UCRL-LR-144556
Thesis

Development of Components for Wavelength Division Multiplexing Over Parallel Optical Interconnects

R. R. Patel
Doctor of Philosophy

July 5, 2001

U.S. Department of Energy
Lawrence Livermore National Laboratory

Approved for public release; further dissemination unlimited

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This work was performed under the auspices of the U. S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-Eng-48.

This report has been reproduced directly from the best available copy.

Available electronically at http://www.doe.gov/bridge

Available for a processing fee to U.S. Department of Energy
and its contractors in paper from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-mail: reports@adonis.osti.gov

Available for the sale to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-mail: orders@ntis.fedworld.gov
Online ordering: http://www.ntis.gov/ordering.htm

OR

Lawrence Livermore National Laboratory
Technical Information Department's Digital Library
http://www.llnl.gov/tid/Library.html

Development of Components for Wavelength Division Multiplexing Over Parallel Optical Interconnects

BY

RAJESH RAMESHCHANDRA PATEL

B.S. (University of Connecticut) 1992
B.A. (University of Connecticut) 1992
M.E.S. (Rice University) 1995
M.S. (University of California Davis) 1997

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Engineering – Applied Science

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

_____

_____

_____

Committee in Charge

2001

Copyright © 2001 by Rajesh R. Patel

ACKNOWLEDGEMENTS

There are so many people who have helped me on the journey of learning and discovery that has led to this dissertation. Though I could not possibly mention everyone's name here I thank each and every one of them from the bottom of my heart.

My thesis advisor, Denise Krol, has always provided guidance and insight in matters both technical and otherwise. I have learned a lot about the process of scientific research from her. She has helped me in my research as well as in other matters such as choosing my next career move. Denise has always been fair and unbiased in her advice and having her as an advisor was one of the best things about my graduate experience.

I thank Professor Jonathan Heritage of the UC Davis Department of Electrical and Computer Engineering for his time and effort in serving on my dissertation committee.

I would like to thank all of the members (current and former) of the photonics group at LLNL. Without them there would not have been a project for me to work on. Specifically, Mike Pocha has served as my research advisor as well as a member of my dissertation committee. Mike's help and support over the past 3 years is greatly appreciated. Steven Bond has provided insight and support and was always willing to listen and help out. I thank Steven for all of his time and help during my work with the photonics group. I thank Glenn Meyer for all of his help in the clean room, and for all of the conversations we've had both technical and non-technical. Many of the experiments described in this dissertation would not have been possible without his help. Elaine Behymer has also provided help and support in the clean room, allowing me to get some of my experiments done in half the time it would have taken me to do them myself. I thank Mark Emanuel and Victor Sperry for the MOCVD growth of the filters used in my research. A special thanks to Victor for growing filters for me in his spare time even after he moved on to another position at the lab. I could not have finished my dissertation research without Victor's contributions.

I'd like to thank Phil Stefan for the countless occasions when he conjured up just the thing I needed in a piece of equipment or when I needed to figure out a way to do certain tasks related to the packaging and testing of my filters. I have learned a lot from Phil and I am truly grateful to him. I also learned a lot from Henry Garrett who taught me, among other things, the basic processes used to package our filters, how to run the dicing saw, and the basic GaAs processing steps used for substrate removal. I thank Mark Lowry for his help and support and for serving as my research advisor for almost a year after Bob Deri left LLNL.

I thank Bob Deri for taking me on as a graduate student. Bob taught me so many things about photonics, and in general about how to get things done in scientific research. He always made sure that I was growing as a scientist and he treated me as a member of the team rather than 'just a graduate student'

I wish to thank Professor Rao Vemuri for his help and support when I originally applied to the Department of Applied Science. Rao championed my case with the student policy committee regarding the award of a student employee appointment. Without his help I would not have been able to attend graduate school at DAS. I also wish to thank all of my professors at DAS for the time and effort they put into teaching their classes. Special thanks to Professor Ann Orel for teaching the first year mathematical methods

class. This course has been a pivotal part of my graduate work at DAS. Professor Fred Wooten's Solid State Physics class during my second year at DAS was also very enlightening for me. I thank Professor Rick Freeman for his rigor on the oral exam and for his help and encouragement in making sure I succeeded.

I would like to thank all of my fellow graduate students for the friendship and mutual support they have provided. A special thank you goes to John Adams. John has always been there with me throughout my graduate research. He was always quick to give encouragement and to give honest advice. I will miss our lunchtime conversations very much.

I thank Professor Philip Gould of the university of Connecticut Physics Department for giving me a chance to work in his lab under the National Science Foundation's Research Experience for Undergraduates program. This was my first real research opportunity and I learned a lot from working there. I also wish to thank all of my grade school teachers especially those who taught me mathematics and science. Their efforts helped form the foundation for my successful college and graduate career. I would especially like to thank Craig Robinson of the Central Connecticut State University's Planetarium. My internship at the planetarium during high school cemented my decision to pursue a career in science and taught me many skills that have helped me along the way.

I thank all of my friends for their support and encouragement. They have always believed in me and this has helped me believe in myself.

I thank my wife Mamata for her unwavering support and encouragement during my entire time at DAS. Words cannot express the countless ways she has enriched my life. I truly would not have been able to complete this work without her. I thank my son Keshav for his constant bright smiles, his unconditional love, and for letting dad go to work, even on weekends, to finish this Ph.D. I thank my family for always encouraging me to chase after my dreams and for always supporting all of my endeavors. I thank god for all of the talents and abilities given to me and for giving me the strength and perseverance to complete this work.

This work was performed under the auspices of the United States Department of Energy by the University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

CHAPTER 1 INTRODUCTION 1

1.1 MOTIVATION 1
1.2 TECHNICAL APPROACHES FOR WDM MPP INTERCONNECTS 7
1.3 λ-CONNECT SYSTEM DESCRIPTION 11
1.4 THE SCOPE OF THIS DISSERTATION 18
REFERENCES FOR CHAPTER 1 20

CHAPTER 2 FILTER DESIGN AND MODELING 23

2.1 INTRODUCTION 23
2.2 THE IDEAL WDM BANDPASS FILTER 23
2.3 FILTER REQUIREMENTS FOR λ-CONNECT 25
2.4 A SIMPLE MODEL FOR BANDPASS FILTERS 27
2.5 THIN FILM FILTER DESIGN 49
2.6 SUMMARY 59
REFERENCES FOR CHAPTER 2 61

CHAPTER 3 FILTER FABRICATION, PACKAGING AND CHARACTERIZATION 63

3.1 INTRODUCTION 63
3.2 FILTER FABRICATION 64
3.3 FILTER PACKAGING 68
3.4 FILTER MODULE PERFORMANCE 76
3.5 CONCLUSIONS 81
REFERENCES FOR CHAPTER 3 83

CHAPTER 4 LATERAL OXIDATION OF ALGAAS FOR WDM FILTER APPLICATIONS 84

4.1 INTRODUCTION 84
4.2 APPLICATIONS OF LATERAL OXIDATION TO FILTERS 86
4.3 LATERAL OXIDATION SETUP AND PROCESS 94
4.4 PASSBAND SHIFTING FILTER DESIGNS WITH THICK TUNING LAYERS 99
4.5 OXIDE ENHANCED REJECTION FILTER 106
4.6 EXPERIMENTAL RESULTS AND DISCUSSION 110
4.7 SUMMARY 121
REFERENCES FOR CHAPTER 4 123

CHAPTER 5 WAVELENGTH ROUTING – AN ALTERNATE ARCHITECTURE FOR λ–XONNEXT 125

5.1 INTRODUCTION 125
5.2 WAVELENGTH ROUTER – CONCEPT 127
5.3 DEVICE DESIGN 129
5.4 EXPERIMENTAL RESULTS 135
5.5 ROUTING WITH MULTIPLE ROUTERS 139
5.6 IMPLICATIONS OF WAVELENGTH ROUTING FOR λ–XONNEXT 143
5.7 SUMMARY 145
REFERENCES FOR CHAPTER 5 146

CHAPTER 6 SUMMARY 147

APPENDIX A MATLAB SCRIPTS FOR MC-FP FILTER SIMULATIONS 152

APPENDIX B THE TRANSFER MATRIX METHOD 161

APPENDIX C JAVA CODE LISTINGS FOR BRAGGMODELLER 171

APPENDIX D FULL LISTING OF FILTER LAYER STRUCTURES 241

Rajesh Rameshchandra Patel
September 2001
Engineering – Applied Science

Development of Components for Wavelength Division Multiplexing over Parallel Optical Interconnects

Abstract

Parallel optical interconnects based on multimode fiber ribbon cables are emerging as a robust, high-performance data link technology that enhances throughput by using parallel arrays of fibers. While this technology has primarily been implemented as single wavelength point-to-point links, it can be significantly enhanced by wavelength division multiplexing (WDM). WDM enables both increased point-to-point bandwidth as well as more complex interconnect topologies and routing approaches that are particularly attractive for massively parallel processing (MPP) systems. Exploiting the advantages of WDM interconnects requires multi-wavelength sources, a low loss routing fabric, and small footprint wavelength selective filter modules. The Lambda-connect project (λ-Connect) at Lawrence Livermore National Laboratory is a technology development and proof-of-principle demonstration of the enabling hardware for WDM parallel optical interconnects for use in massively parallel processing systems and other high-performance data link applications. This dissertation demonstrates several key system components and technologies for λ-Connect.

We demonstrate 2-port and 3-port byte-wide thin film filter modules based on III-V semiconductors and having robust, low-cost plastic packages with passive alignment, no micro-optics, and direct compatibility with ribbon cables terminated with Mechanically Transferable ferrules. The 2-port modules are suitable for broadcast and select architectures while the 3-port devices provide add/drop multiplexing. The design, modeling, fabrication, packaging and characterization of these filter modules is discussed herein. Notably, a multi-cavity Fabry-Perot filter with 5 coupled cavities was designed to meet the channel selection requirements for λ-Connect. Packaged devices exhibited fiber-to-fiber center wavelength uniformity better than $\pm$0.5nm, 1.6 $\pm$ 0.6 dB insertion loss, 7.5nm FWHM and could support up to 4 WDM channels at 10nm channel spacing with less than –23dB crosstalk from adjacent channels.

We also address two issues that can potentially enhance the performance and reduce the production cost of a λ-Connect system. First, we address the need to increase the size of the filter DBR stop-bands in order to increase the number of wavelength channels that can be implemented in λ-Connect. Second, we describe a method to shift the passband of an epitaxially grown filter with a post-growth process. Since WDM systems require a comb of passband filters matching the system wavelengths, this type of process has the potential to reduce the production cost of filters for λ-Connect. Lateral oxidation of buried AlGaAs layers is explored as a potential solution to both of these problems. A 3-cavity bandpass filter with aluminum oxide low-index layers and >500 nm DBR stop-bands was demonstrated. This could enable coarse-WDM systems with ~10nm channel spacings and more than 16 channels, something that is not possible with III-V semiconductor based filters alone. We have also shown that the passband of a multi-

cavity filter can be shifted with a post-growth lateral oxidation process. The theory for passband shifting filter design was developed and a 2-cavity filter with 54nm passband shift upon oxidation was demonstrated experimentally. The filters exhibited high insertion losses (6 to 9 dB after oxidation) which are largely attributed to process problems. If these process issues can be overcome, lateral oxidation shows great promise for WDM filter applications.

Finally, a bit-parallel wavelength router suitable for coarse WDM over multimode fiber was developed. Using a diffraction grating and lens combination along with 3-port add/drop filters, full wavelength re-use was achieved. A 3x3 wavelength router with 2 parallel bit-lines per port and 37nm channel spacing was demonstrated. This is, to our knowledge, the first demonstration of a bit-parallel wavelength router. Scaling of the device to accommodate 16 wavelengths and an arbitrary number of parallel bit-lines is addressed. This device provides a new tool for WDM over multimode fiber.

Approved: Denise Krol, Chairperson, Committee in Charge _____ Date

Chapter 1 Introduction

1.1 Motivation

Increasing the speed and capabilities of computer systems to meet emerging computational demands has naturally led to the idea of parallel processing. Rather than simply improving the speed of individual processors, parallel processing involves multiple processors working together on the same problem. One author describes the basic premise behind parallel processing as follows[1]:

> The hope is that if a single processor can generate X floating point operations per second (FLOPS), then ten of these may be able to produce 10X FLOPS, and, in the case of massively parallel processing, a thousand processors may be able to produce 1000X FLOPS.

Parallel processing finds applications in problems that involve a large number of computations. Matrix multiplication, data base searches, finite element analysis, weather prediction, signal processing, optimization, fluid dynamics, continuum mechanics, ocean-climate modeling, and high energy physics are just a few of the type of problems where parallel processing is being applied.

Massively parallel processing (MPP) where thousands, $O(\sim 10^4)$, of processors are connected together to work on the same problem is considered vital to the national security of the United States. For example, in order to ensure the safety and integrity of the nation's aging nuclear weapons stockpile detailed computer simulations of weapons physics are being pursued. To meet the computational needs of these problems the accelerated strategic computing initiative (ASCI) within the United States Department of Energy calls for 100+ trillion floating point operations per second (TFLOPS) of sustained computing capability by 2004.[2] This level of computation requires MPP. However, even

if the economic challenge of obtaining and interconnecting a sufficient number of processors and memories is overcome, achieving 100 TFLOPS operation is a formidable task.

Massively parallel processing systems consist of a large number of processors (CPUs), memory elements, a network, called an interconnect, that connects the processors and memories to each other and other components such as fixed disk storage and front-end interfaces (workstations, monitors etc.). Simply adding processors to a MPP system does not necessarily improve its performance. Data and instructions must be provided in a timely manner in order for processors in the system to perform useful computations[3]. Delay, called latency, in providing required information reduces CPU efficiency and degrades overall system performance.

Although individual CPU performance has steadily increased following Moore's law[4,5], interconnects between processors have not improved as fast. This has resulted in a widening gap between the performance of individual processors and the (scaled) performance of MPP systems.

In MPP systems processor to processor (and processor to remote memory) communication bottlenecks limit the "scaling" of performance with the number of nodes (processors). Depending on the nature and size of the problem and the nature of the MPP system, there is some optimum number of processors that can work efficiently in parallel. After this point, adding more processors fails to give a proportional increase in performance, and in some cases, can actually degrade performance because processors stall waiting for data from remote parts of the system.

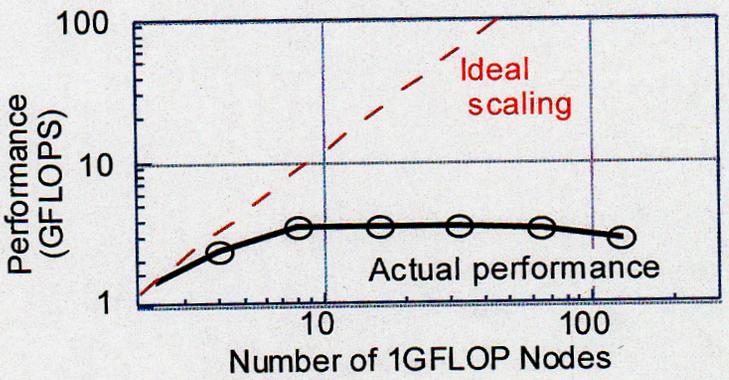


Figure 1. Performance vs. number of nodes (processors) for a 1024x1024 matrix-vector multiply on a bus-based MPP system using 1 GFLOPS processors and a bus bandwidth of 1 Gbyte/sec.

Figure 1 shows the degradation of MPP system performance for a 1024x1024 matrix-vector multiply algorithm executed on a (simulated) MPP system with 1 GFLOPS nodes and 1 Gbyte/sec interconnect bandwidth. The data for this figure were generated using a discrete event simulator that simulates the MPP system by tracking each instruction that would be executed by every processor in the system during the course of the 1024x1024 matrix-vector multiplication.[6] As indicated in the figure, the ideal scaling is a linear relationship where the full benefit of adding a node to the system is maintained regardless of the number of nodes in the system. Clearly the performance of the simulated system is far from ideal.

Improving MPP performance relies on careful programming at the application level, efficient use of memory resources, and effective communication amongst processors and between processors and memory. A full description of all of the factors that affect MPP system performance on specific problems is beyond the scope of the present discussion.

To set the stage for this dissertation work, **our primary focus is the communication bottleneck caused by the hardware and architecture of the MPP interconnect.**

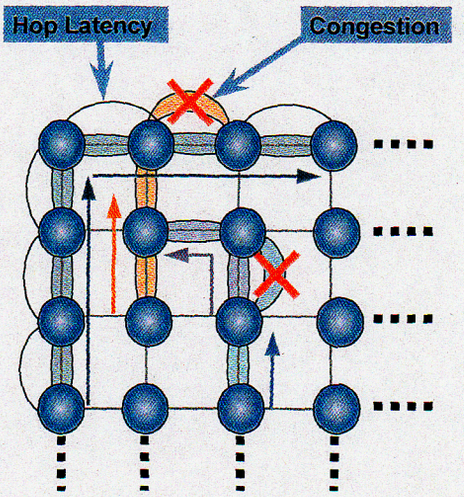


Figure 2. Conceptual illustration of a 2-dimensional MPP interconnect. The spheres represent computing nodes. The colored arrows represent the paths of messages between nodes in the system. Hop latency and network congestion, two factors that can degrade MPP system performance are indicated.

At the hardware level, latency in providing instructions or data from remote nodes is governed by the speed (data rate or throughput), size, and architecture of the interconnect. Figure 2 shows an example of a 2 dimensional MPP interconnect. Each processor has 4 nearest neighbors. Messages sent to nodes that are not nearest neighbors must hop from node to node until reaching their destination. Each hop incurs a time delay, called hop latency, because the message header must be read and a routing decision made. The cumulative effect of these hop latencies is one of the factors that degrade MPP system performance. Hop latency can be reduced by reducing the number of routing decisions made in sending a message from source to destination. In the ideal case, if each node were connected to all other nodes in the system, messages are fully routed at the source (source routing) because there is only one hop from source to destination. Another factor

that degrades system performance is network congestion. Two packets of data cannot be serviced at the same time by a given node. Therefore, if two messages arrive simultaneously at any node, at least one of them will be blocked until the node is free.

As the number of nearest neighbors for each node, called node fanout, increases, the number of possible connection paths within the system increases. On average, this tends to reduce the number of hops that a message needs to take to reach its destination. Increasing node fanout also tends to decrease overall network congestion. Figure 3 illustrates how 4096 nodes might be connected with a node fanout of 2, 4, or 6. The maximum number of hops between two nodes in the system, called the system diameter, is 2048, 64, and 24 for nodes with fanout of 2, 4, and 6 respectively. Clearly increasing node fanout will yield better overall system performance.

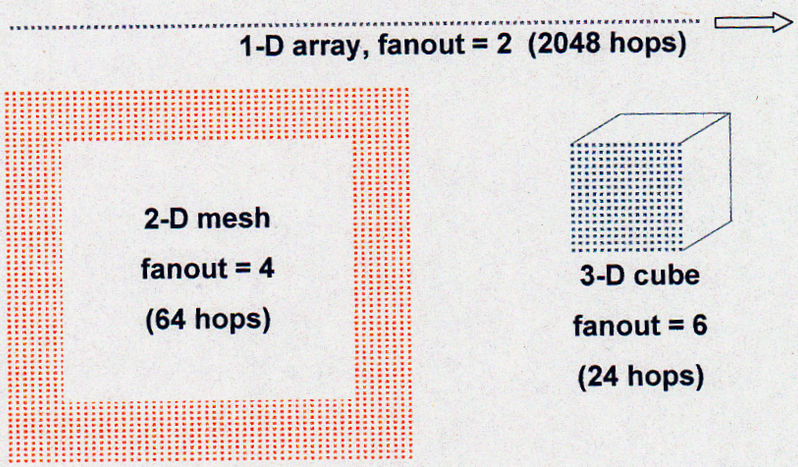


Figure 3. 4096 nodes interconnected with node fanout of 2, 4, and 6. The maximum number of hops in the system (system diameter) for each case is indicated. Increasing node fanout clearly increases the connectivity of the system as indicated by the dramatic reduction in system diameter.

The electronic interconnects that are most commonly employed in MPP systems today are quite limited in the fanout that can be achieved. Each node-to-node link is typically several bytes wide (8 to 16 bytes is not uncommon), each byte consists of 8 bits,

and each bit requires 2 wires for differential signal transmission, thus an 8-byte wide link requires 8x8x2 = 128 wires. Since each nearest neighbor link requires a separate set of wires the pinout and wire density requirements quickly outstrip the available space on the edge of the computer boards[7]. Furthermore electronic interconnects suffer from several limitations including capacitive loading, attenuation due to skin effect, and susceptibility to electro-magnetic interference.

The large available bandwidth, electromagnetic interference and ground loop insensitivity, and increased fanout and bandwidth-distance characteristics of optical fibers (compared to electronic interconnects) combined with the inherent concurrency allowed by wavelength division multiplexing (WDM) has led to interest in using WDM optical links for multi-processor interconnects[8,9,10]. WDM enables both increased point-to-point link bandwidths as well as more complex interconnect topologies and routing approaches that can benefit MPP systems.

DeGroot et. al.[11] used simulations to determine the number of system wavelengths required to impact MPP system performance. Specifically they investigated the case where the number of wavelengths, N, was much smaller than the number of nodes, M, in the system to determine the relative sizes (M and N) where WDM offered significant performance benefits for MPP systems. Instruction level simulations with a discrete event simulator indicated that for N=8 and M=256, an MPP system could deliver ~200x speedup compared to a single CPU on common scientific application kernels. Ideal scaling would be a speedup of 256x with M=256, nevertheless it is remarkable and quite promising that such high performance might be obtained with so few system wavelengths. In fact, only a slight increase in speedup was found by going from N=8 to

N=32 for the fixed problem sizes that could be simulated with their model. These results suggest that a WDM system with 8 to 16 wavelengths is of practical interest for MPP interconnect applications.

1.2 Technical Approaches for WDM MPP interconnects

One potential way to realize multi-wavelength optical interconnects for MPP is to use dense wavelength division multiplexing (DWDM) over high-speed serial single mode fiber (SMF) links[12,13]. This technology has been developed for use in the telecommunications industry and the required components are readily available from many suppliers. However, the cost-effectiveness of this technology for MPP interconnect applications is currently in question. The stringent alignment tolerances (0.2 - 2 μm) of SMF and narrow channel spacing (2 – 0.2 nm) of DWDM result in complex, high cost components. These components have been developed to operate at high speeds (2.5 – 10 GHZ) and to maximize bandwidth-distance performance for link lengths of 10's or 100's of km. However, in MPP applications the longest links are about 500m and key requirements are to maximize fanout (connectivity) and to minimize latency.[14] Furthermore, use of serial links between processors could potentially add latency as incoming parallel electrical signals at the system logic speed are serialized and multiplexed to a faster serial optical data-rate. At the receiving end de-serialization along with clock recovery is necessary to match the logic speed, again adding latency. This type of solution is unfavorable for MPP systems because the cost of the components cannot be amortized over a large number of users as in telecommunication systems.

Much research has also been done into using free-space optical interconnects for computing systems[15,16]. Some of this work also involves WDM[17]. These approaches

focus on short-distance links O(~10cm) and are suitable for communications between boards within a single computing rack. Free-space optical links are not suitable for the 1 to >100m links of interest for the rack to rack links in MPP systems such as ASCI.

The technical approach pursued at Lawrence Livermore National Laboratory (LLNL) is to use coarse wavelength division multiplexing (CWDM) over parallel, multimode fiber (MMF) optical links. The ensemble of parallel fibers in a fiber ribbon cable provides a word-wide communication channel for each wavelength, with each fiber in the cable carrying a separate bit of the message. Carrying multiple wavelengths over a single fiber cable assembly maximizes the total bandwidth per cable and allows the cable to carry multiple independent data channels at the same time. This maximizes the number of simultaneous communication channels in the system (system concurrency).

This approach has several attractive features for MPP interconnects. For example, the use of parallel fibers permits modest data rates per-fiber (1 – 2 Gb/s) resulting in improved reliability and reduced component cost and complexity. Using optical baud rates that match the system logic speed can potentially reduce or eliminate the need for clock recovery. Furthermore, using a sufficient number of parallel optical bit lines can eliminate the latency associated with serialization and de-serialization. CWDM with ~10nm (or greater) channel spacings can allow ~5nm tolerance windows for component wavelength variations (at 1dB excess loss) resulting in reduced component cost.

Parallel optical interconnects (POIs) based on multimode fiber (MMF) ribbon cables are emerging as a robust high-performance data link technology that uses linear arrays of MMF to increase link throughput. MMF ribbon cable is readily available from many vendors and offers inexpensive aggregate throughput using 2 to as many as 32 fibers in

parallel. A photograph of a typical MMF ribbon cable is provided later in Figure 5. *Single wavelength* parallel optical transceivers for use with MMF ribbon cable have recently become available from several vendors (W.L. Gore & Associates, Agilent Technologies, Siemens). These products provide (30Gb/s) interconnects priced at ~$30 per Gigabit/s of (point-to-point) bandwidth with small footprints and link lengths of ~300m or more. MPP applications such as the ASCI will most likely employ such single wavelength links to interconnect nodes that are physically separated by 50m – 100m. To meet its 100+ TFLOPs goal, it is anticipated that ASCI will require roughly 256 Gbyte/s interconnections for each of 128 processor/compute racks. The physical size of this system requires ~2Gb/s (per fiber) parallel optical links for the rack-to-rack interconnects because it is cost prohibitive to drive electronic interconnects at these speeds over the required distances.[18]

Leveraging the growing commercial single wavelength POI activity, CWDM over MMF ribbon cables can enable the enhanced connectivity of WDM within the cost-performance constraints of the MPP interconnect market. Robust, low-cost solutions that maximize the use of commercial-off-the-shelf technology are required to ensure a practical system design with the potential for commodity pricing and multiple suppliers for system components.

The Lambda-Connect (λ-Connect) project at Lawrence Livermore National Laboratory (LLNL) is a technology development and proof of principle demonstration of the enabling hardware for a multi-wavelength, multi-fiber (parallel) optical link for MPP interconnects and other short distance (≤100m) data communication applications. This approach involves innovation in both the interconnect hardware and architecture. At the

hardware level, the idea of using WDM over parallel multimode fiber cables is unique. At the architecture level, our approach lets each node communicate directly with a large number of other nodes within a single hop. Messages are source routed by selecting a transmission wavelength. Different wavelength streams from the same source and in the same cable are sent to different destinations through the use of optical color filters. This approach de-couples the node fanout from the physical cabling. By simply using more wavelengths over the same cables fanout can be increased without requiring more cable connections to the computer board. The system architecture is described in detail in section 1.3.1 below.

The key areas of research in the λ-Connect project are multi-wavelength bit-parallel optical transmitters, a parallel optical interconnect "fabric", wavelength-selective filter modules, and advanced interconnect architectures made possible by multiple system wavelengths. This dissertation specifically addresses the wavelength selective filter modules and advanced interconnect architecture aspects of λ-Connect.

1.3 λ-Connect System Description

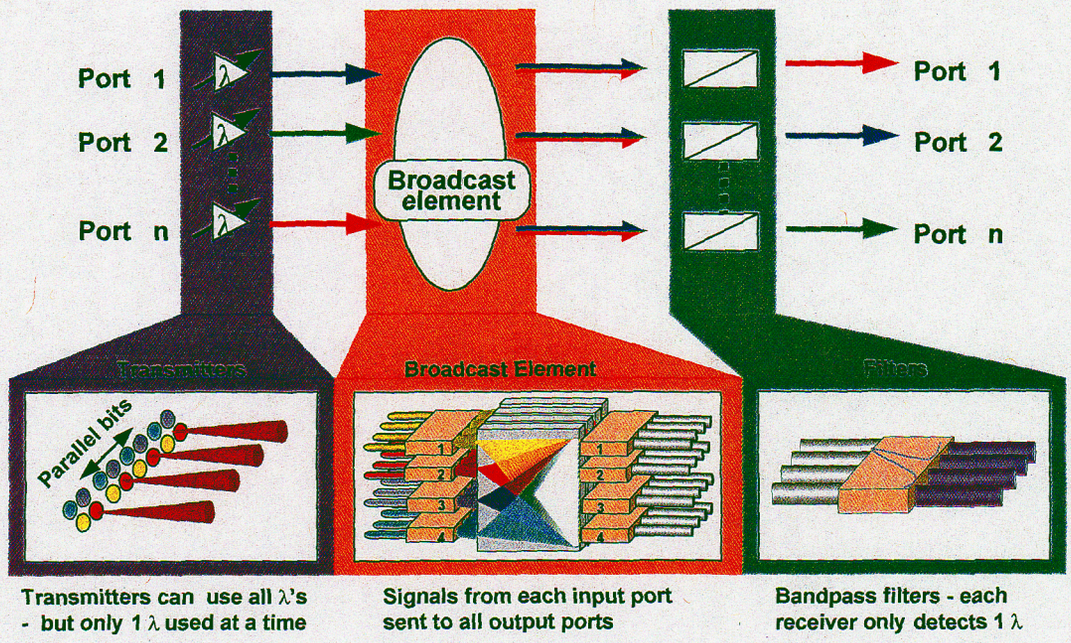


Figure 4. λ-Connect system architecture. Each port connects to a separate node in the MPP system. Each node has a multiple-wavelength bit-parallel transmitter. The broadcast element sends each incoming signal to all output ports. Fixed wavelength filters at the receivers select a single wavelength channel to pass on to the photo-detectors (not shown). The bottom of the figure shows conceptual illustrations of the bit-parallel system components.

1.3.1 System Architecture

λ-Connect employs a broadcast and select architecture with tunable transmitters and fixed receivers as shown in Figure 4. The top portion of this figure is a schematic of the λ-Connect architecture; conceptual illustrations of the bit-parallel system components are shown on the bottom. Each port in the figure is connected to a separate computing node in the MPP system. Each node has a multiple-wavelength, byte wide (parallel) transmitter that can transmit on any of the N system wavelengths. In general the transmitters only use a single wavelength at a given time since most communications in MPP systems are 1 to 1 rather than 1 to many. Independent information channels on

different wavelengths are simultaneously broadcast to many nodes through the output ports of the broadcast element. At the receivers, fixed wavelength filters are used to select a single information channel. Use of WDM creates multiple concurrent logical bus channels over a common physical medium. Messages are routed at the source simply by selecting the transmission wavelength providing high-throughput and low-latency because there is only a single hop between all nodes.

If the number of system wavelengths is equal to the number of nodes λ-Connect functions as a non-blocking (cross-bar) interconnect. That is, new connections between free nodes can be established without disturbing connections that are already in progress. Contention only arises when two nodes need to transmit to the same receiving node (output contention) [11]. In an actual system, a procedure that decides which transmitter can use a particular wavelength at any given time would resolve this type of contention. Procedures of this kind are called medium access control (MAC). Implementation of MAC was beyond the scope of λ-Connect, however, the performance of WDM optical interconnects with a broadcast-and-select architecture using a passive star coupler as the broadcast element has been modeled extensively[19,20,21,22,23]. It has been reported[11,20] that use of random access (slotted ALOHA), fixed allocation (time division multiple access), or "replicated arbitration" MAC schemes should provide acceptable performance.

To use λ-Connect for MPP applications we eventually envision a system with 8 or 16 wavelengths and 10nm channel spacing. A 5 nm tolerance window is provided around each channel to allow for thermal drift of components. The system would support byte-wide interconnection with at least 10 parallel bit lines. The system should support link lengths up to 100 meters with 1 mw of launched power and –25 dBm of received power

at the detectors. Cross talk from adjacent channels must be below –25 dB to ensure error free performance at 1.25 to 2.5 Gb/s data rates. Table 1 summarizes the desired λ-Connect system parameters. The focus of the project was to develop technologies and packaging techniques that can eventually be scaled to meet the desired system criteria.

Parameter	Value	Units	Notes
Wavelength Range	780 – 860[a], 970-1000[b]	nm	a) GaAs pin photo-diode range b) InGaAsP detector range
Channel spacing	10	nm	initially larger
Channel count	8 – 16	-	eventually desired
Tolerance window	5	nm	allows for thermal drift
Word-width	8 - 12	bits	determined by fiber count
Link length (max)	~100	m	
Launched power (avg.)	0	dBm	
Received power (min)	-25	dBm	based on receiver sensitivity
Cross-talk	<-25	dB	
Data rate	1.25 – 2.5	Gb/s	per fiber

Table 1. Desired System Parameters for λ-Connect.

1.3.2 System Components and Technology Choices

1.3.2.A Optical Backbone

The optical backbone of λ-Connect uses ribbon cables consisting of 12 MMF with 62.5μm diameter graded index (GRIN) cores terminated with mechanically transferable (MT) ferrules. The MT ferrule is a low-cost plastic connector that uses guide pins for alignment. This particular cable configuration offers excellent bandwidth per footprint and is employed by all of the single wavelength POI transceivers mentioned earlier. Furthermore, experimental versions of the MT ferrule with up to 60 MMFs in a 14mm^2 area have been reported[24] indicating that even more bandwidth per footprint may be

available in the near future. Figure 5 shows a photograph of a 12-fiber ribbon cable along with the MT ferrule. The inset in the figure shows the ferrule end-face with the 12 fibers illuminated by a white light source. For λ-Connect each fiber in the ribbon carries a separate bit-line.

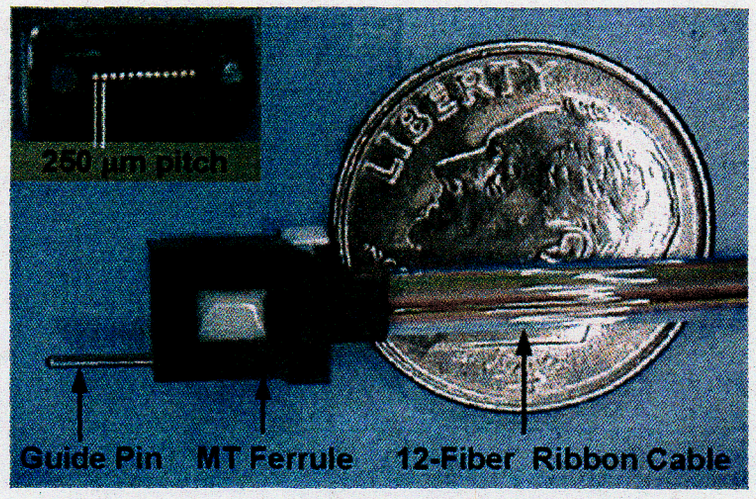


Figure 5. 12-Fiber ribbon cable. The fibers are 62.5 µm core-diameter MMFs. The molded plastic MT ferrule uses guide pins for fiber alignment. Inset shown the end-face of the ferrule with the 12 fibers illuminated by white light.

GRIN MMF provides ample bandwidth-distance product for MPP interconnect applications (~ 1 Ghz-km)[25] and has several advantages compared to SMF. The larger MMF core size requires ~10x looser alignment tolerances than for SMF resulting in reduced transceiver cost. Furthermore, the large fiber cores are less sensitive to shock, vibration, particulates, and thermal variations. Another advantage is that multiple VCSEL sources can be coupled into each MMF enabling multi-wavelength transmitters by packaging separate VCSEL die in close proximity to each other. This is illustrated in Figure 6.

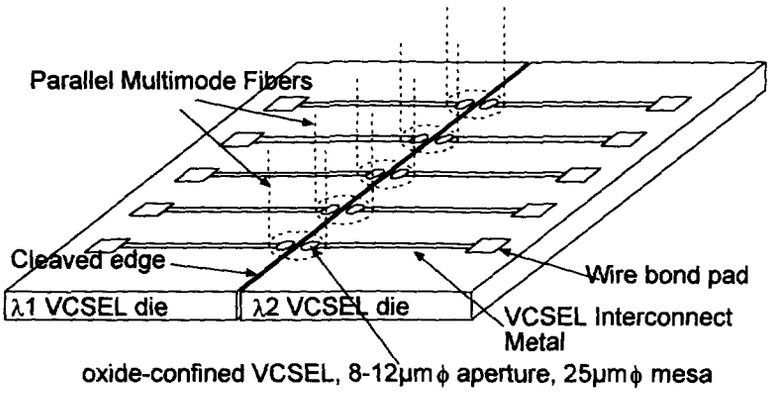


Figure 6. Multi-wavelength byte-wide transmitters based on direct fiber coupling. The large core size of MMF allows more than one VCSEL to couple into a single fiber by simply placing the VCSELs in close proximity to each other.

1.3.2.B Multi-Wavelength Optical Sources

The vertical cavity surface emitting laser (VCSEL) is a natural candidate for parallel optical sources because of the ease of fabrication of 1- and 2-dimensional VCSEL arrays. VCSEL based multi-wavelength optical sources for single-fiber (serial) links have previously been demonstrated[26,27]. Byte-wide multi-wavelength transmitters require either one array of tunable lasers or multiple arrays, one for each system wavelength, of fixed-wavelength lasers and appropriate multiplexing. Spectral channel spacings of order ~ 10nm (or greater) are desirable to allow for VCSEL drift with minimal thermal management (ideally no active cooling). Thus an 8-wavelength system requires an 80nm spectral range. Fast wavelength switching (1–10 ns) over the entire spectral range is crucial to minimize latency. Although some interesting and novel wavelength tunable devices based on electro-optic[28] or micro-mechanical tuning[29] have been reported, the tuning ranges (5-20 nm) and tuning times (μs to ms) are insufficient for MPP applications.

For the reasons discussed above, λ-Connect employs multiple arrays of fixed-wavelength VCSELs and electronic rather than optical wavelength switching. The challenge is to package the multi-wavelength sources in an integrated circuit or multi-chip-module form factor. Using a combination of direct fiber coupling (as in Figure 6) and broad band add/drop multiplexing we have developed a 4-wavelength, byte-wide (10 parallel bits actually), VCSEL-based transmitter in a standard pin grid array package with MT compatible optical output. Details of the transmitter have been reported elsewhere[30].

1.3.2.C Broadcast Element

The broadcast element is an M-input by M-output device that takes signals from each transmitter and distributes them to every receiver. For λ-Connect the broadcast element must preserve the parallelism of the ribbon cables. The functionality of the broadcast element could be achieved in several ways including fiber- or waveguide- based MxM couplers[31], diffractive optical fanout elements[32], or kaleidoscopic fanout devices[33]. Several off-the-shelf solutions are available for the broadcast element. For example, MMF star couplers for single fiber (serial) links are commercially available from many suppliers; the bit-parallel broadcast element could be implemented by using arrays of MxM star couplers with one coupler per bit-line. Bit-parallel arrays of MxM waveguide trees are also commercially available from a few vendors.

Because off-the-shelf solutions are available, the λ-Connect project did not focus on developing a broadcast element. The basic design philosophy is: "why build it if you can buy it?" The proof-of-principle nature of the project focuses on developing the other technologies necessary to complete the multi-wavelength parallel optical link.

1.3.2.D Wavelength Selective Receivers

Parallel receiver arrays consisting of detectors and associated de-modulation circuitry in small footprint packages are commercially available for the single wavelength POI links mentioned previously. The TT-FR architecture of λ-Connect takes advantage of these commercially available components by using MT-compatible WDM filter modules in front of the detectors to achieve wavelength selection.

Low insertion loss is critical for these filters because the transceivers exhibit link power budgets well below those of telecom WDM systems and the MMF cabling precludes the use of optical amplifiers. The high numerical aperture (NA) of MMF (NA=0.275) poses a challenge when trying to package filters for ribbon cables. Furthermore, the use of micro-optics is undesirable because it complicates packaging and would increase the cost of the filter modules. For these reasons we use high index III-V semiconductors (AlGaAs) for the thin film filter materials.

1.3.3 λ-Connect Project Accomplishments

The major accomplishments of the 4-year λ-Connect project include:

- development of a 4-wavelength, byte-wide transmitter packaged in an IC form factor;
- development of a post-growth process to produce monolithic multi-wavelength VCSEL arrays with up to 4 wavelengths from a single epitaxially grown wafer.
- development of byte-wide add/drop and bandpass filter modules packaged to be compatible with MMF ribbon cables;
- design and development of a post-growth process to shift the passband of epitaxially grown bandpass filters.
- development of a bit-parallel passive wavelength router for use with CWDM over MMF;
- and, demonstration of a point-to-point WDM link with bit-error-rate $<10^{-14}$.

To our knowledge, these are the first ever demonstrations of components with the above-mentioned functionality. The innovative technical approach of the λ-Connect project has led to the filing of six patents, two of which are a direct result of this dissertation work,

1.4 The Scope of this Dissertation

This dissertation addresses all aspects of thin film filter design, modeling, fabrication, packaging and characterization associated with the λ-Connect project as well as advanced filter and interconnect concepts (chapters 4 and 5) that lie beyond the original scope of the project. An overview of the chapters of this dissertation is presented below.

Chapter 2 Filter Design and Modeling

The general approach to filter design used for λ-Connect is described in chapter 2. The effect of changes in mirror reflectivities and cavity lengths on the transmission properties of coupled Fabry-Perot resonators are explored to gain insight for the design of thin film WDM filters. The filter design techniques and modeling tools used in this dissertation research as well as the choice of thin film materials are also discussed. As part of this dissertation a bandpass filter for λ-Connect was developed. The filter design is presented in this chapter.

Chapter 3 Filter Fabrication, Packaging and Characterization

Chapter 3 describes the filter fabrication and packaging approach developed for λ-Connect and presents results from packaged filter modules. Two-port passband filter modules suitable for 10nm channel spacing at –23dB crosstalk are described. Results are

also presented for 3-port filter modules for add/drop multiplexing applications such as the fiber-guided superstrate.

Chapter 4 Lateral Oxidation of AlGaAs for WDM Filter Applications

This chapter addresses two issues that can potentially enhance the performance and reduce the production cost of a λ-Connect system. First, we address the need to increase the size of the filter DBR stop-bands in order to increase the number of wavelength channels that can be implemented in λ-Connect. Second, we describe a method to shift the passband of an epitaxially grown filter with a post-growth process. Since WDM systems require a comb of passband filters matching the system wavelengths, this type of process has the potential to reduce the production cost of filters for λ-Connect. Lateral oxidation of buried AlGaAs layers is explored as a potential solution to both of these problems.

Chapter 5 Wavelength Routing – An Alternate Architecture for λ-Connect

In chapter 5 we examine an alternate architecture for λ-Connect based on wavelength routing. The concept of wavelength routing is described and experimental results for a bit-parallel wavelength router with 3 input and output ports are presented. Scaling of this device to higher numbers of input/output ports and more parallel bits is also addressed.

Chapter 6 Summary

Chapter 6 summarizes this dissertation work and points out future directions for the research described herein.

References for Chapter 1

[1] Hord, R. M., Understanding Parallel Supercomputing, IEEE Press, New York, © 1999, pg.1.

[2] Deri, R. J., Winkler, K.-H., Hale, A., "ASCI Path-Forward Project Description", http://www.llnl.gov/asci-pathforward.

[3] Hennessy, J. L., Patterson, D. A., Computer Architecture: A Quantitative Approach, Morgan Kaufman, San Mateo CA, ©1990.

[4] Moore, G., E., "Cramming more Components onto Integrated Circuits", Electronics, V38, 1965, pg. 114.

[5] Moore, G. E., "Progress in Digital Integrated Electronics", IEDM Tech. Dig., 1975, pg. 11-13.

[6] Deri, R. J., DeGroot, A. J., Haigh, R. E., "Feasibility of Optically Interconnected Parallel Processors using Wavelength Division Multiplexing", Lawrence Livermore National Laboratory, report number: UCRL-ID-123593, 1996.

[7] Daly, W. J., "Performance Analysis of k-ary n-cube Interconnection Networks", IEEE Trans. Comput. V 39, 1990, pg. 775.

[8] Haugen, P., Rychnovsku, S., Husain, A., Hutcheson, L., "Optical interconnects for High Speed Computing", Optical Eng., V 25, 1986, pg. 1076-1085.

[9] Dowd, P. W., Bogineni, K., Aly, K. A., Perreault, J. A., "Hierarchical Scalable Photonic Architectures for High-Performance Processor Interconnection", IEEE, Trans. Comput. V 42, 1993, pg. 1105-1120.

[10] Krchnavek, R. R., Chamberlain, R. D., Barry, T., Malhotra, V., Dittia, Z., "Optical interconnect Design for a Manufacturable Multicomputer", IEEE Proc. 2nd Intl. Conf. On Massiv. Parallel Proc. Using Optical Interconn., 1995, pg. 279-288.

[11] DeGroot, A. J., Deri, R. J., Haigh, R. E., Patterson, F. G., DiJaili, S. P., "High-Performance Parallel Processors Based on Star Coupled WDM Optical Interconnects", Proc. 3rd IEEE Conf. On Massively Parallel Processing using Optical Interconnects, 1996, pg. 62-69.

[12] Dowd, P. W., "Lightning: A Scalable Dynamically Reconfigurable Hierarchical WDM Network for High Performance clustering", Proc. Petaflops Frontier Workshop, NASA/CESDIS TR-95-161, 1995, pg. 69-72.

[13] Brackett, C. A., "Dense Wavelength Division Multiplexing Networks: Principles and Applications", IEEE J. Select Areas Commun., V8, 1990, pg. 948-964.

[14] Yuen, Al, "Strategies for Competitive Optical Interconnection Technologies", Panelist Position Statement for Panel Discussion, IEEE Proc. 5th Intl. Conf. on Massiv. Parallel Proc. Using Opt. Interconn., 1998, pg.62.

[15] Tsang, D. Z., Goblick, T. J., "Free-Space Optical Interconnection Technology In parallel Processing Systems", Opt. Eng., V35, 1994, pg. 1524-1531.

[16] Sakano, T., Matsumoto, T, Noguchi, K., "Three-Dimensional Board-to-Board Free-Space Optical Interconnects and Thier Application to the Prototype Multiprocessor System: COSINE III", Applied Optics, V 34, 1995, pg. 1815-1822.

[17] Kajita, M., Kasahara, K., Kim, T. J., Ogura, I., Redmond, I., Schenfeld, E., "Free-Space Wavelength division Multiplexing Optical interconnection for Massively Parallel Processing Systems", Proc. Opt. Compt., 1996, pg. 24-25.

[18] Benner, Alan F., "Design Options for Interconnecting a 100+ Tflop/sec Parallel Supercomputer in 2004", Panelist Position Stetement for Panel Discussion, IEEE Proc. 5th Intl. Conf. on Massiv. Parallel Proc. Using Opt. Interconn., 1998.

[19] Arthurs, E., Cooper, J. M., Goodman, M. S., Kobrinski, H., Tur, M., Vecchi, M. P., "Multiwavelength Optical Crossconnect for Parallel-Processing Computers", Electon. Lett. V 24, 1988, pg. 119-120.

[20] Dowd, P. W., "Random Access Protocols for High-Speed Interprocessor Communication based on an Optical Passive Star Topology", IEEE, J. Lightwave Tech. V LT-9, 1991, pg. 799-808.

[21] Chiarulli, D.M., Levitan, S.P., Melhem, R.P., Teza, J.P., Gravenstreter, G., "Multiprocessor interconnection networks using partitioned optical passive star (POPS) topologies and distributed control", Proc. IEEE 1st Intl. Conf. On Massively Parallel Processing Using Optical Interconnects, 1994, pg. 70–80.

[22] Desprez, F.; Ferreira, A.; Tourancheau, B., "Efficient Communication Operations On Passive Optical Star Networks", Proc. IEEE 1st Intl. Conf. On Massively Parallel Processing Using Optical Interconnects, 1994, pg. 52-58.

[23] Desai, K.R., Ghose, K., "An Evaluation of Communication Protocols For Star-Coupled Multidimensional WDM Networks for Multiprocessors", Proc. IEEE 1st Intl. Conf. On Massively Parallel Processing Using Optical Interconnects, 1994, pg. 42-49.

[24] Takaya, M., Nagasawa, S., and Murakami, Y., "Design and Performance of Very High-Density Multifiber Connectors Employing Monolithic 60-Fiber Ferrules", IEEE Phot. Tech. Lett., V 11, 1999, pg. 1446-1448.

[25] Miller, C. M., Mettler, S. C., and White, I. A., Optical Fiber Splices and Connectors: Theory and Methods, Marcel Dekker, Inc. New York, NY, 1986, pg. 20.

[26] Hu, S.-Y., Ko, J., Hegblom, E.R., Coldren, L.A., "Multimode WDM optical data links with monolithically integrated multiple-channel VCSEL and photodetector arrays" IEEE J. Quant. Elec., V34, 1998, pg. 1403 –1414.

[27] Aronson, L.B., Lemoff, B.E., Buckman, L.A., Dolfi, D.W., "Low-cost multimode WDM for local area networks up to 10 Gb/s", IEEE Phot. Tech. Lett., V10, 1998, pg. 1489 –1491.

[28] Chang-Hasnain, C.J., Harbison, J.P., Zah, C.E., Florez, L.T., Andreadakis, N.C., "Continuous wavelength tuning of two-electrode vertical cavity surface emitting lasers", Electronics Letters, V27, 1991, pg. 1002 –1003.

[29] Larson M.C., Massengale A.R., Harris J.S., "Continuously tunable micromachined vertical cavity surface emitting laser with 18nm wavelength range" Electronics Letters, V 32, 1996, pg. 330-332.

[30] Larson, M.C., Pocha, M.D., Garrett, H.E., Petersen, H.E., Bond, S.W., Drayton, R.F., Patel, R.R., Emanuel, M.A., Deri, R.J., and Lowry, M.E., "Multiwavelength VCSEL Transmitter for WDM Parallel Optical Fiber Interconnects", IEEE LEOS 12th Annual Meeting, 1999, paper ThM3, pg. 764-765.

[31] Takato, N., Kurokawa, T., "Polymer Waveguide Star Coupler", App. Opt. V19, 1982, pg. 1940-1942.

[32] Bengtsson, J., Eriksson, N., Larsson, A., "Small-Feature Size Fan-out Kinoform Etched in GaAs", App. Opt., V 35, 1996, pg. 801-806.

[33] Shimoji, M., Othman, A., Crosbie, R., Frietman, E. E., "Free-space Optoelectronic Data-Distributor for Fully-Connected Massively parallel processors", SPIE V3288, 1998, pg. 220-231.

Chapter 2 Filter Design and Modeling

2.1 Introduction

In this chapter we describe the general approach to filter design used during the course of this work. The emphasis is on bandpass filter designs for the λ-Connect system as indicated in the previous chapter. We begin with some initial considerations regarding the ideal WDM bandpass filter. Following this we review the filter requirements for λ-Connect. Thin film bandpass filters can be thought of as multiple cavity Fabry-Perot (MC-FP) resonators. Therefore, we review the basic properties of the Fabry-Perot cavity (FPC) and then treat the problem of multiple coupled Fabry-Perot cavities (FPCs) focusing on the implications for bandpass filter design. We are interested in how the transmission properties of coupled resonators are affected by changes in the number of cavities, the cavity lengths, and the mirror reflectivities. Understanding this problem provides useful insight for the design of thin film filters for WDM applications. The effects of distributed Bragg reflectors on the simple model are briefly discussed prior to describing the general filter design techniques used in this dissertation research. A bandpass filter meeting the λ-Connect filter requirements was designed as part of this dissertation research. We conclude the chapter with a description of this filter design.

2.2 The Ideal WDM Bandpass Filter

The ideal WDM filter has 100% transmission within the passband and 100% rejection (reflection) outside of the passband as shown in Figure 1. The passband edges

are infinitely steep thereby allowing data channels to be spaced right next to each other without causing any channel to channel cross-talk.

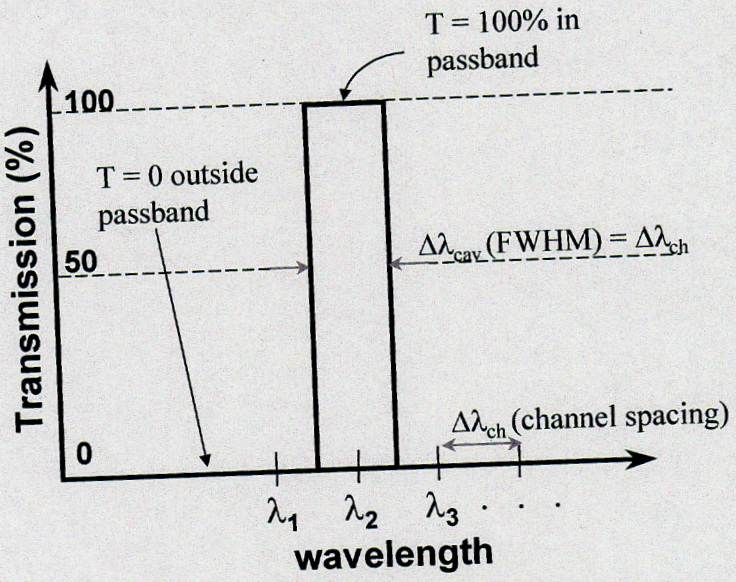


Figure 1. The ideal WDM bandpass filter. The filter has 100% transmission in the passband, zero transmission elsewhere, and squareness, S = 1. This allows spectral channels to be placed right next to each other with FWHM = $\Delta\lambda_{ch}$.

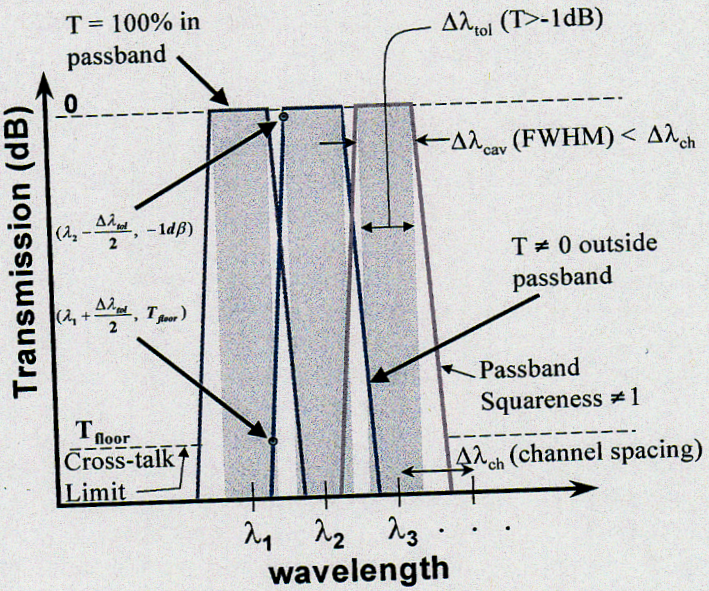


Figure 2. Realistic WDM bandpass filters. The passband squareness is < 1, implying that the FWHM is < $\Delta\lambda_{ch}$. Also, the system cross-talk limits and channel spacing determine the filter FWHM.

Figure 2 shows the pass band shape of realistic filters. In real systems flat-topped passbands are possible; however, the edges of the filter passband are never infinitely steep. This means that there will always be some level of transmission outside of the passband (cross-talk) which forces us to increase the channel spacing compared to the ideal case so that adjacent wavelength channels will not cause errors in the desired data. This increased channel spacing can limit the overall channel count of a system since all of the wavelength channels must fall within the sensitivity range of the photo-detector. Therefore, to maximize system channel count it is desirable to have filters with very steep passband edges.

To compare filters in terms of their usefulness for WDM applications we use a figure of merit, S, that is defined as:

$$S = \frac{\Delta\omega_{-3dB}}{\Delta\omega_{-20dB}} \tag{1}$$

In this equation, $\Delta\omega_{-3dB}$ is the filter full width at half maximum (FWHM) and $\Delta\omega_{-20dB}$ is the passband width at 1% transmission (T = -20dB). This figure of merit gives a measure of the "squareness" of the filter transmission function. The ideal WDM filter with a perfectly rectangular transmission function has S=1.

2.3 Filter Requirements for λ-Connect

The filter requirements for λ-Connect are as follows:

- spectral channel spacing $\Delta\lambda_{ch} \sim 10$nm
- channel tolerance window $\Delta\lambda_{tol} = 5$nm (at 1dB excess loss)
- crosstalk rejection ≥ 25 dB
- insertion loss < 2dB
- passband squareness $S \geq 0.45$

The squareness requirements for λ-Connect can be calculated by assuming that the filters have a trapezoidal passband shape (transmission vs. wavelength) and using the other system parameters given above. To calculate the squareness, we need to know the wavelengths where the passband transmission is –3dB and –20dB of its maximum value. These wavelengths can be calculated if the slope of the filter passband edge is known.

To find the slope of the passband edge 2 points on the transmission vs. wavelength (T vs λ) curve are needed. The system crosstalk and channel tolerance window requirements provide these two points as described below.

Consider 2 adjacent channels 1 and 2 with wavelengths λ_1 and λ_2 ($\lambda_2 > \lambda_1$) that are nominally separated by the system channel spacing $\Delta\lambda_{ch}$. Around each channel there is a tolerance window $\Delta\lambda_{tol}$ to allow for thermal drift of system components. under normal operating conditions signals for channel 1 could be transmitted anywhere in the range $\lambda_1 \pm \Delta\lambda_{tol}/2$ depending on the heat load at a given transmitter,. Receivers that use channel 2 have a bandpass filter whose center wavelength is λ_2. This filter must provide at least 25dB of rejection for signals from channel 1 to meet the system cross-talk requirements. This means that the passband transmission function for this filter must pass through the point: ($\lambda_1 + \Delta\lambda_{tol}/2$, -25dB). Furthermore, the edges of the tolerance window for each channel are defined at the points where there is 1 dB of excess loss. By excess loss we mean loss in addition to the filter insertion loss at the channel center wavelength. This gives us another point along the passband transmission curve namely: ($\lambda_2 - \Delta\lambda_{tol}/2$, -1dB). Using these two points the slope of the filter passband edge can be calculated. From this slope, the width of the passband at –3dB and –20dB can easily be calculated. The filter

squareness is the ratio of these widths. This calculation reveals that the λ-Connect system requires S ≈ 0.45 to allow for 10 nm channel spacings and 5nm tolerance windows with no more than -25 dB cross-talk from adjacent channels.

In the following sections we shall see how coupled Fabry-Perot cavities can be used to create filter responses that meet the λ-Connect design target for squareness.

2.4 A Simple Model for Bandpass Filters

2.4.1 The Fabry-Perot Cavity[1]

The reflection and transmission properties of the FPC are derived assuming infinite plane wave fields, lossless thin dielectric mirrors, and steady state conditions. This serves as the basis for the model of coupled FPCs that will be developed in the next section. As seen in Figure 3, the ideal FPC consists of 2 mirrors M_1 and M_0 separated by a distance L. The electric field in the vicinity of the cavity can be broken up into: the field incident on M_1, denoted by E_i, the reflected field E_r, the field circulating inside the cavity, E_c, and the field transmitted by the cavity, E_t. The wavelength-dependent transmission and reflection properties of the FPC are found by relating the fields E_r, E_c, and E_t to the incident field E_i.[2]

The field circulating inside a FPC consists of the portion of the incident field that is transmitted by M_1, and the field that was previously inside the cavity, having made a round-trip through the cavity and reflecting off both mirrors.

$$E_c = jt_1 E_i + r_1 r_0 e^{-2jkL} E_c \tag{2}$$

Where r_s and t_s are amplitude reflectivity and transmissivity of mirror s, k is the wave number, L is the cavity length and $j^2 = -1$.

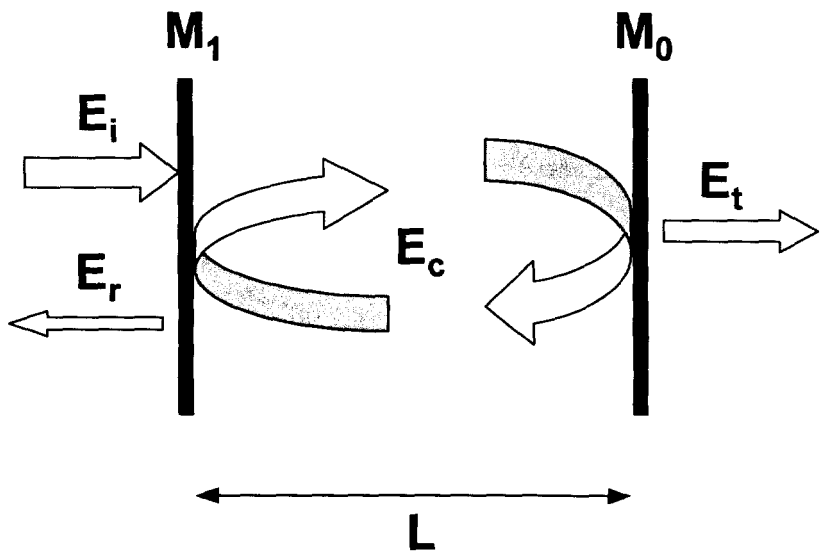


Figure 3. The simple Fabry-Perot cavity. Mirrors M_1 and M_0 are separated by cavity length L. The incident field is E_i. E_r is the reflected field. E_c is the field circulating inside the cavity and E_t is the transmitted field.

From this equation we see that E_c is related to E_i by:

$$\frac{E_c}{E_i} = \frac{jt_1}{1 - r_1 r_0 e^{-2jkL}} \tag{3}$$

Since r_0 and r_1 are less than 1, the denominator of equation 2 represents the sum of an infinite geometric series whose ratio is $r_1 r_0 e^{-2jkL}$. This means that equation 2 represents the summation of an infinite number of beams each making one more roundtrip around the cavity than the previous one.

The field transmitted by the FPC is the portion of E_c that is transmitted by M_0. Since the reference plane for E_c is the surface of M_1 just inside the cavity, the phase shift accumulated in a single pass through the cavity must be accounted for in calculating E_t.

$$E_t = jt_0 e^{-jkL} E_c \tag{4}$$

Using equation (3) in equation (4) we find

$$\frac{E_t}{E_i} = \frac{-t_1 t_0 e^{-jkL}}{1 - r_1 r_0 e^{-2jkL}} \tag{5}$$

The reflected field consists of the portion of the incident field that is reflected by M_1 plus the portion of E_c that makes a round-trip through the cavity and is transmitted by M_1:

$$\frac{E_r}{E_i} = r_1 + jt_1 r_0 e^{-2jkL} \frac{E_c}{E_i} \quad = \quad \frac{r_1 - r_0 e^{-2jkL}}{1 - r_1 r_0 e^{-2jkL}} \tag{6}$$

Equations 3, 5 and 6 describe the circulating, transmitted, and reflected fields from a FPC in terms of the incident field. The transmitted and reflected intensities (transmittance and reflectance) are the square magnitudes of the transmitted and reflected fields respectively.

$$T = \frac{|E_t|^2}{|E_i|^2} = \frac{T_1 T_0}{1 - 2r_1 r_0 \cos(2kL) + R_1 R_0} \tag{7 a}$$

$$R = \frac{|E_r|^2}{|E_i|^2} = \frac{R_1 - 2r_1 r_0 \cos(2kL) + R_0}{1 - 2r_1 r_0 \cos(2kL) + R_1 R_0} \tag{7 b}$$

where, $T_j = t_j^2$ and $R_j = r_j^2$ are respectively the intensity transmission and reflection coefficients of mirror j.

Clearly the denominator in equation 7a governs the resonance behavior of T. The minima of the denominator are the transmission maxima. Solving for these leads to the well-known condition that on resonance the cavity length is equal to an integer number of half-wavelengths.

$$2kL = 2m\pi \Rightarrow k = \frac{m\pi}{L} \quad or \quad L = \frac{m\lambda}{2} \tag{8}$$

The term $2kL$ represents the phase change of the field upon one round-trip through the cavity, therefore, equation (8) indicates that, on resonance, the round-trip phase change of

the field must be an integer multiple of 2π. This equation is known as the **resonant phase condition**. The spacing between two transmission maxima (axial mode spacing or free spectral range) is $\Delta\omega_{ax} = 2\pi c/2L$. For the case of matched mirrors, where M_1 and M_0 are identical (and lossless) the cavity exhibits 100% transmission on resonance. This is known as the **impedance matched condition.**

The width of the passband at any fraction of the maximum transmission ($T = T_{max}/f$) is given by those frequencies where the denominator of equation (7a) is f times its minimum value:

$$D(k) = fD_{\min} \quad \Rightarrow \quad 1 - 2r_1 r_0 \cos(2kL) + r_1^{\,2} r_0^{\,2} = f(1 - r_1 r_0)^2 \tag{9}$$

The width of the passband at $T = T_{max}/f$ is then:

$$\Delta\omega_{T/f} = \frac{2c}{L}\sin^{-1}\left(\sqrt{\frac{(f-1)D_{\min}}{4r_1 r_0}}\right) = \frac{2c}{L}\sin^{-1}\left(\sqrt{\frac{(f-1)(1-r_1 r_0)^2}{4r_1 r_0}}\right) \tag{10}$$

The full-width-at-half-maximum (FWHM) of the cavity resonances (designated $\Delta\omega_{cav}$) are given by those frequencies where the transmission is equal to one-half of its maximum value. Using f=2 in equation (10) above the FWHM is given by:

$$\Delta\omega_{cav} = \frac{2c}{L}\sin^{-1}\left(\frac{1 - r_1 r_0}{2\sqrt{r_1 r_0}}\right) \approx \frac{\Delta\omega_{ax}}{\pi}\left(\frac{1 - r_1 r_0}{\sqrt{r_1 r_0}}\right) \tag{11}$$

The finesse, F, of a FPC is defined as $\Delta\omega_{ax}/\Delta\omega_{cav}$. It gives a measure of the fine-ness of the transmission maxima compared to the free spectral range. From equation (11) we see that the finesse is:

$$F = \left(\frac{\pi\sqrt{r_1 r_0}}{1 - r_1 r_0}\right) \tag{12}$$

As mentioned previously, the squareness figure of merit characterizes how close a filter is to the ideal WDM filter (S=1). Using equation (11) and equation (10) (with f = 100), the squareness figure for the single FPC is:

$$S = \frac{\sin^{-1}\left(\frac{1-r_1 r_0}{2\sqrt{r_1 r_0}}\right)}{\sin^{-1}\left(\frac{\sqrt{99}(1-r_1 r_0)}{2\sqrt{r_1 r_0}}\right)} \tag{13}$$

Note that the mirror reflectivities must be high enough to ensure that the off-resonance transmission falls to less than $T_{max}/100$ between axial modes in order for the squareness figure to be well defined. Using sin(x) ~ x in the limit of high mirror reflectivities the squareness figure for the single FPC has a limiting value of approximately 0.1 . Clearly the single FPC is a long way from the ideal case. More than 50% of the incident optical power falls outside of the lorentzian lineshape of the FPC passband. In the next section we examine the properties of coupled FPC resonators to see how they improve upon the performance of the single FPC. Table 1 summarizes the FPC properties described above.

Transmitted Field	$\frac{E_t}{E_i} = \frac{-t_1 t_0 e^{-jkL}}{1 - r_1 r_0 e^{-2jkL}}$
Reflected Field	$\frac{E_r}{E_i} = \frac{r_1 - r_0 e^{-2jkL}}{1 - r_1 r_0 e^{-2jkL}}$
Transmittance	$T = \frac{\lvert E_t \rvert^2}{\lvert E_i \rvert^2} = \frac{T_1 T_0}{1 - 2r_1 r_0 \cos(2kL) + R_1 R_0}$
Maximum Transmittance	$T_{\max} = \frac{T_1 T_0}{(1 - r_1 r_0)^2}$
Transmittance (alternate form)	$T = T_{\max}\left(1 + \left(\frac{2F}{\pi}\right)^2 \cdot \sin^2(kL)\right)^{-1}$
Reflectance	$R = \frac{\lvert E_r \rvert^2}{\lvert E_i \rvert^2} = \frac{R_1 - 2r_1 r_0 \cos(2kL) + R_0}{1 - 2r_1 r_0 \cos(2kL) + R_1 R_0}$
Maximum Reflectance	$R_{\max} = \frac{(r_1 + r_0)^2}{(1 + r_1 r_0)^2} = \frac{4R_0}{(1 + R_0)^2}$ $(if\ r_1 = r_0)$
Axial Mode Spacing	$\Delta\omega_{ax} = \frac{\pi c}{L}$
FWHM	$\Delta\omega_{cav} \approx \frac{\Delta\omega_{ax}}{\pi}\left(\frac{1 - r_1 r_0}{\sqrt{r_1 r_0}}\right)$
Width at T = T_{max}/f	$\Delta\omega_{T/f} = \frac{2c}{L}\sin^{-1}\left(\sqrt{\frac{(f-1)(1 - r_1 r_0)^2}{4r_1 r_0}}\right)$
Finesse	$F = \left(\frac{\pi\sqrt{r_1 r_0}}{1 - r_1 r_0}\right)$
Squareness	$S = \frac{\sin^{-1}\left(\frac{1 - r_1 r_0}{2\sqrt{r_1 r_0}}\right)}{\sin^{-1}\left(\frac{\sqrt{99}(1 - r_1 r_0)}{2\sqrt{r_1 r_0}}\right)}$
Note	Transmission resonances occur when kL = mπ
	If $R_1 = R_0$ (impedance matched condition) → T=1 on resonance

Table 1. Properties of the Fabry-Perot Cavity.

2.4.2 Coupled Fabry Perot Cavities

We now consider a system of coupled FPCs as illustrated in Figure 4. We shall call the overall system the composite resonator. An individual cavity within the system is called a sub-resonator.

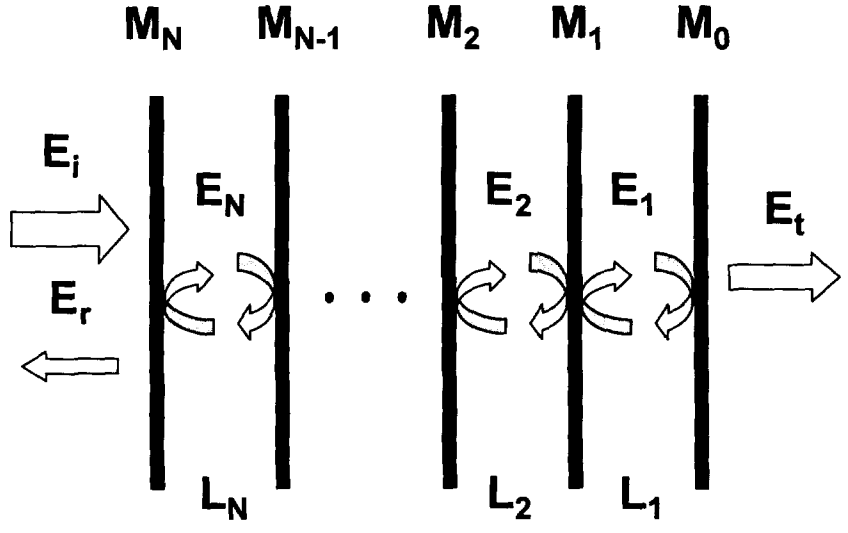


Figure 4. N-Cavity Fabry-Perot Resonator.

2.4.2.A Two Cavity Resonator

Before considering the general problem of N-coupled resonators it is useful to analytically examine the case for N=2. This will highlight all of the general properties that make coupled resonators desirable for WDM filters.

Consider two coupled Fabry-Perot resonators as shown in Figure 5a. In finding the transmission and reflection properties of this system, we take advantage of the results derived in the previous section. Equation 5 relates the field transmitted by a FPC to the field that is incident on the cavity. Similarly, Equation 6 relates the field reflected from a FPC to the incident field. These, by definition, are the amplitude transmission and reflection coefficients for a FPC treated as a single reflecting element. Thus an

equivalent reflector whose transmission and reflection coefficients are given by equations 5 and 6 respectively can replace the two mirrors of a FPC in calculations of overall system behavior. The coefficients of the equivalent reflector are both complex and wavelength-dependent.

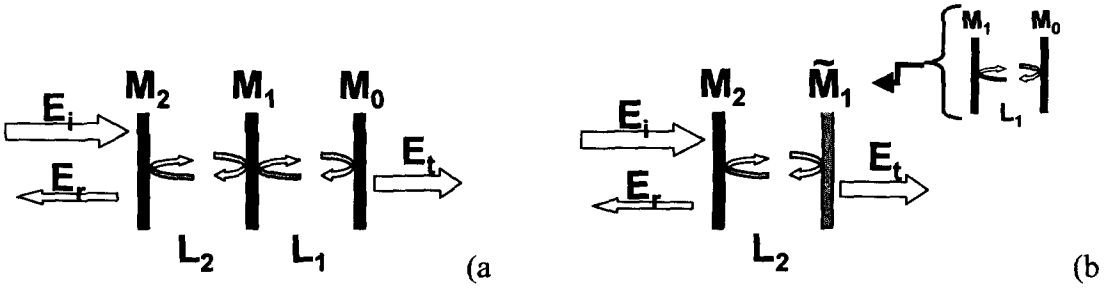


Figure 5. a) Two-cavity resonator consisting of mirrors M_0, M_1, and M_2. b) Two-cavity resonator treating cavity 1 as an equivalent reflector.

We analyze the two-cavity resonator by replacing the two rightmost mirrors in Figure 5a by an equivalent reflector as shown in Figure 5b. Since the system is now simply a FPC with mirrors M_2 and $\tilde{M}_1$, equations 5, 6 and 7 apply. As mentioned above, the amplitude reflectivity of the equivalent reflector is given by equation 6 while its transmissivity is given by equation 5. Thus the behavior of the two-cavity resonator is found by applying equations 5 and 6 twice recursively. The transmitted and reflected fields for the two-cavity resonator are:

$$\frac{E_t}{E_i} = \frac{jt_2\tilde{t}e^{-jkL_2}}{1-r_2\tilde{r}e^{-2jkL_2}} = \frac{-jt_2t_1t_0e^{-jk(L_2+L_1)}}{1-r_2r_1e^{-2jkL_2}-r_1r_0e^{-2jkL_1}+r_2r_0e^{-2jk(L_1+L_2)}} \tag{14}$$

and

$$\frac{E_r}{E_i} = \frac{r_2-\tilde{r}e^{-2jkL_2}}{1-r_2\tilde{r}e^{-2jkL_2}} = \frac{r_2\left(1-r_1r_0e^{-2jkL_1}\right)-\left(r_1-r_0e^{-2jkL_2}\right)e^{-2jkL_2}}{1-r_2r_1e^{-2jkL_2}-r_1r_0e^{-2jkL_1}+r_2r_0e^{-2jk(L_1+L_2)}} \tag{15}$$

The denominator of equations 14 and 15 has terms involving the two sub-resonators as well as a term involving the two end mirrors. The transmitted intensity is given by:

$$T = \frac{T_2 T_1 T_0}{1 + r_1^2 r_0^2 + r_2^2 r_1^2 + r_2^2 r_0^2 - 2r_1 r_0 (1 + r_2^2)\cos(2kL_1) - 2r_2 r_1 (1 + r_0^2)\cos(2kL_2) + 2r_2 r_0 [\cos(2k(L_1 + L_2)) + r_1^2 \cos(2k(L_1 - L_2))]}$$

(16)

As with the single FPC, the transmission maxima are the minima of the denominator. The derivative of the denominator of equation (16) is:

$$\frac{\partial D}{\partial k} = 4r_1 r_0 (1 + r_2^2) L_1 \sin(2kL_1) + 4r_2 r_1 (1 + r_0^2) L_2 \sin(2kL_2) - 4r_2 r_0 [(L_1 + L_2)\sin(2k(L_1 + L_2)) + r_1^2 (L_1 - L_2)\sin(2k(L_1 - L_2))] \quad (17)$$

The extrema of the transmission are then the zeroes of equation 17. Although equation 17 appears rather cumbersome and unrevealing, we can gain useful insight by observing that, for WDM filter applications, the filters are generally symmetric. That is, the sequence of layers is identical regardless of which direction the filter is traversed. This guarantees that the impedance-matched condition is satisfied on resonance. For the coupled FPC resonator we can make the **symmetric filter assumption** by setting the reflectivities of the mirrors and the lengths of the cavities on both sides of the composite resonator equal to each other. For the two-cavity case, this requires that $r_2 = r_0$ and $L_2 = L_1$. Under these conditions, equation 17 simplifies to:

$$8r_0 r_1 (1 + r_0^2) L_1 \sin(2kL_1) - 8r_0^2 L_1 \sin(4kL_1) \quad (18)$$

Solving for the maxima of the transmission we find the condition that:

$$\cos(2kL_1) = \frac{r_1 (1 + r_0^2)}{2r_0} \quad \Rightarrow \quad \omega = ck = \frac{c}{2L_1}[2m\pi \pm a\cos(\frac{r_1 (1 + r_0^2)}{2r_0})] \quad (19)$$

Equation 19 states that the round-trip phase change of the field circulating inside cavity 2 plus the phase change due to reflection from cavity 1 must equal an integer multiple of 2π. This is the resonant phase condition corresponding to the transmission maxima of the two-cavity system. Using equation 19 to substitute for $\cos(2kL_1)$ in equation 7b we find

that, on resonance, the reflectance of the equivalent reflector equals R_0. Since $R_2 = R_0$ by design this shows that the impedance match condition is met when the transmission is maximum.

The term ($m\pi c/L_1$) in equation 19 is the resonant frequency for a single FPC. Since the cosine function is even (i.e. cos(-x) = cos(x)) the two-cavity resonator has 2 transmission peaks that are shifted away from the single cavity transmission maxima (one shifted lower in frequency, and one shifted higher in frequency). This type of "mode splitting" behavior is typical of coupled resonant systems. The spacing between the transmission maxima of the two-cavity resonator is:

$$\Delta\omega = \frac{c}{L_1} a\cos\left(\frac{r_1(1+r_0^2)}{2r_0}\right) = \frac{\Delta\omega_{ax}}{\pi} a\cos\left(\frac{r_1(1+r_0^2)}{2r_0}\right) \qquad (20)$$

Therefore, as the cavity lengths are increased (recall that $L_2 = L_1$ by assumption), the spacing between the maxima decreases. This property is useful when trying to design a filter with a desired passband width.

If r_1 is chosen correctly, the wavelength shift between the two modes can be eliminated:

$$r_1 = \frac{2r_0}{1+r_0^2} \quad \Rightarrow \quad \Delta\omega = 0 \qquad (21)$$

At this point, the transmission of the two-cavity resonator has only a single maximum with 100% transmission in each order. This condition is called critical coupling. If r_1 is greater than the value given in (21), the two sub-resonators are under-coupled. In this case the transmission of the two-cavity system exhibits 1 maximum with a transmission much less than 100%. If r_1 is less than the critical value, the two sub-resonators are over-coupled and two distinct peaks are observed as described above. Thus the reflectivity of

the coupling mirror is another tool that can be employed to alter the shape of the filter transmission function.

The FWHM of the two-cavity resonator is not as well defined as it was for the single FPC. If the coupling mirror reflectivity is low enough, the dip between the two maxima could be less than 50% of the maximum transmission. In this case, each peak would have its own well-defined FWHM. However, if the coupling mirror reflectivity is high enough, the transmission minima between the two peaks will not fall to 50% of the maxima. In this case, the peaks are not individually distinguishable. In order to compare with the single FPC we shall consider the FWHM of the passband to be the largest frequency interval (in a single order) between points where the transmission of the two-cavity system is 50% of its maximum value.

The frequencies where $T = T_{max}/f$, can be found by using the denominator of the two-cavity resonator on both sides of equation 9. The width of the two-cavity passband at $T = T_{max}/f$ is then:

$$\Delta\omega_{T/f} = \frac{c}{L} a\cos\left(\frac{r_1(1+r_0^{\,2})}{2r_0} - \frac{\sqrt{(f-1)D_{\min}}}{2r_0}\right) \tag{22}$$

The first term in the argument of the arccosine is the same as the argument of the arccosine term in equation (19). Under the symmetric filter assumption, the minimum value of the denominator is:

$$D_{\min} = \left(1-r_o^{\,2}\right)^2\left(1-r_1^{\,2}\right) \tag{23}$$

Using $f = 2$ and equations 22 and 23 the FWHM of the two-cavity resonator is:

$$\Delta\omega_{cav} = \frac{c}{L} a\cos\left(\frac{r_1(1+r_0^{\,2})}{2r_0} - \frac{\sqrt{(1-r_0^{\,2})^2(1-r_1^{\,2})}}{2r_0}\right) \tag{24}$$

The squareness figure for the two-cavity resonator is:

$$S = \frac{a\cos\left(\frac{r_1(1+r_0^2)}{2r_0} - \frac{\sqrt{(1-r_0^2)^2(1-r_1^2)}}{2r_0}\right)}{a\cos\left(\frac{r_1(1+r_0^2)}{2r_0} - \frac{\sqrt{99(1-r_0^2)^2(1-r_1^2)}}{2r_0}\right)} \tag{25}$$

As the reflectivity of either the outside mirrors or the coupling mirror approach 100% ($r_0 \rightarrow 1$, or $r_1 \rightarrow 1$) the squareness approaches 1 because the terms $(1-r_0^2)$ or $(1-r_1^2)$ respectively approach zero. Thus, the coupling mirror can both increase the width and improve the squareness of the passband compared to a single FPC.

The maximum reflectance of a composite resonator gives a measure of the ability of the system to reject unwanted signals outside of the passband (cross-talk suppression). The maximum reflectance of a coupled resonator system is:

$$R_{max} = 1 - T_{min} = 1 - \frac{D_{min}}{D_{max}} \tag{26}$$

where D_{max} and D_{min} are the maximum and minimum value of the denominator of the reflectance (and transmittance) function. Recall that numerator of equation (16) must equal D_{min} for the Transmission to be 100% on-resonance. Furthermore, the minimum transmission occurs when the denominator of equation (16) is maximized. This results in equation (26). It is intuitive that the reflectance maxima should fall halfway between the passbands of adjacent orders.

For the single FPC the minimum value of the denominator is $(1-R_0)^2$. From equation (23) we see that the minimum value of the denominator for the two-cavity system is smaller than the minimum value for a single FPC. A similar calculation would show that the maximum value of the denominator is greater for the two-cavity system compared to

the single cavity. Equation (26) then indicates that, off resonance, the two-cavity system has a higher reflectance than the single FPC. For filter applications this indicates that the composite resonator has better cross-talk suppression than the single FPC.

Thus far we have analytically examined the two-cavity resonator problem. Treating the first cavity as an equivalent reflector we were able to compute the amplitude and intensity transmission of the two-cavity system. Making the symmetric filter assumption allowed us to find the frequencies of maximum transmission for the composite resonator. In general there are two transmission peaks within a given order. The spacing between the transmission peaks was shown to depend directly upon the reflectivity of the mirrors and inversely upon the cavity lengths. By choosing the reflectivity of the coupling mirror correctly the spacing between the transmission peaks can be reduced to zero. We saw that when the resonant phase condition is met for the two-cavity system the impedance match condition is also met. This indicates that the two-cavity resonator has unity transmission on resonance if the system is symmetric. In the limit of very high mirror reflectivities we saw that the squareness figure of the coupled system approaches unity. Outside of the passband the coupled system has better cross-talk suppression than the single FPC due to its higher reflectivity.

The effect of the coupling between the two cavities was to introduce new terms in the denominator of the transmission and reflection functions as compared to the single cavity case. These new terms led to new resonances. The presence of the new resonances broadens the passband, increases its squareness and improves the cross-talk suppression of the coupled system compared to a single FPC. Table 2 summarizes the properties of the two-cavity resonator. We now turn to the more general case of N coupled cavities.

Transmitted Field	$\frac{E_t}{E_i} = \frac{jt_2\tilde{t}e^{-jkL_2}}{1-r_2\tilde{r}e^{-2jkL_2}} = \frac{-jt_2t_1t_0e^{-jk(L_2+L_1)}}{1-r_2r_1e^{-2jkL_2}-r_1r_0e^{-2jkL_1}+r_2r_0e^{-2jk(L_1+L_2)}}$
Reflected Field	$\frac{E_r}{E_i} = \frac{r_2-\tilde{r}e^{-2jkL_2}}{1-r_2\tilde{r}e^{-2jkL_2}} = \frac{r_2\left(1-r_1r_0e^{-2jkL_1}\right)-\left(r_1-r_0e^{-2jkL_2}\right)e^{-2jkL_2}}{1-r_2r_1e^{-2jkL_2}-r_1r_0e^{-2jkL_1}+r_2r_0e^{-2jk(L_1+L_2)}}$
Transmittance	$\frac{T_2T_1T_0}{1+r_1^2r_0^2+r_2^2r_1^2+r_2^2r_0^2-2r_1r_0(1+r_2^2)\cos(2kL_1)-2r_2r_1(1+r_0^2)\cos(2kL_2)+2r_2r_0[\cos(2k(L_1+L_2))+r_1^2\cos(2k(L_1-L_2))]}$
Resonant Phase condition	$\cos(2kL_1) = \frac{r_1(1+r_0^2)}{2r_0} \quad \Rightarrow \quad \omega = ck = \frac{c}{2L_1}[2m\pi \pm a\cos(\frac{r_1(1+r_0^2)}{2r_0})]$
Maximum Reflectance	$R_{max} = \frac{R_1(1+R_0)^2+4R_0+4r_1r_0(1+R_0)}{(1+R_0)^2+4R_1R_0+4r_1r_0(1+R_0)}$ *(for symmetric filters)*
FWHM	$\Delta\omega_{cav} = \frac{c}{L}a\cos(\frac{r_1(1+r_0^2)}{2r_0}-\frac{\sqrt{(1-r_0^2)^2(1-r_1^2)}}{2r_0})$
Width at T = T_{max}/f	$\Delta\omega_{T/f} = \frac{c}{L}a\cos\left(\frac{r_1(1+r_0^2)}{2r_0}-\frac{\sqrt{(f-1)D_{min}}}{2r_0}\right)$
D_{min}	$D_{min} = (1-r_0^2)^2(1-r_1^2)$
Squareness	$S = \frac{a\cos\left(\frac{r_1(1+r_0^2)}{2r_0}-\frac{\sqrt{(1-r_0^2)^2(1-r_1^2)}}{2r_0}\right)}{a\cos\left(\frac{r_1(1+r_0^2)}{2r_0}-\frac{\sqrt{99(1-r_0^2)^2(1-r_1^2)}}{2r_0}\right)}$
Note	If $R_2 = R_0$ and $L_2 = L_1$ (symmetric filter assumption) → T=1 on resonance

Table 2. Properties of the Two-Cavity Resonator

2.4.2.B Generalization To N-Cavities

Using the concept of equivalent reflectors the results of equations (14) and (15) can be generalized for multiple coupled resonators by re-writing them as:

$$\tilde{t}_N \equiv \left.\frac{E_t}{E_i}\right|_N = \frac{jt_N\tilde{t}_{(N-1)}e^{-jkL_N}}{1-r_N\tilde{r}_{(N-1)}e^{-2jkL_N}} \tag{27}$$

and

$$\tilde{r}_N \equiv \left.\frac{E_r}{E_i}\right|_N = \frac{r_N - \tilde{r}_{(N-1)}e^{-2jkL_N}}{1 - r_N\tilde{r}_{(N-1)}e^{-2jkL_N}} \tag{28}$$

where $\tilde{r}_N$ and $\tilde{t}_N$ are the amplitude reflectivity and transmissivity of the equivalent reflector consisting of the N sub-resonators to the right of (and including) the Nth mirror. In these equations r_N and t_N are the amplitude reflectivity and transmissivity of the Nth mirror respectively. Also $\tilde{r}_{(N-1)}$ and $\tilde{t}_{(N-1)}$ are the amplitude reflectivity and transmissivity of the equivalent reflector consisting of the N-1 sub-resonators to the right of the Nth mirror. Starting with $\tilde{r}_0 = r_0$, and $\tilde{t}_0 = t_0$, the transmitted and reflected fields for any number of coupled resonators can be calculated by recursively applying equations (27) and (28) respectively[3]. This concept is illustrated in Figure 6. Since equations (27) and (28) are recursive, the nature of the solution is analytically intractable for more than 2 or 3 coupled cavities.

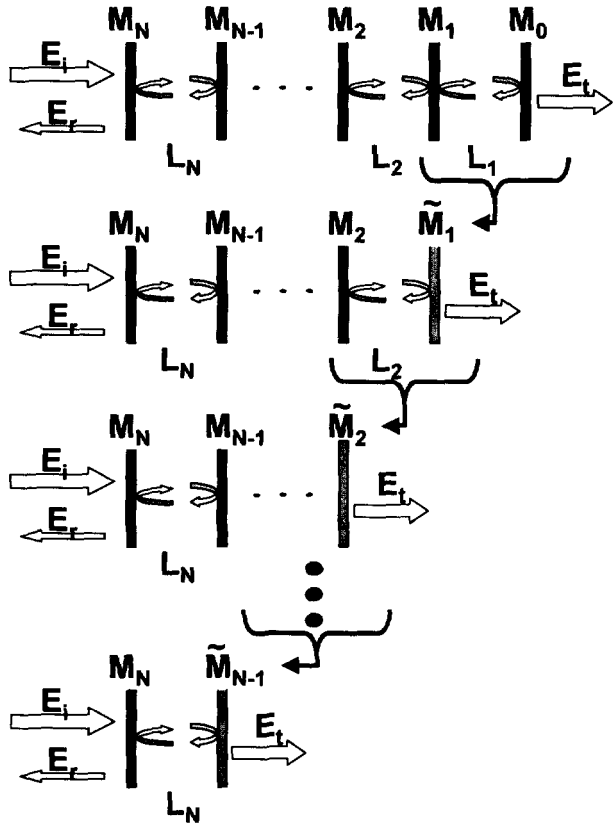


Figure 6. Analysis of the N-cavity resonator through recursive application of the equivalent resonator concept. In each stage the two right-most reflectors are treated as a single equivalent reflector. Repeating the process N-1 times results in the system shown at the bottom of the figure.

2.4.3 Numerical Simulations

Having understood the fundamental nature of the coupled resonator problem we now use numerical simulations to illustrate the general trends of interest for WDM filter design. Thin film interference filters for WDM typically employ cavities that are $\lambda_0/2$ to a few times $\lambda_0/2$ in optical thickness (where λ_0 is the filter center wavelength). Therefore, in the numerical simulations described here, all cavity lengths are $\lambda_0/2$ unless otherwise stated. A center wavelength of 800nm was chosen in order to plot transmission curves with respect to wavelength rather than offset from center wavelength.

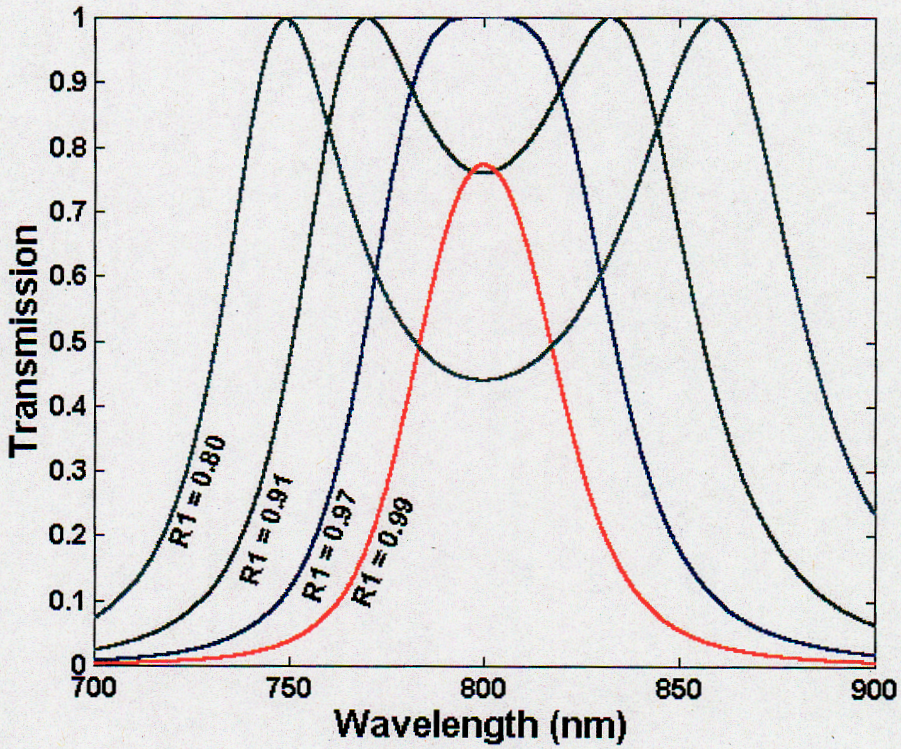


Figure 7. Transmission of two-cavity resonator for various coupling mirror reflectivities. The outer mirror reflectivities are both 0.70.

A short program that computes equations (27) and (28) with a varying number of reflectors was written using Matlab. A listing of the program is given in appendix AAA. Figure 7 shows the Transmission curve for two coupled FPCs as a function of wavelength for various coupling mirror reflectivities (R_1). The transition from over-coupled to critically coupled to under-coupled is demonstrated as the two peaks of unity transmission move closer together, become one, and then drop to a much lower value of maximum transmission. Figure 8 shows the effect of increasing the cavity length in multiples of $\lambda_0/2$ for a two-cavity system. As indicated by equation (20), increasing the cavity lengths pulls the two transmission peaks closer together. It also tends to make the peaks narrower, however, the squareness of the system is almost unchanged.

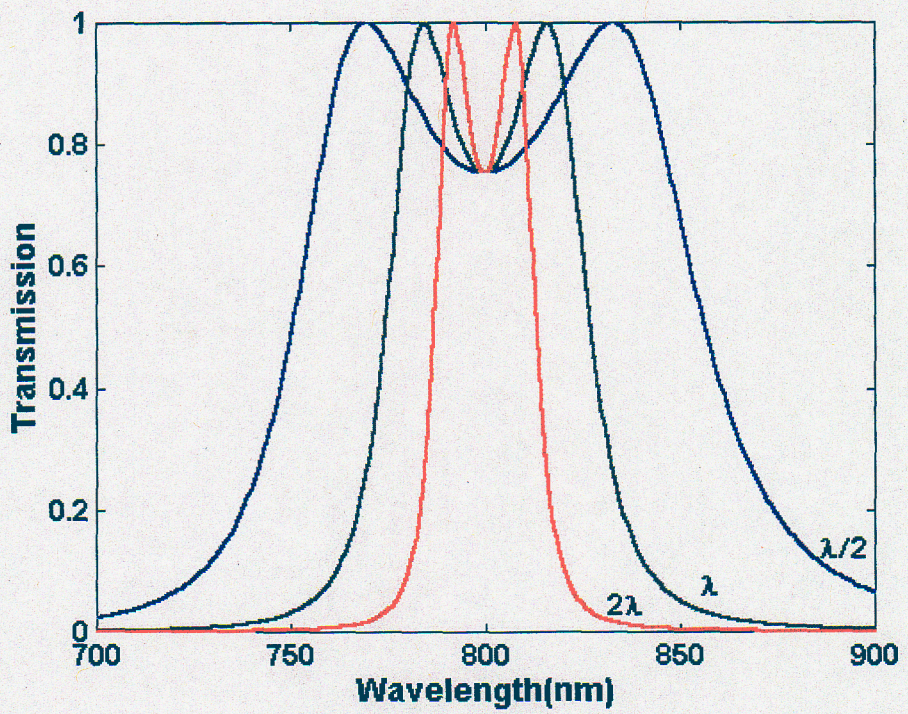


Figure 8. Transmission of two-cavity resonator for various cavity lengths. The cavity lengths are indicated in fractions of the center wavelength. The outer mirror reflectivities are both 0.70 and the coupling mirror reflectivity is 0.91.

Figure 9 shows the effect of increasing the number of cavities in the N-cavity resonator. This figure assumes that all cavities are of equal length and that all of the reflectors are identical. The number of transmission peaks is equal to the number of cavities in the composite resonator. As the number of cavities increases the width of the passband increases. By passband we refer to the wavelength interval between the outer edges of the outermost transmission maxima. Also, the squareness of the passband tends to increase as the number of cavities is increased (ignoring ripple in the passband).

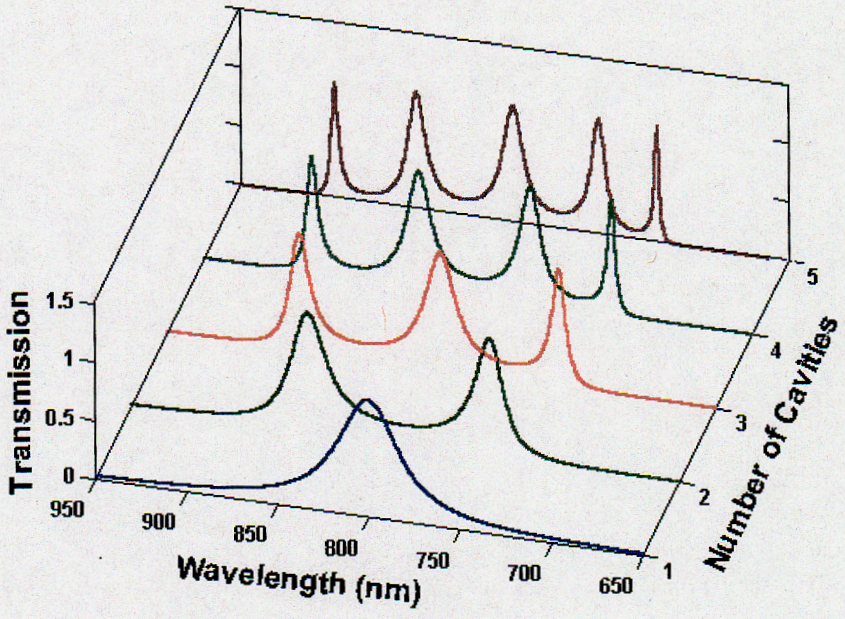


Figure 9. Transmission of composite resonator for 1,2,3,4, and 5 coupled cavities. Reflectance of all mirrors is 0.85. Each peak exhibits 100% Transmission.

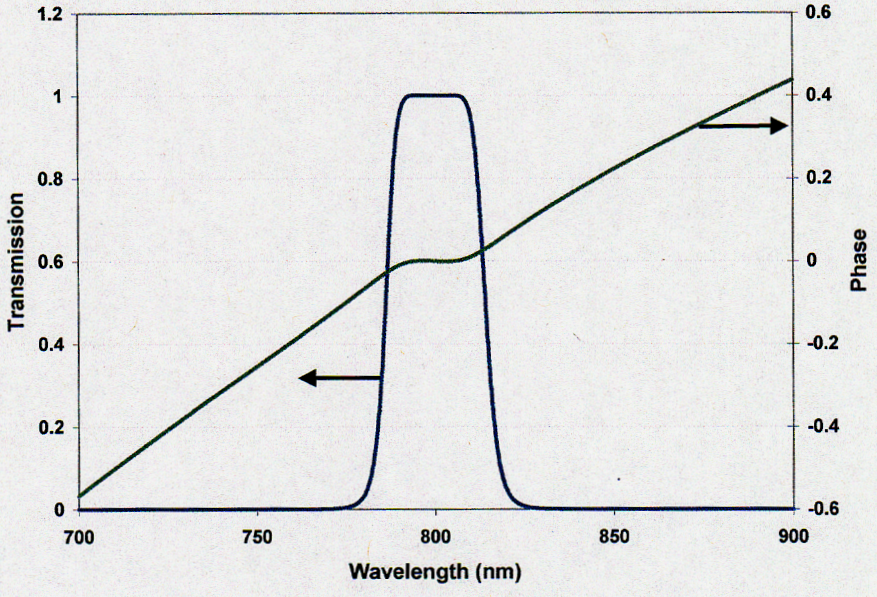


Figure 10. Flat-top transmission of a 5-cavity resonator. Also shown is the net round-trip phase change of the field in the central cavity. When the phase is zero the transmission is maximum. The reflectivities of the mirrors, from the center of the filter outwards are: 98.4%, 97.0%, and 64.3%.

Figure 10 shows that choosing appropriate mirror reflectivities can flatten the passband of the N-cavity resonator. This is qualitatively understood by thinking of the N-1 mirrors inside the cavity as an equivalent reflector that controls the coupling between the two outer mirrors. As seen for the two-cavity resonator, the reflectivity of the coupling mirror determines the spacing between the transmission peaks within a single order. If the coupling mirror reflectivity is chosen properly the peaks will all combine. Also shown in this figure is the net round-trip phase change of the field circulating inside the central cavity. This phase change consists of the round-trip phase change due to propagation around the cavity plus the change in phase due to reflection from the equivalent reflectors. When the net phase change is zero (or a multiple of 2π) the transmission is maximum. This provides a visual verification of the resonant phase condition. The flat-topped transmission demonstrated in the figure was obtained using 5-cavities and adjusting the mirror reflectivities through trial and error. The mirror reflectivities from the center of the filter outward (in both directions) are: 98.4%, 97.0%, and 64.3%.

2.4.4 Effect of Distributed Reflectors

The simple model developed thus far assumes "hard" mirrors where the reflection from each mirror occurs at the mirror surface. However, in thin film interference filters, the mirrors are distributed Bragg reflectors (DBRs). DBRs consist of alternating quarter wave thick layers of high and low index materials. Each high/low pair is known as a Bragg pair. At the center wavelength, successive reflections from all the material interfaces are in-phase leading to high reflectivity within a certain spectral band called

the stop band. The reflectivity of a DBR increases as the number of Bragg pairs is increased. The spectral extent of the stop band depends upon the index contrast of layers in the DBR as described by equation (29).

$$\frac{\Delta\lambda_{DBR}}{\lambda_o} \approx \frac{4}{\pi}\cdot\frac{n_2 - n_1}{n_2 + n_1} = \frac{2}{\pi}\cdot\frac{\Delta n}{n_{avg}} \tag{29}$$

Where $\Delta\lambda_{DBR}$ is the stop-band width, λ_o is the center wavelength, n_1 and n_2 are the refractive indices of the two layer materials, Δn is the index difference ($n_2 - n_1$) and n_{avg} is the average index.

Since the DBRs high reflectivity arises from the in phase summation of many small reflections distributed throughout the depth of the mirror. It is not possible to pinpoint the exact position where the reflection "actually happens" leading to ambiguity regarding the length of the cavity. This also means that the phase change upon reflection from a DBR is not constant but rather it varies almost linearly within the stop band.

The properties of DBRs have been extensively studied.[4,5] For normally incident light, the maximum reflectivity of a DBR consisting of *m* Bragg pairs is given by[6,7]:

$$R_{DBR} = \left(\frac{1-(n_1/n_2)^{2m}}{1+(n_1/n_2)^{2m}}\right)^2 \approx \tanh^2\left(\frac{m\cdot\Delta n}{n_{avg}}\right) \tag{30}$$

Taking the DBR reflectivity to be equal to its value on resonance we can group the DBR phase dispersion with the phase change due to propagation within the cavity. With DBR mirrors the FPC transmission then becomes:

$$T = \frac{|E_t|^2}{|E_i|^2} = \frac{\widetilde{T}_1\widetilde{T}_0}{1-2|\widetilde{r}_1||\widetilde{r}_0|\cos(2kL+2\phi(\lambda))+\widetilde{R}_1\widetilde{R}_0} \tag{31}$$

Here the terms $|\tilde{r}_1|$ and $|\tilde{r}_0|$ are the amplitude reflectivities (at the center wavelength) of DBRs 1 and 0 respectively. The term $2\phi(\lambda)$ is the phase shift due to reflection from the DBRs. For simplicity we have assumed that both DBRs have the same phase shift upon reflection. The phase of the reflected beam from a DBR at the center wavelength is either 0 or π which means that the on-resonance phase is essentially unchanged. However, the argument of the cosine in equation (31) varies more rapidly with wavelength than the corresponding term for the hard mirror FPC. Thus the DBR phase dispersion narrows the width of the FPC transmission function - as long as the DBR and the cavity have the same center wavelength. If the center wavelengths are not the same the phase dispersion could lead to a shifting of the transmission maxima away from the cavity center wavelength.

Figure 11 shows the transmission of a FPC with hard mirrors compared to a FPC using DBR mirrors having the same reflectivity on-resonance. The figure clearly demonstrates that FPCs with DBR mirrors tend to exhibit narrower transmission widths than equivalent FPCs with hard mirrors. The calculations of filter transmission for the DBR case were made using the transfer matrix method (see Appendix B). We now review the overall filter design process used in this dissertation.

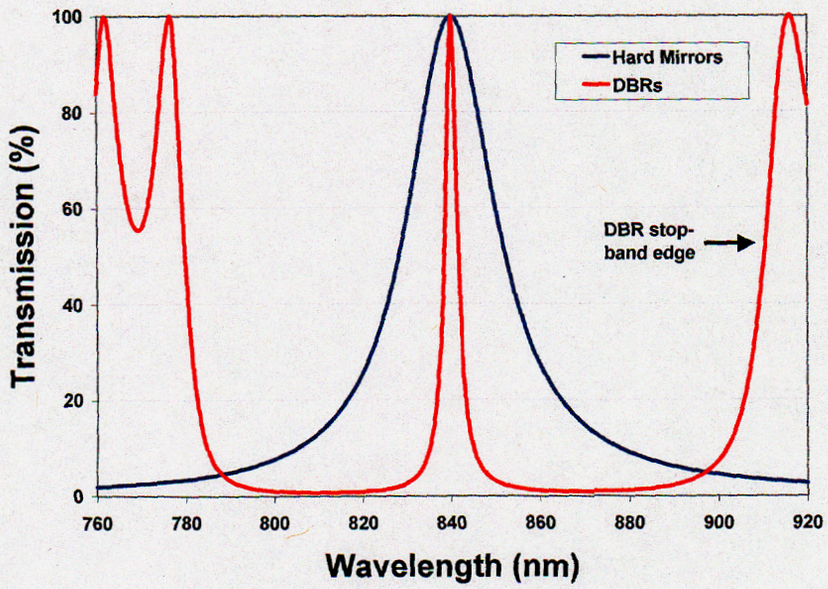


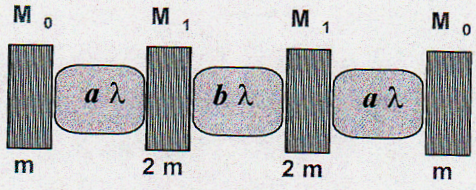
Figure 11 Effect of distributed Bragg reflectors (DBRs) on Fabry-Perot cavity transmission. The wider passband corresponds to the case of "hard" mirrors. The phase dispersion of the DBRs dramatically reduces the passband width. The maximum reflectivity of the DBRs is equal to the reflectivity of the hard mirror. The stop-band edges for the DBR are also seen in the figure.

2.5 Thin Film Filter Design

2.5.1 Filter Design Process

Thin film filter design is an extensive topic that cannot be fully covered in a few pages. Many books[4,5,6,7] on the subject have been written. We forego a general discussion of filter design since the previous sections have already indicated that multiple cavity Fabry-Perot (MC-FP) filters should meet the λ-Connect system requirements. Instead, we present here the general guidelines followed in designing filters for λ-Connect. The basic filter design approach is to use DBR mirrors and half-wave cavities to create MC-FP filters with the desired performance.

From earlier discussions we have seen that symmetric filters are required in order to achieve unity transmission on resonance. With this in mind the filters have a structure such as:



The cavity lengths are integer multiples of $\lambda_o/2$ as indicated by the terms ***a*** and ***b*** above. The symbols **m** and **2m** represent the number of Bragg pairs in a given DBR.

The filter design process consists of the following steps.

1. Decide on the materials for high and low index layers – index contrast determines stop band extent. Also high index contrast means higher reflectivity per Bragg pair. This can affect overall filter thickness to a large extent. The choice of layer materials for λ-Connect filters is discussed below.

2. Choose an initial filter design by picking some value for m in the filter expression above. In general, the number of Bragg pairs per DBR should decrease from the center of the filter towards the outer edges.

3. Model the performance of the initial design using the computer program described below.

4. Vary the number of Bragg pairs in each mirror, modeling the filter performance with each modification until the desired filter characteristics are achieved.

5. Add additional cavities to broaden the passband and improve squareness if necessary.

The central mirrors should usually have the highest reflectivities, with the reflectivity of the DBRs decreasing away from the center of the filter. This is accomplished by decreasing the number of Bragg pairs per DBR. To improve the squareness, the reflectivity of the outer mirrors generally has to be increased. This in turn has to be offset by increasing one or more of the cavity lengths (in multiples of λ/2) to preserve the flat-

topped feature. Table 3 summarizes how changes in various aspects of the filter design generally affect the filter performance (design tradeoffs). In the table, the outer mirror reflectivities refer to the reflectivities of the two outermost mirrors. The inner mirror reflectitivies refers to all of the mirrors except the outer two. The table provides rough guidelines regarding which parameters should be adjusted and whether they should be increased or decreased. We must keep in mind that changing any particular parameter will have several simultaneous effects on the filter performance.

2.5.2 Filter Modeling

The λ-Connect bandpass filters were designed generally following the guidelines indicated above. A key step in the process is to model the performance of a filter design to see if it meets the criteria for the system. The main filter modeling technique is the transfer matrix method[8] (TMM). The transmission and reflection behavior of a multilayer sequence (filter) can be calculated using the TMM.

Appendix A outlines the derivation of the transfer matrix equations from Maxwell's equations. This derivation can be found in almost every filter design text[9,10,11,12]. Briefly, the TMM relates the Electric and Magnetic fields between one side of a multilayer and the other using the boundary conditions on the electric and magnetic fields at each interface and accounting for the phase change due to propagation across each layer. This leads to a 2x2-matrix equation relating the incident, reflected and transmitted fields. The matrix that relates the fields is called the transfer matrix. Once the transfer matrix of a sequence of layers is known, the transmittance and reflectance are easily calculated as shown in appendix A. The TMM solves Maxwell's equations exactly, providing a very

powerful design tool. For the large numbers of layers required in WDM filters, the TMM is best implemented on a computer.

	Squareness	FWHM	Passband ripple	Max. Rejection
Outer Mirror Reflectivities ↑	↑	↑	↑	↑
Inner Mirror Reflectivities ↑	↓	↓	↓	↑
Cavity Lengths ↑	↓	↓	↓	↑
Number of Cavities ↑	↑	↑	**varies**	↑

Table 3. Filter design tradeoffs. Increases in outer mirror reflectivities assume that the original value is less than the inner mirror reflectivities. Increases in cavity lengths are in multiples of $\lambda/2$.

A TMM calculation program called BraggModeller was written in the Java™ programming language as a part of this dissertation research. Due to the nature of the Java language, the program is both objected oriented and cross-platform. The program consists of an input file parser, a TMM calculation module, a control and output module, and mathematical libraries. Due to the recent introduction of the Java language, mathematical libraries for matrix manipulation and complex numbers were not available at the time the program was written. These libraries were developed to facilitate the TMM calculations.

BraggModeller allows the user to specify the wavelength range of interest, the wavelength step size, the range of incident angles and angular step size, the filter center wavelength, as well as options for plotting the data as it is being calculated. The input file parser allows the user to define any number of layer types and then use these layer types to build up a sequence of multilayers. Currently support for layers of constant refractive index, layers of aluminum gallium arsenide - $Al_xGa_{1-x}As$ with aluminum fraction "x" being user selectable, and layers of indium gallium aluminum phosphide - $In_{0.5}(Ga_{1-y}Al_y)_{0.5}P$ with varying Al fraction "y", has been implemented. For the AlGaAs and InGaAlP layer types, the material dispersion is taken into account by providing a wavelength dependent index of refraction. The refractive indices are calculated on the fly based on empirical formulas[13,14,15] as opposed to using interpolation from lookup tables. Due to the object-oriented nature of the program new material types with known dispersion can be added without modifying or re-compiling the input parser. Appendix B contains class diagrams for the program as well as full source code listings for all of the classes that were developed for this program. Also, sample input files are given showing

the available options in the program. Although a simple graphical user interface was written for BraggModeller, ultimately it was simplest to use text files to configure the model and define multilayer sequences.

The source code was tested and validated by comparing model output with simple multilayer sequences whose properties could be calculated by hand, and with published filter designs[16]. The model was checked to make sure basic phenomena such as Brewster's angle and shifting of Fabry-Perot mode with changes in incident angle were correctly predicted. Once validated, the model was used to calculate the transmission and reflection performance of various filter designs. BraggModeller provides a fast and inexpensive method of testing new filter designs without resorting to costly and time-consuming iterations of filter growth – provided that the material parameters are accurately known and the filter growth technique produces multilayers with precisely controlled layer thicknesses.

Figure 12 compares calculations from BraggModeller with actual filter transmission measurements for a single FPC filter with 10 Bragg pairs per mirror and a $\lambda/2$ cavity. The high and low index materials are $Al_xGa_{1-x}As$ with $x = 0.15$ and 1.0 respectively. These materials have refractive indices of 3.5 and 3.0 respectively at λ=850nm. The filter was grown by MOCVD as described in the next section of this chapter. The full layer sequence of this filter is given in appendix C.

The model calculation agrees very well with the measured data. The FWHM and the edges of the rejection band of the filter match up quite well. On the short wavelength side, the model predicts more rejection than was measured, causing an offset in the rejection band edge compared to the measured data. The offset probably occurs because

the model calculations: rely on empirically fitted material indices, ignore loss due to scattering (walk-off), and ignore possible absorption in the semiconductor layers (which becomes a factor on the short wavelength end of the transmission spectrum as photon energy approaches the semiconductor bandgap). This mismatch is quite acceptable since the overall filter parameters (FWHM, free spectral range, S-figure) are accurately predicted despite the discrepancy.

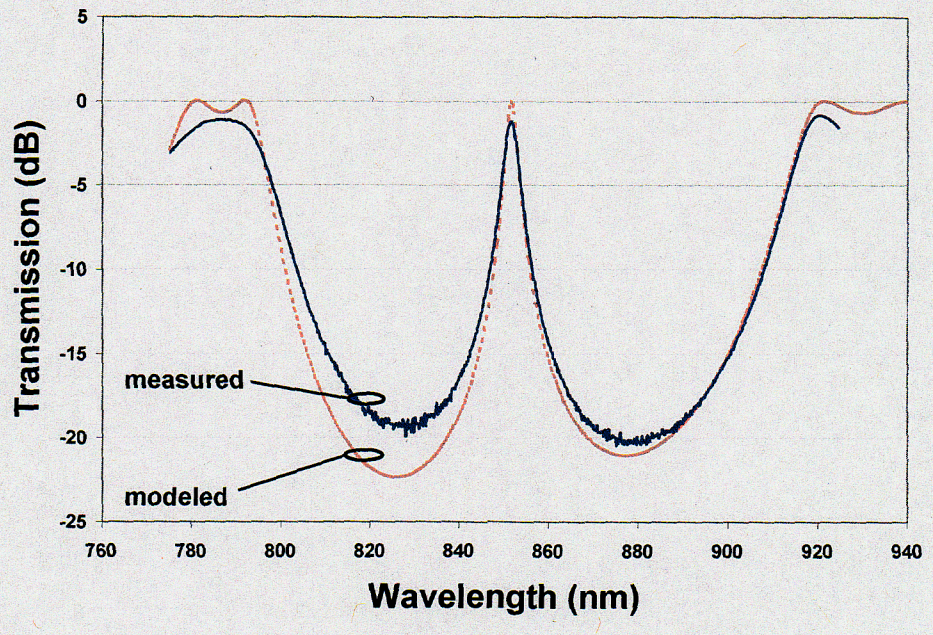


Figure 12. Comparison of filter transmission calculated by BraggModeller with actual filter results. The filter is a single cavity Fabry-Perot filter with 10 Bragg pairs per mirror. Insertion loss due to absorption and scattering are not included the model calculations.

2.5.3 Choice of Materials

Based on preliminary modeling efforts done prior to the present dissertation research[17], the decision was made to use epitaxially grown III-V semiconductor materials for the thin film filters for λ-Connect. This decision stemmed from the fact that the

numerical aperture of MMF required relatively high index films to minimize losses due to beam spreading while traversing the filter. It was found that thin films based on $Al_xGa_{1-x}As$ had suitably high refractive index (n ~3.1 – 3.5 for $\lambda \sim 820$ nm) to facilitate filter modules without micro-optic collimation lenses. Furthermore, either molecular beam epitaxy (MBE) or metalo-organic chemical vapor deposition (MOCVD) can be used to produce very high quality films with accurately controlled layer thicknesses.

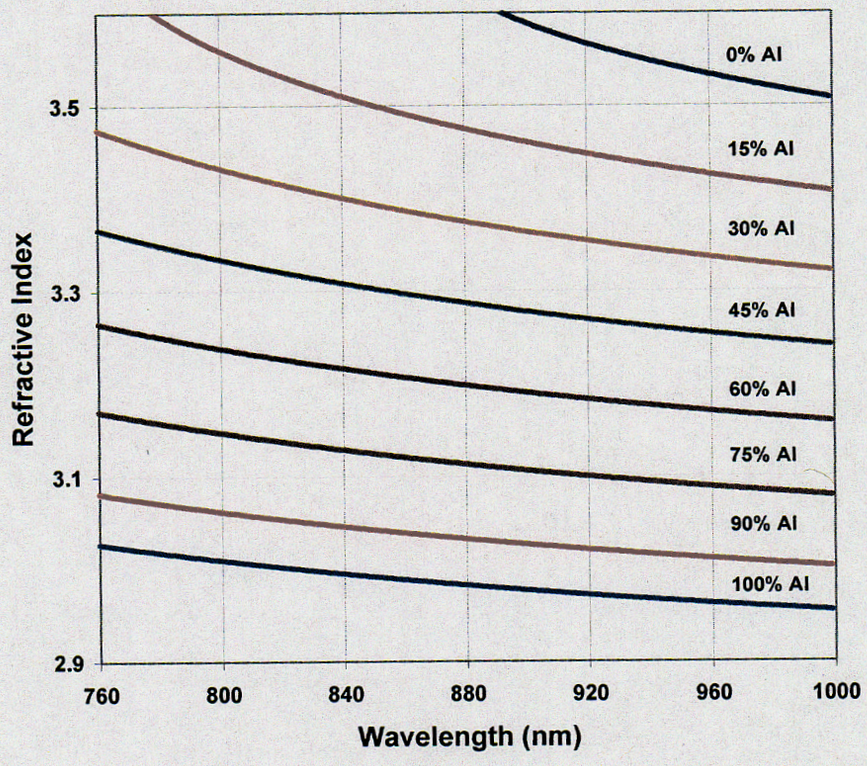


Figure 13 Refractive index of $Al_xGa_{1-x}As$ as a function of wavelength for various aluminum fractions 'x'. The refractive indices were calculated using an empirical fit to experimental data as described in reference [13], with corrections for an error in that paper [14].

By varying the aluminum mole fraction (changing "x" in the $Al_xGa_{1-x}As$) a range of refractive indices can be obtained. Figure 13 shows a plot of the refractive index of $Al_xGa_{1-x}As$ as a function of wavelength for various aluminum fractions. The plot is

based upon an empirical fit to experimental data for the refractive index of $Al_xGa_{1-x}As$ below the band gap[13] with corrections for a typographical error in that reference[14]. As seen from the figure, the refractive index increases with decreasing aluminum fraction. Using a combination of AlAs (x=1) and $Al_{0.15}Ga_{0.85}As$ an index contrast, Δn, of 0.33 is obtained (at a wavelength of 830 nm). These materials were used as the two layer materials for most of the thin film filters developed for λ-Connect.

2.5.4 Multi-cavity Bandpass Filter for λ-Connect

As mentioned in section 2.3, the goal for λ-Connect is a bandpass filter with a 5nm tolerance window, low insertion loss, and suitable for 10 nm channel spacings with less than –25 dB crosstalk between adjacent channels. Using BraggModeller and following the filter design process described in section 2.5.1 a MC-FP filter meeting these requirements was designed.

The filter has 5 coupled cavities and its structure can be visualized as in Figure 14. In this figure, the numbers denote the number of Bragg pairs in each DBR and shaded regions show the thickness (in units of the optical center wavelength λ) of the cavities. The filter is symmetric to ensure maximum transmission on resonance. Fractional Bragg pair values indicate low index spacer layers required to maintain symmetry in MC-FP designs. The high and low index layers are $Al_xGa_{1-x}As$ with x = 0.15 and 1.0 respectively. The outermost layers of the filter as well as all of the cavities are high index layers. The layer sequence for this filter is given in Appendix C.

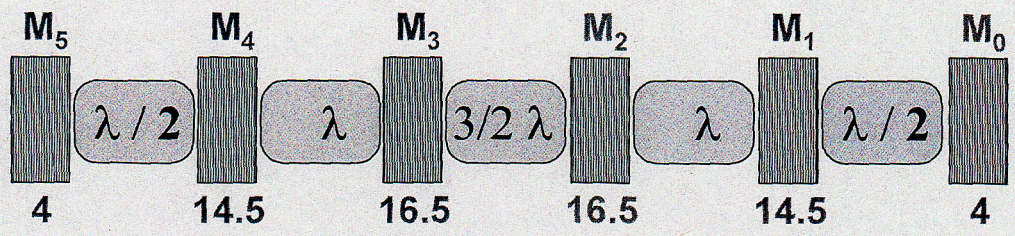


Figure 14. Layer structure of the 5-cavity MC-FP filter. Shaded regions are cavities of the indicated optical length (in terms of filter center wavelength λ). The number of Bragg pairs in each DBR is also indicated. Fractional numbers indicate extra low-index layers necessary in MC-FP designs.

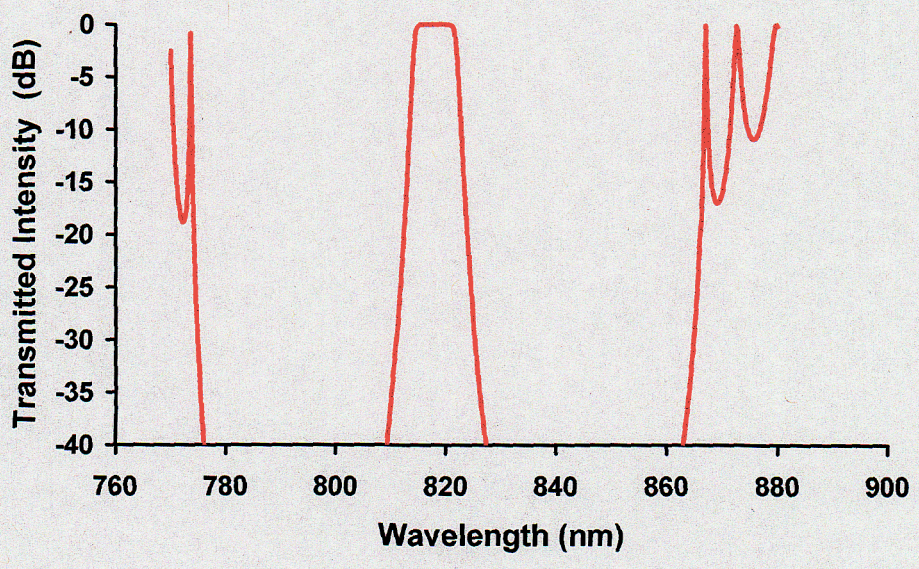


Figure 15. BraggModeller calculation of the transmission spectrum for the 5-cavity MC-FP filter designed for λ-Connect.

Figure 15 shows a BraggModeller calculation of the transmission spectrum for this filter design. Based on the calculations, this design has a FWHM of 7.8 nm, a squareness of S=0.68 and over 35 dB of cross-talk rejection. In chapter 3 we will compare these calculations with experimental results.

2.6 Summary

The simple model of coupled FPC resonators developed in this chapter illustrates the general trends of the transmission function with respect to changes in the number of cavities, the mirror reflectivities, and the lengths of the cavities. By choosing the mirror reflectivities and cavity lengths correctly, a nearly ideal filter response can be obtained. The key features of coupled resonators of interest for WDM filter applications are:

1. The ability to broaden the passband of the system.
2. The ability to improve the squareness of the passband.
3. Increased reflection outside of the passband to suppress cross-talk.

As the number of cavities in a coupled system is increased all three of these benefits can be realized simultaneously. Furthermore, using symmetric resonator designs with properly chosen reflectivities, the maximum transmission on resonance can be 100%.

The results of this analysis are applied by treating the WDM filter as a coupled FPC system consisting of DBR mirrors and (multiple) half-wave cavity layers. Starting with a base filter design and iterating using BraggModeller to predict filter performance, new filters can be designed quite readily. BraggModeller provides accurate predictions of filter performance as long as the material parameters are well known.

In order to meet the requirements of λ-Connect, filters based on epitaxially grown $Al_xGa_{1-x}As$ are used. For most of the filters designed in this work the high and low index materials were $Al_{0.15}Ga_{0.85}As$ and AlAs respectively. As described in the next chapter, the high-index ($n \sim 3.1 - 3.5$) of these materials allow for the production of packaged filter modules that have low insertion loss and do not require costly micro-optic collimation lenses.

Using the filter design process described herein, a 5-cavity MC-FP filter was designed for λ-Connect. The filter design satisfies all of the λ-Connect filter requirements as verified through BraggModeller simulations. In the next chapter we address the fabrication, packaging and characterization of actual filters based on the designs given in this chapter.

References for Chapter 2

[1] Fabry, P. MM, and Perot, A., 'Theorie Et Applications D'Une Nouvelle Methode De Spectroscopie Interferentielle', Annales De Chimie Physique, V. 16, 1899, pg. 115-144.

[2] Siegman, A. E., Lasers, University Science Books, Sausalito, California, 1986, pg. 413-425.

[3] Shakir, S. A., and Chow, W. W., 'Semiclassical Theory of Coupled Lasers', Opt. Lett. V. 9 No. 6, June 1984, pg. 262-264.

[4] Coldren, L. A., Corzime, S. W., Diode Lasers and Photonic integrated Circuits, John Wiley & Sons, New York, NY, 1991.

[5] Yariv, A., Quantum electronics, 3rd edition, John Wiley & Sons, New York, NY, 1989.

[6] Wilmsen, C., Temkin, H, Coldren, L. A., editors, Vertical-Cavity Surface-Emitting Lasers: Design, Fabrication, Characterization, and Applications, Cambridge University Press, New York, NY, 1999, pg.40, 73.

[7] Corzine, S.W., Yan, R.H., Coldren, L.A., "A Tanh Substitution Technique for the Analysis of Abrupt and Graded Interface Multilayer Dielectric Stacks" IEEE J. Quant. Electron., V27, 1991, pg. 2086-2090.

[8] Yeh, P., Optical Waves in Layered Media, John Wiley & Sons, New York, NY, 1988,

[9] MacLeod, H.A., Thin-film Optical filters, McGraw-Hill, Colorado Springs, Co. 1989.

[10] Liddell, H. M., Computer-aided Techniques for the Design of Multilayer Filters, Adam Higler, Bristol, 1981.

[11] Thelen, A. F., Design of Optical Interference Coatings, McGraw-Hill, New York, NY, 1989.

[12] Jacobson, M. R. – Editor, Selected Papers on Design of Optical Coatings, SPIE Milestone Series Volume MS 26, SPIE Optical Engineering Press, 1990.

[13] Deri, R. J., Emanuel, M. A., "Consistent Formula for the Refractive Index of AlxGa1–xAs Below the Band Edge", J. App. Phys., V77, 1995, pg. 4667-4672.

[14] Deri, R. J., personal communication.

[15] Tanaka, H., Kawamura, Y., and Asahi, H., "Refractive Indicies of $In_{0.49}Ga_{0.51-x}Al_xP$ lattice matched to GaAs", J. Appl. Phys. V 59, No. 3, 1 February, 1986, pg. 985 – 986.

[16] ibid. MacLeod

[17] Deri, R. J., λ-Connect Filter Progress Report, (unpublished), 1996

Chapter 3 Filter Fabrication, Packaging and Characterization

3.1 Introduction

In the previous chapter we discussed the general concepts of multi-cavity filter design, described the filter design process and modeling tools used for this dissertation work and presented a 5-cavity MC-FP filter design that meets the filter requirements for λ-Connect. The next step is to grow the filter and then to package and test it.

As mentioned in the previous chapter the filter approach for λ-Connect was epitaxial growth of $Al_xGa_{1-x}As$ multilayers via MOCVD. In section 3.2 we provide details of the filter fabrication.

In order to use thin film filters for parallel optical interconnect (POI) applications, they must be packaged into filter modules that protect the thin film material while allowing optical access to the filter. For λ-Connect the POI platform is based on multimode fiber (MMF) ribbon cables terminated with MT ferrules. The passive alignment provided by guide pins makes ribbon cable links extremely simple to connect and it is essential that any POI filter module preserve this feature. Therefore, our filter module approach is based on machining of commercially available MT ferrules to accommodate the thin film filters. Using this method we have developed both 2-port and 3-port filter modules in simple, robust plastic packages[1,2,3] as described in section 3.3. Section 3.4 presents results from filter module characterization.

Figure 1 shows the geometry of our filter modules. The 2-port modules employ a bandpass filter and serve as the channel selection filters at the receivers in the λ-Connect system. The 3-port filters use an edge filter and they are suitable for add/drop

multiplexing. The multi-wavelength transmitters developed for λ-Connect use this type of 3-port filter to multiplex optical signals from different VCSEL die[4].

Prior to the start of this dissertation research Deri and co-workers[1] decided upon the general filter packaging approach for λ-Connect and developed early prototypes of the 3-port filter modules. These modules used a simple DBR as the filter element. In this dissertation research, 2-port filter modules with MC-FP filters were designed and developed for channel selection at the receivers in the λ-Connect system. The wavelength router developed as part of this dissertation research (see Chapter 5) uses our 3-port filter modules for add/drop multiplexing. Therefore, we present both the 2-port and 3-port filter module work in this chapter.

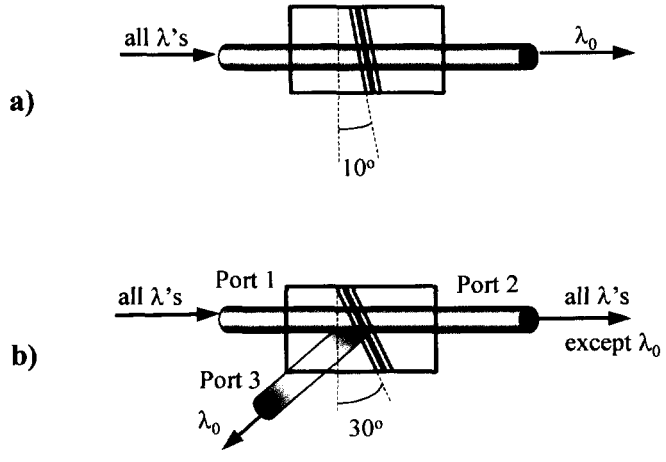


Figure 1 Schematic of a) 2-port, and b) 3-port filter modules.

3.2 Filter Fabrication

The wavelength selective element in the 3-port modules is a DBR with 31 Bragg pairs. Prior to packaging, the DBR is anti-reflection coated with Tantalum oxide and Silicon dioxide. This creates a coarse WDM add/drop filter with wavelengths in the DBR stop-band being sent to the reflected port.

In the 2-port configuration we have demonstrated both single cavity and multiple cavity Fabry-Perot filters. The single cavity filter design was described in section 2.3.3 of chapter 2 while the design of the MC-FP filter was described in section 2.5.4. Although the single cavity FP filter design does not meet the bandpass filter requirements for λ-Connect, it was used to demonstrate that packaging the filters without collimating micro-optics yields acceptable performance. As described below, the filters in the 2-port modules are mounted at 10° to increase return loss. The filter designs take this into account by adjusting all layer thicknesses to satisfy the Bragg condition at the chosen design angle.

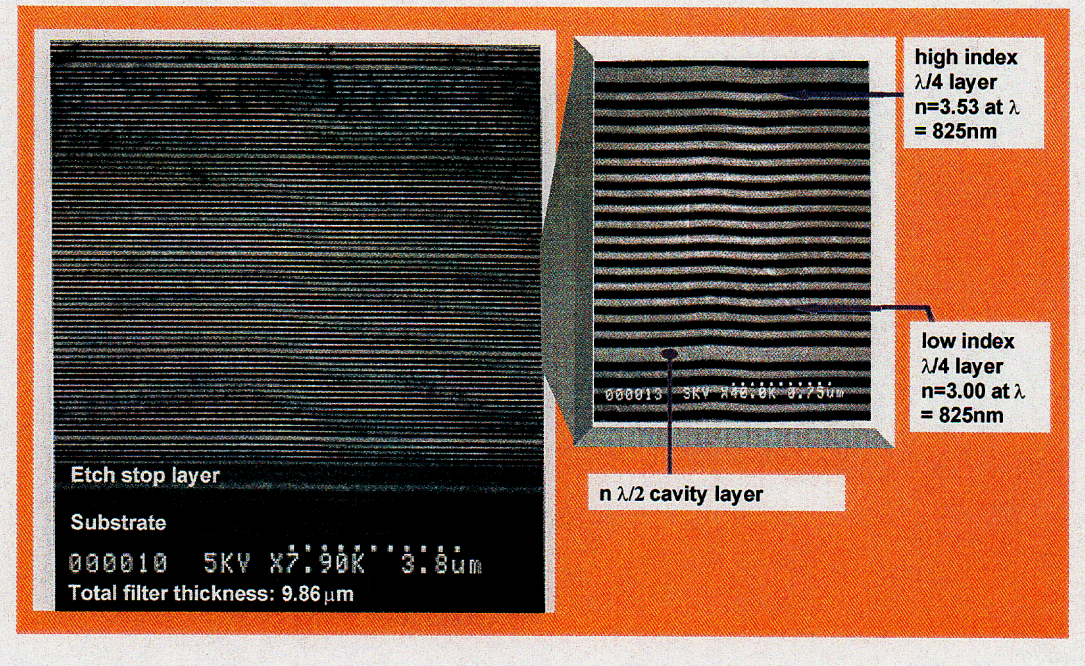


Figure 2. SEM photograph of the 5-cavity MC-FP filter. The darker layers are AlAs while lighter layers are $Al_{0.15}Ga_{0.85}As$. The InGaP etch-stop layer is also visible. Inset shows a close-up of the central portion of the filter. 16 Bragg Pairs and 1 extra low index layer are clearly seen between the 2 cavities in this image.

All of the filters use epitaxially grown $Al_{0.15}Ga_{0.85}As$ and AlAs as the high and low index materials respectively and they are all grown by metal-organic chemical vapor

deposition on GaAs substrates at 710 °C and 80 mbar. In addition to the actual filter layer structure described above, 10 nm thick GaAs cap layers were inserted on both sides of the MC-FP filter to minimize oxidation of the AlAs. A 400nm InGaP etch-stop layer, lattice matched to GaAs, is grown on the substrate prior to deposition of the filter layers. This layer facilitates substrate removal as described in section 3.3.2.A. Figure 2 shows an SEM photograph of the 5-cavity filter as grown. The alternating high and low index layers in the DBRs, the 5 cavities, and the InGaP etch stop layer are clearly visible in this figure. The figure inset is a close-up of the central portion of the filter.

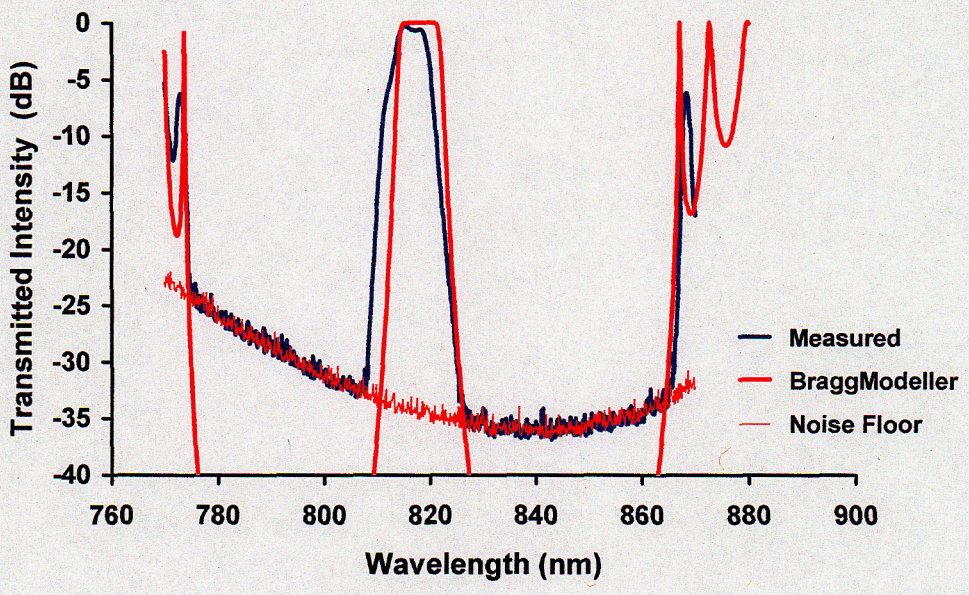


Figure 3. Measured and calculated transmission spectra for the five cavity MC-FP filter. Measured Spectra were obtained from a piece of filter material mounted on a glass slide and using collimated light. Calculated spectra are from BraggModeller simulations.

Figure 3 shows the MC-FP filter transmission as a function of wavelength. This spectrum was taken from a piece of the filter material mounted on a glass slide using collimated light. Also shown is the transmission spectrum calculated using BraggModeller (c.f. Chapter 2 Figure 15). The agreement between the measured filter

spectrum and the BraggModeller calculation is very good. The rejection band edges line up exactly while the filter passband is shifted lower in wavelength by approximately 2 nm. Also, some ripple in the passband and a minor deformation on the short wavelength side of the passband is evident. These variations from the original design indicate errors in the MOCVD growth process. The quarter-wave thickness layers of the filter call for process control better than 1%. Since the filter is nearly 10 μm thick, the deposition conditions in the growth chamber vary a great deal during a filter growth. For example, materials deposited on the chamber walls alter the vapor pressure inside the MOCVD growth chamber. This change in vapor pressure changes the deposition rate of the AlGaAs during the filter growth in a non-linear fashion. Although the grower has taken great pains to compensate for these changes, some minor variations are to be expected. Overall the process variations are quite acceptable given the complexity of the MC-FP design. In production a feedback control system with in-situ monitoring (laser reflectometry for example) can be employed to eliminate these process variations.

3.3 Filter Packaging

3.3.1 General

Figure 4 provides a perspective view of a 2-port module. The filter material is sandwiched between parallel arrays of 12 MMFs housed in the modified MT ferrules. Despite the large numerical aperture of MMF (NA = 0.275), the high refractive index of the filter enables narrow passbands and sharp passband edges without additional micro-optic collimation. Furthermore, for the 3-port modules, the use of high-index materials facilitates a large angle of incidence to ease opto-mechanical packaging while minimizing bandpass spreading and polarization sensitivity.

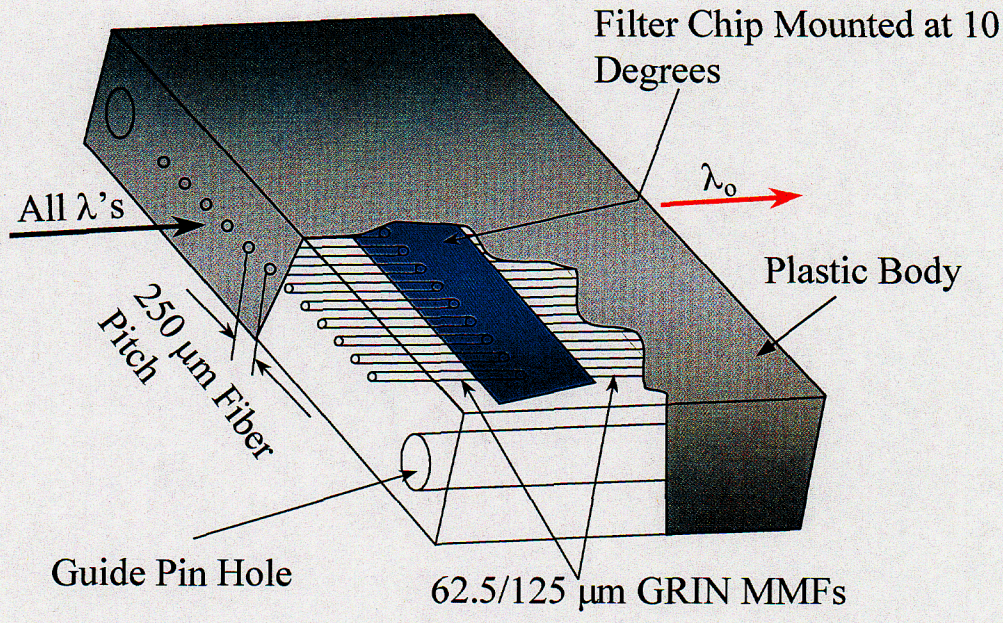


Figure 4. Perspective view of a 2-port filter module packaged to be compatible with MT ferrules.

The use of commercial ferrules minimizes device size (2.7 X 6.4 X 7.6 mm^3 for 2-port modules and 5.7 X 6.4 X 7.8 mm^3 for 3-port modules) , and simplifies assembly and packaging via passive alignment using guide pins. For the 2-port modules, alignment is 100% passive with the guide pins, while for the 3-port modules, the longitudinal alignment of port 3 along the port 1-2 axis is done by active alignment – this could be eliminated in production by using a modified ferrule design. In addition, these modules mate directly with MT-terminated fiber ribbon cables without the need for additional connectors, couplers or adapters. The minimal complexity of the assembly, coupled with the plastic molding used to realize the ferrules, indicates the potential for cost-effective manufacture of such devices.

3.3.2 Filter Module Construction

The MT ferrule provides accurate positioning of the fibers only within 2mm of the ferrule end-face. For this proof-of-principle demonstration the filter modules were constructed using the 2mm section (near the end-face) of several (typically 4) MT ferrules. The filter module construction process is as follows.

3.3.2.A 2-Port Modules

Four MT ferrule tips are cut away from the ferrule bodies using a dicing saw. By ferrule tip we refer to the ~2mm section of the MT ferrule adjacent to the end-face wherein the fiber to fiber pitch is accurately maintained at 250 μm. The ferrules are unpopulated, that is, they do not have any fibers in them. The four ferrule tips are then assembled as shown in Figure 5.

First, 2 ferrule tips are arranged such that the original end-faces are in contact. These 2 ferrule tips form the center of the filter body. Having the end-faces mate creates a ~4mm long section with precisely controlled fiber to fiber pitch. Next, the remaining two ferrule tips are arranged such that the original end-faces point away from the filter body. This ensures that the filter module can be mated with MT ferrule terminated ribbon cables using the standard guide pins of the MT system without the need for additional alignment.

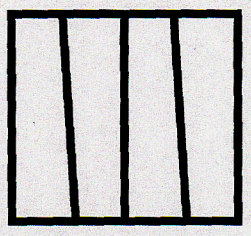


Figure 5. Arrangement of 4 MT ferrule tips in a filter module. Straight lines indicate original MT ferrule end-faces. Diagonal lines indicate end-faces exposed through dicing of the ferrules. The diagonal lines are an extreme exaggeration merely for illustrative purposes, in actuality, the end-face from the saw cut is quite straight and square.

The four filter tips, arranged as described, are mounted on a set of guide pins and populated with MMF. Twelve short sections of MMF are prepared by stripping away all protective coverings, leaving the "bare" fiber. For each of the 12 parallel channels, one strand of fiber is inserted through all four ferrule tips. Insertion of the fibers is a tedious process that is best done under microscope inspection. Typically several hours were required to manually populate one filter module. For volume production this process would be automated and would only take a few seconds.

After populating the fiber holes, the ferrule tips are epoxied together. Care must be taken to ensure that the epoxy covers all 12 fibers between each of the ferrule tips and enters into the fiber holes so that the fibers are permanently held in place. Also, each surface between ferrule tips must be covered with epoxy to bond the ferrule tips to one another. Once the epoxy has cured, a broken fiber within the filter body cannot be replaced. Therefore it is critical that a small bead of epoxy be left on each end-face of the filter body to allow for polishing of the fibers with minimal fiber breakage.

After the epoxy has fully cured, the end-faces of the filter module are polished. Polishing consists of dry lapping and/or wet lapping with a series of finer and finer grit lapping

films. Typically, a 12 μm grit film is used to remove the bulk of the epoxy bead. Next a 3 μm grit film is used. The final polishing is done with a 0.3μm grit film. Exact polishing times depend upon the size of the epoxy bead on the end-faces. Because the filter module end-faces serve as the mating surfaces with the ribbon cables, care must be taken to minimize contact of the end-faces with the lapping film during polishing. The filter module was usually held in a holder to keep the body perpendicular to the lapping surface. However, hand polishing without a holder was also employed occasionally.

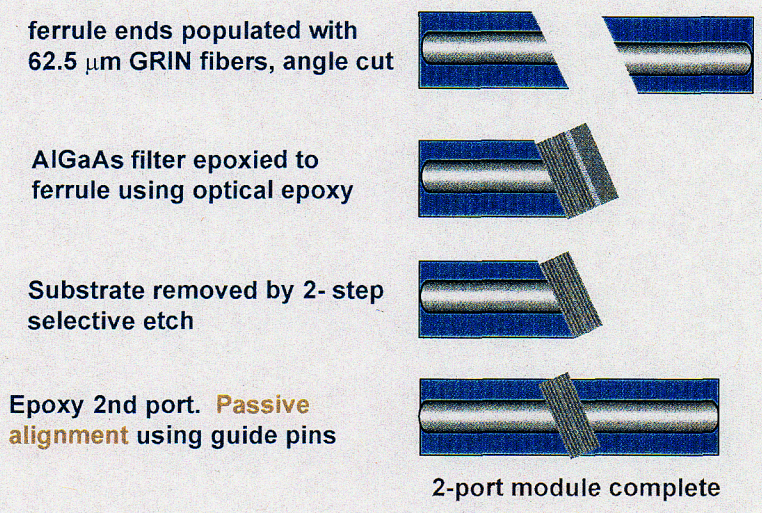


Figure 6. Summary of 2-Port Filter module construction process.

Figure 6 summarizes the major steps in the 2-port filter module construction process. Once the filter module body has been prepared as described above, it is cut in half using a dicing saw to allow for insertion of the filter material. The saw cut is made at an angle by mounting the filter module body on a holder at the desired angle (wax mounting on a prism of the appropriate apex angle was found to be adequate). For the 2-port devices, ferrule endfacets are angled at 10° to improve return loss outside the filter passband. When making the angled saw cut, a 0.3 μm grit blade is used and the lateral cutting speed

is kept around 10 mm/min. This results in a near optical polish on the newly exposed fiber facets. The two new end-faces are then given a final hand polish to clean any defects from the saw cut.

At this point, the thin film filter material is attached to one of the ferrule pieces using transparent epoxy. For this work we used Master Bond epoxy model EP30 LV. This is a 2-part epoxy with a low viscosity to ease handling and aid spreading between the small parts.

To minimize diffractive and absorptive losses in these devices, the GaAs substrate is removed using a selective wet etch consisting of H_2SO_4:H_2O_2:H_2O, in a ratio of 1:8:1 (by volume) followed by selective removal of the InGaP etch-stop layer using a 1:1 ratio of H_3PO_4:HCl. Following these etches, the final assembly step is epoxy attachment of the second half of the filter module to the first, using guide pins to passively align all 12 fibers in the module simultaneously. Figure 7 shows a fully assembled 2-port filter module. In this figure, the filter module end-face on the right hand side is slightly rounded. This rounding is caused by the end-face contacting the lapping film during fiber polishing.

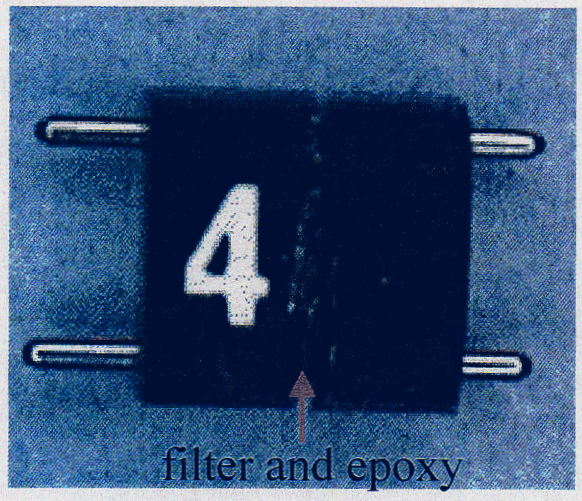


Figure 7. Photograph of a completed 2-port filter module

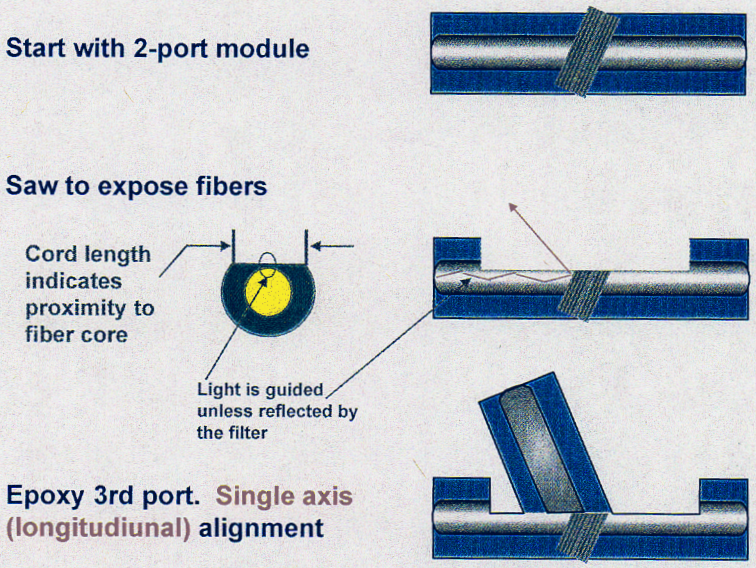


Figure 8. Summary of 3-Port Filter Module Construction Process

3.3.2.B 3-Port Modules

Figure 8 summarizes the steps in the 3-port filter module construction process. First, two sets of ferrule tips are prepared at 30° (ports 1 and 2) and the third at 60° (port 3). Once ports 1 and 2 are assembled as described above, the filter to fiber interface is exposed through a wafer saw cut from above. This allows introduction of the third port.

Care is required in controlling the depth of the wafer saw cut and in the longitudinal positioning of the third port.

The saw cut must remove enough of the fiber cladding so that, upon reflection from the filter, light is reflected up out of the plane of the incident fiber such that it can be captured by another fiber. At the same time, enough cladding must remain to provide guiding until the light encounters the filter. Therefore, the saw cut must come within a few microns of the fiber core. While the module is being cut, the cut depth is monitored through microscope inspection. The proximity to the fiber core and the cord length across the top of each fiber are related through simple geometric considerations as follows:

$$x = r_{clad} \cos\left(a\sin\left(\frac{L/2}{r_{clad}} \right) \right) - r_{core} \tag{1}$$

Here, x is the distance from the sawed surface to the edge of the fiber core. The radii of the core and cladding are r_{core} and r_{clad} respectively and L is the cord length of the saw cut. Figure 9 depicts the geometry that gives rise to equation (1).

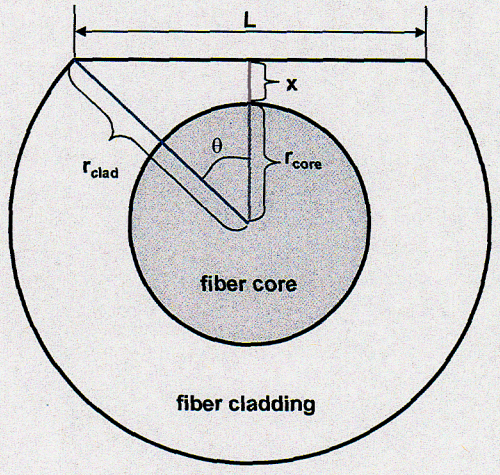


Figure 9. Geometry of the wafer saw cut. The cord length L is directly measured by microscope inspection during sawing of the 3-port filter module.

In order to get a smooth and good quality surface the dicing saw blade must be dressed to ensure that the blade's cutting surface is flat and orthogonal to the side of the blade.

Longitudinal alignment of the third port is currently done by shining light through port 3 and measuring the output from port 1 with a power meter. The third port is translated until the reading on the power meter is maximized at which point the third port is epoxied in place. A photograph of a fully assembled 3-port filter module is given in Figure 10.

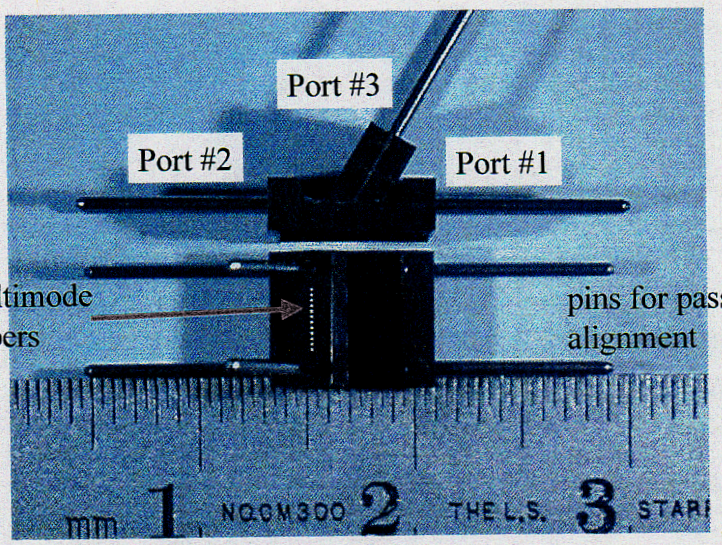


Figure 10. Photograph of a 3-port filter module (side and top views).

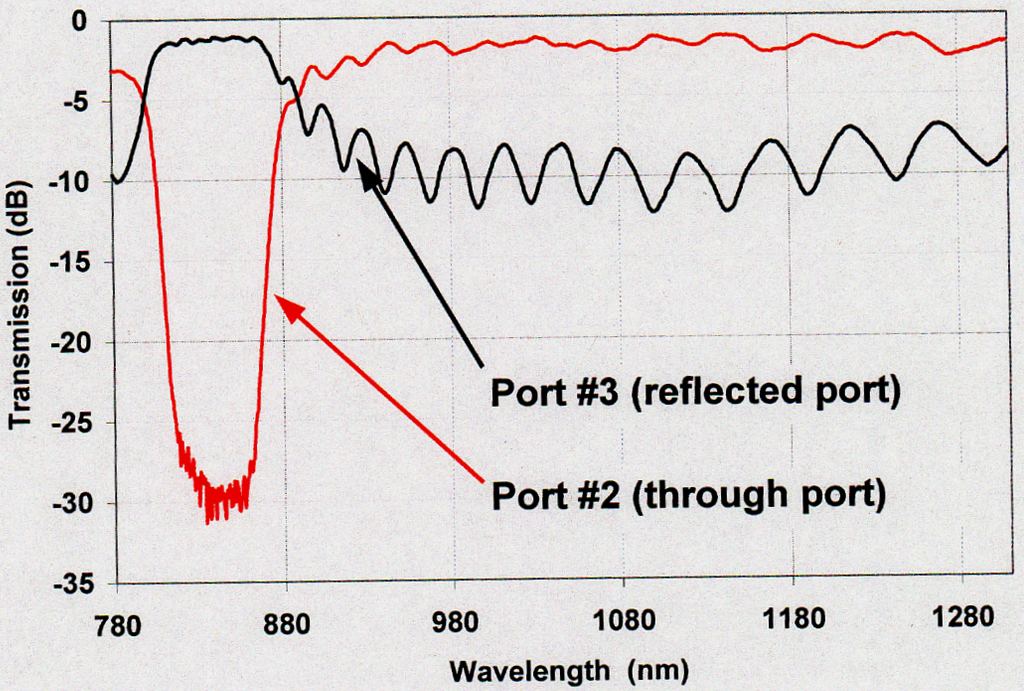


Figure 11. Transmission characteristics of a typical 3-port filter module.

3.4 Filter Module Performance

3.4.1 Transmission Spectra and Insertion Loss

The fully assembled filter modules exhibit endface-to-endface separations of approximately 40 μm due to the filter and epoxy. Figure 11 shows the transmission characteristics for a typical 3-port device and Figure 12 shows the transmission characteristics of our 2-port filter modules. These devices exhibit low fiber-to-fiber insertion loss. For the 3-port devices the insertion losses are 1.1 dB for the port 1-2 path and 1.3 dB for the port 1-3 path. It is critical that the port 1-2 insertion loss of 3-port devices be minimized since this will ultimately limit the number of filters that can be serially cascaded in ring networks. For the 2-port devices insertion losses are 1.6 and 2.6

dB for MC- and single cavity devices, respectively. The higher insertion loss for the 4nm bandpass filter is likely due to problems which occurred with the batch of optical epoxy being used at the time. The transmission spectra were obtained using an optical spectrum analyzer with illumination from a tungsten lamp coupled through ribbon cables mated to the filter modules as shown in Figure 13. An adapter cable that allowed individual fibers in the ribbon cable to be measured was used between the filter module and the optical spectrum analyzer. Insertion loss measurements were made by comparing transmitted signals to the optical throughput of the system without any filters present.

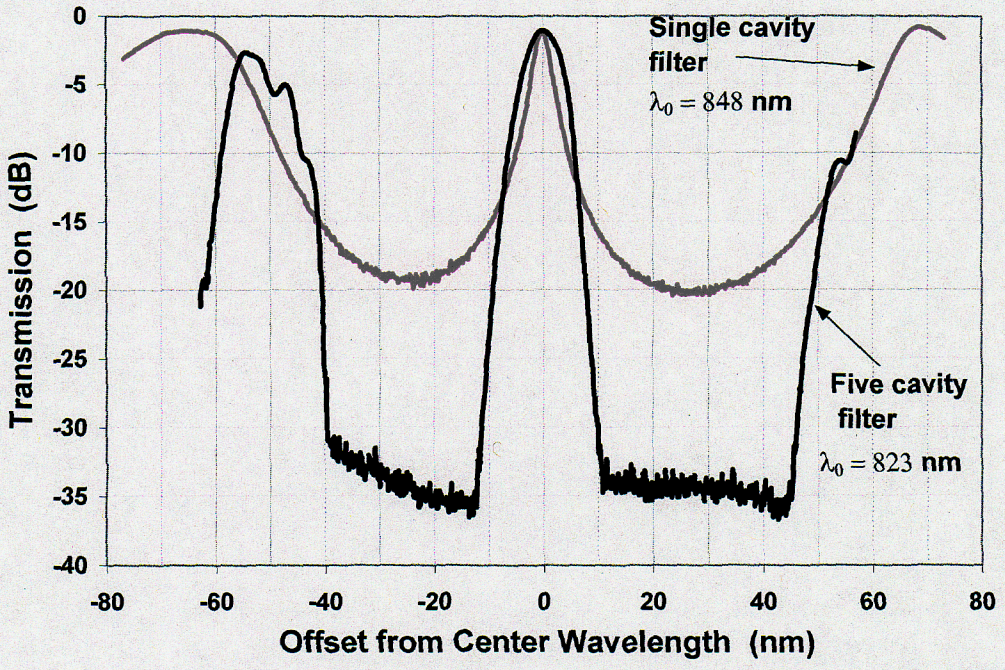


Figure 12. 2-port filter module transmission spectra plotted against offset from center wavelength to highlight passband shape differences. The actual center wavelengths are 848 nm and 823 nm for the single and multiple-cavity devices respectively.

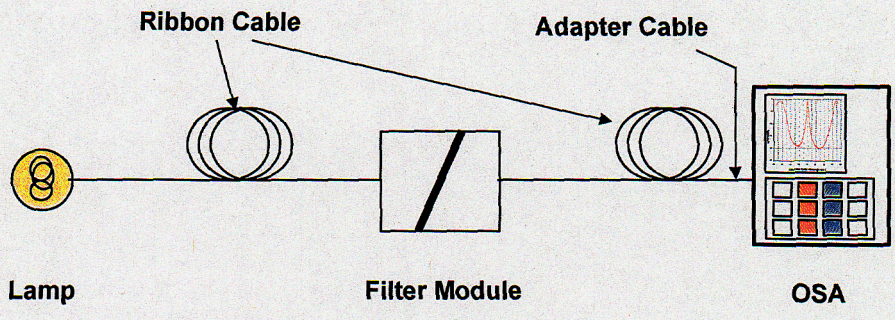


Figure 13. Experimental set-up for filter transmission spectra measurements.

The 3-port filter modules were designed to have a ~40 nm passband centered around 840 nm to be compatible with commercially available byte-wide transceivers that are manufactured to moderate wavelength tolerances. The 2-port devices exhibit narrow passbands: 7.6 nm full-width at half maximum (FWHM) for MC-FP, and 3.8 nm FWHM for single-cavity devices, respectively. Maintaining optimum filter FWHM with sharp pass-band edges is a key to achieving tightly spaced channels while allowing for transmitter drift and thermal variations. Figure 12 shows that the multi-cavity design significantly improves these features - as expected from the previous chapter. This improvement can be quantified by the squareness figure of merit (S figure). Since the single cavity filter exhibits a maximum rejection of –18 dB, we take the S figure to be the ratio of filter -3 dB bandwidth to -18 dB bandwidth in order to compare the single cavity and MC-FP filters. The S figure of the single cavity FP filter is 0.13 compared with 0.46 for the MC-FP design. Figure 12 also shows improved crosstalk suppression for the multi-cavity design compared to the single cavity filter. The 35dB suppression shown in the figure is actually a lower limit due to the noise floor of our measurement system rather than the filter characteristics. Based on this data, the MC-FP filters should provide crosstalk suppression > 23 dB for 10 nm channel spacing.

As seen in Figure 3, the MC-FP filter exhibits a flat-topped passband under collimated light. The shape of the MC-FP passband in Figure 12 is due to the fact that the filter modules do not use collimating optics. The high numerical aperture of the MMF causes the incident light to be spread through a range of angles. For FP devices the filter center wavelength shifts to shorter wavelengths with increasing angles of incidence. Therefore, at a given wavelength, light incident at angles other than the design angle is clipped by the shifted filter passband and is not fully transmitted by the filter. The net effect is a rounding of the flat-topped passband characteristic as seen in Figure 12. Nevertheless, the MC-FP filter modules exhibit a large enough FWHM to make them well suited to the needs of λ-Connect.

3.4.2 Filter Module Uniformity

For POI applications, uniform filter performance must be maintained across all the fibers within a filter module. Figure 14 shows typical uniformity results for several of our 3-port filter modules. Low insertion loss is maintained across all fibers in each module and repeatability of the assembly process is indicated by the results from more than one module. Figure 15 shows uniformity results from a typical 2-port module. The filter center wavelengths are maintained to ±0.4 nm across all fibers in the module. The average insertion loss is 1.6 dB with a variation of $\pm$ 0.6 dB. As seen in the figure, the fibers on the outsides of the filter module (numbers 1 and 12) exhibit the highest loss. A "figure-8" pattern is used during hand polishing of the filter module end-faces. This pattern tends to over-polish the outer-most fibers resulting in the increased insertion loss.

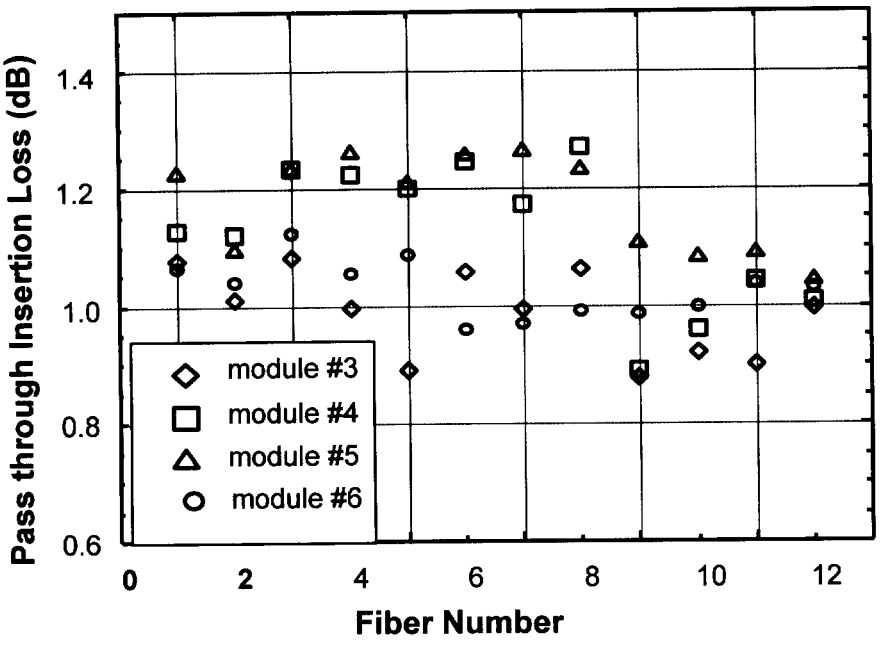


Figure 14. Uniformity data for 3-port filter modules. Insertion loss for the straight-through light path for each fiber in several 3-port filter modules is shown.

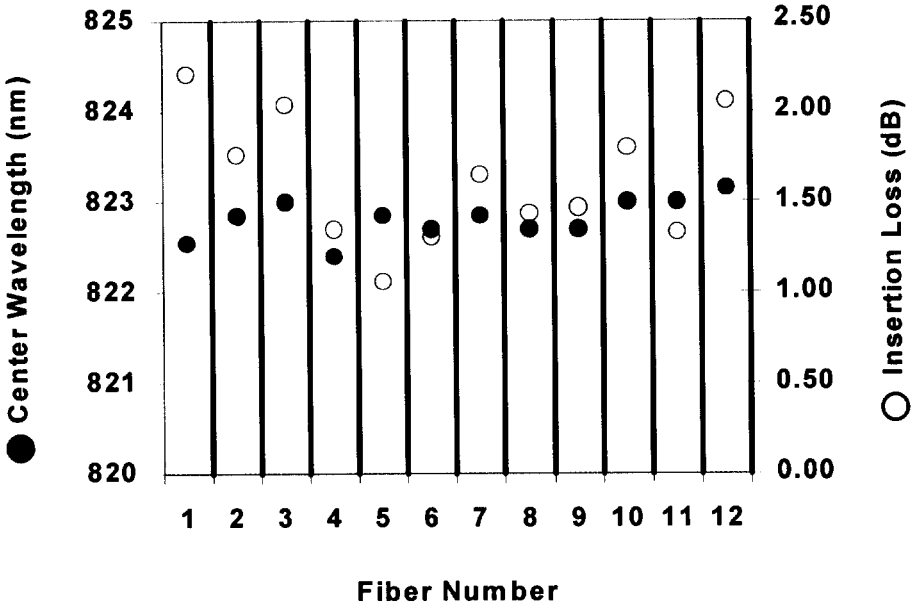


Figure 15. Fiber to fiber uniformity data for a typical 2-port MC-FP filter module. Center wavelength (solid markers) and insertion loss (open markers) for each fiber in the module are shown.

3.4.3 Cascaded 3-Port Filter Link Demonstration

We demonstrated a dual wavelength link using the 3-port filter modules and commercially available, byte-wide transceivers (Optobahn, Optobus)[5]. 2^{23}-1 pseudo-random-bit-sequence signals at ~850 and 1310nm, at bit rates of 500 and 1000 Mbit/s/fiber respectively, were transmitted through a link consisting of up to eight filters connected in series (cascaded) with a total of ~40m of 62.5μm core GRIN fiber ribbon as shown in Figure 16. This link exhibited a total insertion loss, through all 8 filters, of 10.3 dB at 1310 nm and yielded a bit error rate below 10^{-14}, including crosstalk effects between different wavelengths and different fibers in the ribbon cables. The excess power penalty for 8 filters was only 1dB. Shaking of the fiber to induce potential error rate floors had no effect, indicating that the filters introduce negligible mode selective loss.

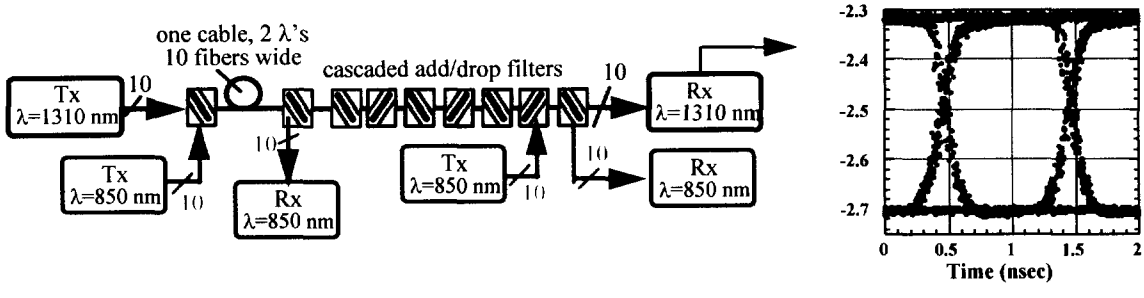


Figure 16. Schematic of 3-port filter cascade demonstration and eye diagram for transmission through 8 filters at 1Gb/s (λ = 1310nm). Filter modules were each separated by ~ 4 meters of MMF ribbon cable.

3.5 Conclusions

In conclusion, WDM filter modules for byte-wide parallel optical interconnects have been demonstrated. Filter modules suitable for add/drop multiplexing (3-port devices) and broadcast and select architectures (2-port devices) have been shown. The filter

modules use low-cost plastic ferrules to provide passive alignment, and exhibit excellent transmission characteristics and channel to channel uniformity. Using multiple cavity Fabry-Perot filter designs, the 2-port devices exhibited transmission characteristics which are suitable for 10nm channel spacing. Two-wavelength byte-wide WDM link operation was demonstrated using 3-port filter modules and commercial transceivers. Simulations indicate that our 3-port filter approach is suitable for 15-30 nm channel separations. These devices directly enable several interesting WDM interconnects such as chordal rings[6]. Furthermore, when coupled with byte-wide multi-wavelength transceivers, this technology can be used to realize WDM interconnect fabrics with substantial source-routing capability and high channel bandwidth.

References for Chapter 3

[1] Deri, R. J., et al., 'Simple Fabrication of WDM Filters for Byte-Wide Multimode Cable Interconnects', Proc. 1998 Integrated Photonics Research, p. 419 (1998).

[2] R.R. Patel, H.E. Garrett, M.A. Emanuel, M.C. Larson, M.D. Pocha, D. M. Krol, R.J. Deri, and M.E. Lowry, "Compact Low-Crosstalk, WDM Filter Elements for Multimode Ribbon Fiber Data Links", Proc., 49th Electronic Components and Technology Conference (ECTC), 1999, pg. 1261-1264.

[3] R.R. Patel, H.E. Garrett, M.A. Emanuel, M.C. Larson, M.D. Pocha, D. M. Krol, R.J. Deri, and M.E. Lowry, "WDM Filter Modules in Compact, Low-cost Plastic Packages for Byte-wide Multimode Fiber Ribbon Cable Data Links", Electronics Letters, V 35 No. 10, pg. 840 -842, May 13, 1999.

[4] Larson, M.C., Pocha, M.D., Garrett, H.E., Petersen, H.E., Bond, S.W., Drayton, R.F., Patel, R.R., Emanuel, M.A., Deri, R.J., and Lowry, M.E., "Multiwavelength VCSEL Transmitter for WDM Parallel Optical Fiber Interconnects", IEEE LEOS 12th Annual Meeting, 1999, paper ThM3, pg. 764-765.

[5] Schwartz, D. B., et al., 'A Low-cost High-Performance Optical interconnect', IEEE Trans. Components, Packaging, and Manufacturing Technology - B, V19, 1996, pg. 532-539.

[6] Arden B. W., LEE, H., "Analusis of Chordal Ring Network", IEEE Trans.Comput., V30, 1981, pg. 291-295.

Chapter 4 Lateral Oxidation of AlGaAs for WDM Filter Applications

4.1 Introduction

Thus far we have designed MC-FP filters that meet the basic performance criteria for λ-Connect and developed simple yet robust packaging that mates directly with MT terminated MMF ribbon cables. We now address two issues that can potentially enhance the performance and reduce the production cost of a λ-Connect system. First, we address methods to increase the width of the filter DBR stop-bands. As will be shown in section 4.2.1 increasing the DBR stop-band width allows bandpass filters to reject more spectral channels at a fixed channel spacing ($\Delta\lambda_{ch}$). This can allow λ-Connect to scale to higher numbers of channels while maintaining large channel spacings. Second, we describe a method to produce band-pass filters with different center wavelengths from a single epitaxially grown wafer (after the wafer is grown). Since each of our filter modules uses about 14 mm^2 of material hundreds of filters can potentially be produced from a single wafer. A method to produce filters at different center wavelengths (spaced by $\Delta\lambda_{ch}$) from a single wafer would help to increase the yield, and thus reduce the production cost, of the filter modules. Lateral oxidation of buried AlGaAs layers is proposed as a potential solution to both of these problems.

Dallesasse and co-workers first reported the conversion of AlAs (and high Aluminum fraction $Al_xGa_{1-x}As$) into mechanically stable oxides of Aluminum (Al_xO_y) via "wet" thermal oxidation.[1] They found that the conversion from semiconductor to oxide causes a dramatic change in the refractive index of the layers from n ~3 to n~1.6.

The geometry of the original experiment was such that the oxidation process proceeded vertically from the surface into the epitaxial structure. Soon after this discovery, selective oxidation of buried AlAs and $Al_xGa_{1-x}As$ layers was demonstrated by etching patterns into samples to expose the epitaxial structure. This technique is called lateral oxidation because the oxidation proceeds laterally inward from the exposed edges of the sample. The surrounding layers in this case are not oxidized by proper choice of materials. Figure 1 illustrates the geometry for both vertical and lateral oxidation.

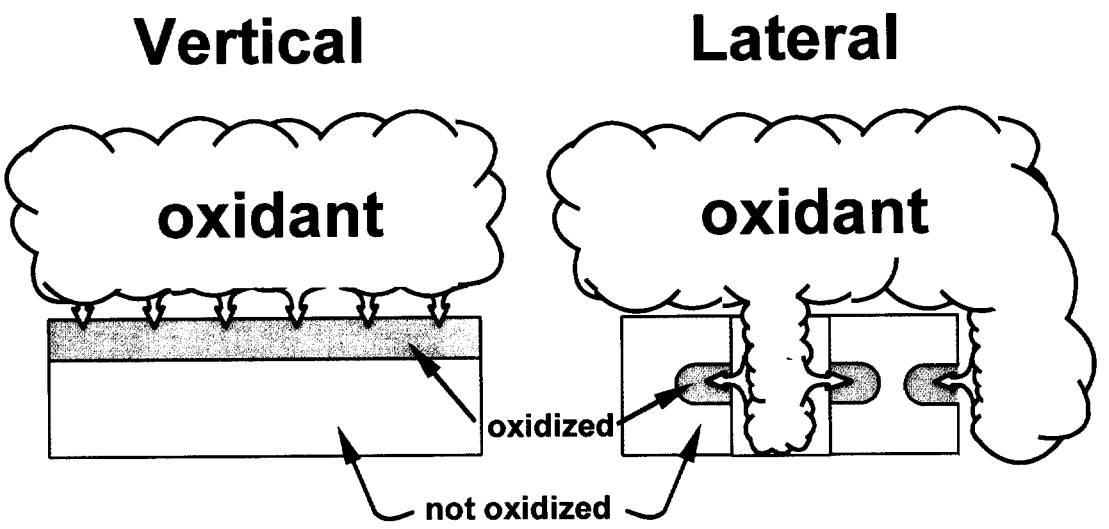


Figure 1. Geometry for Vertical and Lateral oxidation. In vertical oxidation the oxidant diffuses down into the epitaxial structure from the top surface. In lateral oxidation a buried layer is oxidized inwards from the edges of mesas or trenches patterned in the sample. The surrounding layers in this case are not oxidized by proper choice of materials.

These discoveries spurred a flurry of research aimed at characterizing the oxide[2,3,4], modeling the oxidation kinetics[5,6,7], and incorporating the oxide into opto-electronic devices[8,9,10,11]. The majority of device related research for lateral oxidation has focused on active devices such as vertical cavity surface emitting lasers. However, the ability to change the refractive index of a layer or series of layers within a multilayer also poses interesting new possibilities for passive devices as described in this chapter.

In section 4.2 we discuss in detail how lateral oxidation can be applied to WDM filters. In section 4.3 we describe the lateral oxidation process, the setup that was used for our experiments, and the implications of the lateral oxidation process for filter design. Section 4.4 addresses the design of passband shifting filters with thick tuning layers. Setion 4.5 presents a new filter structure, the oxide enhanced rejection filter, that combines benefits of using lateral oxidation with those of filters that do not employ lateral oxidation. In section 4.6 we present our experimental results and section 4.7 concludes the chapter.

4.2 Applications of Lateral Oxidation to Filters

4.2.1 Increasing DBR Stop-band Width

Figure 2 shows the filter passbands of a hypothetical WDM system with $\Delta\lambda$=15nm channel spacing. The shading in the figure indicates spectral regions where channels $\lambda_1$and λ_4 would experience unacceptable crosstalk from each other due to overlap of a DBR stop-band edge with the passband of the other channel. Only 3 non-interfering channels are possible in this hypothetical scenario.

For a given spectral range, $\Delta\lambda_{spectrum}$, and a given channel spacing, $\Delta\lambda_{ch}$, the maximum number of channels that could be deployed in a WDM system is given by:

$$m = \frac{\Delta\lambda_{specrtum}}{\Delta\lambda_{ch}} \tag{1}$$

Where the channel count, m, is rounded down to the nearest whole number.

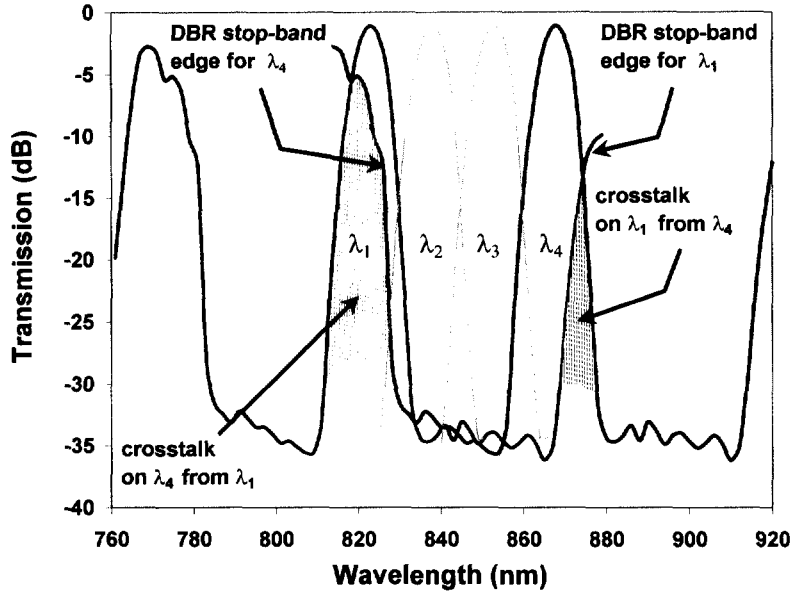


Figure 2. Limitation of system channel count due to DBR stop-band size. The spectra are inferred from the 5-cavity MC-FP filter described in chapter 3. Shading indicates spectral regions where channels λ_1 and λ_4 would experience unacceptable crosstalk from each other due to overlap of a DBR stop-band edge with the passband of the other channel. In this hypothetical system the spectral channel spacing, $\Delta\lambda_{ch}$, is 15nm.

From the example of Figure 2, we see that, for MC-FP filters with DBR mirrors, all of the spectral channels must fit within one half of the DBR stop-band width. This is most easily understood by considering the shortest wavelength channel in the system, λ_1. Since the filter passbands fall in the center of the DBR stop-bands, all of the other channels must fit between the passband edge of λ_1 and the long-wavelength stop-band edge of the DBR for that channel. Thus, for systems employing filters with DBRs, the available spectral range is given by:

$$\Delta\lambda_{spectrum} = \frac{\Delta\lambda_{DBR}}{2} \tag{2}$$

In equation (2) $\Delta\lambda_{DBR}$ is the stop-band width of the filter DBRs. For simplicity, we assume that the DBR stop-band widths are the same for all of the spectral channels.

Using equation (2) in equation (1), it is clear that $\Delta\lambda_{DBR}$ limits the scalability (number of spectral channels that can be supported) of the filters in a broadcast and select system such as λ-Connect.

In order to scale λ-Connect to a higher number of channels (8 or 16 channels is desirable), either the channel spacing must be decreased or $\Delta\lambda_{DBR}$ must be increased. The first of these two possibilities is not favorable because decreasing the channel spacing tightens the requirements on the VCSEL transmitters. This approach can lead to costly electronics and hardware to control the transmitter temperature. Since extending the DBR stop-bands can be accomplished without the need for additional electronics, it is worthwhile exploring this solution further.

One way to increase the MC-FP filter stop-band widths is to use edge filters in conjunction with the bandpass filters of the previous chapter. Essentially, each receiver would have two or more filters in series in front of the detector. The use of 2 filters comes with an increase in overall system insertion loss of 1.0 to 1.5 dB. Furthermore, the channels in the middle of the spectrum would require both long- and short- wave pass filters to block unwanted light on both sides of the spectrum. This means that channels in the center of the spectrum would experience higher insertion losses than channels on either end of the spectral range.

As mentioned in chapter 2, $\Delta\lambda_{DBR}$ scales with the ratio of index difference between the DBR materials and the average index.

$$\frac{\Delta\lambda_{DBR}}{\lambda_o} \approx \frac{4}{\pi}\cdot\frac{n_2 - n_1}{n_2 + n_1} = \frac{2}{\pi}\cdot\frac{\Delta n}{n_{avg}} \tag{3}$$

Where λ_o is the center wavelength of the DBR, Δn is the index difference between the DBR materials, and n_{avg} is the average index of refraction. Using equations (2) and (3) in equation (1) we see that the number of channels that DBR-based filters can support scales with Δn:

$$m = \frac{\lambda_o}{\pi \Delta\lambda_{ch}} \cdot \frac{\Delta n}{n_{avg}} = \frac{2}{\pi} \cdot \frac{\lambda_o}{\Delta\lambda_{ch}} \cdot \frac{\Delta n}{\Delta n + 2n_1} \tag{4}$$

Differentiating equation (4) with respect to Δn:

$$\frac{\partial m}{\partial \Delta n} = \frac{2}{\pi} \cdot \frac{\lambda_o}{\Delta\lambda_{ch}} \cdot \frac{2n_1}{(\Delta n + 2n_1)^2} \tag{5}$$

Since the derivative is positive, the maximum number of channels that can be supported increases as the index difference increases. Therefore, in order to support more channels, the index difference of the filter DBR materials must be increased.

Lateral oxidation can potentially enhance the scalability of passband filters by extending the DBR stop-bands. The large refractive index difference between $Al_{0.15}Ga_{0.85}As$ and oxides of Aluminum (Al_xO_y) results in a 400% increase in $\Delta\lambda_{DBR}$ as compared to using $Al_{0.15}Ga_{0.85}As$ and AlAs layers. The ~100 nm stop-band of Figure 2 would be increased to ~400 nm if the low index layers consisted of Al_xO_y. This hypothetical system could then support 12 optical channels with 15nm channel spacing.

4.2.2 Filter Passband Shifting

Since WDM systems employ a comb of wavelength channels with fixed channel spacing, a corresponding comb of passband filters is required in order to deploy a system. One way to generate such a comb of filters is simply to grow a series of filters with different thin film layer thicknesses corresponding to the desired filter center

wavelengths. For channel spacings of ~10nm this requires growth to growth absolute accuracy of approximately 1% (10 nm compared to ~850nm filter center wavelength = 1.2%). While layer to layer thickness control of 1% within a single growth is easily attainable, it is more difficult (and therefore more expensive) to achieve such accuracy from growth to growth using metal-organic chemical vapor deposition (MOCVD). Using molecular beam epitaxy (MBE) it is possible to count monolayers during the growth, thereby achieving the desired accuracy, however, at the present time MBE is still considered too specialized (read expensive) to use in full production.

To change the center wavelength of passband filters after growth, the optical thickness of the layer structure must be changed. For example, compressing the filter using a piezoelectric actuator can change the physical thickness of the filter layers. This approach is undesirable since the thin films are extremely fragile and would not survive very long under these conditions. The refractive index of a layer or series of layers can be altered via the electro-optic or thermo-optic effects[12]. This approach produces actively tunable filters, however; the associated electronics to effect the tuning would make the filters prohibitively expensive for data link markets. A method that irreversibly changes the refractive index of a series of layers within the filter structure is desired in order to meet the cost-performance requirements of λ-Connect.

By placing high aluminum content "tuning layers" within the multilayer, filter passbands can be shifted (after epitaxial growth) through lateral oxidation. We take advantage of the change in refractive index upon oxidation to produce filters with different center wavelengths from a single wafer. MacDougal et. al. reported a single FP cavity filter with a 6 nm shift upon oxidation[13]. In their case a single tuning layer was

sufficient since there were no resonant cavity coupling effects to consider. With proper design it should be possible to make an epitaxial structure that is a multi-cavity bandpass filter at λ_o and λ_o-$\Delta\lambda$ before and after oxidation respectively.

Consider a single interface between a high index layer and a low index layer within a filter. To shift the center wavelength a tuning layer with a high aluminum content (Al > ~0.95 mole fraction) is inserted at the interface as illustrated in Figure 3-a. Considering only the layers in the immediate vicinity of the interface, we require that prior to oxidation, the combination of layers is one-half wave optical thickness at λ_o. Furthermore, we require that the layer combination is one-half wave optical thickness at λ_o - $\Delta\lambda$ after the tuning layer is oxidized. The conditions for the shifting are described in equations (6) and (7):

$$G + n_{unox} t_{ox} = \frac{\lambda_0}{2} \tag{6}$$

$$G + \gamma n_{ox} t_{ox} = \frac{\lambda_o - \Delta\lambda}{2} \tag{7}$$

In these equations, the term G represents the optical thickness of the filter layers near the interface. The tuning layer thickness is t_{ox}. The term n_{unox} is the refractive index of the tuning layer prior to oxidation and n_{ox} is its index after oxidation. The factor γ accounts for the change in tuning layer thickness upon oxidation. For wet oxidation at temperatures above ~300 °C AlGaAs layers tend to shrink by ~6% (γ = 0.94).

The thickness of the tuning layer required for a desired wavelength shift is then:

$$t_{ox} = \frac{\Delta\lambda}{2(n_{unox} - \gamma n_{ox})} \tag{8}$$

Note that t_{ox} only depends upon the desired wavelength shift $\Delta\lambda$ and the index of the layer before and after oxidation. The details of the layers surrounding the interface do not affect the tuning layer thickness. As an example, for a 20nm shift upon oxidation with λ_o = 830mn and assuming that the tuning layer is composed of pure AlAs, the required tuning layer thickness is 6.6nm.

In order to find the thicknesses of the high and low index layers surrounding the tuning layer, we must determine the value of the factor G in equations (6) and (7). Figure 3b shows a close-up of a tuning layer along with the surrounding layers within a DBR. As indicated in the figure, the thickness of the high and low index layers is reduced as compared to the original DBR in order to accommodate the tuning layers within the $\lambda_o/2$ Bragg period. The right most portion of the figure illustrates the $\lambda_o/2$ region in the immediate vicinity of a single tuning layer. From this we see that the factor G is given by:

$$G = n_{unox}\frac{t_{ox}}{2} + n_H t_{H^*} + n_L t_{L^*} + n_{unox}\frac{t_{ox}}{2} \tag{9}$$

In equation (9) the terms t_{H^*} and t_{L^*} represent the reduced thicknesses of the high and low index layers in the DBR with tuning layers. Using equations (8) and (9) in either equation (6) or (7) we are left with one equation and 2 unknowns (the thicknesses of the high and low index layers).

To solve for the high and low index layer thicknesses we are free to choose any condition that involves one or both of the layer thicknesses. For example, we might require high and low index layers to have equal optical thicknesses:

$$n_H t_{H^*} = n_L t_{L^*} \tag{10}$$

Another example would be to have the high and low index layers have the same physical thicknesses. Many other conditions could be chosen if desired. For simplicity, we shall restrict ourselves to the condition of equal optical thicknesses as given in equation (10).

Equations (6) through (10) illustrate the basic concept of shifting the resonant wavelength via lateral oxidation. The design method is to preserve the phase relationships of the reflected and transmitted electric fields at λ_o and $\lambda_o - \Delta\lambda$ before and after oxidation respectively. These equations allow the design of a Bragg "pair" that includes a thin tuning layer.

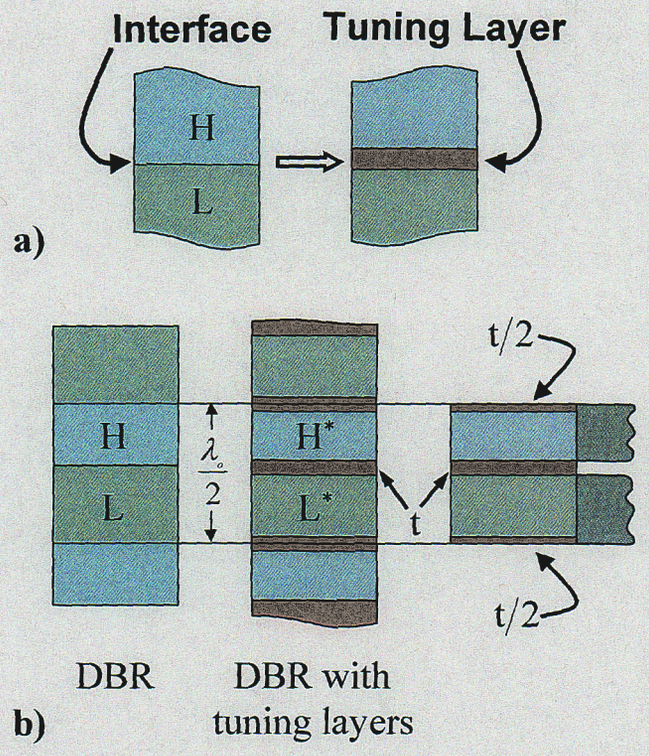


Figure 3. Placement of passband shifting tuning layers.
a) A single tuning layer is placed at the high/low interface.
b) Within a DBR, tuning layers are placed between each high/low index interface. Compared to the original DBR, the thickness of the high and low index layers is reduced to accommodate the tuning layers. The shaded region on the right-most portion of the figure illustrates the term 'G' from equations 6 and 7.

4.3 Lateral Oxidation Setup and Process

Choquette et. al. have published an excellent review article that describes the oxidation process in great detail[14]. Here we briefly mention only those factors that are relevant to understand the experiments carried out as part of this dissertation research.

Figure 4 shows a schematic of the lateral oxidation system used for the experiments described in this chapter. The basic process is to expose high Al-content layers to water vapor at elevated temperatures (350 °C – 500°C). An inert gas (in our case Nitrogen) is bubbled through a heated water bath (the bubbler) generating water vapor. The vapor and gas are transported to the furnace through a line that is heated to avoid condensation. Flowing the N_2 carrier gas and steam through the tube equilibrates the furnace prior to inserting samples for oxidation. Samples to be oxidized are placed on a pre-heated quartz plate inside the 2-inch diameter tube furnace.

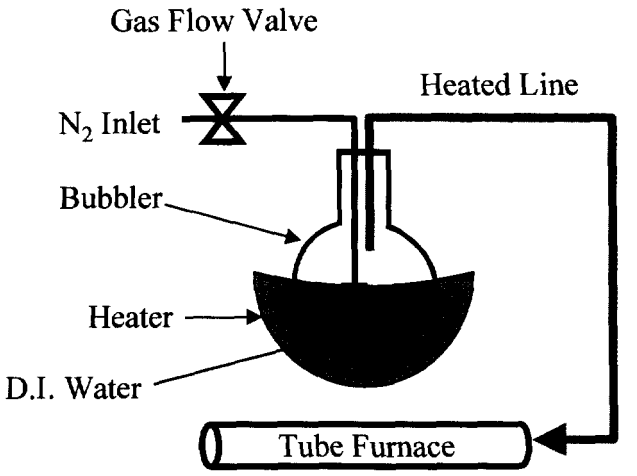


Figure 4. Schematic of the lateral oxidation set-up. Samples are heated to 350°C – 500°C in the tube furnace. Water vapor carried by an inert gas is responsible of the conversion from semiconductor to oxide.

Following the Deal and Grove model (for oxidation of silicon) [15], the oxidation process for $Al_xGa_{1-x}As$ is often modeled as a diffusion-reaction process. The steam in the furnace must diffuse across the region where oxidation has already taken place to reach the oxide/semiconductor interface (see Figure 1). At the interface, the steam chemically reacts with the $Al_xGa_{1-x}As$ to form the Al_xO_y. The oxidation rate is either linear or parabolic with time depending on whether the process is reaction rate limited or diffusion limited respectively.

The oxidation rate is affected by:

1. The Aluminum fraction of the layers being oxidized
2. The oxidation conditions (furnace and bubbler temperatures, gas flow rate)
3. The thickness of the layer(s)
4. The composition of adjacent layers.

The oxidation rate is strongly dependent upon the Al-fraction of the layers being oxidized. For example, it has been reported[14] that, at 420°C, 100nm thick layers of AlAs and $Al_{0.84}Ga_{0.16}As$ oxidized at a rate of 3 μm/min and 0.02 μm/min respectively. Also, in the 350 – 500 °C temperature range, very low Al content layers ($x \leq \sim 0.3$) are not oxidized at all. Therefore, surrounding a high Al fraction layer with GaAs or low Al-fraction $Al_xGa_{1-x}As$ layers allows the creation of a buried oxide layer.

The temperature dependence of the oxidation process is modeled as an Arrhenius relationship of the form $R(T) = R_o\exp(-E_a/KT)$. Here R(T) represents the oxidation rate as a function of temperature, E_a is an activation energy, and K is Boltzman's constant. The diffusion and reaction terms of the oxidation process have different activation energies and rate constants. In general the oxidation rate increases as the temperature increases in the 350 to 500 °C range.

The bubbler temperature and gas flow rate affect the concentration of steam in the ambient environment of the furnace. It has been reported that, for bubbler temperatures of 60 °C and 90 °C, carrier gas flow rates greater than 3.0 and 1.5 liters per minute respectively provide a vapor saturated ambient environment in a 4-inch diameter furnace[14].

Buried layers thinner than ~10nm apparently cannot be oxidized[16]. This is partly due to the fact that the grain size of the amorphous Al_xO_y is approximately 8nm[17]. Furthermore, for layers thinner than ~50nm the curvature of the oxidation front results in an increased activation energy for the oxidation reaction[18]. This rise in activation energy results in a decrease in oxidation rate with decreasing layer thickness.

The composition of the layers adjacent to the layer to be oxidized (oxidation layer) also affects the oxidation rate. The oxidation layer serves as a source of oxidant for the adjacent layers. Therefore, if the surrounding layers have a high Al-fraction they will oxidize vertically. This vertical oxidation effectively enhances the lateral oxidation rate of the surrounding layers. Furthermore, the vertical oxidation depletes the supply of oxidant reaching the oxide/semiconductor interface in the oxidation layer.

Due to the strong interest in using lateral oxidation to define current apertures in VCSELs, most studies of lateral oxidation to date have used samples with mesas etched into the epitaxial structure. The oxidation proceeds inwards from edges towards the centers of the mesas. VCSEL mesas tend to be much smaller (< ~25 μm diameter) than the core of multimode fibers. Furthermore, the fibers in our POI filter modules occupy an area of ~1 mm^2. Therefore an oxidation process that can cover areas much larger than what has been explored to date is needed. To address this issue a mesh of holes in a triangular lattice was used to pattern our samples. The close-packed nature of this geometry minimizes the distance that the oxidant must travel between holes to fully cover a given area. This approach allows us to laterally oxidize over large areas while preserving the maximum amount of filter material.

The holes patterned into the filters will introduce crosstalk due to lack of filtering. We can estimate the crosstalk introduced by taking the ratio of the area of the holes to the area covered by filter material. For example, the crosstalk due to 2- and 4-μm diameter holes with a lattice constant of 30μm is approximately –24 dB and –18 dB respectively. Thus, if the area of the holes is small compared to the area covered by filter material the crosstalk should be within acceptable limits.

4.3.1 Implications of Lateral Oxidation for Filter Design

As seen in the previous section, the lateral oxidation process is affected by a number of parameters including the composition of the layer to be oxidized, the composition of the surrounding layers, and the oxidation conditions. These parameters must be accounted for when trying to use lateral oxidation in WDM filters.

The first implication of the lateral oxidation process relates to the thickness of tuning layers for passband shifting filters. As mentioned above, layers thinner than ~10nm cannot be laterally oxidized. This means that the 6.6nm tuning layer thickness calculated previously is not a practical design for lateral oxidation. Although the BraggModeller calculations indicate that the approach of equations (6) – (8) works in theory, for small passband shifts ($\Delta\lambda < 40$ nm) it results in filter designs that cannot be oxidized because the tuning layers are too thin. In order to achieve smaller passband shifts in practice a filter design with thicker tuning layers (for a given $\Delta\lambda$) is required. The design of passband shifting filters with thicker tuning layers is discussed in the next section.

The second implication of the lateral oxidation process relates to the composition of the layers adjacent to the tuning layers in passband shifting filter designs. We recall that the tuning layers are always placed at the interface between a high and low index layer. The passband shifting filter calls for the tuning layer to oxidize without affecting the surrounding layers. Since low index layers in the $Al_xGa_{1-x}As$ material system have a high aluminum fraction, these layers can experience enhanced oxidation due to an adjacent tuning layer. Therefore passband shifting filter designs must mitigate the enhanced oxidation rate of the low index layers. The approach followed in this work was to use AlAs for the tuning layers and AlGaAs with 15% and 85% aluminum fraction for the

high and low index layers respectively. To prevent vertical oxidation thin barrier layers were placed between tuning layers and adjacent low index layers as described in more detail in section 4.6.4.

Finally, the high reflectivity per Bragg pair (~42%) when using $Al_{0.15}Ga_{0.85}As$ and aluminum oxide can limit ability to meet filter design goals. The reflectivity of a DBR increases by a huge amount every time a Bragg pair is added. This huge change in reflectivity does not allow for fine-tuning of filter designs. We have designed a new filter structure that overcomes this limitation as described in section 4.5.

4.4 Passband Shifting Filter Designs with Thick Tuning Layers

Equations (6) and (7) can be generalized as follows:

$$G + n_{unox} t_{ox} = m \cdot \frac{\lambda_0}{2} \tag{11}$$

$$G + \gamma m_{ox} t_{ox} = k \cdot \frac{\lambda_o - \Delta\lambda}{2} \tag{12}$$

where the factors m and k are integers. The most general form of equation (8) is then:

$$t_{ox} = \frac{(m-k)\lambda_o + k\Delta\lambda}{2(n_{unox} - \gamma m_{ox})} \tag{13}$$

Clearly if m and k take on values other than unity the tuning layer thickness for a given passband shift is increased. For simplicity, the case of $m \neq k$ is ignored since it has little practical use. However, even with m = k, the tuning layer thickness can be increased.

There are 2 ways to make m and k greater than 1. The first method is to use DBRs with higher order periods. For example, instead of $\lambda_o/4$, each layer could be made $3\lambda_o/4$ in optical thickness. However, higher order periods decrease the DBR stop-band width

$\Delta\lambda_{DBR}$ compared to $\lambda_o/4$ layers. This is undesirable for the filter scalability reasons as discussed previously.

The second way to make m and k greater than 1 is to use fewer tuning layers spaced further apart within the DBR. Effectively, each tuning layer shifts several Bragg pairs instead of just one pair. The approach of equations (6) through (10) preserves the phase relationship of the fields within each $\lambda/2$ interval in the DBR. To make filters with thicker tuning layers we use fewer tuning layers and let each one preserve the phase relationship over a $k\lambda/2$ interval in the DBR. Clearly, in order to maintain the advantages of the multi-cavity filter, removing certain tuning layers and increasing the thickness of others must be done in a manner that still preserves the overall phase relationships across the filter. The concept of equivalent layers provides some guidance as to which tuning layers should be removed.

4.4.1 Equivalent Filter Concept

As noted by Epstein[19], the characteristic matrix of a symmetric sequence of thin film layers has the same functional form as a single layer. Therefore, symmetric layer sequences can be (mathematically) treated as a single **equivalent layer** having some equivalent index N_{eq} and equivalent thickness Γ_{eq}. In general, the equivalent index can be much higher (or lower) than the refractive indices of the individual layers. The equivalent thickness can also be different from the sum of the optical thicknesses of the individual layers. For our purposes, the details of how to calculate the equivalent index and equivalent thickness are not important (see Thelen[20] for these details). We note that, since they have the same mathematical form as individual layers, the approach of

preserving the optical thickness at λ_o and λ_o-$\Delta\lambda$ before and after oxidation respectively can be applied to these equivalent layers.

4.4.2 Design Process

Passband shifting filters with thicker tuning layers can be designed as follows:

1. Design a normal passband filter following the procedure of chapter 2.
2. Group the layers of the filter DBRs into symmetric layer sequences.
3. Convert the symmetric sequences into equivalent layers resulting in an **equivalent filter**.
4. Design tuning layers for the equivalent filter using equations (11) through (13).
5. Place the tuning layers in the original layer sequence at interfaces corresponding to the interfaces of the equivalent filter. Adjust the thicknesses of the all high and low index layers in each DBR. The thickness of ($m\lambda_o/2$) cavity layers is not changed.
6. Adjust the thickness of tuning layers immediately adjacent to cavity layers (on both sides) to be $t_{ox}/2$ in order to preserve phase relationships.

To illustrate step 2 of this procedure we consider a DBR with 6 Bragg pairs. The layer sequence can be represented as:

$$\mathbf{H\,L\,H\,L\,H\,L\,H\,L\,H\,L\,H\,L} \tag{14}$$

Where, as usual, the letters H and L represent quarter wave layers of high and low index respectively. Starting from the left, the first 3 layers form the symmetric sequence HLH (≡A). The next 3 layers form the symmetric sequence LHL (≡B). Treating these two layer sequences as equivalent layers an equivalent DBR for (14) can be expressed as:

$$\mathbf{A\,B\,A\,B} \tag{15}$$

Thus 2 pairs of equivalent layers replace the 6 Bragg pairs of the original DBR.

Figure 5 illustrates the equivalent filter concept of step 3 applied to a 2-cavity filter design. As above, 2 pairs of equivalent layers replace the 6 Bragg pairs of the outer DBRs. Five pairs of equivalent layers replace the coupling mirror and the cavity layers are unchanged.

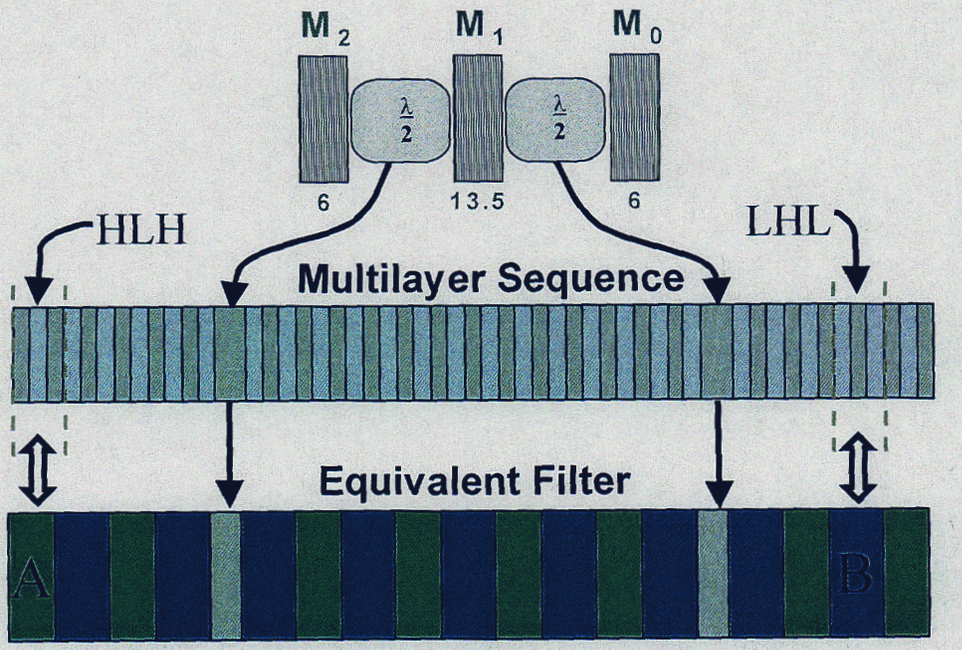


Figure 5. Equivalent filter concept. The layer sequence of a 2-cavity filter is shown along with an equivalent filter design that mathematically treats the 3-layer sequences HLH and LHL as single layers.

For step 4 we note that the equivalent layers A and B each represent 3 quarter-wave layers; therefore, the factor k in equations (11) through (13) is equal to 3. Thus, the tuning layer thickness required to achieve a passband shift of $\Delta\lambda$ is:

$$t_{ox} = \frac{3}{2} \cdot \frac{\Delta\lambda}{(n_{unox} - \gamma n_{ox})} \tag{16}$$

In step 5 the original filter design is modified by placing tuning layers at interfaces corresponding to the interfaces in the equivalent filter. Following the same logic used in arriving at equation (9), the factor G for the equivalent filter is given by:

$$G = n_{unox}\frac{t_{ox}}{2} + n_A t_{A*} + n_B t_{B*} + n_{unox}\frac{t_{ox}}{2} \qquad (17)$$

where the subscripts A and B refer to the equivalent layers. The terms t_{A*} and t_{B*} indicate that the optical thicknesses of the equivalent layers are reduced when the tuning layers are inserted. Recall that this thickness reduction occurs because the layers must be made physically thinner to make room for the tuning layers within the $3\lambda_o/2$ optical thickness of the sequence AB in the original DBR.

Strictly speaking, equation (17) is the correct approach for calculating the required layer thicknesses. The expressions relating the equivalent thicknesses t_{A*}, t_{B*} to the actual layer thicknesses t_H and t_L were first published by Ohmer[21]. These expressions are quite cumbersome and there is little point in taking such a rigorous approach because our estimated value of the index of Al_xO_y would limit the precision of the calculations. Rather, we note that, in the original filter design, the high and low index layers have the same optical thickness. In finding the reduced layer thicknesses we shall require that this condition still be satisfied.

We assume that G can be expressed as a simple sum of the optical thicknesses of all of the layers within the equivalent layers A and B.

$$G = n_{unox}t_{ox} + 3n_H t_{H*} + 3n_L t_{L*} \qquad (18)$$

here the subscripts H and L refer to the high and low index layers respectively and the asterisks remind us that we are seeking the reduced layer thicknesses. The factors of 3 in (18) stem from the fact that there are a total of 3 high index layers and 3 low index layers

in each AB equivalent layer pair. The condition of equal optical thicknesses is given in equation (10). Using equations (10), (11), (16) and (18) the thickness of all the layers can be found.

The rule of step 6 is understood by recalling that the $m\lambda_o/2$ cavity layers have no net effect on the phase of the fields when $\lambda=\lambda_o$. Furthermore, there is always a tuning layer on both sides of each cavity. This means that the tuning layers immediately adjacent to the cavity layers have the combined effect of a single tuning layer with double the thickness. Therefore, in order to achieve a desired $\Delta\lambda$ the thickness of the tuning layers immediately adjacent to the cavities must be $t_{ox}/2$.

The base filter design of Figure 5 (without tuning layers) was re-designed using the process described above. The tuning layer thickness for a passband shift of $\Delta\lambda$=20nm was calculated to be 20.03nm. The optical thickness of the high and low index layers was reduced from 0.25 waves ($\lambda_o/4$) to 0.226 waves to accommodate the tuning layers. Figure 6 shows the layer structure of the re-designed filter DBRs. Figure 7 shows BraggModeller calculations of the transmission spectra for this filter design before and after oxidation. Upon oxidation the passband shifts ~21nm and the FWHM of the filter is slightly reduced as seen in the figure.

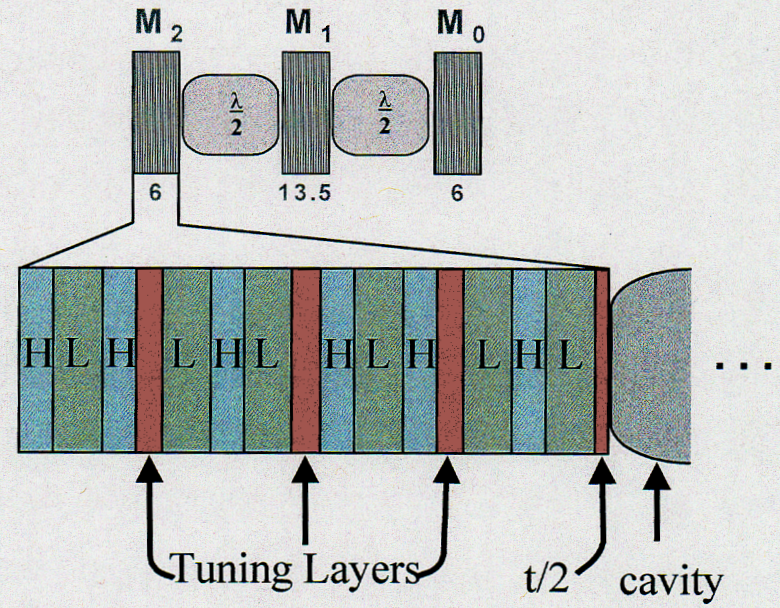


Figure 6. Layer structure of the passband shifting filter DBRs with thicker tuning layers.

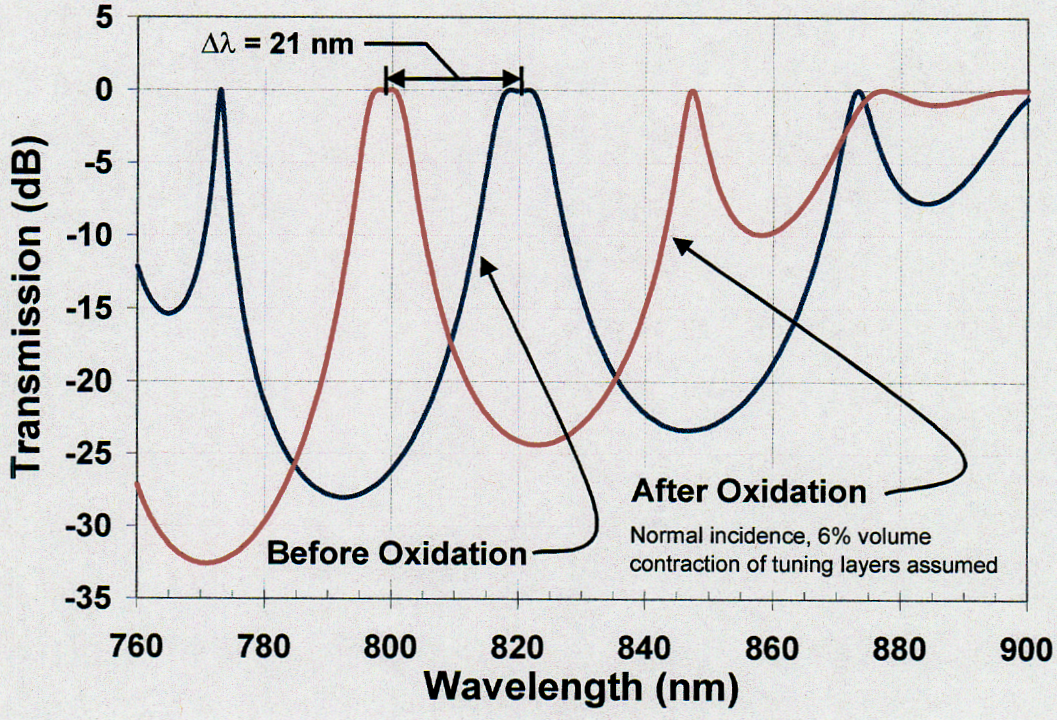


Figure 7. A BraggModeller calculation of transmission vs. Wavelength before and after oxidation for the 2-cavity filter with thicker tuning layers.

4.5 Oxide Enhanced Rejection Filter

The high reflectance (~42%) per Bragg pair of $Al_{0.15}Ga_{0.85}As$ and Al_xO_y (Aluminum oxide) presents a limitation when trying to meet filter design goals. As seen in Figure 8, every time a single Bragg pair is added to a DBR the reflectivity changes by a huge amount when using Aluminum oxide for the low index layers. In contrast, the reflectivity varies much more slowly as a function of the number of Bragg pairs for DBRs using AlAs low index layers. As discussed in chapter 2, tailoring the FWHM, and squareness while maintaining flat passbands calls for the addition of more and more cavities to a filter design with the requirement that the mirrors in the center of the filter have the highest reflectivities. When $Al_xGa_{1-x}As$ with low aluminum fractions are used for the high index layers, use of AlAs for the low index layers allows greater flexibility in filter design compared to using Aluminum oxide. However, the large DBR stop bands that are possible when using Aluminum oxide facilitate higher system channel counts as described previously.

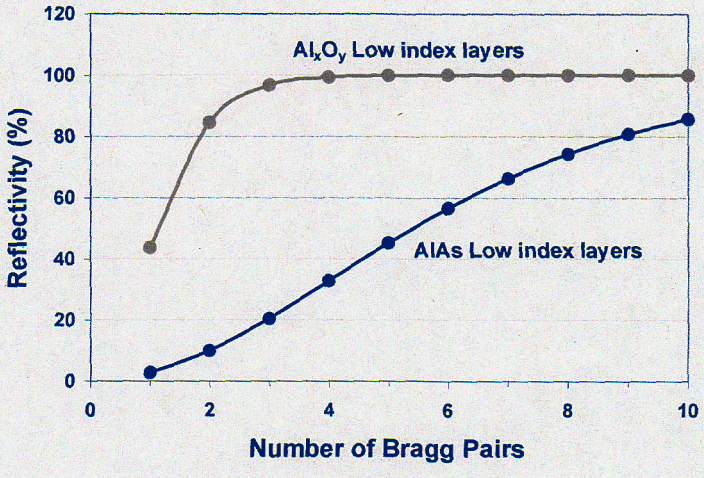


Figure 8. DBR reflectivity as a function of the number of Bragg pairs for AlAs and Al_xO_y low index layers with $Al_{0.15}Ga_{0.85}As$ high index layers. For more than 3 Brag pairs the reflectivity of the Al_xO_y DBR is greater than 99.99 %.

Ideally, a filter that combines the large stop-band of the $Al_xO_y/Al_{0.15}Ga_{0.85}As$ material pair with the design flexibility afforded by AlAs low index layers is desired. In order to meet these two goals we propose a new filter structure called the Oxide-Enhanced-Rejection filter (OER).

Figure 9 illustrates the concept of the OER filter. The device consists of a very wide-passband MC-FP filter using Aluminum oxide and $Al_{0.15}Ga_{0.85}As$ grown on top of a narrow passband MC-FP filter using AlAs and $Al_{0.15}Ga_{0.85}As$. The filter with the aluminum oxide layers (the oxide filter) provides large reject bands while the filter with AlAs layers (the AlAs filter) provides passband design flexibility. The concept of the OER filter is similar to the idea of cascading thick and thin etalons to achieve narrow passbands with large free spectral ranges.

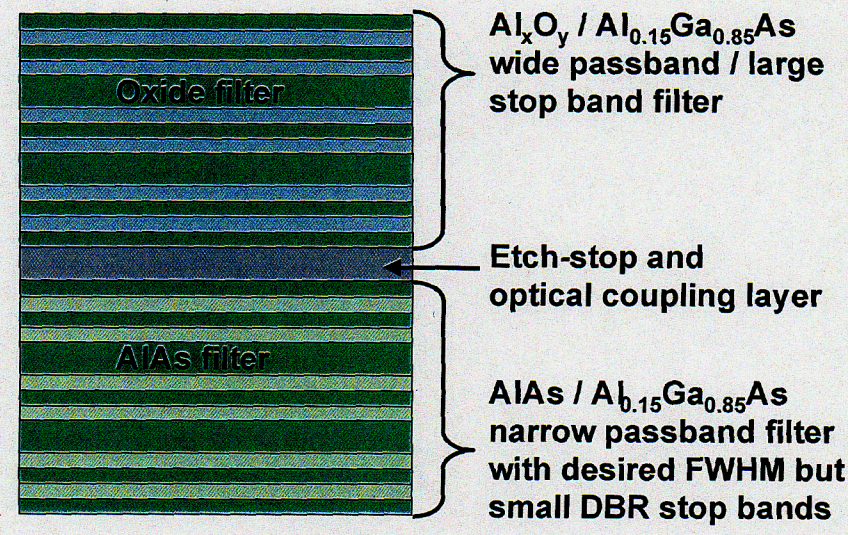


Figure 9. Oxide Enhanced Rejection filter concept. The OER filter consists of a wide passband filter using Al_xO_y / $Al_{0.15}Ga_{0.85}As$ (the oxide filter) monolithically grown on top of a narrow passband filter using AlAs / $Al_{0.15}Ga_{0.85}As$ (the AlAs filter). The two filters must be appropriately coupled to yield the desired filter response. An etch-stop layer between the two filters is also desirable to facilitate patterning and oxidation of the upper filter while protecting the lower one.

In order to grown two filters atop one another and preserve their collective filter response, it is critical that the two filters are properly coupled. Moreover, since the OER filter requires that one portion of the filter be oxidized while the other part remain un-oxidized, we must have a way to expose a certain number of filter layers and leave the rest protected.

Figure 10 shows the layer structure of a filter that was designed to validate the OER filter concept. This design uses the 5-cavity filter of chapter 3 as the AlAs filter and monolithically incorporates a 5-cavity oxide filter. BraggModeller calculations indicate that replacing the last $\lambda_o/4$ layer of $Al_{0.15}Ga_{0.85}As$ in the oxide filter with a $\lambda_o/4$ layer of Indium-Gallium-Phosphide (InGaP) provides adequate optical coupling of the two filters. Since InGaP is a highly effective etch-stop for the selective wet etching of $Al_xGa_{1-x}As$ the oxide filter can easily be patterned for lateral oxidation without exposing the underlying AlAs filter. Thus InGaP may prove to be an ideal coupling layer.

Figure 11 shows BraggModeller calculations of the OER filter transmission along with the individual spectra of the oxide and AlAs filters. As desired, the OER filter has the narrow passband of the AlAs filter and the large DBR stop bands of the oxide filter thus facilitating a great deal of design flexibility. The calculations of Figure 11 indicate that the OER filter concept should prove very useful for coarse WDM passband filter applications.

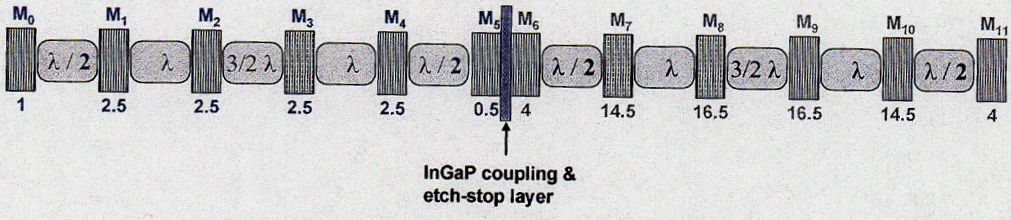


Figure 10. Layer structure of an actual OER filter design. The oxide filter is on the left side while the AlAs filter is on the right side. The two filters are coupled via a $\lambda_o/4$ thick layer of InGaP.

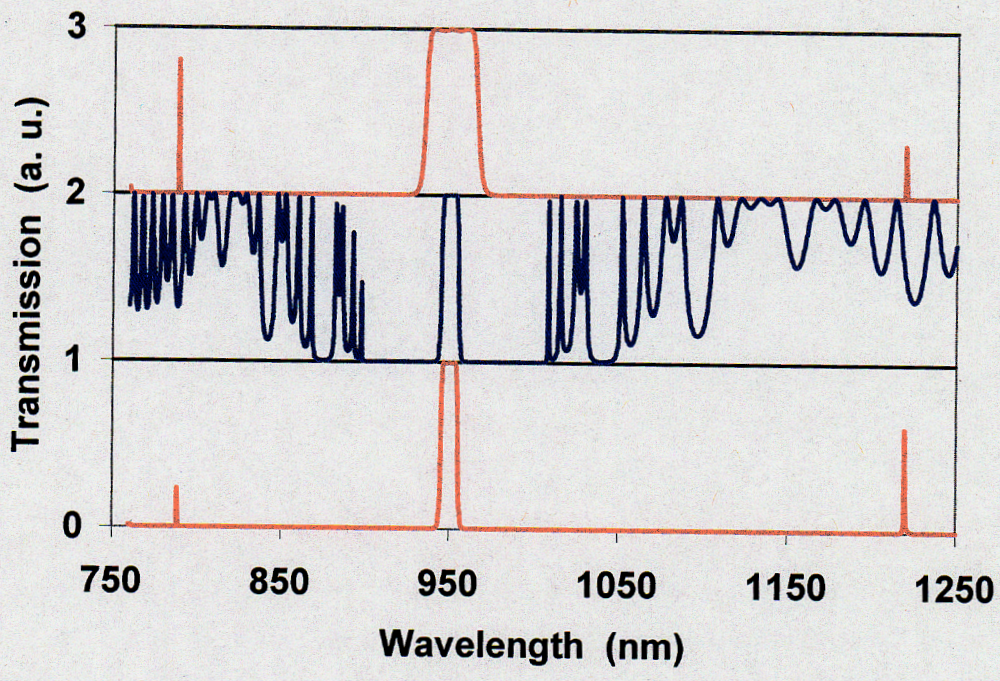


Figure 11. BraggModeller calculations for the OER filter. The top, middle and bottom curves are for the oxide filter, AlAs filter and the overall device respectively.

4.6 Experimental Results and Discussion

4.6.1 Sample Descriptions

Table 1 summarizes the samples that were designed and processed to investigate the use of lateral oxidation for filter applications. Sample A is a 3-cavity band-pass filter whose layer structure is given in Figure 12. The filter is designed to use Al_xO_y for the low index layers and $Al_{0.15}Ga_{0.85}As$ as the high index layers to test the idea of using lateral oxidation to make large stop-band filters. Prior to oxidation the high and low index layers consist of $Al_{0.15}Ga_{0.85}As$ and AlAs respectively. Upon oxidation the AlAs layers turn into Aluminum oxide (Al_xO_y). Due to the huge index contrast $\Delta n \approx 1.5$ a single Bragg pair with these materials has a reflectance of about 42%. Therefore the DBRs in this filter have very few Bragg pairs. As discussed earlier, this high reflectivity per Bragg pair can become a limiting factor in trying to meet filter design goals.

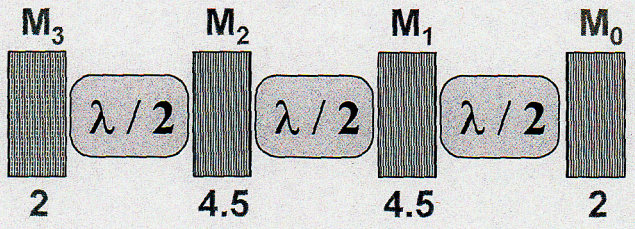


Figure 12. Layer structure of Sample A, the 3-cavity enhanced stop-band filter. Cavity lengths are indicated in multiples of the center wavelength λ. The number of Bragg pairs in each DBR is also given.

Sample B is a 2-cavity filter with 23nm thick tuning layers to explore the passband shifting filter concept. The layer structure for this filter is similar to that given in Figure 6 except that the tuning layers adjacent to the two cavities have not been reduced in thickness. This filter is designed to give a 40nm shift upon oxidation. As mentioned

previously, AlAs and AlGaAs with 15% and 85% aluminum fraction were used for the tuning layers, and high and low index layers respectively. Samples C and D have the same layer structure as sample B except that they incorporate thin barrier layers between the tuning layers and adjacent low index layers to prevent vertical oxidation. Each LHL layer sequence in Sample B is replaced by the sequence BLHLB, where B represents a thin $Al_{0.15}Ga_{0.85}As$ barrier layer. The barrier layers are 5 and 15 nm thick in samples C and D respectively.

Samples E and F are test structures to further characterize the effect of barrier layers in preventing unwanted vertical oxidation. The layer structure of these samples is described in more detail later.

Sample	**Description**
A	3-cavity large $\Delta\lambda_{DBR}$ filter
B	Passband Shifting – no barriers
C	Passband Shifting - 5nm barriers
D	Passband shifting – 15nm barriers
E	Barrier thickness study
F	Barrier composition study

Table 1. Summary of samples used to study lateral oxidation for filter applications.

4.6.2 Sample Growth and Processing

All samples were grown by MOCVD on GaAs substrates (100) 2° off towards (110). Pauses were used between each layer to avoid composition grading at the interfaces. For Samples A through D a 400nm InGaP etch-stop layer was grown between the substrate and the filter layers to allow for substrate removal. Samples were patterned with a triangular lattice of access holes, for the reasons described on page 97, and then etched using chemically assisted ion beam etching (CAIBE) with chlorine gas as the catalyst.

After etching, the samples were oxidized for varying amounts of time using the set-up of Figure 4 with a furnace temperature of 425°C, a bubbler temperature of 90°C and an N_2 flow rate of about 1.5 liters per minute.

To test the optical properties of the filters, oxidized samples were attached (epi side down) onto glass slides using an optically clear epoxy. The GaAs substrate was wet-etched using H_2O:H_2O_2:H_2SO_4 in a ration of 1:8:1. After substrate removal, the InGaP etch-stop layer was etched away using a selective wet etch consisting of a 1:1 ratio of HCl:H_3PO_4. Transmission spectra were obtained using a tungsten light source and an optical spectrum analyzer. Sample cross-sections were also examined with a scanning electron microscope.

4.6.3 Results

Figure 13 shows the transmission spectrum of Sample A after 60 minutes of oxidation. The passband has a 5 nm FWHM, with a center wavelength of 809nm. The insertion loss at the center wavelength is 13.3 dB. The inset in the figure shows that the long-wavelength DBR stop-band edge occurs at 1100nm. Due to equipment limitations,

the short wavelength stop-band edge could not be measured; however, it is expected that it should occur between 500 and 600nm.

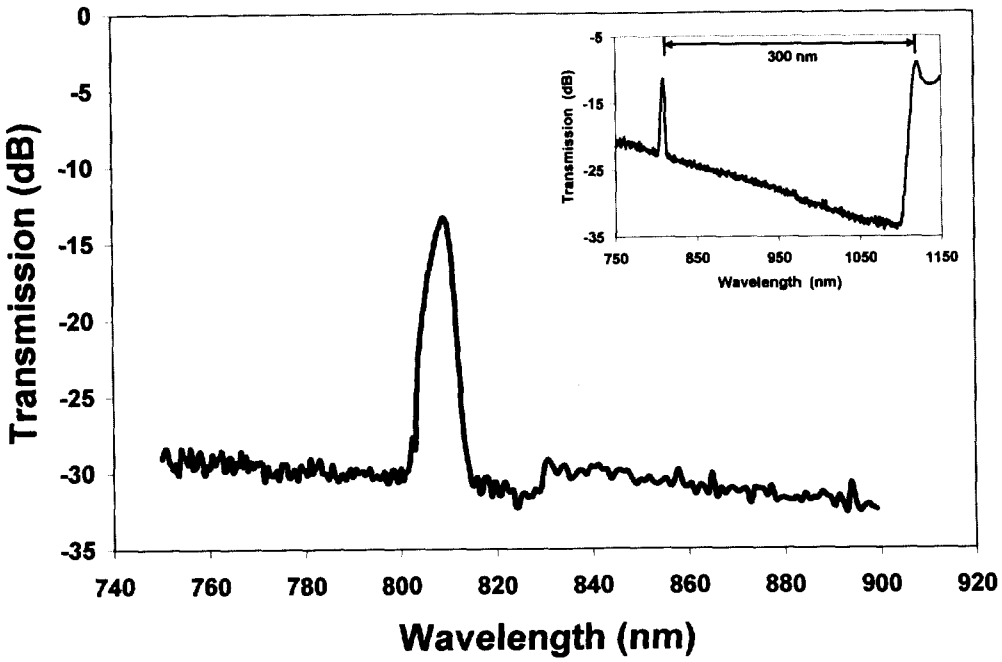


Figure 13. Transmission spectrum for Sample A. The sample was oxidized at 425°C for 60 minutes. The center wavelength is 809 nm and the FWHM of the filter is 5nm. The inset in the figure shows that the DBR stop-band edge occurs around 1100nm.

For Sample B, the low index layers adjacent to the tuning layers were completely oxidized due to vertical supply of oxidant from the tuning layers. This sample showed no passband response at all.

Figure 14 shows an SEM cross section of Sample C after 30 minutes of oxidation. The 5nm thick barrier layers do not prevent vertical oxidation. The HLH layer sequences in these filter designs provide a base rate of lateral oxidation of the low index layers (85% aluminum fraction). For 30 minutes of oxidation, the low index layers laterally oxidize 0.96μm. However, the low index layers adjacent to tuning layers have oxidized more

than 1 μm. Furthermore, the black areas extending vertically from tuning layers into adjacent low index layers indicate vertical oxidation (see arrows in Figure 14). Due to this vertical oxidation there were no clear oxidation fronts in these layers.

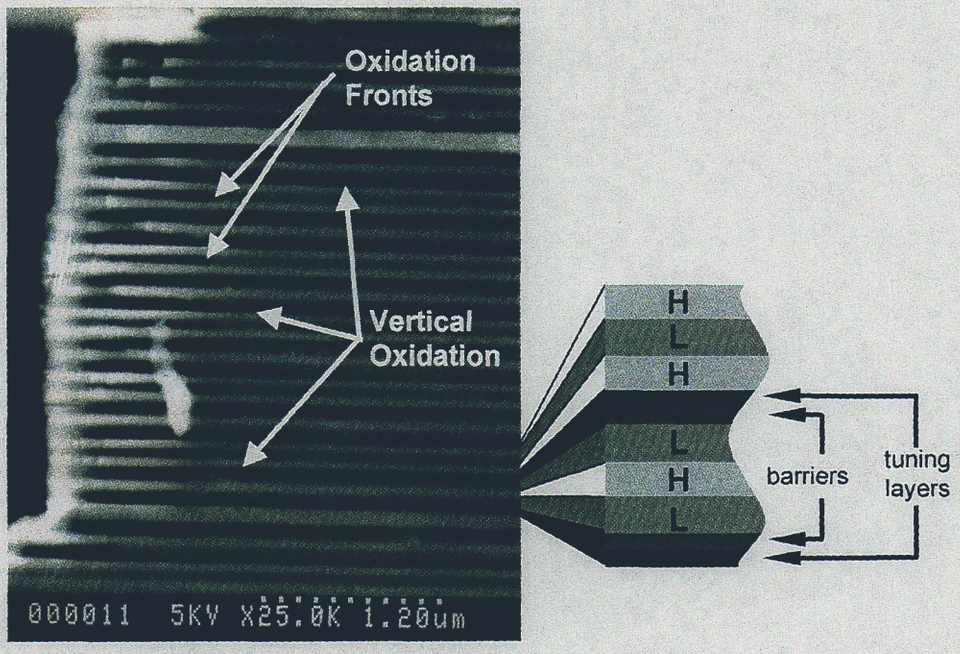


Figure 14. SEM cross-section of Sample C after 30 minutes of oxidation. The arrows indicate areas where the 85% aluminum low index layers have oxidized due to vertical supply of oxidant from adjacent tuning layers.

Figure 15 shows a photograph of the surface of Sample D after 45 minutes of oxidation. The yellow regions around each access hole are areas where the low index layers have been oxidized as verified by SEM analysis (not shown). The green regions are areas where only the tuning layers have been oxidized (as desired). The purple area on the right side of the figure is unoxidized because it was not patterned with access holes. Measurements from the photograph show that approximately 30% of the filter area is over-oxidized (yellow color). Although the photograph has been computer enhanced for clarity, the original sample does have a stark color contrast between the

three regions (green, yellow and purple). This contrast is a result of the large change in refractive index of the tuning layers and low index layers upon oxidation.

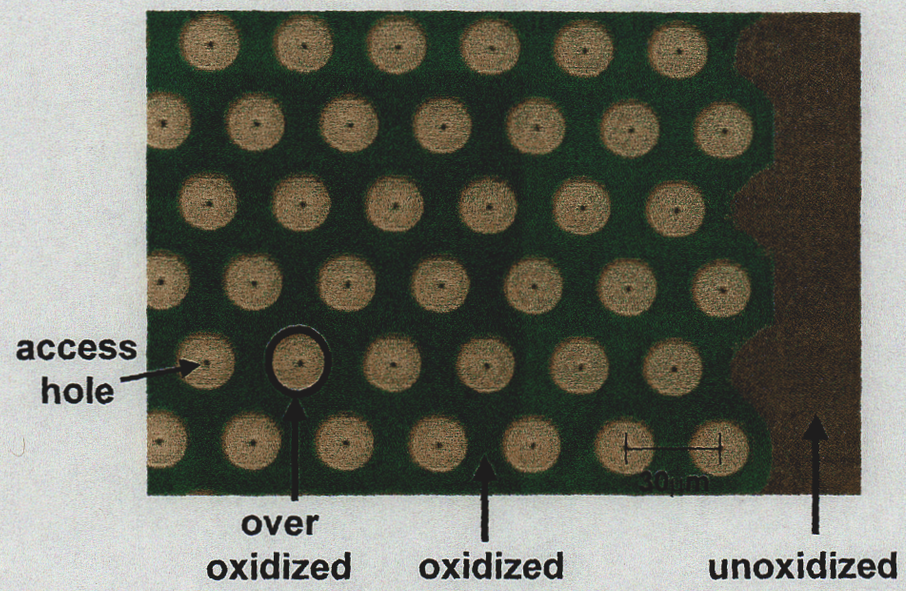


Figure 15. Photograph of the surface of Sample D after 45 minutes of oxidation. The Green regions are properly oxidized. Around each access hole there is a yellow region where the filter is over-oxidized. The unpatterned purple area on the right side of the figure is unoxidized. The colors in the photograph have been computer-enhanced for clarity.

Figure 16 shows the transmission spectra of sample D after 45 minutes of oxidation. The blue curve is from an area of the epi that is not oxidized because it was not patterned with access holes (see Figure 15). The red curve shows an area of epi that was patterned with 4 μm diameter holes on a 30μm triangular lattice. The oxidized sample exhibits 9dB excess insertion loss (compared to the unoxidized region), however there is a clear passband shape that is shifted 54nm with respect to the unoxidized sample. The large excess loss after oxidation is due to significant lateral oxidation of the low index layers in the vicinity of the access holes. The small peak in the oxidized filter spectrum at around

920nm is due to the transmission spectrum of the over-oxidized regions as verified with BraggModeller simulations (not shown).

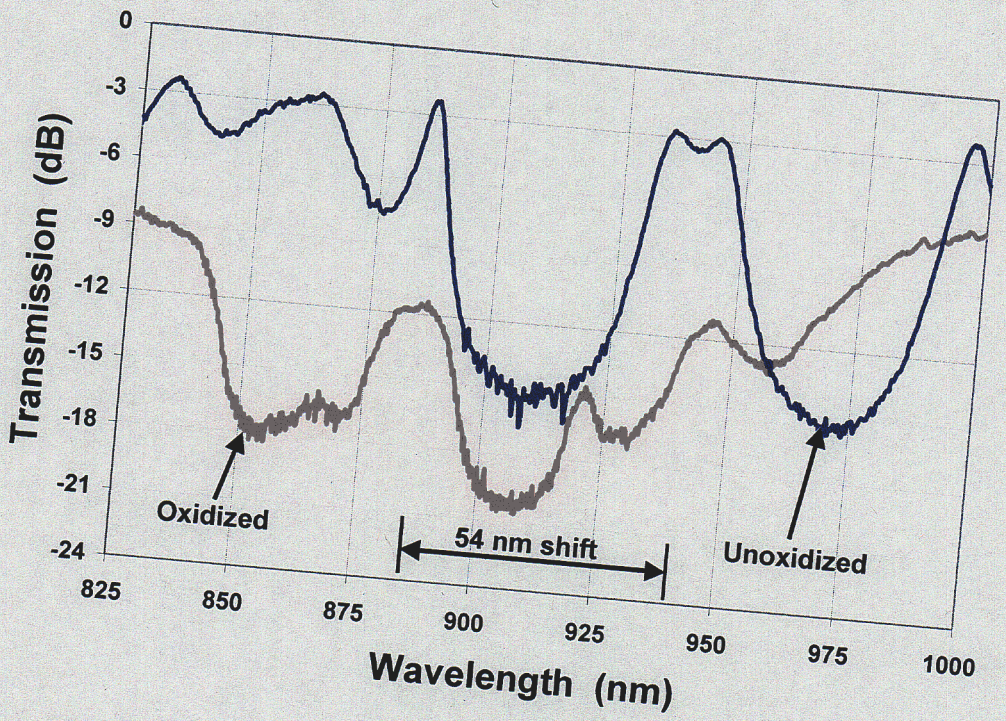


Figure 16. Transmission spectra for Sample D in oxidized and unoxidized regions. The sample was oxidized for 45 minutes at 425 °C. The blue curve is from an area of epi that is not oxidized because it was not patterned with access holes. The red curve is from an area of epi that was patterned with 4μm diameter holes on a 30μm triangular lattice.

4.6.4 Barrier thickness and Composition Study

The usefulness of the passband shifting filter concept hinges on the ability to oxidize the tuning layers without oxidizing other layers to a significant extent. The breakthrough of the 5nm thick barriers in sample C and the high insertion loss of sample D after oxidation prompted a study of the barrier layers. Blum et. al. previously studied the effect of barrier layer thickness variation on the wet oxidation process[22]. In their work very thin GaAs barriers (0.7 – 12 nm thickness) were placed between 45 nm thick $Al_{0.98}Ga_{0.02}As$ and $Al_{0.94}Ga_{0.06}As$ layers. The motivation for the thin barriers in that study was to control

the shape of the oxidation front by allowing the barriers to be penetrated. They found that the extent of oxidation of the 94% aluminum fraction AlGaAs layers increased as the barrier thickness decreased suggesting that either reactants (or products) were being supplied (or removed) via the adjacent higher aluminum content layer. For passband shifting filters, we are interested in completely isolating the low index layers from the adjacent tuning layers, therefore, the barriers studied here are 15nm and thicker.

Samples E and F are test structures aimed at studying the effects of variations in barrier thickness and Aluminum fraction respectively on the lateral oxidation process.

To simulate the layer structure of the passband shifting filters, the following basic layer sequence was used:

TBUHUBTHPH (19)

In (19) H signifies high index layers of quarter wave optical thickness (at λ_o=840nm). The symbols P and U represent protected and unprotected low index layers with thicknesses of 68.3 nm and 51.2 nm respectively. The U layers are separated from tuning layers by only a thin barrier layer B, while the P layers are clad by (thick) H layers on both sides and therefore protected from unwanted vertical oxidation. T signifies 23.2nm thick tuning layers, and B represents the barrier layers. The U layers are thinner than the P layers to account for the optical thickness of the barrier layers. $Al_xGa_{1-x}As$ with x= 0.15, 0.85, and 1.0 (pure AlAs) was used for the high index, low index and tuning layers respectively. In sample E, the barrier layers were $Al_xGa_{1-x}As$ with x=0.15 having thicknesses of 20, 25, 30, 35, 50, and 75 nm. In Sample F the barriers consisted of 15nm thick $Al_xGa_{1-x}As$ with x = 0, 0.05, 0.1, 0.2, 0.3, 0.5, and 0.7. For both samples, the basic

layer sequence of (19) was repeated twice for each value of barrier thickness or composition.

The samples were prepared in the same manner as Samples A through D as described above. After CAIBE (etching), the samples were oxidized at 425°C for 10, 20, 30, 60, and 90 minutes to study the behavior of the barriers as oxidation proceeds. Sample cross-sections were examined under SEM to determine the extent of lateral oxidation of the low index layers adjacent to tuning layers (U layers). Within each basic layer sequence, the P layers provide a baseline lateral oxidation rate for the 85% Aluminum layers. Penetration of the barrier layers is signified by a difference in the lateral extent of the oxide in the U and P layers.

No significant difference in the oxide depths for the U and P layers was observed in sample E for oxidation times up to 60 minutes indicating that all of the barriers prevented vertical oxidation from tuning layers to adjacent U layers. After 90 minutes of oxidation penetration of the thinnest barriers (20nm thickness) was evident in one piece of sample E but in another piece (oxidized in a separate oxidation run) these barriers were not penetrated. In all cases, barriers 25nm and thicker prevented vertical oxidation.

From sample F it was found that, 15 nm thick barriers with Al fraction up to 50% prevented vertical oxidation for at least 60 minutes. On the other hand, the 70%-Al barriers were penetrated in less than 20 minutes as indicated by the much larger oxide depth of the U layers compared to the P layers.

Figure 17 shows the cross section of sample F after 90 minutes of oxidation. A schematic of the layers after oxidation is given in Figure 18 to aid in interpreting the photograph. The U layers are oxidized further than the P layers for every barrier

composition. The stepwise nature of the oxidation front reflects the variation in barrier composition from $Al_{0.7}Ga_{0.3}As$ on the surface to GaAs near the substrate. The barrier layer composition clearly affects the vertical oxidation of the lower aluminum content U layers adjacent to the AlAs tuning layers.

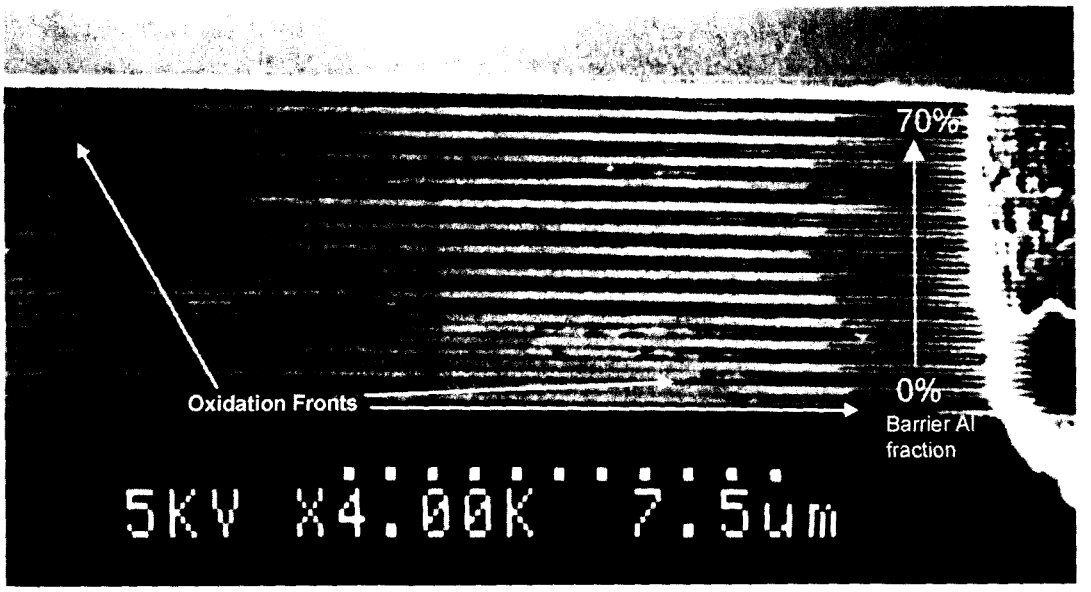


Figure 17. SEM cross-section of Sample F (barrier composition study) after 90 minutes of oxidation. The U layers are oxidized much further than the P layers indicating that the barriers are not preventing vertical oxidation. The stepwise oxidation fronts reflect the change in barrier layer composition from GaAs at the substrate to $Al_{0.7}Ga_{0.3}As$ at the surface. The circled region is used as a reference for a higher magnification image of the oxidation fronts given in Figure 19.

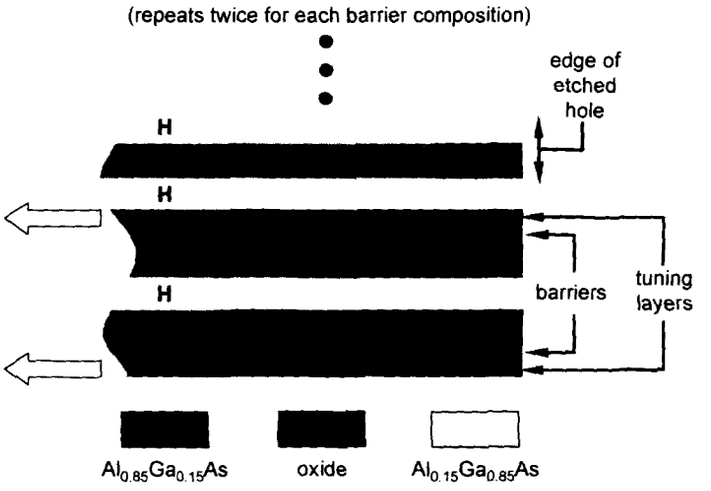


Figure 18. Schematic of Sample F after oxidation. For simplicity, the dark pockets of vertical oxidation in the U layers are not depicted. The barrier layers are $Al_xGa_{1-x}As$ with varying Aluminum fractions: ($0 \leq x \leq 0.7$).

Figure 19 shows a higher magnification image of the oxidation fronts in sample F. As seen in this figure, the vertical oxidation does not occur uniformly across the barriers. In each U layer, there is an oxidation front that extends from the access hole for some distance horizontally into the layer. Further into the layers and separated from this oxidation front several dark "pockets" are seen in each U layer. The same pattern was evident at every access hole in the 90 minute oxidized sample. The exact composition of the dark pockets could not be determined; however, the number of dark pockets increases as the barrier composition increases, and none of the P layers exhibit this phenomenon. This suggests that the dark pockets are indeed aluminum oxide, and therefore that the barriers did not uniformly prevent vertical oxidation. The exact nature and cause of this non-uniformity is unknown at present; however, it is clear that the effect of the tuning layers on the overall oxidation rate of the U layers cannot be modeled as a simple problem of diffusion across a uniform boundary.

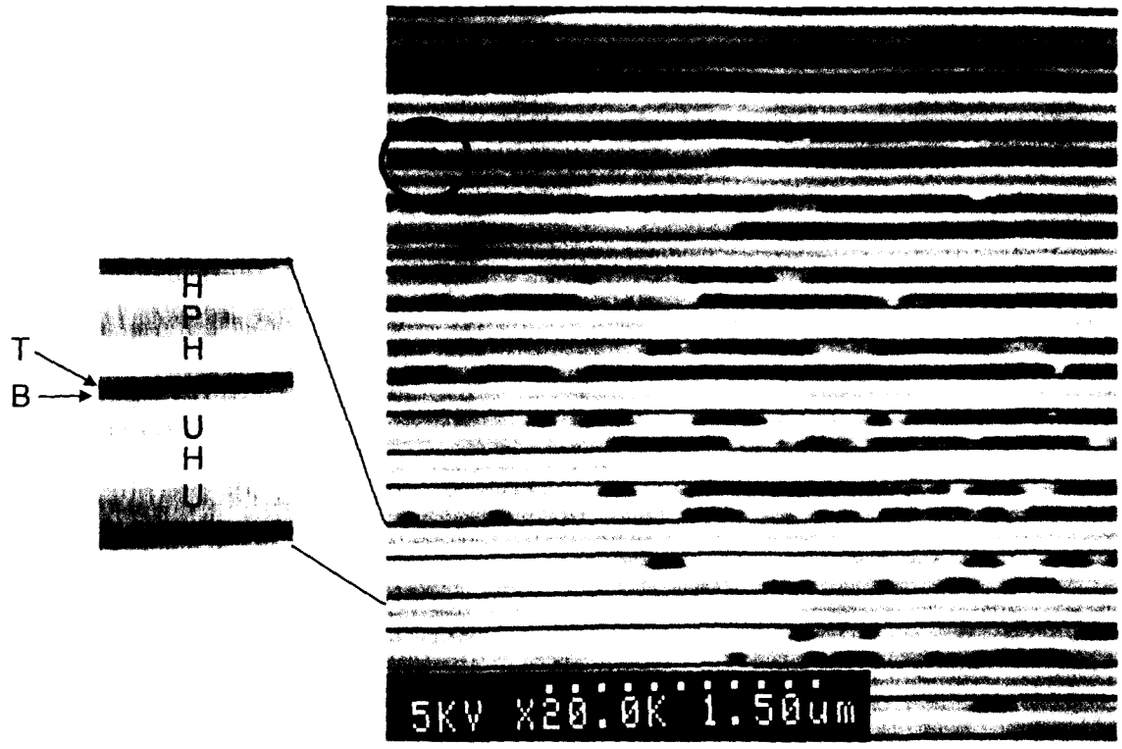


Figure 19. Close-up of the oxidation fronts in sample F after 90 minutes of oxidation. The circled region in this figure is the same region that was circled in Figure 17. The dark "pockets" in the U-layers are oxidized due to breakthrough of the barrier layers. Clearly the barriers do not uniformly protect the U-Layers.

4.7 Summary

Lateral oxidation of high aluminum content $Al_xGa_{1-x}As$ for filter applications was examined theoretically and experimentally. A 3-cavity bandpass filter with >500nm DBR stop-bands and a 2-cavity filter with a 54 nm passband shift upon oxidation were demonstrated.

The high insertion loss of the large stop-band filter can most likely be reduced through further process development. The insertion loss could be due to a number of factors including absorption, scattering, uneven oxidation, and larger than expected index change or volume contraction upon oxidation. Investigation of the exact cause of the loss for this particular filter sample is beyond the scope of this work.

The vertical oxidation of layers adjacent to the tuning layers must be eliminated to improve passband shifting filter performance after oxidation. To this end, the effects of variations in layer thickness and composition of thin barriers between the tuning layers and adjacent high Al content layers were studied. From the barrier composition study it was found that 15nm thick barriers with aluminum fractions of up to 50% can prevent vertical oxidation for up to 60 minutes. After 90 minutes, 15nm thick barriers were penetrated regardless of their composition. The barrier thickness study indicated that $Al_{0.15}Ga_{0.85}As$ layers 25nm and thicker prevented vertical oxidation for 90 minutes at (425 ^{o}C). 20nm Thick barriers prevent vertical oxidation for at least 60 minutes and may be effective for up to 90 minutes.

We have, in-principle, demonstrated that multiple cavity filters can be produced, and their passbands shifted, via lateral oxidation. Assuming that the outstanding process development issues are adequately resolved, these results could potentially remove limitations on system channel count due to the filter DBR stop-bands for coarse WDM systems and could potentially reduce the production cost of filter modules for such systems.

A new filter structure, the OER filter, that allows great flexibility in filter passband design while still taking advantage of the large DBR stop-bands afforded by Aluminum oxide was presented and a design example was provided. Even if the loss issues cannot be solved for lateral oxidation of AlAs, the OER filter concept is still viable. For example, amorphous thin films (with appropriately high Δn) could be sputtered or evaporated onto the narrow passband AlAs filter to provide the large DBR stop bands.

References for Chapter 4

[1] Dallesasse, J. M., Holonyak, N.-Jr., Sugg, A. R., Richard, T. A., and El-Zein, N., "Hydrolyzation Oxidation of $Al_xGa_{1-x}As$ –AlAs-GaAs Quantum Well Heterostructures and Super Lattices," Appl. Phys. Lett., V 57, 1990, pg. 2844-2846.

[2] Twesten, R. D., Follstaedt, D. M., Choquette, K. D., and Schneider, R. P. Jr., "Microstructure of Laterally Oxidized $Al_xGa_{1-x}As$ Layers in Vertical-Cavity Lasers", Appl. Phys. Lett. V69, 1996, pg. 19-21.

[3] Sugg A. R., Chen, E. I., Holonyak, N. Jr., Hsieh, K. C., Baker, J. E., and Finnegan, N., "Effects of Low-Temperature Annealing on the Native Oxide of $Al_xGa_{1-x}As$", J. App. Phys. V74, 1993, pg. 3880-3885.

[4] Choquette, K. D., Geib, K. M., Chui, H. C., Hammons, B. E., Hou, H. Q., Drummond, T. J., and Hull, R., "Selective Oxidation of AlGaAs Versus AlAs", Appl. Phys. Lett. V69, 1996, pg. 1385-1387.

[5] Burton, R., S., and Schlesinger, T. E., "Wet Thermal Oxidation of $Al_xGa_{1-x}As$ Compounds", J. Appl. Phys., V76, 1994, pg. 5503-5507.

[6] Ochiai, M., Giudice, G. E., Temkin, H., Scott, J. W., and Cockerill, T. M., "Kinetics of Thermal Oxidation of AlAs in Water Vapor", Appl. Phys. Lett. V68, 1996, pg. 1888-1900.

[7] Ashby, C. I. H., bridges, M. M., Allerman, A.A., Hammons, B. E., and Hong, H. Q., "Origin of the Time Dependence of Wet Oxidation of AlGaAs", Appl. Phys. Lett. V75, 1999, pg. 73-75.

[8] Dallesasse, J. M., and Holonyak, N. Jr., "Native-Oxide Stripe-Geometry $Al_xGa_{1-x}As$-GaAs Quantum Well Heterostructure Lasers", Appl. Phys. Lett. V58, 1991, pg. 394-396.

[9] Huffaker, D. L. Deppe, D. G., Kumar, K., and Rogers, T. J., "Native-Oxide Defined ring Contact for Low Threshold Vertical-Cavity Lasers", Appl. Phys. Lett. V65, 1994, pg. 97-99.

[10] Choquette, K. D., Lear, K. L., Schneider, R. P., Jr., Geib, K. M., Figiel, J. J., and hull, R., "Wavelength Insensitive Performance of Robust Selectively Oxidized Vertical-Cavity Lasers", IEEE Phot. Tech. Lett., V7, 1995, pg. 1237-1239.

[11] Fiore, A., Akulova, A., Ko, J., Hegblom, E. R., and Coldren, L. A., "Multiple-Wavelength Vertical-Cavity Laser Arrays Based on Post-Growth Lateral-Vertical Oxidation of AlGaAs.

[12] Boyd, R. W., Nonlinear Optics, Harcourt Brace & Company, New York, ©1992, pg. 399-411.

[13] MacDougal, M. H., and Dapkus, P. D., "Wavelength shift of Selevtively Oxidized Al_xO_y-AlGaAs-GaAs Distributed Bragg Reflectors", IEEE Phot. Tech. Lett. V9, 1997, pg. 884-886.

[14] Choquette, K. D., Geib, K. M., Ashby, C. I. H., Twesten, R. D., Blum, O., Hou, H. Q., Follstaedt, D. M., Hammons, E. B., Mathes, D., and Hull, R., "Advances in Selective Wet Oxidation of AlGaAs Alloys", IEEE JSTQE, V3, 1997, pg. 916-926.

[15] Deal, B. E., and Grove, A. S., "General relationship for the Thermal Oxidation of Silicon", J. Appl. Phys. V36, 1965, pg. 3770-3778.

[16] Kim, J. H., Lim, D. H., Kim, K. S., Yang, G. M., Lim, K. Y., and Lee, H. J., "Lateral Wet oxidation of $Al_xGa_{1-x}As$ Depending on its Structures", Appl. Phys. Lett. V69, 1996, pg. 3357-3359.

[17] Guha, S., Agahi, F., Pezeshki, B., Kash, J. A., Kisker, D. W., and Bojarczuk, N. A., "Microstructure of AlGaAs-Oxide Heterolayers Formed by Wet Oxidation", Appl. Phys. Lett. V68, 1996, pg. 906-908.

[18] Naone, R. L., and Coldren, L. A., "Surface Energy Model for the Thickness Dependence of the Lateral Oxidation of AlAs", J. Appl. Phys. V82, 1997, pg. 2277-2280.

[19] Epstein, L., "The Design of Optical Filters", JOSA V42, 1952, pg. 806-810.

[20] Thelen, A., Design of Optical Interference Coatings, McGraw-Hill, New York, NY, 1989.

[21] Ohmer, M. C., "Design of Three-Layer Equivalent Films", JOSA V68 1978, pg. 137-139.

[22] Blum, O., Ashby, C. I. H., and Hou, H. Q., "Barrier-Layer-Thickness Control of Selective Wet Oxidation of AlGaAs for Embedded Optical Elements", Appl. Phys. Lett. V70, 1997, pg. 2870-2872.

Chapter 5 Wavelength Routing – An Alternate Architecture for λ-Connect

5.1 Introduction

The broadcast and select architecture of λ-Connect relies on multiple wavelength transmitters, a non-wavelength selective broadcast medium, and wavelength selective filters at the receiver. Because the broadcast element sends every incoming signal to all output ports, there is an inherent splitting loss that scales with the number of output ports, N, as show in Figure 1. Due to power budget constraints, this splitting loss eventually limits the scaling of λ-Connect to higher node counts.

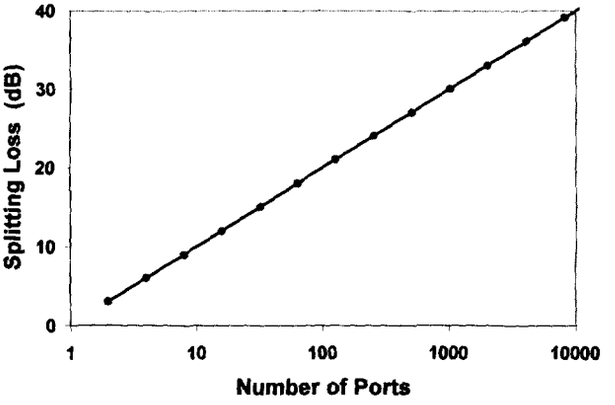


Figure 1. Splitting Loss as a function of the number of output ports for a wavelength insensitive broadcast element.

To mitigate the 1/N splitting loss, an architecture that does not spread incoming signals to all outputs is required. The broadcast element of λ-Connect must be replaced by a device that sends (routes) signals to the intended destination without sending photons to other ports. This can be accomplished in several ways. For example, the broadcast element could be replaced by an NxN cross-bar switch. Generally cross-bar

switches are wavelength insensitive; however, they are active devices in that they can be re-configured (switched) to facilitate individual (1 to 1) connections between each input and output port. The ability to make 1 to 1 connections and switch between the output ports rather than distributing the signal to all outputs (in a 1 to N manner) avoids the 1/N splitting loss. Furthermore, a fully non-blocking cross-bar switch has the ability to connect any input/output pair without disrupting already established connections. This is a key to achieving high throughput and connectivity. The drawback to these switches is the complex circuitry required to control the switch. This increases system cost compared to the passive broadcast element of λ-Connect.

At the present time there is a large commercial effort to develop all-optical switches for the long-haul DWDM market. These switches generally are MEMs based devices with tiny micro-mirrors for beam steering. Based on the development cycle of other WDM technologies, it is likely that once the long distance telecommunications market has been saturated, companies will re-engineer this technology to be cost effective for shorter distance optical links. However, if the switch is not wavelength selective, the source routing allowed by multiple system wavelengths in λ-Connect would be lost.

Another way to mitigate the splitting loss is to use a passive but wavelength-selective device to distribute signals. This approach is called wavelength routing because it relies on the wavelength of the signal to provide routing of the data. This approach follows the λ-Connect paradigm of using multiple system wavelengths to facilitate source routing. In this chapter we examine the concept of wavelength routing as an alternative architecture for λ-Connect and present results for a bit-parallel 3x3 passive wavelength router for MMF based CWDM systems.

5.2 Wavelength Router – Concept

A wavelength router is an NxN multiplexer that provides a unique mapping of wavelength vs. output for each of the N input ports. This means that the combination of input port number and wavelength uniquely determine the destination of a message. Figure 2 shows the routing table for a wavelength router having 3 input and output ports. Note that each output port receives all 3 wavelengths, however, each wavelength comes from a different input.

Routing Table

outputs \ inputs	A	B	C
X	λ_1	λ_3	λ_2
Y	λ_2	λ_1	λ_3
Z	λ_3	λ_2	λ_1

Figure 2. Routing Table for a 3x3 wavelength router. Each input (A,B,C) has a unique mapping of wavelength to output (X,Y,Z).

Besides avoiding the 1/N splitting loss, routing incoming signals to only one output port makes it possible to connect N inputs to N outputs using only N wavelengths - as long as each wavelength is used N times in the routing table. For example, λ_1 can be used to connect input A to output X, input B to output Y and input C to output Z. Similarly, λ_2 can connect input A to output Y, input B to output Z and so forth. This functionality, called wavelength re-use[1], makes maximum use of the available wavelengths with more

than one transmitter using the same wavelength at the same time. In contrast, the broadcast and select architecture only allows one transmitter to use a given system wavelength at any given time.

For single mode fiber DWDM applications this passive wavelength routing functionality is provided by arrayed waveguide grating routers (AWGs)[2,3,4]. As shown in Figure 3, the AWG router consists of an optical phased array (the arrayed waveguide grating), two slab waveguides that function as lenses and a series of input and output waveguides.

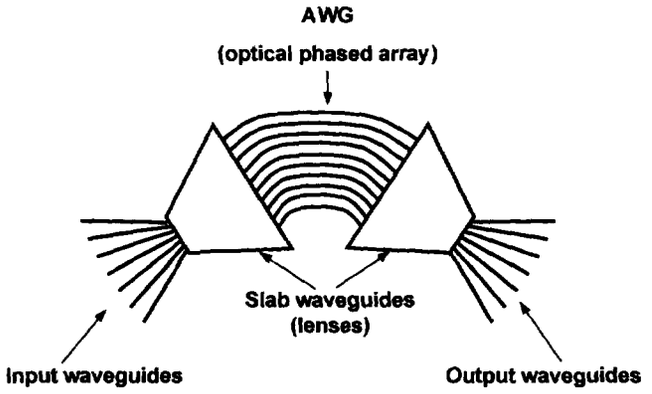


Figure 3. Schematic of a passive wavelength router based on an optical phased array for single mode applications.

Each waveguide in the AWG has a slightly different path length than its neighbors. The combined effect is to tilt the direction of the beam upon exiting the grating. The amount of tilting varies depending on the wavelength of the light and the incident angle. By placing the input and output waveguides at appropriate places along the Rowland circle of the grating, the variation in exit angle with wavelength can be used to precisely couple adjacent spectral channels to adjacent outputs. Wavelength re-use is accomplished by using overlapping diffraction orders of the grating.

AWG routers cannot be used with MMF due to the excessive losses in coupling from MMF to single mode waveguides. Furthermore, the AWG is like a diffraction grating operating in a very high order. In one report, the AWG for a DWDM application was designed to operate in the 118th order [3]. Due to the small free spectral range in such high orders, these devices are unsuitable for CWDM systems. We have developed a wavelength router for use with MMF based CWDM optical networks. Furthermore, we have demonstrated a bit-parallel version of this router where each input and output port consists of two fibers in parallel. The device uses a blazed diffraction grating as the dispersive element and broadband add/drop filters to provide wavelength re-use for fully non-blocking NxN interconnection. In the remainder of this chapter we explain the design of the router, present the experimental results, discuss methods to scale the device to higher port counts and more parallel bits, and discuss other approaches to wavelength routing for MMF CWDM systems.

5.3 Device Design

5.3.1 Partial Routing with Diffraction Gratings

Figure 4 shows the partial wavelength routing that can be achieved with a diffraction grating. For each input the N system wavelengths diffract into N output angles according to the grating equation:

$$m\lambda = d\left(\sin\theta_i + \sin\theta_o\right) \tag{1}$$

Where m is the diffraction order, λ is the wavelength, d is the groove pitch (or tooth pitch) of the grating, and θ_i, and θ_o are the incident and exit angles respectively. To fill in the routing table we require that λ_l from input k ($\lambda_{l,k}$) be diffracted in the same direction

as λ_2 from input k-1 ($\lambda_{2,k-1}$). Uniform spectral channel spacing, $\Delta\lambda$, implies that aligning $\lambda_{1,k}$ with $\lambda_{2,k-1}$ also aligns the other spectral channels $\lambda_{i,k}$ with $\lambda_{i+1,\,k-1}$ for i<N. The last channel $\lambda_{N,k}$ will not overlap with any signal from input k-1. As seen in the figure, every input is represented by one column in the routing table and each successive input adds one row to the table. Thus, N inputs (with N wavelengths) produce 2N-1 output beams. The Nth output (output 3 in the figure) has all system wavelengths properly routed. The remainder of the outputs can be grouped into pairs of complementary beams. For example output 1 in the figure contains only λ_1 while output 1* contains λ_2 and λ_3. Similarly, outputs 2 and 2* in the figure are complementary to each other. In the general case output k contains λ_i where $1 \leq i \leq k$ and output k* contains λ_j where $k+1 \leq j \leq N$.

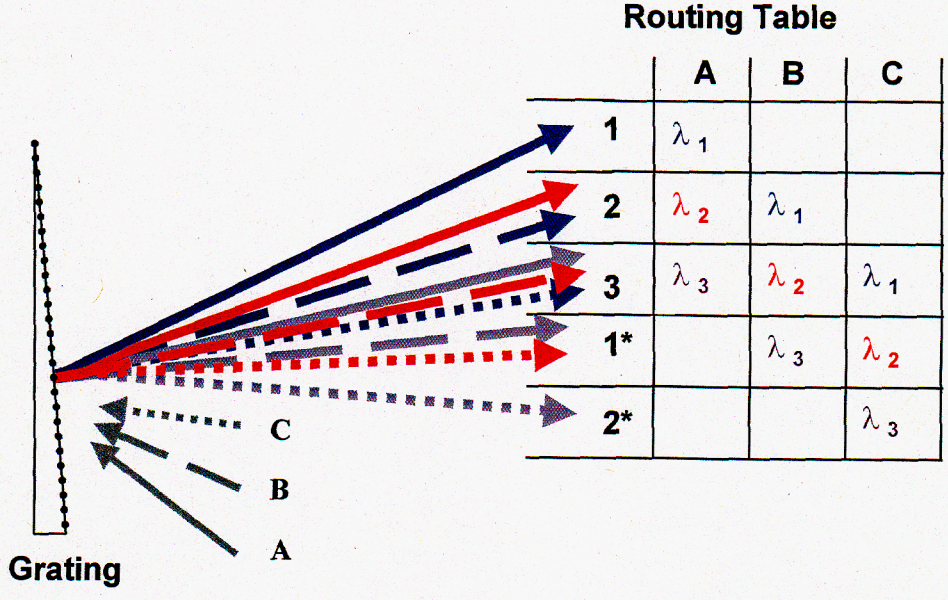


Figure 4. Partial wavelength routing with a diffraction grating. The input angle of each successive input port is calculated using the grating equation and requiring that λ_1 from input k diffract at the same angle as λ_2 from input k-1. Outputs 1* and 2* are complementary to outputs 1 and 2 respectively. Solid dashed and dotted lines represent signals from input ports A, B, and C respectively.

Wavelength routers using a concave diffraction grating have been reported previously[5,6]. A concave grating can serve as both the focusing and dispersive element to couple light into and out of fibers. Both of these devices used single mode fibers and were designed for DWDM. Although both of these devices were called NxN routers, in actuality they had N input and 2N-1 output ports and routing tables similar to the one in Figure 4. Therefore, although each input can address N output ports (with N wavelengths), the N inputs do not address the same N outputs.

In order to achieve true NxN interconnection, the routing table given in Figure 4 must be converted into the routing table of Figure 2. This is accomplished by combining each pair of complementary outputs into a single output. Since there are N-1 pairs of complementary outputs this process results in N-1 outputs. Thus the end result is a device with N inputs and N fully routed outputs.

The complementary outputs could be combined with simple (non-wavelength selective) 2x1 couplers; however, this would result in an extra 3dB loss. Instead, our device employs 3-port add/drop filters (described earlier in chapter 3). Since the filters are wavelength selective they avoid the 3 dB coupling loss.

In this application the add/drop filters function as edge filters reflecting certain channels and transmitting the rest. As system channel spacing $\Delta\lambda$ is decreased, the edge filter must become steeper and steeper. The add/drop filters discussed in chapter 3 could most likely be re-designed to support channel spacings of approximately 10nm. If the channel spacing were to be lower than this, the grating router presented here would have to rely upon 3-dB couplers.

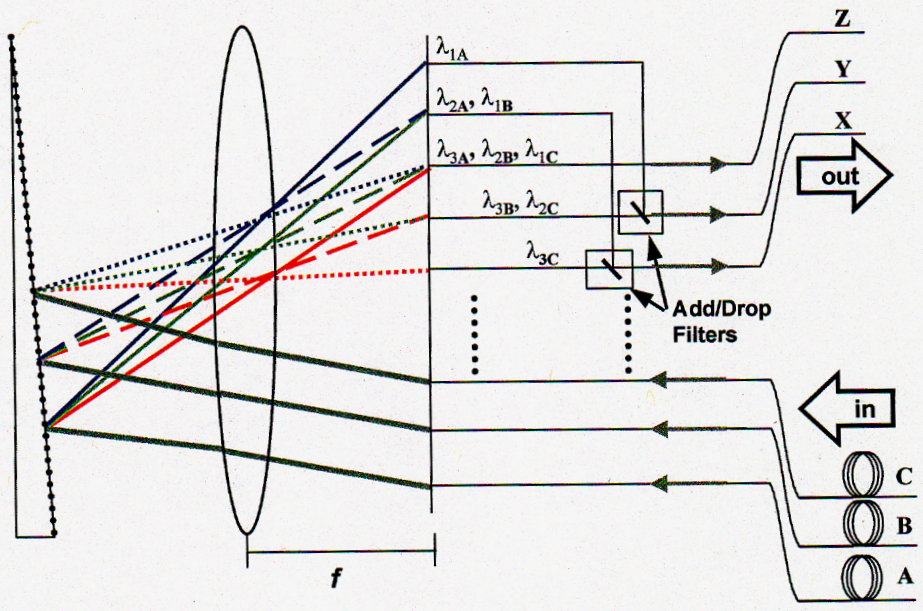


Figure 5. Setup for the experimental demonstration of the grating router with add/drop filters. Inputs and outputs are 62.5 μm core GRIN MMF. The lens used was a 0.4 NA microscope objective (f = 16mm), the gold coated grating had 400 lines/mm, and the add/drop filters were similar to those described in chapter 3.

5.3.2 Experimental Setup

Figure 5 shows the experimental setup used to demonstrate the wavelength router. In this figure, the inputs A, B, and C are mapped to outputs 1, 2, 3, 1*, and 2*, which are subsequently combined with add/drop filters to produce the final 3 outputs X, Y, and Z. Wavelength routing was demonstrated using 3 wavelength channels: 828, 865, and 902 nm. Graded index (GRIN) 62.5/125 μm MMF inputs and outputs were terminated in an MT ferrule to provide a fiber to fiber pitch of 250 μm. Three fibers (A, B, C) were illuminated with white light from a tungsten lamp. A lens was used to collimate the incident light from the inputs and focus the diffracted light from the grating. Based on

the fiber pitch and spectral channel spacing a linear dispersion of $\Delta x/\Delta\lambda = 250/37 = 6.76\mu m/nm$ was required in the focal plane of the lens.

The linear dispersion of a lens and grating combination used in the Littrow configuration is given by:

$$\frac{\Delta x}{\Delta\lambda} \approx \frac{2f\tan(\theta)}{\lambda} \tag{2}$$

where f is the focal length of the lens, and θ is the blaze angle of the grating. This equation is valid for wavelengths near the blaze wavelength. Equation 2 is derived as follows. Differentiating the grating equation (1) with respect to λ we find the angular dispersion of the grating:

$$\frac{\Delta\theta}{\Delta\lambda} = \frac{m}{d\cos(\theta_o)} \tag{3}$$

multiplying both numerator and denominator by $\sin(\theta_o)$ gives:

$$\frac{\Delta\theta}{\Delta\lambda} \approx \frac{2\tan(\theta)}{\lambda} \tag{4}$$

in (4) we have made the approximation that $\theta_i \sim \theta_o$ and substituted λ for $2d\sin(\theta)/m$ (from the grating equation). Furthermore, in the Littrow configuration $\theta_i \sim \theta_o \sim \theta$ where θ is the blaze angle of the grating. In the focal plane of a lens, the linear dispersion is just the focal length (f) times the angular dispersion thus equation (4) leads to equation (2).

The diffraction grating used in this demonstration had a groove density of 400 lines/mm (d = 2.5 μm), blaze angle of 9.962 degrees (blaze wavelength = 845 nm for Littrow mounting), and was gold coated for high reflectivity. Based on the grating parameters, a 0.4 NA microscope objective lens with a focal length of 16mm was used to expand and focus the light to and from the fibers. By matching the linear dispersion of

the lens and grating combination to the fiber pitch and spectral channel spacing, adjacent spectral channels from a single input are focused to adjacent output fibers. For example, input A, sends λ = 828, 865, and 902 nm to outputs 1, 2, and 3 respectively. Furthermore, by spacing the input fibers with the same pitch as the outputs, adjacent inputs send adjacent spectral channels to the same output. Thus, output 3 receives λ = 902, 865, and 828 nm from inputs A, B, and C respectively.

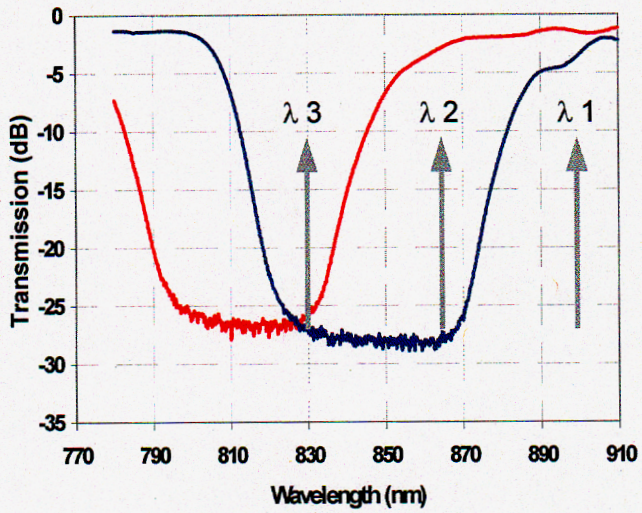


Figure 6. Through port (port 1 to port 2) transmission of the add/drop filters used to achieve wavelength re-use. The filter corresponding to the red curve has a center wavelength of ~810 nm while the filter corresponding to the blue curve has a center wavelength of ~850 nm.

Figure 6 shows the port 1 to port 2 (see Chapter 3, Figure 10) transmission spectra for the two add/drop filters used to re-combine the two pairs of complementary output beams. In this application, two inputs are multiplexed onto a single output, one input being reflected by the filter and the other being transmitted through the filter. Note that the filter corresponding to the red curve red curve (λ_o ~810 nm) reflects λ_3 and transmits λ_1 and λ_2. On the other hand the filter represented by the blue curve (λ_o ~850 nm)

reflects λ_2 and λ_3 and only transmits λ_1. Thus the filters represented by the red and blue curves multiplex outputs 2, 2* and 1, 1* respectively.

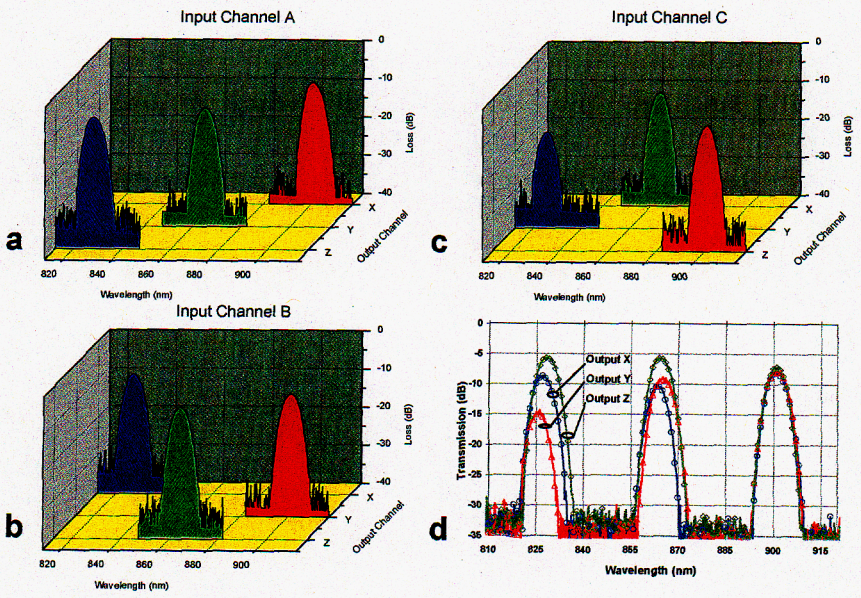


Figure 7. Grating router transmitted signal as a function of wavelength. The routing for input ports A, B, and C are shown in figures 7-a, -b, and -c respectively. Figure 7-d shows the signals for each output port. Although not apparent from 7-d, the three wavelength peaks in each output spectrum come from different input ports following the routing table given in Figure 2.

5.4 Experimental Results

Figure 7 shows the grating router output spectra. The signals are grouped by input port in Figure 7-a through Figure 7-c. Figure 7-d shows the spectra grouped by output port (labeled X, Y, and Z in Figure 5). Each output channel contains all three spectral channels. Although not apparent from Figure 7-d, each channel in each output spectrum originates from a different input fiber as indicated in Figure 7-a through Figure 7-c. In this initial demonstration the channels had a FWHM of 6 nm. The average insertion loss of the device was 8.7 dB with a standard deviation of 0.81.

Optimizing all of the system components will significantly reduce the insertion loss. For example, the add/drop filters were originally developed for another application and exhibit uncharacteristically high insertion losses, up to a maximum of 4.2 dB, for the spectral channels of this experiment. These filters are reported to exhibit average insertion losses of 1 to 1.5 dB when optimized for the wavelengths of interest (see chapter 3). Furthermore, the maximum grating efficiency was 87% (0.6 dB loss). Finally, the maximum insertion loss of 14.7 dB corresponds to the spectral channel that subtends the largest angle from input to output, travelling from input A to output 1 on Figure 5. This light path has very high loss due to vignetting as confirmed by ray tracing.

5.4.1 Elimination of Vignetting

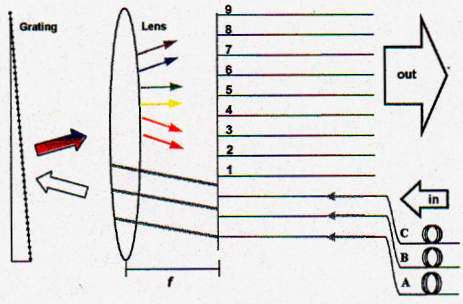


Figure 8. Experimental setup used to characterize the grating router with a new lens. The lens is a camera lens with f/# = 1.1 and f = 37.5 mm focal length. Three fibers in an MT ferrule were illuminated with white light, the diffracted light was collected from the remaining 9 fibers in the ferrule.

The lens originally used in the system was a microscope objective with a 16mm diameter pupil. To verify that the insertion loss of the router could be improved by optimizing the optics, we replaced the microscope objective with a larger diameter, f = 37.5mm, f/1.1 camera lens. Figure 8 shows the experimental setup used to characterize the system with the new lens. Three fibers in an MT ferrule were illuminated with white

light. The remaining 9 fibers in the ferrule were used to collect diffracted light from the grating. By measuring the spectral response in all 9 fibers, we were able to determine that the problem of vignetting had been eliminated. This is indicated by the fact that the insertion loss of the system was about 5dB for all wavelengths in all fibers. Figure 9 shows the transmission of the system with the new lens. Each output fiber receives 3 separate wavelengths because three input fibers are illuminated. Signals from fibers 1, 4, 6, and 9 are shown. The signals from fibers 4 and 6 partially overlap each other. The inset in the figure shows signals from all 9 fibers. The increased focal length of the new lens resulted in a linear dispersion of 20.8 μm/nm. With this lens the system supports spectral channel spacings of 12nm rather than the 37nm required by the available add/drop filters. Although demonstration of full wavelength routing (with wavelength re-use) was not possible, the average insertion loss of the lens and grating combination was reduced to about 6 dB and the system exhibited excellent fiber-to-fiber and wavelength-to-wavelength uniformity. Ultimately, we believe that a device exhibiting insertion losses of 3 to 4 dB per channel will be attainable.

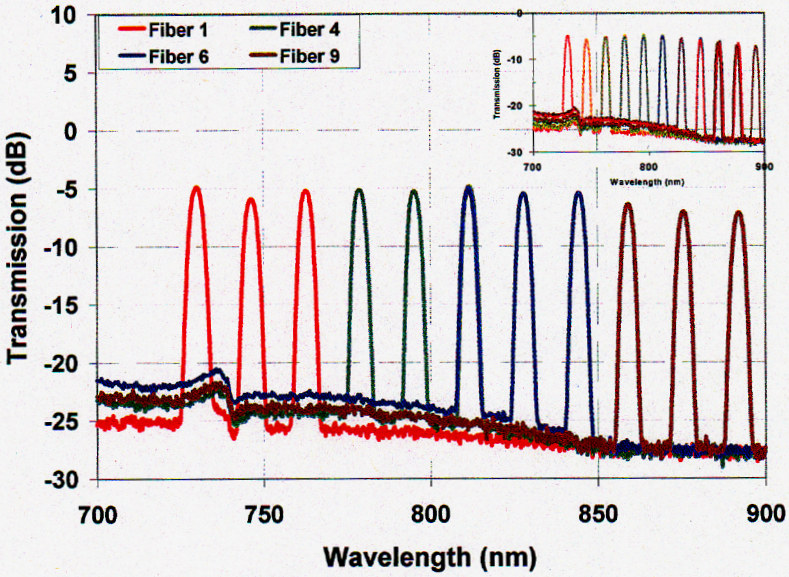


Figure 9. Grating router transmission (without filters) using a f = 37mm lens. fibers 1, 4, 6 and 9 are shown. The signals for fibers 4 and 6 overlap at λ=810nm. The Inset shows signals for all 9 fibers. Insertion loss is about 6 dB for all channels on all fibers indicating that the vignetting problem has been eliminated.

5.4.2 Bit-Parallel Wavelength Router

For use with λ-Connect we ultimately require a bit-parallel wavelength router having 12 or more parallel bits. To this end, we demonstrated bit-parallel operation of the 3x3-wavelength router. A ferrule with 2 parallel rows of 12 fibers each was used to hold the input and output fibers. Thus each input and output port had 2 parallel bit lines. Figure 10 shows the spectra for the 2 parallel bit-lines of each output port. The spectra exhibit excellent bit to bit uniformity.

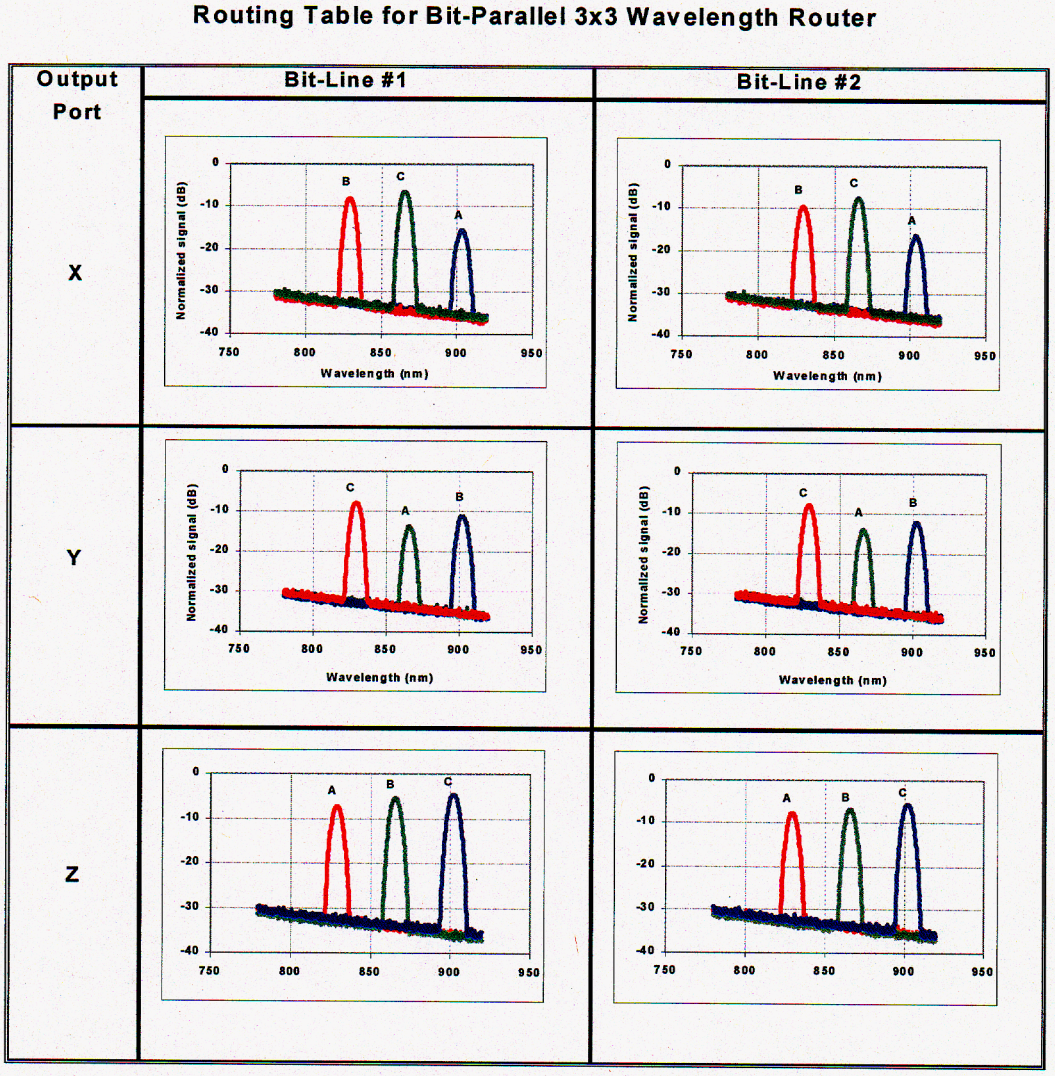


Figure 10. Output spectra for the bit-parallel 3x3 wavelength router.

5.5 Routing with Multiple Routers

As seen in the previous section, in order to route N wavelengths a total of 3N-1 fibers are required in this wavelength router design. (N input fibers plus 2N-1 output fibers.) As the wavelength count increases the requirements on the optics of the system will become more and more demanding. For large wavelength count, the lens vignettes the outer-most fibers. As the lens diameter increases, the smallest f/# achievable also

increases. Since the f/# of multimode fibers is 1.81, any f-ratio larger than this will fail to capture all of the light emitted from the input fiber resulting in unacceptable insertion loss.

In a practical implementation of λ-Connect 8 or 16 system wavelengths would most likely be used. Simulations conducted by DeGroot et. al.[7] indicate that little performance benefit is gained by increasing N past 16. Therefore we present a method that can reasonably scale from the 3x3 router demonstrated above to a case with N = 16 by using multiple routers.

In general, a grating router with K inputs, each carrying N wavelengths, will produce N+K-1 partially routed output beams (without filters). The routing table for such a router is similar to that in Figure 4. K inputs reproduce the first K columns of the routing table for the N wavelength system. Therefore, by properly combining the outputs of multiple routers, each having less than N input ports it is possible to achieve full NxN routing.

As seen in Figure 11, the routing functionality of an 8-wavelength system can be achieved using 2 routers (A and B) having 4 inputs and 11 outputs each. Inputs 1-4 of the 8-wavelength system are sent to router A, while inputs 5-8 are sent to router B. Each of these routers reproduces four columns of the 8-wavelength routing table. To reconstruct the 8-wavelength routing table output 1 from router B (B1) must be combined with output 5 from router A (A5). Similarly, we must combine B2 with A6, B3 with A7, B4 with A8, 5B with A1*, B6 with A2*, and B7 with A3*. This combination of output beams from routers A and B can be easily achieved using add/drop filters similar to the ones used to achieve full wavelength re-use for the case of a single N-input router. Thus

we see that in this example the use of 2 routers requires the introduction of 7 extra add/drop filter modules to achieve the full wavelength re-use.

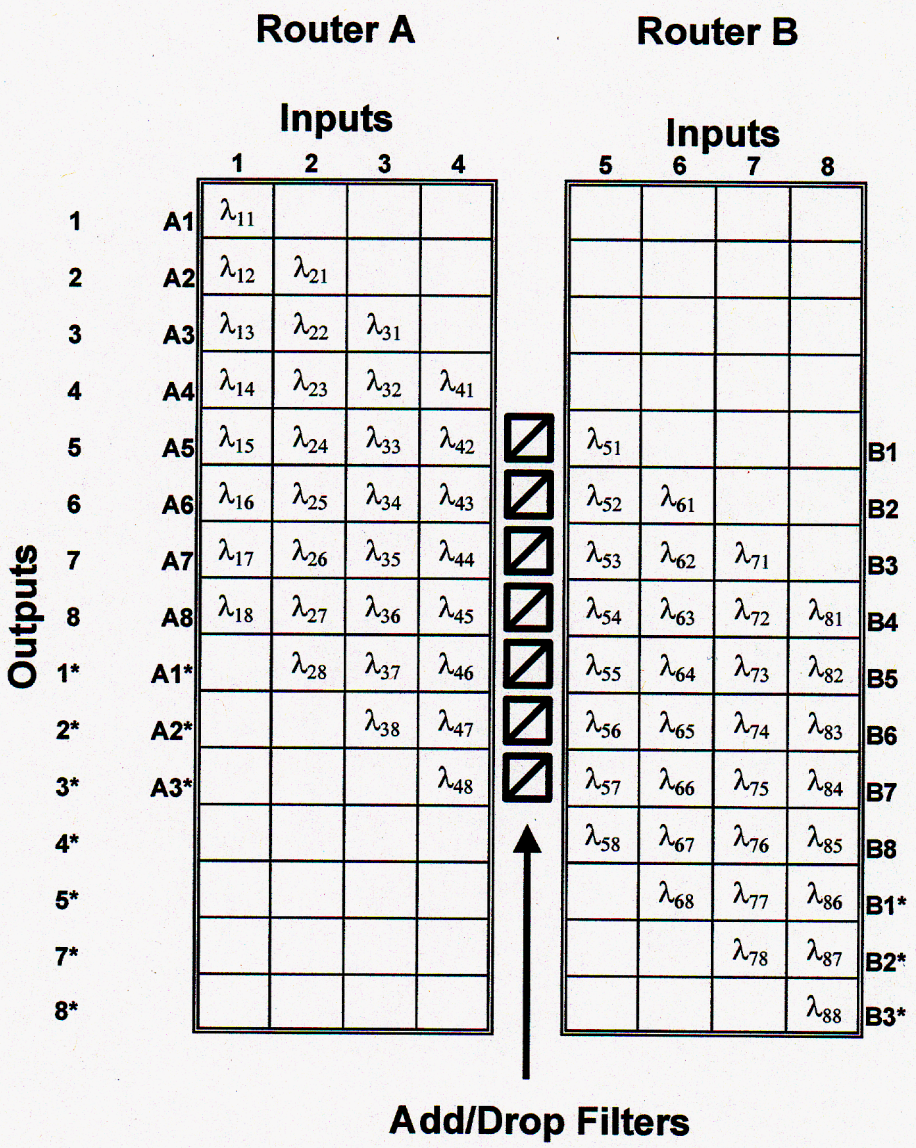


Figure 11. Routing table for an 8-wavlength system realized using 2 routers with 4 inputs each. When the outputs of routers A and B are combined using add/drop filters (as indicated) the resulting routing table is equivalent to that of an 8-input router. The output port labels on the left side of the figure correspond to the 8-input router and are provided for reference. The wavelengths are labeled for the 8-wavelength system as a whole.

When using multiple routers, it is not necessary that each of the routers have the same number of inputs, the only requirement is that the total number of input ports equals N. For example, in the 8-wavelength case, a router having 6 inputs and a router having 2 inputs could be used to achieve the full wavelength re-use.

Any number of routers (less than N) can be used to achieve the full routing functionality. In the extreme case, N routers having 1 input port and N output ports could be employed. Each of the multiple routers breaks up the full N-wavelength routing table along a column. When re-combining the outputs from multiple routers, N-1 filters are required at each column-break in order to preserve the wavelength routing of the N-wavelength system. Therefore, using M routers, M-1 column-breaks must be removed. This calls for (M-1)*(N-1) add/drop filters. Once the N-wavelength routing table is reproduced, a final set of N-1 filters is required to achieve full wavelength re-use as described earlier in this chapter. Thus, in order to route N wavelengths using M routers, a total of M*(N-1) add/drop filters are required.

With the 1.5 dB average insertion loss of the add/drop filters discussed in chapter 3, using more than 2 routers is not desirable because the insertion loss that different wavelengths experience becomes increasingly uneven. For example, consider output port 3 of an 8-wavelength system using two 4-input routers. In the output port $\lambda 1$ - $\lambda 3$ experience 1.5dB loss due to filters while $\lambda 4$ - $\lambda 8$ experience 3dB loss because these channels go through 2 filters. In order to use more than 2 routers to route N wavelengths, add/drop filters with much lower loss need to be developed. This could be accomplished by using a filter package that includes micro-optics.

A wavelength router for 16 wavelengths is feasible using multiple routers each having less than 16 input ports. The experiments discussed in this chapter have demonstrated that 12 fibers on a 250 μm pitch can be imaged with good uniformity and low loss (see Figure 9). Scaling to 16 requires a fast lens with a large diameter and a fixture that can hold 24 to 32 fibers in parallel (silicon v-grooves could serve this function). For example, an 8-input router carrying 16 wavelengths produces 23 output beams. Such a router requires 31 fibers to be imaged by the lens (8 inputs plus 23 outputs). Using 2 such routers along with 30 add/drop filter modules would facilitate full 16x16 interconnection with wavelength re-use. Alternatively, 4 routers having 4 inputs and 19 outputs (a total of 23 fibers in parallel) and 60 add/drop filters could be used to accomplish the routing - subject to the caveat above regarding the loss of the add/drop filters. Using, in the extreme case, a separate router for each bit line the design presented in this chapter could be scaled to as many parallel bit lines as desired. Therefore, the number of parallel bit lines is not a limiting factor in the scalability of the router for λ-Connect. For reliability reasons it is desirable to use the fewest number of components possible to accomplish the wavelength routing. This entails designing the lens and a fan-in/fan-out array to handle more bits in parallel. The design of these components is beyond the scope of this dissertation.

5.6 Implications of Wavelength Routing for λ-Connect

The preceding discussion and the experimental results presented in this chapter show that a grating based wavelength router could be implemented for a system such as λ-Connect. Figure 12 shows a schematic of the λ-Connect system using a wavelength

routed architecture rather than broadcast and select. The transmitters in this system would be identical to the λ-Connect transmitters. As seen in the figure, the band-pass filters of the broadcast and select architecture are not required in a wavelength routed architecture because the wavelength router makes 1 to 1 connections between inputs and outputs.

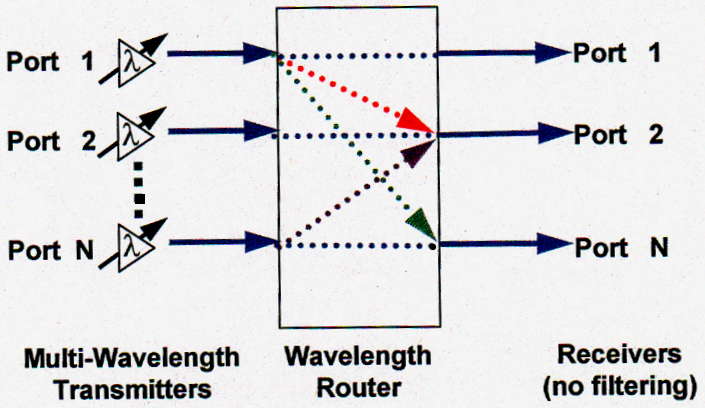


Figure 12. Schematic of a wavelength routed λ-Connect system. Each input has a unique mapping of wavelength vs. Output. The blue lines connecting inputs 1, 2, and n to outputs 1, 2, and n respectively illustrate the wavelength re-use.

Because each input uses a different wavelength to connect to a given output, no data collisions occur internally within the router; however, contention still arises when 2 inputs need to address the same output. This can be resolved by using a medium access control (MAC) protocol to decide which input port gets to connect to a given output (as in the broadcast and select architecture), or by de-multiplexing incoming signals at each receiver. The second option, while potentially allowing greater connectivity and throughput, may be significantly more expensive than using MAC.

The number of ports that can be fully interconnected (N inputs addressing the same N outputs) by a wavelength router is determined by the number of available system wavelengths. Therefore, with 16 system wavelengths, wavelength routing would enable

16x16 connectivity with a device insertion loss of 3 to 4 dB (after all optical components are optimized). In comparison, a 16x16 broadcast and select architecture would have a (best-case) insertion loss of approximately 14dB (12dB splitting loss plus 2dB excess loss). In this example wavelength routing clearly provides better overall power margin.

5.7 Summary

We have demonstrated a MMF compatible coarse WDM wavelength router using a diffraction grating and broad-band add/drop filters to achieve wavelength re-use. The scalability of this device to higher channel counts is ultimately limited by the requirements on the lens, the fact that N-1 filter modules are needed to fully route N wavelengths and the spatial fill factor (Δx) of the input/output fiber arrays. The device provides a new tool for WDM over MMF. Furthermore, this design enables a bit-parallel wavelength router by using multiple fibers in parallel for each input and output port. Coupled with multi-wavelength transmitter and receiver arrays such a device can provide high throughput, low latency optical interconnects with wavelength re-use. Because the device established 1 to 1 connections between input and output ports rather than distributing inputs to all output ports it provides more power margin than the broadcast and select approach of λ-Connect.

References for Chapter 5

[1] Tandon, V., Wilby, M., and Burton, F., "A Novel Upgrade Path for Transparent Optical Networks Based on Wavelength Reuse", INFOCOM '95. 14th Annual Joint Conference of the IEEE Computer and Communications Societies., Proceedings., IEEE , 1995 Page(s): 1308 -1315 vol.3.

[2] Dragone, C., "An NxN Optical Multiplexer Using a Planar Arrangement of Two Star Couplers", IEEE Phot. Tech. Lett. V3, 1991, pg. 812-815.

[3] Takahashi, H., Oda, K., Toba, H., Inoue, Y., "Transmission Characteristics of Arrayed Waveguide NxN Wavelength Multiplexer", IEEE J. Lightwave. Tech., V13, 1995, pg. 447-455.

[4] Brackett, C. A., "Dense Wavelength Division Multiplexing Networks: Principles and Applications", IEEE J. Select Areas Commun., V8, 1990, pg. 948-964.

[5] Churin, E. G., and Bayvel, P., "Design of Free-Space WDM Router Based on Holographic Concave Grating", IEEE Phot. Tech. Lett., V11, 1999, pg. 221-223.

[6] Laude, J. P. And Fessard, D., "Very Dense N*N Wavelength Routers Based on a New Diffraction Grating Configuration", in Proc. ECOC '97, V3, 1997, pg. 87-90.

[7] DeGroot, A. J., Deri, R. J., Haig, R. E., Patterson, F. G., and Dijali, S. P., "High Performance Parallel Processors Based on Star Coupled WDM Optical Interconnects", in Proc. 3rd conf. On Massively Parallel Processing Using OpticalInterconnects, 1996, pg. 62-69.

Chapter 6 Summary

The developments described in this thesis, together with other parts of the λ-Connect project demonstrate that wavelength division multiplexed parallel optical interconnects can offer a practical alternative to electronic interconnects for MPP systems. The use of multiple system wavelengths facilitates architectures with enhanced connectivity and source routing capabilities that can reduce hardware related problems such as hop latency and network congestion. The research performed for this dissertation demonstrates several key system components and technologies.

We have demonstrated thin film filter modules having robust, low-cost plastic packages with passive alignment and compatibility with MT ferrules and requiring no micro-optics. III-V semiconductors (at wavelength below the bandgap) provide high refractive indices ($n \sim 3 - 3.5$) to minimize insertion loss due to the large numerical aperture of MMF. The MC-FP filter design enables broad passbands, steep passband edges, and increased cross-talk rejection. The ability to fabricate MC-FP filters using standard III-V growth techniques (MOCVD, MBE) was demonstrated and fabricated filters compared very well with the original filter designs despite the challenging demands placed on the capabilities of the MOCVD system. By machining MT ferrules byte-wide 2-port and 3-port parallel optical filter modules were demonstrated. Particularly, in the 2-port configuration, a packaged 5-cavity filter module exhibited fiber-to-fiber center wavelength uniformity of better than 1 nm, 1.6 ± 0.6 dB insertion loss, 7.5 nm FWHM, and filter squareness of 0.46. This 2-port filter module meets the filter requirements for λ-Connect and could support up to 4 WDM channels with 10nm channel spacing at –23dB crosstalk from adjacent channels.

Lateral oxidation of high Aluminum content AlGaAs for filter applications was explored theoretically and experimentally. This application of lateral oxidation has received little attention in the literature to-date because the focus has been on VCSELs. We demonstrated a 3-cavity bandpass filter exhibiting >500 nm DBR stop-bands. This could enable CWDM systems with ~10nm channel spacings and more than 16 WDM channels, something that is not possible with the III-V semiconductor based filters alone.

We have shown that the passband of a MC-FP filter can be shifted with a post-growth lateral oxidation process. The theory for passband shifting filter design was developed and a 2-cavity filter with 54nm passband shift upon oxidation was demonstrated experimentally. It was found that the presence of the AlAs tuning layers tended to enhance the oxidation rate of adjacent low index layers (having 85% Aluminum fraction) through vertical oxidation. To alleviate this problem, thin oxidation barriers were employed between the tuning layers and the low index layers.

A study of variations in barrier thickness and composition was conducted to develop an understanding of the problem. The barrier thickness study indicated that 15% aluminum fraction barriers 25nm and thicker prevent oxidation for 90 minutes (at 425 ^{o}C) while 20nm thick barriers prevent oxidation for at least 60 minutes and may be effective for up to 90 minutes. The barrier composition study found that 15nm thick barriers with aluminum fractions up to 50% can prevent vertical oxidation for up to 60 minutes. Barriers with 70% aluminum fraction did not prevent vertical oxidation at all, in fact, the barriers themselves were oxidized. After 90 minutes 15nm thick barriers were penetrated regardless of their composition. SEM images reveal that the vertical oxidation across the barriers occurs at discrete points rather than uniformly across the interface. This suggests

that, at least for thick barriers, vertical oxidation cannot be treated as a simple linear enhancement of the lateral oxidation rate. In general, barrier layer penetration tends to increase with barrier layer aluminum content as seen in SEM images.

The aluminum oxide/$Al_{0.15}Ga_{0.85}As$ material pair has a reflectivity of ~42% per Bragg pair. A new filter structure, the oxide enhanced rejection filter, was designed to overcome limitations in filter design flexibility caused by this high reflectivity per Brag pair. The OER filter takes advantage of the large DBR stop-bands afforded by Aluminum oxide while allowing passband filter design flexibility afforded by using a lower index contrast material pair such as AlAs/$Al_{0.15}Ga_{0.85}As$.

Finally, a bit-parallel wavelength router suitable for coarse WDM over MMF was developed. Using a diffraction grating and lens combination along with 3-port add/drop filters full wavelength re-use is achieved. A 3x3 wavelength router with 2 parallel bit-lines per port and 37nm channel spacing was demonstrated. This is, to our knowledge, the first demonstration of a bit-parallel wavelength router. Scaling of the device to accommodate 16 wavelengths and an arbitrary number of parallel bit-lines was discussed. This device provides a new tool for WDM over multimode fiber. Because it establishes 1 to 1 connections between input and output ports a wavelength router exhibits better power margin than the broadcast and select architecture of λ-Connect.

Several of the experiments performed for this dissertation could benefit from further research. A full system demonstration of the λ-Connect link is the ultimate goal of the project, however, this requires additional development work in areas outside the scope of this dissertation (multi-wavelength transmitters for example). The 2-port filter module work has been carried as far as can be expected in a proof-of-principle project. Filter

response would be improved by using in-situ monitoring and feedback during filter growth. The uniformity of insertion loss between fibers within a filter module could be improved by using fiber polishers specifically designed to polish parallel fiber arrays. Such devices are commercially available and cost approximately $50,000. In actual production the approach of machining MT ferrules to accommodate filters is not practical because of the number of man-hours required. An MT-compatible module specifically designed for filters would be used to package the filters. To make such a module a master mold is needed. This does not require further technology development however it involves a one-time expense to develop the master mold. These improvements are really product development issues and, as such, they should be taken up when commercializing this technology.

The use of lateral oxidation for filters requires additional research. The high insertion losses (9 to 12 dB) exhibited by our filters after oxidation indicate the need for process development. The lateral oxidation process itself is fairly new and a comprehensive model of all of the factors that can affect the oxidation has not yet been established. Our lateral oxidation setup could be improved by using a mass flow controller on the nitrogen gas line to accurately quantify and precisely control the oxidation conditions. A multi-zone oxidation furnace and pyrolytic steam generation to produce ultra-pure steam may also improve process repeatability. Further characterization of the barrier layers and design and testing of a new set of passband shifting filters using information gained from the barrier studies would improve the performance of the passband shifting filters. Also, a different photo-mask with alternate geometries for the oxidation (for example using mesas instead of a network of holes) may help mitigate the unwanted oxidation of layers

other than the tuning layers. Use of different materials for the low index layers in passband shifting filter designs should also be explored. materials that can be epitaxially grown and lattice-matched with $Al_xGa_{1-x}As$ but having low or zero aluminum fractions would be ideal candidate materials.

The wavelength router could be improved by using 3-port filters that are specifically designed for this application. Also, as indicated in Chapter 5, optimizing the optics of the system will reduce the insertion loss and improve the channel to channel and port-to-port uniformity of the device. Using a specialized fan-in/out structure to hold the input and output fibers closer together may also help reduce the device insertion loss. Ultimately, a monolithic design where the lens and diffraction grating are integrated into a solid piece of glass (or other material) will yield a stable, robust design that can easily be deployed in real systems.

The components and technologies described in this dissertation are key components for multi-wavelength parallel optical interconnects. This work provides the necessary foundation for development byte-wide filter modules and passive wavelength routers and it is my hope that these components will eventually become commercially available products that enhance the performance of future generation computing systems.

Appendix A Matlab Scripts for MC-FP Filter simulations

```
%12/10/99 - This is the 5 cavity filter simulation that models
%the filter L2074 which we grew.
%
%
%Rajesh Patel 4/26/99 written in preparation for Oral Exams

clear all;
R1 = 0.9; %R1 = 0.64263;
r1 = sqrt(R1);
R2=R1; %R2 = 0.96963;
R3=R1; %R3 = 0.98436;
R4 = R3;
R5 = R2;
R6 = R1;
r2 = sqrt(R2);r3 = sqrt(R3);r4 = r3; r5 = r2; r6 = r1;
T1 = 1-R1;
T2=1-R2;T3=1-R3;T4=T3;T5 = T2;T6=T1;
t1 = sqrt(T1);t2 = sqrt(T2);t3 = sqrt(T3);t4 = t3;t5 = t2; t6 = t1;


light = 3e17;
cwl = 800;
wo = 2*pi*light/cwl;
lambda = [700:0.1:900];
omega = 2*pi*light./lambda;
detun = wo-omega;

l1 = 4*401;
l2 = 4*400.5;
l3 = 4*400;
l4 = 4*400.5;
l5 = 4*401;

g1 = exp(-2*j.*omega/light*l1);
g2 = exp(-2*j.*omega/light*l2);
g3 = exp(-2*j.*omega/light*l3);
g4 = exp(-2*j.*omega/light*l4);
g5 = g1;

temp = ones(1,length(lambda));
p1 = r1*temp;
p2 = r2*temp;
p3 = r3*temp;
p4 = r4*temp;
p5 = r5*temp;
p6 = r6*temp;
t11 = t1*temp;
t22 = t2*temp;
t33 = t3*temp;
t44 = t4*temp;
t55 = t22;t66=t11;
[p12,t12] = fabry(p1,p2,t11,t22,g1);
[p23,t23] = fabry(p12,p3,t12,t33,g2);
```

```
[p34,t34] = fabry(p23,p4,t23,t44,g3);
[p45,t45] = fabry(p34,p5,t34,t55,g4);
[p56,t56] = fabry(p45,p6,t45,t66,g5);
%Transmission coefficients from each equivalent reflector

T12 = t12.*conj(t12);
T23 = t23.*conj(t23);
T34 = t34.*conj(t34);
T45 = t45.*conj(t45);
T56 = t56.*conj(t56);

Tmax = max(T56);
%Phase due to reflection from the equivalent reflectors
phr34 = angle(p34)/pi;
pht34 = angle(t34)/pi;
phr12 = angle(p12)/pi;
phr23 = angle(p23)/pi;
phr34 = angle(p34)/pi;
phr45 = angle(p45)/pi;


%round-trip phase for each equivalent cavity
phg1 = angle(g1)/pi;
phg2 = angle(g2)/pi;
phg3 = angle(g3)/pi;
phg4 = angle(g4)/pi;
phg5 = angle(g5)/pi;

%refelction coefficeint from each equivalent reflector
R12 = p12.*conj(p12);
R23 = p23.*conj(p23);
R34 = p34.*conj(p34);
R45 = p45.*conj(p45);
R56 = p56.*conj(p56);

%reflection coefficient from each "real" reflector
R1 = R1*temp; R2 = R2*temp; R3 = R3*temp;
R4 = R4*temp; R5 = R5*temp; R6 = R6*temp;
%add up phases for two-cavity equivalent and the round-trip for the
%last cavity
%phtemp = phr45+phg5;
phtemp = phr23+phr23;
%make a vector to draw in the zero-line
zeroline = zeros(1,length(temp));

%plot transmission through each successive equivalent cavity
%g = figure;
%plot(lambda,T12,lambda,T23,lambda,T34);

%plot the Total transmission and the phase of the last equivalent
%reflector+cavity
%h = figure;
%subplot(2,1,1);
%plot(lambda,T34);
%title('Transmission');
%subplot(2,1,2);
%plot(lambda,phtemp,lambda,zeroline);
```

```
%set(gca,'YLIM',[-0.2,0.2]);
%title('Phase');
M1 = num2str(R1(1));M2 = num2str(R2(1));
M3 = num2str(R3(1));M4 = num2str(R4(1));
g = figure;
subplot(2,2,1);plot(lambda,phr23,lambda,phg3);
set(gca,'YLIM',[-0.5,0.5]);
title('Individual Phases');
phtemp = phtemp+phg3;
subplot(2,2,2);plot(lambda,phtemp,lambda,zeroline,'m');
set(gca,'YLIM',[-0.5,0.5]);
title('Total Phase');
subplot(2,2,3);plot(lambda,R23);
rstring = ['R1 = ',M1,', R2 = ',M2,', R3 = ',M3];
set(gca,'YLim',[0,1]);
title(rstring);
phscale = phtemp+1.01;
subplot(2,2,4);plot(lambda,T56,lambda,R23,lambda,phscale);
set(gca,'YLim',[0,1.2]);
tprompt = ['Transmission, max = ',num2str(Tmax)];
title(tprompt);
```

```
%MCFPscript
%Matlab script for multi-cavity filters
%To obtain high transmission, the reflectivity of the "effective"
%mirror and the single mirror must be equal.  Also, total phase of
%the E-field (round-trip + due to reflection from effective mirror)
%must equal zero (or pi?)
%Only when these two conditions are met do we obtain unity
%transmission.  By adjusting the reflectivities we can play with the
%exact locations of phase-match and reflectivity match - giving us
%the ability to tailor the pass-band of the filters.
%
%Note: when considering the phase condition, we only look at the
%round-trip phase for the equivalent cavity and the equivalent
%reflector - the phase information from all of the other "nested"
%cavities is included in the phase shift on reflection term.
%
%
%Rajesh Patel 4/26/99 written in preparation for Oral Exams
%MODIFIED 6/23/99 for filter studies -RRP

clear all;

%define constants

light = 3e17;
cwl = 800;
wo = 2*pi*light/cwl;
lambda = [650:0.5:1050];
omega = 2*pi*light./lambda;
detun = wo-omega;

option = 2;  %switch to change which parameter is adjusted
             %1 = Changing reflectivities , 2 = Changing Cav. Lengths,
3 = # cavities
%Make a loop to adjust either R or L2
num_steps = 3;
nsteps = num_steps-1;   %adjust for the fact that the loop starts at
zero
Rs = 0.95;
Re = 0.65;
ncav = 5; %number of cavities for option 3;
%Switches to change the reflectivities
dR = (Rs-Re)/nsteps;
%dR = 0;
%Switches to change the cav. length
l2start = cwl/2;
l2end = 4*l2start;
%l2end = l2start+40;
l2list =[cwl/2;cwl;2*cwl]; % added this bit on 8/23/2000 for a better
graph

if (option==2)
   dl2 = (l2end-l2start)/nsteps;
else
   dl2 = 0;
end;
```

```
for x = 0:1:nsteps;
   if (option==1)
      rv = Rs-(x*dR);
   else;
      rv = 0.70;
   end;
      R1 = rv;
r1 = sqrt(R1);
%R2 = rv;
%R3 = rv;
%R4 = 0.70;
rv2=0.91;
R2=rv2;
R3=rv;
r2 = sqrt(R2);r3 = sqrt(R3);
%r4 = sqrt(R4);
T1 = 1-R1;
T2=1-R2;T3=1-R3;
%T4=1-R4;
t1 = sqrt(T1);t2 = sqrt(T2);t3 = sqrt(T3);
%t4 = sqrt(T4);


l1 = cwl/2;  %changed from cwl/4 to cwl/2 12/8/99
%l2base = 400;
if (option==2)
   %l2 = l2start+x*(dl2) old code - changed to below for a better graph
    l2=l2list(x+1);
   if (x==0)
      l2tally = l2;
   else
      l2tally=[l2tally;l2];
   end;
else
   l2 = cwl/2; %changed from cwl/4 to cwl/2 12/8/99
end;
   %l3 = 400;
%l4 = 400;

g1 = exp(-2*j.*omega/light*l1);
g2 = exp(-2*j.*omega/light*l2);
%g3 = exp(-2*j.*omega/light*l3);
%g4 = exp(-2*j.*omega/light*l4);

temp = ones(1,length(lambda));
p1 = r1*temp;
p2 = r2*temp;
p3 = r3*temp;
%p4 = r4*temp;
t11 = t1*temp;
t22 = t2*temp;
t33 = t3*temp;
%t44 = t4*temp;

if (option==3)
   if x==0
```

```
      [pc,tc] = fabry(p1,p2,t11,t22,g1);
            p = pc;t = tc;
      for y = 2:ncav
            [pc,tc] = fabry(pc,p2,tc,t22,g2);
            %p = [p;pc];t=[t;tc];
             p = pc; t = tc;
         end; %for loop
      end; %if x == 0
 end; % if option 3
[p12,t12] = fabry(p1,p2,t11,t22,g2);
[p23,t23] = fabry(p12,p3,t12,t33,g2);
%[p34,t34] = fabry(p23,p4,t23,t44,g4);

%Transmission coefficients from each equivalent reflector

T12 = t12.*conj(t12);
T23 = t23.*conj(t23);
%T34 = t34.*conj(t34);

%Phase due to reflection from the equivalent reflectors
%phr34 = angle(p34)/pi;
%pht34 = angle(t34)/pi;
phr12 = angle(p12)/pi;
phr23 = angle(p23)/pi;

%round-trip phase for each equivalent cavity
phg1 = angle(g1)/pi;
phg2 = angle(g2)/pi;
%phg4 = angle(g4)/pi;

%refelction coefficeint from each equivalent reflector
R12 = p12.*conj(p12);
R23 = p23.*conj(p23);
%R34 = p34.*conj(p34);

%reflection coefficient from each "real" reflector
R1 = R1*temp;
R2 = R2*temp;
R3 = R3*temp;
%R4 = R4*temp;

%add up phases for two-cavity equivalent and the round-trip for the
%last cavity
%phtemp = phr23+phg4;
phtemp = phr12+phg2;
st = T23;
st = sort(st);
mx1 = st(401);
mx2 = st(400);
[dummy,a] = find(T23==mx1);
[dummy,b] = find(T23==mx2);
delt = abs((lambda(a)-lambda(b)));
%the find section below is for calculating squareness - added 9/1/2000
fr = 2;
sd = abs(1/fr - T23); % find the 3-db width
sd2 = sort(sd);
```

```
mx1 = sd2(1);
mx2 = sd2(2);
[dummy,a] = find(sd==(mx1));
[dummy,b] = find(sd==(mx2));
w3 = abs((lambda(a)-lambda(b)));
ctr = 2;
while (1==1)% loop to find the right wavelength for 20 db width
mx2=sd2(ctr);
ctr=ctr+1;
[dummy,b] = find(sd==(mx2))
w3 = abs((lambda(a)-lambda(b)));
if  (sign(lambda(a)-cwl)~=sign(lambda(b)-cwl))  %should  get  one  #  less
than cwl and 1 greater
      break;

end;
end;%while loop

%repeat for the 20db width
fr = 100;
s2d = abs((1/fr - T23)); % find the 3-db width

s2d2 = sort(s2d);
mx1 = s2d2(1);
%mx2 = s2d2(2)
[dummy,a] = find(s2d==(mx1));
ctr=2;
while (1==1)% loop to find the right wavelength for 20 db width
mx2=s2d2(ctr)
ctr=ctr+1
[dummy,b] = find(s2d==(mx2));
w20 = abs((lambda(a)-lambda(b)));
if  (sign(lambda(a)-cwl)~=sign(lambda(b)-cwl))  %should  get  one  #  less
than cwl and 1 greater
   if(w20>w3)
      break;
   end;
end;

end;%while loop

if (x==0)
   w3db=w3;
   w20db=w20;
else
   w3db=[w3db;w3];
   w20db=[w20db;w20];
end

if (x==0)
   rtphase = phtemp;
   R12v = R12;
   refphase = phr12;
   refvar = R3;
   T23v = T23;
   cavphase = phg2;
   cavlen = l2;
```

```
%   delpeak = delt;
else
   rtphase = [rtphase;phtemp];
   R12v = [R12v;R12];
   refphase = [refphase;phr12];
   refvar = [refvar;R3];
   T23v = [T23v;T23];
   cavphase = [cavphase;phg2];
   cavlen = [cavlen,l2];
%   delpeak = [delpeak,delt];
end;

end; %for loop
%make a vector to draw in the zero-line
zeroline = zeros(1,length(temp));
tm = max(T23v);
Tmax = max(tm);
%plot transmission through each successive equivalent cavity
g = figure;
%plot(lambda,T12,lambda,T23,lambda,T34);
plot(lambda,T23v);
%g = figure;
%plot(lambda,T12,lambda,T23);

%plot the Total transmission and the phase of the last equivalent
%reflector+cavity
%h = figure;
%subplot(2,1,1);
%plot(lambda,T34);
%title('Transmission');
%subplot(2,1,2);
%plot(lambda,phtemp,lambda,zeroline);
%set(gca,'YLIM',[-0.2,0.2]);
%title('Phase');

%g = figure;
%subplot(2,2,1);plot(lambda,phr23,lambda,phg4,lambda,phr12,lambda,phg2)
;
%title('Individual Phases');
%subplot(2,2,2);plot(lambda,phtemp,lambda,zeroline,'m');
%title('Total Phase');
%subplot(2,2,3);plot(lambda,R23,lambda,R4,lambda,R12);
%title('Reflectivities');
%subplot(2,2,4);plot(lambda,T34);
%title('Transmission');

if (option==3)
   Tcav = t.*conj(t);
   Tcav=T23v;

%Rcav = p.*conj(p);
phr = angle(p)/pi;
%phcav =
M1 = R1(1);M2 = R2(1);M3 = R3(1);
g = figure;
subplot(2,2,1);plot(lambda,refphase,lambda,cavphase);
title('Individual Phases');
```

```
subplot(2,2,2);plot(lambda,rtphase,lambda,phr,lambda,zeroline,'y');
title('Total Phase');
subplot(2,2,3);plot(lambda,R12v,lambda,refvar);
rstring = ['R1 = ',num2str(M1),', R2 = ',num2str(M2),', R3 =
',num2str(M3)];
set(gca,'YLim',[0,1]);
title(rstring);
subplot(2,2,4);plot(lambda,Tcav);
set(gca,'YLim',[0,1]);
tprompt = ['Transmission, max = ',num2str(Tmax)];
title(tprompt);
end;
%h = figure;
%plot(lambda,phr23,lambda,phg4,lambda,zeroline,lambda,phtemp);
```

Appendix B The Transfer Matrix Method

The transfer matrix method (TMM) is a standard method used to solve Maxwell's equations for a multi-layer sequence. Since it solves Maxwell's equations, it provides very accurate predictions of filter behavior as seen in Chapters 2 and 3 of this dissertation. The BraggModeller program computes the transmission and reflection properties of thin film filters using the TMM. We provide here a brief outline of the TMM for the sake of completeness.

Maxwell's equations are:

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{1}$$

$$\nabla \times \vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t} \tag{2}$$

$$\nabla \cdot \vec{D} = \rho \tag{3}$$

$$\nabla \cdot \vec{B} = 0 \tag{4}$$

The constitutive relations are:

$$\vec{J} = \sigma \vec{E} \tag{5}$$

$$\vec{D} = \varepsilon \vec{E} \tag{6}$$

$$\vec{B} = \mu \vec{H} \tag{7}$$

We shall assume that $\mu = \mu_o$, since this is valid for the materials of interest in thin film filters. Furthermore, we restrict the discussion to lossless, homogeneous, isotropic, dielectric media, with no sources. In this case, we obtain the wave equation for the electric field by taking the curl of (1) and substituting on the right hand side with (2), (6) and (7).

$$\nabla^2 \vec{E} = \mu\varepsilon \frac{\partial^2 \vec{E}}{\partial^2 t} \tag{8}$$

Similarly, the wave equation for the magnetic field is obtained by taking the curl of (2) and substituting on the right hand side using (1), (6) and (7).

$$\nabla^2 \vec{H} = \mu\varepsilon \frac{\partial^2 \vec{H}}{\partial^2 t} \tag{9}$$

These equations have well-known plane wave solutions of the form:

$$\vec{E}(\vec{r},t) = Ee^{i(\vec{k}\cdot\vec{r}-\omega t)} \tag{10}$$

Single interface:

Let the x-z plane be the plane of incidence, with the x-direction normal to the interface. On both sides of the interface we consider 2 plane waves, one traveling towards the interface, having wave vectors:

$$k_{right} = k_o \kappa_i \hat{x} + k_o \beta \hat{z} \tag{11}$$

$$k_{left} = -k_o \kappa_i \hat{x} + k_o \beta \hat{z} \tag{12}$$

Here, the term k_o is the wavevector in free space ($k_o=2\pi/\lambda$). The terms k_{right} and k_{left} are the wave-vectors for right and left travelling waves in medium 'i'. These satisfy the condition: $|k_{right}| = n_i|k_o|$, and the same for k_{left}. The terms κ_i and β satisfy: $|k_{right}|^2 = (\kappa_i^2 + \beta^2)$. If the angle in medium 'i' is θ_i then we have:

$$\kappa_i = n_i \cos(\theta_i) \tag{13}$$

$$\beta = n_i \sin(\theta_i) \tag{14}$$

From Snell's law we know that β will be the same in all media and therefore we need not specify it with a subscript.

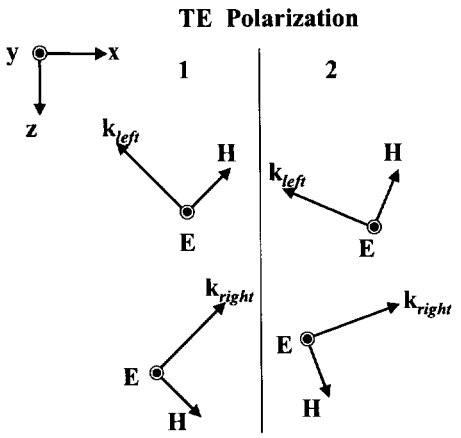


Figure 1. Single Interface, TE Polarization

Take the TE case, with the electric field polarized in the y-direction (see Figure 1):

$$E_1 = \left[A_1 e^{-jk_o\kappa_1 x} + B_1 e^{+jk_o\kappa_1 x}\right] e^{-jk_0\beta z}\,\hat{y} \tag{15}$$

$$E_2 = \left[A_2 e^{-jk_o\kappa_2 x} + B_2 e^{+jk_o\kappa_2 x}\right] e^{-jk_0\beta z}\,\hat{y} \tag{16}$$

The tangential components of the electric and magnetic fields must be continuous at the interface (taken to be at x=0) therefore:

$$A_1 + B_1 = A_2 + B_2 \tag{17}$$

$$\kappa_1 A_1 - \kappa_1 B_1 = \kappa_2 A_2 - \kappa_2 B_2 \tag{18}$$

In (18) we use (1) to find the magnetic field in terms of the electric field. Writing (17) and (18) in matrix form we have:

$$\begin{pmatrix} E_1 \\ H_1 \end{pmatrix}_{tang} = \begin{pmatrix} E_2 \\ H_2 \end{pmatrix}_{tang} \Rightarrow \begin{pmatrix} 1 & 1 \\ \kappa_1 & -\kappa_1 \end{pmatrix}\begin{pmatrix} A_1 \\ B_1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ \kappa_2 & -\kappa_2 \end{pmatrix}\begin{pmatrix} A_2 \\ B_2 \end{pmatrix} \tag{19}$$

The subscript '*tang*' in (19) indicates that only the tangential components of the electric and magnetic fields are being considered. We define the scattering matrix of an interface as:

$$S_i \equiv \begin{pmatrix} 1 & 1 \\ \kappa_i & -\kappa_i \end{pmatrix} \tag{20}$$

equation (19) can then be re-written as:

$$S_1 \begin{pmatrix} A_1 \\ B_1 \end{pmatrix} = S_2 \begin{pmatrix} A_2 \\ B_2 \end{pmatrix} \Rightarrow \begin{pmatrix} A_1 \\ B_1 \end{pmatrix} = S_1^{-1} S_2 \begin{pmatrix} A_2 \\ B_2 \end{pmatrix} \tag{21}$$

Thus, (21) relates the electric field across an interface between materials 1 and 2.

From the form of the electric field, we can relate the field at a point x to the field at a point x_o within a medium:

$$E_1(x) = \left[A_1(x_0)e^{-jk_o\kappa_1(x-x_0)} + B_1(x_0)e^{+jk_o\kappa_1(x-x_0)}\right]e^{-jk_0\beta z}\hat{y} \tag{22}$$

This gives us a propagation matrix within a medium:

$$\begin{pmatrix} A_1(x) \\ B_1(x) \end{pmatrix} = \begin{pmatrix} e^{-jk_o\kappa_1(x-x_0)} & 0 \\ 0 & e^{+jk_o\kappa_1(x-x_0)} \end{pmatrix} \begin{pmatrix} A_1(x_0) \\ B_1(x_0) \end{pmatrix} \tag{23}$$

For a layer of thickness l, we define the propagation matrix as:

$$P_1 \equiv \begin{pmatrix} e^{-jk_o\kappa_1 l} & 0 \\ 0 & e^{+jk_o\kappa_1 l} \end{pmatrix} \tag{24}$$

Multilayer Sequence

Now consider a sequence of alternating layers 1 and 2

incident medium |1|2|1|2|1|...|1||2|1|2|1| substrate medium

Since (21) related the electric field across an interface, and (24) propagates the field across a layer, we can use these equations repeatedly in series to relate the electric field across the multilayer sequence.

There is no reflected wave in the substrate medium, therefore the electric field there is:

$$E = \begin{pmatrix} A_s \\ 0 \end{pmatrix} \tag{25}$$

If there are a total of N layers then the field just inside the last layer is:

$$\begin{pmatrix} A_N \\ B_N \end{pmatrix} = S_N{}^{-1} S_{subtrate} \begin{pmatrix} A_s \\ 0 \end{pmatrix} \tag{26}$$

The field just inside the N-1th layer is found by propagating the result of (26) back through layer N and across the N-1/N interface. Since we propagate backwards, we must use the inverse of the propagation matrix. It is trivial to show that the inverse of (24) is given by replacing each component by its complex conjugate.

$$\begin{pmatrix} A_{N-1} \\ B_{N-1} \end{pmatrix} = S_{N-1}{}^{-1} S_N P_N{}^{-1} \begin{pmatrix} A_N \\ B_N \end{pmatrix} \tag{27}$$

substituting from (26) into (27) we find:

$$\begin{pmatrix} A_{N-1} \\ B_{N-1} \end{pmatrix} = S_{N-1}{}^{-1} S_N P_N{}^{-1} S_N{}^{-1} S_{subtrate} \begin{pmatrix} A_s \\ 0 \end{pmatrix} \tag{28}$$

Equation (27) gives the general relationship between the electric field just inside layer i (at the i/i+1 interface) and the field just inside layer i+1 (at the i+1/i+2 interface). Thus, to relate the electric field incident on the multilayer to the transmitted field equation (27) is applied recursively for each layer. Thus:

$$\begin{pmatrix} A_{inc} \\ B_{inc} \end{pmatrix} = S^{-1}{}_{incident} S_1 P_1^{-1} S_1^{-1} S_2 P_2^{-1} S_2^{-1} \bullet\bullet\bullet S_{N-1}{}^{-1} S_N P_N{}^{-1} S_N{}^{-1} S_{subtrate} \begin{pmatrix} A_s \\ 0 \end{pmatrix} \tag{29}$$

Notice that, for each layer in the multilayer sequence, there is a sequence of matrices in (29) of the form:

$$S_i P_i^{-1} S_i^{-1} \tag{30}$$

All three of these matrices only depend on the properties of layer i. The product of these matrices is:

$$M_i = \begin{pmatrix} \cos(\phi_i) & \frac{j\sin(\phi_i)}{\kappa_i} \\ j\kappa_i \sin(\phi_i) & \cos(\phi_i) \end{pmatrix} \tag{31}$$

where the term ϕ_i is given by:

$$\phi_i = k_o \kappa_i l_i = \frac{2\pi n_i \cos(\theta_i)}{\lambda_o} l_i \tag{32}$$

Here, λ_o is the wavelength in free space, n_i is the refractive index of the layer, l_i is the thickness of the layer, and θ_i is the angle of the beam inside the layer.

The matrix M_i is called the characteristic matrix of layer i. Re-writing (29) using characteristic matrices we have:

$$\begin{pmatrix} A_{inc} \\ B_{inc} \end{pmatrix} = S^{-1}{}_{incident} M_1 M_2 M_3 \bullet\bullet\bullet M_{N-1} M_N S_{substrate} \begin{pmatrix} A_s \\ 0 \end{pmatrix} = M_{Total} \begin{pmatrix} A_s \\ 0 \end{pmatrix} \tag{33}$$

Thus we see that the product of the characteristic matrices for each of the layers mathematically represents the multilayer sequence. The two outer matrices represent properties of the incident and substrate media. The matrix M_{Total} is a 2x2 matrix that represent the transfer function of the multi-layer sequence (hence the name transfer matrix method). We can now find the fields incident and reflected from the multilayer sequence in terms of the transmitted field:

$$\begin{pmatrix} A_{inc} \\ B_{inc} \end{pmatrix} = \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} \begin{pmatrix} A_s \\ 0 \end{pmatrix} \tag{34}$$

$$A_{inc} = M_{11} A_s \quad \text{and} \quad B_{inc} = M_{21} A_s \tag{35}$$

Recall that B_{inc} is actually the wave travelling away from the multilayer in the incident medium, meaning that it is the field reflected from the multilayer. Expressing (35) in terms of the incident field we have:

$$A_s = \frac{1}{M_{11}} A_{inc} \text{ and } \quad B_{inc} = \frac{M_{21}}{M_{11}} A_{inc} \tag{36}$$

Equation (36) gives us the amplitude transmissivity and reflectivity of the multilayer sequence. The reflectance and transmittance are the square magnitudes of the quantities in (36) i.e.:

$$R = \left| \frac{M_{21}}{M_{11}} \right|^2 \quad \text{and} \quad T = \left| \frac{1}{M_{11}} \right|^2 \tag{37}$$

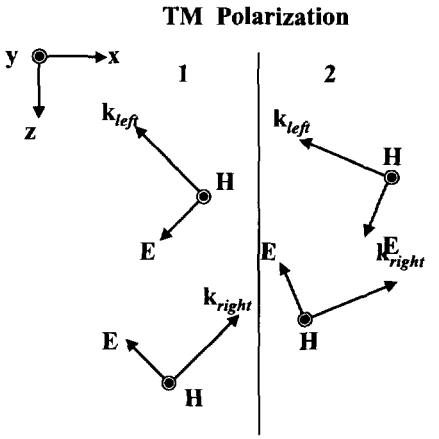


Figure 2. Single Interface, TM polarization

For the TM case (see Figure 2), the formulation follows exactly the same argument. However, equations (15) and (16) now represent the magnetic field instead of the electric

field. This means that equation (17) represents the tangential components of the magnetic field at the interface. In analogy to equation (18) we write the electric field in terms of the magnetic field. Recalling that we have plane wave fields that are harmonic in time, we re-write (2) as follows:

$$\nabla \times \vec{H} = j\omega\varepsilon\vec{E} \Rightarrow \vec{E} = \frac{-j}{\omega\varepsilon}\nabla \times \vec{H} \tag{38}$$

Note that the term ε in the denominator can be written as:

$$\varepsilon = \varepsilon_r \varepsilon_0 = n^2 \varepsilon_0 \tag{39}$$

Where ε_o is the permittivity of free-space, ε_r is the relative petmittivity of the medium, and n is the refractive index. Equating the tangential components of the electric field at x=0 then gives us:

$$\frac{\kappa_1}{n_1^2}A_1 - \frac{\kappa_1}{n_1^2}B_1 = \frac{\kappa_2}{n_2^2}A_2 - \frac{\kappa_2}{n_2^2}B_2 \tag{40}$$

We can now write (17) and (40) in matrix form for the TM case:

$$\begin{pmatrix} E_1 \\ H_1 \end{pmatrix}_{\tan g} = \begin{pmatrix} E_2 \\ H_2 \end{pmatrix}_{\tan g} \Rightarrow \begin{pmatrix} \kappa_1/n_1^2 & -\kappa_1/n_1^2 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} A_1 \\ B_1 \end{pmatrix} = \begin{pmatrix} \kappa_2/n_2^2 & -\kappa_2/n_2^2 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} A_2 \\ B_2 \end{pmatrix}$$

$$\Rightarrow \quad S_i \equiv \begin{pmatrix} \frac{\kappa_i}{n_i^2} & -\frac{\kappa_i}{n_i^2} \\ 1 & 1 \end{pmatrix} \tag{41}$$

In (41) we organize the matrix such that the equations arising from the electric field are in the top row and those from the magnetic field in the bottom. This maintains consistency with equation (20) from the TE case. As seen below this allows us to create a unified formulation for the transfer matrices for both polarizations:

The propagation matrix is unchanged since it does not depend upon the polarization. Thus the behavior of the multilayer sequence can be found for TM polarization following the same argument outlined for the TE case. The scattering matrices (and their inverses) have slightly different form than for the TE case, however, in analogy to equation (30), we can define a characteristic matrix for each layer for TM polarization. The characteristic matrix for layer i has the form:

$$M_i = \begin{pmatrix} \cos(\phi_i) & \dfrac{j\sin(\phi_i)}{\left(n_i^{\,2}/\kappa_i\right)} \\ \left(n_i^{\,2}/\kappa_i\right) j\sin(\phi_i) & \cos(\phi_i) \end{pmatrix} \tag{42}$$

Unified formulation for TE and TM polarizations:

Comparing (31) and (42) we see that the characteristic matrix for a layer can be expressed as:

$$M_i = \begin{pmatrix} \cos(\phi_i) & \dfrac{j\sin(\phi_i)}{\delta_i} \\ j\delta_i \sin(\phi_i) & \cos(\phi_i) \end{pmatrix} \tag{43}$$

where

for TE polarization: $\delta_i = n\cos(\theta_i)$ and for TM polarization: $\delta_i = \dfrac{n}{\cos(\theta_i)}$

Note that for normal incidence the matrix is the same for both polarizations.

Using (43) we can find the transfer matrix of a multilayer sequence for either polarization. We must remember to use the appropriate form of the scattering matrices for the incident and substrate media in order to calculate the reflection and transmission properties of a multilayer sequence correctly.

To calculate the properties of a filter with a computer, each layer in the filter is represented by its characteristic matrix. The computer can then rapidly calculate the reflection and transmission properties as a function of wavelength and incident angle. The BraggModeller program implements the theory outlined above to calculate the performance of multilayer thin film filters.

Appendix C Java Code Listings for BraggModeller

The BraggModeller program is an object-oriented program that performs transfer matrix method calculations for thin film filter design. As such, it consists of many class files for different objects in the program as well as supporting objects that are not directly part of the BraggModeller program. In this appendix I provide a complete list of the codes that I have written for BraggModeller during the course of this dissertation research. Three sample input files for BraggModeller are also provided at the end of this appendix. I have also written several other codes for diagnostic and test purposes that are omitted for brevity. The three main classes for BraggModeller are BraggModeller, StackManager and StackCalculator.

This appendix contains listings for the following classes:

BraggModeller
StackManager
StackCalculator
BraggLayer
BoundaryLayer
EpiLayer
GaBraggLayer
IncidentLayer
InGaPBraggLayer
SubstrateLayer
IndexCalc
InGaP_Index
LayerVendor
Stackable
HashKey
java_code.utilities.Keyboard
java_code.utilities.KeyboardReader
java_code.GUI.Plotter
java_code.GUI.QuickGuiwindow
java_code.mathematics.Matrix
java_code.mathematics.ComplexNumber
java_code.modelling.Model
java_code.modelling.Model2
java_code.configurable

I have organized my support codes into a generic package called java_code. Under java_code there are 4 sub-packages: GUI, utilities, mathematics, and modelling.

As mentioned in Chapter 2, when this project began Java was fairly new and very few utilities were yet available for the language. Therefore, I wrote my own classes to represent complex numbers and matrices. The matrix code is not optimized in any way, but for the 2x2 matrices required for the transfer matrix method optimizations would not make any difference in performance. These classes are in the java_code.mathematics sub-package. The java_code.utilites package contains a public class called Keyboard. This class provides a simple way to obtain user input using a few standard methods. Package java_code.GUI contains objects that provide standard graphical user interface functions that are commonly used for modelling codes. Package java_code.modelling contains the classes Model and Model2 as well as the interface Configurable. These are intended to be higher level classes that contain the generic functionality that all kinds of models (not just BraggModeller) would require. For example, methods to get file names and create input files, methods to assign values to variables based on data from text files etc. The configurable interface was never implemented, but the concept was to have a generic interface for all different types of models. This way a single program could be written that could be the handler for many different types of models.

In addition to the codes listed here, the BraggModeller uses a package called ptplot to provide plotting of the filter transmission data as they are being calculated. This package was written by Edward A Lee and Christopher Hylands of the EECS department at U.C. Berkeley as part of the Ptolmey project. As of this writing, Ptplot is available for download and public use from http://ptolemy.eecs.berkeley.edu/java/ptplot/ .

```
import java.io.*;
```

```
import java.text.NumberFormat;
import java_code.utilities.*;
import java_code.modelling.*;
import java_code.gui.*;

/**
* This is the shell class for setting up thin film filter runs.
* To run this Application type the following at the command line:<p>
* <b>c:>java BraggModeller</b></p>
* This will start the application.  The application will prompt the
user for the
* name of the input file.  This should be a text file which has the
congifuration
* info for the model run.
*<p>The configuration information includes:
* <UL>
*    <LI>starting wavelength
*    <LI>ending wavelength
*    <LI>wavelength increment
*    <LI>starting angle
*    <LI>ending angle
*    <LI>angle increment
*    <LI>filter design angle (that is, the angle at which the thin film
filter is designed to be used)
*    <LI>filter center wavelength
*    <LI>flags for whether or not to dymanically plot the data as it's
being modelled.
*</UL><BR>
* The file <b>BraggModelSample.in</b> is a sample setup file that can
be modified
* and used.
* <p> Once the configuration information is read in, the application
will ask the user for the name
* of the Stack input file.  This is a text file which contains Layer
Definitions for the different types
* of epilayers and a definition of the overall filter stack that is to
be modelled.
* <p> This file <b>BraggStackExample.in</b> is a sample Stack input
file that can be modified and used
* @author Rajesh R. Patel
* @verion 1.02
* 7/22/99 - 8/6/99
*/

public class BraggModeller extends Model{

//String fileName = null;

/* Wavelength information follows   NOTE:  the symbol [=] signifys
units */

/**
* the starting wavelength for this model run (in nanometers)
*/
public double lambdaStart = 500;        //starting wavelengthm, [=] nm

/**
```

```
* the ending wavelength for this model run  (in nanometers)
*/
public double lambdaEnd = 2000;        //ending wavelength, [=] nm

/**
* the wavelength increment for this model run  (in nanometers)
*/
public double deltaLambda =1;        //delta lambda, [=] nm

/**
* the center wavelength of the filter for this model run (in
nanometers)
*/
public double lambdaCenter = 1000;      //central wavelength [=] nm
private final double c = 2.99792E+17;   //speed of light in vacuum -
ref from brandsen & joachhain

/* Angle information for this model run follows*/

/**
* the starting angle for this model run  (in degrees)
*/
public double thetaStart = 0;  //Starting angle [=] deg, measured from
optic axis

/**
* the ending angle for this model run  (in degrees)
*/
public double thetaEnd = 0;   //ending angle [=] deg, measured freom
the optic axis

/**
* the angle increment for this model run  (in degrees)
*/
public double deltaTheta = 0;  //change in angle [=] deg

/**
* the filter design angle for this model run (in degrees);
* This is the angle at which the filter is designed to be used - layer
thicknesses
* will automatically be adjusted according to this angle.
*/
public double designAngle = 0.0;
//#######################################################

/**
* the minimum value for the y-axis if dynamic plotting is being used
(in dB)
*/
public double ymin = -33;  //minimum y axis coordinate

/**
* the maximum value for the y-axis if dynamic plotting is being used
(in dB)
*/
public double ymax = 3;   //maximum y axis coordinate
```

```
/**
* flag for plotting the transmission curve for TE polarization - the
curve is plotted if this flag is true and dynamicPlotting is true
*/
public boolean plotTE = true;

/**
* flag for plotting the transmission curve for TM poloarization - the
curve is plotted if this flag is true and dynamicPlotting is true
*/
public boolean plotTM = false;

/**
* flag for plotting the reflection curve for TE polarization - the
curve is plotted if this flag is true and dynamicPlotting is true
*/
public boolean plotRTE = false;

/**
* flag for plotting the reflection curve for TM polarization - the
curve is plotted if this flag is true and dynamicPlotting is true
*/
public boolean plotRTM = false;

/**
* flag to indicate whether or not data is dynamically plotted - data is
plotted if this flag is true
* <BR><b>note:</b> if this flag is false, the flags plotTE, plotTM,
plotRTE, and plotRTM are overridden and data <b>will not </b>be plotted
* as it is being modelled.
*/
public boolean dynamicPlotting = true;

/**
* flag to indicate if the data for each angle in the range thetaStart
to thetaEnd should be parsed separately
* if true, a separate file with a single column of data for each angle
will be created - this is primarily for
* calculations of the effect of fiber NA on overall filter performance.
*/
public boolean parseAngles = false;

/**
* flag for whether or not to save the data from this model run;
* If true, the data is saved, if false, the data is not saved
* This flag can be used in conjunction with the flag dynamicPlotting to
make several quick model runs without writing to disk.
*/
public boolean saveData = true;
//public boolean trackPhases = false;   These variables are obsolete
//public boolean chirpError = false;
//public double chirpErrorAmount = 0.0;


//#######################################################
```

```
public static void main(String arg[]){

BraggModeller b = new BraggModeller();
Model2.assignModelParameters(b);
/************************/
long tstart = 0; //times the model run - not critical to the
calculation
try{
StackManager manager = new StackManager(b.lambdaCenter);
 if(b.designAngle>0){
 manager.tiltFilter((b.designAngle*Math.PI/180));
 }
StackCalculator calc = new StackCalculator(manager);

double RC_SWave,RC_PWave,TC_SWave,TC_PWave,relLambda;
String d=",    ";
String out="Wavelength"+d+" frequency"+d+" angle"+d+" R for S_Wave"+d+"
T for S_wave"+d+" R for P_Wave"+d+" T for P_Wave "+"\n";
String n="\n";
out = out+n;
PrintWriter ink = null;
if (b.saveData){
 ink = b.getOutputFileName("Enter the name of the Output File");
 ink.println(out);
 }
PrintWriter blot = null;
PrintWriter splot = null;
String out2 = "Wavelength"+d;
String out4 = "Phasetrack"+d+" LayerNumber"+d;
if (b.parseAngles){
 blot = b.getOutputFileName("Enter the angle output file name");
 blot.print(out2);
}

double th = b.thetaStart;
out2=th+d;
 int angcount = (int)((b.thetaStart-b.thetaEnd)/b.deltaTheta);
for (int zqw=0;zqw<angcount;zqw++){
 th = th-b.deltaTheta;
 out2=out2+th+d;
}
 out2 = out2+out2;
if (b.parseAngles){
blot.println(out2);
}
String simName = "";
 if (b.saveData){
   simName = Keyboard.readln("Enter the name for this simulation");
 }
double count = ((b.lambdaEnd-b.lambdaStart)/b.deltaLambda);
int cnt = (int)count;
double lambda = b.lambdaStart;
tstart = System.currentTimeMillis();
System.out.println("Working.....");
Plotter plotter = null;
Plotter rplotter = null;
boolean connect = false;
```

```
if (b.dynamicPlotting){
plotter = new Plotter("Transmission");
connect = false;

if (b.plotRTE|b.plotRTM){
rplotter = new Plotter("Reflection");
rplotter.setXRange(b.lambdaStart,b.lambdaEnd);
rplotter.setYRange(b.ymin,b.ymax);
}
//System.out.println(b.plotTE);
//System.out.println(b.plotTM);
//System.out.println(b.plotRTE);
//System.out.println(b.plotRTM);
plotter.setXRange(b.lambdaStart,b.lambdaEnd);
plotter.setYRange(b.ymin,b.ymax);
}//if dynamic plotting
double dB, TMdB, rdB, rTMdB;
double logten = Math.log(10);
String out3 = "";
for (int x = 0; x<=cnt;x++){ //wavelength loop
  out2 = lambda+d;
  double thet = b.thetaStart;
  manager.assignWavelengthDependentInfo(lambda);
  double freq = b.c/lambda;

  for (int y = 0; y<=angcount;y++){
      out="";
      double thetrad = (thet/180)*Math.PI;
      manager.assignAngleDependentInfo(thetrad);
      calc.calculate();
      RC_SWave = calc.reflectionCoefficient("S");
      TC_SWave = calc.transmissionCoefficient("S");
      RC_PWave = calc.reflectionCoefficient("P");
      TC_PWave = calc.transmissionCoefficient("P");
   //   relLambda = b.lambdaCenter/lambda;
      dB = (10*(Math.log(TC_SWave)/logten));
      TMdB = (10*(Math.log(TC_PWave)/logten));
      rdB = (10*(Math.log(RC_SWave)/logten));
      rTMdB = (10*(Math.log(RC_PWave)/logten));
      if (b.saveData){
         out =
lambda+d+freq+d+thet+d+RC_SWave+d+TC_SWave+d+RC_PWave+d+TC_PWave+d+dB+d
;
         ink.println(out);
         }
      out2 = out2+dB+d;
      out3=out3+TMdB+d;

     if (b.dynamicPlotting){
        if (lambda==b.lambdaStart){
         connect = false;
         String title = null;
         if(b.plotTE){
         title = "TE at "+thet+" degrees";
         plotter.addLegend(y,title);
         if (b.plotRTE){
          rplotter.addLegend(y,title);
```

```
          }
         }
         if(b.plotTM){
         title = "TM at "+thet+"degrees";
         plotter.addLegend(y+angcount+1,title);
           if (b.plotRTM){
             rplotter.addLegend(y,title);
             }
         }
        }
        else{
         connect = true;
         }
        if (b.plotTE){
         plotter.addDataPoint(y,lambda,dB,connect);
        }
        if (b.plotTM){
         plotter.addDataPoint(y+angcount+1,lambda,TMdB,connect);
        }
        if (b.plotRTE){
          rplotter.addDataPoint(y,lambda,rdB,connect);
          }
        if (b.plotRTM){
         rplotter.addDataPoint(y+angcount+1,lambda,rTMdB,connect);
         }
      }//dynamic plotting
      thet=thet-b.deltaTheta;
  } //theta loop
lambda=lambda+b.deltaLambda;
out2 = out2+out3;
if (b.parseAngles){
blot.println(out2);}
out2="";
out3="";
}//lambda loop
if (b.saveData){
  ink.println();
  ink.println("Simulation Name: "+d+"  "+d+"  "+d+simName+d);
  ink.println();
  calc.writeStackInfo(ink);
  ink.close();
}
if (b.parseAngles){blot.close();}

}//try
catch(Exception e){
System.err.println("an error occured!");
System.err.println(e.toString());
e.printStackTrace();
System.exit(1);
}
System.out.println("Done in "+(System.currentTimeMillis()-
tstart)/1000+" seconds!!!");
}//main

}//BraggModeller
```

```
/** StackManager.java  - parses input files for BraggModel
*       @Author  Rajesh Patel
*       @Date 7/20/99
*       @Version 2.0.1
*/

import java.io.*;
import java.util.*;
import java_code.modelling.*;
import java_code.utilities.*;

public class StackManager{

public static final int BLOCK_START = (int)'[';
public static final int BLOCK_END = (int)']';
public static final int STACK_START = (int) '{';
public static final int STACK_END = (int)'}';
public double cwl;

//public Vector keys;
public StreamTokenizer s;

public StackManager(double designWl){
 cwl=designWl;
 System.out.println();
 s = new StreamTokenizer(Model2.getInputFileName("Enter the Name of the
Stack Input File"));
 readLayerDefinitions(s);
}

private void readLayerDefinitions(StreamTokenizer L){

  int pipeline = (int)'|';
  int eq = (int)'=';
  int comma = (int)',';

  L.parseNumbers();
  L.eolIsSignificant(true); //parse eol as a token not whitespace
  L.lowerCaseMode(true); //automatically de-capitalize all strings
  L.whitespaceChars(pipeline,pipeline); //pipeline character is
whitespace
  L.whitespaceChars(eq,eq);
  L.whitespaceChars(comma,comma);

  String key;
  String layerType;
  double layerIndexInfo;
  double layerThicknessInfo;
  double layerWl;

  Stackable tempLayer;

  try{

   L.nextToken();
   int test = 0;
  while(L.ttype!=STACK_START){
```

```
    layerWl = this.cwl;
    key = L.sval;
  //  System.out.println("Key = "+key);
    L.nextToken();
    layerType=L.sval;
  // System.out.println("Layertype = "+layerType);
    L.nextToken();
    layerIndexInfo=L.nval;
  //  System.out.println("Layer index info = "+layerIndexInfo);
      tempLayer = LayerVendor.vend(layerType,layerIndexInfo);
    L.nextToken(); //skip the t= symbol - not needed
  //  System.out.println(L);
    L.nextToken(); //this is layer thickness info
  //  System.out.println(L +"This should be layer thickness info");
     layerThicknessInfo=L.nval;
       L.nextToken();
  //     System.out.println(L+" This should be the EOL token unless
layer wl is diff. or t in nm");

       if (L.ttype==StreamTokenizer.TT_EOL){
            tempLayer.setThickness(layerThicknessInfo,layerWl);
//          System.out.println("just set the thickness");
         }
      else{
         if ((L.sval).equals("nm")){
                  tempLayer.setThickness(layerThicknessInfo);
                }
         else{//layer wavelength is different than cwl
                L.nextToken();//read in desired layer wavelength
                layerWl = L.nval;
                tempLayer.setThickness(layerThicknessInfo,layerWl);
              }
           L.nextToken(); //read the EOL token now
        }//else
     HashKey.addKey(key,tempLayer);
     L.nextToken();
    }//while
  //   System.out.println("read in all layer defs - pushing back stack
start token");
     L.pushBack();//put the stack start token back in the stream;
   }//try

   catch(IOException e){
    e.printStackTrace();
    System.exit(-1);
   }//catch
    L.eolIsSignificant(false);
  }//readLayerDefinitions


public EpiLayer[] getStack(){
  return makeStack(this.s);
}


private EpiLayer[] makeStack(StreamTokenizer L){
```

```
/*Here we read in the stack info and parse it to make the stack which
the
*StackCalculator will then use to do calculations.
*NOTE: Since we will only be making references to the already defined
layers
*the array of EpiLayers will not take up as much memory as the stack
did in the
*previous version of the program (eg GaBraggStack.java).
*/

  int carat = (int)'^';

  L.whitespaceChars(carat,carat);

  Vector stack = readStackDefs(L);
   stack.trimToSize();

  EpiLayer[] returnable = new EpiLayer[stack.size()];  //make the array
for the stack
    int i=0;
    for(Enumeration e = stack.elements();e.hasMoreElements();){
         returnable[i] =
(EpiLayer)(HashKey.getBoxContents((String)e.nextElement()));
         i++;
     }
   return returnable;
  }



private static Vector readStackDefs(StreamTokenizer L){
  Vector temp = new Vector();
  Vector block = new Vector();
 try{

   while(L.ttype!=StreamTokenizer.TT_EOF){

     L.nextToken();

     switch(L.ttype){

     case BLOCK_START:
       block = readStackDefs(L);
       for (Enumeration e = block.elements();e.hasMoreElements();){
        temp.addElement(e.nextElement());
       }
      break;

     case BLOCK_END:
          L.nextToken(); //remember the '^' char is already whitespace
          if (L.ttype==StreamTokenizer.TT_NUMBER){
            //the block is to be repeated L.nval number of times
             if (L.nval<1){
               throw new
IllegalArgumentException(String.valueOf(L.nval));
              }
            else{
```

```
              Vector dummy = (Vector) temp.clone();
              for(int i=1;i<L.nval;i++){
                for(Enumeration e =
dummy.elements();e.hasMoreElements();){
                 temp.addElement(e.nextElement());
                }//for
               }//for
              }//the above block does the necessary repitition of the
block of layers
           }//if TT_NUMBER
           else{
           L.pushBack();
           }           //the block is returned - control goes back to
the blockStart of the
         return temp; //higher level.

     case STACK_END:
       return temp;

     case STACK_START:
       L.nextToken();
       if (!(L.sval.equals("incident"))){
         throw new IllegalArgumentException("The stack MUST start with
a layer of type 'incident' ");
       }
       else L.pushBack();
       break;

     default:
       temp.addElement(L.sval);
     }//switch
    }//while
   //if we get here without a return - there was a stack end error
   throw new IOException("EOF reached before Stack End flag");
 }//try

 catch(Exception iae){
  System.err.println(iae.getMessage());
  iae.printStackTrace();
  System.exit(-1);
 }//catch

    return null;   //this should never be triggered actually
 }//readstackdefs


public void assignWavelengthDependentInfo(double wl){
  for(Enumeration e=HashKey.getInventory();e.hasMoreElements();){
    ((EpiLayer)e.nextElement()).assignWavelengthDependentInfo(wl);
  }
}

public void assignAngleDependentInfo(double ang){
   EpiLayer first = (EpiLayer)(HashKey.getBoxContents(1)); //get the
incident layer;
   EpiLayer current;
   for(Enumeration e=HashKey.getInventory();e.hasMoreElements();){
```

```
      current = (EpiLayer)e.nextElement();
      current.setAngle(snell(first.getIndex(),current.getIndex(),ang));
      current.setLayerMatrices();
       }
}

public void setLayerMatrices(){
 EpiLayer current;
 for(Enumeration e=HashKey.getInventory();e.hasMoreElements();){
  current = (EpiLayer)e.nextElement();
  current.setLayerMatrices();
  }
}
private double snell(double n1, double n2, double th1){
   return Math.asin(n1/n2*Math.sin(th1));
}

public void tiltFilter(double ang){
    this.assignWavelengthDependentInfo(this.cwl);
    this.assignAngleDependentInfo(ang);
    EpiLayer layer;
    for(Enumeration e=HashKey.getInventory();e.hasMoreElements();){
      layer = (EpiLayer)e.nextElement();
      layer.setThickness(layer.getThickness()/Math.cos(layer.angle));
    }
}


}//class
```

```
/* StackCalculator.java by Rajesh Patel 9/2/97
*this class is used by BraggModeller or its subclasses to
*create the actual BraggStack for the model run
*/

import java.lang.Math;
import java.lang.reflect.Array;
import java.io.*;
import java.util.*;
import java_code.utilities.*;
import java_code.mathematics.*;

public class StackCalculator{

private ComplexNumber p12 = new ComplexNumber(0,1);
private ComplexNumber p21 = new ComplexNumber(0,1);
private ComplexNumber s12 = new ComplexNumber(0,1);
private ComplexNumber s21 = new ComplexNumber(0,1);
private EpiLayer[] TheStack;
int num;
int[] phasor;
Vector temp;
boolean trackPhases;
double Angle_0,CA,ndx,mu_s,mu_p,Clmicrons;
Matrix Whole_SWave,Whole_PWave;//,CurrentLayer_S,CurrentLayer_P;
```

```
boolean doped;
double highIndexError,lowIndexError;
String d = "    , ";
String n = "\n";

StackCalculator(StackManager boss) throws IOException{ //constructor
doped = true;
this.TheStack = boss.getStack();
this.num=TheStack.length-1; //adjust for java arrays begining with 0
instead of 1
this.printStackInfo();
Whole_SWave = new Matrix(2,2);
Whole_PWave = new Matrix(2,2);
//CurrentLayer_S = new Matrix(2,2);
//CurrentLayer_P = new Matrix(2,2);
//this.calculate();
}

public void printStackInfo(){

 double slabThickness = 0.0;

 for (int y=0;y<TheStack.length;y++){
  System.out.println("Layer number "+(y+1)+" :
"+TheStack[y].getInfo());
  if((y>0)&&(y<TheStack.length-1)){
  slabThickness+=TheStack[y].thickness;
   }
 }
 System.out.println();
 System.out.println("The Total Thickness of the Filter is:
"+(slabThickness/1000)+" microns");
 if ((slabThickness/1000)>10){
   try{
   if(!(Keyboard.readYN("The filter is greater than 10 microns thick -
this will be difficult to grow - continue? "))){
     System.exit(0);
   }//if
  }//try
  catch(Exception e){
   System.err.println("The following exception occured: "+e);
   System.exit(1);
   }//catch
  }//if thickness too big
}

public void chirpError(double amount){
  int num2 = (int)(num/2);
  double num = (double) num2;
  double temp;
  for (int w = 0;w<=num2;w++){

    temp = amount*((double)(w/num));
    System.out.println(temp);
    TheStack[w].thickness*=(1-temp);
    System.out.println("Changed layer "+w+" by "+100*(1-temp)+"
percent");
```

```
 }//loop
}

public void writeStackInfo(PrintWriter spooler){
 String d ="    , ";
 double slabThickness = 0.0;

 for (int y=0;y<TheStack.length;y++){
  spooler.println("Layer number "+(y+1)+" : "+TheStack[y].getInfo()+d);
  slabThickness+=TheStack[y].getThickness();
 }
 spooler.println();
 spooler.println("The Total Thickness of the Filter is:
"+(slabThickness/1000)+" microns");

}

/*
public void setCurrentLayerMatrix(int j){

 CA = Math.cos(TheStack[j].angle);
 mu_s = TheStack[j].getIndex()*CA;
 mu_p = TheStack[j].getIndex()/CA;

 this.p12.setImag(TheStack[j].phimag/mu_p);
 this.p21.setImag(TheStack[j].phimag*mu_p);
 this.s12.setImag(TheStack[j].phimag/mu_s);
 this.s21.setImag(TheStack[j].phimag*mu_s);

 this.CurrentLayer_S.assign(1,1,TheStack[j].phreal);
 this.CurrentLayer_S.assign(2,2,TheStack[j].phreal);
 this.CurrentLayer_S.assign(1,2,this.s12);
 this.CurrentLayer_S.assign(2,1,this.s21);
 this.CurrentLayer_P.assign(1,1,TheStack[j].phreal);
 this.CurrentLayer_P.assign(2,2,TheStack[j].phreal);
 this.CurrentLayer_P.assign(1,2,this.p12);
 this.CurrentLayer_P.assign(2,1,this.p21);

}//setCurrentLayerMatrix()
*/

public void calculate(){
//IncidentLayer.calculate(Lambda,InitialAngle);     //deprecated as of
V2.0.1
Whole_SWave.flush();
Whole_PWave.flush();
Whole_SWave =
Matrix.matrixSum(Whole_SWave,((BoundaryLayer)TheStack[0]).DInv_SWave);
Whole_PWave =
Matrix.matrixSum(Whole_PWave,((BoundaryLayer)TheStack[0]).DInv_PWave);

 for (int i = 1; i<this.num; i++){
   //setCurrentLayerMatrix(i);
//   Whole_SWave = Matrix.matrixProd(Whole_SWave,CurrentLayer_S);
//   Whole_PWave = Matrix.matrixProd(Whole_PWave,CurrentLayer_P);
   Whole_SWave = Matrix.matrixProd(Whole_SWave,TheStack[i].SWave);
   Whole_PWave = Matrix.matrixProd(Whole_PWave,TheStack[i].PWave);
```

```
 }//loop

Whole_SWave =
Matrix.matrixProd(Whole_SWave,((BoundaryLayer)TheStack[num]).D_SWave);
Whole_PWave =
Matrix.matrixProd(Whole_PWave,((BoundaryLayer)TheStack[num]).D_PWave);

}//calculate

public double reflectionCoefficient(String pol){
Matrix Whole;
pol.toUpperCase();

 if (pol.equals("S")){
    Whole=Whole_SWave;
 }
 else{
   Whole = Whole_PWave;
 }

double R =
ComplexNumber.complexMagSQ(ComplexNumber.complexDiv(Whole.element(2,1),
Whole.element(1,1)));

return R;

}// reflection Coefficient


public double transmissionCoefficient(String pol){
  pol.toUpperCase();

  if(pol.equals("S")){
    return StransmissionCoefficient(Whole_SWave);
  }
  else{
    return PtransmissionCoefficient(Whole_PWave);
  }
}

private double StransmissionCoefficient(Matrix Whole){

//ComplexNumber I = new ComplexNumber(1,0);
double T =
((TheStack[num].index*Math.cos(TheStack[num].angle))/(TheStack[0].index
*Math.cos(TheStack[0].angle)))*(1/ComplexNumber.complexMagSQ(Whole.elem
ent(1,1)));

//this is for lossless media

return T;
}// Stransmission coefficient

private double PtransmissionCoefficient(Matrix Whole){

//ComplexNumber I = new ComplexNumber(1,0);
```

```
double T =
((TheStack[num].index*Math.cos(TheStack[0].angle))/(TheStack[0].index*M
ath.cos(TheStack[num].angle)))*(1/ComplexNumber.complexMagSQ(Whole.elem
ent(1,1)));

//this is for lossless media

return T;
}// Ptransmission coefficient


}//class
```

```
/**BraggLayer.java  8/27/97  Rajesh Patel
  Modified 6_99  Version 2.01

This class represents a thin film layer with constant (non wavelength
dependent) index of refraction */

class BraggLayer extends EpiLayer{

//public static final double CONSTANT_INDEX_LAYER_FLAG = -1.0;
//public static final int LAYER_TYPE_FLAG = -2;
//public double index;
/**The following 2 lines were added on 5/30/98 they might need to be
removed*/
//public final String LayerType = "BraggLayer";
//public double mol;

//double k; //wavevector - this quantity depends on Lambda
//double wavelengthInMedium;
//double phase;
//double phreal, phimag;

//public double thickness; //this will usually be Lambda/(4*index)

public BraggLayer(double index, double t){ //constructor

this.index=index;
this.thickness = t;
}

public BraggLayer(){
 this.index=1.0;
 this.thickness=1.0;
 }

public BraggLayer(double index){
 this.index=index;
 }


public void setIndex(double wave){
```

```
} //for constant index materials - the index doesn't change

public double getAlContent(){
 return BraggLayer.CONSTANT_INDEX_LAYER_FLAG;
 }

public int getLayerType(){
 return BraggLayer.LAYER_TYPE_FLAG;

}

public void setThickness(double thickness){ //thickness in nm
 this.thickness=thickness;
}

public void setThickness(double thickness, double wl){//thickness in
waves
 this.setThickness(wl*thickness/this.index);
}
}//BraggLayer
```

```
/**BoundaryLayer.java  9/10/97  Rajesh Patel
*  modified 7/22/99 - made to inheret from EpiLayer to make version
2.0.1 Compliant
*
*/
import java_code.mathematics.*;

class BoundaryLayer extends EpiLayer{

Matrix D_SWave;        // the matrix for the interface between media 1
& 2
Matrix DInv_SWave;  //the inverse of D

Matrix D_PWave;
Matrix DInv_PWave;

BoundaryLayer(double index){ //constructor
super(index,1);
D_SWave = new Matrix(2,2);
DInv_SWave = new Matrix(2,2);
DInv_PWave = new Matrix(2,2);
D_PWave = new Matrix(2,2);
//this.index=index;
//this.thickness=1;
}

private void calculate(){

double CA = Math.cos(this.angle);
double mu_s = this.index*CA;
double mu_p = this.index/CA;

this.D_SWave.assign(1,1, 1.0);
this.D_SWave.assign(1,2, 1.0);
```

```
this.D_SWave.assign(2,1, (mu_s));
this.D_SWave.assign(2,2, (-1*mu_s));

this.D_PWave.assign(1,1, 1.0);
this.D_PWave.assign(1,2, 1.0);
this.D_PWave.assign(2,1, mu_p);
this.D_PWave.assign(2,2, (-1*mu_p));

this.DInv_SWave.assign(1,1, 0.5);
this.DInv_SWave.assign(2,1, 0.5);
this.DInv_SWave.assign(1,2, (0.5/(mu_s)));
this.DInv_SWave.assign(2,2, (-0.5/(mu_s)));
this.DInv_PWave.assign(1,1, 0.5);
this.DInv_PWave.assign(2,1, 0.5);
this.DInv_PWave.assign(1,2, (0.5/mu_p));
this.DInv_PWave.assign(2,2, (-0.5/mu_p));

}//calculate()

public void setAngle(double ang){
 super.setAngle(ang);
 this.calculate();
}

public String getInfo(){
 String  returnable = "boundary Layer - index :"+this.index;
 return returnable;
 }
}//BoundaryLayer
```

```
/**EpiLayer.java  8/27/97  Rajesh Patel
  Modified 6_99  Version 2.01

This class represents thin film layers - it uses the classes Matrix and
ComplexNumber which were written by Rajesh Patel */

import java_code.mathematics.*;

class EpiLayer implements java.io.Serializable, Stackable{

public static final double CONSTANT_INDEX_LAYER_FLAG = -1.0;
public static final int LAYER_TYPE_FLAG = -2;
public double index;
/**The following 2 lines were added on 5/30/98 they might need to be
removed*/
public final String LayerType = "EpiLayer";

double k; //wavevector - this quantity depends on Lambda
double wavelengthInMedium;
double phase;
double phreal, phimag;
double angle = 0;
public Matrix SWave = new Matrix(2,2);
public Matrix PWave = new Matrix(2,2);

private ComplexNumber p12 = new ComplexNumber(0,1);
```

```
private ComplexNumber p21 = new ComplexNumber(0,1);
private ComplexNumber s12 = new ComplexNumber(0,1);
private ComplexNumber s21 = new ComplexNumber(0,1);
private double CA,mu_s,mu_p;

public double thickness; //this will usually be Lambda/(4*index)

public EpiLayer(double index, double t){ //constructor

this.index=index;
this.thickness = t;
}

public EpiLayer(){
 this.index=1.0;
 this.thickness=1.0;
 }

public EpiLayer(double index){
 this.index=index;
 }


public void setIndex(double wave){

} //for constant index materials - the index doesn't change

public double getAlContent(){
 return EpiLayer.CONSTANT_INDEX_LAYER_FLAG;
 }

public int getLayerType(){
 return EpiLayer.LAYER_TYPE_FLAG;

}

public void setThickness(double thickness){ //thickness in nm
 this.thickness=thickness;
}

public void setThickness(double thickness, double wl){//thickness in
waves
 this.setThickness(wl*thickness/this.index);
}

public void assignWavelengthDependentInfo(double wl){
 this.wavelengthInMedium = wl/this.index;
 this.k = (2*Math.PI)/this.wavelengthInMedium;
}

public void setAngle(double ang){
 this.angle=ang;
 this.phase=this.k*this.thickness*Math.cos(this.angle);
 this.phreal=Math.cos(this.phase);
 this.phimag=Math.sin(this.phase);
}
```

```
public double getIndex(){
 return this.index;
}

public double getThickness(){
 return this.thickness;
}

public String getInfo(){
  String returnable = "  index, "+this.index+" , thickness
,"+this.thickness;
  return returnable;
}

public double getAngle(){
 return this.angle;
}

public void setLayerMatrices(){
 CA = Math.cos(this.angle);
 mu_s = this.index*CA;
 mu_p = this.index/CA;

 this.p12.setImag(this.phimag/mu_p);
 this.p21.setImag(this.phimag*mu_p);
 this.s12.setImag(this.phimag/mu_s);
 this.s21.setImag(this.phimag*mu_s);

 this.SWave.assign(1,1,this.phreal);
 this.SWave.assign(2,2,this.phreal);
 this.SWave.assign(1,2,this.s12);
 this.SWave.assign(2,1,this.s21);
 this.PWave.assign(1,1,this.phreal);
 this.PWave.assign(2,2,this.phreal);
 this.PWave.assign(1,2,this.p12);
 this.PWave.assign(2,1,this.p21);
}

}//EpiLayer
```

```
/**GaBraggLayer.java  5/20/98  Rajesh Patel
Modified 6_99  Version 2.01

This class represents thins film layers of AlxGa1-xAs */

class GaBraggLayer extends EpiLayer{

public double mol; //aluminum mole fraction of the layer
//public static final int LAYER_TYPE_FLAG = 1;
//public double thickness; //this will usually be Lambda/(4*index)
//public final String LayerType = "GaBraggLayer";

GaBraggLayer(double al, double t){ //constructor

this.mol=al;
```

```
this.thickness=t;
}

public GaBraggLayer(double al){
 this.mol=al;
}


public void setIndex(double lambda){
 //convert the wavelength to microns as required by the IndexCalc class
 double l = lambda/1000;
 this.index = IndexCalc.index(mol,l,false);
}

public double getAlContent(){
 return this.mol;
 }

public int getLayerType(){
 return GaBraggLayer.LAYER_TYPE_FLAG;
}

public void setThickness(double thickness,  double wl){
 this.setIndex(wl);
 this.setThickness(thickness*wl/this.index);
}

public void assignWavelengthDependentInfo(double wl){
 this.setIndex(wl);
 super.assignWavelengthDependentInfo(wl);
}

public String getInfo(){
  String returnable = "  aluminum fraction, "+this.mol+" , thickness
,"+this.thickness;
  return returnable;
}


}//GaBraggLayer
/**BoundaryLayer.java  9/10/97  Rajesh Patel
*  modified 7/22/99 - made to inheret from EpiLayer to make version
2.0.1 Compliant
*
*/
import java_code.mathematics.*;

-------------------------------------------------------------------------------------------

class IncidentLayer extends BoundaryLayer{

Matrix D_SWave;        // the matrix for the interface between media 1
& 2
Matrix DInv_SWave;  //the inverse of D

Matrix D_PWave;
```

```
Matrix DInv_PWave;
public String layerDescriptor = "incident";
IncidentLayer(double index){ //constructor
super(index,1);
D_SWave = new Matrix(2,2);
DInv_SWave = new Matrix(2,2);
DInv_PWave = new Matrix(2,2);
D_PWave = new Matrix(2,2);
//this.index=index;
//this.thickness=1;
}

private void calculate(){

double CA = Math.cos(this.angle);
double mu_s = this.index*CA;
double mu_p = this.index/CA;

this.D_SWave.assign(1,1, 1.0);
this.D_SWave.assign(1,2, 1.0);
this.D_SWave.assign(2,1, (mu_s));
this.D_SWave.assign(2,2, (-1*mu_s));

this.D_PWave.assign(1,1, 1.0);
this.D_PWave.assign(1,2, 1.0);
this.D_PWave.assign(2,1, mu_p);
this.D_PWave.assign(2,2, (-1*mu_p));

this.DInv_SWave.assign(1,1, 0.5);
this.DInv_SWave.assign(2,1, 0.5);
this.DInv_SWave.assign(1,2, (0.5/(mu_s)));
this.DInv_SWave.assign(2,2, (-0.5/(mu_s)));
this.DInv_PWave.assign(1,1, 0.5);
this.DInv_PWave.assign(2,1, 0.5);
this.DInv_PWave.assign(1,2, (0.5/mu_p));
this.DInv_PWave.assign(2,2, (-0.5/mu_p));

}//calculate()

public void setAngle(double ang){
 super.setAngle(ang);
 this.calculate();
}

public String getInfo(){
 String  returnable = "boundary Layer - index :"+this.index;
 return returnable;
 }

}//IncidentLayer
```

```
/**InGaPBraggLayer.java  4/18/2000  Rajesh Patel
Version 2.01

*/
//represents thin film layers of InGaP with variable Aluminum fraction
class InGaPBraggLayer extends EpiLayer{

public double mol; //aluminum mole fraction of the layer

InGaPBraggLayer(double al, double t){ //constructor

this.mol=al;
this.thickness=t;
}

public InGaPBraggLayer(double al){
 this.mol=al;
}


public void setIndex(double lambda){
 this.index = InGaP_Index.index(mol,lambda);
}

public double getAlContent(){
 return this.mol;
 }
/*
*    public int getLayerType(){
*     return InGaPBraggLayer.LAYER_TYPE_FLAG;
*    }
*/
public void setThickness(double thickness,  double wl){
 this.setIndex(wl);
 this.setThickness(thickness*wl/this.index);
}

public void assignWavelengthDependentInfo(double wl){
 this.setIndex(wl);
 super.assignWavelengthDependentInfo(wl);
}

public String getInfo(){
  String returnable = "  InGaAlP - aluminum fraction, "+this.mol+" ,
thickness ,"+this.thickness;
  return returnable;
}


}//InGaPBraggLayer
```

```
/** 9/10/97  Rajesh Patel
*  modified 7/22/99 - made to inheret from EpiLayer to make version
2.0.1 Compliant
*
*/
import java_code.mathematics.*;

class SubstrateLayer extends BoundaryLayer{

Matrix D_SWave;        // the matrix for the interface between media 1
& 2
Matrix DInv_SWave;  //the inverse of D

Matrix D_PWave;
Matrix DInv_PWave;
public String layerDescriptor = "substrate";
SubstrateLayer(double index){ //constructor
super(index,1);
D_SWave = new Matrix(2,2);
DInv_SWave = new Matrix(2,2);
DInv_PWave = new Matrix(2,2);
D_PWave = new Matrix(2,2);
//this.index=index;
//this.thickness=1;
}

private void calculate(){

double CA = Math.cos(this.angle);
double mu_s = this.index*CA;
double mu_p = this.index/CA;

this.D_SWave.assign(1,1, 1.0);
this.D_SWave.assign(1,2, 1.0);
this.D_SWave.assign(2,1, (mu_s));
this.D_SWave.assign(2,2, (-1*mu_s));

this.D_PWave.assign(1,1, 1.0);
this.D_PWave.assign(1,2, 1.0);
this.D_PWave.assign(2,1, mu_p);
this.D_PWave.assign(2,2, (-1*mu_p));

this.DInv_SWave.assign(1,1, 0.5);
this.DInv_SWave.assign(2,1, 0.5);
this.DInv_SWave.assign(1,2, (0.5/(mu_s)));
this.DInv_SWave.assign(2,2, (-0.5/(mu_s)));
this.DInv_PWave.assign(1,1, 0.5);
this.DInv_PWave.assign(2,1, 0.5);
this.DInv_PWave.assign(1,2, (0.5/mu_p));
this.DInv_PWave.assign(2,2, (-0.5/mu_p));

}//calculate()

public void setAngle(double ang){
 super.setAngle(ang);
 this.calculate();
}
```

```
public String getInfo(){
 String  returnable = "boundary Layer - index :"+this.index;
 return returnable;
 }

}//SubstrateLayer
```

```
//IndexCalc.java 9/2/97 Rajesh Patel
import java.lang.Math.*;
import java.io.*;

/*this class is called by all instances of GaBraggLayer to calculate
the index of refraction as a function of aluminum fraction and
wavelength */

class IndexCalc{

static double n;
//the main portion of this code was imported from a file by Bob Deri
// Deri's comments below:
/*
*
*      index.c
*        algas indices from my modeling efforts
*
*       contains two functions:
*         aindex for algas,  ngndex for gaas near/above gap
*
*
*/

static double cindex[] = new double[2]; /* pointer to imag index value
passed by cgndex */

//double aindex(double x, double ww);
//double cgndex(double x, double ww, double doped, double cindex[]);

//double fx(double x0);  /* adachi function for E0 edges */

public  static double index(double x, double wl, boolean doped){

int idoped;
double nk1;
double  n = 0;   //index
//boolean doped; /* yes/no-doped */

/*    print indices if input is OK  */
        if ((x==0.0) & (wl < 0.8856))
      {
                n=IndexCalc.cgndex(x,wl,doped,IndexCalc.cindex);
               // System.out.println("complex index= "+n+"
"+IndexCalc.cindex[1]+"i"+"\n");
        }
```

```
      else
      {
        n=IndexCalc.aindex(x,wl);
   //     System.out.println("refractive index = "+n+"\n");
       }

  return n;
}//index


/*
*********       FUNCTION  aindex(xx,ww) ***************************
*/
public static double aindex(double xx, double ww){
 double ehv, e1c; /* photon energy, E1 cutoff energy*/
 double e0,e1,e0d0;      /*  E0, E1, and E0 spin-orbit bandedges */
 double n2;       /* index squared */
 double a0, b0, c0, x1, x1c;        /* coeffs of adachi functions*/

 ehv=1.23986/ww;
/*
*       calculate bandgaps
*/
        if (xx <= 0.45)
         e0=1.424+1.45*xx-0.25*xx*xx;
         else
         {
         e0=1.4217 + 1.5439*xx;
           e0=e0-0.82554*xx*xx + 0.8779*xx*xx*xx;
         }
//System.out.println("Bandgap (eV)="+e0+" = "+1.23986/e0+"(um)\n");
e0d0=e0+0.34-0.04*xx;   /*  spin-orbit bandedge */
e1=2.9259+0.67171*xx-0.32424*xx*xx+0.61724*xx*xx*xx;
e1c=6.4;

 a0=5.6684 +1.4500*Math.exp(3.5584*xx)+10.464*xx;
 b0=9.1813-4.5059*xx-1.4304*xx*xx+2.2388*xx*xx*xx;
 c0=2.1582+0.80331*xx-0.091127*xx*xx-2.6906*xx*xx*xx;
 if (xx == 0.0){
          a0=5.3693;
          b0=10.512;
          c0=1.4725;
          }
n2=(Math.sqrt(e0)*IndexCalc.fx(ehv/e0)+0.5*Math.sqrt(e0d0)*IndexCalc.fx
(ehv/e0d0))/(ehv*ehv);
       /* E0 & spin.orb*/
x1=ehv/e1;
x1c=ehv/e1c;
n2=a0*n2 -b0*Math.log( (1.0-x1*x1)/(1.0-x1c*x1c) )/(x1*x1) +c0;

return Math.sqrt(n2);
}

/*
************ function fx is the Adachi E0 function for the index
********
*/
```

```
private static double fx(double x0){
return (2.0-Math.sqrt(1.0+x0)-Math.sqrt(1.0-x0));
}

/*
*******   function cgndex does complex index of GaAs ***********
*/
public static double cgndex(double xx, double ww, boolean mod, double
cindex[]){
double nn1, nk1;        /* complex index to return */
double alpha;           /*  absorption coeff   */
double de, delta;
double nm0 = 0;
double nm1 = 0;
double nm2 = 0;
double nm3 = 0;
double ehv = 0;

ehv=1.23986/ww;   /*  photon energy */
/*
*        nn1 is index of xal=0 GaAs at ww
*/
         if (ehv<1.421)
            nn1=aindex(xx,ww);
            else
            {
          de=ehv-1.40;
            nn1=3.6487-0.4371*de+4.9403*de*de;
            nn1=nn1-11.658*de*de*de +10.309*de*de*de*de;
            }
/*
*       index correction for n+ doping obtained from Casey etal
*        scaled so index unchanged at 1.4eV
*        for n+ substrate; n=2E18
*
*/
        if (mod) { //if GaAs is Degenerately doped
/*
*        nn1 is index of xal=0 GaAs at ww
*       curvefit to Aspnes data 1.3-1.7eV
*/
         nn1=-7.7793+21.247*ehv-13.329*ehv*ehv;
         nn1=nn1+2.8333*ehv*ehv*ehv;
          de=ehv-1.40;
          delta=-0.002257+de*1.2775-14.05*de*de;
          delta=delta+38.141*de*de*de;
          nn1=nn1+delta;
          }
/*
*      Imaginary part of index
*/
          if (!mod)
/*       Aspnes data for k of undoped GaAs
*/
         {   //GaAs is not degenerately doped
         nk1=0.0;
            if (ehv > 1.43)
```

```
          {
              nk1=-8.1423+12.051*ehv-3.554*ehv*ehv;
              nk1=nk1-1.6982*ehv*ehv*ehv+0.76394*ehv*ehv*ehv*ehv;
             }
           else
           {
             de=ehv-1.40;
             nk1=0.0003-0.22*de+70.0*de*de;
             }
           }
           else
           {
/*
*         Casey, Sell, and Wecht data for n=2E18 GaAs
*/
              if (ehv<1.36)
              {
              nm0=0.0;
              nm1=0.0;
              nm2=0.0;
                   }
                else if (ehv<1.395 && ehv>1.36 )
              {
                  nm0=77791.0;
                nm1=-1.1431e5;
                          nm2=42000.0;
                   }
                 else if (ehv<1.43 && ehv>1.395)
                {
                  nm0=2.3395e6;
                      nm1=-3.344e6;
                    nm2=1.195e6;
                   }
                 else if (ehv<1.53 && ehv>1.43)
              {nm0=1.2802e6;
              nm1=-1.8359e6;
              nm2=6.5838e5;
                   }
                 else if (ehv>1.53)
                {
/*      fit good to 1.60eV      */
                   nm0=-7.2643e4;
                   nm1=+5.5714e4;
                   nm2=0.0;
                   }
         alpha=nm0+nm1*ehv+nm2*ehv*ehv;
         nk1=1.0e-7*ww*alpha/(4.0*Math.acos(-1.0));
          }

IndexCalc.cindex[1]=nk1;
return nn1;
}

}
```

```
import java.lang.Math;

public class InGaP_Index{

//calculates the index of InGaAlP - called by all instances of
InGaPBraggLayer

public static double index(double x, double waveLength){
// wavelength is in nm
// x is the AlAs mole fraction (0-1)

double eV ;
double Eo;
double Ed;
double returnable;

eV = 1239.86/waveLength;

        Eo=3.39+0.62*x;
        Ed=28.07 + 1.27*x;
        returnable = Math.sqrt((1 + Eo * Ed / (Eo*Eo - eV*eV)));
        return returnable;
}

}
```

```
/*LayerVerndor -- this class was originally called StackVendor
*vends objects which implement the Stackable interface.
*by putting this into a separate class, different layer types can be
easily
*added in the future without affecting the Bragg modelling code.
*/

public class LayerVendor{
//remember: since the stream tokenizer has lowercasemode = true all
comparisons should be lc
public static Stackable vend(String layerType, double layerIndexInfo){
 Stackable returnable;
  if
(layerType.equals("boundary")|(layerType.equals("incident")|layerType.e
quals("substrate"))){
   System.out.println("Getting a boundary Layer");
   return getBoundaryLayer(layerIndexInfo);
   }
  if(layerType.equals("al")){
    System.out.println("Getting a GaBraggLayer");
    return getGaBraggLayer(layerIndexInfo);
    }
  if(layerType.equals("ingap")){
    System.out.println("Getting a InGaPBraggLayer");
    return getInGaPBraggLayer(layerIndexInfo);
    }
  else{
    System.out.println("Getting a BraggLayer");
```

```
    return getBraggLayer(layerIndexInfo);
    }
 }

private static BraggLayer getBraggLayer(double index){
 return new BraggLayer(index);
}

private static GaBraggLayer getGaBraggLayer(double al){
  return new GaBraggLayer(al);
}

private static InGaPBraggLayer getInGaPBraggLayer(double al){
  return new InGaPBraggLayer(al);
}

private static BoundaryLayer getBoundaryLayer(double index){
 return new BoundaryLayer(index);
}

}
```

```
/* the stackable interface is implemented by all thin film layers -
it provides a generic interface so that StackManager can handle all
types of thin film layers now and in the future without needing to know
the implementation details of the layers*/

public interface Stackable{

abstract public void setThickness(double thickness);

abstract public void setThickness(double thickness, double wavelength);

abstract public int getLayerType();
}
```

```
import java.util.*;
/**HashKey - implements a haskey with a numerical index as well as a
name
*       Rajesh Patel
*       7/19/99
*/

public class HashKey{

public static int COUNT=0;
private String name; //the string that the rest of the world uses to
call this key
private int keyNum;  //the global order number of this key
private Object box; // the contents of this key
private static Vector keys = new Vector();
```

```
protected HashKey(String ref, Object box){//nobody should directly call
this constructor
 Enumeration e = keys.elements();
try{
 while(e.hasMoreElements()){
    HashKey hk = (HashKey)e.nextElement();
        if (ref.equals(hk.getName())){
           throw new DuplicateNameException(ref);
        }
   }
 }
catch(DuplicateNameException dne){
 System.err.println("The HashKey name "+dne.getName()+" is already in
use please try again");
 dne.printStackTrace();
 System.exit(-1);
}
 this.name=ref;
 HashKey.incrementCounter();
 this.keyNum=COUNT;
 this.box=box;
 keys.addElement(this);
}

private static void incrementCounter(){
//increments the global count of existing keys
COUNT++;
}

public static int getKeyCount(){
 return COUNT;
}

public int getKeyNum(){
 return this.keyNum;
}

public String getName(){
 return this.name;
}

public static void addKey(String s, Object box){
  HashKey h = new HashKey(s,box);
}
public static void addKey(String s){
  addKey(s,null);
}

public static int getIndexOf(String s){

 Enumeration e = keys.elements();
  while(e.hasMoreElements()){
    HashKey hk = (HashKey)e.nextElement();
    if (s.equals(hk.getName())){
      return hk.getKeyNum();
      }
   }
```

```
 return -1;//this is an error code - it means the key was not found.
}

public static Object getBoxContents(String s){
 Enumeration e = keys.elements();
  while(e.hasMoreElements()){
    HashKey hk = (HashKey)e.nextElement();
    if (s.equals(hk.getName())){
      return ((hk.box==null) ? "empty" : hk.box );
      }
   }
 return null;//this is an error code - it means the key was not found.
}

public static Object getBoxContents(int boxNum){
try{
 if ((boxNum <0)|(boxNum>COUNT)){
  throw new IndexOutOfBoundsException("The index "+boxNum+" is out of
bounds");
  }
 Enumeration e = keys.elements();
  while(e.hasMoreElements()){
   HashKey hk = (HashKey)e.nextElement();
   if (hk.getKeyNum()==boxNum){
     return ((hk.box==null) ? "empty" : hk.box);
    }
 }

 }//try
catch(IndexOutOfBoundsException ioob){
 System.out.println(ioob.getMessage());
 ioob.printStackTrace();
 System.exit(-1);
 }
return null;
}

public static Enumeration elements(){
 return keys.elements();
}

public static Enumeration getInventory(){
 Vector temp = new Vector();

    for(Enumeration e=HashKey.elements();e.hasMoreElements();){
        temp.addElement(((HashKey)e.nextElement()).box);
    }

 return temp.elements();
}

public static void flush(){
  COUNT=0;
  keys = null;
  keys = new Vector();
 }
```

```
}//class


class DuplicateNameException extends Exception {

 String keyname;

public DuplicateNameException(String name){
    this.keyname=name;
}

public String getName(){
  return this.keyname;
}

}
```

```
package java_code.utilities;
import java.io.*;

/**
* This class is a utility class that facilitates
* use of the Keyboard as if it were a default device.
*
* For Example, if a variable of type <code>double</code> needed to be
* obtained from the user, the following code would be used:
* <p>
*<code> double d = (Double.valueOf(Keyboard.readln("give me a double
")));.doubleValue();</code>
* <p>
* The user would see the prompt "give me a double" and the program will
pause for a response.
* This class makes no type checks on requested data - the calling
method is responsible for
* required type checks.
*<p>
* This class saves that time of having to constantly initialize the
readers to access the
* standard system input device.
*
* @version      1.02 8/4/99
* @author       Rajesh Patel
* @since        Java_code 1.01
*
*/

public class Keyboard {


private Keyboard(){   //you can't instantiate this class

}

/**
* Displays a prompt and then returns a string of input from
```

```
* the keyboard when the enter key is pressed. This method
* conforms to the writeln, println grammar of the standard java
devices.
*
* @param prompt An input prompt to give instructions to the user.
*
* @return A <code>string</code> of input from the user.
*/

public static String readln(String prompt){
 return Keyboard.readLine(prompt);
}

/**
* Returns a string of input from the keyboard.
*
* @deprecated since Version 1.02 replaced by readln()
*  this method name does not follow the preferred java Grammar
* @see #readln(String)
*/

public static String readLine(String prompt){

/*
* making this method static defines it as a class method.
* by doing this - I can use the following syntax to get a line
* of input from the keyboard:  String input =
Keyboard.readline(prompt);
* where prompt is a string expression which is printed to the screen
before
* the line is read in.
* this method also closes the keyboard reader so that
* it can be garbage collected
*/

  KeyboardReader key = new KeyboardReader();
    String line = null;

    line = key.readLine(prompt);

    key = null;
   return line;
} //readline

/**
* Returns a boolean in response to a yes/no question.  Acceptable
* responses are "y", "n", "yes" or "no"  the response is not case
sensitive.
* This method will loop until an acceptable response has been entered.
* @param prompt The yes/no question
*
* @return a boolean - <code>true</code> if the response was yes and
*                     <code>false</code> if the response was no
*/
public static boolean readYN(String prompt) throws IOException{

   KeyboardReader key = new KeyboardReader();
```

```
   String line = key.readLine(prompt);
   line = line.trim();
   String errorprompt = "you must enter Y or N";

   boolean returnable = false; //default

   while(!checkResult(line)) {
     line = key.readLine(errorprompt);
     }
   line = line.toUpperCase();

   //System.out.println(line);
   String y = "Y";
   String yes = "YES";
   if ((line.equals(yes))|(line.equals(y))){
      returnable=true;
      key = null;
      return returnable;
      }
   key = null;
   return returnable;
}//readYN

private static boolean checkResult(String entry){
 entry = entry.toUpperCase();
 String y = "Y";
 String n = "N";
 String yup = "YES";
 String nope = "NO";
 if
((entry.equals(y))|(entry.equals(yup))|(entry.equals(n))|(entry.equals(
nope))){
   return true;
 }
 else{
  return false;
 }
}


}//class Keyboard
```

```
package java_code.utilities;
import java.io.*;

class KeyboardReader extends BufferedReader{

protected KeyboardReader(){  //constructor

super(new InputStreamReader(System.in));

//reads from the standard input stream

}
```

```
protected String readLine(String prompt){

  String returnable = null;
  System.out.println(prompt+"\n");

try{
  returnable =  super.readLine();
   }
 catch(IOException e){
 System.err.println("an IOException occured in method readline of class
KeyboardReader");
 System.err.println("this method is now returning null");
 }

return returnable;

}

}
```

---

package java_code.gui;

```
import java.awt.*;
import ptplot.*;
import java.util.*;

/* This class uses the class Plat from the package ptplot.  This
package was developed by _______ at U.C. Berkeley and is available for
public use. Plot allow the plotting of the BraggModeller data while it
is being calculated.  */

public class Plotter{  //Rajesh Patel 7/27/98

Plot p;
static QuickGuiWindow b;
Image offScreen;
private static Vector plots = new Vector();
private static Dimension d;
private static int count = 0;

public Plotter(String title){
 if (plots.isEmpty()){
   b = new QuickGuiWindow(title,800,480);
   this.d=b.getSize();
 }
 this.p = new Plot();
 //b = new QuickGuiWindow(title,500,500);
 b.invalidate();
 incrementCounter();
 b.add(p);
 b.doLayout();
 b.validate();
 b.show();
 p.init();
 plots.addElement(this);
```

```
 for (Enumeration e=plots.elements();e.hasMoreElements();){

(((Plotter)e.nextElement()).p).resize(((int)(d.width*0.9/(count))),((in
t)(d.height*0.75)));
  }
 b.invalidate();
 b.doLayout();
 b.validate();
 this.offScreen = p.createImage(p.getSize().width,p.getSize().height);
 p.setMarksStyle("none");

 }

public void addDataPoint(int series, double x, double y, boolean line){
//adds a data point
 p.addPoint(series,x,y,line);
 p.paint(offScreen.getGraphics());
 p.getGraphics().drawImage(offScreen,0,0,p);
}

public void setXRange(double min, double max){ //set the x axis range
  p.setXRange(min,max);
 }

public void setYRange(double min, double max){ //set the x axis range
  p.setYRange(min,max);
 }

public void addLegend(int set, String legend){
 p.addLegend(set,legend);
}

private void incrementCounter(){
 count++;
}

}
```

```
package java_code.gui;
import java.awt.*;
import java.awt.event.*;

/* This class is a convenience class that makes a GUI window with a
built-in anonymous listener class to implement correct window behavior.
This saves a lot of tiume when writing simple GUI programs in Java */

public class QuickGuiWindow extends Frame{

public QuickGuiWindow(){
  this("");
  }

public QuickGuiWindow(String title){
  super(title);
```

```
  setLayout(new FlowLayout(FlowLayout.LEFT));
  addWindowListener(new WindowAdapter(){
                                          public void
windowClosing(WindowEvent e){
                                               System.exit(0);
                                               }
                                               });
  }//constructor

public  QuickGuiWindow(String title, int w, int h){
      this(title);
      setSize(w, h);
    }//constructor


    }//class QuickGuiWindow
```

```
package java_code.mathematics;
/**
*@author Rajesh Patel
*@date 9/2/97
*@version 1.02 8/5/99
*/

import java.util.*;

/* This is a generic matrix of complex entries -- if m is a matrix, m.A
is the
multidimensional array of ComplexNumber (s) which represent the matrix.
future functionality to be added includes Matrix.Det() to find the
determinant of a non-singular matrix.
as well as Matrix.Inv() to find the inverse of a matrix.  */

/* Note:  the row and column indices for various methods in the class
may be
confusing at times.  As a general rule, if a method can be called from
outside
an instance of this class, the row and column indices for that method
call
should use normal matrix notation - with the first element in the
matrix being
1,1.  The methoid should then compensate for the array indexing
convention of java.
Internal methods such as elemMinor should be passed indices which
correspond
to the array conventions of java - that is, the first element of the
matrix is
0,0. */

/* The method assign() is an external access method - the method
assignElem()
is meant for internal use only -  aassignElem() requires input in java
array idex form
while assign() takes standard matrix notation*/
```

```
public class Matrix {

int rows, cols, dim;
ComplexNumber[][] A;
private String name;
/**
* constructs a Matrix of <code>ComplexNumber</code>s with the specified
number of rows and columns
* the entries in the matrix are initialized to 0+0i
*
* @param i the number or rows in this matrix
* @param j the number of columns in this matrix
*/
public Matrix(int i, int j){
//constructor

/*
* note:  8/5/99
* the preferred way to do this is
* int dims = [i,j];
* A =
Array.newInstance(Class.forName("java_code.mathematics.ComplexNumber",d
ims);
*
*this would require that we import java.lang.reflect.Array - but it's
cleaner - oh well
*/
ComplexNumber[][] B = new ComplexNumber[i][j];      /** here is an
interesting
example of lexical scope.  I originally tried to use 'A' in the line
above
this complied just fine, but, it resulted in the array A[][] of the
Matrix being assigned to
null.  The line above is within the lexical scope of the constructor.
If the variable 'B' has
the same name as the 'A' instance variable - the instance variable is
never initialized.  */

//i & j are the # of rows & colums respectively

rows = i;
cols = j;
dim = rows*cols;

 for (int x=0;x<rows;x++){
   for(int y=0;y<cols;y++){
     B[x][y] = new ComplexNumber(0,0);
   }//y loop
 }//x loop
A = B;
}//constructor

/**
* constructs a Matrix of <code>ComplexNumber</code>s with the specified
number of rows and columns and the specified name
* the entries in the matrix are initialized to 0+0i
*
```

```
* @param i the number or rows in this matrix
* @param j the number of columns in this matrix
* @param n the name of this matrix
*/
public Matrix(int i, int j, String n){  // another constructor

this(i,j);
name = n;
}//constructor
//#########################################################
/**
* constructs a matrix with the same dimensions as the given matrix
* and with the given name.
*
* @param m the matrix whose dimensions are to be used
* @param n the name of this matrix
*/
public Matrix(Matrix m, String n){//yet another constructor
this(m);
name = n;
}// constructor
//#########################################################
/**
* constructs a matrix with the same dimensions as the given matrix
*
* @param m the matrix whose dimensions are to be used
*/

public Matrix(Matrix m){  // another constructor

this(m.rows,m.cols);

}//constructor

//#########################################################
/**
* accessor method to assign the name of a Matrix
*
* @param m the matrix whose name is to be set
* @param n the name to be assigned to m
*/
public static void setName(Matrix m, String n){
m.name = n;
}
//#########################################################
/**
* accessor method to get the name of a Matrix
*
* @param m the Matrix whose name is sought
*
* @return a String containing the name of the matrix m
*/
public static String getName(Matrix m){
return m.name;
}

//#########################################################
```

```
/**
* assigns a value to a given element of a matrix
*
* @param s the matrix whose element is to be assigned
* @param u the row index of the lement to be assigned
* @param v the column index of the element to be assigned
* @param value the ComplexNumber to be assigned to element s(u,v) of
matrix s
*/
public static void assign(Matrix s, int u, int v, ComplexNumber value){
u-=1; // convert u & v to array indices
v-=1; //this is so we can use normal matrix notation

s.A[u][v] = value;

}//assign
//#########################################################
/**
* assigns a value to a given element of this matrix
*
* @param u the row index of the lement to be assigned
* @param v the column index of the element to be assigned
* @param value the ComplexNumber to be assigned to element (u,v) of
this matrix
*/

public void assign(int u, int v, ComplexNumber value){
u-=1; // convert u & v to array indices
v-=1; //this is so we can use normal matrix notation


(this.A[u][v]).setReal(value.getReal());
(this.A[u][v]).setImag(value.getImag());

//this.A[u][v] = value;
//this should perhaps be changed to make a new ComplexNumber in the
matrix
//since objects are passed by reference - change the variable 'value'
//in the calling program would change the matrix entry - very
undesirable.
//changed 4-4-2000 R.R.P.
}//assign
//#########################################################
/**
* assigns a double value to a given element of this matrix.
* this method is a convenience method since many matricies involve only
real quantities.
*
* @param u the row index of the lement to be assigned
* @param v the column index of the element to be assigned
* @param value the double to be assigned to element (u,v) of this
matrix
*/


public void assign(int u, int v, double value){
```

```
this.assign(u,v, new ComplexNumber(value,0));

}//assign
//#########################################################

private static void assignElem(Matrix s, int u, int v, ComplexNumber
value){
// don't convert u & v to array indices -  this in an internal method

s.A[u][v] = value;

}//assignElem
//#########################################################
/**
* resets all entries in this matrix to 0+0i
*/
public void flush(){
 for (int i = 1;i<=this.rows;i++){
  for (int j = 1;j<=this.cols;j++){
    assign(i,j,0);
  }
 }
}//flush

//#########################################################
private static ComplexNumber matrixElement(Matrix m, int i, int j){
/* this is a private method - indices are java array indices*/

ComplexNumber answer = null;

 if ((i <= m.rows) && (j <= m.cols)) {
     answer = m.A[i][j];
  }
 else { // handle error
   System.out.println("This matrix is smaller than the element you
asked for!");
  }

return answer;
}//Element
//#########################################################
/**
* accessor method to get a specified element form this matrix
*
* @param i the row index of the element being requested
* @param j the column index of the element being requested
*
* @return the ComplexNumber entry located at element (i,j) of this
matrix
*/
public ComplexNumber element(int i, int j){
/* this is a public method - indices are regular matrix indices*/
i=i-1;
j=j-1;  //convert to java array indices

ComplexNumber answer = null;
```

```
 if ((i <= this.rows) && (j <= this.cols)) {
     answer = this.A[i][j];
  }
 else { // handle error
   System.out.println("This matrix is smaller than the element you
asked for!");
  }

return answer;
}//element
//#########################################################
/**
* Generates an identity matrix.
*
* @param n the size of the matrix (eg. number of rows and columns.
*
* @return a square matrix with 1's along the diagonal, and zeros
everywhere else.
*/
public static Matrix identityMatrix(int n){
 Matrix returnable = new Matrix(n,n);
 ComplexNumber dummy;
     for(int i=1;i<n+1;i++){
      dummy = returnable.element(i,i);
      dummy.setReal(1);
      dummy.setImag(0);
     }
/*since objects get passed by reference, assign dummy to equal each
diagonal
*element in turn simply points to each element - changin dummy then
changes
*the value in the matrix itself.
*/

return returnable;
}
//#########################################################
/**
* performs multiplication of two matricies in accordance with the rules
of matrix multiplication
*
* @param m the first matrix
* @param n the second matrix
*
* @return a matrix representing the product m*n (in that order)
*/

public static Matrix matrixProd(Matrix m, Matrix n){
//Performs Matrix Multiplication

Matrix p;
if  (m.cols != n.rows){

p = null;

}
```

```
else{
 p = new Matrix(m.rows,n.cols);
  for (int i = 0; i<m.rows; i++){
     for (int j = 0; j< n.cols; j++){
        ComplexNumber d = new ComplexNumber(0,0);
         for (int k = 0; k < m.cols; k++){
               ComplexNumber mSubIK = m.A[i][k];
               ComplexNumber nSubKJ = n.A[k][j];
               d =
ComplexNumber.complexSum(d,ComplexNumber.complexProd(mSubIK,nSubKJ));
         }//for loop - k
              p.A[i][j] = d;
     }//for loop j
  }//for loop i
}//else statement
 return p;

}//method matrixProd()
//#########################################################
/**
* multiplies a matrix by a complexNumber
*
* @param m the matrix to be multiplied
* @param aScalar the "scalar" ComplexNumber to be multiplied
*
* @return a new matrix whose elements equal the elements of m
multiplied by the complex number aScalar
*/
public static Matrix matrixProd(Matrix m, ComplexNumber aScalar){
//Multiplies a Matrix by a scalar

Matrix p;

 p = new Matrix(m.rows,m.cols);
  for (int i = 0; i<m.rows; i++){
     for (int j = 0; j< m.cols; j++){
        ComplexNumber d =
ComplexNumber.complexProd(Matrix.matrixElement(m,i,j),aScalar);
              Matrix.assignElem(p,i,j,d);
     }//for loop j
  }//for loop i

 return p;

}//method matrixProd() -- overloaded for scalar multiplication
//#########################################################
/**
* multiplies a matrix by a double
*
* @param m the matrix to be multiplied
* @param aNonComplexScalar the "scalar" double to be multiplied
*
* @return a new matrix whose elements equal the elements of m
multiplied by the double aNonComplexScalar
*/
public static Matrix matrixProd(Matrix m, double aNonComplexScalar){
//Multiplies a Matrix by a scalar
```

```
Matrix p;
ComplexNumber aScalar = new ComplexNumber(aNonComplexScalar,0);
 p = new Matrix(m.rows,m.cols);
  for (int i = 0; i<m.rows; i++){
     for (int j = 0; j< m.cols; j++){
        ComplexNumber d =
ComplexNumber.complexProd(Matrix.matrixElement(m,i,j),aScalar);
              Matrix.assignElem(p,i,j,d);
     }//for loop j
  }//for loop i

 return p;

}//method matrixProd() -- overloaded for scalar multiplication
//#########################################################
/**
* adds two matrices together following the usual rules of matrix
addition
*
* @param m the first matrix
* @param n the second matrix
*
* @return a new matrix which is the sum of matricies m and n
*/
public static Matrix matrixSum(Matrix m, Matrix n){
//Adds two matrices

Matrix p; //the matrix which is returned

if ((m.rows==n.rows) && (m.cols==n.cols)){   /* can only add matrices
if they have the
same dimensions*/
 p = new Matrix(m.rows,m.cols);
  for (int i=0;i<m.rows;i++){
     for (int j=0;j<m.cols;j++){
       p.A[i][j] = ComplexNumber.complexSum(m.A[i][j],n.A[i][j]);
     } //j loop
  }//i loop
 }// if
else{
 p = null;
 System.out.println("These Matrices don't have the same dimensions");
}//else

return p;
}// matrixSum
//#########################################################
/**
* subtracts two matrices following the usual rules of matrix addition
*
* @param m the first matrix
* @param n the second matrix
*
* @return a new matrix which equals m-n
*/
```

```
public static Matrix matrixDiff(Matrix m, Matrix n){
//subtracts two matrices

Matrix p; //the matrix which is returned

if (m.rows==n.rows && m.cols==n.cols){   /* can only add matrices if
they have the
same dimensions*/
 p = new Matrix(m.rows,m.cols);
  for (int i=0;i<m.rows;i++){
     for (int j=0;j<m.cols;j++){
       p.A[i][j] = ComplexNumber.complexDiff(m.A[i][j],n.A[i][j]);
     } //j loop
  }//i loop
 }// if
else{
 p = null;
 System.out.println("These Matrices don't have the same dimensions");
}//else

return p;
}// matrixDiff
//#########################################################
/**
* checks whether a given matrix is square.
*
* @param m the matrix
*
* @return <code>true</code> if the number of rows equals the number of
columns
*         <code>flase</code> otherwise
*/
public static boolean isSquare(Matrix m){

boolean flag = false;

 if (m.rows==m.cols) flag = true;
return flag;
}

//#########################################################
/**
* the trace of a given matrix
*
* @param m the matrix whose trace is sought
*
* @return a ComplexNumber whose value is equal to the trace of m
*/
public static ComplexNumber matrixTrace(Matrix m){

ComplexNumber Trace; //need a variable

if (m.isSquare(m)){
 Trace = new ComplexNumber(0,0);
 for (int i=0;i<m.rows;i++){
   Trace=ComplexNumber.complexSum(Trace,m.A[i][i]);
 }//for loop
```

```
}//if block

else{
  System.out.println("This Matrix is not a square matrix!!!!");
  Trace = null;
}//else block

return Trace;
}//matrixTrace
//#########################################################
/**
* prints a given matrix to the standard System output device
*
* @param m the matrix to be printed
*/
public static void matrixPrint(Matrix mat){

//prints the matrix to the screen
if (mat.name!=null) System.out.println("The matrix "+mat.name+" is :");
 System.out.println();
 for (int i = 0;i<mat.rows;i++){
  for (int j=0;j<mat.cols;j++){
   System.out.print(mat.A[i][j].real+"+"+mat.A[i][j].imag+"i  ");
  }//j loop
  System.out.println();
 }// i loop

System.out.println();
}//matrixPrint
//#########################################################
/**
* the determinant of a given matrix
*
* @param m the matrix whose determinant is sought
*
* @return a ComplexNumber whose value is the determinant of m
*/
public static ComplexNumber matrixDet(Matrix m){

/*This method uses the first row of the matrix to calculate the
determinant by the procedure of
multiplication by minors and cofactors  Since the first row is always
used, the loop counter i
plus 1 (i+1) is the exponent for the calculation of the cofactor.*/

ComplexNumber D = null; // to hold the value of the determinant

if (Matrix.isSquare(m)){ // can't find the determinant of a non-square
matrix

  D = new ComplexNumber(0,0);

    if (m.dim==1){
     D = Matrix.matrixElement(m,0,0);
    }
    else{
```

```
/**     if (m.rows==2){      // teach me how to do a 2x2 matrix!
      ComplexNumber Q =
ComplexNumber.complexProd(Matrix.matrixElement(m,0,0),Matrix.matrixElem
ent(m,1,1));
      ComplexNumber T =
ComplexNumber.complexProd(Matrix.matrixElement(m,0,1),Matrix.matrixElem
ent(m,1,0));
      D = ComplexNumber.complexDiff(Q,T);
      }

   else{   */

       for (int i=0;i<m.rows;i++){ // loop through and find the
determinant
         ComplexNumber Q =
ComplexNumber.complexProd(Matrix.matrixElement(m,0,i),Matrix.matrixDet(
Matrix.minorOfElement(m,0,i)));
         Q =
ComplexNumber.complexProd(Matrix.cofactorOfElement(m,0,i),Q);
         D = ComplexNumber.complexSum(D,Q);
         Q = null;
       }//loop
     }//else
}

else {

     if (Matrix.getName(m) == null) {
       System.out.println("This matrix is not Square - the Determinant
can't be calculated");
         }
     else{
       System.out.println("The matrix "+Matrix.getName(m)+" is not
square - the Determinant can't be calculated");
       }//nested else

}// else for non-square matrix


return D;
}//matrixDet
//#########################################################
private static Matrix minorOfElement(Matrix n, int i, int j){
// i and j are the row and col. indices respectively

Matrix returnable = null;

 if ((n.rows==1) || (n.cols==1)){
    System.out.println("This matrix has no minors!!!");

 }

 else {
    Matrix ret = new Matrix((n.rows),(n.cols));

     for (int a=0;a<n.rows;a++){
         for (int b=0;b<n.cols;b++){
```

```
            if ((b==j)||(a==i)){

              Matrix.assignElem(ret,a,b, new ComplexNumber(-99999,-
99999));
            }
                 /** if its the same row or column as the element
you're taking the minor
                     of - assign -99999+-99999i, or else assign the
value*/
            else{

              Matrix.assignElem(ret,a,b, Matrix.matrixElement(n,a,b));
            }
          }//b loop
        }//a loop

        returnable = new Matrix((ret.rows-1),(ret.cols-1));
         int w=0;
        for (int a=0;a<ret.rows;a++){
          int z = 0;
           for (int b=0;b<ret.cols;b++){
              if (Matrix.matrixElement(ret,a,b).real!=-99999){
              assignElem(returnable,w,z,Matrix.matrixElement(ret,a,b));
              z++;
              }
          /** copy all of the non-null entries to the new matrix to be
returned*/
              if (z>=returnable.cols){
                z = 0;
                w++;
              }
           } // b loop
         }// a loop

 } // else

return returnable;
}//minorOfElement

private static double cofactorOfElement(Matrix m,int i, int j){

/**note: this method doesn't really need the matrizx to be passed
it just makes the method call easier to read elsewhere in the code.*/

double cofact;

if ((i+j)%2==0) {
 cofact = 1;
 }
else {
 cofact = -1;
}

return cofact;
}//cofactorOfElement
```

```
/**
* the inverse of a given matrix
*
* @param m the matrix whose inverse is sought
*
* @return a new Matrix which is the inverse of m (if m is a square,
non-singular matrix)
*  otherwise null
*/
public static Matrix matrixInv(Matrix m){
//Finds the inverse of the given matrix

Matrix returnable = null;

if (Matrix.isSquare(m)){
  if ((Matrix.matrixDet(m).real)!=0) {
    returnable = new Matrix(m.rows,m.cols);
    for (int i=0;i<m.rows;i++){
      for (int j=0;j<m.cols;j++){
        Matrix temp =
Matrix.matrixProd(Matrix.minorOfElement(m,j,i),Matrix.cofactorOfElement
(m,i,j));
        ComplexNumber entry =
ComplexNumber.complexDiv(Matrix.matrixDet(temp),Matrix.matrixDet(m));
        Matrix.assignElem(returnable,i,j,entry);
      }
     }
   }
   else{
    System.out.println("This matrix is square but it's singular - no
inverse exists!!");
   }
}
else{
 System.out.println("This matrix is not square - i cannot calculate an
inverse!");
 }
 return returnable;
 }//matrixInv

}//class
```

```
package java_code.mathematics;
//import Functions;

/**
* This class provides a java implementation of complex numbers.
* Both rectangular and polar forms of complex numbers are supported.
* Mathematical operations such as addition, subtraction, multiplication
and
* division are provided.  Also, complex conjugation and complex square
roots are
* provided.
* @author Rajesh R. Patel
```

```
* @date 9/1/97
* @version 1.02 8/5/99
*/

public class ComplexNumber{
/* This class provides functionality for complex arithmetic*/

double real,imag,mag,phase;

/**
* A no-arg concstuctor - this should only be used in beans
developement.
*/
public ComplexNumber(){}; //generic Constructor

/**
* Constructs a complex number with the specified real and imaginary
parts in rectangular form
*  note: the polar form is automatically set once the number is
created.
*
* @param r the real part of the complex number
* @param i the imaginary part of the complex number
*/
public ComplexNumber(double r, double i){            //Constructor

  this.real = r;
  this.imag = i;
  setPolarForm();

}

/**
* Constructs a complex number in either rectangular or polar form
depending on the
*  value of the <code>boolean</code> flag polar.
*
* @param m the magnitude of the number if polar is true
*          or the real part of the number if polar is false.
* @param ph the phase of the complex number if ploar is true
*          or the imaginary part of the number if polar is false.
* @param polar the flag to determine the format of the data
*        true indicates that the data is in polar form (magnitude and
phase given)
*        false indicates that the data is in rectangular form (real and
imaginary parts given)
*/

public ComplexNumber(double m, double ph, boolean polar){

 this(m,ph);
 if (polar){
   this.mag = m;
   this.phase = ph;
   setRectForm();
  }
```

```
}//constructor 2

/**
* provides multiplication of two complex numbers
* @param x the first number
* @param y the second number
*
* @return a new <code>ComplexNumber</code> which is the product of x
and y
*/
public static ComplexNumber complexProd(ComplexNumber x, ComplexNumber
y){
// provides multiplication

double a = (x.real*y.real) - (x.imag*y.imag);
double b = (x.real*y.imag) + (x.imag*y.real);

return new ComplexNumber(a,b);
}
//#########################################################
/**
* provides addition of two complex numbers
* @param x the first number
* @param y the second number
*
* @return a new <code>ComplexNumber</code> which is the sum of x and y
*/

public static ComplexNumber complexSum(ComplexNumber x, ComplexNumber
y){
// adds two complex numbers

double a = x.real+y.real;
double b = x.imag+y.imag;

return new ComplexNumber(a,b);
}
//#########################################################
/**
* provides subtraction of two complex numbers
* @param x the first number
* @param y the second number
*
* @return a new <code>ComplexNumber</code> which equals y subtracted
from x
*/

public static ComplexNumber complexDiff(ComplexNumber x, ComplexNumber
y){
// subtracts two complex numbers

double a = x.real-y.real;
double b = x.imag-y.imag;

return new ComplexNumber(a,b);
}
//#########################################################
```

```
/**
* provides complex conjugation of a complex number
* @param x the complex number to be conjugated
*
* @return a new <code>ComplexNumber</code> which is the complex
conjugate of x
*/

public static ComplexNumber complexConjugate(ComplexNumber x){

double a = x.real;
double b = (-1*x.imag);

return new ComplexNumber(a,b);
}
//#######################################################
/**
* the square magnutide of a complex number
* @param x the complex number
*
* @return a <code>double</code> which is the square magnitude of x
*/


public static double complexMagSQ(ComplexNumber x){
//the absolute magnitude squarde of a complex number

/*ComplexNumber s = complexConjugate(x);
*return ComplexNumber.complexProd(x,s);
*/

return (x.mag*x.mag); // there won't be an imaginary part
}
//#######################################################
/**
* provides division of two complex numbers
* @param x the dividend
* @param y the divisor
*
* @return a new <code>ComplexNumber</code> which is the quotient of x
divided by y
*/

public static ComplexNumber complexDiv(ComplexNumber x, ComplexNumber
y){
// divides a complex number x by another complex number y
  double a =
(complexProd(x,complexConjugate(y)).real)/(complexMagSQ(y));
  double b =
(complexProd(x,complexConjugate(y)).imag)/(complexMagSQ(y));

return new ComplexNumber(a,b);
}
//#######################################################
/**
* provides convenient multiplication of a double with a complex number
* @param x the double
```

```
* @param y the complex number
*
* @return a new <code>ComplexNumber</code> which is the product of x
and y
*/


public static ComplexNumber complexProd(double x, ComplexNumber y){

double a = x*y.real;
double b = x*y.imag;

return new ComplexNumber(a,b);
}
//#########################################################
/**
* the magntude of a complex number
* @param x the comple number
*
* @return a <code>double</code> which is the magnitude of x
*/

public static double complexMag(ComplexNumber x){
// gives the magnitude of a complex number

/*
*return Math.sqrt(complexMagSQ(x));
*/
return x.mag;

}
//#########################################################
/**
* calculates one of the square root of a complex number
* @param n the complex number
*
* @return a new <code>ComplexNumber</code> which is the square root of
x.
* The magnitude of the result is the square root of the absolute
magnitude of n.
* The phase is equal to the phase of n divided by 2.
*/

public static ComplexNumber complexSqrt1(ComplexNumber n){

  return(new ComplexNumber(Math.sqrt(n.mag),(n.phase/2),true));
}
//#########################################################
/**
* calculates one of the square root of a complex number
* @param n the complex number
*
* @return a new <code>ComplexNumber</code> which is the square root of
x.
* The magnitude of the result is the square root of the absolute
magnitude of n.
* The phase is equal to PI plus the phase of n divided by 2.
```

```
*/

public static ComplexNumber complexSqrt2(ComplexNumber n){

  return(new ComplexNumber(Math.sqrt(n.mag),(n.phase/2+Math.PI),true));
}

//#######################################################
/**
* called to update the rectangular form of the ComplexNumber whenever a
change
* is made to the polar form. Updates are usually performed
automatically.  The method
* is made available only to support legacy code.
*/
public void setRectForm(){
  this.real = this.mag*Math.cos(this.phase);
  this.imag = this.mag*Math.sin(this.phase);
  }

/**
* called to update the polar form of this ComplexNumber whenever a
change is made
* to the rectangular form.  Updates are usually performaed
automatically.  This method
* is made availabde only to support legacy code.
*/

public void setPolarForm(){
  this.mag = Math.sqrt(this.real*this.real + this.imag*this.imag);
  if (this.mag==0) {
   this.phase = 0;
   return;
   }

  if (this.real==0){
      this.phase = (Math.PI/2);
      if (this.imag<0){
       this.phase*=-1; //make it -PI/2
       }
    }//real==0
  else{
    this.phase = Math.atan2(this.imag,this.real);
  /*
  *  if this.phase<0{
  *   this.phase+=(2*Math.PI);
  *   }
  */

  }//else

 }//setPolarForm
//#######################################################
/**
* accessor method to mofidy the imaginary part of this ComplexNumber.
* The polar form is automatically updated when a change is made.
*
```

```
* @param num the new value for the imaginary part of this complexNumber
*/
public void setImag(double num){

 this.imag = num;
 setPolarForm();
}
//#########################################################
/**
* accessor method to mofidy the real part of this ComplexNumber.
* The polar form is automatically updated when a change is made.
*
* @param num the new value for the real part of this complexNumber
*/

public void setReal(double num){
 this.real = num;
 setPolarForm();
}

//#########################################################
/**
* accessor method to mofidy the magnitude of this ComplexNumber.
* The rectangular form is automatically updated when a change is made.
*
* @param m the new value for the magnitude of this complexNumber
*/

public void setMag(double m){
 this.mag = m;
 setRectForm();
}
//#########################################################
/**
* accessor method to mofidy the phase of this ComplexNumber.
* The rectangualt form is automatically updated when a change is made.
*
* @param p the new value for the phase of this complexNumber
*/

public void setPhase(double p){
 this.phase = p;
 setRectForm();
}
//#########################################################
/**
* accessor method to get the argument (phase) of this ComplexNumber.
*
* @param z the complexNumber whose argument is sought
*
* @return a double equal to the phase of z in radians
*/

public static double arg(ComplexNumber z){
 return z.phase;
}
//#########################################################
```

```
/**
* checks whether this ComplexNumber is purely real.
* note: due to numerical truncation, this method may provide
analytically inaccurate results if used on
* a ComplexNumber whose imaginary part was determined through numerical
calculations.
*
* @return <code>true</code> if the imaginary part of this complex
number equals zero,
*         <code>false</code> otherwise
*/
public boolean isReal(){
 return ((this.imag==0.0) ? true:false);
}

//#####################################################
/**
* checks whether this ComplexNumber is purely imaginary.
* note: due to numerical truncation, this method may provide
analytically inaccurate results if used on
* a ComplexNumber whose real part was determined through numerical
calculations.
*
* @return <code>true</code> if the real part of this complex number
equals zero,
*         <code>false</code> otherwise
*/

public boolean isImaginary(){
 return ((this.real==0.0) ? true:false);
}

/**
* accessor method to get the real part of this ComplexNumber.
*
* @return a double equal to the real part of this ComplexNumber.
*/
public double getReal(){
 return real;
}

/**
* accessor method to get the imagiunary part of this ComplexNumber.
*
* @return a double equal to the imaginary part of this ComplexNumber.
*/
public double getImag(){
 return imag;
 }

/**
* accessor method to get the magnitude part of this ComplexNumber.
*
* @return a double equal to the magnitude (polar form) of this
ComplexNumber.
*/
public double getMag(){
```

```
 return mag;
 }

/**
* accessor method to get the phase part of this ComplexNumber.
*
* @return a double equal to the phase part of this ComplexNumber.
*/
public double getPhase(){
 return phase;
 }

}//end of class ComplexNumber
```

```
package java_code.modelling;
import java.io.*;
import java_code.utilities.*;
/**
*A generic base class for models.  This class provides default input
and output
*file allocation methods.
*
*@author Rajesh R. Patel
*@version 1.1
*/
public class Model implements Serializable, Configurable{

/*this is a utility class*/

String infn,outfn;

/**
*gets an input file to read in model configuration information
*
*@return a BufferedReader to read a text file containing model setup
information.
*/
public BufferedReader getInputFileName(){
  BufferedReader returnable = null;
  this.infn = Keyboard.readLine("Enter the name of the input file : ");
  try{

    File infile = new File(infn);
     if (!(infile.exists())){
      throw new FileNotFoundException("This file does not exist!!!");
     }
    returnable = new BufferedReader(new FileReader(infile));

    }//try
  catch(FileNotFoundException fnf){
    System.out.println("The file "+infn+" was not found - please try
again");
    }
  return returnable;

}//getInputFileName()
```

```
//#########################################################

/**
*A no-arg version of method getOutputFileName(String).
*
*@return the PrintWriter to write textual output from this model.
*@see #getOutputFileName(String)
*/
public PrintWriter getOutputFileName(){

   PrintWriter returnable = getOutputFileName("Enter the name of the
output file");

/*
   this.outfn = Keyboard.readLine("Enter the name of the output file :
");
   System.out.println ("The output file name is : "+outfn);
 try{
   File outfile = new File(outfn);
   returnable = new PrintWriter(new BufferedWriter(new
FileWriter(outfn)));
     }
 catch(IOException e){
      System.out.println("There was an i/o exception in method
getOutputFileName()");
      }

*/
   return returnable;
 }
//#########################################################

/**
*gets a <code>PrintWriter</code> to which model output can be saved as
text.
*
*@param prompt the prompt to ask the model user the file name
*@return a PrintWriter to which output will be written.
*/
public PrintWriter getOutputFileName(String prompt){

   PrintWriter returnable = null;
   this.outfn = Keyboard.readLine(prompt);
   if (outfn.endsWith(".in")){
   try{
    if (Keyboard.readYN("You chose a file name ending with .in shall I
change it to .out?")){
      String outtemp = outfn.substring(0,outfn.length()-3);
      outfn = outtemp+".out";
      }
    }//try
   catch(Exception e){
    System.err.println("Look What Happened! "+e);
    System.exit(1);
    }
    }
  try{
```

```
     File outfile = new File(outfn);
     if (outfile.exists()) {

         if (verifyOverWrite(outfn)==true){
           returnable = new PrintWriter(new BufferedWriter(new
FileWriter(outfn)));
           }
         else{//don't over-write
           String Default = this.getClass().getName()+"_Default.out";
           System.out.println("The output file will be called :
"+Default);
           returnable = new PrintWriter(new BufferedWriter(new
FileWriter(new File(Default))));
           }
       } //if exists

      else{ //the file doesn't already exist --
         returnable = new PrintWriter(new BufferedWriter(new
FileWriter(outfn)));
          }
   }//try
 catch(IOException e){
      System.out.println("There was an i/o exception in method
getOutputFileName()");
      }
   return returnable;
 }
//#########################################################
/**
*called when the file outfn already exists.  This method is meant to
prevent inadvertent
*erasure of data files from different model runs.
*
*@param outfn a String containing the name of the file to be verified
*@return true if the file should be overwritten false otherwise.
*/
public static boolean verifyOverWrite(String outfn) throws IOException{
  boolean returnable;

  System.out.println("The file "+outfn+" already exists!!!");
  returnable =  Keyboard.readYN("Do you want to OverWrite this file y/n
??? ");
  return returnable;

}
//#########################################################

/**
*The following methods are stubs for the interface Configurable.
*Note that subclasses of Model MUST override isReady() in order for the
*model to run at all since the default method returns false.
*/
public void go(){}

/*
*eventually will be used to make sure the model is properly configured
before
```

```
*a model run is initiated.  This method is intended for GUI support.
*
*return whether the model is properly configured to start a model run.
*/
public boolean isReady(){
 return false;
 }

} //class model
```

```
package java_code.modelling;
import java.io.*;
import java_code.utilities.*;
import java.lang.reflect.*;

public final

class Model2{

/*this is a utility class*/

private Model2(){} //you can't instantiate this class

public static BufferedReader getInputFileName(){
  BufferedReader returnable = null;
  String infn = Keyboard.readLine("Enter the name of the input file :
");
  try{

    File infile = new File(infn);
     if (!(infile.exists())){
      throw new FileNotFoundException("This file does not exist!!!");
     }
    returnable = new BufferedReader(new FileReader(infile));

    }//try
  catch(FileNotFoundException fnf){
    System.out.println("The file  "+infn+"  was not found - please try
again");
    System.exit(1);
    }
 return returnable;

}//getInputFileName()

public static BufferedReader getInputFileName(String prompt){
  BufferedReader returnable = null;
  String infn = Keyboard.readLine(prompt);
  try{

    File infile = new File(infn);
     if (!(infile.exists())){
      throw new FileNotFoundException("This file does not exist!!!");
     }
    returnable = new BufferedReader(new FileReader(infile));
```

```
    }//try
  catch(FileNotFoundException fnf){
    System.out.println("The file  "+infn+"  was not found - please try
again");
    System.exit(1);
    }
 return returnable;

}//getInputFileName()


public static PrintWriter getOutputFileName(){

   PrintWriter returnable = null;
   String outfn = Keyboard.readLine("Enter the name of the output file
: ");
   System.out.println ("The output file name is : "+outfn);
 try{
   File outfile = new File(outfn);
   returnable = new PrintWriter(new BufferedWriter(new
FileWriter(outfn)));
     }
 catch(IOException e){
      System.out.println("There was an i/o exception in method
getOutputFileName()");
      }
   return returnable;
 }


public static PrintWriter getOutputFileName(String prompt){

   PrintWriter returnable = null;
   String outfn = Keyboard.readLine(prompt);
  try{
     File outfile = new File(outfn);
     if (outfile.exists()) {

         if (verifyOverWrite(outfn)==true){
           returnable = new PrintWriter(new BufferedWriter(new
FileWriter(outfn)));
           }
         else{//don't over-write
           String Default = "Default.out";
           System.out.println("The output file will be called :
"+Default);
           returnable = new PrintWriter(new BufferedWriter(new
FileWriter(new File(Default))));
           }
       } //if exists

      else{ //the file doesn't already exist --
         returnable = new PrintWriter(new BufferedWriter(new
FileWriter(outfn)));
          }
   }//try
 catch(IOException e){
```

```
      System.out.println("There was an i/o exception in method
getOutputFileName()");
      }
   return returnable;
 }

public static boolean verifyOverWrite(String outfn) throws IOException{
  boolean returnable;

  System.out.println("The file "+outfn+" already exists!!!");
  returnable =  Keyboard.readYN("Do you want to OverWrite this file y/n
??? ");
  return returnable;

}

public static void assignModelParameters(Object b){

StreamTokenizer s = new StreamTokenizer(getInputFileName());
Field[] vars = b.getClass().getFields();
try{
s.parseNumbers();
s.nextToken();
s.slashStarComments(true);
s.slashSlashComments(true);

while(s.ttype!=s.TT_EOF){

 if (s.ttype==s.TT_WORD){
   String v = s.sval;
   v = v.toUpperCase();
   v = v.trim();
 find:  //this creates a named for loop
   for (int i=0;i<vars.length;i++){
     String n = vars[i].getName();
     n = n.toUpperCase();
     n = n.trim();

     if (v.equals(n)){
      System.out.println("Found variable "+n);
      s.nextToken();

       if (s.ttype==s.TT_NUMBER){
         assign(b,vars[i],s.nval);
           }
       else { //this will have to be expanded later  to cover all data
types - string, boolean, character etc.
         assign(b,vars[i],s.sval);
            }//else
       break find;
     }
    }//find loop
  }
 s.nextToken();
 }//while

}//try
```

```
catch(IOException e){
 System.out.println("There was an IOException in method
AssignModelParameters");
 System.exit(1);
}//catch
}//assignModelParameters


public static void assign(Object b, Field f, double num){

 Double d = new Double(num);
 String vtype = f.getType().getName();

try{

 if (vtype.equals("byte")){
    byte x = d.byteValue();
    f.setByte(b,x);
    }

 if (vtype.equals("int")){
   int x = d.intValue();
   f.setInt(b,x);
   }

if (vtype.equals("float")){
  float x = d.floatValue();
  f.setFloat(b,x);
  }

if (vtype.equals("long")){
  long x = d.longValue();
  f.setLong(b,x);
  }

if (vtype.equals("short")){
 short x = d.shortValue();
 f.setShort(b,x);
 }

if (vtype.equals("double")){
 double x = d.doubleValue();
 f.setDouble(b,x);
 }

}//try
catch(IllegalAccessException e){
 System.out.println("There was a problem assigning variable
"+f.getName());
}//catch


}//assign- for numbers

private static void assign(Object b, Field f, String s){

 String  vtype = f.getType().getName();
```

```
try{
  if (vtype.equals("boolean")){
   boolean bv = (Boolean.valueOf(s)).booleanValue();
   f.setBoolean(b,bv);
   }
  else{
  f.set(b,s);
    }
  }
catch(IllegalAccessException e){
 System.out.println("There was a problem accessing the string variable
"+f.getName());
}

}//assign for strings

} //class model2
```

```
package java_code.modelling;

public interface Configurable{

/**The Configurable interface will be implemented by all "Models" which
*can be run.  This will enable me to build a generic UI for all models
*which will instattiate a model object, read in the expected fields,
and
*permit saving of the model configuration through object serialization.
*
*@Date          1/24/98
*@Version       1.0a
*@Author        Rajesh Patel
*/


/*The <code> go() <code/> method is the method which runs the model.
Implementations of the
*<code> go() <code/> method will have to check that the model is
properly
*configured top run.
*/

 public void go();


/*The <code> isReady() <code/> method must return true if the model is
properly
*configured to run.  Implementations of this method can be balnk,
simply returning
*a boolean, or can have checks on the fields of model to make sure that
model parameters
*are configured correctly.
*/
 public boolean isReady();

}//interface
```

Example Input File for BraggModeller

```
/*
This is a congifuragtion input file for the model: BraggModeller.
Enter the value for each variable after the variable name

Do not use commas only spaces - the variable type is indicated
You can enter comments at the bottom of this header before the star-
slash

*/

lambdaStart  800   //double - in nanometers

lambdaEnd   900  //double - in nanometers - should be greater than
                  //lambdaStart

deltaLambda  2  //double - in nanometers

lambdaCenter  850   //double - in nanometers - used for default layer
                   //thickness reference

thetaStart 45   //double - in degrees

thetaEnd  45   //double  - in degrees - should be <= theta start

deltaTheta  1.0   //double - in degrees

ymin -33 //double - in dB graph limit for dynamicPlotting

ymax 3  // double - in dB graph limit for dynamicPlatting

plotTE true  //boolean plots TE transmission (if dynamicPlotting=true)

plotTM true  //boolean plots TM transmission (if dynamicPlotting=true)

designAngle 45   //in degrees angle which the filter is designed for

dynamicPlotting true  //boolean set to true to see the R and T curves
                     //as they are calculated

parseAngles false   //boolean saves data for each incident angle

plotRTE true   //boolean set to true to plot TE reflection graph

plotRTM true  //boolean set to true to plot TM reflection graph

saveData false  //boolean  indicates whether data is sved or not
```

Example Filter Definition File for BraggModeller

```
incident|incident=1.5,t=1
H|al=0.15,t=0.25
L|n=2.1,t=100nm
M|n=3,t=0.5,wl=825
cavity|al=1.0,t=0.5
substrate|boundary=2.5,t=1
{incident H M L [H L]^10 cavity [L H]^10
M [H [L M]^3 H]^2
substrate}

/*
*  STACK INPUT FILE USAGE AND CONVENTIONS:
*
*  The layer definitions MUST come at the top of the file,
*  only one definition per line.
*  All Stacks MUST begin with the LayerSymbol incident and of type
incident or
*     type boundary.
*  All stacks MUST end with a layer of type incident, substrate or
boundary.
*
*  Layer Definition syntax is as follows:
*
*
LayerSymbol|LayerType=value,LayerThickness=value,<optional>wl=value
*
*  Where
*
*  LayerSymbol is the symbol for this layer in the stack definition.
*
*  LayerType can be:
*
*   incident (for first and last layers only)
*
*   substrate (last layer only)
*
*   boundary (for first or last layer - this is an alternate syntax)
*
*   al - indicates a Bragglayer of Aluminum Galium Arsenide
*      with the indicated aluminum mole fraction (must be <=1)
*
*   any other LayerType indicates a BraggLayer with indicated constant
index
*      of refraction such as layer "L" above which uses "n=2.1" to
*      indicate a layer with constant index of 2.1
*
*  LayerThickness:
*
*   indicates the thickness of the layer
*      in waves at the center wavelength (default)
*
*   Layer thicknesses can also be given in nanometers
*      as show in the layer called "L"
*
```

```
*   Layer Thickness in waves at a wavelength other than the center
wavelength
*       can be defined using the optional flag: wl=value
*       to indicate the reference wavelength in nanometers - for example
*       the layer "M" above uses a reference wavelength of 825
nanometers.
*
*  An opening curly brace "{" MUST immediately follow the last layer
*       definition this is the flag that indicates
*       the start of the stack definition.
*
*  The LayerSymbols are used to define the filter stack.
*       Each symbol must be separated by a space or comma
*        for the stack be be parsed properly.
*
*  A block of layers can be designated with square brackets "[" and "]"
*       once a block is designated, it can be repeated any number of
times
*       simply by using the carat and an integer to indicate the
*       number of repititions.
*       for example: [H L]^10  will repeat the layers H and L 10 times.
*
*  Blocks can be nested any number of times
*       for example [[[H L]^2 M]^2 X]^3 - this sequence will produce:
*         H L H L M H L H L M X H L H L M H L H L M X H L H L M H L H L
M X
*
*  The user must make sure that all blocks are properly started and
ended.
*
*  A closing curly brace "}" MUST be used to terminate the stack
definition.
*
*  Following the stack definition, any comments can be placed
*       using the c or c++ language conventions.
*       This comment block is an example of such comments.
*/
```

BraggModeller Filter Definition Input file for the 5-Cavbity λ-Connect filter

```
incident|incident=1.5,t=1
h|al=0.15,t=0.25
l|al=1.0,t=0.25
ca|al=0.15,t=0.5
cw|al=0.15,t=1.0
cl|al=0.15,t=1.5
{incident [h l]^4 ca l [h l]^14 cw l [h l]^16 cl l
[h l]^16 cw l [h l]^14 ca [l h]^4
incident}
```

Appendix D Full Listing of Filter Layer Structures

This appendix provides full listings of the layer structures of the filters grown for this dissertation research. Listings are provided for:

Single Cavity Fabry-Perot filter described in Chapters 2 and 3

5-Cavity Filter designed for λ-Connect

Samples A – F for Lateral Oxidation Studies.

Chapters 2 & 3

Single Cavity Filter, 10 Bragg Pairs per DBR

Layer Number	Aluminum Fraction	Thickness (nm)	
1	0.15	60.85	
2	1.00	71.18	
3	0.15	60.85	
4	1.00	71.18	
5	0.15	60.85	
6	1.00	71.18	
7	0.15	60.85	
8	1.00	71.18	
9	0.15	60.85	
10	1.00	71.18	
11	0.15	60.85	
12	1.00	71.18	
13	0.15	60.85	
14	1.00	71.18	
15	0.15	60.85	
16	1.00	71.18	
17	0.15	60.85	
18	1.00	71.18	
19	0.15	60.85	
20	1.00	71.18	
21	0.15	121.70	Cavity
22	1.00	71.18	
23	0.15	60.85	
24	1.00	71.18	
25	0.15	60.85	
26	1.00	71.18	
27	0.15	60.85	
28	1.00	71.18	
29	0.15	60.85	
30	1.00	71.18	
31	0.15	60.85	
32	1.00	71.18	
33	0.15	60.85	
34	1.00	71.18	
35	0.15	60.85	
36	1.00	71.18	
37	0.15	60.85	
38	1.00	71.18	
39	0.15	60.85	
40	1.00	71.18	
41	0.15	60.85	

5-Cavity Filter For λ-Connect

Layer Number	Aluminum Fraction	Thickness (nm)	
1	0.15	58.48	
2	1.00	68.74	
3	0.15	58.48	
4	1.00	68.74	
5	0.15	58.48	
6	1.00	68.74	
7	0.15	58.48	
8	1.00	68.74	
9	0.15	116.96	half-wave cavity
10	1.00	68.74	
11	0.15	58.48	
12	1.00	68.74	
13	0.15	58.48	
14	1.00	68.74	
15	0.15	58.48	
16	1.00	68.74	
17	0.15	58.48	
18	1.00	68.74	
19	0.15	58.48	
20	1.00	68.74	
21	0.15	58.48	
22	1.00	68.74	
23	0.15	58.48	
24	1.00	68.74	
25	0.15	58.48	
26	1.00	68.74	
27	0.15	58.48	
28	1.00	68.74	
29	0.15	58.48	
30	1.00	68.74	
31	0.15	58.48	
32	1.00	68.74	
33	0.15	58.48	
34	1.00	68.74	
35	0.15	58.48	
36	1.00	68.74	
37	0.15	58.48	
38	1.00	68.74	
39	0.15	233.92	Full wave cavity
40	1.00	68.74	
41	0.15	58.48	
42	1.00	68.74	

43	0.15	58.48	
44	1.00	68.74	
45	0.15	58.48	
46	1.00	68.74	
47	0.15	58.48	
48	1.00	68.74	
49	0.15	58.48	
50	1.00	68.74	
51	0.15	58.48	
52	1.00	68.74	
53	0.15	58.48	
54	1.00	68.74	
55	0.15	58.48	
56	1.00	68.74	
57	0.15	58.48	
58	1.00	68.74	
59	0.15	58.48	
60	1.00	68.74	
61	0.15	58.48	
62	1.00	68.74	
63	0.15	58.48	
64	1.00	68.74	
65	0.15	58.48	
66	1.00	68.74	
67	0.15	58.48	
68	1.00	68.74	
69	0.15	58.48	
70	1.00	68.74	
71	0.15	58.48	
72	1.00	68.74	
73	0.15	350.88	1.5 wave cavity
74	1.00	68.74	
75	0.15	58.48	
76	1.00	68.74	
77	0.15	58.48	
78	1.00	68.74	
79	0.15	58.48	
80	1.00	68.74	
81	0.15	58.48	
82	1.00	68.74	
83	0.15	58.48	
84	1.00	68.74	
85	0.15	58.48	
86	1.00	68.74	

87	0.15	58.48	
88	1.00	68.74	
89	0.15	58.48	
90	1.00	68.74	
91	0.15	58.48	
92	1.00	68.74	
93	0.15	58.48	
94	1.00	68.74	
95	0.15	58.48	
96	1.00	68.74	
97	0.15	58.48	
98	1.00	68.74	
99	0.15	58.48	
100	1.00	68.74	
101	0.15	58.48	
102	1.00	68.74	
103	0.15	58.48	
104	1.00	68.74	
105	0.15	58.48	
106	1.00	68.74	
107	0.15	233.92	full wave cavity
108	1.00	68.74	
109	0.15	58.48	
110	1.00	68.74	
111	0.15	58.48	
112	1.00	68.74	
113	0.15	58.48	
114	1.00	68.74	
115	0.15	58.48	
116	1.00	68.74	
117	0.15	58.48	
118	1.00	68.74	
119	0.15	58.48	
120	1.00	68.74	
121	0.15	58.48	
122	1.00	68.74	
123	0.15	58.48	
124	1.00	68.74	
125	0.15	58.48	
126	1.00	68.74	
127	0.15	58.48	
128	1.00	68.74	
129	0.15	58.48	
130	1.00	68.74	

131	0.15	58.48	
132	1.00	68.74	
133	0.15	58.48	
134	1.00	68.74	
135	0.15	58.48	
136	1.00	68.74	
137	0.15	116.96	half-wave cavity
138	1.00	68.74	
139	0.15	58.48	
140	1.00	68.74	
141	0.15	58.48	
142	1.00	68.74	
143	0.15	58.48	
144	1.00	68.74	
145	0.15	58.48	

Chapter 4 – Lateral Oxidation

Sample 'A' 3-cavity bandpass filter with large DBR stop-bands

All Layers are AlGaAs with the indicated Al percent and thickness (except Layer 1)

Layer Number	Thickness (nm)	Aluminum Percent	
1	400.00	InGaP	Etch Stop for substrate removal.
2	58.03	15	
3	136.30	100	
4	58.03	15	
5	136.30	100	
6	116.06	15	
7	136.30	100	
8	58.03	15	
9	136.30	100	
10	58.03	15	
11	136.30	100	
12	58.03	15	
13	136.30	100	
14	58.03	15	
15	136.30	100	
16	116.06	15	
17	136.30	100	
18	58.03	15	
19	136.30	100	
20	58.03	15	
21	136.30	100	
22	58.03	15	
23	136.30	100	
24	58.03	15	
25	136.30	100	
26	116.06	15	
27	136.30	100	
28	58.03	15	
29	136.30	100	
30	58.03	15	

Chapter 4 – Lateral Oxidation

Sample 'B' 2-cavity passband shifting filter, no oxidation barriers

note: all layers (except #1) are AlGaAs with the indicated Aluminum Percentages

Epi Layer Breakdown for 2- Cavity Passband Shifting Filter - no barriers	**Aluminum Percentage**	**Thickness (nm)**
Layer number 1 : InGaP Etch Stop	---	400.00
Layer number 2 :	15	59.82
Layer number 3 :	85	68.33
Layer number 4 :	15	59.82
Layer number 5 : Tuning Layer	100	21.84
Layer number 6 :	85	68.33
Layer number 7 :	15	59.82
Layer number 8 :	85	68.33
Layer number 9 : Tuning Layer	100	21.84
Layer number 10 :	15	59.82
Layer number 11 :	85	68.33
Layer number 12 :	15	59.82
Layer number 13 : Tuning Layer	100	21.84
Layer number 14 :	85	68.33
Layer number 15 :	15	59.82
Layer number 16 :	85	68.33
Layer number 17 : Tuning Layer	100	21.84
Layer number 18 :	15	119.64
Layer number 19 : Tuning Layer	100	21.84
Layer number 20 :	85	68.33
Layer number 21 :	15	59.82
Layer number 22 :	85	68.33
Layer number 23 : Tuning Layer	100	21.84
Layer number 24 :	15	59.82
Layer number 25 :	85	68.33
Layer number 26 :	15	59.82
Layer number 27 : Tuning Layer	100	21.84
Layer number 28 :	85	68.33
Layer number 29 :	15	59.82
Layer number 30 :	85	68.33
Layer number 31 : Tuning Layer	100	21.84
Layer number 32 :	15	59.82
Layer number 33 :	85	68.33
Layer number 34 :	15	59.82
Layer number 35 : Tuning Layer	100	21.84
Layer number 36 :	85	68.33
Layer number 37 :	15	59.82
Layer number 38 :	85	68.33
Layer number 39 : Tuning Layer	100	21.84

Layer number 40 :	15	59.82
Layer number 41 :	85	68.33
Layer number 42 :	15	59.82
Layer number 43 : Tuning Layer	100	21.84
Layer number 44 :	85	68.33
Layer number 45 :	15	59.82
Layer number 46 :	85	68.33
Layer number 47 : Tuning Layer	100	21.84
Layer number 48 :	15	59.82
Layer number 49 :	85	68.33
Layer number 50 :	15	59.82
Layer number 51 : Tuning Layer	100	21.84
Layer number 52 :	85	68.33
Layer number 53 :	15	59.82
Layer number 54 :	85	68.33
Layer number 55 : Tuning Layer	100	21.84
Layer number 56 :	15	59.82
Layer number 57 :	85	68.33
Layer number 58 :	15	59.82
Layer number 59 : Tuning Layer	100	21.84
Layer number 60 :	85	68.33
Layer number 61 :	15	59.82
Layer number 62 :	85	68.33
Layer number 63 : Tuning Layer	100	21.84
Layer number 64 :	15	119.64
Layer number 65 : Tuning Layer	100	21.84
Layer number 66 :	85	68.33
Layer number 67 :	15	59.82
Layer number 68 :	85	68.33
Layer number 69 : Tuning Layer	100	21.84
Layer number 70 :	15	59.82
Layer number 71 :	85	68.33
Layer number 72 :	15	59.82
Layer number 73 : Tuning Layer	100	21.84
Layer number 74 :	85	68.33
Layer number 75 :	15	59.82
Layer number 76 :	85	68.33
Layer number 77 : Tuning Layer	100	21.84
Layer number 78 :	15	59.82
Layer number 79 :	85	68.33
Layer number 80 :	15	59.82

Chapter 4 – Lateral Oxidation

Sample 'C' 2-cavity passband shifting filter, 5nm oxidation barriers

note: all layers (except #1) are AlGaAs with the indicated Aluminum Percentages

Epi Layer Breakdown for 2-Cavity Passband Shifting Filter 5-nm Barriers		Aluminum Percentage	Thickness (nm)
Layer number	1	InGaP	400.00
Layer number	2	15	59.82
Layer number	3	85	68.33
Layer number	4	15	59.82
Layer number	5	100	21.84
AlGaAs barrier - Layer number	6	15	5.00
Layer number	7	85	68.33
Layer number	8	15	59.82
Layer number	9	85	68.33
AlGaAs barrier Layer number	10	15	5.00
Layer number	11	100	21.84
Layer number	12	15	59.82
Layer number	13	85	68.33
Layer number	14	15	59.82
Layer number	15	100	21.84
AlGaAs barrier - Layer number	16	15	5.00
Layer number	17	85	68.33
Layer number	18	15	59.82
Layer number	19	85	68.33
AlGaAs barrier - Layer number	20	15	5.00
Layer number	21	100	21.84
Layer number	22	15	119.64
Layer number	23	100	21.84
AlGaAs barrier - Layer number	24	15	5.00
Layer number	25	85	68.33
Layer number	26	15	59.82
Layer number	27	85	68.33
AlGaAs barrier - Layer number	28	15	5.00
Layer number	29	100	21.84
Layer number	30	15	59.82
Layer number	31	85	68.33
Layer number	32	15	59.82
Layer number	33	100	21.84
AlGaAs barrier - Layer number	34	15	5.00
Layer number	35	85	68.33
Layer number	36	15	59.82
Layer number	37	85	68.33
AlGaAs barrier - Layer number	38	15	5.00
Layer number	39	100	21.84

	Layer number	40	15	59.82
	Layer number	41	85	68.33
	Layer number	42	15	59.82
	Layer number	43	100	21.84
AlGaAs barrier -	Layer number	44	15	5.00
	Layer number	45	85	68.33
	Layer number	46	15	59.82
	Layer number	47	85	68.33
AlGaAs barrier -	Layer number	48	15	5.00
	Layer number	49	100	21.84
	Layer number	50	15	59.82
	Layer number	51	85	68.33
	Layer number	52	15	59.82
	Layer number	53	100	21.84
AlGaAs barrier -	Layer number	54	15	5.00
	Layer number	55	85	68.33
	Layer number	56	15	59.82
	Layer number	57	85	68.33
AlGaAs barrier -	Layer number	58	15	5.00
	Layer number	59	100	21.84
	Layer number	60	15	59.82
	Layer number	61	85	68.33
	Layer number	62	15	59.82
	Layer number	63	100	21.84
AlGaAs barrier -	Layer number	64	15	5.00
	Layer number	65	85	68.33
	Layer number	66	15	59.82
	Layer number	67	85	68.33
AlGaAs barrier -	Layer number	68	15	5.00
	Layer number	69	100	21.84
	Layer number	70	15	59.82
	Layer number	71	85	68.33
	Layer number	72	15	59.82
	Layer number	73	100	21.84
AlGaAs barrier -	Layer number	74	15	5.00
	Layer number	75	85	68.33
	Layer number	76	15	59.82
	Layer number	77	85	68.33
AlGaAs barrier -	Layer number	78	0	15.00
	Layer number	79	100	21.84
	Layer number	80	15	119.64
	Layer number	81	100	21.84
AlGaAs barrier -	Layer number	82	0	15.00
	Layer number	83	85	68.33

	Layer number	84	15	59.82
	Layer number	85	85	68.33
AlGaAs barrier -	Layer number	86	15	5.00
	Layer number	87	100	21.84
	Layer number	88	15	59.82
	Layer number	89	85	68.33
	Layer number	90	15	59.82
	Layer number	91	100	21.84
AlGaAs barrier -	Layer number	92	15	5.00
	Layer number	93	85	68.33
	Layer number	94	15	59.82
	Layer number	95	85	68.33
AlGaAs barrier -	Layer number	96	15	5.00
	Layer number	97	100	21.84
	Layer number	98	15	59.82
	Layer number	99	85	68.33
	Layer number	100	15	59.82

Chapter 4 – Lateral Oxidation

Sample 'D' 2-cavity passband shifting filter, 15nm oxidation barriers

note: all layers (except #1) are AlGaAs with the indicated Aluminum Percentages

Epi Layer Breakdown for 2-cavity Passband Shifting Filter 15-nm Barriers			Aluminum Percentage	Thickness (nm)
	Layer number	1	InGaP	400.00
	Layer number	2	15	59.82
	Layer number	3	85	68.33
	Layer number	4	15	59.82
	Layer number	5	100	21.84
AlGaAs barrier-	Layer number	6	15	15.00
	Layer number	7	85	68.33
	Layer number	8	15	59.82
	Layer number	9	85	68.33
AlGaAs barrier-	Layer number	10	15	15.00
	Layer number	11	100	21.84
	Layer number	12	15	59.82
	Layer number	13	85	68.33
	Layer number	14	15	59.82
	Layer number	15	100	21.84
AlGaAs barrier-	Layer number	16	15	15.00
	Layer number	17	85	68.33
	Layer number	18	15	59.82
	Layer number	19	85	68.33
AlGaAs barrier-	Layer number	20	15	15.00
	Layer number	21	100	21.84
	Layer number	22	15	119.64
	Layer number	23	100	21.84
AlGaAs barrier-	Layer number	24	15	15.00
	Layer number	25	85	68.33
	Layer number	26	15	59.82
	Layer number	27	85	68.33
AlGaAs barrier-	Layer number	28	15	15.00
	Layer number	29	100	21.84
	Layer number	30	15	59.82
	Layer number	31	85	68.33
	Layer number	32	15	59.82
	Layer number	33	100	21.84
AlGaAs barrier-	Layer number	34	15	15.00
	Layer number	35	85	68.33
	Layer number	36	15	59.82
	Layer number	37	85	68.33
AlGaAs barrier-	Layer number	38	15	15.00
	Layer number	39	100	21.84

	Layer number	40	15	59.82
	Layer number	41	85	68.33
	Layer number	42	15	59.82
	Layer number	43	100	21.84
AlGaAs barrier-	Layer number	44	15	15.00
	Layer number	45	85	68.33
	Layer number	46	15	59.82
	Layer number	47	85	68.33
AlGaAs barrier-	Layer number	48	15	15.00
	Layer number	49	100	21.84
	Layer number	50	15	59.82
	Layer number	51	85	68.33
	Layer number	52	15	59.82
	Layer number	53	100	21.84
AlGaAs barrier-	Layer number	54	15	15.00
	Layer number	55	85	68.33
	Layer number	56	15	59.82
	Layer number	57	85	68.33
AlGaAs barrier-	Layer number	58	15	15.00
	Layer number	59	100	21.84
	Layer number	60	15	59.82
	Layer number	61	85	68.33
	Layer number	62	15	59.82
	Layer number	63	100	21.84
AlGaAs barrier-	Layer number	64	15	15.00
	Layer number	65	85	68.33
	Layer number	66	15	59.82
	Layer number	67	85	68.33
AlGaAs barrier-	Layer number	68	15	15.00
	Layer number	69	100	21.84
	Layer number	70	15	59.82
	Layer number	71	85	68.33
	Layer number	72	15	59.82
	Layer number	73	100	21.84
AlGaAs barrier-	Layer number	74	15	15.00
	Layer number	75	85	68.33
	Layer number	76	15	59.82
	Layer number	77	85	68.33
AlGaAs barrier-	Layer number	78	0	15.00
	Layer number	79	100	21.84
	Layer number	80	15	119.64
	Layer number	81	100	21.84
AlGaAs barrier-	Layer number	82	0	15.00
	Layer number	83	85	68.33

	Layer number	84	15	59.82
	Layer number	85	85	68.33
AlGaAs barrier-	Layer number	86	15	15.00
	Layer number	87	100	21.84
	Layer number	88	15	59.82
	Layer number	89	85	68.33
	Layer number	90	15	59.82
	Layer number	91	100	21.84
AlGaAs barrier-	Layer number	92	15	15.00
	Layer number	93	85	68.33
	Layer number	94	15	59.82
	Layer number	95	85	68.33
AlGaAs barrier-	Layer number	96	15	15.00
	Layer number	97	100	21.84
	Layer number	98	15	59.82
	Layer number	99	85	68.33
	Layer number	100	15	59.82

Chapter 4 – Lateral Oxidation

Sample 'E' Barrier Thickness Study

note: all layers are AlGaAs with the indicated Aluminum Percentages

Layer Number	Thickness (nm)	Aluminum Al%	
1	50	15	
2	23.23	100	T
3	20	15	B
4	68.33	85	U
5	59.82	15	H
6	68.33	85	U
7	20	15	B
8	23.23	100	T
9	59.82	15	H
10	68.33	85	P
11	59.82	15	H
12	23.23	100	(repeats)
13	20	15	*
14	68.33	85	*
15	59.82	15	*
16	68.33	85	
17	20	15	
18	23.23	100	
19	59.82	15	
20	68.33	85	
21	59.82	15	
22	23.23	100	
23	25	15	
24	68.33	85	
25	59.82	15	
26	68.33	85	
27	25	15	
28	23.23	100	
29	59.82	15	
30	68.33	85	
31	59.82	15	
32	23.23	100	
33	25	15	
34	68.33	85	
35	59.82	15	
36	68.33	85	
37	25	15	
38	23.23	100	
39	59.82	15	

40	68.33	85
41	59.82	15
42	23.23	100
43	30	15
44	68.33	85
45	59.82	15
46	68.33	85
47	30	15
48	23.23	100
49	59.82	15
50	68.33	85
51	59.82	15
52	23.23	100
53	30	15
54	68.33	85
55	59.82	15
56	68.33	85
57	30	15
58	23.23	100
59	59.82	15
60	68.33	85
61	59.82	15
62	23.23	100
63	35	15
64	68.33	85
65	59.82	15
66	68.33	85
67	35	15
68	23.23	100
69	59.82	15
70	68.33	85
71	59.82	15
72	23.23	100
73	35	15
74	68.33	85
75	59.82	15
76	68.33	85
77	35	15
78	23.23	100
79	59.82	15
80	68.33	85
81	59.82	15
82	23.23	100
83	40	15

84	68.33	85
85	59.82	15
86	68.33	85
87	40	15
88	23.23	100
89	59.82	15
90	68.33	85
91	59.82	15
92	23.23	100
93	40	15
94	68.33	85
95	59.82	15
96	68.33	85
97	40	15
98	23.23	100
99	59.82	15
100	68.33	85
101	59.82	15
102	23.23	100
103	75	15
104	68.33	85
105	59.82	15
106	68.33	85
107	75	15
108	23.23	100
109	59.82	15
110	68.33	85
111	59.82	15
112	23.23	100
113	75	15
114	68.33	85
115	59.82	15
116	68.33	85
117	75	15
118	23.23	100
119	59.82	15
120	68.33	85
121	59.82	15

Chapter 4 – Lateral Oxidation

Sample 'F' Barrier Composition Study

note: all layers are AlGaAs with the indicated Aluminum Percentages

Layer Number	Thickness (nm)	Aluminum Al%	
1	50.00	15	
2	23.23	100	T
3	15	0	B
4	68.33	85	U
5	59.82	15	H
6	68.33	85	U
7	15	0	B
8	23.23	100	T
9	59.82	15	H
10	68.33	85	P
11	59.82	15	H
12	23.23	100	(repeats)
13	15	0	*
14	68.33	85	*
15	59.82	15	*
16	68.33	85	
17	15	0	
18	23.23	100	
19	59.82	15	
20	68.33	85	
21	59.82	15	
22	23.23	100	
23	15	5	
24	68.33	85	
25	59.82	15	
26	68.33	85	
27	15	5	
28	23.23	100	
29	59.82	15	
30	68.33	85	
31	59.82	15	
32	23.23	100	
33	15	5	
34	68.33	85	
35	59.82	15	
36	68.33	85	
37	15	5	
38	23.23	100	
39	59.82	15	

40	68.33	85
41	59.82	15
42	23.23	100
43	15	10
44	68.33	85
45	59.82	15
46	68.33	85
47	15	10
48	23.23	100
49	59.82	15
50	68.33	85
51	59.82	15
52	23.23	100
53	15	10
54	68.33	85
55	59.82	15
56	68.33	85
57	15	10
58	23.23	100
59	59.82	15
60	68.33	85
61	59.82	15
62	23.23	100
63	15	20
64	68.33	85
65	59.82	15
66	68.33	85
67	15	20
68	23.23	100
69	59.82	15
70	68.33	85
71	59.82	15
72	23.23	100
73	15	20
74	68.33	85
75	59.82	15
76	68.33	85
77	15	20
78	23.23	100
79	59.82	15
80	68.33	85
81	59.82	15
82	23.23	100
83	15	30

84	68.33	85
85	59.82	15
86	68.33	85
87	15	30
88	23.23	100
89	59.82	15
90	68.33	85
91	59.82	15
92	23.23	100
93	15	30
94	68.33	85
95	59.82	15
96	68.33	85
97	15	30
98	23.23	100
99	59.82	15
100	68.33	85
101	59.82	15
102	23.23	100
103	15	50
104	68.33	85
105	59.82	15
106	68.33	85
107	15	50
108	23.23	100
109	59.82	15
110	68.33	85
111	59.82	15
112	23.23	100
113	15	50
114	68.33	85
115	59.82	15
116	68.33	85
117	15	50
118	23.23	100
119	59.82	15
120	68.33	85
121	59.82	15
122	23.23	100
123	15	75
124	68.33	85
125	59.82	15
126	68.33	85
127	15	75

128	23.23	100
129	59.82	15
130	68.33	85
131	59.82	15
132	23.23	100
133	15	75
134	68.33	85
135	59.82	15
136	68.33	85
137	15	75
138	23.23	100
139	59.82	15
140	68.33	85
141	59.82	15