Circuit Board Prototyping System

Results of Technology Investigation

CS400 – Senior Design

Milwaukee School of Engineering

October 10, 1997

Client: MSOE Technical Support Center

Project Engineers: Karin A. Fritz
James R. Grant
Keyur H. Khambholja
Jeff Krueger

Project Advisor: Dr. Steven Barnicki

Table of Contents

Introduction

The goal of our Senior Design project is to produce a fully functional and easy-to-use circuit board prototyping system based on an IBC BoardMaker 912. We chose this project because it involves both software and hardware design components. Also, when we are successful, Milwaukee School of Engineering (MSOE) students and staff will benefit. They will be able to produce prototype circuit boards for various projects. In this report, we will explain the numerous technologies that we will use to complete the project.

IBC BoardMaker

The IBC BoardMaker is a machine tool designed specifically for the production of prototype circuit boards. Driven by an IBM PC, the IBC BoardMaker automatically engraves single-sided and double-sided boards of any size up to 9" x 12". It can also be controlled manually through the PC.

The IBC BoardMaker produces circuit boards by engraving narrow channels on standard copper-clad[1] boards to isolate the conductive material. Commands are issued from the PC using the IBC (DOS) software and sent to the IBC BoardMaker through the PC's parallel port. The IBC BoardMaker can only handle one command at a time, and provides no feedback to the PC that it has completed an operation. For example, when the machine is told to move 1", the IBC software will wait a predetermined amount of time while that operation is carried out. The IBC software will then send another command without knowing if the previous operation was completed. The IBC software currently will only run on an Intel 286 due to its timing limitations. As you can see this configuration is very limited.

Project Requirements

There are two basic requirements: getting the IBC BoardMaker working with a 486 or higher PC through a buffering system, and to provide user-friendly software. The Windows 95/NT platform should be targeted – but Windows 95 has a priority over Windows NT. Initially, only single-sided boards need to be implemented.

Project Divisions

The project is split into three divisions: Machine Division, Hardware Division, and Software Division. These topics and their related issues will be discussed independently.

[1] Standard copper-clad boards contain epoxy, glass, and resin forming a solid structure to which copper is applied to one or both sides of the structure forming either single or double-sided boards.

Machine Division

Milling/Drilling

Figure 1 depicts the milling/drilling head of the IBC BoardMaker. This head is controlled by a solenoid that, when activated, pushes the entire head down onto a copper-clad board. The head will only go as far as the depth-limiter will allow. The knurled nut allows for the depth limiter to be moved up (clockwise) or down (counterclockwise).

By adjusting the depth limiter the width of the engraved channel change. Since the tip of the mill bit is tapered, milling more deeply into the board produces a wider engraved channel. By raising the depth-limiter, the engraved channel becomes narrower. Figure 2 shows the relationship between the mill bit tip and the width of the engraved channel.

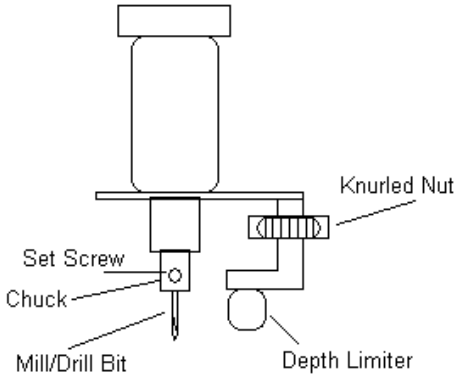


Figure 1. Milling/Drilling Head Configuration.

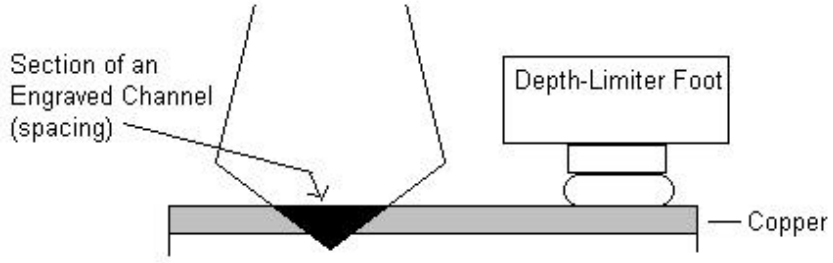


Figure 2. Enlarged view of mill bit showing depth of engraving channel.

From this discussion one can see that the depth to which the *mill* bit is inserted into the chuck is critical in the engraving process, for it controls how deep each channel becomes. If the depth is too shallow then all copper may not be removed, causing a conductive path greater than desired. Also, the depth to which a *drill* bit is set changes the drilling process. A shallow setting will not allow a hole to be cut properly, while a deep setting will allow the drill to penetrate the underlay material, which is discussed in the following section.

Initial testing showed that even if these bits are placed as far into the chuck as possible the depth limiter must be changed for each. By specifying the depth to which each bit can be set, the depth-limiter can be set permanently. All bits will then be at the same depth each time if there is a way of limiting their depth in the chuck.

Three possible solutions could be used to ensure that bits are inserted to the correct depth each time: painting a ring on each bit showing proper depth, putting a rubber O-ring on each bit, or soldering a washer on each bit.

Paint can be rubbed off or is misleading as to exact position, where the rubber O-ring will prevent the bit from going any further into the chuck.

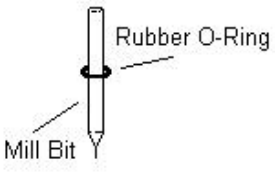


Figure 3. Mill bit with depth limiting O-ring.

Figure 3 depicts the mill bit with an O-ring attached. This will allow the bit to be placed into the chuck at the same depth every time, stopping any further adjustment of the depth-limiter, and ensuring exact placement of all bits. However, machine vibrations or handling of the bit may move the O-ring from its original position.

A third option is to solder a washer in place of the O-ring. This washer would be permanently affixed, eliminating any variations in bit placement.

Underlay System

The underlay, seen in Figure 4, is the component on which the copper-clad board rests during engraving. During testing of the engraving process it was noticed that the current underlay was thin and flimsy causing the height between the board and the mill/drill head to change. This, as the previous discussion showed, changes the depth of the engraved line. The current underlay is made of a material known as phenolic consisting of fiber and glass, approximately 1/16" thick. Phenolic comes in various makes and thicknesses some of which decrease flexibility.

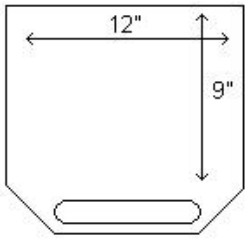


Figure 4. IBC BoardMaker Underlay showing opening to vise for board securing.

Use of a thicker material, one in which flexibility is mostly eliminated, would make a better underlay. Increasing the thickness too much will not work, due to the clearance requirements of the mill/drill head.

A number of different materials were considered in the replacement of the current underlay. These include wood, metal, and a paper-glass phenolic.

The first potential underlay replacement was wood. When the drill produces a hole through the copper-clad board it will also partially enter the underlay material. A drill bit would not be damaged or worn by such an action. The immediate problem with a wood underlay is that it is very difficult to find wood in the 1/16 ” to 1/8 ” range. Wood this thin becomes very brittle, and could break easily.

Metal is another possibility. Metal is malleable, it can be drawn out in thin, stiff sheets. But if the drill bit is to penetrate the material, metal will not only wear the bit down, but it may damage it.

McMaster Carr [Chicago] sells phenolic in a paper-glass form that is less flexible than the fiber-glass (not fiberglass) form. It sells in 1/16 ” and 1/8 ” thicknesses in sheets of 24 ” x 24 ”. Two 1/16 ” and two 1/8 ” sheets were purchased and cut to fit. The

1/16" paper-glass phenolic is more rigid than the 1/16" fiber-glass phenolic. The 1/8" is even less flimsy and was tested as the new underlay. The 1/8" paper-glass phenolic performed remarkably. We will provide the Technical Support Center an additional underlay (excluding the one on the IBC BoardMaker) for future use.

Board Securing System

Securing the board is the most crucial part of the milling/drilling process. If the board slips at all while milling or drilling, the entire board is rendered useless.

The opening in the underlay is used to allow a vise below the underlay to hold the board being milled in place. Current implementation requires the drilling of two holes in the copper board, through which two steel set pins are inserted. The board is then laid on the underlay and the vise is tightened around the set pins. This system holds the board in place and does not allow for shifting of the board during operation. The problem that arises with this type of securing is the board can be placed in different positions anywhere along the vise length (it can be put anywhere without consistency). Therefore the entire milling process must be completed in one sitting, which may or may not be possible depending on how large the project is. It also does not allow for double-sided boards to be created. If the board is released from the vise and flipped over, a requirement of double-sided processing, it will be very unlikely the board will be in the exact same position.

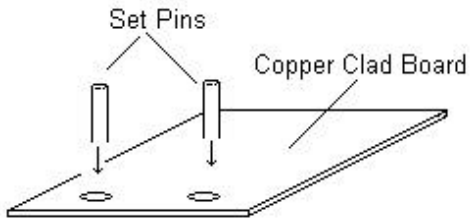


Figure 5. Set Pin holding strategy

A board securing system will be developed that allows for the board to be removed and put back in the exact same place every time. It should also allow for the board to be turned over while maintaining its positioning for double-sided processing (a future consideration). A securing system requires that the securing unit would not move while in the vise, would allow for different size boards, and provide a quick attachment of copper-clad boards to the system.

The securing system described above requires two set pins that would not allow for the system to move while in the vise. Tapped holes on the top of the unit would allow thumbscrews to attach the board into the securing system. Figure 6 depicts the desired securing system that could be machined from a piece of steel.

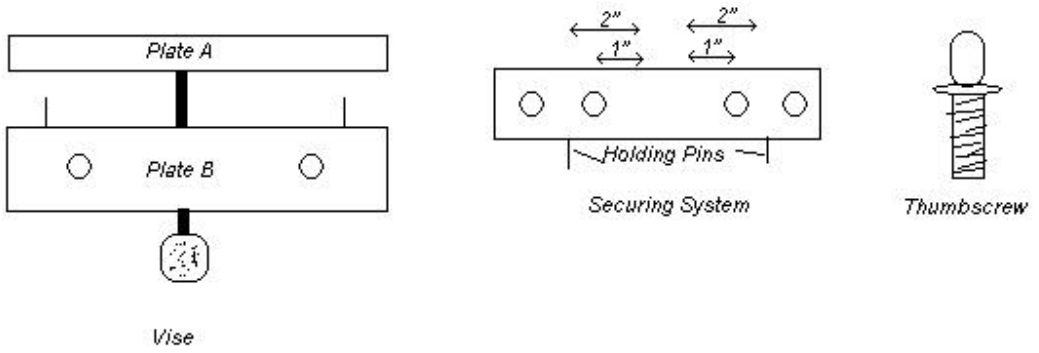


Figure 6. Securing System.

The securing system of Figure 6 is placed between Plates A and B of the vise. Holding pins on the securing system slide into holes drilled in Plate B (these holes are not shown) and the vise is tightened. Tapped holes on top of the securing system allow for thumbscrews to be inserted, which hold the copper-clad board in place. Four holes are used - one set of holes is exactly one inch from the center point on each side and could be used for small to medium sized copper-clad boards. While another set is two inches from the center, used for medium to large boards.

If holes are required to be drilled into the copper-clad board for securing purposes then a method must be developed through which holes can easily be drilled and distances properly maintained. A template can be machined from the same material the securing system was created. This would allow the user to place the copper-clad board directly onto the template to drill securing holes. Figure 7 shows this template. By finding the exact middle of the copper-clad board, it can be aligned with the middle mark of the template. Once this is done drilling holes becomes a simple task. This process will allow a board to be flipped over and maintain exact locations.

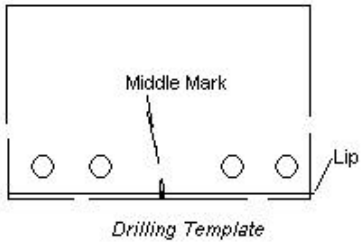


Figure 7. Drilling Template

Cleaning System

Tests performed during the engraving of simple boards showed that the debris produced is sufficient enough to require a cleaning system.

Located on the back of the milling/drilling head is a vacuum cleaner pipe. The vacuum clears the work area of debris created by machining operations. A clean work area is essential for accurate milling. Any particle of debris underneath the depth limiter might raise the mill/drill head and decrease the width of the line being milled. A large enough particle might stop the milling from cutting all the way through the copper.

The cleaning system does not require any special filtration of particles and requires only a standard vacuum cleaner hose. Any particles that may be produced will be small enough such that they will easily be picked up. The amount of waste produced is not great enough to require a large vacuuming unit. A 1 HP, 1-gallon vacuum is

sufficient for the cleaning of all projects. This type of vacuum is readily available at all hardware stores.

Hardware Division

Hardware Overview

This division will explain the buffering between the IBC BoardMaker and PC. The buffer will act as a liaison between the IBC BoardMaker and PC, providing the PC with required feedback while receiving and delivering commands for machining. It will also allow the PC to be free for continuous operation. The buffer is based on a microcontroller that can store all received commands into memory (buffering) and interpret and deliver those commands to the IBC BoardMaker. With a microcontroller, the buffer can be programmed to respond to signals received from the IBC BoardMaker and interpret commands issued from the PC.

The Buffer

Although a PLD could be programmed to act as a buffer for this system, it would have no way of responding to signals sent by the IBC BoardMaker. A microcontroller can fulfill both of these roles.

Intel and Motorola both manufacture an assortment of microcontrollers. Since our focus at MSOE has been on the Motorola line of microcontrollers, specifically the HC11E9, we will build the buffering system around that microcontroller. The HC11E9 is readily available from the Technical Support Center and has a robust set of features.

The HC11E9 will receive all IBC BoardMaker directives from the PC storing them in memory. When the data buffer is full, the HC11E9 sends a wait signal to the PC, halting any continued data flow. The HC11E9 will pull each stored command from memory, one at a time, and output it to the IBC BoardMaker for processing. The operations must be timed to determine when the next command should be retrieved from memory and sent for processing. After all commands for that data set have been processed, a clear signal will be sent to the PC instructing it to send more data. Processing will continue in this fashion until the entire design has been completed.

The software that will be written to control the microcontroller will be burned into EEPROM memory located in the HC11E9. If power is ever lost to the HC11E9 the program will remain.

Programming Language

Implementation of the buffering application will be performed through assembly. We chose assembly over Interactive C for the following reasons:

- Easier bitwise manipulation
- Familiarity
- More efficient use of code space
- Speed

Communications

There are two types of communications that will be performed during any operation: PC ⇔ buffer and buffer ⇔ IBC BoardMaker.

Communication between the PC and the HC11E9 is accomplished through serial communications. The PC's RS232 port will send out a logic-1 as -12 volts and a logic-0 as +12 volts. The HC11E9 contains CMOS circuitry and can not handle this type of signaling. The MAXIM MAX233 chip converts the RS232 voltage to TTL/CMOS levels, which is acceptable to the HC11E9. Because the PC and MAX233 are both data terminal equipment (DTE), the receive and transmit signals between the buffer and PC must be interchanged, creating a NULL modem. This will ensure the PC transmitted signals are received by the buffer interface. A basic phone wire consisting of three wires for Tx, Rx, and Gnd is all that is needed for communication between the PC and the HC11E9.

The IBC BoardMaker uses two Superior Electric Translator Motor Drivers to drive two stepper motors: one to move the milling/drilling head in the x direction and one to move the underlay in the y direction. Each Translator Motor Driver accepts pulses on its (PU) 'pin. When the pulse is low the Translator Motor Driver drives the motor in small steps. It accepts TTL logic values output by the HC11E9 so no further interfacing is needed between the two.

Supporting Hardware

The HC11E9 microcontroller requires support for its communication, memory, and serial interfacing tasks. There are two readily available HC11E9 microcontroller systems with all the required memory, communication, and interfacing components built into one unit: Handy Board and the Rug Warrior.

The Handy Board was designed by MIT for experimentation and development of HC11E9 applications. It comes with an HC11E9, 32k of static RAM, serial interface, and a LCD for viewing code output. The Handy Board is priced around $200 (*not assembled*).

The second board that could be used is the Rug Warrior board used by MSOE's Technical Support Center. It contains all the needed features at a much lower price ($80, *not assembled*) than the Handy Board, and is readily available from the Technical Support Center. Because of availability and price, we have chosen to implement the buffer using the Rug Warrior board. It is not reasonable to design our own board when something of high quality and proven reliability already exists.

Software Division

Software Overview

This portion of the project resides on the PC. It includes the user-interface and translating code. The user will be able to input a Gerber file and send it to the HC11E9 (which will send it to the IBC BoardMaker to be milled, drilled, and routed). The user will be prompted for changing bits at the appropriate time, and will know the progress of the operation via a software progress bar.

Our software will replace the software that was shipped with the IBC BoardMaker. This software was difficult to use and would run only on a 286 machine. Our goal is to provide an intuitive and easy–to-use interface to the IBC BoardMaker.

Platform

Although there are many choices for platforms, the client requires support for Windows 95. Since programming for Windows 95 and Windows NT 4.0 are similar, we will attempt to make the application compatible with both. The most significant difference is the registry layout. To account for differences, our code can call the `GetVersionEx` Windows API function. A pointer to an `OSVERSIONINFO` data structure is passed with this call. If the function is successful, the `dwPlatformId` member variable will contain one of the following values:

`VER_PLATFORM_WIN32s`	Win32s on Windows 3.1.
`VER_PLATFORM_WIN32_WINDOWS`	Win32 on Windows 95.
`VER_PLATFORM_WIN32_NT`	Win32 on Windows NT.

From this information, operating system specific code can be executed as appropriate.

Software Tools

Design Tools

The main tool that will be used for design is Rational Rose/C++. This tool allows for documenting object-oriented analysis and design. It also has the ability to generate C++ code for the class definitions from the model. Rational Rose has the ability to reverse-engineer a project, allowing changes in the implementation to be updated in the design model. These features would save us a lot of time and keep our design and implementation closely linked.

We will also use Microsoft Visual C++ 5.0 to create sample screens. These screen shots will be used in our design documentation. These screens could also later be used in the actual user interface.

Development Tools

Our chosen C++ compiler for the Win32 platform is Microsoft Visual C++ 5.0. This is the latest C++ package produced by Microsoft Corporation. It allows for quick development of Windows applications through the use of Microsoft Foundation Classes (MFC), and *Wizards*.

Many of the foundation classes are based on *Windows* (CWnd class) and *Objects* (CObject class). Most GUI items are derived from *Windows* while data the user manipulates is stored in classes derived from *Objects.*

There are a couple *Wizards* worth mentioning that Visual C++ offers that increased productivity. The *AppWizard* creates the basic functionality of a Windows application including support for dialog based, single document interface (SDI), and multiple document interface (MDI) applications. The *ClassWizard* allows you to easily maintain classes and Windows message handlers.

Since we are already familiar with this product from use in school and in the work place, we are pressed for time, and it offers all of the features of competing products, we feel that there is no reason for us to learn a new programming environment.

Input Files

Gerber File Format

The Gerber format is an industry standard format for printed circuit board (PCB) layouts. It originated in the early 1960's and has been revised many times since. It is based on the photoplotter concept. With a photoplotter, a light is shined through an aperture, or shape, to "draw" a line. The shudder can be kept closed when no line is needed. There are two basic variations of this format: RS-274D and RS-274X. These will both be discussed in detail.

Our client requested that we primarily support the Gerber format. This is a wise decision because Gerber is widely used and many PCB layout software packages produce it. Initially, though, we may choose to support only the RS-274D format.

RS-274D

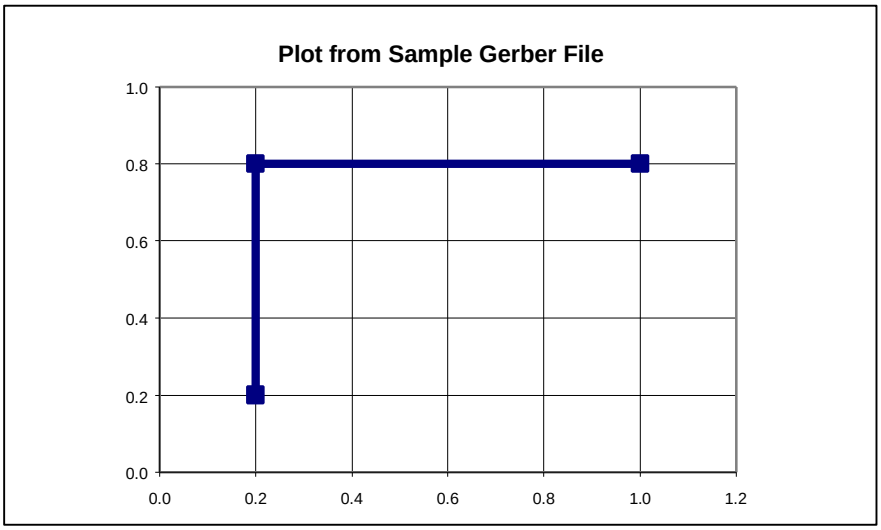
This format is in ASCII and supports four basic pieces of information:

- The numbers representing x and y data - these tell the machine where to go in the horizontal and vertical directions respectively. The information can be in either an absolute or an incremental format. There is no decimal place in the data. Normally, 2 numbers to the left of the decimal and 3 to the right are assumed. Trailing spaces are eliminated.
- Aperture selection - the size and shape to be "drawn".
- Shutter commands - the shutter can be flashed, held open, or held closed.
- End of line character - the Gerber format could be on a single line. The EOL character signifies the end of a command. Commonly, the asterisk character (*) is used.

A typical file with the explanation to the right is as follows (adapted from Layton):

X002Y002D02*	Move to (0.2, 0.2) with shutter closed
D11*	Select aperture 2
D03*	Flash-selected aperture
D10*	Select aperture 1
X002Y008D01*	Move to (0.2, 0.8) with shutter open
D11*	Select aperture 2
D03*	Flash-selected aperture
D10*	Select aperture 1
X01Y008D01*	Move to (1.0, 0.8) with shutter open
D11*	Select aperture 2
D03*	Flash-selected aperture

This should produce the following plot:



RS-274X

The main problem with the RS-274D is who defines what an aperture is. A PCB layout program could define an aperture as one thing, and then a photoplotter could assume it is another. The RS-274X format solves this problem by embedding the aperture definitions in the beginning of the Gerber data.

Implementation

This is the most difficult part on the software side. The Gerber format defines where the circuit board traces actually lie, and the IBC BoardMaker is designed to trace the outline of the wires by milling "unwanted" copper. We need to convert the *traces* of wires into the *outlines* of the wires. This will be a problem since once an outline is milled, no other wire can be connected to it.

The current IBC software contains a Gerber translator that translates Gerber files into BASICAD format. BASICAD is IBC's computer-aided design system that allows the user to do simple editing of a circuit board. Additional IBC software converts BASICAD data into the milling and drilling files [2] that contains the data to drive the

[2] The milling files contain the data that marks the engraved channels that outline the wires. The drilling files contain the data that drill the holes.

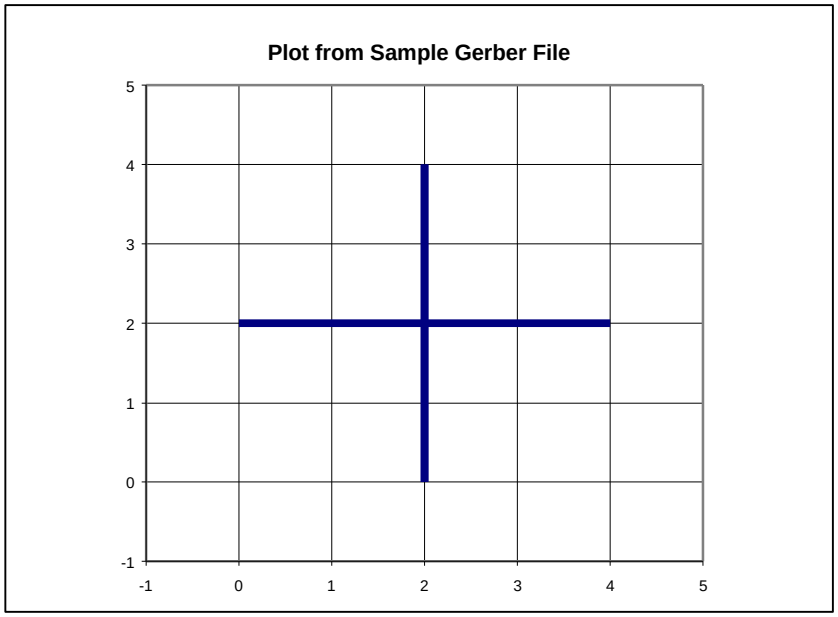
IBC BoardMaker. This type of flow is what we need to imitate. However, we will translate the Gerber file into our own type of format.

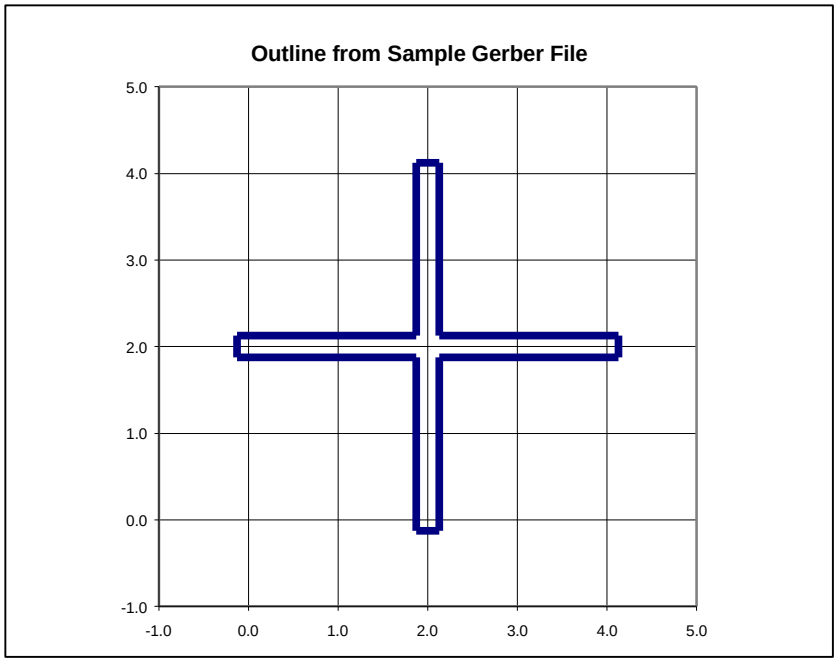
The translation of traces to outlines can be done in to ways: raster graphics and vector graphics. It wouldn t make much sense to use raster graphics – we would have to convert from vector (Gerber file) to raster, process the information, and then convert back to vector (so the machine can understand it). It would also be difficult to use raster graphics because the resolution of the machine is 0.001 inches. The data files would be very large.

Since this translation is a major hurdle, we came up with some initial ideas on how we could accomplish it. The following simple Gerber file will be used to explain how to get an outline (for simplicity, rather large numbers are being used):

X00Y02D02*	Move to (0, 2) with shutter closed
D10*	Select aperture 1 (0.25 ”square)
X004Y002D01*	Move to (4, 2) with shutter open
X02Y00D02*	Move to (2, 0) with shutter closed
X02Y04D01*	Move to (2, 4) with shutter open

This produces the first plot; we want the second plot:





The following is a translation of the Gerber commands to the outline representation:

`X00Y02D02*` Move to (0, 2) with shutter closed
No action taken.

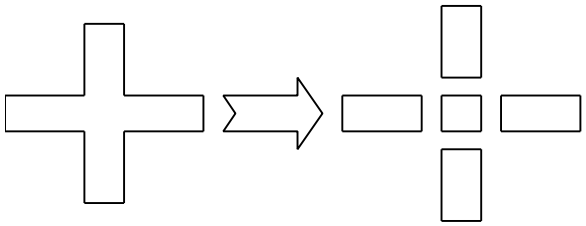
`D10*` Select aperture 1 (0.25" square)
Since drawing width is 0.25", we need a 0.125" overlap on every line.

`X004Y002D01*` Move to (4, 2) with shutter open
Record the lines and that they are visible:

Line Start	**Line End**	**Visible**
(-0.125, 1.875)	(4.125, 1.875)	TRUE
(4.125, 1.875)	(4.125, 2.125)	TRUE
(4.125, 2.125)	(-0.125, 2.125)	TRUE
(-0.125, 2.125)	(-0.125, 1.875)	TRUE

`X02Y00D02*` Move to (2, 0) with shutter closed
No action taken.

`X02Y04D01*` Move to (2, 4) with shutter open
Realize that this line crosses a previous outline and split into five pieces:



Note that these edges of the five pieces are actually touching – they were expanded just to show that they are separate. The left rectangle would be represented as follows:

Line Start	Line End	Visible
(-0.125, 1.875)	(4.125, 1.875)	TRUE
(1.875, 1.875)	(4.125, 2.125)	TRUE
(1.875, 2.125)	(-0.125, 2.125)	FALSE
(-0.125, 2.125)	(-0.125, 1.875)	TRUE

Note that the right edge of the rectangle is not visible and therefore, not etched.

After the complete file is processed, all of the "visible" lines can be compiled into a list, and the "non-visible" lines can be discarded. Then, they can be sorted so the milling machine will start a line near the line it just ended. This sorting will reduce non-etching movement, and therefore reduce the overall time required to make a board.

If time permits, we may research and support other formats such as HPGL, PADS, and VeriBest. Since these are not of high importance to our client, they will be put on the back burner for now.

Communication Issues

Serial

Our chosen method of communications is serial. This is a change from the parallel communications the IBC BoardMaker supports. This change was made because serial communications to an HC11E9 is very easy. Also, we will not have to write our own driver for parallel communications.

To use serial communications in Windows 95/NT, there are a series of Windows API functions. A Senior Design Group from a previous year has already encapsulated these functions into a class. We plan on using this class for our needs.

Intermediate Format of Data

When an input file, such as a Gerber format, is processed, it is converted into a proprietary format. The data will be split into two divisions: milling and drilling.

Feedback

Interspersed in this data will be progress indicators. The hardware will send the progress back to the PC when it has processed all the commands up to that point. This will allow the user application to accurately represent the progress of the machine.

Application Structure

Model

We have discussed the main types of applications that Visual C++ will create. We would like to choose one that the user will be most comfortable with and of course, makes sense.

Dialog

The dialog-based application uses one main window that does not contain a menu or tool bar. The window would contain controls (edit box, list box, buttons, labels, etc.) to allow the user to interact with the application. This would suitable choice but the other models have some features that we think are useful.

SDI

SDI stands for Single Document Interface. This type of application allows a user to open one document to display its contents in a single window (typically like a word processor or a spreadsheet). Menus and toolbars allow the user to interact with the application to perform operations on his/her document.

MDI

MDI stands for Multiple Document Interface. This is exactly like a SDI, however the user can open up multiple documents in several windows.

User Indicators

The most standard form of updating the user is the progress bar. We agree that this is the most efficient way of keeping the user informed of the progress of his/her file. The user will also be prompted to change bits through the use of dialog boxes.

ActiveX

The ActiveX specification allows certain functionality of a program to be encapsulated in an "ActiveX control." These controls can be easily plugged into various containers including Visual C++, Visual Basic, and Internet Explorer.
For this project, the portion of the PC software that communicates with the IBC BoardMaker equipment could be encapsulated into an ActiveX control. It then could easily be reused later, such as in a PCB layout program.

Conclusion

This report contains the core technologies and issues that we must consider during the design phase of our project. This document will be a valuable resource during this time. Not all technology decisions have been made and will be resolved during the design process. This will allow us to choose the best possible solution. We hope that this report has provided the reader with an understanding of our project and the technologies involved.

References

MS Visual C++/C++

Kruglinski, David J. Inside Visual C++. Fourth Edition. Microsoft Press. Redmond, WA 1997.

Meyers, Scott. Effective C++: 50 Specific Ways to Improve Your Programs and Designs. Second Edition. Addison-Wesley. Reading, Mass. 1998.

ActiveX

Li, Sing and Panos Economopoulos. Professional Visual C++ 5 ActiveX/COM Control Programming. Wrox Press Ltd. Olton, Canada 1997.

Gerber Format

Layton, R. W. "Gerber basics: a PCB primer – what is this Gerber format, and why do I need it?" Printed Circuit Fabrication, vol. 16, no. 7, p. 30-32. July 1993.

Nedbal, R. "Is it time to retire the 274 format?" Printed Circuit Design, vol. 12, no. 6, p. 13-14, 16, 56. June 1995.

Seaman, K. L. "RS-274-X Explained" Printed Circuit Design, vol. 13, no. 3, p. 18-22. March 1996.