMAX),VVERTEX(0:NBMAX,0:NZMAX)
        DOUBLE PRECISION FVOLUVW(NBMAX,NBMAX,N2U,N2V,N2W)
        DOUBLE PRECISION BEAMLETVOL(NBMAX,NBMAX,N2W)


C DECLARATION OF 'TEMPORARY' VARIABLES ------------------------------
```

```
          INTEGER IZC,J,K,FIRST,LAST,COUNT,PTR,IWC
          INTEGER IBU,IBV,IU,IV,IW,IUG,IVG,IWG
          INTEGER NUTOTAL,NVTOTAL,NWTOTAL,NUG,NVG,NWG

          DOUBLE PRECISION UHI,ULO,VHI,VLO,FVOLTEMP,TOTVOL
          DOUBLE PRECISION UHICHECK,ULOCHECK,VHICHECK,VLOCHECK
          DOUBLE PRECISION UHICHECK2,ULOCHECK2,VHICHECK2,VLOCHECK2
          DOUBLE PRECISION WCUT(10),VOL,SUBDW,HOLD

          DOUBLE PRECISION UCMAX,UCMIN,VCMAX,VCMIN
          DOUBLE PRECISION UBMAX(2),UBMIN(2),VBMAX(2),VBMIN(2)

          DOUBLE PRECISION DBU,UEDGE,URATIO,WDIST,DBV,VEDGE,VRATIO
          DOUBLE PRECISION WDIST1,WDIST2,WDIST3


C CONVERSION OF GEOMETRICAL DATA FOR BEAMS PROPAGATING IN THE --------
C NEGATIVE W DIRECTION (FOR WDIRN < 0)

          IF (WDIRN.LT.0) THEN

                UCENTRE=UCENTRE-(DW*NWTOTAL*TAN(UPROPANGLE))
                VCENTRE=VCENTRE-(DW*NWTOTAL*TAN(VPROPANGLE))

                UWIDTH=UWIDTH+(2*DW*NWTOTAL*TAN(0.5D0*UDIVGANGLE))
                VWIDTH=VWIDTH+(2*DW*NWTOTAL*TAN(0.5D0*VDIVGANGLE))

                UDIVGANGLE=(-1.D0)*UDIVGANGLE
                VDIVGANGLE=(-1.D0)*VDIVGANGLE

          END IF

C CALCULATION OF DERIVED QUANTITIES --------------------------------

          DBU=UWIDTH/(DBLE(NBU))
          DBV=VWIDTH/(DBLE(NBV))

          NUTOTAL=N1U+N2U+N3U
          NVTOTAL=N1V+N2V+N3V
          NWTOTAL=N1W+N2W+N3W

          NUG=N2U
          NVG=N2V
          NWG=N2W


          IF (DUMPPRN.EQ.2) THEN
                WRITE(13,'(A20,D15.5)'),'UCENTRE=',UCENTRE
                WRITE(13,'(A20,D15.5)'),'UWIDTH=',UWIDTH
                WRITE(13,'(A20,D15.5)'),'UDIVGANGLE=',UDIVGANGLE
                WRITE(13,'(A20,D15.5)'),'UPROPANGLE=',UPROPANGLE

                WRITE(13,'(A20,D15.5)'),'VCENTRE=',VCENTRE
                WRITE(13,'(A20,D15.5)'),'VWIDTH=',VWIDTH
                WRITE(13,'(A20,D15.5)'),'VDIVGANGLE=',VDIVGANGLE
```

```
                    WRITE(13,'(A20,D15.5)'),'VPROPANGLE=',VPROPANGLE

                    WRITE(13,'(A20,I5)'),'NBU=',NBU
                    WRITE(13,'(A20,I5)'),'NBV=',NBV

                    WRITE(13,'(A20,D15.5)'),'DBU=',DBU
                    WRITE(13,'(A20,D15.5)'),'DBV=',DBV


                    WRITE(13,'(A20,D15.5)'),'DW=',DW
                    WRITE(13,'(A20,I5)'),'WDIRN=',WDIRN
                    WRITE(13,'(3(A15,I5))'),'N1U=',N1U,
    *               'N2U=',N2U,'N3U=',N3U
                    WRITE(13,'(3(A15,I5))'),'N1V=',N1V,
    *               'N2V=',N2V,'N3V=',N3V
                    WRITE(13,'(3(A15,I5))'),'N1W=',N1W,
    *               'N2W=',N2W,'N3W=',N3W


          END IF


C CALCULATE DIRECTIONS OF BEAMLET EDGES (U AND V). -------------------
C FUTURE: BEAM REFRACTION EFFECTS.

          IF (DUMPPRN.EQ.2) THEN
                    WRITE(13,'(A45)'),' '
                    WRITE(13,'(A45)'),'---CALC. OF UVERTEX(IBU,0),UBEAMANGLE---'
          END IF



          DO IBU=0,NBU

                    UEDGE=(-0.5D0*UWIDTH)+DBU*DBLE(IBU)
                    UVERTEX(IBU,0)=UCENTRE+UEDGE
                    URATIO=UEDGE/(0.5D0*UWIDTH)

          UBEAMANGLE(IBU)=UPROPANGLE+ATAN(URATIO*TAN(0.5D0*UDIVGANGLE))
                    UREFRANGLE(IBU)=ASIN(SIN(UBEAMANGLE(IBU))/REFRINDEX)

                    IF (DUMPPRN.EQ.2) THEN
                              WRITE(13,'(A4,I3,2(A8,D10.3),3(A14,D10.3))'),
    *                         'IBU=',IBU,'UEDGE=',UEDGE,'URATIO=',URATIO,
    *                         'UVERTEX(,)=',UVERTEX(IBU,0),
    *                         'UBEAMANGLE()=',UBEAMANGLE(IBU),
    *                         'UREFRANGLE()=',UREFRANGLE(IBU)

                    END IF

          END DO


          IF (DUMPPRN.EQ.2) THEN
                    WRITE(13,'(A45)'),' '
```

```
            WRITE(13,'(A45)'),'----CALC. OF VVERTEX(IBV,0),VBEAMANGLE----'
      END IF


      DO IBV=0,NBV

            VEDGE=(-0.5D0*VWIDTH)+DBV*DBLE(IBV)
            VVERTEX(IBV,0)=VCENTRE+VEDGE
            VRATIO=VEDGE/(0.5D0*VWIDTH)


      VBEAMANGLE(IBV)=VPROPANGLE+ATAN(VRATIO*TAN(0.5D0*VDIVGANGLE))
            VREFRANGLE(IBV)=ASIN(SIN(VBEAMANGLE(IBV))/REFRINDEX)

            IF (DUMPPRN.EQ.2) THEN
                  WRITE(13,'(A4,I3,2(A8,D10.3),3(A14,D10.3))'),
    *             'IBV=',IBV,'VEDGE=',VEDGE,'VRATIO=',VRATIO,
    *             'VVERTEX(,)=',VVERTEX(IBV,0),
    *             'VBEAMANGLE()=',VBEAMANGLE(IBV),
    *             'VREFRANGLE()=',VREFRANGLE(IBV)


            END IF

      END DO


C CALCULATE LOCATIONS OF BEAMLET VERTEXES. ---------------------------

      IF (DUMPPRN.EQ.2) THEN
            WRITE(13,'(A45)'),' '
            WRITE(13,'(A45)'),'----CALC. OF UVERTEX,VVERTEX---------'
      END IF


      DO IW=1,NWTOTAL

C CALCULTION OF WDIST MUST CHANGE TO ACCOUNT FOR N0NUNIFORM CELLS
C           WDIST=DW*DBLE(IW)


C     AIR SPACE BEFORE THE GAIN REGION

            IF (IW.LE.N1W) THEN
                  WDIST1=DWAIR*DBLE(IW)


             DO IBU=0,NBU


      UVERTEX(IBU,IW)=UVERTEX(IBU,0)+WDIST1*TAN(UBEAMANGLE(IBU))

                  IF (DUMPPRN.EQ.2) THEN
```

```
                          WRITE(13,'(2(A5,I3),2(A15,D11.4))'),
     *                    'IW=',IW,'IBU=',IBU,'WDIST1=',WDIST1,
     *                    'UVERTEX(,)=',UVERTEX(IBU,IW)
                    END IF

              END DO


              DO IBV=0,NBV


      VVERTEX(IBV,IW)=VVERTEX(IBV,0)+WDIST1*TAN(VBEAMANGLE(IBV))

                    IF (DUMPPRN.EQ.2) THEN
                          WRITE(13,'(2(A5,I3),2(A15,D11.4))'),
     *                    'IW=',IW,'IBV=',IBV,'WDIST1=',WDIST1,
     *                    'VVERTEX(,)=',VVERTEX(IBV,IW)
                    END IF


              END DO

C     INSIDE THE GAIN REGION

             ELSE IF (IW.LE.(N1W+N2W)) THEN
                    WDIST1=(DWAIR*DBLE(N1W))
                    WDIST2=(DW*DBLE(IW-N1W))


              DO IBU=0,NBU

                    UVERTEX(IBU,IW)=UVERTEX(IBU,0) +
     *              WDIST1*TAN(UBEAMANGLE(IBU)) +
     *              WDIST2*TAN(UREFRANGLE(IBU))


                    IF (DUMPPRN.EQ.2) THEN
                          WRITE(13,'(2(A5,I3),3(A15,D11.4))'),
     *                    'IW=',IW,'IBU=',IBU,'WDIST1=',WDIST1,
     *                    'WDIST2=',WDIST2,
     *                    'UVERTEX(,)=',UVERTEX(IBU,IW)
                    END IF

              END DO


              DO IBV=0,NBV

                    VVERTEX(IBV,IW)=VVERTEX(IBV,0) +
     *              WDIST1*TAN(VBEAMANGLE(IBV)) +
     *              WDIST2*TAN(VREFRANGLE(IBV))

                    IF (DUMPPRN.EQ.2) THEN
                          WRITE(13,'(2(A5,I3),3(A15,D11.4))'),
     *                    'IW=',IW,'IBV=',IBV,'WDIST1=',WDIST1,
     *                    'WDIST2=',WDIST2,
```

```
     *                   'VVERTEX(,)=',VVERTEX(IBV,IW)
                   END IF


              END DO

C     AIR SPACE AFTER THE GAIN REGION

             ELSE
                   WDIST1=DWAIR*DBLE(N1W)
                   WDIST2=DW*DBLE(N2W)
                   WDIST3=DWAIR*DBLE(IW-(N1W+N2W))

              DO IBU=0,NBU

                   UVERTEX(IBU,IW)=UVERTEX(IBU,0) +
     *             WDIST1*TAN(UBEAMANGLE(IBU)) +
     *             WDIST2*TAN(UREFRANGLE(IBU)) +
     *             WDIST3*TAN(UBEAMANGLE(IBU))

                   IF (DUMPPRN.EQ.2) THEN
                         WRITE(13,'(2(A5,I3),4(A15,D11.4))'),
     *                   'IW=',IW,'IBU=',IBU,'WDIST1=',WDIST1,
     *                   'WDIST2=',WDIST2,'WDIST3=',WDIST3,
     *                   'UVERTEX(,)=',UVERTEX(IBU,IW)
                   END IF

              END DO


              DO IBV=0,NBV

                   VVERTEX(IBV,IW)=VVERTEX(IBV,0) +
     *             WDIST1*TAN(VBEAMANGLE(IBV)) +
     *             WDIST2*TAN(VREFRANGLE(IBV)) +
     *             WDIST3*TAN(VBEAMANGLE(IBV))

                   IF (DUMPPRN.EQ.2) THEN
                         WRITE(13,'(2(A5,I3),4(A15,D11.4))'),
     *                   'IW=',IW,'IBV=',IBV,'WDIST1=',WDIST1,
     *                   'WDIST2=',WDIST2,'WDIST3=',WDIST3,
     *                   'VVERTEX(,)=',VVERTEX(IBV,IW)
                   END IF


              END DO


             END IF

       END DO

C DETERMINE WHICH GAIN CELLS OVERLAP EACH BEAMLET. ------------------

       IF (DUMPPRN.EQ.2) THEN
```

```
                WRITE(13,'(A45)'),' '
                WRITE(13,'(A45)'),'----CALC. OF IBU1,IBU2,IBV1,IBV2-----'
        END IF


        DO IWG=1,NWG

C CONVERT TO 'FULL W' INDEX

                IW=IWG+N1W

                DO IUG=1,NUG

                        IU=IUG+N1U


C ACCOUNT FOR NONUNIFORM CELLS
                        ULO=(DWAIR*DBLE(N1U)) + (DW*DBLE(IUG-1))
                        UHI=ULO+DW
C                       ULO=DW*DBLE(IU-1)
C                       UHI=DW*DBLE(IU)

C                       INITIALIZE ARRAYS TO ZERO.
                        IBU1(IUG,IWG)=0
                        IBU2(IUG,IWG)=0

                        DO IBU=1,NBU

C                               MUST ALSO CHECK 'LEFTMOST' VERTEXES FOR OVERLAP
                                IF ( ( (UVERTEX(IBU,IW).GT.ULO    ).OR.
    *                             (UVERTEX(IBU,IW-1).GT.ULO  ) ) .AND.
    *                      ( (UVERTEX(IBU-1,IW).LT.UHI  ).OR.
    *                             (UVERTEX(IBU-1,IW-1).LT.UHI) ) ) THEN


                                        IBU1(IUG,IWG)=IBU

                                        GOTO 1000

                                END IF
                        END DO

  1000                  IBU2(IUG,IWG)=IBU1(IUG,IWG)

                        DO IBU=IBU1(IUG,IWG),NBU
                                IF(  (UVERTEX(IBU,IW).GE.UHI).AND.
    *                           (UVERTEX(IBU,IW-1).GE.UHI) ) THEN

                                        IBU2(IUG,IWG)=IBU
                                        GOTO 1001

                                END IF
                        END DO

                        IBU2(IUG,IWG)=NBU
```

```
1001                    IF (DUMPPRN.EQ.2) THEN
                                WRITE(13,'(2(A5,I3),2(A5,D15.5),2(A8,I3))'),
     *                          'IUG=',IUG,'IWG=',IWG,'ULO=',ULO,'UHI=',UHI,
     *                          'IBU1()=',IBU1(IUG,IWG),'IBU2()=',IBU2(IUG,IWG)
                        END IF

                END DO



                DO IVG=1,NVG

                        IV=IVG+N1V

C ACCOUNT FOR NONUNIFORM CELLS
                        VLO=(DWAIR*DBLE(N1V)) + (DW*DBLE(IVG-1))
                        VHI=VLO+DW
C                       VHI=DW*DBLE(IV)
C                       VLO=DW*DBLE(IV-1)

C                       INITIALIZE ARRAYS TO ZERO.
                        IBV1(IVG,IWG)=0
                        IBV2(IVG,IWG)=0



                        DO IBV=1,NBV

C                               MUST ALSO CHECK 'LEFTMOST' VERTEXES FOR OVERLAP
                                IF ( ( (VVERTEX(IBV,IW).GT.VLO    ).OR.
     *                            (VVERTEX(IBV,IW-1).GT.VLO  ) ) .AND.
     *                    ( (VVERTEX(IBV-1,IW).LT.VHI  ).OR.
     *                            (VVERTEX(IBV-1,IW-1).LT.VHI) ) ) THEN


                                        IBV1(IVG,IWG)=IBV

                                        GOTO 1002

                                END IF
                        END DO

 1002                   IBV2(IVG,IWG)=IBV1(IVG,IWG)

                        DO IBV=IBV1(IVG,IWG),NBV
                                IF(  (VVERTEX(IBV,IW).GE.VHI).AND.
     *                          (VVERTEX(IBV,IW-1).GE.VHI) ) THEN

                                        IBV2(IVG,IWG)=IBV
                                        GOTO 1003

                                END IF
                        END DO
```

```
                        IBV2(IVG,IWG)=NBV

1003                    IF (DUMPPRN.EQ.2) THEN
                                WRITE(13,'(2(A5,I3),2(A5,D15.5),2(A8,I3))'),
     *                          'IVG=',IVG,'IWG=',IWG,'VLO=',VLO,'VHI=',VHI,
     *                          'IBV1()=',IBV1(IVG,IWG),'IBV2()=',IBV2(IVG,IWG)

                        END IF


                END DO


        END DO


C CALCUALTE VOLUME OVERLAP FRACTIONS. -------------------------------

        IF (DUMPPRN.EQ.2) THEN
                WRITE(13,'(A45)'),' '
                WRITE(13,'(A45)'),'--CALC. OF FVOLUVW(IBU,IBV,IUG,IVG,IWG)---'
        END IF


        DO IWG=1,NWG

                IW=IWG+N1W

C               WDIST=DW*DBLE(IW)

C       REVISED CALCULATION OF WDIST

                IF (IW.LE.N1W) THEN
                        WDIST=DWAIR*DBLE(IW)
                ELSE IF (IW.LE.(N1W+N2W)) THEN
                        WDIST=(DWAIR*DBLE(N1W)) + (DW*DBLE(IW-N1W))
                ELSE
                        WDIST=(DWAIR*DBLE(N1W)) + (DW*DBLE(N2W)) +
     *                  ( DWAIR*DBLE(IW-(N1W+N2W)) )
                END IF

C               CALCULATE ALL BEAMLET VOLUMES FOR THIS IW STEP.
C               STORE THE RESULTS IN BEAMLETVOL(IBU,IBV,IWG).

                DO IBU=1,NBU
                        DO IBV=1,NBV

                                        UBMAX(1)=UVERTEX(IBU,IW-1)
                                        UBMIN(1)=UVERTEX(IBU-1,IW-1)
                                        VBMAX(1)=VVERTEX(IBV,IW-1)
                                        VBMIN(1)=VVERTEX(IBV-1,IW-1)

                                        UBMAX(2)=UVERTEX(IBU,IW)
                                        UBMIN(2)=UVERTEX(IBU-1,IW)
```

```
                         VBMAX(2)=VVERTEX(IBV,IW)
                         VBMIN(2)=VVERTEX(IBV-1,IW)

                         UCMAX=MAXVAL(UBMAX)
                         UCMIN=MINVAL(UBMIN)
                         VCMAX=MAXVAL(VBMAX)
                         VCMIN=MINVAL(VBMIN)

                         IF (DUMPPRN.EQ.2) THEN
                              WRITE(13,'(A20)'),'---------------'
                              WRITE(13,'(A10,I10)'),'IBU=',IBU
                              WRITE(13,'(A10,I10)'),'IBV=',IBV
                              WRITE(13,'(A10,I10)'),'IWG=',IWG

                    WRITE(13,'(A20,D15.5)'),'UBMAX(1)=',UBMAX(1)
                    WRITE(13,'(A20,D15.5)'),'UBMIN(1)=',UBMIN(1)

                    WRITE(13,'(A20,D15.5)'),'VBMAX(1)=',VBMAX(1)
                    WRITE(13,'(A20,D15.5)'),'VBMIN(1)=',VBMIN(1)

                    WRITE(13,'(A20,D15.5)'),'UBMAX(2)=',UBMAX(2)
                    WRITE(13,'(A20,D15.5)'),'UBMIN(2)=',UBMIN(2)

                    WRITE(13,'(A20,D15.5)'),'VBMAX(2)=',VBMAX(2)
                    WRITE(13,'(A20,D15.5)'),'VBMIN(2)=',VBMIN(2)
                         END IF


     BEAMLETVOL(IBU,IBV,IWG)=CALCVOL(DW,UCMAX,UCMIN,
  *
     VCMAX,VCMIN,UBMAX,UBMIN,VBMAX,VBMIN,DUMPPRN)


                         IF (DUMPPRN.EQ.2) THEN
                              WRITE(13,'(A25,D15.5)'),
  *                       'BEAMLETVOL(IBU,IBV,IWG)='
  *                           ,BEAMLETVOL(IBU,IBV,IWG)
                              WRITE(13,'(A20)'),''

                         END IF

C                        INITIALIZE FVOLUVW ARRAY TO ZERO
                         DO IUG=1,NUG
                              DO IVG=1,NVG

     FVOLUVW(IBU,IBV,IUG,IVG,IWG)=0.D0
                              END DO
                         END DO


               END DO
          END DO
```

```
                DO IUG=1,NUG

                        IU=IUG+N1U

C ACCOUNT FOR NONUNIFORM CELLS
                        ULO=(DWAIR*DBLE(N1U)) + (DW*DBLE(IUG-1))
                        UHI=ULO+DW
C                       ULO=DW*DBLE(IU-1)
C                       UHI=DW*DBLE(IU)


                        DO IVG=1,NVG

                                IV=IVG+N1V

C ACCOUNT FOR NONUNIFORM CELLS
                        VLO=(DWAIR*DBLE(N1V)) + (DW*DBLE(IVG-1))
                        VHI=VLO+DW

C                               VHI=DW*DBLE(IV)
C                               VLO=DW*DBLE(IV-1)


                                IF (( IBU1(IUG,IWG).LT.1).OR.
   *                            ( IBU2(IUG,IWG).LT.1).OR.
   *                            ( IBV1(IVG,IWG).LT.1).OR.
   *                            ( IBV2(IVG,IWG).LT.1)  ) THEN


                                        GOTO 2000

                                END IF

                                DO IBU=IBU1(IUG,IWG),IBU2(IUG,IWG)
                                        DO IBV=IBV1(IVG,IWG),IBV2(IVG,IWG)


                                        IF (DUMPPRN.EQ.2) THEN
                                                WRITE(13,'(A20)'),'---------------'
                                                WRITE(13,'(A10,I10)'),'IBU=',IBU
                                                WRITE(13,'(A10,I10)'),'IBV=',IBV
                                                WRITE(13,'(A10,I10)'),'IUG=',IUG
                                                WRITE(13,'(A10,I10)'),'IVG=',IVG
                                                WRITE(13,'(A10,I10)'),'IWG=',IWG
                                        END IF


C                                       CHECK WHAT THE SHAPE OF THE OVERLAP IS.

                                ULOCHECK=( UVERTEX(IBU-1,IW)-ULO )*
   *                               ( UVERTEX(IBU-1,IW-1)-ULO )


                                UHICHECK=( UVERTEX(IBU,IW)-UHI )*
   *                                    ( UVERTEX(IBU,IW-1)-UHI )
```

```
                  VHICHECK=( VVERTEX(IBV,IW)-VHI )*
     *               ( VVERTEX(IBV,IW-1)-VHI)

                  VLOCHECK=( VVERTEX(IBV-1,IW)-VLO )*
     *                    ( VVERTEX(IBV-1,IW-1)-VLO )


                  ULOCHECK2=( UVERTEX(IBU-1,IW)-UHI )*
     *                    ( UVERTEX(IBU-1,IW-1)-UHI )


                  UHICHECK2=( UVERTEX(IBU,IW)-ULO )*
     *                    ( UVERTEX(IBU,IW-1)-ULO )


                  VHICHECK2=( VVERTEX(IBV,IW)-VLO )*
     *                         ( VVERTEX(IBV,IW-1)-VLO)

                  VLOCHECK2=( VVERTEX(IBV-1,IW)-VHI )*
     *                    ( VVERTEX(IBV-1,IW-1)-VHI )

C ADD WRITE STATEMENTS HERE.


                         IF (DUMPPRN.EQ.2) THEN
                                WRITE(13,'(A20,D15.5)'),'ULOCHECK=',
     *                          ULOCHECK
                                WRITE(13,'(A20,D15.5)'),'UHICHECK=',
     *                          UHICHECK
                                WRITE(13,'(A20,D15.5)'),'VHICHECK=',
     *                          VHICHECK
                                WRITE(13,'(A20,D15.5)'),'VLOCHECK=',
     *                          VLOCHECK

                                WRITE(13,'(A20,D15.5)'),'ULOCHECK2=',
     *                          ULOCHECK2
                                WRITE(13,'(A20,D15.5)'),'UHICHECK2=',
     *                          UHICHECK2
                                WRITE(13,'(A20,D15.5)'),'VHICHECK2=',
     *                          VHICHECK2
                                WRITE(13,'(A20,D15.5)'),'VLOCHECK2=',
     *                          VLOCHECK2

                         END IF


C                        CASE #1: OVERLAP VOLUME IS A PURE PRISMATOID
C                                 (PARALLEL BASES ARE IN THE XY PLANE).

                         IF ( ( ULOCHECK.GT.(0.D0)  ) .AND.
     *                 ( UHICHECK.GT.(0.D0) ) .AND.
     *                 ( VHICHECK.GT.(0.D0)   ) .AND.
     *                ( VLOCHECK.GT.(0.D0)) .AND.
```

```
     *                  ( ULOCHECK2.GT.(0.D0)  ) .AND.
     *                   ( UHICHECK2.GT.(0.D0) ) .AND.
     *                   ( VHICHECK2.GT.(0.D0)   ) .AND.
     *                   ( VLOCHECK2.GT.(0.D0)) ) THEN

                                 UBMAX(1)=UVERTEX(IBU,IW-1)
                                 UBMIN(1)=UVERTEX(IBU-1,IW-1)
                                 VBMAX(1)=VVERTEX(IBV,IW-1)
                                 VBMIN(1)=VVERTEX(IBV-1,IW-1)

                                 UBMAX(2)=UVERTEX(IBU,IW)
                                 UBMIN(2)=UVERTEX(IBU-1,IW)
                                 VBMAX(2)=VVERTEX(IBV,IW)
                                 VBMIN(2)=VVERTEX(IBV-1,IW)


C                                CALCVOL
                                 FVOLTEMP=CALCVOL(DW,UHI,ULO,VHI,
     *                        VLO,UBMAX,UBMIN,VBMAX,VBMIN,DUMPPRN)



C                           CASE #2: OVERLAP VOLUME IS CALCULATED AS
C                                         ADDITIVE PRISMATOIDS.
C                                         (UP TO 9 SUB-VOLUMES)



                            ELSE

C                       CALCULATE THE 'CUT-LINES'
C                       SEP. 29 /98: CALCULATE THE 4 ADDITIONAL CUT LINES
C                       CHANGE ARRAYS AS NEEDED.


                                 WCUT(1)=WDIST-DW


                                 WCUT(2)=( ULO-UVERTEX(IBU-1,0) )/
     *                                   TAN(UBEAMANGLE(IBU-1))
                                 WCUT(3)=( UHI-UVERTEX(IBU,0) )/
     *                                   TAN(UBEAMANGLE(IBU))
                                 WCUT(4)=( VLO-VVERTEX(IBV-1,0) )/
     *                                   TAN(VBEAMANGLE(IBV-1))

                                 WCUT(5)=( VHI-VVERTEX(IBV,0) )/

     *                                   TAN(VBEAMANGLE(IBV))

                                 WCUT(6)=( ULO-UVERTEX(IBU,0) )/
     *                                   TAN(UBEAMANGLE(IBU))
                                 WCUT(7)=( UHI-UVERTEX(IBU-1,0) )/
     *                                   TAN(UBEAMANGLE(IBU-1))
                                 WCUT(8)=( VLO-VVERTEX(IBV,0) )/
```

```
     *                                  TAN(VBEAMANGLE(IBV))

                                 WCUT(9)=( VHI-VVERTEX(IBV-1,0) )/

     *                                  TAN(VBEAMANGLE(IBV-1))

                                 WCUT(10)=WDIST


C                          USE SELECTION SORT ALGORITHM TO SORT WCUT
C                          ARRAY INTO ASCENDING ORDER.  TEST THIS OUT.

C                          REF. 'STRUCTURED FORTRAN 77 FOR ENGINEERS
C                                AND SCIENTISTS', D.M. ETTER, 3RD ED. (1990)
C                                PP. 205-208


C UNSOTED LIST OF WCUT:

                                 IF (DUMPPRN.EQ.2) THEN
                                  WRITE(13,'(A30)'),'UNSORTED LIST:'
                                  WRITE(13,'(A30,D15.5)'),'WCUT(1)=',WCUT(1)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(2)=',WCUT(2)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(3)=',WCUT(3)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(4)=',WCUT(4)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(5)=',WCUT(5)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(6)=',WCUT(6)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(7)=',WCUT(7)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(8)=',WCUT(8)
                                  WRITE(13,'(A30,D15.5)'),'WCUT(9)=',WCUT(9)
WRITE(13,'(A30,D15.5)'),'WCUT(10)=',WCUT(10)
                                 END IF

C                          RE-ASSIGN THE WCUT'S IF THEY EQUAL
C                          INFINITY OR NAN (FOR STRAIGHT BEAMLETS)
C
C



                                 DO IWC=2,9

                                      IF ( (WCUT(IWC).EQ.NaN).OR.
     *                                 (WCUT(IWC).EQ.Infinity).OR.
     *                                 (WCUT(IWC).EQ.-Infinity) ) THEN

                                          WCUT(IWC)=-23.D0

                                      END IF

                                 END DO
```

```
                                        COUNT=10
                                        LAST=COUNT
                                        DO J=1,COUNT-1
                                             PTR=J
                                             FIRST=J+1
                                             DO K=FIRST,LAST
                                                  IF (WCUT(K).LT.WCUT(PTR))
                                                  PTR=K
                                             END DO
                                             HOLD=WCUT(J)
                                             WCUT(J)=WCUT(PTR)
                                             WCUT(PTR)=HOLD
                                        END DO
C SORTED LIST


                                        IF (DUMPPRN.EQ.2) THEN
                                         WRITE(13,'(A30)'),'SORTED LIST:'
                                         WRITE(13,'(A30,D15.5)'),'WCUT(1)=',WCUT(1)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(2)=',WCUT(2)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(3)=',WCUT(3)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(4)=',WCUT(4)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(5)=',WCUT(5)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(6)=',WCUT(6)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(7)=',WCUT(7)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(8)=',WCUT(8)
                                         WRITE(13,'(A30,D15.5)'),'WCUT(9)=',WCUT(9)

WRITE(13,'(A30,D15.5)'),'WCUT(10)=',WCUT(10)
                                        END IF


C                                       CALCULATE THE SUB-VOLUMES.


                                        FVOLTEMP=0.D0


                                        DO IZC=2,10



                                             IF ( (WCUT(IZC).GT.WDIST).OR.
  *                                           (WCUT(IZC-1).LT.(WDIST-DW) ).OR.
  *                                           (WCUT(IZC).EQ.WCUT(IZC-1) ) ) THEN


C                                       NO OVERLAP: SUB-VOLUME EQUALS ZERO.

                                                  SUBDW=0.D0
                                                  VOL=0.D0

                                             ELSE
C                                                 SUBVOLUME IS A PISMATOID
```

```
                          SUBDW=WCUT(IZC)-WCUT(IZC-1)

C                         SUB-BEAMLET LIMITS (FRONT FACE)
                          UBMIN(1)=UVERTEX(IBU-1,0) +
     *                    WCUT(IZC-1)*TAN(UBEAMANGLE(IBU-1))
                          UBMAX(1)=UVERTEX(IBU,0) +
     *                    WCUT(IZC-1)*TAN(UBEAMANGLE(IBU))
                          VBMAX(1)=VVERTEX(IBV,0) +
     *                    WCUT(IZC-1)*TAN(VBEAMANGLE(IBV))
                          VBMIN(1)=VVERTEX(IBV-1,0) +
     *                    WCUT(IZC-1)*TAN(VBEAMANGLE(IBV-1))


C                         SUB-BEAMLET LIMITS (BACK FACE)

                          UBMIN(2)=UVERTEX(IBU-1,0) +
     *                    WCUT(IZC)*TAN(UBEAMANGLE(IBU-1))
                          UBMAX(2)=UVERTEX(IBU,0) +
     *                    WCUT(IZC)*TAN(UBEAMANGLE(IBU))
                          VBMAX(2)=VVERTEX(IBV,0) +
     *                    WCUT(IZC)*TAN(VBEAMANGLE(IBV))
                          VBMIN(2)=VVERTEX(IBV-1,0) +
     *                    WCUT(IZC)*TAN(VBEAMANGLE(IBV-1))

                          IF (DUMPPRN.EQ.2) THEN
                                 WRITE(13,'(A30)'),'.........'
                                 WRITE(13,'(A30,I10)'),'IZC=',IZC
                                 WRITE(13,'(A30,D15.5)'),
     *                           'WCUT(IZC-1)=',WCUT(IZC-1)

                                        WRITE(13,'(A30,D15.5)'),
    *
                                        'WCUT(IZC)=',WCUT(IZC)

                                        WRITE(13,'(A30,D15.5)'),'SUBDW=',
     *                                        SUBDW


                                        WRITE(13,'(A30,D15.5)'),
     *                                  'UBMIN(1)=',UBMIN(1)

                                        WRITE(13,'(A30,D15.5)'),
     *                                  'UBMAX(1)=',UBMAX(1)

                                        WRITE(13,'(A30,D15.5)'),
     *                                  'VBMIN(1)=',VBMIN(1)

                                        WRITE(13,'(A30,D15.5)'),
     *                                  'VBMAX(1)=',VBMAX(1)



                                        WRITE(13,'(A30,D15.5)'),
     *                                  'UBMIN(2)=',UBMIN(2)
```

```
                                          WRITE(13,'(A30,D15.5)'),
     *                                    'UBMAX(2)=',UBMAX(2)

                                          WRITE(13,'(A30,D15.5)'),
   *                                      'VBMIN(2)=',VBMIN(2)

                                          WRITE(13,'(A30,D15.5)'),
   *                                      'VBMAX(2)=',VBMAX(2)


                                                  END IF


C                         SKIP VOLUME CALCULATION IF THERE IS
C                         NO OVERLAP:


                              IF ( ( (UBMAX(1).LE.ULO).AND.
   *                            (UBMAX(2).LE.ULO) ) .OR.
   *                                    ( (UBMIN(1).GE.UHI).AND.
   *                                      (UBMIN(2).GE.UHI) ) .OR.
   *                                     ( (VBMAX(1).LE.VLO).AND.
   *                                      (VBMAX(2).LE.VLO) ) .OR.
   *                                    ( (VBMIN(1).GE.VHI).AND.
   *                                      (VBMIN(2).GE.VHI) ) ) THEN

                                        VOL=0.D0

C                     CALCVOL

                              ELSE


                                        VOL=CALCVOL(SUBDW,UHI,ULO,
   *                                            VHI,VLO,UBMAX,UBMIN,
   *                                            VBMAX,VBMIN,DUMPPRN)

                              IF (DUMPPRN.EQ.2) THEN
                                        WRITE(13,'(A30,D15.5)'),
   *                                    'VOL=',VOL
                                        WRITE(13,'(A30,D15.5)'),
   *                                    'FVOLTEMP=',FVOLTEMP
                              END IF

                          END IF


                      END IF



                  FVOLTEMP=FVOLTEMP+VOL

                  END DO
```

```
                                   END IF


C                                  REMEMBER TO CONVERT VOLUMES TO VOLUME
C                                  OVERLAP FRACTIONS.


                                   FVOLUVW(IBU,IBV,IUG,IVG,IWG)=
   *                               FVOLTEMP/BEAMLETVOL(IBU,IBV,IWG)

                                   IF (DUMPPRN.EQ.2) THEN

                                        WRITE(13,'(A20,D15.5)'),'FVOLTEMP=',
   *                                    FVOLTEMP
                                        WRITE(13,'(A20,D15.5)'),'FVOLUVW()=',
   *                                    FVOLUVW(IBU,IBV,IUG,IVG,IWG)
                                        WRITE(13,'(A20)'),''

                                   END IF


                                   END DO
                              END DO

2000                          CONTINUE


                         END DO
                    END DO

               END DO


C COPY DATA TO COMMON ARRAYS ------------------------------------------
C MIGHT AS WELL DO THIS FOR ALL OUTPUT DATA


C IBU1,IBU2

          DO IUG=1,NUG
               DO IWG=1,NWG

                    IF (PASSTYPE.EQ.'SEED') THEN
                         IBX1S(PASSNUMBER,IUG,IWG)=IBU1(IUG,IWG)
                         IBX2S(PASSNUMBER,IUG,IWG)=IBU2(IUG,IWG)
                    ELSE IF (PASSTYPE.EQ.'PMPY') THEN
                         JBZ1PY(PASSNUMBER,IUG,IWG)=IBU1(IUG,IWG)
                         JBZ2PY(PASSNUMBER,IUG,IWG)=IBU2(IUG,IWG)
                    ELSE
                         IBX1PZ(PASSNUMBER,IUG,IWG)=IBU1(IUG,IWG)
                         IBX2PZ(PASSNUMBER,IUG,IWG)=IBU2(IUG,IWG)
                    END IF
```

```
                END DO
        END DO


C IBV1,IBV2

        DO IVG=1,NVG
                DO IWG=1,NWG

                        IF (PASSTYPE.EQ.'SEED') THEN
                                IBY1S(PASSNUMBER,IVG,IWG)=IBV1(IVG,IWG)
                                IBY2S(PASSNUMBER,IVG,IWG)=IBV2(IVG,IWG)
                        ELSE IF (PASSTYPE.EQ.'PMPY') THEN
                                JBX1PY(PASSNUMBER,IVG,IWG)=IBV1(IVG,IWG)
                                JBX2PY(PASSNUMBER,IVG,IWG)=IBV2(IVG,IWG)
                        ELSE
                                IBY1PZ(PASSNUMBER,IVG,IWG)=IBV1(IVG,IWG)
                                IBY2PZ(PASSNUMBER,IVG,IWG)=IBV2(IVG,IWG)
                        END IF

                END DO
        END DO

C BEAMLETVOL,FVOLUVW


        DO IBU=1,NBU
                DO IBV=1,NBV
                        DO IUG=1,NUG
                                DO IVG=1,NVG
                                        DO IWG=1,NWG

                                                IF (PASSTYPE.EQ.'SEED') THEN

                                FVOLZ(PASSNUMBER,IBU,IBV,IUG,IVG,IWG)=
     *                          FVOLUVW(IBU,IBV,IUG,IVG,IWG)
                                                ELSE IF (PASSTYPE.EQ.'PMPY') THEN

                                FVOLPY(PASSNUMBER,IBV,IBU,IVG,IWG,IUG)=
     *                          FVOLUVW(IBU,IBV,IUG,IVG,IWG)
                                                ELSE

                                FVOLPZ(PASSNUMBER,IBU,IBV,IUG,IVG,IWG)=
     *                          FVOLUVW(IBU,IBV,IUG,IVG,IWG)
                                                END IF

                                        END DO
                                END DO
                        END DO
                END DO
        END DO


        IF (PASSTYPE.EQ.'SEED') THEN
                DO IBU=1,NBU
                        DO IBV=1,NBV
```

```
                          DO IWG=1,NWG
                                SBEAMVOL(PASSNUMBER,IBU,IBV,IWG)=
     *                          BEAMLETVOL(IBU,IBV,IWG)
                          END DO
                    END DO
              END DO
        END IF


C UVERTEX,VVERTEX -



        DO IBU=0,NBU
              DO IW=0,NWTOTAL

                    IF (PASSTYPE.EQ.'SEED') THEN
                          XVS(PASSNUMBER,IBU,IW)=UVERTEX(IBU,IW)
                    ELSE IF (PASSTYPE.EQ.'PMPY') THEN
                          ZVPY(PASSNUMBER,IBU,IW)=UVERTEX(IBU,IW)
                    ELSE
                          XVPZ(PASSNUMBER,IBU,IW)=UVERTEX(IBU,IW)
                    END IF

              END DO
        END DO

        DO IBV=0,NBV
              DO IW=0,NWTOTAL

                    IF (PASSTYPE.EQ.'SEED') THEN
                          YVS(PASSNUMBER,IBV,IW)=VVERTEX(IBV,IW)
                    ELSE IF (PASSTYPE.EQ.'PMPY') THEN
                          XVPY(PASSNUMBER,IBV,IW)=VVERTEX(IBV,IW)
                    ELSE
                          YVPZ(PASSNUMBER,IBV,IW)=VVERTEX(IBV,IW)
                    END IF

              END DO
        END DO


        RETURN
        END
```

……

```
    SUBROUTINE INICRT01

         INCLUDE 'Lascrt01.inc'

C LOCAL DECLARATIONS ---------------------------------------------

         CHARACTER*8 CHRDATE
         CHARACTER*10 CHRTIME

C SET UP THE NAMELISTS FOR THE INPUT PARAMETERS --------------------

         NAMELIST /XTALPAR/ COMMENT1,COMMENT2,COMMENT3,
    *    DT,NSTEPS,DUMPOU,MIRREFL,IPSOURCE,
    *    N1X,N2X,N3X,N1Y,N2Y,N3Y,N1Z,N2Z,N3Z,DENSNUM,SIGG,SIGABS,
    *    TGLIFE,ALPHAG,SYSTEMLEVELS,SLABOUTSW,
    *    POPXYSW,POPYZSW,POPZXSW,LASSPATSW,
    *    PUMPCALCSW,SEEDCALCSW,GEOMCALCSW,JT1,JT2,JT3,ESATFUDGE
    *   ,REFRINDEX

         NAMELIST /SEEDPAR/ THETAPROPX,THETAPROPY,THETADIVGX,THETADIVGY,
    * XBEAMCENTRE,XBEAMWIDTH,YBEAMCENTRE,YBEAMWIDTH,RSEEDSW,TSEEDSW,
    * SEEDENERGY,SEEDDURN,TSEED,SEEDRADE,LAMBDASEED,SEEDPER,
    * NBX,NBY,NPASS,INJPT,FILE50,FILE51,NPT50,NPT51,DT50


         NAMELIST /PUMPPAR/
RPUMPSW,TPUMPSW,PUMPENERGY,PUMPRADE,LAMBDAPUMP,
    * PUMPPER,PUMPDURN,TPUMP,
    * YTHPRPGPX,ZTHPRPGPX,YTHDIVGPX,ZTHDIVGPX,
    * XTHPRPGPY,ZTHPRPGPY,XTHDIVGPY,ZTHDIVGPY,
    * YTHPRPGPZ,XTHPRPGPZ,YTHDIVGPZ,XTHDIVGPZ,
    * XBEAMWIDTHPZ,YBEAMWIDTHPZ,XBEAMWIDTHPY,ZBEAMWIDTHPY,
    * YBEAMWIDTHPX, ZBEAMWDITHPX,
    * XBEAMCENTREPZ,YBEAMCENTREPZ,XBEAMCENTREPY,ZBEAMCENTREPY,
    * YBEAMCENTREPX,ZBEAMCENTREPX,
    * RP12,RP34,RP56,
    * NPX,NPY,MPX,MPZ,LPY,LPZ,FILE54,FILE55,NPT54,NPT55,DT54


         NAMELIST /ASEPAR/ ASEENERGY,ASEDURN,TASE,ASERADE,ASESR,ASEREFL,
    * TASESW,RASESW,FILE52,FILE53,NPT52,NPT53,DT52

C DEFAULT FILENAMES FOR THE DATA FILES. ----------------------------
C THIS IS TO AVOID OPENING AN UNNAMED FILE IF IT IS NOT SPECIFIED

         FILE50='file50.inp'
         FILE51='file51.inp'
         FILE52='file52.inp'
         FILE53='file53.inp'
         FILE54='file54.inp'
         FILE55='file55.inp'

C OPEN FILES. ------------------------------------------------

     OPEN(11,FILE='inp.par')
         OPEN(12,FILE='summary.out')
```

```
      OPEN(13,FILE='dump.out')

      OPEN(20,FILE='slabout.csv')
      OPEN(21,FILE='popxy.csv')
      OPEN(22,FILE='popyz.csv')
      OPEN(23,FILE='popzx.csv')
      OPEN(24,FILE='lasspat.csv')
      OPEN(25,FILE='pumpcalc.csv')
      OPEN(26,FILE='seedcalc.csv')
      OPEN(27,FILE='geomcalc.csv')

C WRITE THE DATE AND TIME TO SUMMARY.OUT AND DUMP.OUT---------------

      CALL DATE_AND_TIME (CHRDATE,CHRTIME)
      WRITE(12,'(A16,A8)'),'DATE(YYYYMMDD)=',CHRDATE
      WRITE(13,'(A16,A8)'),'DATE(YYYYMMDD)=',CHRDATE
      PRINT '(A16,A8)','DATE(YYYYMMDD)=',CHRDATE

      WRITE(12,'(A23,A10)'),'START TIME(hhmmss.sss)=',CHRTIME
      WRITE(13,'(A23,A10)'),'START TIME(hhmmss.sss)=',CHRTIME
   PRINT '(A23,A10)','START TIME(hhmmss.sss)=',CHRTIME

C DEFAULT INITIALIZATIONS ----------------------------------------

      SEEDENERGY=0.D0
      PUMPENERGY(1)=0.D0
      PUMPENERGY(2)=0.D0
      PUMPENERGY(3)=0.D0
      PUMPENERGY(4)=0.D0
      PUMPENERGY(5)=0.D0
      PUMPENERGY(6)=0.D0

C READ INPUT PARAMETERS FROM INP.PAR ------------------------------

 100  READ(11,XTALPAR,ERR=123)
 200  READ(11,SEEDPAR,ERR=234)
 300  READ(11,PUMPPAR,ERR=345)
 400  READ(11,ASEPAR,ERR=456)

      GOTO 500

 123 WRITE(12,'(A40)'),'ERROR READING XTALPAR FROM INP.PAR'
      PRINT '(A40)','ERROR READING XTALPAR FROM INP.PAR'

      IQUIT=1
      GOTO 200
 234 WRITE(12,'(A40)'),'ERROR READING SEEDPAR FROM INP.PAR'
      PRINT '(A40)','ERROR READING SEEDPAR FROM INP.PAR'
      IQUIT=1
      GOTO 300
 345 WRITE(12,'(A40)'),'ERROR READING PUMPPAR FROM INP.PAR'
      PRINT '(A40)','ERROR READING PUMPPAR FROM INP.PAR'
      IQUIT=1
      GOTO 400
 456 WRITE(12,'(A40)'),'ERROR READING ASEPAR FROM INP.PAR'
      PRINT '(A40)','ERROR READING ASEPAR FROM INP.PAR'
```

```
          IQUIT=1
          GOTO 500

 500 CONTINUE

C CHECK IF INJPT EXCEEDS NPASS
          IF (INJPT.GT.NPASS) THEN
                    PRINT '(A40)','INJPT EXCEEDS NPASS'
                    WRITE(13,'(A40)'),'INJPT EXCEEDS NPASS'
                    IQUIT=1
                    RETURN
          END IF


C OPEN DATA FILES ---------------------------------------------------

          OPEN(50,FILE=FILE50)
          OPEN(51,FILE=FILE51)
          OPEN(52,FILE=FILE52)
          OPEN(53,FILE=FILE53)
          OPEN(54,FILE=FILE54)
          OPEN(55,FILE=FILE55)

C WRITE PARAMETERS TO SUMMARY.OUT AND DUMP.OUT -----------------------

          WRITE(12,XTALPAR)
          WRITE(12,SEEDPAR)
          WRITE(12,PUMPPAR)
          WRITE(12,ASEPAR)

          WRITE(13,XTALPAR)
          WRITE(13,SEEDPAR)
          WRITE(13,PUMPPAR)
          WRITE(13,ASEPAR)

C WRITE COMMENTS TO THE .CSV FILES ---------------------------------

          DO IFNUM=20,27
                    WRITE(IFNUM,'(A75)'),COMMENT1
                    WRITE(IFNUM,'(A75)'),COMMENT2
                    WRITE(IFNUM,'(A75)'),COMMENT3
          END DO

C CHECK FOR INVALID REFRACTIVE INDEX

          IQUIT=0
          IF ( REFRINDEX.LE.(0.D0) )  THEN
                    PRINT '(A50)',
     *              'REFRACTIVE INDEX IS LESS THAN OR EQUAL TO ZERO'
                    WRITE(12,'(A50)'),
     *              'REFRACTIVE INDEX IS LESS THAN OR EQUAL TO ZERO'
                    IQUIT=1
                    RETURN
          END IF
```

```
C CALCULATE DERIVED QUANTITIES. -----------------------------------

C REVISED DEFINITIONS OF DX,DY,DZ,DWAIR - ACCOUNT FOR INDEX EFFECTS
        DX=DT*0.03D0/REFRINDEX
        DY=DT*0.03D0/REFRINDEX
        DZ=DT*0.03D0/REFRINDEX
        DWAIR=DT*0.03D0

        DA=DX*DY
        DV=DX*DY*DZ

        NX=N1X+N2X+N3X
        NY=N1Y+N2Y+N3Y
        NZ=N1Z+N2Z+N3Z


        IF (MOD(N2X,2).EQ.0) THEN
                IXGMID=N2X/2
        ELSE
                IXGMID=(N2X+1)/2
        END IF


        IF (MOD(N2Y,2).EQ.0) THEN
                IYGMID=N2Y/2
        ELSE
                IYGMID=(N2Y+1)/2
        END IF


        IF (MOD(N2Z,2).EQ.0) THEN
                IZGMID=N2Z/2
        ELSE
                IZGMID=(N2Z+1)/2
        END IF


        EPHOTONPUMP=(1.D6*H)*VLIGHT/LAMBDAPUMP
        EPHOTONSEED=(1.D6*H)*VLIGHT/LAMBDASEED

C CALCULATE SAT. FLUXES ACCORDING TO SYSTEMLEVELS -------------------

        IF (SYSTEMLEVELS.EQ.-3) THEN
                SATFLXPUMP=EPHOTONPUMP/SIGABS
                SATFLXSEED=EPHOTONSEED/(2.D0*SIGG)
        ELSE IF (SYSTEMLEVELS.EQ.3) THEN
                SATFLXPUMP=EPHOTONPUMP/(2.D0*SIGABS)
                SATFLXSEED=EPHOTONSEED/SIGG
        ELSE IF (SYSTEMLEVELS.EQ.4) THEN
                SATFLXPUMP=EPHOTONPUMP/SIGABS
                SATFLXSEED=EPHOTONSEED/SIGG
        ELSE
                PRINT '(A33)','IMPROPER VALUE FOR SYSTEMLEVELS'
                WRITE(12,'(A33)'),'IMPROPER VALUE FOR SYSTEMLEVELS'
                IQUIT=1
                RETURN
```

```
      END IF


      IF ( (ESATFUDGE.EQ.1).AND.(NBX.EQ.1).AND.(NBY.EQ.1) ) THEN
              ESATSEED=SATFLXSEED*(XBEAMWIDTH(1)*YBEAMWIDTH(1))
              WRITE(12,'(A51)'),
     *        'ESATSEED ADJUSTED TO ACCOUNT FOR SMALL BEAMLETS'
      ELSE
              ESATSEED=SATFLXSEED*DA
      END IF

      ESATPUMP=SATFLXPUMP*DA


      KSEED=(1.D0)/(EPHOTONSEED*DZ)
      KPUMP=(1.D0)/(EPHOTONPUMP*DY)

      IF ( MOD(NPASS,2) .EQ.0) THEN
              NPPARITY=0
              NPFINALEVEN=NPASS
              NPFINALODD=NPASS-1
      ELSE
              NPPARITY=1
              NPFINALEVEN=NPASS-1
              NPFINALODD=NPASS
      END IF


      IF ( MOD(INJPT,2) .EQ.0) THEN
              IZSTARTINJPT=NZ
      ELSE
              IZSTARTINJPT=1
      END IF

C WRITE DERIVED QUANTITIES TO SUMMARY.OUT ----------------------------
C USE THE EN23.10 FORMAT

      WRITE(12,'(A5)'),' '

      WRITE(12,'(A7,EN23.10,A7)'),'DWAIR=',DWAIR,' cm'
      WRITE(12,'(A5,EN23.10,A7)'),'DX=',DX,' cm'
      WRITE(12,'(A5,EN23.10,A7)'),'DA=',DA,' cm^2'
      WRITE(12,'(A5,EN23.10,A7)'),'DV=',DV,' cm^3'

      WRITE(12,'(A5,I5)'),'NX=',NX
      WRITE(12,'(A5,I5)'),'NY=',NY
      WRITE(12,'(A5,I5)'),'NZ=',NZ

      WRITE(12,'(A13,EN28.15,A4)'),'EPHOTONPUMP=',EPHOTONPUMP,' uJ'
      WRITE(12,'(A13,EN28.15,A10)'),'SATFLXPUMP=',SATFLXPUMP,' uJ/cm^2'
      WRITE(12,'(A13,EN28.15,A4)'),'ESATPUMP=',ESATPUMP,' uJ'
      WRITE(12,'(A13,EN28.15,A13)'),'KPUMP=',KPUMP,' (uJ*cm)^-1'


      WRITE(12,'(A13,EN28.15,A4)'),'EPHOTONSEED=',EPHOTONSEED,' uJ'
      WRITE(12,'(A13,EN28.15,A10)'),'SATFLXSEED=',SATFLXSEED,' uJ/cm^2'
```

```
        WRITE(12,'(A13,EN28.15,A4)'),'ESATSEED=',ESATSEED,' uJ'
        WRITE(12,'(A13,EN28.15,A13)'),'KSEED=',KSEED,' (uJ*cm)^-1'

        WRITE(12,'(A13,I5)'),'NPPARITY=',NPPARITY
        WRITE(12,'(A13,I5)'),'NPFINALEVEN=',NPFINALEVEN
        WRITE(12,'(A13,I5)'),'NPFINALODD=',NPFINALODD
        WRITE(12,'(A13,I5)'),'IZSTARTINJPT=',IZSTARTINJPT

        WRITE(12,'(A5)'),' '

C WRITE HEADER LINES TO THE .CSV FILES ------------------------------

        WRITE(20,4590), ',TIME [ps], GPOP(IXGMID;IYGMID;IZGMID) [cm^-3] ,
   *   EPUMPSLAB(1) [uJ],EPUMPSLAB(2) [uJ],
   *    EPUMPSLAB(3) [uJ],EPUMPSLAB(4) [uJ],'
   *            ,('ESEEDSLAB(' ,IP, ') [uJ],EASESLAB(' ,IP, ') [uJ] ,',
   *            IP=1,NPASS)

 4590 FORMAT (A135,<NPASS>(A16,I3,A18,I3,A10) )

        WRITE(25,'(A22,I3,A6,I3,A6,I3)'),'ALL DATA IS FOR JXG=',
   *    IXGMID,' JYG=',IYGMID,' JZG=',IZGMID
        WRITE(25,'(A130)'),',TIME [ps],APOPINV [cm^-3],
   *    EPUMPTOTIN [uJ],CPMP [DQ],SATFRC [DQ],PARENTH [DQ],
   *    LNARG [DQ],EABS [uJ],EPUMPTOTOUT [uJ]'

        WRITE(26,'(A22,I3,A6,I3,A6,I3)'),'ALL DATA IS FOR IXG=',
   *    IXGMID,' IYG=',IYGMID,' IZG=',IZGMID
        WRITE(26,'(A130)'),',TIME [ps],GPOPINV [cm^-3],
   *    ESEEDTOTIN [uJ],CSEED [DQ],SATFRC [DQ],PARENTH [DQ],
   *    LNARG [DQ],EGAIN [uJ],ESEEDTOTOUT [uJ]'


C CHECK IF SPECIFIED ARRAY SIZES EXCEED THE MAXIMUM VALUES -----------
C
C (NBX,NBY,NPX,NPY,MPX,MPZ,LPY,LPZ) > NBMAX ?
C NX > NXMAX ?
C NY > NYMAX ?
C NZ > NZMAX ?
C (N2X,N2Y,N2Z) > (N2XMAX,N2YMAX,N2ZMAX) ?
C NPASS > NPMAX ?

        IQUIT=0

        IF ( (NBX.GT.NBMAX).OR.(NBY.GT.NBMAX).OR.(NPX.GT.NBMAX).OR.
   *    (NPY.GT.NBMAX).OR.(MPX.GT.NBMAX).OR.(MPZ.GT.NBMAX).OR.
   *    (LPY.GT.NBMAX).OR.(LPZ.GT.NBMAX) ) THEN

            PRINT '(A33)','ARRAY DIMENSION EXCEEDS NBMAX'
            WRITE(12,'(A33)'),'ARRAY DIMENSION EXCEEDS NBMAX'
            IQUIT=1
            RETURN
        END IF

        IF ( (NX.GT.NXMAX).OR.(NY.GT.NYMAX).OR.(NZ.GT.NZMAX).OR.
```

```
*      (NPASS.GT.NPMAX) ) THEN
              PRINT '(A33)',
*             'ARRAY DIMENSION EXCEEDS NXMAX,NYMAX,NZMAX OR NPMAX'
              WRITE(12,'(A33)'),
*             'ARRAY DIMENSION EXCEEDS NXMAX,NYMAX,NZMAX OR NPMAX'
              IQUIT=1
              RETURN
       END IF

       IF ( (N2X.GT.N2XMAX).OR.(N2Y.GT.N2YMAX).OR.(N2Z.GT.N2ZMAX) ) THEN
              PRINT '(A33)',
*             'ARRAY DIMENSION EXCEEDS N2XMAX,N2YMAX OR N2ZMAX'
              WRITE(12,'(A33)'),
*             'ARRAY DIMENSION EXCEEDS N2XMAX,N2YMAX OR N2ZMAX'
              IQUIT=1
              RETURN
       END IF


C CALL SUBROUTINES FOR SEED AND PUMP BEAM INITIALIZATION -------------
       CALL PMICRT02

       CLOSE(UNIT=54, STATUS='KEEP')
       OPEN(54,FILE=FILE54)

   CALL PMZCRT01
       CALL SDICRT02

C CHECK FOR NON-OVERLAPPING BEAMS.

       IF (MAXVAL(FVOLZ).LT.(1.D-50)) THEN
              PRINT '(A53)',
*             'WARNING:  SEED BEAM DOES NOT OVERLAP GAIN REGION'
              WRITE(12,'(A53)'),
*             'WARNING: SEED BEAM DOES NOT OVERLAP GAIN REGION'
              WRITE(12,'(A53)'),
*             'BEAM CENTRES ARE MEASURED FROM CAVITY EDGE'


       END IF

       IF (MAXVAL(FVOLPY).LT.(1.D-50)) THEN
              PRINT '(A53)',
*             'WARNING:  PUMP BEAM DOES NOT OVERLAP GAIN REGION'
              WRITE(12,'(A53)'),
*             'WARNING: PUMP BEAM DOES NOT OVERLAP GAIN REGION'
              WRITE(12,'(A53)'),
*             'BEAM CENTRES ARE MEASURED FROM CAVITY EDGE'

       END IF


C                 ---------------------------------------------
C WRITE DATA TO GEOMCALC.CSV [UNIT 27]
```

```
      IF (GEOMCALCSW.EQ.1) THEN

             WRITE(27,'(A35)'),'GEOMETRICAL DATA FOR PUMP Y BEAMS'
             WRITE(27,'(A60)'),
   *   ',JP,JBX,JBZ,JXG,JYG,JZG,JBX1PY,JBX2PY,JBZ1PY,JBZ2PY,FVOLPY'
             DO JP=3,4
              DO JBX=1,MPX
               DO JBZ=1,MPZ
                DO JXG=1,N2X
                      DO JYG=1,1
                            DO JZG=1,N2Z

                             WRITE(27,'(10(A2,I4),A2,E15.6)')
   *                        ,',',JP,',',JBX,',',JBZ,',',JXG,',',JYG,',',JZG
   *                        ,',',JBX1PY(JP,JXG,JYG),',',JBX2PY(JP,JXG,JYG)
   *                        ,',',JBZ1PY(JP,JZG,JYG),',',JBZ2PY(JP,JZG,JYG)
   *                        ,',',FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG)

                            END DO
                           END DO
                END DO
                    END DO
              END DO
             END DO
C            ------------------------------------------------

             WRITE(27,'(A35)'),'GEOMETRICAL DATA FOR PUMP Z BEAMS'
             WRITE(27,'(A60)'),
   *   ',IP,IBX,IBY,IXG,IYG,IZG,IBX1PZ,IBX2PZ,IBY1PY,IBY2PZ,FVOLPZ'
             DO IP=1,2
              DO IBX=1,NPX
               DO IBY=1,NPY
                DO IXG=1,N2X
                      DO IYG=1,N2Y
                            DO IZG=1,1

                             WRITE(27,'(10(A2,I4),A2,E15.6)')
   *                        ,',',IP,',',IBX,',',IBY,',',IXG,',',IYG,',',IZG
   *                        ,',',IBX1PZ(IP,IXG,IZG),',',IBX2PZ(IP,IXG,IZG)
   *                        ,',',IBY1PZ(IP,IYG,IZG),',',IBY2PZ(IP,IYG,IZG)
   *                        ,',',FVOLPZ(IP,IBX,IBY,IXG,IYG,IZG)

                            END DO
                           END DO
                END DO
                    END DO
              END DO
             END DO

C            ------------------------------------------------
             WRITE(27,'(A35)'),'GEOMETRICAL DATA FOR SEED BEAMS'
             WRITE(27,'(A60)'),
   *   ',IP,IBX,IBY,IXG,IYG,IZG,IBX1S,IBX2S,IBY1S,IBY2S,FVOLZ'
             DO IP=1,NPASS
              DO IBX=1,NBX
```

```
               DO IBY=1,NBY
                DO IXG=1,N2X
                      DO IYG=1,N2Y
                            DO IZG=1,1

                             WRITE(27,'(10(A2,I4),A2,E15.6)')
  *                         ,',',IP,',',IBX,',',IBY,',',IXG,',',IYG,',',IZG
  *                         ,',',IBX1S(IP,IXG,IZG),',',IBX2S(IP,IXG,IZG)
  *                         ,',',IBY1S(IP,IYG,IZG),',',IBY2S(IP,IYG,IZG)
  *                         ,',',FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)

                            END DO
                           END DO
                END DO
                    END DO
              END DO
             END DO
C            ----------------------------------------------
       ELSE IF (GEOMCALCSW.EQ.2) THEN

             WRITE(27,'(A35)'),'GEOMETRICAL DATA FOR PUMP Y BEAMS'
             WRITE(27,'(A60)'),
  *     ',JP,JBX,JBZ,JXG,JYG,JZG,JBX1PY,JBX2PY,JBZ1PY,JBZ2PY,FVOLPY'
             DO JP=3,4
              DO JBX=1,MPX
               DO JBZ=1,MPZ
                DO JXG=1,N2X
                      DO JYG=1,N2Y
                            DO JZG=1,N2Z

                             WRITE(27,'(10(A2,I4),A2,E15.6)')
  *                         ,',',JP,',',JBX,',',JBZ,',',JXG,',',JYG,',',JZG
  *                         ,',',JBX1PY(JP,JXG,JYG),',',JBX2PY(JP,JXG,JYG)
  *                         ,',',JBZ1PY(JP,JZG,JYG),',',JBZ2PY(JP,JZG,JYG)
  *                         ,',',FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG)

                            END DO
                           END DO
                END DO
                    END DO
              END DO
             END DO
C            ----------------------------------------------

             WRITE(27,'(A35)'),'GEOMETRICAL DATA FOR PUMP Z BEAMS'
             WRITE(27,'(A60)'),
  *     ',IP,IBX,IBY,IXG,IYG,IZG,IBX1PZ,IBX2PZ,IBY1PY,IBY2PZ,FVOLPZ'
             DO IP=1,2
              DO IBX=1,NPX
               DO IBY=1,NPY
                DO IXG=1,N2X
                      DO IYG=1,N2Y
                            DO IZG=1,N2Z

                             WRITE(27,'(10(A2,I4),A2,E15.6)')
  *                         ,',',IP,',',IBX,',',IBY,',',IXG,',',IYG,',',IZG
```

```
     *                     ,',',IBX1PZ(IP,IXG,IZG),',',IBX2PZ(IP,IXG,IZG)
     *                     ,',',IBY1PZ(IP,IYG,IZG),',',IBY2PZ(IP,IYG,IZG)
     *                     ,',',FVOLPZ(IP,IBX,IBY,IXG,IYG,IZG)

                            END DO
                           END DO
                    END DO
                        END DO
                   END DO
                  END DO


C                 ------------------------------------------
                  WRITE(27,'(A35)'),'GEOMETRICAL DATA FOR SEED BEAMS'
                  WRITE(27,'(A60)'),
     *      ',IP,IBX,IBY,IXG,IYG,IZG,IBX1S,IBX2S,IBY1S,IBY2S,FVOLZ'
                  DO IP=1,NPASS
                   DO IBX=1,NBX
                    DO IBY=1,NBY
                     DO IXG=1,N2X
                           DO IYG=1,N2Y
                                 DO IZG=1,N2Z

                                  WRITE(27,'(10(A2,I4),A2,E15.6)')
     *                            ,',',IP,',',IBX,',',IBY,',',IXG,',',IYG,',',IZG
     *                            ,',',IBX1S(IP,IXG,IZG),',',IBX2S(IP,IXG,IZG)
     *                            ,',',IBY1S(IP,IYG,IZG),',',IBY2S(IP,IYG,IZG)
     *                            ,',',FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)

                                 END DO
                                END DO
                     END DO
                         END DO
                    END DO
                   END DO


         END IF

C CLOSE AND RE-OPEN THE TEMPORAL INPUT DATA FILES SO THAT THEY -------
C CAN BE READ AGAIN FROM THE START

         CLOSE(UNIT=50, STATUS='KEEP')
         CLOSE(UNIT=52, STATUS='KEEP')
         CLOSE(UNIT=54, STATUS='KEEP')

         OPEN(50,FILE=FILE50)
         OPEN(52,FILE=FILE52)
         OPEN(54,FILE=FILE54)

      RETURN
         END
..............................................................................................
```

```
      SUBROUTINE LPICRT01


C THIS SUBROUTINE PERFORMS LOOP INITIALIZATION CALCULATIONS
C (INPUT ENERGY IN THE BEAMS FOR EACH TIME STEP).
C

C

      INCLUDE 'Lascrt01.inc'

C LOCAL DECLARATIONS ---------------------------------------------

      DOUBLE PRECISION CALCTIME,PUMPEXPT,PUMPEXPR,SEEDEXPT,SEEDEXPR
      DOUBLE PRECISION ASEEXPT,ASEEXPR
      DOUBLE PRECISION PUMPCLOCK,SEEDCLOCK,ASECLOCK


C INITIALIZATIONS

      IF (JT.EQ.1) THEN
            DATA50=0.D0
            DATA52=0.D0
            DATA54=0.D0
            PUMPCLOCK=0.D0
            SEEDCLOCK=0.D0
            ASECLOCK=0.D0
      END IF

      CALCTIME=TIME-(DT*0.5D0)

C PUMP PULSE: CALCULATE TIME MULTIPLIER ----------------------------

      IF (TPUMPSW.EQ.0) THEN
C           READ FROM DATA FILE

            IF ( (TIME.GT.TPUMP).AND.
     *      (TIME.LT.(TPUMP+DT54*(NPT54+1))) ) THEN

                  IF (DUMPOU.EQ.8) THEN
                        WRITE(13,'(A20,D13.5)'),'PUMPCLOCK= ',PUMPCLOCK

                  END IF

                  IF (PUMPCLOCK.GE.DT54) THEN
C                       READ NEW DATA POINT
                        PUMPCLOCK=PUMPCLOCK-DT54
                        READ(54,*),DATA54
                        PUMPEXPT=DATA54
                  ELSE
C                       OLD DATA POINT
                        PUMPEXPT=DATA54
                  END IF

                  PUMPCLOCK=PUMPCLOCK+DT
```

```
                ELSE

                        PUMPEXPT=0.D0

                END IF


        ELSE IF (TPUMPSW.EQ.1) THEN
C               TEMPORAL GAUSSIAN
                PUMPEXPT=EXP( -4.D0*DLOG(2.D0)*
   *            ((CALCTIME-TPUMP)/PUMPDURN)**2 )


        ELSE IF (TPUMPSW.EQ.2) THEN
C               TEMPORAL UNIFORM
                JPUMPON=INT( (TPUMP-(0.5D0*PUMPDURN))/DT )
                JPUMPOFF=INT( (TPUMP+(0.5D0*PUMPDURN))/DT )

                IF ( (JT.GE.JPUMPON).AND.(JT.LT.JPUMPOFF) ) THEN
                        PUMPEXPT=1.D0
                ELSE
                        PUMPEXPT=0.D0
                END IF

        END IF


C PUMP Y PULSE: CALCULATE SPATIAL MULTIPLIER ------------------------

        DO JBX=1,MPX
                DO JBZ=1,MPZ


                        IF (RPUMPSW.EQ.0) THEN
C                               READ FROM DATA FILE
                                PUMPEXPR=PUMPRDATA(JBX,JBZ)
                        ELSE IF (RPUMPSW.EQ.2) THEN
C                               SPATIAL UNIFORM
                                PUMPEXPR=1.D0
                        END IF

C                       CALCULATE PUMP INTENSITY AND ENERGY ----------------

                        PUMPINTENS3=PUMPPKINTENS3*PUMPEXPT*PUMPEXPR
                        PUMPINTENS4=PUMPPKINTENS4*PUMPEXPT*PUMPEXPR

                        EPUMPINPY(3,JBX,JBZ)=PUMPINTENS3*DT*PUMPDBA(3)
                        EPUMPINPY(4,JBX,JBZ)=PUMPINTENS4*DT*PUMPDBA(4)

                        EPUMPTOTALINPY(3)=EPUMPTOTALINPY(3)+EPUMPINPY(3,JBX,JBZ)
                        EPUMPTOTALINPY(4)=EPUMPTOTALINPY(4)+EPUMPINPY(4,JBX,JBZ)

                        IF (DUMPOU.EQ.8) THEN
                                WRITE(13,'(2(A6,I5))'),'JBX= ',JBX,' JBZ= ',JBZ
```

```
               WRITE(13,'(A20,D13.5)'),'CALCTIME= ',CALCTIME
               WRITE(13,'(A20,I10)'),'JPUMPON= ',JPUMPON
               WRITE(13,'(A20,I10)'),'JPUMPOFF= ',JPUMPOFF
               WRITE(13,'(A20,D13.5)'),'PUMPEXPT= ',PUMPEXPT
               WRITE(13,'(A20,D13.5)'),'PUMPEXPR= ',PUMPEXPR

               WRITE(13,'(A20,D13.5)'),'PUMPINTENS3= ',PUMPINTENS3
               WRITE(13,'(A20,D13.5)'),'PUMPINTENS4= ',PUMPINTENS4
               WRITE(13,'(A20,D13.5)'),'EPUMPINPY(3,,)= ',
     *         EPUMPINPY(3,JBX,JBZ)
               WRITE(13,'(A20,D13.5)'),'EPUMPINPY(4,,)= ',
     *         EPUMPINPY(4,JBX,JBZ)
               WRITE(13,'(A20,D13.5)'),'EPUMPTOTALINPY(3)= ',
     *         EPUMPTOTALINPY(3)
               WRITE(13,'(A20,D13.5)'),'EPUMPTOTALINPY(4)= ',
     *         EPUMPTOTALINPY(4)
            END IF

         END DO
      END DO

C PUMP Z PULSE: CALCULATE SPATIAL MULTIPLIER -------------------------

      DO IBX=1,NPX
         DO IBY=1,NPY

            IF (RPUMPSW.EQ.0) THEN
C              READ FROM DATA FILE
               PUMPEXPR=PUMPRDATA(IBX,IBY)
            ELSE IF (RPUMPSW.EQ.2) THEN
C              SPATIAL UNIFORM
               PUMPEXPR=1.D0
            END IF

C           CALCULATE PUMP INTENSITY AND ENERGY --------------------

            PUMPINTENS1=PUMPPKINTENS1*PUMPEXPT*PUMPEXPR
            PUMPINTENS2=PUMPPKINTENS2*PUMPEXPT*PUMPEXPR

            EPUMPINPZ(1,IBX,IBY)=PUMPINTENS1*DT*PUMPDBA(1)
            EPUMPINPZ(2,IBX,IBY)=PUMPINTENS2*DT*PUMPDBA(2)

            EPUMPTOTALINPZ(1)=EPUMPTOTALINPZ(1)+EPUMPINPZ(1,IBX,IBY)
            EPUMPTOTALINPZ(2)=EPUMPTOTALINPZ(2)+EPUMPINPZ(2,IBX,IBY)

            IF (DUMPOU.EQ.8) THEN
               WRITE(13,'(2(A6,I5))'),'IBX= ',IBX,' IBY= ',IBY

               WRITE(13,'(A20,D13.5)'),'CALCTIME= ',CALCTIME
               WRITE(13,'(A20,I10)'),'JPUMPON= ',JPUMPON
               WRITE(13,'(A20,I10)'),'JPUMPOFF= ',JPUMPOFF
               WRITE(13,'(A20,D13.5)'),'PUMPEXPT= ',PUMPEXPT
               WRITE(13,'(A20,D13.5)'),'PUMPEXPR= ',PUMPEXPR

               WRITE(13,'(A20,D13.5)'),'PUMPINTENS1= ',PUMPINTENS1
               WRITE(13,'(A20,D13.5)'),'PUMPINTENS2= ',PUMPINTENS2
```

```
                              WRITE(13,'(A20,D13.5)'),'EPUMPINPY(1,,)= ',
     *                        EPUMPINPZ(1,IBX,IBY)
                              WRITE(13,'(A20,D13.5)'),'EPUMPINPZ(2,,)= ',
     *                        EPUMPINPZ(2,IBX,IBY)
                              WRITE(13,'(A20,D13.5)'),'EPUMPTOTALINPZ(1)= ',
     *                        EPUMPTOTALINPZ(1)
                              WRITE(13,'(A20,D13.5)'),'EPUMPTOTALINPZ(2)= ',
     *                        EPUMPTOTALINPZ(2)
                         END IF

                    END DO
               END DO
C


C SEED PULSE: CALCULATE TIME MULTIPLIER ------------------------------

               IF (TSEEDSW.EQ.0) THEN
C                   READ FROM DATA FILE

                    IF ( (TIME.GT.TSEED).AND.
     *              (TIME.LT.(TSEED+DT50*(NPT50+1))) ) THEN

                         IF (SEEDCLOCK.GE.DT50) THEN
C                             READ NEW DATA POINT
                              SEEDCLOCK=SEEDCLOCK-DT50
                              READ(50,*),DATA50
                              SEEDEXPT=DATA50
                         ELSE
C                             OLD DATA POINT
                              SEEDEXPT=DATA50
                         END IF

                         SEEDCLOCK=SEEDCLOCK+DT

                    ELSE

                         SEEDEXPT=0.D0

                    END IF


               ELSE IF (TSEEDSW.EQ.1) THEN
C                   TEMPORAL GAUSSIAN
                    SEEDEXPT=EXP( -4.D0*DLOG(2.D0)*
     *              ((CALCTIME-TSEED)/SEEDDURN)**2 )


               ELSE IF (TSEEDSW.EQ.2) THEN
C                   TEMPORAL UNIFORM
                    JSEEDON=INT( (TSEED-(0.5D0*SEEDDURN))/DT )
                    JSEEDOFF=INT( (TSEED+(0.5D0*SEEDDURN))/DT )

                    IF ( (JT.GE.JSEEDON).AND.(JT.LT.JSEEDOFF) ) THEN
                         SEEDEXPT=1.D0
                    ELSE
```

```
              SEEDEXPT=0.D0
          END IF

      END IF


C SEED PULSE: CALCULATE SPATIAL MULTIPLIER --------------------------

      DO IBX=1,NBX
          DO IBY=1,NBY


              IF (RSEEDSW.EQ.0) THEN
C                 READ FROM DATA FILE
                  SEEDEXPR=SEEDRDATA(IBX,IBY)
              ELSE IF (RSEEDSW.EQ.2) THEN
C                 SPATIAL UNIFORM
                  SEEDEXPR=1.D0
              END IF

C             CALCULATE SEED INTENSITY AND ENERGY --------------------

              SEEDINTENS=SEEDPKINTENS*SEEDEXPT*SEEDEXPR

              ESEEDINP(INJPT,IBX,IBY)=SEEDINTENS*DT*SEEDDBA(INJPT)

              ESEEDTOTALIN(INJPT)=ESEEDTOTALIN(INJPT)
     *        +ESEEDINP(INJPT,IBX,IBY)


              IF (DUMPOU.EQ.7) THEN
                  WRITE(13,'(2(A6,I5))'),'IBX= ',IBX,' IBY= ',IBY

                  WRITE(13,'(A20,D13.5)'),'CALCTIME= ',CALCTIME
                  WRITE(13,'(A20,I10)'),'JSEEDON= ',JSEEDON
                  WRITE(13,'(A20,I10)'),'JSEEDOFF= ',JSEEDOFF
                  WRITE(13,'(A20,D13.5)'),'SEEDEXPT= ',SEEDEXPT
                  WRITE(13,'(A20,D13.5)'),'SEEDEXPR= ',SEEDEXPR

                  WRITE(13,'(A20,D13.5)'),'SEEDINTENS= ',SEEDINTENS
                  WRITE(13,'(A20,D13.5)'),'ESEEDINP(INJPT,,)= ',
     *            ESEEDINP(INJPT,IBX,IBY)
                  WRITE(13,'(A20,D13.5)'),'ESEEDTOTALIN(INJPT)= ',
     *            ESEEDTOTALIN(INJPT)
              END IF


          END DO
      END DO

C


C I/P ASE PULSE: CALCULATE TIME MULTIPLIER --------------------------

      IF (TASESW.EQ.0) THEN
```

```
C               READ FROM DATA FILE

                IF ( (TIME.GT.TASE).AND.
     *          (TIME.LT.(TASE+DT52*(NPT52+1))) ) THEN

                        IF (ASECLOCK.GE.DT52) THEN
C                               READ NEW DATA POINT
                                ASECLOCK=ASECLOCK-DT52
                                READ(52,*),DATA52
                                ASEEXPT=DATA52
                        ELSE
C                               OLD DATA POINT
                                ASEEXPT=DATA52
                        END IF

                        ASECLOCK=ASECLOCK+DT

                ELSE

                        ASEEXPT=0.D0

                END IF


        ELSE IF (TASESW.EQ.1) THEN
C               TEMPORAL GAUSSIAN
                ASEEXPT=EXP( -4.D0*DLOG(2.D0)*
     *          ((CALCTIME-TASE)/ASEDURN)**2  )


        ELSE IF (TASESW.EQ.2) THEN
C               TEMPORAL UNIFORM
                JASEON=INT( (TASE-(0.5D0*ASEDURN))/DT )
                JASEOFF=INT( (TASE+(0.5D0*ASEDURN))/DT )

                IF ( (JT.GE.JASEON).AND.(JT.LT.JASEOFF) ) THEN
                        ASEEXPT=1.D0
                ELSE
                        ASEEXPT=0.D0
                END IF

        END IF



C I/P ASE PULSE: CALCULATE SPATIAL MULTIPLIER ---------------------------

        DO IBX=1,NBX
                DO IBY=1,NBY


                        IF (RASESW.EQ.0) THEN
C                               READ FROM DATA FILE
                                ASEEXPR=ASERDATA(IBX,IBY)
                        ELSE IF (RASESW.EQ.2) THEN
C                               SPATIAL UNIFORM
```

```
               ASEEXPR=1.D0
            END IF

C           CALCULATE I/P ASE INTENSITY AND ENERGY -----------------

            ASEINTENS=ASEPKINTENS*ASEEXPT*ASEEXPR


            EASEINP(INJPT,IBX,IBY)=ASEINTENS*DT*SEEDDBA(INJPT)

            EASETOTALIN(INJPT)=EASETOTALIN(INJPT)+
     *      EASEINP(INJPT,IBX,IBY)

            IF (DUMPOU.EQ.7) THEN
               WRITE(13,'(2(A6,I5))'),'IBX= ',IBX,' IBY= ',IBY

               WRITE(13,'(A20,D13.5)'),'CALCTIME= ',CALCTIME
               WRITE(13,'(A20,I10)'),'JASEON= ',JASEON
               WRITE(13,'(A20,I10)'),'JASEOFF= ',JASEOFF
               WRITE(13,'(A20,D13.5)'),'ASEEXPT= ',ASEEXPT
               WRITE(13,'(A20,D13.5)'),'ASEEXPR= ',ASEEXPR

               WRITE(13,'(A20,D13.5)'),'ASEINTENS= ',ASEINTENS
               WRITE(13,'(A20,D13.5)'),'EASEINP(INJPT,,)= ',
     *         EASEINP(INJPT,IBX,IBY)
               WRITE(13,'(A20,D13.5)'),'EASETOTALIN(INJPT)= ',
     *         EASETOTALIN(INJPT)
            END IF




         END DO
      END DO




      RETURN
      END
```

……

```
SUBROUTINE OUTCRT01

        INCLUDE 'Lascrt01.inc'



C WRITE DATA TO SLABOUT.CSV ------------------------------------
C MAY 31/2000: ADD GPOP TO SLABOUT.CSV


        IF ( (SLABOUTSW.GT.0).AND.(MOD(JT,SLABOUTSW).EQ.0) ) THEN
                WRITE(20,4580),
   *    ',' ,TIME,
   *    ',' ,GPOP(IXGMID,IYGMID,IZGMID),
   *    ',' ,       EPUMPSLAB(1), ',' ,EPUMPSLAB(2),
   *    ',' ,       EPUMPSLAB(3), ',' ,EPUMPSLAB(4), ',' ,
   *    (ESEEDSLAB(IP),',',EASESLAB(IP),',',IP=1,NPASS)
        END IF



 4580 FORMAT (A2,E15.5E5,5(A2,E15.5E5),A2,
   *    <NPASS>(E15.5E5,A2,E15.5E5,A2))




        IF (  ((JT.EQ.JT1).OR.(JT.EQ.JT2)).OR.(JT.EQ.JT3)  ) THEN

C WRITE DATA TO POPXY.CSV ----------------------------------------

                IF (POPXYSW.GT.0) THEN

                        WRITE(21,'(A35,I8,A7,EN15.7,A5)'),
   *                    'GPOP(IXG;IYG;IZGMID) FOR JT=',JT,'TIME=',TIME,' ps'
                        WRITE(21,'(A2,<N2X>(A4,I3,A2))'),',',
   *                    ('IXG=',IXG,',',IXG=1,N2X)
                        DO IY=N1Y+1,N1Y+N2Y
                                IYG=IY-N1Y
                                WRITE(21,'(A4,I3,A2,<N2X>(E15.7,A2))'),
   *                            'IYG=',IYG,',',
   *                            ( GPOP(IXG,IYG,IZGMID),',',IXG=1,N2X )
                        END DO

                END IF

C WRITE DATA TO POPYZ.CSV ----------------------------------------

                IF (POPYZSW.GT.0) THEN

                        WRITE(22,'(A35,I8,A7,EN15.7,A5)'),
   *                    'GPOP(IXGMID;IYG;IZG) FOR JT=',JT,'TIME=',TIME,' ps'
                        WRITE(22,'(A2,<N2Y>(A4,I3,A2))'),',',
   *                    ('IYG=',IYG,',',IYG=1,N2Y)
                        DO IZ=N1Z+1,N1Z+N2Z
```

```
                        IZG=IZ-N1Z
                        WRITE(22,'(A4,I3,A2,<N2Y>(E15.7,A2))'),
     *                  'IZG=',IZG,',',
     *                  ( GPOP(IXGMID,IYG,IZG),',',IYG=1,N2Y )
                  END DO

            END IF

C WRITE DATA TO POPZX.CSV ----------------------------------------

            IF (POPZXSW.GT.0) THEN

                  WRITE(23,'(A35,I8,A7,EN15.7,A5)'),
     *            'GPOP(IXG;IYGMID;IZG) FOR JT=',JT,'TIME=',TIME,' ps'
                  WRITE(23,'(A2,<N2Z>(A4,I3,A2))'),',',
     *            ('IZG=',IZG,',',IZG=1,N2Z)
                  DO IX=N1X+1,N1X+N2X
                        IXG=IX-N1X
                        WRITE(23,'(A4,I3,A2,<N2Z>(E15.7,A2))'),
     *                  'IXG=',IXG,',',
     *                  ( GPOP(IXG,IYGMID,IZG),',',IZG=1,N2Z )
                  END DO

            END IF

C WRITE DATA TO LASSPAT.CSV ----------------------------------------

            IF (LASSPATSW.GT.0) THEN

C SEED ENERGY SPATIAL DISTRIBUTION ---------------------------------

                  DO IP=1,NPASS

                        WRITE(24,'(A13,I3,A26,I8,A7,EN15.7,A5)'),
     *                  'ESEED(IP=',IP,';IBX;IBY;IZGMID) FOR JT='
     *                  ,JT,'TIME=',TIME,' ps'

                        WRITE(24,'(A2,<NBX>(A4,I3,A2))'),',',
     *                  ('IBX=',IBX,',',IBX=1,NBX)

                        DO IBY=1,NBY
                              WRITE(24,'(A4,I3,A2,<NBX>(E15.7,A2))'),
     *                        'IBY=',IBY,',',
     *                        ( ESEED(IP,IBX,IBY,IZGMID),',',IBX=1,NBX )
                        END DO

C ASE ENERGY SPATIAL DISTRIBUTION ----------------------------------

                        WRITE(24,'(A13,I3,A26,I8,A7,EN15.7,A5)'),
     *                  'EASE(IP=',IP,';IBX;IBY;IZGMID) FOR JT='
     *                  ,JT,'TIME=',TIME,' ps'

                        WRITE(24,'(A2,<NBX>(A4,I3,A2))'),',',
     *                  ('IBX=',IBX,',',IBX=1,NBX)

                        DO IBY=1,NBY
```

```
                  WRITE(24,'(A4,I3,A2,<NBX>(E15.7,A2))'),
     *            'IBY=',IBY,',',
     *            ( EASE(IP,IBX,IBY,IZGMID),',',IBX=1,NBX )
               END DO

            END DO

C PUMP Y ENERGY SPATIAL DISTRIBUTION -------------------------------

            DO JP=3,4

               WRITE(24,'(A13,I3,A26,I8,A7,EN15.7,A5)'),
     *         'EPUMPY(JP=',JP,';JBX;JBZ;IYGMID) FOR JT='
     *         ,JT,'TIME=',TIME,' ps'

               WRITE(24,'(A2,<MPX>(A4,I3,A2))'),',',
     *         ('JBX=',JBX,',',JBX=1,MPX)

               DO JBZ=1,MPZ
                  WRITE(24,'(A4,I3,A2,<MPX>(E15.7,A2))'),
     *            'JBZ=',JBZ,',',
     *            ( EPUMPY(JP,JBX,JBZ,IYGMID),',',JBX=1,MPX )
               END DO

            END DO

C PUMP Z ENERGY SPATIAL DISTRIBUTION -------------------------------

            DO IP=1,2

               WRITE(24,'(A13,I3,A26,I8,A7,EN15.7,A5)'),
     *         'EPUMPZ(IP=',IP,';IBX;IBY;IZGMID) FOR JT='
     *         ,JT,'TIME=',TIME,' ps'

               WRITE(24,'(A2,<NPX>(A4,I3,A2))'),',',
     *         ('IBX=',IBX,',',IBX=1,NPX)

               DO IBY=1,NPY
                  WRITE(24,'(A4,I3,A2,<NPX>(E15.7,A2))'),
     *            'IBY=',IBY,',',
     *            ( EPUMPZ(IP,IBX,IBY,IZGMID),',',IBX=1,NPX )
               END DO

            END DO


         END IF

      END IF

      RETURN
      END
```

……

```
      SUBROUTINE PMICRT02

C PUMP BEAM INITIALIZATION CALCULATIONS.
C THIS IS FOR Y-PROPAGATING PUMP BEAMS ONLY

C
C THE PUMP BEAM INITIALIZATION CALCULATIONS IS CARRIED OUT BY
C TWO SUBROUTINES:
C
C PMICRT01    'MASTER' SUBROUTINE - CALLS SLAVE SUBROUTINE TO DO
C                    SPECIFIC CALCULATIONS FOR THE DIFFERENT PUMP PASSES

C
C GEOCRT01   'SLAVE' SUBROUTINE - PERFORMS GEOMETRICAL CALCULATIONS
C        USING DUMMY ARGUMENTS FOR THE DIFFERENT PUMP (AND SEED?)
C        PASSES.
C

C PUMP PASS NOTATION: -------------------------------------------
C
C PASS #      DIRECTION    COUNTER VARIABLES      TOTAL VARIABLES
C ------ ---------  ----------------    --------------
C 1           POS. Z       I                            N
C 2           NEG. Z       I                            N
C 3           POS. Y       J                            M
C 4           NEG. Y       J                            M
C 5           POS. X       K                            L
C 6           NEG. X       K                            L
C


      INCLUDE 'Lascrt01.inc'

C DECLARATION OF INPUT AND OUTPUT VARIABLES (DUMMY ARGUMENTS) --------

      INTEGER PASSNUMBER
      CHARACTER*4 PASSTYPE

      INTEGER N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W
      INTEGER NBU,NBV,WDIRN,DUMPPRN
      DOUBLE PRECISION DW,UPROPANGLE,VPROPANGLE
      DOUBLE PRECISION UDIVGANGLE,VDIVGANGLE,UCENTRE,VCENTRE
      DOUBLE PRECISION UWIDTH,VWIDTH

C DECLARATION OF TEMPORARY VARIABLES

      DOUBLE PRECISION TEMPVOL

C CO-ORDINATE CONVERSIONS FOR THE Y-PROPAGATING PUMP BEAMS:
C
C REMEMBER: "U CROSS V EQUALS W"
C
C       Z -----> U
C   X -----> V
C       Y -----> W (DIRECTION OF PROPAGATION).
C
```

```
        PRINT '(A30)','START PMICRT01'

C CALCULATE BEAMLET X-SCN. AREAS. ---------------------------------
C THIS IS ONLY FOR THE START OF EACH PASS. OTHERS NEEDED?


        DO JP=3,4

        PUMPDBA(JP)=XBEAMWIDTHPY(JP)*ZBEAMWIDTHPY(JP)/(DBLE(MPX*MPZ))


                WRITE(12,'(A4,I3,A17,EN23.10)'),
     *          'JP=',JP,' PUMPDBA(JP)= ',PUMPDBA(JP)

        END DO



C COPY INPUT DATA TO THE INPUT DUMMY ARGUMENTS
        N1U=N1Z
        N2U=N2Z
        N3U=N3Z

        N1V=N1X
        N2V=N2X
        N3V=N3X

        N1W=N1Y
        N2W=N2Y
        N3W=N3Y

        NBU=MPZ
        NBV=MPX
        DW=DY

        WRITE(13,'(A10,D10.3)'),'DW=',DW

        IF (DUMPOU.EQ.6) THEN
                DUMPPRN=2
        ELSE
                DUMPPRN=0
        END IF

C LOOP THRU ALL PASSES FOR Y-PROPAGATING PUMP BEAMS

        DO JP=3,4

C               EVEN NUMBERED PASSES TRAVEL IN THE 'NEGATIVE W' DIRECTION

                IF (MOD(IP,2).EQ.0) THEN
                                WDIRN=-1
                ELSE
                                WDIRN=1
```

```
                    END IF

                    UPROPANGLE=ZTHPRPGPY(JP)
                    VPROPANGLE=XTHPRPGPY(JP)
                    UDIVGANGLE=ZTHDIVGPY(JP)
                    VDIVGANGLE=XTHDIVGPY(JP)
                    UCENTRE=ZBEAMCENTREPY(JP)
                    VCENTRE=XBEAMCENTREPY(JP)
                    UWIDTH=ZBEAMWIDTHPY(JP)
                    VWIDTH=XBEAMWIDTHPY(JP)

          WRITE(13,'(A10,D10.3)'),'VWIDTH=',VWIDTH

          PASSNUMBER=JP
          PASSTYPE='PMPY'


C CALL GEOCRT02

                    CALL GEOCRT02(PASSNUMBER,PASSTYPE,
     *              NBU,NBV,WDIRN,DUMPPRN
     *    ,DW,UPROPANGLE,VPROPANGLE,UDIVGANGLE,VDIVGANGLE,UCENTRE
     *    ,VCENTRE,UWIDTH,VWIDTH,N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W)

C NOTE: ASSIGNMENT OF THE O/P DATA IS NOW PERFORMED BY GEOCRT02
C THE FOLLOWING SECTION PRINTS DATA IF DUMPOU.EQ.6

                    IF (DUMPOU.EQ.6) THEN

                              DO JYG=1,N2Y
                                        DO JZG=1,N2Z
                                                  WRITE(13,'(A7,I3,A7,I3,A10,I4,A10,I4)'),
     *                                  'JYG=',JYG,'JZG=',JZG,'JBZ1PY=',
     *        JBZ1PY(JP,JZG,JYG),'JBZ2PY=',JBZ2PY(JP,JZG,JYG)
                                        END DO

                                        DO JXG=1,N2X
                                                  WRITE(13,'(A7,I3,A7,I3,A10,I4,A10,I4)'),
     *                                  'JYG=',JYG,'JXG=',JXG,'JBX1PY=',
     *        JBX1PY(JP,JXG,JYG),'JBX2PY=',JBX2PY(JP,JXG,JYG)
                                        END DO

                                        DO JBZ=1,MPZ
                                                  DO JBX=1,MPX
                                                            DO JZG=1,N2Z
                                                                      DO JXG=1,N2X



          TEMPVOL=FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG)

                                                                                IF ( TEMPVOL.NE.(0.D0) )
 THEN

          WRITE(13,'(5(A7,I3),A10,D10.3)'),
```

```
     *
       'JYG=',JYG,'JBX=',JBX,'JBZ=',JBZ,
     *                                                        'JXG=',JXG,'JZG=',JZG,
     *                                                        'FVOLPY=',
     *
       FVOLPY(JP,JBX,JBZ,JXG,JYG,JZG)
                                                              END IF

                                                       END DO
                                                  END DO
                                             END DO
                                        END DO
                                   END DO


                                   DO JY=0,NY
                                        DO JBZ=0,MPZ

                                             WRITE(13,'(A7,I3,A7,I3,A10,D10.3)'),
     *                                       'JY=',JY,'JBZ=',JBZ,'ZVPY=',ZVPY(JP,JBZ,JY)

                                        END DO

                                        DO JBX=0,MPX

                                             WRITE(13,'(A7,I3,A7,I3,A10,D10.3)'),
     *                                       'JY=',JY,'JBX=',JBX,'XVPY=',XVPY(JP,JBX,JY)

                                        END DO

                                   END DO


                              END IF

C END OF 'PASS LOOP'

                         END DO


C CALCULATE THE PEAK INTENSITY FOR PUMP ---------------------------
C THIS IS ASSUMED TO BE FOR PASS 3


                         IF (RPUMPSW.EQ.0) THEN

C                             READ DATA FOR RADIAL PROFILE OF PUMP BEAM FROM DATA FILE

                              RFUDGE=0.D0

                              DO JBX=1,MPX
                                   DO JBZ=1,MPZ
                                        READ(55,*) DATA55
                                        PUMPRDATA(JBX,JBZ)=DATA55
                                   END DO
```

```
                    END DO


                    DO JBX=2,MPX
                            DO JBZ=2,MPZ
                                    RFUDGE=RFUDGE + ( PUMPDBA(3)*
     *                              ( PUMPRDATA(JBX,JBZ)+PUMPRDATA(JBX,JBZ-1)+
     *                                PUMPRDATA(JBX-1,JBZ)+PUMPRDATA(JBX-1,JBZ-1) )
     *                              *0.25D0 )

                            END DO
                    END DO


            ELSE IF (RPUMPSW.EQ.2) THEN

C                   RADIAL(SPATIAL) PROFILE IS UNIFORM

                    RFUDGE=ZBEAMWIDTHPY(3)*XBEAMWIDTHPY(3)

            ELSE

C                   NON-PERMISSIBLE VALUE FOR RSEEDSW
                    PRINT '(A35)','NON-PERMISSIBLE VALUE FOR RSEEDSW'
                    WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR RSEEDSW'
                    IQUIT=1
                    RETURN

            END IF


            IF (TPUMPSW.EQ.0) THEN

C                   READ DATA FOR TEMPORAL PROFILE OF PUMP BEAM FROM DATA FILE

                    TFUDGE=0.D0

                    DO IDATA=1,NPT54

                            READ(54,*),DATA54
                            TFUDGE=TFUDGE+(DATA54*DT54)

                    END DO


            ELSE IF (TPUMPSW.EQ.1) THEN

C                   TEMPORAL PROFILE IS GAUSSIAN

                    TFUDGE=(DSQRT(PI) / (2.D0*DSQRT(DLOG(2.D0))) )*PUMPDURN


            ELSE IF (TSEEDSW.EQ.2) THEN
```

```
C               TEMPORAL PROFILE IS UNIFORM

                TFUDGE=1.D0*PUMPDURN


          ELSE

C               NON-PERMISSIBLE VALUE FOR TSEEDSW
                PRINT '(A35)','NON-PERMISSIBLE VALUE FOR TSEEDSW'
                WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR TSEEDSW'
                IQUIT=1
                RETURN

          END IF

          PUMPPKINTENS3=( PUMPENERGY(3) )/(RFUDGE*TFUDGE)
          PUMPPKINTENS4=( PUMPENERGY(4) )/(RFUDGE*TFUDGE)

          WRITE(12,'(A30,EN23.10,A9)'),
     *          'PUMP RFUDGE=',RFUDGE,' cm^2'
          WRITE(12,'(A30,EN23.10,A9)'),
     *          'PUMP TFUDGE=',TFUDGE,' ps'


          WRITE(12,'(A30,EN23.10,A9)'),
     *          'PASS 3 PUMP PEAK INTENSITY=',PUMPPKINTENS3,' MW/cm^2'
          WRITE(12,'(A30,EN23.10,A9)'),
     *          'PASS 4 PUMP PEAK INTENSITY=',PUMPPKINTENS4,' MW/cm^2'




          RETURN
          END
```

……

```
      SUBROUTINE PMZCRT01

C PUMP BEAM INITIALIZATION CALCULATIONS. - Z PROPAGATING
C
C PUMP PASS NOTATION: -------------------------------------------
C
C PASS #       DIRECTION      COUNTER VARIABLES       TOTAL VARIABLES
C ------ --------  ---------------  --------------
C 1            POS. Z         I                              N
C 2            NEG. Z         I                              N
C 3            POS. Y         J                              M
C 4            NEG. Y         J                              M
C 5            POS. X         K                              L
C 6            NEG. X         K                              L
C

      INCLUDE 'Lascrt01.inc'

C DECLARATION OF INPUT AND OUTPUT VARIABLES (DUMMY ARGUMENTS) ---------

      INTEGER PASSNUMBER
      CHARACTER*4 PASSTYPE

      INTEGER N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W
      INTEGER NBU,NBV,WDIRN,DUMPPRN
      DOUBLE PRECISION DW,UPROPANGLE,VPROPANGLE
      DOUBLE PRECISION UDIVGANGLE,VDIVGANGLE,UCENTRE,VCENTRE
      DOUBLE PRECISION UWIDTH,VWIDTH

C DECLARATION OF TEMPORARY VARIABLES

      DOUBLE PRECISION TEMPVOL

C CO-ORDINATE CONVERSIONS FOR THE Z-PROPAGATING PUMP BEAMS:
C
C REMEMBER: "U CROSS V EQUALS W"
C
C       X -----> U
C   Y -----> V
C       Z -----> W (DIRECTION OF PROPAGATION).
C


      PRINT '(A30)','START PMZCRT01'

C CALCULATE BEAMLET X-SCN. AREAS. ----------------------------------


      DO IP=1,2
         PUMPDBA(IP)=YBEAMWIDTHPZ(IP)*XBEAMWIDTHPZ(IP)/(DBLE(NPY*NPX))


         WRITE(12,'(A4,I3,A17,EN23.10)'),
     *   'IP=',IP,' PUMPDBA(IP)= ',PUMPDBA(IP)
```

```
      END DO


C COPY INPUT DATA TO THE INPUT DUMMY ARGUMENTS
      N1U=N1X
      N2U=N2X
      N3U=N3X

      N1V=N1Y
      N2V=N2Y
      N3V=N3Y

      N1W=N1Z
      N2W=N2Z
      N3W=N3Z

      NBU=NPX
      NBV=NPY
      DW=DZ

      WRITE(13,'(A10,D10.3)'),'DW=',DW

      IF (DUMPOU.EQ.6) THEN
            DUMPPRN=2
      ELSE
            DUMPPRN=0
      END IF

C LOOP THRU ALL PASSES FOR Y-PROPAGATING PUMP BEAMS

      DO IP=1,2

C           EVEN NUMBERED PASSES TRAVEL IN THE 'NEGATIVE W' DIRECTION

            IF (MOD(IP,2).EQ.0) THEN
                        WDIRN=-1
            ELSE
                        WDIRN=1
            END IF

            UPROPANGLE=XTHPRPGPZ(IP)
            VPROPANGLE=YTHPRPGPZ(IP)
            UDIVGANGLE=XTHDIVGPZ(IP)
            VDIVGANGLE=YTHDIVGPZ(IP)
            UCENTRE=XBEAMCENTREPZ(IP)
            VCENTRE=YBEAMCENTREPZ(IP)
            UWIDTH=XBEAMWIDTHPZ(IP)
            VWIDTH=YBEAMWIDTHPZ(IP)

      WRITE(13,'(A10,D10.3)'),'VWIDTH=',VWIDTH

      PASSNUMBER=IP
      PASSTYPE='PMPZ'
```

```
C CALL GEOCRT02

                CALL GEOCRT02(PASSNUMBER,PASSTYPE,
     *          NBU,NBV,WDIRN,DUMPPRN
     *    ,DW,UPROPANGLE,VPROPANGLE,UDIVGANGLE,VDIVGANGLE,UCENTRE
     *    ,VCENTRE,UWIDTH,VWIDTH,N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W)

C NOTE: ASSIGNMENT OF THE O/P DATA IS NOW PERFORMED BY GEOCRT02
C THE FOLLOWING SECTION PRINTS DATA IF DUMPOU.EQ.6

                IF (DUMPOU.EQ.6) THEN

                        DO IZG=1,N2Z
                                DO IXG=1,N2X
                                        WRITE(13,'(A7,I3,A7,I3,A10,I4,A10,I4)'),
     *                          'JYG=',JYG,'JZG=',JZG,'IBX1PZ=',
     *          IBX1PZ(IP,JZG,JYG),'IBX2PZ=',IBX2PZ(IP,JZG,JYG)
                                END DO

                                DO IYG=1,N2Y
                                        WRITE(13,'(A7,I3,A7,I3,A10,I4,A10,I4)'),
     *                          'JYG=',JYG,'JXG=',JXG,'IBY1PZ=',
     *          IBY1PZ(IP,JXG,JYG),'IBY2PZ=',IBY2PZ(IP,JXG,JYG)
                                END DO

                                DO IBX=1,NPX
                                        DO IBY=1,NPY
                                                DO IXG=1,N2X
                                                        DO IYG=1,N2Y

C MAKE SURE THAT ORDER OF INDEXES IS CORRECT!


        TEMPVOL=FVOLPZ(IP,IBY,IBX,IXG,IYG,IZG)
                                                IF ( TEMPVOL.NE.(0.D0) ) THEN
                                                        WRITE(13,'(5(A7,I3),A10,D10.3)'),
     *                                                  'IYG=',IYG,'IBY=',IBY,'IBX=',IBX,
     *                                                          'IXG=',IXG,'IZG=',IZG,
     *                                                          'FVOLPZ=',
     *                                                  FVOLPZ(IP,IBY,IBX,IXG,IYG,IZG)
                                                END IF

                                                        END DO
                                                END DO
                                        END DO
                                END DO
                        END DO


C       =UBEAMANGLE(0:NBMAX)
C       =VBEAMANGLE(0:NBMAX)

                        DO IZ=0,NZ
                                DO IBX=0,NPX
```

```
                                        WRITE(13,'(A7,I3,A7,I3,A10,D10.3)'),
     *                                  'IZ=',IZ,'IBX=',IBX,'XVPZ=',XVPZ(IP,IBX,IZ)

                                END DO

                                DO IBY=0,NPY

                                        WRITE(13,'(A7,I3,A7,I3,A10,D10.3)'),
     *                                  'IZ=',IZ,'IBY=',IBY,'YVPZ=',YVPZ(IP,IBY,IZ)

                                END DO

                        END DO


                END IF

C END OF 'PASS LOOP'

        END DO



C CALCULATE THE PEAK INTENSITY FOR PUMP ------------------------------
C THIS IS ASSUMED TO BE FOR PASS 3


        IF (RPUMPSW.EQ.0) THEN

C               READ DATA FOR RADIAL PROFILE OF PUMP BEAM FROM DATA FILE

                RFUDGE=0.D0

                DO IBY=1,NPY
                        DO IBX=1,NPX
                                READ(55,*) DATA55
                                PUMPRDATA(IBY,IBX)=DATA55
                        END DO
                END DO


                DO IBY=2,NPY
                        DO IBX=2,NPX
                                RFUDGE=RFUDGE + ( PUMPDBA(3)*
     *                          ( PUMPRDATA(IBY,IBX)+PUMPRDATA(IBY,IBX-1)+
     *                            PUMPRDATA(IBY-1,IBX)+PUMPRDATA(IBY-1,IBX-1) )
     *                          *0.25D0 )

                        END DO
                END DO


        ELSE IF (RPUMPSW.EQ.2) THEN
```

```
C               RADIAL(SPATIAL) PROFILE IS UNIFORM

                RFUDGE=XBEAMWIDTHPZ(1)*YBEAMWIDTHPZ(1)

          ELSE

C               NON-PERMISSIBLE VALUE FOR RSEEDSW
                PRINT '(A35)','NON-PERMISSIBLE VALUE FOR RSEEDSW'
                WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR RSEEDSW'
                IQUIT=1
                RETURN

          END IF


          IF (TPUMPSW.EQ.0) THEN

C               READ DATA FOR TEMPORAL PROFILE OF PUMP BEAM FROM DATA FILE

                TFUDGE=0.D0

                DO IDATA=1,NPT54

                        READ(54,*),DATA54
                        TFUDGE=TFUDGE+(DATA54*DT54)

                END DO


          ELSE IF (TPUMPSW.EQ.1) THEN

C               TEMPORAL PROFILE IS GAUSSIAN

                TFUDGE=(DSQRT(PI) / (2.D0*DSQRT(DLOG(2.D0))) )*PUMPDURN


          ELSE IF (TSEEDSW.EQ.2) THEN

C               TEMPORAL PROFILE IS UNIFORM

                TFUDGE=1.D0*PUMPDURN


          ELSE

C               NON-PERMISSIBLE VALUE FOR TSEEDSW
                PRINT '(A35)','NON-PERMISSIBLE VALUE FOR TSEEDSW'
                WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR TSEEDSW'
                IQUIT=1
                RETURN

          END IF

          PUMPPKINTENS1=( PUMPENERGY(1) )/(RFUDGE*TFUDGE)
```

```
      PUMPPKINTENS2=( PUMPENERGY(2) )/(RFUDGE*TFUDGE)

      WRITE(12,'(A30,EN23.10,A9)'),
     *          'PUMP RFUDGE=',RFUDGE,' cm^2'
      WRITE(12,'(A30,EN23.10,A9)'),
     *          'PUMP TFUDGE=',TFUDGE,' ps'


      WRITE(12,'(A30,EN23.10,A9)'),
     *          'PASS 1 PUMP PEAK INTENSITY=',PUMPPKINTENS1,' MW/cm^2'
      WRITE(12,'(A30,EN23.10,A9)'),
     *          'PASS 2 PUMP PEAK INTENSITY=',PUMPPKINTENS2,' MW/cm^2'


      RETURN
      END
..........................................................................................................................
```

```
          SUBROUTINE PRFCRT01


          INCLUDE 'Lascrt01.inc'

          CHARACTER*8 CHRDATE
          CHARACTER*10 CHRTIME

C WRITE INPUT ENERGIES TO SUMMARY.OUT ------------------------------

          DO IP=1,2
                  WRITE(12,'(A7,I3,A20,EN23.10,A4)'),'PASS=',IP,
     *            'INPUT PUMP ENERGY=',EPUMPTOTALINPZ(IP),' uJ'

          END DO


          DO JP=3,4
                  WRITE(12,'(A7,I3,A20,EN23.10,A4)'),'PASS=',JP,
     *            'INPUT PUMP ENERGY=',EPUMPTOTALINPY(JP),' uJ'

          END DO

          WRITE(12,'(A5)'),''

          DO IP=1,NPASS
                  WRITE(12,'(A7,I3,A20,EN23.10,A4)'),'PASS=',IP,
     *            'INPUT SEED ENERGY=',ESEEDTOTALIN(IP),' uJ'
          END DO

          WRITE(12,'(A5)'),''

          DO IP=1,NPASS
                  WRITE(12,'(A7,I3,A20,EN23.10,A4)'),'PASS=',IP,
     *            'INPUT ASE ENERGY=',EASETOTALIN(IP),' uJ'
          END DO

          WRITE(12,'(A5)'),''

C WRITE OUTPUT ENERGIES TO SUMMARY.OUT ------------------------------

          DO JP=1,4
                  WRITE(12,'(A7,I3,A20,EN23.10,A4)'),'PASS=',JP,
     *            'OUTPUT PUMP ENERGY=',EPUMPTOTALOUT(JP),' uJ'

          END DO

          WRITE(12,'(A5)'),''

          DO IP=1,NPASS
                  WRITE(12,'(A7,I3,A20,EN23.10,A4)'),'PASS=',IP,
     *            'OUTPUT SEED ENERGY=',ESEEDTOTALOUT(IP),' uJ'
          END DO

          WRITE(12,'(A5)'),''
```

```
        DO IP=1,NPASS
                WRITE(12,'(A7,I3,A20,EN23.10,A4)'),'PASS=',IP,
    *           'OUTPUT ASE ENERGY=',EASETOTALOUT(IP),' uJ'
        END DO

        WRITE(12,'(A5)'),''

C WRITE THE FINISH TIME TO SUMMARY.OUT ------------------------------

        CALL DATE_AND_TIME (CHRDATE,CHRTIME)
        WRITE (12,'(A25,A10)'),'FINISH TIME (hhmmss.sss)=',CHRTIME
        WRITE (12,'(A25,A10)'),'FINISH DATE (YYYYMMDD) =',CHRDATE
        PRINT '(A16,A8)','DATE(YYYYMMDD)=',CHRDATE
        PRINT '(A23,A10)','END TIME(hhmmss.sss)=',CHRTIME



        RETURN
        END
.........................................................................................................
```

```
SUBROUTINE PRPCRT01


C PROPAGATION ROUTINE FOR LASCRT01.F
C
C

   INCLUDE 'Lascrt01.inc'

C LOCAL DECLARATIONS -----------------------------------------------

        DOUBLE PRECISION ESEEDTEMPORARY(NPMAX,NBMAX,NBMAX)
        DOUBLE PRECISION EASETEMPORARY(NPMAX,NBMAX,NBMAX)
        DOUBLE PRECISION EPUMPTEMPORARY(NPMAX,NBMAX,NBMAX)

C INITIALIZE DATA STORAGE ARRAYS (SLAB ENERGIES)


        DO IP=1,NPASS
                ESEEDSLAB(IP)=0.D0
                EASESLAB(IP)=0.D0
        END DO

        DO JP=1,4
                EPUMPSLAB(JP)=0.D0
        END DO

C CALCULATE SLAB ENERGIES AND STORE TEMPORARY ENERGIES ---------------

        DO IBX=1,NBX
                DO IBY=1,NBY

C                       ODD PASSES
                        DO IP=1,NPFINALODD,2
                                ESEEDSLAB(IP)=ESEEDSLAB(IP)+ESEED(IP,IBX,IBY,NZ)
                                EASESLAB(IP)=EASESLAB(IP)+EASE(IP,IBX,IBY,NZ)

                                ESEEDTEMPORARY(IP,IBX,IBY)=ESEED(IP,IBX,IBY,NZ)
                                EASETEMPORARY(IP,IBX,IBY)=EASE(IP,IBX,IBY,NZ)
                        END DO

C                       EVEN PASSES
                        DO IP=2,NPFINALEVEN,2
                                ESEEDSLAB(IP)=ESEEDSLAB(IP)+ESEED(IP,IBX,IBY,1)
                                EASESLAB(IP)=EASESLAB(IP)+EASE(IP,IBX,IBY,1)

                                ESEEDTEMPORARY(IP,IBX,IBY)=ESEED(IP,IBX,IBY,1)
                                EASETEMPORARY(IP,IBX,IBY)=EASE(IP,IBX,IBY,1)
                        END DO

                END DO
        END DO

        DO JBX=1,MPX
                DO JBZ=1,MPZ
                        EPUMPSLAB(3)=EPUMPSLAB(3)+EPUMPY(3,JBX,JBZ,NY)
```

```
                    EPUMPSLAB(4)=EPUMPSLAB(4)+EPUMPY(4,JBX,JBZ,1)

                    EPUMPTEMPORARY(3,JBX,JBZ)=EPUMPY(3,JBX,JBZ,NY)
                    EPUMPTEMPORARY(4,JBX,JBZ)=EPUMPY(4,JBX,JBZ,1)
                END DO
            END DO

            DO IBX=1,NPX
                DO IBY=1,NPY
                    EPUMPSLAB(1)=EPUMPSLAB(1)+EPUMPZ(1,IBX,IBY,NZ)
                    EPUMPSLAB(2)=EPUMPSLAB(2)+EPUMPZ(2,IBX,IBY,1)

                    EPUMPTEMPORARY(1,IBX,IBY)=EPUMPZ(1,IBX,IBY,NZ)
                    EPUMPTEMPORARY(2,IBX,IBY)=EPUMPZ(2,IBX,IBY,1)
                END DO
            END DO



C CALCULATE TOTAL ENERGIES ----------------------------------------

            DO IP=1,NPASS
                ESEEDTOTALOUT(IP)=ESEEDTOTALOUT(IP)+ESEEDSLAB(IP)
                EASETOTALOUT(IP)=EASETOTALOUT(IP)+EASESLAB(IP)
            END DO

            EPUMPTOTALOUT(1)=EPUMPTOTALOUT(1)+EPUMPSLAB(1)
            EPUMPTOTALOUT(2)=EPUMPTOTALOUT(2)+EPUMPSLAB(2)

            EPUMPTOTALOUT(3)=EPUMPTOTALOUT(3)+EPUMPSLAB(3)
            EPUMPTOTALOUT(4)=EPUMPTOTALOUT(4)+EPUMPSLAB(4)

C PROPAGATE SEED AND ASE BEAMS IN THE POSITIVE Z DIRECTION -----------
C (ODD PASSES)

            DO IZ=NZ-1,1,-1
                DO IBX=1,NBX
                    DO IBY=1,NBY
                        DO IP=1,NPFINALODD,2
                            ESEED(IP,IBX,IBY,IZ+1)=ESEED(IP,IBX,IBY,IZ)
                            EASE(IP,IBX,IBY,IZ+1)=EASE(IP,IBX,IBY,IZ)
                        END DO
                    END DO
                END DO
            END DO


C PROPAGATE SEED AND ASE BEAMS IN THE NEGATIVE Z DIRECTION -----------
C (EVEN PASSES)

            DO IZ=2,NZ
                DO IBX=1,NBX
                    DO IBY=1,NBY
                        DO IP=2,NPFINALEVEN,2
                            ESEED(IP,IBX,IBY,IZ-1)=ESEED(IP,IBX,IBY,IZ)
```

```
                              EASE(IP,IBX,IBY,IZ-1)=EASE(IP,IBX,IBY,IZ)
                        END DO
                  END DO
            END DO
      END DO


C PROPAGATE PUMP BEAMS IN THE POSITIVE Y DIRECTION --------------------
C (PUMP PASS 3)

      DO JY=NY-1,1,-1
            DO JBX=1,MPX
                  DO JBZ=1,MPZ
                        EPUMPY(3,JBX,JBZ,JY+1)=EPUMPY(3,JBX,JBZ,JY)
                  END DO
            END DO
      END DO

C PROPAGATE PUMP BEAMS IN THE NEGATIVE Y DIRECTION --------------------
C (PUMP PASS 4)

      DO JY=2,NY
            DO JBX=1,MPX
                  DO JBZ=1,MPZ
                        EPUMPY(4,JBX,JBZ,JY-1)=EPUMPY(4,JBX,JBZ,JY)
                  END DO
            END DO
      END DO

C PROPAGATE PUMP BEAMS IN THE POSITIVE Z DIRECTION --------------------
C (PUMP PASS 1)

      DO IZ=NZ-1,1,-1
            DO IBX=1,NPX
                  DO IBY=1,NPY
                        EPUMPZ(1,IBX,IBY,IZ+1)=EPUMPZ(1,IBX,IBY,IZ)
                  END DO
            END DO
      END DO

C PROPAGATE PUMP BEAMS IN THE NEGATIVE Z DIRECTION --------------------
C (PUMP PASS 2)

      DO IZ=2,NZ
            DO IBX=1,NPX
                  DO IBY=1,NPY
                        EPUMPZ(2,IBX,IBY,IZ-1)=EPUMPZ(2,IBX,IBY,IZ)
                  END DO
            END DO
      END DO
```

```
C CALCULATE MIRROR REFLECTIVITIES FOR ALL BEAMS ---------------------

        DO IBX=1,NBX
                DO IBY=1,NBY

C                       ODD PASSES
                        DO IP=1,NPFINALODD,2
                                ESEED(IP,IBX,IBY,1)=MIRREFL(IP)
   *                            *ESEEDTEMPORARY(IPSOURCE(IP),IBX,IBY)
                                EASE(IP,IBX,IBY,1)=ASEREFL(IP)
   *                            *EASETEMPORARY(IPSOURCE(IP),IBX,IBY)

                        END DO

C                       EVEN PASSES
                        DO IP=2,NPFINALEVEN,2
                                ESEED(IP,IBX,IBY,NZ)=MIRREFL(IP)
   *                            *ESEEDTEMPORARY(IPSOURCE(IP),IBX,IBY)
                                EASE(IP,IBX,IBY,NZ)=ASEREFL(IP)
   *                            *EASETEMPORARY(IPSOURCE(IP),IBX,IBY)
                        END DO

                END DO
        END DO

C IF RP34 .GT. ZERO, THEN REFLECT THE PUMP ENERGY FROM PASS 3 TO PASS 4

        IF (RP34.GT.(0.D0)) THEN
                DO JBX=1,MPX
                        DO JBZ=1,MPZ
                                EPUMPY(4,JBX,JBZ,NY)=RP34*EPUMPTEMPORARY(3,JBX,JBZ)
                        END DO
                END DO
        END IF

C IF RP12 .GT. ZERO, THEN REFLECT THE PUMP ENERGY FROM PASS 1 TO PASS 2

        IF (RP12.GT.(0.D0)) THEN
                DO IBX=1,NPX
                        DO IBY=1,NPY
                                EPUMPZ(2,IBX,IBY,NZ)=RP12*EPUMPTEMPORARY(1,IBX,IBY)
                        END DO
                END DO
        END IF


C INJECT NEW ENERGY IN SEED AND PUMP BEAMS --------------------------

        DO IBX=1,NBX
                DO IBY=1,NBY
                        ESEED(INJPT,IBX,IBY,IZSTARTINJPT)=ESEEDINP(INJPT,IBX,IBY)

C CHECK THE IZSTARTINJPT STUFF
                        EASE(INJPT,IBX,IBY,IZSTARTINJPT)=
   *                    EASE(INJPT,IBX,IBY,IZSTARTINJPT)
   *                    +EASEINP(INJPT,IBX,IBY)
```

```
         END DO
      END DO

      DO JBX=1,MPX
         DO JBZ=1,MPZ
            EPUMPY(3,JBX,JBZ,1)=EPUMPINPY(3,JBX,JBZ)

C IF RP34 .LE. ZERO, THEN INJECT NEW ENERGY INTO PUMP PASS 4.

            IF (RP34.LE.(0.D0)) THEN
               EPUMPY(4,JBX,JBZ,NY)=EPUMPINPY(4,JBX,JBZ)
            END IF

            IF (DUMPOU.EQ.8) THEN
               WRITE(13,'(A5,I4,A5,I4,A10)'),
     *         'JBX=',JBX,'JBZ=',JBZ,'*****'
               WRITE(13,'(A24,D13.5)'),
     *         'EPUMPY(4,JBX,JBZ,NY)=',EPUMPY(4,JBX,JBZ,NY)
               WRITE(13,'(A24,D13.5)'),
     *         'EPUMPY(3,JBX,JBZ,1)=',EPUMPY(3,JBX,JBZ,1)
            END IF

         END DO
      END DO

      DO IBX=1,NPX
         DO IBY=1,NPY
            EPUMPZ(1,IBX,IBY,1)=EPUMPINPZ(1,IBX,IBY)

C IF RP12 .LE. ZERO, THEN INJECT NEW ENERGY INTO PUMP PASS 2.

            IF (RP12.LE.(0.D0)) THEN
               EPUMPZ(2,IBX,IBY,NZ)=EPUMPINPZ(2,IBX,IBY)
            END IF

            IF (DUMPOU.EQ.8) THEN
               WRITE(13,'(A5,I4,A5,I4,A10)'),
     *         'IBX=',IBX,'IBY=',IBY,'*****'
               WRITE(13,'(A24,D13.5)'),
     *         'EPUMPZ(2,IBX,IBY,NZ)=',EPUMPZ(2,IBX,IBY,NZ)
               WRITE(13,'(A24,D13.5)'),
     *         'EPUMPZ(1,IBX,IBY,1)=',EPUMPZ(1,IBX,IBY,1)
            END IF

         END DO
      END DO


      RETURN
      END
```

……

```
      SUBROUTINE SDICRT02


C     SEED BEAM INITIALIZATION

C THE CODE IS COPIED FROM INICRT01.F


      INCLUDE 'Lascrt01.inc'

C REMEMBER: DUMMY ARGUMENTS PASSED TO THE SUBROUTINE CANNOT
C APPEAR IN COMMON BLOCKS

C DECLARATION OF INPUT AND OUTPUT VARIABLES (DUMMY ARGUMENTS) --------

      INTEGER PASSNUMBER
      CHARACTER*4 PASSTYPE

      INTEGER N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W
      INTEGER NBU,NBV,WDIRN,DUMPPRN
      DOUBLE PRECISION DW,UPROPANGLE,VPROPANGLE
      DOUBLE PRECISION UDIVGANGLE,VDIVGANGLE,UCENTRE,VCENTRE
      DOUBLE PRECISION UWIDTH,VWIDTH

C DECLARATION OF TEMPORARY VARIABLES

      INTEGER IDATA
      DOUBLE PRECISION TEMPVOL,RFUDGE,TFUDGE

C CO-ORDINATE CONVERSIONS FOR THE SEED AND ASE BEAMS:
C
C     X -----> U
C  Y -----> V
C     Z -----> W (DIRECTION OF PROPAGATION).
C


      PRINT '(A30)','START SDICRT02'

C CALCULATE BEAMLET X-SCN. AREAS. ----------------------------------
C THIS IS ONLY FOR THE START OF EACH PASS. OTHERS NEEDED?


      DO IP=1,NPASS
            SEEDDBA(IP)=XBEAMWIDTH(IP)*YBEAMWIDTH(IP)/(DBLE(NBX*NBY))

            WRITE(12,'(A4,I3,A17,EN23.10)'),
   *        'IP=',IP,' SEEDDBA(IP)= ',SEEDDBA(IP)

            IF (SEEDDBA(IP).EQ.(0.D0)) THEN
                  IQUIT=1
                  PRINT '(A40,I4)','BEAM AREA IS ZERO FOR SEED PASS ',IP
                  WRITE(12,'(A40,I4)'),
   *              'BEAM AREA IS ZERO FOR SEED PASS ',IP
                  RETURN
            END IF
```

```
        END DO



C COPY INPUT DATA TO THE INPUT DUMMY ARGUMENTS
        N1U=N1X
        N2U=N2X
        N3U=N3X

        N1V=N1Y
        N2V=N2Y
        N3V=N3Y

        N1W=N1Z
        N2W=N2Z
        N3W=N3Z

        NBU=NBX
        NBV=NBY
        DW=DZ

        WRITE(13,'(A10,D10.3)'),'DW=',DW

        IF (DUMPOU.EQ.5) THEN
                DUMPPRN=2
        ELSE
                DUMPPRN=0
        END IF


C LOOP THRU ALL PASSES FOR SEED AND ASE.

        DO IP=1,NPASS


C               EVEN NUMBERED PASSES TRAVEL IN THE 'NEGATIVE W' DIRECTION

                IF (MOD(IP,2).EQ.0) THEN
                                WDIRN=-1
                ELSE
                                WDIRN=1
                END IF

                UPROPANGLE=THETAPROPX(IP)
                VPROPANGLE=THETAPROPY(IP)
                UDIVGANGLE=THETADIVGX(IP)
                VDIVGANGLE=THETADIVGY(IP)
                UCENTRE=XBEAMCENTRE(IP)
                VCENTRE=YBEAMCENTRE(IP)
                UWIDTH=XBEAMWIDTH(IP)
                VWIDTH=YBEAMWIDTH(IP)

         WRITE(13,'(A10,D10.3)'),'VWIDTH=',VWIDTH
```

```
                PASSNUMBER=IP
                PASSTYPE='SEED'

C CALL GEOCRT02

                CALL GEOCRT02(PASSNUMBER,PASSTYPE,
   *            NBU,NBV,WDIRN,DUMPPRN
   *     ,DW,UPROPANGLE,VPROPANGLE,UDIVGANGLE,VDIVGANGLE,UCENTRE
   *     ,VCENTRE,UWIDTH,VWIDTH,N1U,N2U,N3U,N1V,N2V,N3V,N1W,N2W,N3W)

C NOTE: ASSIGNMENT OF THE O/P DATA IS NOW PERFORMED WITHIN GEOCRT02
C THE FOLLOWING CODE PRINTS OUT DATA IF DUMPOU.EQ.5

                IF (DUMPOU.EQ.5) THEN
                        DO IZG=1,N2Z

                                DO IXG=1,N2X
                                        WRITE(13,'(A7,I3,A7,I3,A10,I4,A10,I4)'),
   *                            'IZG=',IZG,'IXG=',IXG,'IBX1S=',IBX1S(IP,IXG,IZG),
   *          'IBX2S=',IBX2S(IP,IXG,IZG)
                                END DO

                                DO IYG=1,N2Y
                                        WRITE(13,'(A7,I3,A7,I3,A10,I4,A10,I4)'),
   *                            'IZG=',IZG,'IYG=',IYG,'IBY1S=',IBY1S(IP,IYG,IZG),
   *          'IBY2S=',IBY2S(IP,IYG,IZG)
                                END DO

                                DO IBX=1,NBX
                                        DO IBY=1,NBY
                                                DO IXG=1,N2X
                                                        DO IYG=1,N2Y


        TEMPVOL=FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)

                                        IF ( TEMPVOL.NE.(0.D0) ) THEN
                                                WRITE(13,'(5(A7,I3),A10,D10.3)'),
   *                                            'IZG=',IZG,'IBX=',IBX,'IBY=',IBY,
   *                                            'IXG=',IXG,'IYG=',IYG,
   *                                            'FVOLZ=',
   *                                            FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)
                                        END IF
                                                        END DO
                                                END DO
                                        END DO
                                END DO

                        END DO

 C      =UBEAMANGLE(0:NBMAX)
 C      =VBEAMANGLE(0:NBMAX)
                        DO IZ=0,NZ

                                DO IBX=0,NBX
```

```
                                        WRITE(13,'(A7,I3,A7,I3,A10,D10.3)'),
   *                                    'IZ=',IZ,'IBX=',IBX,'XVS=',XVS(IP,IBX,IZ)
                                END DO

                                DO IBY=0,NBY
                                        WRITE(13,'(A7,I3,A7,I3,A10,D10.3)'),
   *                                    'IZ=',IZ,'IBY=',IBY,'YVS=',YVS(IP,IBY,IZ)
                                END DO

                        END DO

                END IF

C END OF 'PASS LOOP'

        END DO

C CALCULATE PEAK INTENSITY FOR SEED ---------------------------------

        IF (RSEEDSW.EQ.0) THEN

C               READ DATA FOR RADIAL PROFILE OF SEED BEAM FROM DATA FILE



                RFUDGE=0.D0

                DO IBX=1,NBX
                        DO IBY=1,NBY
                                READ(51,*) DATA51
                                SEEDRDATA(IBX,IBY)=DATA51
                        END DO
                END DO


                DO IBX=2,NBX
                        DO IBY=2,NBY
                                RFUDGE=RFUDGE + ( SEEDDBA(INJPT)*
   *                            ( SEEDRDATA(IBX,IBY)+SEEDRDATA(IBX,IBY-1)+
   *                              SEEDRDATA(IBX-1,IBY)+SEEDRDATA(IBX-1,IBY-1) )
   *                            *0.25D0 )

                        END DO
                END DO


        ELSE IF (RSEEDSW.EQ.2) THEN

C               RADIAL(SPATIAL) PROFILE IS UNIFORM

                RFUDGE=XBEAMWIDTH(INJPT)*YBEAMWIDTH(INJPT)

        ELSE

C               NON-PERMISSIBLE VALUE FOR RSEEDSW
```

```
          PRINT '(A35)','NON-PERMISSIBLE VALUE FOR RSEEDSW'
          WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR RSEEDSW'
          IQUIT=1
          RETURN

      END IF


      IF (TSEEDSW.EQ.0) THEN

C         READ DATA FOR TEMPORAL PROFILE OF SEED BEAM FROM DATA FILE

          TFUDGE=0.D0

          DO IDATA=1,NPT50

              READ(50,*),DATA50
              TFUDGE=TFUDGE+(DATA50*DT50)

          END DO


      ELSE IF (TSEEDSW.EQ.1) THEN

C         TEMPORAL PROFILE IS GAUSSIAN

          TFUDGE=(DSQRT(PI) / (2.D0*DSQRT(DLOG(2.D0))) )*SEEDDURN


      ELSE IF (TSEEDSW.EQ.2) THEN

C         TEMPORAL PROFILE IS UNIFORM

          TFUDGE=1.D0*SEEDDURN


      ELSE

C         NON-PERMISSIBLE VALUE FOR TSEEDSW
          PRINT '(A35)','NON-PERMISSIBLE VALUE FOR TSEEDSW'
          WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR TSEEDSW'
          IQUIT=1
          RETURN

      END IF

      SEEDPKINTENS=SEEDENERGY/(RFUDGE*TFUDGE)

      WRITE(12,'(A30,EN23.10,A9)'),
     *    'SEED PEAK INTENSITY=',SEEDPKINTENS,' MW/cm^2'

C CALCULATE PEAK INTENSITY FOR ASE ---------------------------------
C NOTE: THIS IS ONLY NEEDED IF ONE IS SPECIFYING 'INPUT ASE' TO
C AN AMPLIFIER STAGE.
```

```
        IF (RASESW.EQ.0) THEN

C               READ DATA FOR RADIAL PROFILE OF ASE BEAM FROM DATA FILE

                RFUDGE=0.D0

                DO IBX=1,NBX
                        DO IBY=1,NBY
                                READ(53,*) DATA53
                                ASERDATA(IBX,IBY)=DATA53
                        END DO
                END DO


                DO IBX=2,NBX
                        DO IBY=2,NBY
                                RFUDGE=RFUDGE + ( SEEDDBA(INJPT)*
     *                          ( ASERDATA(IBX,IBY)+ASERDATA(IBX,IBY-1)+
     *                            ASERDATA(IBX-1,IBY)+ASERDATA(IBX-1,IBY-1) )
     *                          *0.25D0 )

                        END DO
                END DO


        ELSE IF (RASESW.EQ.2) THEN

C               RADIAL(SPATIAL) PROFILE IS UNIFORM

C MAKE SURE THAT THIS WORKS.
C               RFUDGE=XBEAMWIDTH(INJPT)*YBEAMWIDTH(INJPT)

        ELSE

C               NON-PERMISSIBLE VALUE FOR RASESW
                PRINT '(A35)','NON-PERMISSIBLE VALUE FOR RASESW'
                WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR RASESW'
                IQUIT=1
                RETURN

        END IF


        IF (TASESW.EQ.0) THEN

C               READ DATA FOR TEMPORAL PROFILE OF ASE BEAM FROM DATA FILE

                TFUDGE=0.D0

                DO IDATA=1,NPT52

                        READ(52,*),DATA52
```

```
                        TFUDGE=TFUDGE+(DATA52*DT52)

                  END DO


            ELSE IF (TASESW.EQ.1) THEN

C                 TEMPORAL PROFILE IS GAUSSIAN

                  TFUDGE=(DSQRT(PI) / (2.D0*DSQRT(DLOG(2.D0))) )*ASEDURN


            ELSE IF (TASESW.EQ.2) THEN

C                 TEMPORAL PROFILE IS UNIFORM

                  TFUDGE=1.D0*ASEDURN


            ELSE

C                 NON-PERMISSIBLE VALUE FOR TASESW
                  PRINT '(A35)','NON-PERMISSIBLE VALUE FOR TASESW'
                  WRITE(13,'(A35)'),'NON-PERMISSIBLE VALUE FOR TASESW'
                  IQUIT=1
                  RETURN

            END IF


            ASEPKINTENS=ASEENERGY/(RFUDGE*TFUDGE)

            WRITE(12,'(A30,EN23.10,A9)'),
     *            'I/P ASE PEAK INTENSITY=',ASEPKINTENS,' MW/cm^2'

            RETURN
            END
```

……

```
      SUBROUTINE SPOCRT01

      INCLUDE 'Lascrt01.inc'

C LOCAL DECLARATIONS -------------------------------------------

      DOUBLE PRECISION ESPONT

C LOOP THROUGH ALL GAIN ZONES ---------------------------------

      DO IZ=N1Z+1,N1Z+N2Z

            IZG=IZ-N1Z

            DO IX=N1X+1,N1X+N2X
                  IXG=IX-N1X
                  DO IY=N1Y+1,N1Y+N2Y
                        IYG=IY-N1Y


C SPONTANEOUS EMISION: SAME FOR 3 OR 4 LEVEL SYSTEMS.


                        IF ( GPOP(IXG,IYG,IZG) .GT. (0.D0) ) THEN

                              IF (DUMPOU.EQ.9) THEN

                                    WRITE(13,'(A9,I3,A5,I3,A5,I3,A5,I3)'),
   *                                '** IXG=',IXG,'IYG=',IYG,
   *                                'IZG=',IZG,'JT=',JT
                                    WRITE(13,'(A10,EN23.10)'),
   *                                            'GPOP=',GPOP(IXG,IYG,IZG)

                              END IF

C DEPLETE POPULATION INVERSION DUE TO SPONTANEOUS DECAY --------------

                              GPOP(IXG,IYG,IZG)=( GPOP(IXG,IYG,IZG) )*
   *                          (1-(DT/TGLIFE))
                              APOP(IXG,IYG,IZG)=DENSNUM-GPOP(IXG,IYG,IZG)

C GENERATE NEW SPONTANEOUS EMISSION AND ADD IT TO THE EXISTING ASE ----

                              DO IP=1,NPASS
                               IF ( (IBX1S(IP,IXG,IZG).GT.0).AND.
   *                            (IBY1S(IP,IYG,IZG).GT.0) ) THEN

                                    DO IBX=IBX1S(IP,IXG,IZG),IBX2S(IP,IXG,IZG)
                                     DO IBY=IBY1S(IP,IYG,IZG),IBY2S(IP,IYG,IZG)


            EASE(IP,IBX,IBY,IZ)=EASE(IP,IBX,IBY,IZ) +
   *        (     GPOP(IXG,IYG,IZG)*(DT/TGLIFE)*EPHOTONSEED
   *                  *FVOLZ(IP,IBX,IBY,IXG,IYG,IZG)
   *                  *SBEAMVOL(IP,IBX,IBY,IZG)*
   *                  (ASESR(IP)/(4.D0*PI))   )
```

```
                             IF (DUMPOU.EQ.9) THEN
                                     WRITE(13,'(A7,I3,A5,I3,A5,I3,A5,I3)'),
   *                                 '** IP=',IP,'IBX=',IBX,
   *                                 'IBY=',IBY,'IZ=',IZ
                                     WRITE(13,'(A7,EN23.10)'),'EASE=',EASE
                                     WRITE(13,'(A10,EN23.10)'),
   *                                 'GPOP()=',GPOP(IXG,IYG,IZG)
                             END IF

                         END DO
                      END DO

                INFO
               END DO
       END IF

   END DO
  END DO
 END DO

RETURN
END
```

……

```
C LASCRT01.INC
C COMMON DECLARATION FILE FOR LASCRT01 PROJECT


C PARAMETERS (ARRAY MAX VALUES,CONSTANTS, ETC.) ---------------------

C UNITS:
C       VLIGHT HAS UNITS [cm/s]
C       H HAS UNITS [J*s]
C       IN ALL OTHER CASES, THE TIME UNIT IS [ps]
C                                       THE SPATIAL UNIT IS [cm]
C                                       THE ENERGY UNIT IS [uJ]
C

        INTEGER NBMAX,NXMAX,NYMAX,NZMAX,NPMAX,N2XMAX,N2YMAX,N2ZMAX
        DOUBLE PRECISION PI,VLIGHT,H
   PARAMETER (NBMAX=2,NXMAX=7,NYMAX=7,NZMAX=350,NPMAX=15)

        PARAMETER (N2XMAX=7,N2YMAX=7,N2ZMAX=25)

        PARAMETER (PI=3.1415926D0,H=6.6262D-34,VLIGHT=2.9979458D10)

C COUNTER VARIABLES ----------------------------------------------

        INTEGER JT
        INTEGER IP,JP,KP,IFNUM
        INTEGER IX,IY,IZ,JX,JY,JZ,KX,KY,KZ
        INTEGER IBX,IBY,JBX,JBZ,KBY,KBZ
        INTEGER IXG,IYG,IZG,JXG,JYG,JZG,KXG,KYG,KZG



C OTHER GENERAL VARIABLES -----------------------------------------

        INTEGER SYSTEMLEVELS
        INTEGER IQUIT,NPPARITY,NPFINALEVEN,NPFINALODD
        INTEGER NSTEPS,DUMPOU
        INTEGER N1X,N2X,N3X,N1Y,N2Y,N3Y,N1Z,N2Z,N3Z
        INTEGER NX,NY,NZ
        INTEGER FOO
        INTEGER JT1,JT2,JT3
        INTEGER IXGMID,IYGMID,IZGMID
        INTEGER ESATFUDGE

        DOUBLE PRECISION TIME

        DOUBLE PRECISION DX,DY,DZ,DT,DV,DA
        DOUBLE PRECISION DENSNUM,SIGG,SIGABS,TGLIFE,ALPHAG
        DOUBLE PRECISION REFRINDEX,DWAIR

        INTEGER SLABOUTSW,POPXYSW,POPYZSW,POPZXSW
        INTEGER LASSPATSW,PUMPCALCSW,SEEDCALCSW,GEOMCALCSW

        CHARACTER*70 COMMENT1,COMMENT2,COMMENT3
```

```
C SEED VARIABLES -----------------------------------------------

      INTEGER NBX,NBY,NPASS

      INTEGER JSEEDON,JSEEDOFF

      INTEGER IBX1S(NPMAX,N2XMAX,N2ZMAX)
      INTEGER IBX2S(NPMAX,N2XMAX,N2ZMAX)
      INTEGER IBY1S(NPMAX,N2YMAX,N2ZMAX)
      INTEGER IBY2S(NPMAX,N2YMAX,N2ZMAX)

      INTEGER IPSOURCE(NPMAX)
      DOUBLE PRECISION MIRREFL(NPMAX)

      DOUBLE PRECISION GPOP(N2XMAX,N2YMAX,N2ZMAX),SATFLXSEED,ESATSEED

      DOUBLE PRECISION THETAPROPX(NPMAX),THETAPROPY(NPMAX)
      DOUBLE PRECISION THETADIVGX(NPMAX),THETADIVGY(NPMAX)
      DOUBLE PRECISION XBEAMCENTRE(NPMAX),XBEAMWIDTH(NPMAX)
      DOUBLE PRECISION THETABEAMX(NPMAX,0:NBMAX)
      DOUBLE PRECISION YBEAMCENTRE(NPMAX),YBEAMWIDTH(NPMAX)
      DOUBLE PRECISION THETABEAMY(NPMAX,0:NBMAX)


      DOUBLE PRECISION XVS(NPMAX,0:NBMAX,0:NZMAX)
      DOUBLE PRECISION YVS(NPMAX,0:NBMAX,0:NZMAX)

      DOUBLE PRECISION FVOLZ(NPMAX,NBMAX,NBMAX,N2XMAX,N2YMAX,N2ZMAX)


      DOUBLE PRECISION ESEEDINP(NPMAX,NBMAX,NBMAX)
      DOUBLE PRECISION ESEED(NPMAX,NBMAX,NBMAX,NZMAX)
      DOUBLE PRECISION ESEEDSLAB(NPMAX),ESEEDTOTALOUT(NPMAX)

      INTEGER RSEEDSW,TSEEDSW,INJPT,IZSTARTINJPT
      DOUBLE PRECISION SEEDENERGY,SEEDDURN,TSEED,SEEDRADE,LAMBDASEED
      DOUBLE PRECISION SEEDPER,SEEDPKINTENS

      CHARACTER*13 FILE50,FILE51
      INTEGER NPT50,NPT51
      DOUBLE PRECISION DATA50,DATA51,DT50

      DOUBLE PRECISION SEEDDBA(NPMAX),SEEDRDATA(NBMAX,NBMAX)
      DOUBLE PRECISION SEEDINTENS,ESEEDTOTALIN(NPMAX),KSEED

      DOUBLE PRECISION SBEAMVOL(NPMAX,NBMAX,NBMAX,N2ZMAX)
      DOUBLE PRECISION EPHOTONSEED


C ASE VARIABLES -----------------------------------------------

      INTEGER RASESW,TASESW
      INTEGER JASEON,JASEOFF

      DOUBLE PRECISION ASEENERGY,ASEDURN,TASE,ASERADE
```

```
      DOUBLE PRECISION ASESR(NPMAX),ASEREFL(NPMAX)
      DOUBLE PRECISION EASEINP(NPMAX,NBMAX,NBMAX)
      DOUBLE PRECISION EASE(NPMAX,NBMAX,NBMAX,NZMAX)
      DOUBLE PRECISION EASESLAB(NPMAX),EASETOTALOUT(NPMAX)
      DOUBLE PRECISION ASEPKINTENS

      CHARACTER*13 FILE52,FILE53
      INTEGER NPT52,NPT53
      DOUBLE PRECISION DATA52,DATA53,DT52

      DOUBLE PRECISION ASERDATA(NBMAX,NBMAX),KASE(NPMAX)
      DOUBLE PRECISION ASEINTENS,EASETOTALIN(NPMAX)


C PUMP VARIABLES --------------------------------------------------

      INTEGER NPX,NPY,MPX,MPZ,LPY,LPZ
      INTEGER JPUMPON,JPUMPOFF

      DOUBLE PRECISION RP12,RP34,RP56

      INTEGER IBX1PZ(1:2,N2XMAX,N2ZMAX)
      INTEGER IBX2PZ(1:2,N2XMAX,N2ZMAX)
      INTEGER IBY1PZ(1:2,N2YMAX,N2ZMAX)
      INTEGER IBY2PZ(1:2,N2YMAX,N2ZMAX)

      INTEGER JBX1PY(3:4,N2XMAX,N2YMAX)
      INTEGER JBX2PY(3:4,N2XMAX,N2YMAX)
      INTEGER JBZ1PY(3:4,N2ZMAX,N2YMAX)
      INTEGER JBZ2PY(3:4,N2ZMAX,N2YMAX)

      INTEGER KBY1PX(5:6,N2YMAX,N2XMAX)
      INTEGER KBY2PX(5:6,N2YMAX,N2XMAX)
      INTEGER KBZ1PX(5:6,N2ZMAX,N2XMAX)
      INTEGER KBZ2PX(5:6,N2ZMAX,N2XMAX)


      DOUBLE PRECISION YTHPRPGPX(5:6),ZTHPRPGPX(5:6)
      DOUBLE PRECISION YTHDIVGPX(5:6),ZTHDIVGPX(5:6)

      DOUBLE PRECISION XTHPRPGPY(3:4),ZTHPRPGPY(3:4)
      DOUBLE PRECISION XTHDIVGPY(3:4),ZTHDIVGPY(3:4)

      DOUBLE PRECISION YTHPRPGPZ(1:2),XTHPRPGPZ(1:2)
      DOUBLE PRECISION YTHDIVGPZ(1:2),XTHDIVGPZ(1:2)


      DOUBLE PRECISION YTHBEAMPX(5:6,0:NBMAX)
      DOUBLE PRECISION ZTHBEAMPX(5:6,0:NBMAX)
      DOUBLE PRECISION ZTHBEAMPY(3:4,0:NBMAX)
      DOUBLE PRECISION XTHBEAMPY(3:4,0:NBMAX)
      DOUBLE PRECISION XTHBEAMPZ(1:2,0:NBMAX)
      DOUBLE PRECISION YTHBEAMPZ(1:2,0:NBMAX)


      DOUBLE PRECISION XBEAMWIDTHPZ(1:2),YBEAMWIDTHPZ(1:2)
```

```
      DOUBLE PRECISION XBEAMWIDTHPY(3:4),ZBEAMWIDTHPY(3:4)
      DOUBLE PRECISION YBEAMWIDTHPX(5:6),ZBEAMWIDTHPX(5:6)

      DOUBLE PRECISION XBEAMCENTREPZ(1:2),YBEAMCENTREPZ(1:2)
      DOUBLE PRECISION XBEAMCENTREPY(3:4),ZBEAMCENTREPY(3:4)
      DOUBLE PRECISION YBEAMCENTREPX(5:6),ZBEAMCENTREPX(5:6)


      DOUBLE PRECISION XVPZ(1:2,0:NBMAX,0:NZMAX)
      DOUBLE PRECISION YVPZ(1:2,0:NBMAX,0:NZMAX)
      DOUBLE PRECISION XVPY(3:4,0:NBMAX,0:NYMAX)
      DOUBLE PRECISION ZVPY(3:4,0:NBMAX,0:NYMAX)
      DOUBLE PRECISION YVPX(5:6,0:NBMAX,0:NXMAX)
      DOUBLE PRECISION ZVPX(5:6,0:NBMAX,0:NXMAX)


      DOUBLE PRECISION FVOLPX(5:6,NBMAX,NBMAX,N2XMAX,N2YMAX,N2ZMAX)
      DOUBLE PRECISION FVOLPY(3:4,NBMAX,NBMAX,N2XMAX,N2YMAX,N2ZMAX)
      DOUBLE PRECISION FVOLPZ(1:2,NBMAX,NBMAX,N2XMAX,N2YMAX,N2ZMAX)


      DOUBLE PRECISION EPUMPZ(1:2,NBMAX,NBMAX,NZMAX)
      DOUBLE PRECISION EPUMPINPZ(1:2,NBMAX,NBMAX)
      DOUBLE PRECISION EPUMPY(3:4,NBMAX,NBMAX,NZMAX)
      DOUBLE PRECISION EPUMPINPY(3:4,NBMAX,NBMAX)
      DOUBLE PRECISION EPUMPX(5:6,NBMAX,NBMAX,NZMAX)
      DOUBLE PRECISION EPUMPINPX(5:6,NBMAX,NBMAX)


      DOUBLE PRECISION EPUMPSLAB(NPMAX),EPUMPTOTALOUT(NPMAX)

      DOUBLE PRECISION APOP(N2XMAX,N2YMAX,N2ZMAX)
      DOUBLE PRECISION SATFLXPUMP,ESATPUMP,KPUMP

      INTEGER RPUMPSW,TPUMPSW
      DOUBLE PRECISION PUMPENERGY(6),PUMPRADE,LAMBDAPUMP
      DOUBLE PRECISION PUMPPER,PUMPDURN,TPUMP
      DOUBLE PRECISION PUMPPKINTENS3,PUMPPKINTENS4
      DOUBLE PRECISION PUMPPKINTENS1,PUMPPKINTENS2


      CHARACTER*13 FILE54,FILE55
      INTEGER NPT54,NPT55
      DOUBLE PRECISION DATA54,DATA55,DT54

      DOUBLE PRECISION PUMPDBA(NPMAX),PUMPRDATA(NBMAX,NBMAX)
      DOUBLE PRECISION PUMPINTENS3,PUMPINTENS4,EPUMPTOTALINPY(3:4)
      DOUBLE PRECISION EPHOTONPUMP

      DOUBLE PRECISION PUMPINTENS1,PUMPINTENS2,EPUMPTOTALINPZ(1:2)

C GLOBAL (COMMON) VARIABLES ----------------------------------------
C (THIS SHOULD INCLUDE ALL OF THE ABOVE DECLARED VARIABLES, EXCEPT FOR
C COUNTERS).
```

```
      COMMON /INTVARS/ NSTEPS,DUMPOU,N1X,N2X,N3X,N1Y,N2Y,N3Y,N1Z,N2Z,
     *    N3Z,NX,NY,NZ,IQUIT,SYSTEMLEVELS,JT,JT1,JT2,JT3,
     *    IXGMID,IYGMID,IZGMID,NPPARITY,NPFINALEVEN,NPFINALODD,
     *    IZSTARTINJPT,IPSOURCE

      COMMON /REFRACT/ REFRINDEX,DWAIR

      COMMON /DPETC/ TIME

      COMMON /GEO/ DX,DY,DZ,DT,DV,DA

      COMMON /XTALFOO/ SIGG,SIGABS,TGLIFE,DENSNUM,ASESR,MIRREFL,
     *    ASEREFL,GPOP,APOP,ALPHAG

      COMMON /FILEFOO/ SLABOUTSW,POPXYSW,POPYZSW,POPZXSW,LASSPATSW,
     *    PUMPCALCSW,SEEDCALCSW,GEOMCALCSW

      COMMON /SEEDINT/ IBX1S,IBX2S,IBY1S,IBY2S,
     *    RSEEDSW,TSEEDSW,NBX,NBY,NPASS,INJPT,NPT50,NPT51,
     *  JSEEDON,JSEEDOFF,ESATFUDGE

      COMMON /ASEINT/ RASESW,TASESW,NPT52,NPT53,JASEON,JASEOFF

      COMMON /SEEDGEOM/ THETAPROPX,THETAPROPY,THETADIVGX,THETADIVGY,
     *    XBEAMCENTRE,XBEAMWIDTH,THETABEAMX,YBEAMCENTRE,YBEAMWIDTH,
     *    THETABEAMY,XVS,YVS,FVOLZ,SEEDDBA,SEEDRDATA,SBEAMVOL

C MAKE SURE EVERYTHING IS HERE.

      COMMON /SEEDENRG/ ESEEDINP,ESEED,ESEEDSLAB,
     *    ESEEDTOTALOUT,TSEED,SEEDPKINTENS,SEEDENERGY,SEEDDURN,
     *  SEEDRADE,LAMBDASEED,SEEDPER,DATA50,DATA51,DT50,KSEED,
     *    SATFLXSEED,ESATSEED,SEEDINTENS,ESEEDTOTALIN,EPHOTONSEED



      COMMON /ASEENERG/ EASE,EASESLAB,EASETOTALOUT,ASEENERGY,ASEDURN,
     *    ASERADE,TASE,ASEPKINTENS,EASEINP,DATA52,DATA53,DT52,KASE,
     *    ASERDATA,ASEINTENS,EASETOTALIN


      COMMON /PUMPINT/ IBX1PZ,IBX2PZ,IBY1PZ,IBY2PZ,JBX1PY,JBX2PY,
     *    JBZ1PY,JBZ2PY,KBY1PX,KBY2PX,KBZ1PX,KBZ2PX,RPUMPSW,TPUMPSW,
     *    NPX,NPY,MPX,MPZ,LPY,LPZ,NPT54,NPT55,JPUMPON,JPUMPOFF

      COMMON /PUMPGEOM/ YTHPRPGPX,ZTHPRPGPX,YTHDIVGPX,
     *    ZTHDIVGPX,XTHPRPGPY,
     *    ZTHPRPGPY,XTHDIVGPY,ZTHDIVGPY,YTHPRPGPZ,XTHPRPGPZ,
     *  YTHDIVGPZ,XTHDIVGPZ,
     *    YTHBEAMPX,ZTHBEAMPX,ZTHBEAMPY,XTHBEAMPY,
     *    XTHBEAMPZ,YTHBEAMPZ,XBEAMWIDTHPZ,YBEAMWIDTHPZ,
     *    XBEAMWIDTHPY,ZBEAMWIDTHPY,YBEAMWIDTHPX,ZBEAMWIDTHPX,
     *    XBEAMCENTREPZ,YBEAMCENTREPZ,XBEAMCENTREPY,ZBEAMCENTREPY,
     *    YBEAMCENTREPX,ZBEAMCENTREPX,XVPZ,YVPZ,XVPY,ZVPY,YVPX,ZVPX,
     *    FVOLPX,FVOLPY,FVOLPZ,RP12,RP34,RP56,PUMPDBA,PUMPRDATA
```

```
      COMMON /PUMPENRG/ EPUMPZ,EPUMPINPZ,EPUMPSLAB,EPUMPTOTALOUT,
*     EPUMPY,EPUMPINPY,EPUMPX,EPUMPINPX,PUMPPKINTENS3,PUMPPKINTENS4,
*     PUMPENERGY,PUMPRADE,LAMBDAPUMP,PUMPPER,PUMPDURN,TPUMP,
*     DATA54,DATA55,DT54,KPUMP,SATFLXPUMP,ESATPUMP,
*     PUMPINTENS3,PUMPINTENS4,EPHOTONPUMP,
*     EPUMPTOTALINPY

 COMMON /PUMPADD/ PUMPPKINTENS1,PUMPPKINTENS2,PUMPINTENS1,
*     PUMPINTENS2,EPUMPTOTALINPZ

      COMMON /CHARVAR/ FILE50,FILE51,FILE52,FILE53,FILE54,FILE55,
*     COMMENT1,COMMENT2,COMMENT3

.................................................................................................................
```

References

[1] P. Rambaldi, M. Douard, M. Ulbricht and J-P. Wolf, *Proceedings of the SPIE* **2505**, 32 (1995)

[2] P. F. Moulton, *J. Opt. Soc. Am. B* **3**, 125 (1986)

[3] B. E. A. Saleh and M. C. Teich, Fundamentals of Photonics, 1991, John Wiley and Sons Inc.

[4] M. A. Dubinskii, V. V. Semashko, A. K. Naumov, R. Y. Abdulsabirov and S. L. Korableva, *OSA Proceedings on Advanced Solid-State Lasers* **15**, 195 (1993)

[5] C. D. Marshall, S. A. Payne, J. A. Speth, W. F. Krupke, G. J. Quarles, V. Castillo and B. H. T. Chai, *OSA Proceedings on Advanced Solid-State Lasers* **20**, 389 (1994)

[6] D. J. Ehrlich, P. F. Moulton and R. M. Osgoode, *Opt. Lett.* **4**, 184 (1979)

[7] N. Sarukura, M. A. Dubinskii, Z. Liu, V. V. Semashko A. K. Naumov, S. L. Korableva, R. Y. Abdulsabirov, K. Edamatsu, Y. Suzuki, T. Itoh and Y. Segawa, *IEEE J. Sel. Top. Q.E.* **1**, 792 (1995)

[8] P. Rambaldi, R. Moncorgé, J-P. Wolf, C. Pédrini and J. Y. Gesland, *Opt. Comm.* **146**, 163 (1998)

[9] N. M. Edelstein, ed. Lanthanide and Actinide Chemistry and Spectroscopy, 1980, American Chemical Society

[10] J. Andriessen, H. Merenga, C. M. Combes, P. Dorenbos and C. W. E. van Eijk, *Proc. Int. Conf. on Inorganic Scintillators and Their Applications*, **SCINT95**, 142 (1996)

[11] R. A. Apaev, M. V. Eremin, A. K. Naumov, V. V. Semashko, R. Y. Abdulsabirov and S. L. Korableva, *Opt and Spect.* **84**, 735 (1998)

[12] H. Merenga, "Electronic Structure Calculations on Cerium-containing crystals", Doctoral Thesis, Delft University of Technology, Delft, Netherlands, 1997

[13] M. A. Dubinskii, V. V. Semashko, A. K. Naumov, R. Y. Abdulsabirov and S. L. Korableva, *Laser Physics* **4**, 480 (1994)

[14] K. Lim and D. S. Hamilton, *J. Opt. Soc. Am. B.* **6**, 1401 (1989)

[15] S. M. Hooker, J. S. Cashmore and C. E. Webb, *IEEE LEOS Annual Meeting*, 295 (1994)

[16] N. Sarukura, Z. Liu, S. Izumida, M. Dubinskii, R. Y. Abdulsabirov and S. L. Korableva, *Appl. Opt.* **37**, 6446 (1998)

[17] A. J. S. McGonigle, D. W. Coutts and C. E. Webb, *OSA TOPS* **26**, 123 (1999)

[18] A. J. S. McGonigle, S. Girard, D. W. Coutts and R. Moncorgé, *Elect. Lett.* **35**, 1640 (1999)

[19] Z. Liu, K. Shimamura, K. Nakano, N. Mujilatu, T. Fukuda, T. Kozeki, H. Ohtake and N. Sarukura, *Jpn. J. Appl. Phys.* **39** L88-L89 (2000)

[20] J. Diels and W. Rudolph, Ultrashort Laser Pulse Phenomenon, 1996, Academic Press

[21] L. M. Franz and J. S. Nodvik, *J. Appl. Phys.* **34**, 2346 (1963)

[22] M. Buhr and R. Fedosejevs, *Proceedings of the 1999 IEEE Canadian Conference on Electrical and Computer Engineering*, pp. 611-616 (1999)

[23] Handbook of Mathematical, Scientific and Engineering Formulas, Tables, Functions, Graphs, Transforms, 1984, Research & Education Association

[24] J. N. Broughton, "KeV X-Rays From ps KrF Laser Plasmas", PhD Thesis, Department

of Electrical Engineering, University of Alberta, Edmonton, Alberta, 1992

[25] J. N. Broughton and R. Fedosejevs, *J. Appl. Phys.* **71**, 1153 (1992)

[26] C. Keller and H. Schmutz, *J. Inorg. Nucl. Chem.* **27**, 900 (1965)

[27] G. D. Brunton, H. Insley, T. N. McVay and R. E. Thoma, "Crystallographic Data for some Metal Fluorides, Chlorides and Oxides", Oak Ridge National Laboratory Report ORNL-3761, p. 52 (1965)

[28] M. J. Weber, <u>CRC Handbook of Laser Science and Technology</u>, CRC Press, 1995

[29] M. Born and E. Wolf, <u>Principles of Optics</u>, 1999, Cambridge University Press

[30] I. M. Ranieri, K. Shimamura, K. Nakano, T. Fujita, Z. Liu, N. Sarukura and T. Fukuda, *J. Crystal Growth* **217**, 151 (2000)

[31] C. R. Giles, "Stimulated Brillouin Scattering in a CO_2 Laser-Plasma Interaction", PhD Thesis, Department of Electrical Engineering, University of Alberta, Edmonton, Alberta, 1983

[32] R. N. Jones, R. Venkataraghavan and J. W. Hopkins, *Spectrochimica Acta* **23A**, 925 (1967)

[33] R. N. Jones, R. Venkataraghavan and J. W. Hopkins, *Spectrochimica Acta* **23A**, 941 (1967)

[34] Z. Liu, T. Kozeki, Y. Suzuki, N. Sarukura, K. Shimamura, T. Fukuda, M. Hirano and H. Hosono, *Opt. Lett.* **26**, 301 (2001)

[35] D. C. D. McKen, R. Fedosejevs, M. Arnfield, I. V. Tomov, C. Dormier and A. A. Offenberger, *Rev. Sci. Instrum.* **54**, 845 (1983)

[36] S. Wantabe, M. Wantabe and A. Endoh, *Appl. Phys. Lett.* **43**, 533 (1983)