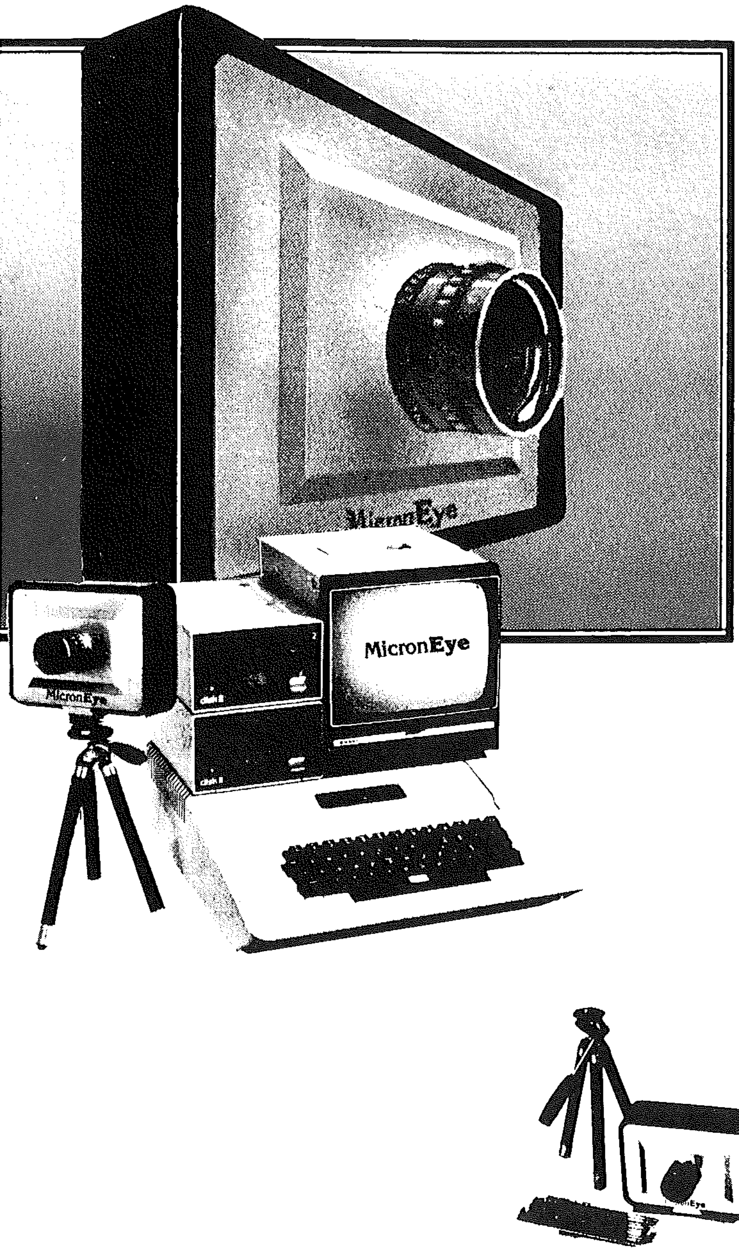
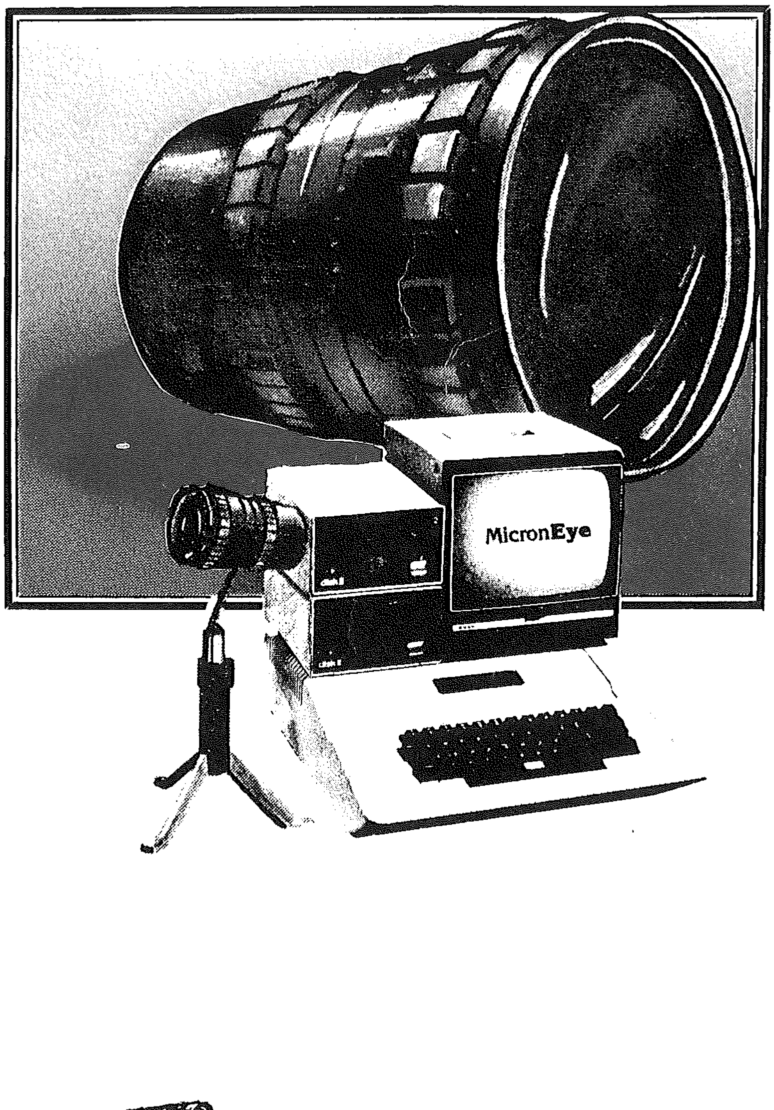
MicronEye

Operator's Manual





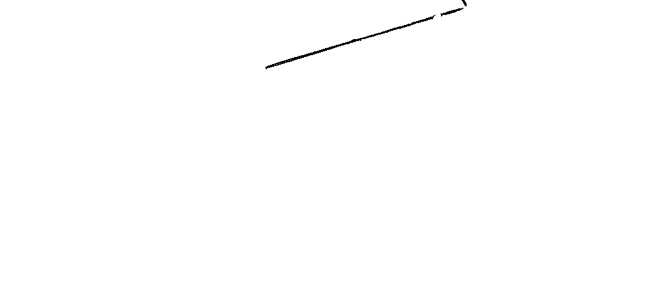


TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The MicronEye is the easiest and least expensive solution to numerous applications requiring a low cost, all digital imaging system. The MicronEye is an electro-optical system suitable for use with your computer as a peripheral. The three necessary dimensions -- optics, hardware and software -- are furnished with the standard package. There are three basic MicronEye systems:

1.1 MICRONEYE BULLET

(Shown in Illustration 1a). This system has the drive electronics located on a 9"x3" card inserted in the computer. The IS32 OpticRAM is located in the 1" diameter cylindrical Bullet case. The Bullet and computer are connected via a standard 16-wire flat ribbon cable. Micron recommends that the cable be less than 5 feet long and furnishes a 4-foot cable with the standard Bullet package.

1.2 MICRONEYE CAMERA

(Shown in Illustration 1b). This system has all of the drive electronics and IS32 OpticRAM on a 6"x3"x1" card which is mounted in a rectangular camera case. It also includes a 3"x3" serial interface card suitable for inserting in the Apple II, IBM PC, Radio Shack TRS-80 Color Computer and Commodore 64. The advantage the Camera has over the Bullet is that the Camera may be located remotely from the computer (up to 50 feet away).

For computers on which custom MicronEye interfaces are not available, an RS-232 compatible version of the MicronEye Camera is available. The RS-232 MicronEye Camera comes equipped with a male DB25P connector. Pin 2 (transmitted data) carries data from the MicronEye to the computer. Pin 3 (received data) sends data from the computer to the MicronEye. Pin 7 is a common ground. Power for the MicronEye (+5V capable of driving a 50 mA load) must be made available to the MicronEye on Pin 11.

1.3 IS32 OPTICRAM

The heart of the MicronEye is the OpticRAM. The OpticRAM was developed and is manufactured by Micron Technology, Inc. The OpticRAM is composed of 65,536 individual image sensing elements called pixels. These pixels are organized into two rectangles (often referred to as arrays) of 128 x 256 pixels each. Each array of cells is separated by an optical "dead" zone of about 25 elements in width.

When an image is focused onto the OpticRAM, a digital representation of the image is "exposed" on the OpticRAM. The MicronEye transmits this image from the OpticRAM to the computer. The software included with the MicronEye takes the transmitted image and displays it on the computer's graphics screen.

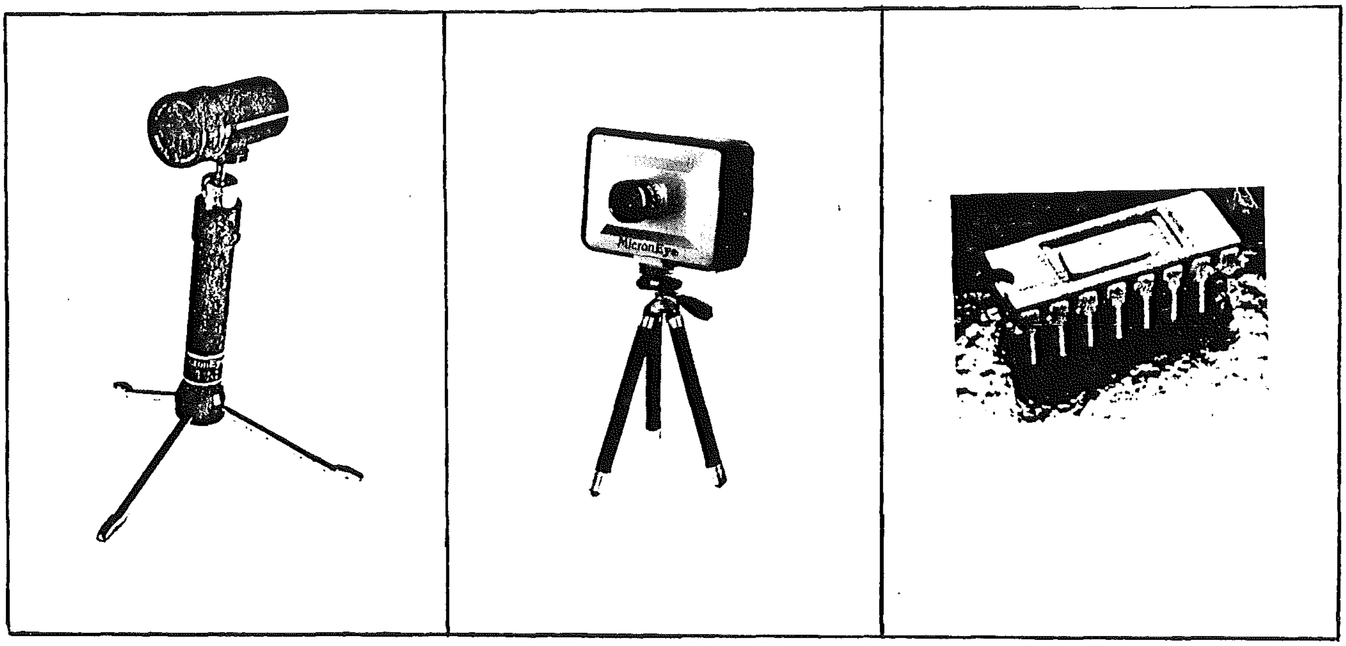
Because the image created by the OpticRAM is digital, the image produced is black and white. The MicronEye may produce shades of gray by multiple scans at different exposure times. MicronEye users with an Epson printer can produce pictures with grey tones with the software provided.

The low cost of the MicronEye is directly attributable to the technological advance represented by Micron's OpticRAM. In terms of cost per pixel, the OpticRAM represents a 1000x reduction in price over earlier generation image-sensing chips such as the CCD. As a result, the MicronEye brings capabilities to your computer which were previously available only to large industrial users.

The electronics in the MicronEye provide an interface between the OpticRAM and computer. It also provides a means by which the MicronEye can receive commands from the computer. Using a crystal to assure accuracy, the MicronEye drive electronics provides all the requisite timing signals and circuitry to execute commands received from the computer. The MicronEye automatically sequences the OpticRAM so that each image sensing element in the OpticRAM is accessed and the appropriate video information is returned to the computer for display or processing.

In addition, the MicronEye's electronic shutter is easily controlled by sending the MicronEye the appropriate commands. A command to the MicronEye to SOAK, "opens" the shutter. After the appropriate period of exposure has elapsed, a command to the MicronEye to REFRESH will "close" the shutter. The software provided automatically performs these functions. Chapter 7 explains the commands available for controlling the MicronEye for users who want to design their own assembly language interfaces. (For most users, the routines provided should be more than adequate.) As you might suspect, the MicronEye's shutter is not a mechanical shutter. The MicronEye controls whether or not the OpticRAM is sensitive to light or not. This feature allows for precise continuous control of the MicronEye's "shutter speed."

If for any reason you must remove the OpticRAM from its socket, caution is imperative. The OpticRAM is susceptible to static and can be damaged by static electricity. Be certain to properly orient the OpticRAM when reinserting it into the socket. For the bullet, the OpticRAM is oriented properly when the red edge of the ribbon cable is on the same side of the camera as the Pin 1 notch on the OpticRAM. For the camera, the OpticRAM is oriented properly when the Pin 1 notch on the OpticRAM is on the same edge as the Pin 1 notch on other IC's in the camera. Removal of the OpticRAM from the bullet may require that the tips of the chip extractor tool be bent out slightly to accomodate the narrowness of the bullet housing.



(a) MicronEye Bullet (b) MicronEye Camera (c) IS32 OpticRAM

Illustration 1

CHAPTER 2

TECHNIQUES FOR OPERATING THE MICRONEYE

2.1 FOCUS AND F-STOP ADJUSTMENTS

The lens supplied with your MicronEye is an F1.6 16mm lens with adjustable f-stop. Please note that the lens has two controls which must be adjusted before the MicronEye will operate successfully: f-stop and focus control. The f-stop controls the amount of light admitted through the lens while the focus control focuses the image on to the surface of the image sensing device (the IS32 OpticRAM).

For normal use, the lowest f-stop setting (1.6) is recommended. Any increase in the f-stop requires a compensating increase in the light source or in the exposure time. Please note there is a "C" setting which completely closes the aperture. A mechanical shutter is not needed since this function is performed electronically by the MicronEye.

The depth of focus (the distance the scene can move in relation to the MicronEye and still be in focus) is increased at higher f-stops. To optimize the result, increase the amount of light and/or the exposure time. A tradeoff of lighting, exposure time, f-stop and scene-to-MicronEye position is necessary to optimize the result.

2.2 CLOSE-UP RING

The lens is designed for viewing objects at a distance of at least 18 inches. Also supplied with the MicronEye is a close-up ring which allows the MicronEye to view objects as near as five inches. From this distance, normal text is clearly readable. The ring can be installed by unscrewing the lens from the MicronEye, inserting the ring over the threads of the lens screw, and screwing the lens back into the MicronEye. The ring acts as a spacer and extends the focal length of the lens. For experimenting with viewing objects as close as two to three inches, an acceptable short-term solution is to slowly unscrew the lens until the object comes into focus (taking care not to unscrew the lens so far that there are insufficient threads to hold the lens onto the MicronEye).

For viewing objects at close range it is recommended that the user purchase a close-up lens. Since the MicronEye utilizes a standard C-mount lens, most camera retailers provide a wide assortment of special purpose lenses directly compatible with the MicronEye.

2.3 LIGHTING CONSIDERATIONS

The MicronEye requires a high contrast scene in order to image the object onto the OpticRAM. Unlike a TV camera which can respond to shades of gray, the OpticRAM is a digital device where each picture element will only respond to a black and white representation of a scene. All portions of the scene lighter than an arbitrary threshold are considered white and all portions of the scene darker than the threshold are considered black. If the exposure time is increased more of the scene falls on the white side of the threshold barrier. As the exposure time is decreased more of the scene falls on the black side of the threshold level.

The threshold level can be affected in one of three ways: (1) changing the exposure time; (2) changing the f-stop on the lens; and (3) changing the light on the scene itself. Doubling the exposure time is the same as opening the f-stop by one stop (changing the f-stop to the next smaller number) or, in other words, doubling the amount of light.

For optimum results from your MicronEye, careful consideration must be paid to lighting. In general, arbitrary lighting of the environment will not produce optimum results as it may result in low-contrast images, reflections, shadowing and extraneous details. A good lighting system illuminates the scene so that the complexity of the image is minimized while the information required for inspection or manipulation is enhanced.

2.3.1 Front Lighting

A front lit scene (where the MicronEye is on the same side of the scene as the light source or ambient light) sometimes lacks adequate contrast. Front lighting with a diffused light source can often be used to increase the contrast in a scene. If defects or points of interest are to be emphasized, side lighting such that the defects or points of interest cast a shadow or appear brighter through increased reflectivity may be used.

To set up a front lit scene, one or more flood lamps (found at most hardware stores) are arranged around the scene far enough away so that there are no shadows. Then the f-stop, focus control and lamps are adjusted for maximum contrast and focus. It is usually helpful to adjust the focus where the smallest part of the scene has the most detail.

In many instances you will want to diffuse the light coming from the flood lamps. Diffusing the light increases the uniformity of the light on the image. You can diffuse the light as simply as placing a piece of paper over the lamp. A better method of diffusion is to take a sheet of frosted mylar, diffused white plastic, or a sandblasted pane of glass and place it between the lamp and the subject. A diffused light source is most commonly used in defect detection and visual inspection applications.

2.3.2 Back Lighting

For a backlit scene, the light comes from behind the scene so that the object being viewed is shadowed into the MicronEye. Backlighting the object for maximum contrast will give the best repeatable results. Backlighting is recommended when using the MicronEye to measure an object or certain aspects of an object. Backlighting is often ideal for part recognition.

The backlit light source must be large enough so that the MicronEye, without the object in the field of view, will see a uniform amount of light. This is normally accomplished by using several flood lamps and shining the flood lamps onto a diffused surface (ground glass, or diffused white plastic, or frosted mylar), such that a uniform light source is created. Placing the object between the diffused surface and the MicronEye will shadow the object into the MicronEye with maximum contrast. Adjust the f-stop to the maximum value that the amount of light and exposure time will allow.

CHAPTER 3

USING THE MICRONEYE WITH THE APPLE

3.1 INSTALLATION AND SET UP

The MicronEye configured for use with the Apple II requires at least 48K of memory. The MicronEye is compatible with the Apple II+ and the Apple IIe.

Remove your MicronEye from its shipping carton. If you have purchased a Bullet, it will already be fully assembled. All that is required of you is to unfold the legs of the tripod and stand the MicronEye upright. If you have purchased a Camera, you will have to connect the Camera to the interface board with the cord which is provided.

Take a moment to examine the lens provided with the MicronEye. You will notice that there are two lens controls which must be adjusted before the MicronEye will operate successfully: f-stop and focus control. The f-stop controls the amount of light admitted through the lens and, for normal use, the lowest setting (1.6) is recommended. Any increase in the f-stop requires a compensating increase in the light source or in the exposure time. The recommended operating distance of the MicronEye is 18 inches or greater from the object it is viewing. You may be required to make a slight adjustment to the f-stop setting and/or the focus control once you have the MicronEye actually viewing an object.

Switch off the power to your computer, and you are ready to install the interface card into any available slot in the Apple. With the computer keyboard facing you, insert the interface card into the computer with the components on the right side of the card. The computer initially expects it in slot 2, but this can be changed from the keyboard once inside the program.

Insert the MicronEye diskette into the disk drive and switch on the power to your Apple. The MICRONEYE program, discussed in detail below, is automatically invoked when the Apple is turned on.

3.2 FILES INCLUDED ON YOUR MICRONEYE DISKETTE

To assist you in developing personal applications for the MicronEye, both source listings and programs have been included in your diskette. A catalog and brief description of the files found on your diskette follows:

APPLESOFT CATALOG	
A 045 MICRONEYE	(MICRONEYE program as discussed below)
A 012 COMMANDER	(COMMANDER program as discussed below)
A 014 GREYPIC	(GREYPIC program as discussed below)
A 011 GREYSCREEN	(GREYSCREEN program as discussed below)
A 011 ENHANCED EYE	(ENHANCED EYE program as discussed below)
A 003 SLIDE SHOW	(runs GREYSCREEN pictures on this disk)
T 033 T.CAMASM	(source for CAMASM)
B 006 CAMASM	(6502 routines for MICRONEYE and CAMASM)
T 011 T.EPRINT	(source for EPRINT)
B 003 EPRINT	(6502 Epson screendump routine)
T 029 T.GREYASM	(source for GREYASM)
B 005 GREYASM	(6502 routines for GREYPIC)
T 018 T.GSCRASM	(source for GSCRASM)
B 003 GSCRASM	(6502 routines for (GREYSCREEN)
T 041 T.ENHANCER	(source for ENHANCER)
B 006 ENHANCER	(6502 routines for ENHANCED EYE)
T 002 EYEPARMS	(parameter file for MICRONEYE)
B 008 MEYEAPP	(6502 routines for APPLICATIONS SUBROUTINES)
T 034 T.MEYEAPP	(source for APPLICATIONS SUBROUTINES)
A 007 GREY16	(16 shades of grey program, BYTE Oct '83)
B 006 GREY16-48K	(6502 routines for GREY16 on a 48K Apple)
B 007 GREY16-64K	(6502 routines for GREY16 on a 64K Apple)
B 034 BAMBI	(picture created using GREYSCREEN)
B 034 BAMBI AND FLOWER	(picture created using GREYSCREEN)
B 034 ROBOTARM	(picture created using GREYSCREEN)
B 034 WINNIE	(picture created using GREYSCREEN)
B 034 BEARS	(picture created using GREYSCREEN)

PASCAL DIRECTORY		
MICRONEYE.CODE	19	(MICRONEYE program as discussed below)
COMMANDER.CODE	4	(COMMANDER program as discussed below)
GREYPIC.CODE	12	(GREYPIC program as discussed below)
CAMASM.CODE	9	(6502 routines for MICRONEYE/COMMANDER)
GREYASM.CODE	6	(6502 routines used by GREYPIC)
SCREENIO.CODE	7	(Screen handling library code file)
MICROCAM.TEXT	30	(Source code for MICRONEYE)
COMMANDIT.TEXT	6	(Source code for COMMANDER)
GREYPIC.TEXT	10	(Source code for GREYPIC)
CAMASM.TEXT	26	(Source code for CAMASM)
GREYASM.TEXT	16	(Source code for GREYASM)
SCREENIO.TEXT	10	(Source code for SCREENIO)

EYEPARMS 1 (Parameter file used by MICRONEYE)

NOTE: The Pascal version of the MicronEye does not have the Applesoft equivalents of GREYSCREEN and ENHANCED EYE.

3.3 THE MICRONEYE PROGRAM

The MICRONEYE program lets a non-technical user harness a great deal of the MicronEye's power. The program incorporates the ability to show pictures transferred from the MicronEye onto your computer's screen, save pictures to disk for future use, and print pictures to a graphics printer.

When the program is invoked, a menu similar to the screen below is displayed:

```
            MICRONEYE DEMONSTRATOR

   WHAT WOULD YOU LIKE TO DO? _

      (1) START CAMERA
      (2) SET UP CAMERA PARAMETERS
      (3) DISPLAY REAL-TIME COMMANDS
      (4) SAVE CURRENT CAMERA SETUP
      (5) RECALL CAMERA SETUP FROM DISK
      (6) CHANGE SLOT AND BAUD RATE
      (7) TARGET PRACTICE
      (8) EXIT PROGRAM
```

3.3.1 START CAMERA

Starting the MicronEye causes the screen to blank, and prepares the computer to begin the display of pictures using your computer's high resolution graphics capabilities. The MicronEye then begins sending what it sees to your computer. The computer then displays this picture onto the computer's screen. The size of the picture displayed can be modified by using the "SET UP CAMERA PARAMETERS" option.

When the MicronEye begins sending pictures to your computer, the MicronEye has no way of knowing if the picture is properly focused or if the proper exposure time has been selected. If you are having difficulty focusing or selecting the proper exposure setting, refer back to the chapter 2 on OPERATING TECHNIQUES.

There are several single-key commands that you can use when the camera is operating. These commands allow you to increase or decrease the exposure time, save pictures to disk, recall pictures from disk, print pictures to a printer, enable and disable the display of information about each picture displayed, select fixed or automatic exposure times, etc. These commands are called real-time commands and are discussed in the "REAL-TIME COMMANDS" section.

While the MicronEye is operating, you can return to the main menu at any time by typing "Q".

3.3.2 SET UP CAMERA PARAMETERS

When you select this option, a screen similar to the one shown below will be displayed:

```
                MICRONEYE SETUP

SELECT LETTER OF DESIRED OPTION... _
       (PRESS <RETURN> TO EXIT)

PICTURE SIZE: A) 128 X 64    C) 256 X 128
              B) 256 X 64    D) 512 X 128

PICTURES/SCREEN:  E) 1 PER SCREEN
                  F) 2 PER SCREEN
EXPOSURE CONTROL: G) FIXED EXPOSURE TIME
                  H) AUTO-ADJUST EXPOSURE
STATUS READOUTS:  I) ENABLED
                  J) DISABLED
LIGHT MARGIN      K)
---------------------------------------

PICTURE SIZE: 256 X 128  (1 PIC/SCREEN)
READOUTS ARE: ENABLED
EXPOSURE IS:  FIXED

EXPOSURE LEN: 250 MSECS
LIGHT LEVEL:   45 %     MARGIN   5 %
```

3.3.2.1 PICTURE SIZE - Options "A" through "D" select the size of the picture that the MicronEye sends to the computer. Each picture is made up of thousands of black and white dots called pixels. When we say a picture is 128 x 64 in size, this means that the picture is made up of 64 rows of dots and that each row contains 128 dots of pixels. A 256 x 128 picture is made up of 32,768 pixels. Each pixel is either black or white.

The 128 x 64 and 256 x 128 picture size selections are compressed in the horizontal direction. The 256 x 64 and 512 x 128 picture size selections produce an image of normal proportions. Only the leftmost 280 pixels of the 512 x 128 picture will fit on the graphics screen.

3.3.2.2 PICTURES PER SCREEN - The MicronEye can take either one or two pictures at a time. If you elect to look at two pictures per screen, the computer will put the second picture right below the first picture. At first glance it may appear that you have just one picture that is twice as high when the computer is showing one picture per screen. If you look closely though, you may see that where the two pictures meet there is a slight discontinuity. For some applications this may not matter. In more exacting applications, you should restrict yourself to using only one picture per screen.

3.3.2.3 EXPOSURE CONTROL - You have the option of using a fixed or variable exposure time. Exposure time corresponds to the shutter speed of conventional 35mm cameras. If the picture from the MicronEye is too dark, a longer exposure time can be specified. If the picture is too light, a shorter exposure time can be specified. Exposure time can alternately be controlled by the use of real-time commands. The exposure time is specified in milliseconds. The speed at which the camera operates is equal to the exposure setting as long as the exposure time is greater than the time required for the MicronEye to transmit the picture to the computer. A more complete discussion of the interaction between exposure time and transmission time can be found in Appendix B.

As an alternative to manual exposure time control, automatic exposure adjustment can be specified from this setup menu or as a real-time command. Selecting the auto-adjust option tells the computer to evaluate the picture as it comes from the MicronEye to determine what percent of the pixels are white and what percent are black. When readouts are enabled, the percentage associated with LIGHT LEVEL is an approximation of how white the picture is: 100% being all white, 0% being all black.

When you select the auto-adjust feature you are requested to specify a light level between 0 and 100 and a margin which specified the allowed discrepancy from the prescribed light level. If you specify a light level of 45% and a margin of 5% then after each picture is received from the MicronEye, the computer will determine if the light level was between 40% and 50% (45% plus/minus 5%). If the light level was within the set bounds then the exposure time is left alone. If the light level is out-of-bounds then the exposure time is adjusted upward or downward to try and bring the next picture into the prescribed range.

The margin setting is also utilized by the alarm mode to set sensitivity. The alarm mode is explained in the section on real-time commands.

3.3.2.4 STATUS READOUTS - After displaying a picture from the MicronEye, the computer can optionally display the exposure time and light level of the picture just displayed. When status readouts are enabled, this information is displayed. Enabling this option, will slow down the rate at which pictures are updated on the screen. How much slower will depend on the exposure time setting and the type of computer you have.

In addition to being able to control readouts from the setup menu, a real-time command is available to enable and disable the readout display. On some computers, you may experience a difference in your picture's light level when switching back and forth between having readouts enabled and disabled.

3.3.2.5 LIGHT MARGIN - This is a convenient way of setting the light margin without altering the light level setting. It is especially useful for changing the MicronEye's sensitivity when being used in the alarm mode.

3.3.3 DISPLAY REAL-TIME COMMAND

There are several keystroke commands that can change how the MicronEye operates. After the computer displays each picture on the screen, it checks to see if a key has been pressed on the keyboard. If a key has been pressed, the computer checks to see if the key hit corresponds with its list of valid real-time commands. If so, the command is executed. If more than one key has been pressed during the scan only the last key struck is used.

Selecting the "DISPLAY REAL-TIME COMMANDS" options shows you the list of valid real-time commands. The screen should look somewhat like this:

```
          REAL-TIME COMMAND SUMMARY

   < -- DECREASE EXPOSURE TIME
   > -- INCREASE EXPOSURE TIME
   A -- TOGGLE ALARM MODE ON/OFF
   C -- CLEAR SCREEN
   F -- FIX EXPOSURE TIME TO CURRENT
        SETTING
   L -- LOAD PICTURE FROM DISK
   N -- PRINT PICTURE NEGATIVE ON EPSON
   P -- PRINT PICTURE ON EPSON
   Q -- QUIT AND RETURN TO MAIN MENU
   R -- TOGGLE DISPLAY READOUTS ON/OFF
   S -- SAVE PICTURE TO DISK
   T -- USING BLACK/WHITE RATIO (LIGHT
        LEVEL) OF CURRENT PICTURE, START
        AUTOMATIC LIGHT LEVEL TRACKING
```

The effects of each real-time command are explained in the pages that follow.

3.3.3.1 DECREASE EXPOSURE TIME - This command is activated by pressing the less-than key (comma also works). Each time this command is issued, the computer will decrease the MicronEye's exposure time. Each time the command is given the computer will decrease the exposure time in larger and larger steps. If the steps get too large, the computer may decide to decrease the exposure time in smaller and smaller steps. You may want to enable readouts and experiment with the increase and decrease exposure commands to get a better feel for how the commands interact and how the step size is increased and decreased by different combinations of the commands.

3.3.3.2 INCREASE EXPOSURE TIME - This command is activated by pressing the greater-than key (period also works). Its operation is similar to the "DECREASE EXPOSURE TIME" command except that the exposure time is increased rather than decreased.

3.3.3.3 TOGGLE ALARM MODE - This command is activated by the "A" key. If the alarm mode is off when you give this command, then alarm mode will be turned on. If the alarm mode is enabled then giving this command will disable the alarm mode. When you issue the command the computer will tell you whether you have enabled or disabled the alarm.

The alarm mode allows the MicronEye to function as a surveillance device. The light margin setting determines the sensitivity of the alarm. The greater the light margin setting, the less sensitive the MicronEye will be to change. The alarm is activated by changes in light level. If an object moves across the camera's field of view, an alarm will sound until a key is struck on the Apple's keyboard.

A user can also customize the computer's response to the alarm being tripped. The computer could automatically dial a phone number, activate recording equipment, etc.

3.3.3.4 CLEAR SCREEN - The computer clears the screen when the "C" key is struck. This command is rarely needed because the computer tries to clean up after itself whenever the size of the viewing area is changed.

3.3.3.5 FIX EXPOSURE TIME TO CURRENT SETTING - This command is invoked by striking the "F" key. The MicronEye normally uses the same exposure setting time after time, and only modifies the exposure setting when told to do so. This is referred to as a fixed exposure setting. The MicronEye can also operate such that the exposure time will change dynamically to maintain a specified light level. This is referred to as an auto-adjust setting.

When the camera is in the auto-adjust mode and you want to return to the fixed exposure mode use this command. The camera will fix the exposure time to the exposure time being used at the time the command is given.

3.3.3.6 LOAD PICTURE FROM DISK - A picture that was previously taken by the MicronEye and saved to disk can be displayed on the computer's screen by using this command. The load command is invoked by striking the "L" key.

The computer will then ask for the name given the picture when it was stored to disk. If the computer can find the file on disk, the picture will be displayed until a key is typed on the keyboard. Otherwise, an error message will be displayed and the computer will resume displaying pictures from the MicronEye. If you simply press the <RETURN> key when prompted for a file name, then the computer will resume displaying pictures.

3.3.3.7 PRINT PICTURE ON EPSON - The "P" key causes the current picture being displayed to be printed on an Epson graphics printer in slot 1. This command can also be used after loading a picture from disk, by typing a "P" when prompted to "press <RETURN> to continue..."

The routine is intended for an Epson printer using a parallel interface. Attempting to select the print option without a printer or a non-Epson parallel printer will cause the program to hang. Some early models of the Epson graphics printer may not work properly either. The reason for all of the problems associated with printing graphics is that the standard PRINT and COUT routines will insert unwanted line feeds and carriage returns into the print stream.

If you have a screen dump routine for your printer, you should modify lines 2010, 1180, and 1190 of the MicronEye program to use your screen dump routine rather than the one supplied. An alternative to this approach would be to save the picture in uncompressed format (refer to SAVE PICTURE section) and then run your screen dump program to print the picture.

3.3.3.8 PRINT PICTURE NEGATIVE ON EPSON - This option is invoked by typing the "N" key. It operates exactly like the normal print option with the exception that white areas on the screen will print black, and black areas will print white.

3.3.3.9 QUIT AND RETURN TO MAIN MENU - You can return to the main menu by typing "Q". When you no longer wish to operate the MicronEye, select this option.

3.3.3.10 TOGGLE DISPLAY READOUTS ON/OFF - Display readouts are enabled or disabled by typing "R". If readouts are enabled then after each picture is received from the MicronEye, the computer will display the exposure time and light level for that picture. When readouts are enabled, the picture rate may be slowed down dramatically, so it is usually advisable to have readouts disabled whenever possible.

3.3.3.11 SAVE PICTURE TO DISK - Typing an "S" when the camera is operating tells the computer to save the current picture to disk. The computer will prompt for a filename and attempt to save the picture to disk. If an error is encountered in attempting to save the picture (usually due to insufficient disk space) then a message is displayed. Otherwise the picture is stored to disk.

(Applesoft only.) Normally, the MicronEye program will compress the picture before storing it to disk. Although this saves a lot of disk space, the pictures saved are incompatible with commercially available graphics manipulation packages and screen dump programs. If you prefer that the MicronEye program store pictures in a conventional, non-compressed format then perform the following sequence of DOS and Applesoft commands:

```
LOAD MICRONEYE
2027 FF = 1
SAVE MICRONEYE
```

The Pascal version of MicronEye saves pictures in compressed format only. This is because a standard format for a .FOTO file has not been defined by Apple or other graphics software companies.

3.3.3.12 TRACK EXPOSURE TIME USING AUTO LIGHT LEVEL ADJUST - The auto-adjust mode is selected by typing a "T". When auto-adjust is selected as a real-time option, the computer will use the light level of the current picture as the ideal light level. The light margin is the acceptable level of deviation from the ideal light level and should have been set previously from the MICRONEYE SETUP screen.

After each picture is received from the MicronEye, the computer determines if the light level was within the established bounds. If not, the computer will increase or decrease the exposure time of the next picture to try and get back to an acceptable light level. The auto-adjust mode is intended for applications where the MicronEye is focused on a fixed or semi-fixed scene.

3.3.4 SAVE CURRENT CAMERA SETUP

Selecting this option from the main menu tells the computer to save the currently defined setup as the setup the computer should initially use when starting the MicronEye program. The setup variables that are stored include PICTURE SIZE, PICTURES PER SCREEN, EXPOSURE METHOD, EXPOSURE TIME, READOUT SETTING, LIGHT LEVEL, LIGHT MARGIN, MICRONEYE SLOT, and BAUD RATE. The setup is saved to a file called EYEPARMS.

3.3.5 RECALL CAMERA SETUP FROM DISK

This option restores the camera setup to the settings in the EYEPARMS file. This is handy when you have been experimenting with a non-standard setup and want to go back to using your normal setup.

3.3.6 CHANGE SLOT AND BAUD RATE

When shipped from the factory the MicronEye has been set to operate at a baud rate of 153,600 bits/second. Also the MicronEye program expects the MicronEye to go in slot 2. The baud rate will not normally be changed by the user. However, since a slot may currently contain another card it is helpful to be able to specify an alternate slot for the MicronEye. It is usually desirable to save the current setup to disk after modifying the slot or baud rate since these changes are fairly permanent in nature.

3.3.7 TARGET PRACTICE

This option may prove useful to some users. It temporarily puts the MicronEye in auto-adjust mode, sets the ideal light level to 50%, and adjusts the exposure time after each frame until a 50% light level is achieved. When in target practice, striking any key on the keyboard will return you to the main menu and return the setup to what it was prior to invoking target practice.

The target practice feature was included mainly to demonstrate how auto-adjust mode works.

3.4 THE COMMANDER PROGRAM

The COMMANDER program is a lower level program than the MICRONEYE program. The program asks for a hexidecimal (Pascal) or decimal (Applesoft) command. This command corresponds with the command descriptions found in Chapter 7.

If the SEND mode is selected in the command byte, the user is also prompted for a soaktime. In the COMMANDER program, soaktime is the time in milliseconds that the program will wait at the end of each frame to allow the camera additional exposure time. If SOAK mode is selected, then the total exposure time will be the transmission time plus the soaktime.

If SOAK mode is not selected then total exposure time will equal the soaktime. The computer will continue to send the camera the specified command until the user types a key on the keyboard. The user will be reprompted for another command unless the letter typed was a "Q". A "Q" will exit the program.

Commands less than decimal 192 (CO hex) will inhibit the camera from operating properly and probably cause the computer to hang.

3.5 THE GREYPIC PROGRAM

The GREYPIC program is a simplistic but effective demonstrator of the MicronEye's grey scale capabilities. By taking the same picture at several exposure settings, the program assigns a grey level to each pixel depending on the number of times it was white throughout the several exposure settings. Utilizing several of the features of the MICRONEYE program, the GREYPIC program allows for real-time adjustment of exposure time, saving and retrieving grey scale pictures on disk, and pasting together several pictures to make a larger composite picture. The GREYPIC program should be easily changed to work on other graphics printers or even on standard dot matrix or line printers.

The program is designed to operate with an EPSON printer (with Graftrax) in slot 1. When the program is run you are asked to specify the slot the MicronEye is in. You are also reminded to make sure that the printer is online to prevent the program from hanging. The program then begins displaying the picture being received from the MicronEye on the upper third of the screen. The exposure time is initially set to 1/3 of a second.

The GREYPIC program requires some setup to get a clear image. A high and low setting for the exposure range must be set. This can be accomplished with the use of the L, H, and B commands. The high exposure setting must be decided upon such that the image is not too light to display all of the details of the object being viewed on the screen. The dark exposure setting should be set so that no streaking occurs on the screen. Any slight discrepency in exposure time can be corrected using the increase and decrease exposure time commands. Be sure that the object is in focus and the F-Stop is at the correct setting.

To make a composite picture, place the object being viewed to show the upper most details which are desired to be displayed on the screen. On the screen, some distance should be allowed between the edge of the picture and the image of the object. After the picture is sharp and clear, a printout can be made. Press the P command and the upper third of the screen will be printed.

Press the "2" key to display the picture on the middle third of the screen. Raise the level of the object until the image on the upper third of the screen is directly on top of the image being displayed on the middle third of the screen. A flowing, continuous picture should be evident on the screen with no recognizable division between the two pictures on the screen. Press the P command and the middle third of the screen will be printed.

Press the "3" key to display the picture on the bottom third of the screen. Again, raise the level of the object until the bottom image is directly alligned with the middle third of the screen. A continuous picture should be displayed on the screen with no obvious breaks between the three sections of the screen. Press the P command and the bottom third of the screen will be printed.

By using the N command to scroll the image, a picture of any desired length can be printed. Using the same method already described, scroll the screen up one third, raise the level of the object and print the section.

If the object being displayed on the screen by the camera is in a fixed position, the height of the camera can be raised or lowered. Keep in mind at all times that the MicronEye should be kept parallel and perpendicular with the object being viewed. Setting an object on a movable platform (like a music stand) is one possible way to raise the level of an object.

If it is desired to save the picture to disk, each section of the screen must be saved separately. Press the S command and the image will be saved to be used at a later date.

The grey-scaled picture can be recalled from disk using the R command. If you desire to print the picture that has been recalled then press "P" rather than <RETURN> once the picture has been displayed.

The following real-time commands control the MicronEye while using the GREYPIC program:

< -- Decrease exposure time (comma also works).
> -- Increase exposure time (period also works).
C -- Clear entire screen.
1 -- Use upper third of screen to display picture on.
2 -- Use middle third of screen to display picture on.
3 -- Use bottom third of screen to display picture on.
N -- Rollup screen. (Middle third of screen moves to upper third, bottom third moves to middle third, and bottom third of screen is cleared. When using this option, it is best to be displaying the picture in the bottom third of the screen.) By using this command and the 1, 2, and 3 commands the user can piece together a picture of any length.
E -- Display the current exposure time and change the exposure time to a new value.
L -- Use current exposure setting as the lowest exposure setting when creating a grey-scaled picture.
H -- Use current exposure setting as the highest exposure setting when creating a grey-scaled picture.
B -- Bracket the exposure range for a grey-scaled picture. (User will be prompted for a high and a low setting.)
P -- Create a grey-scaled picture using the current high and low exposure settings, and print the picture on the Epson. (Seven intermediate exposure levels are used in addition to the high and low values to create a picture with nine levels of grey.)
S -- Save a grey-scaled picture to disk rather than print it.
R -- Recall a grey-scaled picture from disk and print it.
D -- dump (BSAVE) entire hi-res screen to disk.
G -- get previously BSAVE'D hi-res screen from disk and display it.
Q -- Exit program.

The GREYPIC program is easily modified to create images with up to 256 levels of grey. Although your computer has no means of displaying this many levels of grey, there are some rather expensive devices available for displaying and printing such images.

3.6 THE GREYSCREEN PROGRAM

The GREYSCREEN program is a takeoff from the GREYPIC program. However, the GREYSCREEN program attempts to show pseudo-greytone images on the screen. Because the Apple has no true shades of grey, we must simulate the grey by alternating black and white pixels. As in the GREYPIC program the MicronEye uses different exposure times to determine shades of grey.

The GREYSCREEN program uses two different exposure times which are controlled from the keyboard. If a pixel from the camera is white for both exposure times, then the wide pixel on the screen is all white. If a pixel from the camera is black for both exposure times, then the wide pixel on the screen is all black. If a pixel from the camera is white at one exposure time, and black at the other, then one side of the wide pixel on the screen will be black and the other will be white.

As in the GREYPIC program, the screen is divided into three partitions. These partitions are selectable from the keyboard and allow a composite image to be created on the screen which may be printed or stored for later retrieval or manipulation.

The concept utilized by the GREYSCREEN program is easily transferrable to other computers. Computers such as the IBM PC, Commodore 64, and TRS-80 Color Computer have implemented a medium resolution graphics mode which uses two bits to represent the pixel color on the screen. At the very least, black, white, light grey, and dark grey are available for creating an image. The obvious advantage over the Apple is the fact that real shades of grey are available for display.
The real-time commands available for use with the GREYSCREEN program are:

```
<  -- Decrease exposure time (comma also works)
>  -- Increase exposure time (period also works)
1  -- Use top 1/3 of graphics screen
2  -- Use middle 1/3 of graphics screen
3  -- Use bottom 1/3 of graphics screen
N  -- Scroll screen up by one-third (N also works)
S  -- Save (BSAVE) screen to disk
L  -- Load (BLOAD) screen from disk
C  -- Clear screen
E  -- Set exposure time
P  -- Print image to EPSON (slot 1, requires Graftrax)
      (Can also be used immediately after LOAD command)
M  -- Modify delta between high and low exposure (default 20)
Q  -- Quit program
SPACEBAR -- Freezes frame until key hit
```

3.7 THE ENHANCED EYE PROGRAM

The ENHANCED EYE program goes a step further than any of the other Apple MicronEye programs. It manipulates the pixels received from the camera to improve the image quality. Because of the amount of processing required for manipulation, the processing is done between scans. This slows down the frame to frame operation of the camera but provides an image of greater quality than any of the other methods demonstrated in other programs.

When the program begins, the display image is 256 x 64. Once the subject has been focused and the appropriate light level determined, the user can type "E" to enter the ENHANCE mode, "F" to enter the ENHANCE mode with FILLIN, 'U' for UNENHANCED mode, and 'N' to return to the 256 x 64 mode.

The actual enhancement of the image is done relatively fast. But because of the way the Apple high resolution graphics are implemented (1 color bit and 7 data bits per byte) and the fact that the enhancement is performed on a bitmap image (8 data bits per byte), the time required to convert the bitmap image to the Apple format takes in excess of a second. For non-display applications the ENHANCER assembly language routine could be modified to perform the enhancement but skip the display to screen, thereby greatly increasing the operating speed of this program.
The following set of real-time commands are available from the ENHANCED EYE program:

```
<  -- Decrease exposure time (comma also works)
>  -- Increase exposure time (period also works)
E  -- Enhance image without fillin (512 x 128)
F  -- Enhance image with fillin (512 x 128)
U  -- Display unenhanced image
N  -- Display 128 x 64 image (rest of screen is not cleared)
G  -- Create grey-tone image from dual exposures
L  -- Load (BLOAD) screen from disk
P  -- Print image to EPSON (slot 1, requires Graftrax)
      (Can also immediately follow LOAD or GREY command)
S  -- Save (BSAVE) screen to disk
      (Can also immediately follow GREY command)
C  -- Clear screen
T  -- Set exposure time
Q  -- Quit program
SPACEBAR -- Freeze frame until key hit
```

3.8 THE GREY16 PROGRAM

The September and October 1983 issues of BYTE magazine contain two articles on a camera called the Micro DCAM and a review of the MicronEye. The Micro DCAM is functionally equivalent to the MicronEye in all respects. The GREY16 program mentioned in the article has been included on your diskette. (Some minor enhancements have been added to the program described in the article).

The commands available when running the GREY16 program are as follow:

N -- Display the image in normal size (256 x 64)
F -- Display the image in full size (256 x 128).
G -- Create a picture (256 x 128) with 16 levels of grey. This takes about 30 seconds and displays a countdown of the number of exposures.
E -- Change exposure time.
S -- Save picture to disk.
L -- Load picture from disk.
P -- Print picture on Epson.
Q -- Quit program.

CHAPTER 4

USING THE MICRONEYE WITH THE IBM PC

4.1 CREATING A BOOTABLE DISKETTE

The diskette included with your MicronEye is NOT copy-protected and contains source code for all programs. The MicronEye diskette does not contain any system files, and as such is not bootable. This section shows you how to create a "working" copy of your diskette that includes all the necessary system files. Alternately, you could choose to always use the MicronEye diskette in drive B: and change the MEYE.BAT file to read "A:BASIC B:MEYE".

To create a bootable version of the MicronEye diskette and a backup of the diskette you need the following:

1. Your MicronEye diskette;

2. Your DOS system diskette (with the DOS utilities on it);

3. An unused diskette (new unformated or old unused with data).

The two step process below will create a working copy of the MicronEye diskette on the unused diskette. This allows you to keep your original diskette in a safe place as a backup.

4.1.1 MULTIPLE DRIVE SYSTEMS Versus SINGLE DRIVE SYSTEMS

The same process that follows can be used by both multiple and single drive system owners -- the single drive owners just get to shuffle diskettes in and out of their drive a lot more.

4.1.2 STEP 1

This step puts the system on your MicronEye diskette. Insert the MicronEye diskette in drive B (Drive A for you single drive owners) and type "B:MOVDOS" and press return.

For this first step of the process, when the computer asks you to mount a diskette in:

DRIVE A: insert the DOS diskette

DRIVE B: insert the MicronEye diskette.

Step one is complete when the message "Insert target diskette in A" appears.

4.1.3 STEP 2

This step builds your working MicronEye copy on your unused diskette. Be aware that this step will destroy the current content of your unused diskette if it contains data.

For this second part of the process, when the computer asks you to mount a diskette in:

DRIVE A: insert your unused diskette.

DRIVE B: insert the MicronEye diskette.

Step two is complete when the "$$$MOVDOS IS COMPLETE$$$" message appears.

Your working copy of the MicronEye is now ready. Store your original MicronEye diskette in a safe place as a backup.

4.2 INSTALLATION AND SETUP

The MicronEye configured for use with the IBM PC requires at least 96K of memory and the Color/Graphics Monitor Adaptor board. A color monitor is not required. Your monitor should be attached via the composite video connector or the RGB connector. The MicronEye is compatible with both the IBM PC and the IBM XT.

Remove your MicronEye from its shipping carton. If you have purchased a Bullet, it will already be fully assembled. All that is required of you is to unfold the legs of the tripod and stand the MicronEye upright. If you have purchased a Camera, you will have to connect the Camera to the interface board with the cord which is provided.

Take a moment to examine the lens provided with the MicronEye. You will notice that there are two lens controls which must be adjusted before the MicronEye will operate successfully: f-stop and focus control. The f-stop controls the amount of light admitted through the lens and, for normal use, the lowest setting (1.6) is recommended. Any increase in the f-stop requires a compensating increase in the light source or in the exposure time. The recommended operating distance of the MicronEye is 18 inches or greater from the object it is viewing. You may be required to make a slight adjustment to the f-stop setting and/or the focus control once you have the MicronEye actually viewing an object.

Switch off the power to your computer, and you are ready to install the interface card into any available slot in the computer. With the computer keyboard facing you, insert the interface card into the computer with the components on the right side of the card. The interface card does not include a mounting bracket. Remove the retaining bracket corresponding to the slot into which you are inserting the interface card. The cable between the MicronEye and interface card should be routed through the opening created by removing the retaining bracket. Replace the cover on your computer and turn on your computer.

4.3 FILES INCLUDED ON YOUR MICRONEYE DISKETTE

To assist you in developing personal applications for the MicronEye, both source listings and programs have been included in your diskette. A catalog and brief description of the files found on your diskette follow:

MOVDOS.BAT	(Command file to create bootable diskette)
MICRONEY.BAT	(Command file to invoke the MEYE program)
MEYEDRVR.ASM	(Source file for 8088 MicronEye routines)
MEYEDRVR.BAS	(BLOADable 8088 routines for the MicronEye)
MEYE.BAS	(BASIC MEYE program described below)
MEYECOMP.BAT	(Command file to assemble MEYEDRVR)
MEYESAVE.BAS	(BASIC program used by MEYECOMP)
MEYEPARMS	(Optional parameter file for MEYE)
MEYE88.ASM	(MicronEye Applications Library assembly language routines)
MEYE.EXE	(Compiled-BASIC version of the MicronEye program)
MEYEC.BAS	(ASCII version of MEYE.BAS to work with compiled-BASIC
MEYEDRVC.ASM	(Assembly language driver for the MicronEye)
MEYEC.OBJ	(Linkable object module version of MEYEC.BAS)
MEYEDRVC.OBJ	(Linkable object module version of MEYEDRVC.ASM)
MEYE88.OBJ	(Linkable object module version of MEYE88.ASM)
MEYEC.BAT	(Batch file used to create MEYE.EXE)
LCS.BAT	(Sample batch file to link user program with MEYE88)

4.4 THE MEYE PROGRAM

The MICRONEYE program lets a non-technical user harness a great deal of the MicronEye's power. The program incorporates the ability to show pictures transferred from the MicronEye onto your computer's screen, save pictures to diskette for future use, and print pictures to a graphics printer. Run the program by simply typing MICRONEYE or MEYE in response to the system prompt.

When the program is invoked, a menu similar to the screen below is displayed:

MicronEye Demonstrator

MICRONEYE ACTIVITY OPTIONS	MICRONEYE SETUP OPTIONS
Selection... 1) Start MicronEye 2) Change setup 3) Recall setup from diskette 4) Save setup to diskette 5) Explain real-time commands 6) Exit program	Picture size and type a) 128 x 64 (black/white) b) 512 x 64 (black/white) c) 512 x 64 (grey) d) 512 x 128 (black/white) e) 640 x 128 (black/white) f) 640 x 128 (grey)
CURRENT MICRONEYE SETUP	Mode settings (toggled) g) Pictures/screen(1 or 2) h) Status readouts(ON/OFF) i) Exposure (FIXED/AUTO)
Picture size: 640 x 128 (grey) Pics/screen: 1 Readouts: DISABLED Exposure: FIXED Exposure time: 300 Light level: 50% margin: 5%	Exposure control j) Set exposure time k) Set light level l) Set light margin

4.4.1 START CAMERA

Starting the MicronEye causes the screen to blank, and prepares the computer to begin the display of pictures using your computer's high resolution graphics capabilities. The MicronEye then begins sending what it sees to your computer. The computer then displays this picture onto the computer's screen. The size of the picture displayed can be modified by using the "CHANGE SETUP" option.

When the MicronEye begins sending pictures to your computer, the MicronEye has no way of knowing if the picture is properly focused or if the proper exposure time has been selected. If you are having difficulty focusing or selecting the proper exposure setting, refer to the chapter 2 on OPERATING TECHNIQUES.

There are several single-key commands that you can use when the camera is operating. These commands allow you to increase or decrease the exposure time, save pictures to diskette, recall pictures from diskette, print pictures to a printer, enable and disable the display of information about each picture displayed, select fixed or automatic exposure times, etc. These commands are called real-time commands and are discussed in the "REAL-TIME COMMANDS" section.

While the MicronEye is operating, you can return to the main menu at any time by typing "Q".

4.4.2 CHANGE SETUP

After selecting this option the computer expects you to change one of the parameters (A through L) displayed on the right half of the screen. After changing the desired parameters simply press the SPACEBAR to exit the CHANGE SETUP mode.

4.4.2.1 PICTURE SIZE AND TYPE - Options "A" through "F" select the size of the picture that the MicronEye sends to the computer. Each picture is made up of thousands of black and white dots called pixels. When we say a picture is 128 x 64 in size, this means that the picture is made up of 64 rows of dots and that each row contains 128 dots of pixels. A 512 x 128 picture is made up of 65,536 pixels. Each pixel is either black or white.

The 128 x 64 and 512 x 128 picture size selections are compressed in the horizontal direction. The 512 x 64 and 640 x 128 picture size selections produce an image of normal proportions. The 512 x 64 and 640 x 128 pictures sizes allow for two types of pictures--black & white or grey. Although the black and white picture may appear to have grey in it, this is a pseudo-grey caused by closely spaced black and white pixels. The grey picture is created by taking a second exposure of the same picture with a 20% shorter exposure time. The two pictures are then combined in software to produce a single picture on the screen. On a sophisticated imaging system this method is used to produce pictures with over 64 levels of grey.

4.4.2.2 PICTURES PER SCREEN - The MicronEye can take either one or two pictures at a time. If you elect to look at two pictures per screen, the computer will put the second picture right below the first picture. At first glance it may appear that you have just one picture that is twice as high when the computer is showing one picture per screen. If you look closely though, you may see that where the two pictures meet there is a slight discontinuity. For some applications this may not matter. In more exacting applications, you should restrict yourself to using only one picture per screen.

4.4.2.3 EXPOSURE CONTROL - You have the option of using a fixed or variable exposure time. Exposure time corresponds to the shutter speed of conventional 35mm cameras. If the picture from the MicronEye is too dark then a longer exposure time can be specified. If the picture is too light then a shorter exposure time can be specified. Exposure time can alternately be controlled by the use of real-time commands. The exposure time is specified in milliseconds. The speed at which the camera operates is equal to the exposure setting as long as the exposure time is greater than the time required for the MicronEye to transmit the picture to the computer. A more complete discussion of the interaction betwen exposure time and transmission time can be found in the section 5.0 of the manual.

As an alternative to manual exposure time control, automatic exposure adjustment can be specified from this setup menu or as a real-time command. Selecting the auto-adjust option tells the computer to evaluate the picture as it comes from the MicronEye to determine what percent of the pixels are white and what percent are black. When readouts are enabled, the percentage associated with LIGHT LEVEL is an approximation of how white the picture is: 100% being all white, 0% being all black.

When you select the auto-adjust feature you are requested to specify a light level between 0 and 100 and a margin which specified the allowed discrepancy from the prescribed light level. If you specify a light level of 45% and a margin of 5%, then after each picture is received from the MicronEye the computer will determine if the light level was between 40% and 50% (45% plus/minus 5%). If the light level was within the set bounds, the exposure time is left alone. If the light level is out-of-bounds, then the exposure time is adjusted upward or downward to try and bring the next picture into the prescribed range.

The margin setting is also utilized by the alarm mode to set sensitivity. The alarm mode is explained in the section on real-time commands.

4.4.2.4 STATUS READOUTS - After displaying a picture from the MicronEye, the computer can optionally display the exposure time and light level of the picture just displayed. When status readouts are enabled, this information is displayed. Enabling this option, will slow down the rate at which pictures are updated on the screen. How much slower will depend on the exposure time setting and the type of computer you have.

In addition to being able to control readouts from the setup menu, a real-time command is available to enable and disable the readout display. On some computers, you may experience a difference in your picture's light level when switching back and forth between having readouts enabled and disabled.

4.4.2.5 LIGHT MARGIN - This is a convenient way of setting the light margin without altering the light level setting. It is especially useful for changing the MicronEye's sensitivity when being used in the alarm mode.

4.4.3 DISPLAY REAL-TIME COMMAND

There are several keystroke commands that can change how the MicronEye operates. After the computer displays each picture on the screen, it checks to see if a key has been pressed on the keyboard. If a key has been pressed, the computer checks to see if the key hit corresponds with its list of valid real-time commands. If so, the command is executed. If more than one key has been pressed during the scan only the last key struck is used.
Selecting the "DISPLAY REAL-TIME COMMANDS" options shows you the list of valid real-time commands. The screen should look somewhat like this:

```
                 REAL-TIME COMMAND SUMMARY

< -- Decrease exposure time (comma also works)
> -- Increase exposure time (period also works)
A -- Toggle alarm mode on/off
C -- Clear screen
F -- Fix exposure time to current setting
L -- Load picture from diskette
P -- Print picture on printer
Q -- Quit and return to main menu
R -- Toggle display readouts on/off
S -- Save picture to diskette
T -- Use auto-adjust exposure (light level tracking)
/ -- Toggle pictures per screen (1 or 2)

1 -- 128 x 64 picture  (black & white)
2 -- 512 x 64 picture  (black & white)
3 -- 512 x 64 picture  (grey)
4 -- 512 x 128 picture (black & white)
5 -- 640 x 128 picture (black & white)
6 -- 640 x 128 picture (grey)
```

The effects of the various real-time commands are explained in the pages that follow.

4.4.4 DECREASE EXPOSURE TIME

This command is activated by pressing the less-than or comma key.

Each time this command is issued, the computer will decrease the MicronEye's exposure time. Each time the command is given the computer will decrease the exposure time in larger and larger steps. If the steps get too large, the computer may decide to decrease the exposure time in smaller and smaller steps. You may want to enable readouts and experiment with the increase and decrease exposure commands to get a better feel for how the commands interact and how the step size is increased and decreased by different combinations of the commands.

4.4.4.1 INCREASE EXPOSURE TIME - This command is activated by pressing the greater-than, or period key. Its operation is similar to the "DECREASE EXPOSURE TIME" command except that the exposure time is increased rather than decreased.

4.4.4.2 TOGGLE ALARM MODE - This command is activated by the "A" key. If the alarm mode is off when you give this command, then alarm mode will be turned on. If the alarm mode is enabled then giving this command will disable the alarm mode. When you issue the command the computer will tell you whether you have enabled or disabled the alarm.

The alarm mode allows the MicronEye to function as a surveillance device. The light margin setting determines the sensitivity of the alarm. The greater the light margin setting, the less sensitive the MicronEye will be to change. The alarm is activated by changes in light level. If an object moves across the camera's field of view, an alarm will sound until a key is struck on the keyboard.

A user can also customize the computer's response to the alarm being tripped. The computer could automatically dial a phone number, activate recording equipment, etc.

4.4.4.3 CLEAR SCREEN - The computer clears the screen when the "C" key is struck. This command is rarely needed because the computer tries to clean up after itself whenever the size of the viewing area is changed.

4.4.4.4 FIX EXPOSURE TIME TO CURRENT SETTING - This command is invoked by striking the "F" key.

The MicronEye normally uses the same exposure setting time after time, and only modifies the exposure setting when told to do so. This is referred to as a fixed exposure setting. The MicronEye can also operate such that the exposure time will change dynamically to maintain a specified light level. This is referred to as an auto-adjust setting.

When the camera is in the auto-adjust mode and you want to return to the fixed exposure mode use this command. The camera will fix the exposure time to the exposure time being used at the time the command is given.

4.4.4.5 LOAD PICTURE FROM Diskette - A picture that was previously taken by the MicronEye and saved to diskette can be displayed on the computer's screen by using this command. The load command is invoked by striking the "L" key.

The computer will then ask for the name given the picture when it was stored to diskette. If the computer can find the file on diskette, the picture will be displayed until a key is typed on the keyboard. Otherwise, an error message will be displayed and the computer will resume displaying pictures from the MicronEye. If you simply press the <RETURN> key when prompted for a file name, then the computer will resume displaying pictures.

4.4.4.6 PRINT PICTURE ON EPSON - The "P" key causes the current picture being displayed to be printed on an Epson or IBM printer. This command can also be used after loading a picture from diskette, by typing a "P" when prompted to "press <RETURN> to continue..."

4.4.4.7 QUIT AND RETURN TO MAIN MENU - You can return to the main menu by typing "Q". When you no longer wish to operate the MicronEye, select this option.

4.4.4.8 TOGGLE DISPLAY READOUTS ON/OFF - Display readouts are enabled or disabled by typing "R". If readouts are enabled then after each picture is received from the MicronEye, the computer will display the exposure time and light level for that picture. When readouts are enabled, the picture rate may be slowed down dramatically, so it is usually advisable to have readouts disabled whenever possible.

4.4.4.9 SAVE PICTURE TO Diskette - Typing an "S" when the camera is operating tells the computer to save to current picture to diskette. The computer will prompt for a filename and attempt to save the picture to diskette. If an error is encountered attempting to save the picture (usually due to insufficient diskette space) then a message is displayed. Otherwise the picture is stored to diskette.

4.4.4.10 TRACK EXPOSURE TIME USING AUTO LIGHT LEVEL ADJUST - The auto-adjust mode is selected by typing a 'T'. When auto-adjust is selected as a real-time option, the computer will use the light level of the current picture as the ideal light level. The light margin is the acceptable level of deviation from the ideal light level and should have been set previously from the MICRONEYE SETUP screen.

After each picture is received from the MicronEye, the computer determines if the light level was within the established bounds. If not, the computer will increase or decrease the exposure time of the next picture to try and get back to an acceptable light level. The auto-adjust mode is intended for applications where the MicronEye is focused on a fixed or semi-fixed scene.

4.4.5 SAVE CURRENT CAMERA SETUP

Selecting this option from the main menu tells the computer to save the currently defined setup as the setup the computer should initially use when starting the MicronEye program. The setup variables that are stored include PICTURE SIZE, PICTURES PER SCREEN, EXPOSURE METHOD, EXPOSURE TIME, READOUT SETTING, LIGHT LEVEL, and LIGHT MARGIN. The setup is saved to a file called MEYEPARMS.

4.4.6 RECALL CAMERA SETUP FROM Diskette

This option restores the camera setup to the settings in the MEYEPARMS file. This is handy when you have been experimenting with a non-standard setup and want to go back to using your normal setup.

CHAPTER 5

USING THE MICRONEYE WITH THE COMMODORE 64

5.1 INSTALLATION AND SET UP

Remove your MicronEye from its shipping carton. If you have purchased a Bullet, it will already be fully assembled. All that is required of you is to unfold the legs of the tripod and stand the MicronEye upright. If you have purchased a Camera, you will have to connect the Camera to the interface board with the cord which is provided.

Take a moment to examine the lens provided with the MicronEye. You will notice that there are two lens controls which must be adjusted before the MicronEye will operate successfully: f-stop and focus control. The f-stop controls the amount of light admitted through the lens and, for normal use, the lowest setting (1.6) is recommended. Any increase in the f-stop requires a compensating increase in the light source or in the exposure time. The recommended operating distance of the MicronEye is 18 inches or greater from the object it is viewing. You may be required to make a slight adjustment to the f-stop setting and/or the focus control once you have the MicronEye actually viewing an object.

Switch off the power to your computer, and you are ready to install the interface card into the cartridge slot located at the right rear of the computer. Insert the interface card into the slot using the orientation indicated on the enclosure.

Insert the MicronEye diskette into the disk drive and switch on the power to the computer.

5.2 FILES INCLUDED ON YOUR MICRONEYE DISKETTE

To assist you in developing personal applications for the MicronEye, both source listings and programs have been included in your diskette. A catalog and brief description of the files found on your diskette follows:

2	MICRONEYE	(Sample BASIC program using MEYE
12	MEYE6510.EX	(Assembly language routines di
50	MEYE.SRC.1	(Source listing part 1 for MEYE6
39	MEYE.SRC.2	(Source listing part 2 for MEYE6
22	MEYE.SRC.3	(Source listing part 3 for MEYE6
10	EXAMPLE	(Sample BASIC program using MEYE.BIN)
8	MEYE.BIN	(Routines described in Appendix H)
28	MEYE.OBJ	(Object code for MEYE.BIN)
2	MEYE.MAIN	(Subfile of MEYE.BIN)
3	MEYE.MACROS	(Subfile of MEYE.BIN)
6	MEYE.RLE	(Subfile of MEYE.BIN)
9	MEYE.ENHANCE	(Subfile of MEYE.BIN)
4	MEYE.VARIABLES	(Subfile of MEYE.BIN)
13	MEYE.ROUTINES	(Subfile of MEYE.BIN)

5.3 THE MICRONEYE PROGRAM

The MICRONEYE program lets a non-technical user harness a great deal of the MicronEye's power. The program incorporates the ability to show pictures transferred from the MicronEye onto your computer's screen, save pictures to disk for future use, and print pictures to an Epson or Gemini graphics printer.

To execute the MICRONEYE program, type LOAD "MICRONEYE",8 (followed by the RETURN key) and then type RUN. The program will load the assembly language routines (MEYE6510) for the MicronEye from disk. Once the routines have been loaded, the Commodore 64's screen immediately turn white and the MicronEye will begin sending pictures to the computer. You will see on the top two lines of the display the current operating mode of the MicronEye. When the program begins execution the mode display should read "B&W 1-PICTURE NORMAL". The second line of the display should read "SOAK TIME: 350".

Between pictures from the MicronEye, the computer checks for commands entered by the user on the keyboard. Because the MicronEye must operate with the interrupts turned off on the Commodore 64, the computer might not notice a key being pressed unless you keep the key pressed down a bit longer than you may be accustomed. As you work with the MicronEye you will acquire a feel for how long to keep the key pressed down. The best way to tell that the computer has noticed your command is to watch the mode display at the top of the screen. The mode display will be updated as soon as the command is detected.

The only problem associated with keeping the key pressed down is that when issuing the SAVE or LOAD commands you may need to use the DEL key to get rid of any extra characters that are displayed on the screen after FILENAME? before entering the LOAD or SAVE file name.

The MICRONEYE program allows several commands. They are explained in detail below. A summarization of the commands follows the explanation.

5.3.1 BLACK AND WHITE MODE

The BLACK AND WHITE option is selected by typing the "B" key. The MicronEye sees only in black and white. However, the computer can tell the MicronEye to take several pictures of the same scene at varied exposure times. The computer can then combine these several images together into a single picture with grey levels. Normally, a black and white image is adequate for processing an image. In this mode, the computer receives pictures from the MicronEye and displays each picture on the screen after it has been received.

5.3.2 GREY MODE

The GREY mode is selected by typing the "G" key. The GREY mode is the multiply-exposed grey level picture-taking technique alluded to above. In this mode, the computer instructs the MicronEye to take three pictures at varied exposure times. After the computer has received these three pictures, it "adds" them together and displays the result on the screen. Grey mode operates much slower than black and white mode because the computer has to get three pictures from the MicronEye for every picture displayed.

5.3.3 PICTURES PER SCREEN

The MicronEye can take either one or two pictures at a time. This is because the IS32 OpticRAM has 2 separate arrays which are both light-sensitive. By pressing the "1" key, the 1-PICTURE mode is selected. By pressing the "2" key, the 2-PICTURE mode is selected.

If you elect to look at two pictures per screen, the computer will put the second picture right below the first picture. At first glance it may appear that you have just one picture that is twice as high when the computer is showing one picture per screen. If you look closely though, you may see that where the two pictures meet there is a slight discontinuity. For some applications this may not matter. In more exacting applications, you should restrict yourself to using only one picture per screen.

You should be aware that when using the 2-PICTURE mode, the lower picture may have a tendency to be slightly darker than the upper picture. This is because the upper and lower array in the OpticRAM have a slightly different sensitivity to light. Since the OpticRAM was designed with the intent that only one of the arrays was to be used at a time, you might consider the second picture a freebie.

5.3.4 ENHANCED MODE

The ENHANCED mode is selected by typing the "E" key. The MicronEye can send images with either 128 x 64 resolution or 256 x 128 resolution. When using the 256 x 128 image size, resolution is increased fourfold. The increased resolution costs in two ways. First, it takes four times longer to send the 256 x 128 image than the 128 x 64 image. Second, the 256 x 128 image must be massaged through an enhancement algorithm to make a crisp image. This all takes time. To the extent that time is not a factor, the enhanced mode will generate much better pictures than the normal mode.

5.3.5 NORMAL MODE

NORMAL mode is selected by typing the "N" key and is the opposite of enhanced mode. Selecting NORMAL mode instructs the MicronEye to transmit 128 x 64 sized pictures.

5.3.6 DECREASE EXPOSURE TIME BY 10 MILLISECONDS

This command is activated by pressing the less-than key. Each time the less-than key is pressed, the computer will decrease the MicronEye's exposure time by 10 milliseconds. Keeping the less-than key pressed down continually will causes the exposure time to be decreased by some multiple of 10 milliseconds.

5.3.7 DECREASE EXPOSURE TIME BY 1 MILLISECOND

This command is activated by pressing the comma key (unshifted less-than key). Each time the comma key is pressed, the computer will decrease the MicronEye's exposure time by 1 millisecond. Keeping the comma key pressed down continually will causes the exposure time to be decreased by several milliseconds.

5.3.8 INCREASE EXPOSURE TIME BY 10 MILLISECONDS

This command is activated by pressing the greater-than key. Each time the greater-than key is pressed, the computer will increase the MicronEye's exposure time by 10 milliseconds. Keeping the greater-than key pressed down continually will causes the exposure time to be increased by some multiple of 10 milliseconds.

5.3.9 INCREASE EXPOSURE TIME BY 1 MILLISECOND

This command is activated by pressing the period key (unshifted greater-than key). Each time the period key is pressed, the computer will increase the MicronEye's exposure time by 1 millisecond. Keeping the period key pressed down continually will causes the exposure time to be increased by several milliseconds.

5.3.10 LOAD PICTURE FROM DISK

A picture that was previously taken by the MicronEye and saved to disk can be displayed on the computer's screen by using this command. The load command is invoked by pressing the "L" key.

The computer will then ask for the name given the picture when it was stored to disk. If the computer can find the file on disk, the picture will be displayed until a key is typed on the keyboard. Otherwise, an error message will be displayed and the computer will resume displaying pictures from the MicronEye. If you simply press the RETURN key when prompted for a file name, then the computer will resume displaying pictures.

5.3.11 PRINT PICTURE ON EPSON

The "P" key causes the current picture being displayed to be printed on an Epson printer. Pictures previously saved to disk can also be printed after using the LOAD command.

Your printer interface may require special setup for graphics printing. On line 25 POKE location 51251 with the necessary secondary address for graphics printing. If your printout has blank lines that break up the picture also add the statement POKE 51351,0. For example, the CARDCO interface requires the following setup:

```
25 POKE 51251,5: POKE 51351,0
```

5.3.12 SAVE PICTURE TO DISK

Typing an "S" when the MicronEye is operating tells the computer to save to current picture to disk. The computer will prompt for a filename and attempt to save the picture to disk. If an error is encountered attempting to save the picture (usually due to insufficient disk space) then a message is displayed. Otherwise the picture is stored to disk.

5.3.13 QUIT

You can exit the MICRONEYE program by pressing the "Q" key. If you no longer wish to operate the MicronEye, select this option.

5.3.14 COMMAND SUMMARY

The following list summarizes the commands which can be used to control the MicronEye:

E -- Enhanced picture (256 x 128 image)
N -- Normal picture (128 x 64 image)
B -- Black and white imaging (bi-level)
G -- Grey level imaging (4-level)
L -- Load picture from disk
S -- Save picture to disk
P -- Print picture on printer
1 -- Use one array of the OpticRAM
2 -- Use both arrays of the OpticRAM
< -- Decrease exposure time by 10 milliseconds

, -- Decrease exposure time by 1 millisecond
> -- Increase exposure time by 10 milliseconds
. -- Increase exposure time by 1 millisecond

5.4 THE ASSEMBLY LANGUAGE CONNECTION (MEYE6510.EX

The MICRONEYE program discussed above is a simple four line BASIC program that calls the assembly language program MEYE6510.EX. MEYE6510 loads into address $C000 (12*4096) of memory. Before the routine is called the "limit of memory pointer" at location 55-56 should be set to $2000. The MEYE6510 program uses all memory above this for screen storage and the MEYE6510 program itself. Set the limit with the BASIC instruction:

```
POKE 56,2*16 : POKE 55,0 : CLR
```

To help the user here is a general list of the way memory is allocated by the MEYE6510 program. All values are expressed in hexadecimal:

$2000-$3FFF	This area is used for the hi-res screen.
$4000-$7FFF	This area is the buffer used to receive the image from the MicronEye.
$8000-$BFFF	This area is used by the ENHANCED mode. The final image to be displayed is stored here.
$C000-$CB03	This is where MEYE6510 resides.
$E000-$FFFF	This area is used to store the incoming image from the MicronEye when in ENHANCED mode.

The MEYE6510 program was written with the intention that other users could write their own programs in BASIC and manipulate the MicronEye via calls to the various subroutines provided. The program was also designed to be extensible so that additional functions can be added as desired. A description of each of the primary subroutines available in MEYE6510 follows. The source listing is well-documented and should be referred to if you wish to make use of the more primitive subroutines which are not described below.

The number in parenthesis after the routine name is the decimal number that would be used in the SYS command to call the routine.

ONEARRAY (49869): Sets necessary parameters to operate

MicronEye in 1-PICTURE mode.

TWOARRAY (49895): Sets necessary parameters to operate MicronEye in 2-PICTURE mode.

ENHANCED (49947): Sets necessary parameters to operate MicronEye in ENHANCED mode.

NORMAL (49918): Sets necessary parameters to operate MicronEye in NORMAL mode.

BANDW (49583): Sets necessary parameters to operate MicronEye in BLACK AND WHITE mode.

CSHADE (49477): Sets necessary parameters to operate MicronEye in GREY mode.

CRAISE (49511): Decrease exposure time by 10 milliseconds. The exposure time can be set directly by POKEing EXPOSURE TIME / 256 into location 252 and POKEing EXPOSURE TIME mod 256 into location 251.

CRAS1 (49502): Decrease exposure time by 1 millisecond.

CLOWER (49551): Increase exposure time by 10 milliseconds.

CLOW1 (49542): Increase exposure time by 1 millisecond.

DSAVE (51014): Saves the current picture at location $2000-$4002 onto the disk drive. The routine exits hi-res mode, moves the picture to $8000, asks the user for a filename, and then attempts to save the picture to disk.

DLOAD (51086): The user is asked for a filename and the program then attempts to load the file. The error channel is not checked. The parameter set up at the time the picture was saved becomes the new parameter setup for the MicronEye. This routine assumes the hi-res screen is located at $2000.

SDUMP (51237): Dumps the hi-res picture to an Epson or Epson-workalike graphics printer.

UPDATE (50245): Updates the mode display (top 2 lines of hi-res screen) to reflect the current parameter settings and exposure time. The exposure time should be POKE'd in locations 253-254 prior to calling this routine.

ENMODE (49671): When in ENHANCE mode, use this routine to get a picture from the MicronEye and display it on the screen.

GETIT (49594): When in NORMAL mode, use this routine to get a BLACK AND WHITE picture.

SHDBIT (49608): When in NORMAL mode, use this routine to get a GREY picture.

TEXTMD (50931): Exits hi-res mode. Use when exiting from program so that the text screen shows like it should in BASIC.

CHAPTER 6

USING THE RS-232 MICRONEYE CAMERA

6.1 HARDWARE REQUIREMENTS

There are four lines running between the RS-232 MicronEye camera and the computer -- transmit, receive, ground and 5V. The RS-232 MicronEye provides a standard DB-25P connector for interfacing. The pinout for the connector is as follows:

Pin 3 -- Transmit data (from MicronEye)
Pin 2 -- Receive data (from computer)
Pin 7 -- Signal ground
Pin 11 -- 5V (from computer) must drive 50ma load

There are four lines running between the camera and the computer-- transmit, receive, ground, and 5 volts.

Standard RS-232 pinouts do not provide power to the RS-232 MicronEye camera. The user must supply a 5V DC, 50ma power souce on pin 11 of the DB-25P connector. This can be done by tapping the 5V supply on the computer or by using a separate voltage source. In either case, pin 7 is used as the ground line. Be certain that the power source is only 5V. Voltages in excess of 6V can permanently damage the MicronEye.

With power supplied to the cable and the cable attached to the camera and computer, the MicronEye camera is ready to operate.

The interface between the MicronEye and the computer is an RS-232 serial link. The connection is via a 6-line telephone cable. The lines are used for Vcc, ground, receive and transmit. The unused lines are not connected. The operating speed is controlled by the baud rate jumper setting on the MicronEye.

When using real-time image processing the programmer must make certain that the time required to perform special tasks between bytes does not exceed the time available. For example, a 9600 baud transmission rate means that 960 bytes per second will be transmitted.

The user can therefore expect to receive a new byte from the MicronEye every 1041 microseconds. Some computer configurations may require that the baud rate be slowed to guarantee receipt of data. Many single-user systems should be able to increase the baud rate to 19,200 without problem.

The standard RS-232 MicronEye is shipped with the following interface configuration:

One start bit
Eight data bits
One stop bit
9600 baud

This configuration applies to both transmit and receive lines. The user may modify only the baud rate selection. The host computer must be set to conform with the 1 start bit, 1 stop bit, eight data bits (with no parity) protocol expected by the MicronEye.

6.2 SOFTWARE

Appendix E contains the complete assembly language driver used for the IBM PC version of the MicronEye. It is annotated throughout and provides a reasonable baseline for developing sophisticated drivers for the MicronEye for microcomputer and minicomputer systems.

Please note that the IBM is a version B interface with respect to bit ordering while the RS-232 is a version A interface. Keep this in mind when performing shift and rotate instructions.

It is probable that a great deal of the code included in Appendix E will not be required for specific applications. The sofware does demonstrate communication techniques between the computer and MicronEye, enhancement techniques for the 256 x 128 image, 2-bit grey scale, and a printer dump routine for the Epson dot matrix graphics printer. When writing the driver to receive an image, it is a good idea to use a timeout routine to determine the end-of-frame rather than to expect a specified number of bytes. This prevents the computer from hanging if a transmission error occurs.

It is not necessary to use exclusively assembly language when working with the MicronEye. The use of higher level languages is more than appropriate if the code executes with adequate speed. The only time-critical code is the loop that receives an image from the MicronEye. By using the annotated listing it should be fairly easy to translate the various routines into higher level languages.

CHAPTER 7

HOW YOUR COMPUTER TALKS TO THE MICRONEYE

7.1 MICRONEYE VERSIONS

This section explains how to talk to the MicronEye and how to get information back from the MicronEye. We strongly recommend that users who are interested in developing their own assembly language drivers for the MicronEye study this section along with the assembly language routines included on the MicronEye diskette. We feel that the assembly language routines we have prepared are fairly complete and would advise the user to first determine that they would not be adequate for their needs before developing their own assembly language programs from scratch.

As you are aware, there are four different versions of the MicronEye, specifically designed to interface with a particular computer -- the Apple II, IBM-PC, Commodore 64, TRS-80 Color Computer. The RS-232 version is available for persons who do not have access to one of the computers mentioned above. Insofar as hardware configuration, the Apple II and RS-232 are similar and can be categorized together for purposes of this section. They will be referred to as "Version A" systems. The IBM-PC, Commodore 64 and TRS-80, likewise, are similar and will be referred to in this section as "Version B" systems. The difference between the the Version A and Version B systems is in the arrangement of the data bits. In Version A, the least significant bit represents the leftmost image pixel in the byte. In version B, the most significant bit represents the leftmost image pixel in the byte. This affects both commands being transmitted to the MicronEye and data being received from the MicronEye. The reason for the difference lies in the way the various computers display graphic information.

7.2 THE SERIAL CONNECTION

NOTE: RS-232 owners should disregard this section. The RS-232 MicronEye does not use the 6850 ACIA. The RS-232 MicronEye connects directly to an RS-232 port. Configuring the port properly depends on your computer and is discussed on the chapter discussing the RS-232 MicronEye. The RS-232 MicronEye user should disregard the discussion on reading and writing the ACIA status register as this is taken care of by the computer's RS-232 circuitry and firmware. The RS-232 MicronEye user need only worry about sending commands to the MicronEye as discussed in the following section and receiving images from the MicronEye.

The interface between the MicronEye and the computer is a serial link utilizing a Motorola 6850 ACIA. The connection is via a 6-line telephone cable. The lines are used for Vcc, ground, receive, transmit, and external clock. The 6th line is not connected. The operating speed is controlled by the baud rate jumper setting on the MicronEye circuit board. When using real-time image processing, the programmer must make certain that the time required to perform special tasks between bytes does not exceed the time available.

The ACIA is composed of a data register and a status register. Writing to the status register allows the user to configure items such as parity, stop bits, start bits, clocking, etc. Before accessing the MicronEye, the ACIA has to be initialized to the proper configuration, as follows:

VERSION A		VERSION B	
Write to status register:	hex $03	Write to status register:	hex $C0
followed by	hex $14	followed by	hex $28

The first byte performs a master reset on the ACIA, while the second byte specifies that the transmission protocol is 1 start bit, followed by 8 data bits, followed by 1 stop bit; and a x1 clock mode is to be used. (x1 clock mode requires that an external clock accompany the data to and from the computer which is furnished by the standard MicronEye interface card.

Reading the status register allows the user to determine when new data has been received and when the ACIA is ready to send data. The status bits, when set, mean:

VERSION A	VERSION B	STATUS BIT DESCRIPTION (REA
Bit 0	Bit 7	Data has been received from MicronEye. In normal use, this bit is only checked

		when seeing if data is available from the MicronEye.
Bit 1	Bit 6	A command may be sent to the MicronEye.
Bit 4	Bit 3	Received data improperly framed. Usually only used in a debug mode.
Bit 5	Bit 2	Data received before previous byte read. Usually only used in a debug mode.

Once the status register indicates that a command can be sent to the MicronEye, write the command to the data register. Conversely, when receiving an image from the MicronEye, read the data register when the status register indicates that data is available. When receiving an image from the MicronEye, it is a good idea to incorporate a timeout mechanism in case the MicronEye stops sending bytes before the program expects. Otherwise the program can hang if the software misses even a single byte.

7.3 COMMAND DEFINITIONS

The MicronEye has several operating modes. The command byte is organized as follows:

VERSION A	VERSION B	COMMAND
Bit 7	Bit 0	Always 1
Bit 6	Bit 1	Always 1
Bit 5	Bit 2	0 = Even rows and columns only (ALTBIT) 1 = All pixels in array (NOALTBIT)
Bit 4	Bit 3	0 = Double send each pixel (WIDEPIX) 1 = Send normally (NARROWPIX)
Bit 3	Bit 4	0 = 7-bit data bytes for Apple (7BIT) 1 = 8 data bits per byte (8BIT)
Bit 2	Bit 5	0 = transmit 1 array (1ARRAY) 1 = transmit upper and lower array (2ARRAY)
Bit 1	Bit 6	0 = refresh instead of soak (REFRESH) 1 = soak instead of refresh (SOAK)
Bit 0	Bit 7	0 = Send the requested image (SEND) 1 = Don't send -- soak or refresh (NOSEND)

7.3.1 ALTBIT And NOALTBIT MODES

The MicronEye will transmit only the pixels from the even-numbered rows and columns in the array. Because of the placement of the pixels in the image sensor, this mode will usually produce an image of clearer resolution than the NOALTBIT mode unless the image undergoes the enhancements discussed elsewhere in this manual. Software is provided on your disk that performs this enhancement.

With NARROWPIX and ALTBIT the image from the MicronEye is 128 x 64. With WIDEPIX and ALTBIT the image sent to 256 x 64. NARROWPIX and NOALTBIT causes a 256 x 128 image to be transmitted. WIDEPIX and NOALTBIT causes a 512 x 128 image to be sent.

7.3.2 WIDEPIX AND NARROWPIX MODES

The MicronEye will "double transmit" each pixel in the array when WIDEPIX is selected. Since each image sensing element in the IS32 OpticRAM is twice as wide rectangular in shape, "double transmitting" maintains the proper width to height ratio for displaying the image. There are many applications, however, where maintaining the proper ratio is less important than receiving the image as quickly and compactly as possible. In such a situation NARROWPIX would be the appropriate mode choice.

7.3.3 7BIT AND 8BIT MODES

The Apple computer is somewhat peculiar in its implementation of high resolution graphics. The most significant bit of each byte on the graphics page is reserved as the 'color' bit, while the other 7 bits are the pixels being displayed. In 7BIT mode, the MicronEye transmits data so that it is compatible with the Apple's high resolution format; or, in other words, 7 bits of image pixels per byte.

The alternative to 7BIT mode is 8BIT mode. 8BIT mode causes the MicronEye to transmit in normal bitmap format (all 8 bits in the byte contain image data). 8BIT mode is used by all computers other than the Apple. For non-display use on the Apple the 8BIT mode can be useful. (The GREYPIC program in the Apple software uses both the 8BIT mode and 7BIT mode as it creates grey-scale images for the Epson).

7.3.4 1ARRAY AND 2ARRAY MODES

The image sensor used by the MicronEye is comprised of dual 128 x 256 pixel arrays. If you remove the camera lens and look at the image sensor, you can clearly see the two arrays. Using the 1ARRAY mode, only the image focused on the lower array is transmitted from the MicronEye. On the other hand, using the 2ARRAY mode causes both arrays to be transmitted from the MicronEye. The 2ARRAY mode has a split screen effect because of the spacing between the two arrays in the image sensor chip. In addition, the sensitivity to light of the two arrays is usually noticeably different. These two factors tend to make 2ARRAY mode inappropriate for many applications.

7.3.5 REFRESH AND SOAK MODES

The MicronEye takes a picture much like any other camera. The MicronEye must have the proper amount of light to make the image develop properly. Too much light will overexpose the image, while too little light will underexpose the image.

When the REFRESH mode is selected, the circuitry in the MicronEye keeps the OpticRAM refreshed. Refreshing has two effects-- the OpticRAM is made insensitive to light and all the image sensing cells in the OpticRAM are set to either 5 volts (black) or 0 volts (white). All cells which have not leaked below 2.5 volts (threshhold) are refreshed to 5 volts. All cells which have leaked below 2.5 volts are refreshed to 0 volts. REFRESH does NOT set all the cells in the OpticRAM to 5 volts. The only way the cells in the OpticRAM can be reset to 5 volts is to have the MicronEye SEND an image. When the MicronEye reads the OpticRAM's pixels, it automatically sets each cell to 5 volts after reading its value. When the refresh mode is enabled, each cell in the OpticRAM is refreshed about once every 6.5 milliseconds.

When the MicronEye is not in a refresh mode, it is in SOAK mode. Whenever the MicronEye is in SOAK mode, the OpticRAM is sensitive to light. The intensity and duration of light focused on each image sensing element determines how fast and how long the voltage in the sensing element will continue to diminish.

7.3.6 SEND MODE

When a command is sent to the MicronEye with SEND mode selected, the MicronEye will begin transmitting an image. In nearly all cases, the command sent to the MicronEye will have the SEND mode selected. The only time SEND mode is not desirable is the situation where a significant amount of processing must take place between transmission of images. In this situation, a user may chose to receive an image from the MicronEye, send a command to the MicronEye and REFRESH without sending, go away and process the image, send a command to the MicronEye to SOAK without sending, wait for the desired exposure time, and then send a command to have the MicronEye transmit the image.

When the MicronEye is sent a command with the SEND bit clear, the MicronEye begins transmitting an image to the computer. After the image has been sent, the MicronEye stops transmitting data, goes into a soak state, and waits for a new command. When the MicronEye is sent a command with the SEND bit set to 1 then the camera will not transmit data and will refresh or soak depending on the setting of the REFRESH bit.

Please note that when the SEND bit is set to 1, the ALTBIT bit should also be set to 1. Failure to do so will cause the first row of the subsequent image to be offset by 1 pixel.

7.4 EFFECTS OF COMMAND MODE COMBINATIONS

The following table shows the effects of different commands to the MicronEye. The REFRESH/SOAK and SEND/NOSEND bits are not considered for purposes of this table.

VERSION A SYSTEMS

COMMAND (HEX)	COMMAND (DEC)	ROWS	BYTES PER ROW	PIXELS PER ROW	MODE SELECTION
C0	192	64	37	256	ALT WIDEPIX 7BIT 1ARRAY
C4	196	128	37	256	ALT WIDEPIX 7BIT 2ARRAY
C8	200	64	32	256	ALT WIDEPIX 8BIT 1ARRAY
CC	204	128	32	256	ALT WIDEPIX 8BIT 2ARRAY
D0	208	64	19	128	ALT NOWIDEPIX 7BIT 1ARRAY
D4	212	128	19	128	ALT NOWIDEPIX 7BIT 2ARRAY
D8	216	64	16	128	ALT NOWIDEPIX 8BIT 1ARRAY
DC	220	128	16	128	ALT NOWIDEPIX 8BIT 2ARRAY
E0	224	128	73	512	NOALT WIDEPIX 7BIT 1ARRAY
E4	228	256	73	512	NOALT WIDEPIX 7BIT 2ARRAY
E8	232	128	64	512	NOALT WIDEPIX 8BIT 1ARRAY
EC	236	256	64	512	NOALT WIDEPIX 8BIT 2ARRAY
F0	240	128	37	256	NOALT NOWIDEPIX 7BIT 1ARRAY
F4	244	256	37	256	NOALT NOWIDEPIX 7BIT 2ARRAY
F8	248	128	32	256	NOALT NOWIDEPIX 8BIT 1ARRAY
FC	252	256	32	256	NOALT NOWIDEPIX 8BIT 2ARRAY

VERSION B SYSTEMS

COMMAND (HEX)	COMMAND (DEC)	ROWS	BYTES PER ROW	PIXELS PER ROW	MODE SELECTION
03	03	64	37	256	ALT WIDEPIX 7BIT 1ARRAY
07	07	128	73	512	NOALT WIDEPIX 7BIT 1ARRAY
0B	11	64	19	128	ALT NOWIDEPIX 7BIT 1ARRAY
0F	15	128	37	256	NOALT NOWIDEPIX 7BIT 1ARRAY
13	19	64	32	256	ALT WIDEPIX 8BIT 1ARRAY
17	23	128	64	512	NOALT WIDEPIX 8BIT 1ARRAY
1B	27	64	16	128	ALT NOWIDEPIX 8BIT 1ARRAY
1F	31	128	32	256	NOALT NOWIDEPIX 8BIT 1ARRAY
23	35	128	37	256	ALT WIDEPIX 7BIT 2ARRAY
27	39	256	73	512	NOALT WIDEPIX 7BIT 2ARRAY
2B	43	128	19	128	ALT NOWIDEPIX 7BIT 2ARRAY
2F	47	256	37	256	NOALT NOWIDEPIX 7BIT 2ARRAY
33	51	128	32	256	ALT WIDEPIX 8BIT 2ARRAY
37	55	256	64	512	NOALT WIDEPIX 8BIT 2ARRAY
3B	59	128	16	128	ALT NOWIDEPIX 8BIT 2ARRAY
3F	63	256	32	256	NOALT NOWIDEPIX 8BIT 2ARRAY

7.5 RECOMMENDED MICRONEYE COMMAND SEQUENCES

When writing your own software to control the MicronEye, it is very important that the proper sequence of commands is used to control the MicronEye.

7.5.1 INITIALIZATION

The MicronEye should be initialized once after powerup as well as any time a change is made from ALTBIT mode to NOTALTBIT mode. As an example, lets assume that you want to receive pictures using the modes ALTBIT, NARROWPIX, 8BIT and 1ARRAY. Looking at the table for version A systems in the previous section we see that the basic command byte will be 220 (DC hexadecimal). Depending on how we control the SEND/NOSEND bit and the REFRESH/SOAK bit, the command byte we send to the MicronEye will be a value between 220 and 223. Sending the basic command byte to the MicronEye causes the MicronEye to SEND and REFRESH. Command byte plus 1 means REFRESH and NOSEND. Command byte plus 2 means SEND and SOAK. Command byte plus 3 means SOAK and NOSEND.

To initialize the MicronEye go through the following sequence. (Throughout the discussion that follows assume 220 to be the basic command byte.

(1) Tell the MicronEye to SEND and REFRESH (220).

(2) Receive bytes from the MicronEye until a timeout situation occurs. (Don't just count the bytes!) The software should determine that a timeout has occurred when 3 or more character periods have elapsed without receiving a byte from the MicronEye. A character period is defined as 10 / baud rate. For example, the character period at 9600 baud is 10/9600 of a second or 1.04 milliseconds.

(3) Once the MicronEye has stopped sending bytes, tell the MicronEye to REFRESH with NOSEND (221). It is necessary to send this byte as soon as the MicronEye has stopped sending because the MicronEye automatically goes into a SOAK/NOSEND state whenever it completes the transmission of a picture. The REFRESH/NOSEND command defeats the default SOAK/NOSEND state.

This completes the initialization sequence and does not need to be repeated unless you change from ALTBIT to NOALTBIT mode.

7.5.2 GETTING PICTURES FROM THE MICRONEYE

Use the following sequence of commands to get a picture from the MicronEye:

(1) Tell the MicronEye to SOAK with NOSEND (223).

(2) Wait for the desired exposure time. In normal room light with the lens aperature set to 1.6, the exposure time should normally be somewhere between 250 and 400 milliseconds.

(3) Tell the MicronEye to SEND and REFRESH (220).

(4) Receive bytes from the MicronEye until a timeout occurs.

(5) Once the MicronEye has stopped sending bytes, tell the MicronEye to REFRESH with NOSEND (221).

(6) Do any necessary processing and/or enhancing of the image. Repeat from step 1 to take another picture.

APPENDIX A

BAUD RATE MODIFIFICATION

The MicronEye's transmission speed (baud rate) is normally set at the factory to 153,000 baud for the IBM, Apple and Commodore 64 computers. A baud rate of 76,800 is used for the TRS-80 Color Computer. The RS-232 version is factory set to 9,600 baud. If you wish to change the baud rate, proceed according to the following paragraph.

Modifying the baud rate of the MicronEye requires some soldering so caution is advised. The baud rate on the MicronEye is set by soldering two wire jumpers as specified below. An end of one wire has been soldered to pad 9 (located beside IC D1) and one end of the other wire has been soldered to pad 10 (located between IC D5 and IC E5). These two connections are never changed so they do not have to be removed. However, the other end of both of the wires should be unsoldered and resoldered according to the desired baud rate.

Beside IC B5 are 8 pads. Pads 1, 4, and 8 are labeled on the board. To select one of the five standard baud rates, use the table below to rejumper for the desired baud rate.

TO SELECT THIS BAUD RATE	CONNECT THE WIRE FROM PAD 9 TO	CONNECT THE WIRE FROM PAD 10 TO
153,600	PAD 5	PAD 1
76,800	PAD 6	PAD 2
9,600	PAD 7	PAD 3
4,800	PAD 4	PAD 6
300	PAD 8	PAD 4

BAUD RATE MODIFICATION

Other baud rates are obtainable by soldering the wires to certain pins on IC B5. However, this practice does not make a reliable connection and is not recommended. If one of the baud rates listed below is required, use the following table to make the proper connections:

TO SELECT THIS BAUD RATE	CONNECT PAD 9 TO IC B5	CONNECT PAD 10 TO IC B5
38,400	PIN 3	PIN 9
19,200	PIN 2	PIN 7
2,400	PIN 12	PIN 3
1,200	PIN 14	PIN 2
600	PIN 15	PIN 4

APPENDIX B

TRANSMISSION TIME CONSIDERATIONS

The following table outlines the milliseconds required to send an image from the MicronEye to the computer as a function of rows, bytes per row and baud rate. The table may prove useful in doing exposure time calculations. The times are calculated using the following equation:

TIME = (ROWS x BYTES-PER-ROW x 10000) / BAUD-RATE

TRANSMISSION RATE TABLE

The following table is provided as an aid to the programmer by listing all row and column combinations (excluding send and soak bits).

ROWS	BYTES PER ROW	BAUD RATE 300	9600	19200	76800	153600
64	16	34133	1067	533	133	67
64	19	40533	1267	633	158	79
64	32	68267	2133	1067	267	133
64	37	78933	2467	1233	308	154
128	16	68266	2133	1067	267	133
128	19	81066	2533	1267	317	158
128	32	136533	4255	2133	533	267
128	37	157867	4933	2467	617	308
128	64	273067	8533	4267	1067	533
128	73	311466	9733	4867	1217	608
256	32	273067	8533	4267	1067	533
256	37	315733	9867	4933	1233	617
256	64	546133	17066	8533	2133	1066
256	73	622933	19466	9733	2433	1217

APPENDIX C

TROUBLESHOOTING

If you have problems with your MicronEye there is a good chance that the problem is setup-related. If you encounter a problem with your MicronEye, run through this checklist of common setup problems to verify that your MicronEye has been setup properly:

1. Verify that the card is plugged in properly. Symptoms of this problem include a peppered pattern on the screen, or an all white screen that doesn't go away even when you cover the lens, or the computer just 'hanging' when it attempts to send a command or receive an image.

2. Make certain that the lens cap is off and that the aperature setting is not set to 'C' (closed).

3. If the display appears to be all black, set the f-stop to the lowest setting and aim the MicronEye at a light source. If any of the screen turns white then the problem may be exposure time related.

4. If the display is all white, try setting the f-stop up or reducing the exposure time.

5. Try turning off the computer, brush off the interface card with a soft brush, clean the fingers on the card with propanol alcohol and a cotton swab, reinstall the card, and power up the computer.

6. Make certain that the cable connecting the MicronEye and the interface card is firmly in place.

7. If you are still unable to make the MicronEye operate properly, please contact us at Micron:

   MICRON TECHNOLOGY, INC.
VISION SYSTEMS GROUP
2805 E. COLUMBIA RD.
BOISE, ID 83706

   TEL. (208) 383-4046
TWX 910-970-5973

APPENDIX D

IS32 OPTICRAM TECHNICAL INFORMATION

D.1 OPERATION

The heart of the MicronEye is the IS32 OpticRAM, developed and manufactured by Micron Technology, Inc. The integrated circuit is Micron's 64K Dynamic RAM assembled in a standard 16 pin ceramic DIP package with a clear glass lid. The IS32 is composed of 65,536 individual image sensing elements called pixels. These pixels are organized into two arrays of 128 rows and 256 columns. (Typical applications will utilize only one of the sensor arrays since the arrays are separated by an optical non-light sensing zone of amplifiers). Each of the elements in the IS32 is a light sensitive capacitor which can be accessed randomly by simply strobing in the appropriate row and column address of the particular element to be accessed.

The device operates by focusing the reflected light from an object onto the 32,768 light sensitive elements of the array. Light striking a particular element will cause the capacitor, which is initially precharged to a fixed voltage, to discharge toward zero volts. The capacitor will discharge at a rate proportional to both the intensity and duration it is exposed to light.

To determine if a particular element is black or white, the user would read the appropriate row and column address associated with the physical location of that particular element. The IS32 would read the voltage value of the capacitor and perform a digital comparison between the voltage of the capacitor and the fixed threshold voltage. The output pin of the IS32 would be set to a logic level of 1 if the voltage on the capacitor was above the threshold point. It would set the output to a logic level of 0 if it was below the threshold voltage.

The logic level of 0 will be associated with a white pixel. A logic level of 1 will be associated with a black pixel. A white pixel indicates the capacitor was exposed to a light intensity sufficient to discharge the capacitor past the threshold point. A black pixel indicates the light intensity was not enough to discharge the capacitor past the threshold, therefore it retained the charge and is read as a logic 1.

The other significant factor affecting the discharge of the light sensitive capacitors is the length of the time which the capacitors are exposed to light. This period of time is measured from the initial exposure of an element until the time the particular element is read or refreshed.

The combination of the light intensity and the scan rate (the amount of time the elements are exposed before being read) will determine the optimum imaging environment. The faster the elements are scanned, or read, the greater the light intensity is required.

Perhaps the most important consideration the user must keep in mind is that the MicronEye requires a high contrast scene in order to image the object onto the IS32. Unlike a TV camera which can respond to "shades of gray," the IS32 is a digital chip where each picture element will only respond to a dark/light (1/0) binary part of the scene around an arbitrary amount of light used as a threshold. Shades of gray can be achieved by averaging multiple scans together using either a different threshold voltage or varying the scan rate. By changing the threshold voltage, keeping both the image and light intensity constant, the outputs produced during each scan will not change where pixels are definitely black or white. Change will be exhibited where the image is gray and the amount of reflected light striking the capacitors is near the threshold voltage. If an area of the image is a dark shade of gray, the output will generate more logic level 1's than logic level 0's. Where the image is a lighter shade of gray the ouput will generate more logic 0's than logic 1's. By averaging these outputs over a number of scans, the appropriate shade of gray is produced.

The nominal threshold with pin 1 open is 2.1 volts. This threshold can be adjusted via pin 1 from 1.5 volts to 3.0 volts. It is suggested that gray scale capability be achieved by varing the scan rate rather than adjusting the threshold voltage. By varying the scan rate (varying the discharge time) you can more accurately achieve gray scale capability.

If for any reason you must remove the IS32 from its socket, caution is imperative. The IS32 is susceptible to static and can be damaged by static electricity. Removal of the IS32 from the Bullet may require that the tips of the chip extractor tool be bent out slightly to accomodate the narrowness of the Bullet housing. When reinserting an IS32 into the socket, be certain it is properly oriented. For the Bullet, the IS32 is oriented properly when the red edge of the ribbon cable is on the same side of the Bullet as the Pin 1 notch on the IS32. For the Camera, the IS32 is oriented properly when the Pin 1 notch on the OpticRAM is on the same edge as the Pin 1 notch on the other IC's in the camera.

D.2 IS32 TECHNICAL SPECIFICATIONS

There are two versions of the IS32 OpticRAM: the IS32 and IS32A. Beginning in September 1983, the IS32 was replaced in favor of the IS32A. The only difference between the two devices is size. The IS32A is exactly 20 percent smaller in the horizontal and vertical dimensions. The dimensions below are for the IS32A. To calculate dimensions for the larger IS32 device, multiply by 1.25.

D.2.1 DIMENSIONS

1. ARRAY: 128 x 256 electrical addressable elements per array (4420 microns x 876.8 microns). The physical organization of the array is actually a 514 x 129 grid with staggered cell placement as indicated in figure D-1.

2. ROW: 877 microns.

3. COLUMN: 4420 microns.

4. ELEMENT SIZE: 6.4 microns vertical by 6.4 microns horizontal.

5. VERTICAL PITCH (Row Pitch): 6.8 microns.

6. HORIZONTAL PITCH (Column Pitch): 8.6 microns.

7. SPACING between left and right array: 120 microns.

8. DISTANCE from surface of OpticRAM chip to top of the glass = 940 microns (plus or minus 100 microns).

D.2.2 SENSITIVITY

Broad band sensitivity of the IS32 OpticRAM is approximately 2uJ/sq cm.

Silicon detectors have a useful optical sensitivity over the region of the spectrum is which silicon absorbs photons. This extends from 200 nanometers to 1100 nanometers. However, a complete characterization of the IS32 is still under way. The sensitivity follows the silicon characteristic curve since the IS32 is built using silicon. The IS32 is impervious to damage by high light intensity. It has a high quantum efficiency and a binary output that is proportional to the amount of incident light and integration time (referenced to a threshold). However, oversaturation of the IS32 by more than 4 F-stops will, for the duration of oversaturation, make the first half of the array all light and the other half all dark. This is only a temporary situation for the duration of the saturation. The IS32 is sensitive up to near UV.

The IS32 chip is mounted in the package with 20 mils tolerance in both the X and Y axis. This suggests that if an OpticRAM package is replaced in a camera, a physical realignment of the camera to the scene is necessary. The tolerance from surface of the array to the lens mount from camera to camera is 20 mils with a 6 degree rotational tolerance.

D.3 TOPOLOGY

D.3.1 Address Descramble

If you access a cell (pixel) in the OpticRAM using an address of zero for both the row and column, the Optic RAM will not physically select Row 0 and Column 0. This is because the internal address decoding does not provide a one-to-one correspondence between the address count and the physical row and column. A simple circuit, consisting of exclusive OR's and inverters, performs the necessary code conversion to achieve the desired one-to-one correspondence. See Figure D-1

D.3.2 Pixel Layout

One of the primary goals in designing a low cost integrated circuit such as the OpticRAM, is to minimize its physical size. To achieve this goal, the cells in the OpticRAM are arranged in an interleaved pattern. If an image is read out of the OpticRAM by counting successively down the rows and columns, the image will look "fuzzy" around the edges because the pixels will be slightly misplaced in the graphics matrix.

To accomodate the pixel misplacement, the data from the OpticRAM must be mapped into the graphics matrix so that the arrangement of the pixels in the graphic matrix matches the physical arrangement of the cells in the OpticRAM. Due to the interleaved cell pattern on the OpticRAM, the array is much longer than it is wide, resulting is spaces between the cells in the column direction. Because of the spaces, the 128 X 256 array of cells will map very nicely into a 129 X 514 matrix. We will call this matrix the Cell Placement Grid.

D.3.3 Cell Placement Grid

The cell placement grid is shown on page D-6. For a single array, there are a total of 129 rows and 514 columns. Only the corners of the array are shown. The placement grid indicates where the information from each cell in the OpticRAM should be mapped. For instance, if the cell at address Row 1, Column 1, in the OpticRAM is read, the value (a 1 or 0) should be placed in the placement grid at location X=2, Y=3.

When every cell has been read and the values placed in the appropriate locations, about half of the grid remains empty. We will call these empty locations "space pixels." The space pixels can be set all high or all low to provide a light or dark background for the image. Another alternative is to set each space pixel to the level that agrees with the majority of its nearest neighbors. For example, let's say the pixel at grid locations X=2,Y=2 (R1 C1) and X=3,Y=1 (R1 C0) are high, and the pixel at grid location X=3,Y=3 (R3 C0) is low. These are the three nearest neighbors of grid location X=3,Y=2. The majority of these nearest neighbors is high, so the previously empty grid location X=3,Y=2 is set high also. This technique can be applied to all empty grid locations except those near the edge of the array. A modified technique can be used for these edge space pixels, although there is less optical data to work with. Another alternative is to simply not use the edge rows and columns.

Having the cells laid out in the IS32 the way they are, gives the IS32 much greater resolving power than if the cells were laid out linearly.

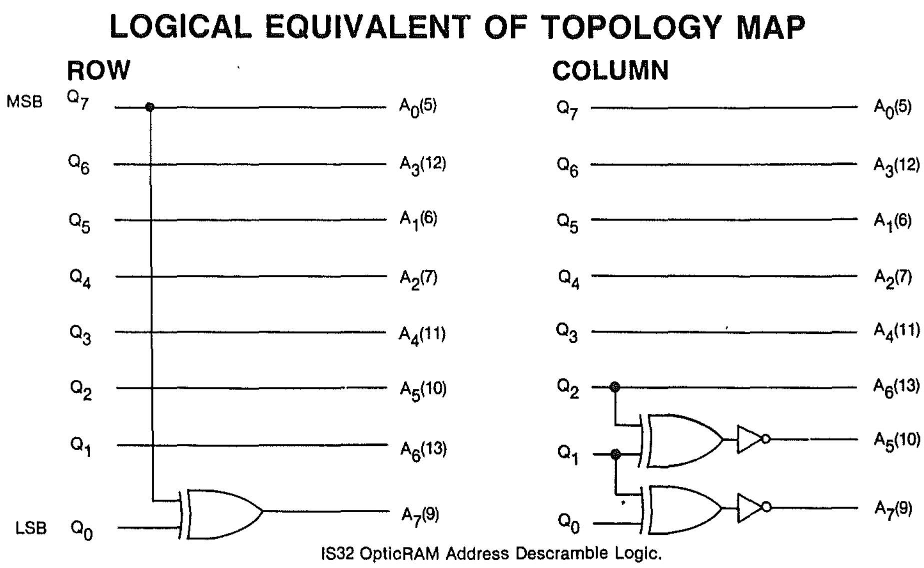
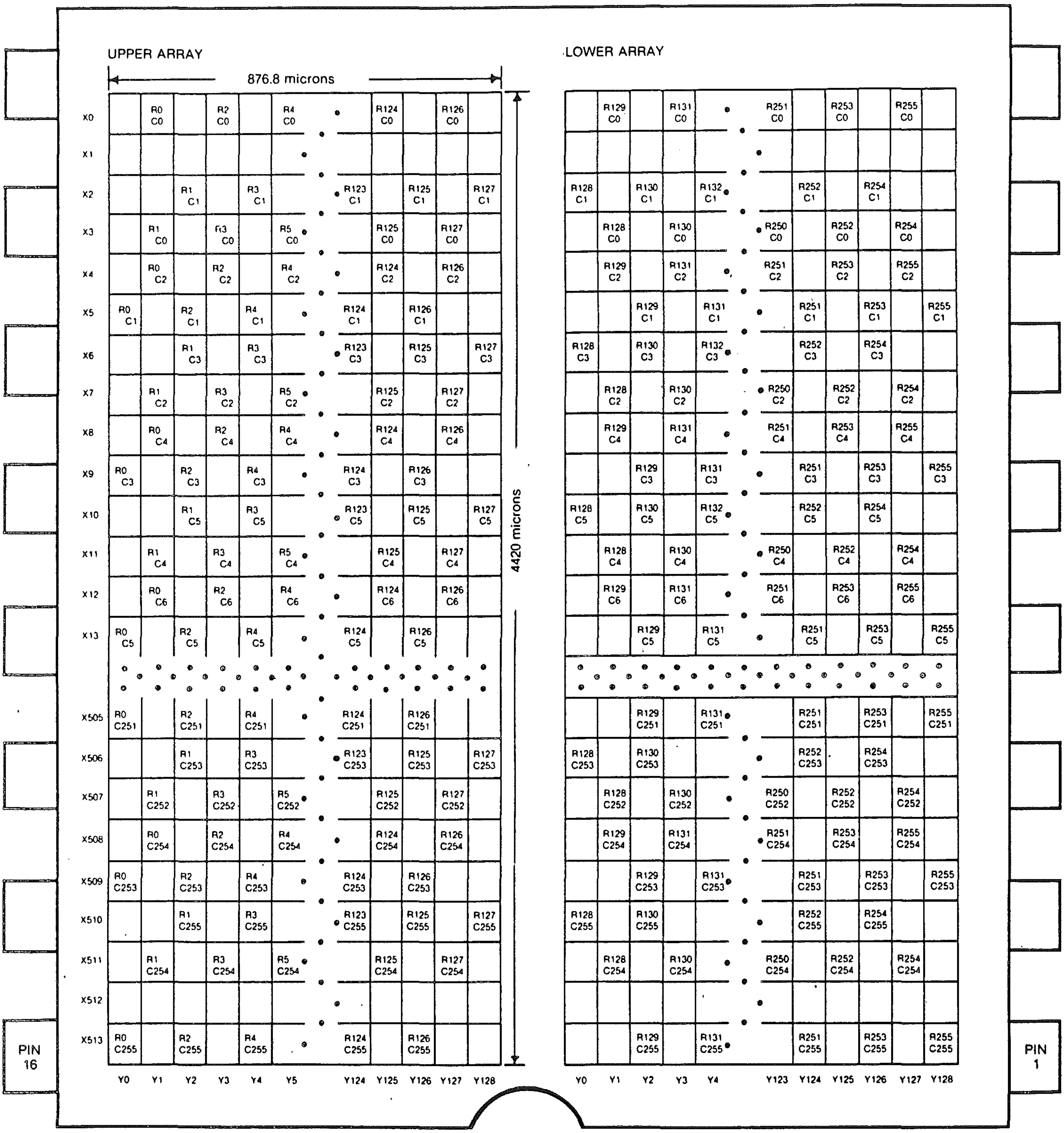
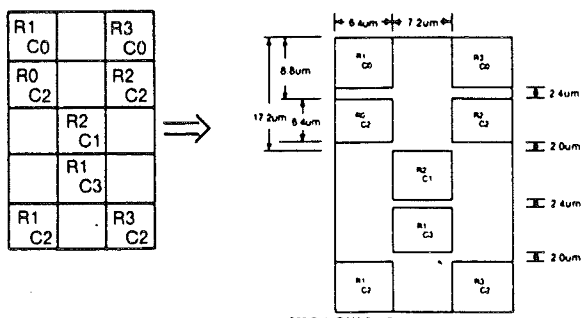


Figure D-1.

IS32 OpticRAM™ Topological Information





IS32 OpticRAM Cell Spacing

MICRON TECHNOLOGY, INC

ALGORITHM FOR TRANSFORMING THE OPTICRAM 256 X 128 IMAGE INTO THE PROPER 512 X 128 ARRAY SPACE.

(* described in psuedo-PASCAL terms. For purposes of simplicity we assume that bits are individually accessible. Implementation on most computers will require bit-twiddling to simulate the effect of what is shown below. This algorithm is accurate for the lower array which is the array used when ONE-ARRAY mode is used on the MicronEye. The changes to be used when transforming the upper array are noted parenthetically. *)

```
VARIABLE DECLARATIONS
grid       : ARRAY [0..513,0..128] OF bits;
optic_ram  : ARRAY [0..255,0.127] OF bits;
x,x3,
y,
col_ctr,
row_ctr    : INTEGER ;

PROCEDURE TRANSFORM;
BEGIN
FOR row_ctr := 0 TO 127 DO
    FOR col_ctr := 0 TO 255 DO
        BEGIN
        IF ODD(row_ctr) THEN
            IF ODD(col_ctr) THEN
                BEGIN
                y := row_ctr + 1;
                x := 2 * col_ctr + 3; (* upper array is 2*col_ctr*)
                END
            ELSE
                BEGIN
                y := row_ctr;
                x := 2 * col_ctr;     (* upper array is 2*col_ctr + 3*)
                END
        ELSE (*even row*)
            IF ODD(col_ctr) THEN
                BEGIN
                y := row_ctr;
                x := 2 * col_ctr;     (* upper array is 2*col_ctr + 3*)
                END
            ELSE (*even column*)
                BEGIN
                y := row_ctr + 1;
                x := 2 * col_ctr + 3; (* upper array is 2*col_ctr*)
                END;
        grid[x,y] := optic_ram[col_ctr,row_ctr];
        END;
END; (* end of transform procedure *)
```

APPENDIX E

ANNOTATED ASSEMBLY LANGUAGE DRIVER FOR THE IBM PC

```
TITLE   MICRONEYE ASSEMBLER ROUTINES (MEYEDRVR)
PAGE    84,132
.SALL
COMMENT !
                MEYEDRVR -- MODULE DESCRIPTION

        This driver module takes care of all the necessary arrangements
for getting an image from the MicronEye to the graphics page in memory
of the IBM PC.  The routine is assembled to be relocatable.
Because the routine normally resides within the BASIC workspace, the first
part of the BASIC program locates these routines as high in the BASIC
segment as possible. Although the program makes certain that there is
enough room initially for both the assembly language routines and the
BASIC program, there is no assurance that the declaration of large amounts
of data space will not overlap the machine language programs.

        There are 4 assembly routines available to the user-- PARMCALC,
SCREENDUMP, XFERSCR and GETPIC.  PARMCALC takes the setup parameters
selected at the main menu and calculates the commands that GETCALC will send
the MicronEye and the number of bytes the MicronEye will return.  SCREENDUMP
dumps the current picture to an IBM or Epson printer.  XFERSCR moves the
picture in the WORKMAP area to the screen. The proper calling
sequence from a BASIC program is as follows:

        DEF SEG=&Hxxxx  'where xxxx specifies the assembler routine address
        BLOAD "MEYEDRVR",0
        GETPIC     = 0
        PARMCALC   = 6
        SCRDUMP    = 12
        SCRXFER    = 16
        DATA.AREA  = 20
           .
           .
           .
        CALL PARMCALC(pic_type,pics_per_screen,exit_at_eof,expose_time)
           .
        CALL SCREENDUMP
           .
        CALL GETPIC(screen_start,white_pct,key_value)
           .
        CALL XFERSCR

        All variables in the argument list are assumed to be of type
INTEGER.  The variables are defined as follows:

        PIC_TYPE -- determines the format of the image transmitted from
                the MicronEye.  The following are valid commands:

                        0 - 128 x  64 picture (black & white)
                        1 - 512 x  64 picture (w/ smoothing)
                        2 - 512 x  64 picture (grey)
                        3 - 512 x 128 picture (black & white)
                        4 - 640 x 128 picture (w/ smoothing)
                        5 - 640 x 128 picture (grey)

                The command byte sent this routine is not the same as the
                control byte sent to the MicronEye.  For a complete
                description of the MicronEye control byte refer to the
                Operator's manual.

        PICS_PER_SCREEN -- If set to 2 then both blocks of image sensors
                on the OpticRAM will be transmitted from the MicronEye.
                The blocks (or arrays) are separated by a dead zone so
                the displayed picture will appear to be split-screen.

        EXIT_AT_EOF -- If true (an odd number) then control is returned to
                the calling program at the end of each picture transmission.
                Otherwise, pictures are continually processed until a key is
                pressed.

        EXPOSE_TIME -- the number of milliseconds for which the image should
                be exposed.

        SCREEN_START -- The byte position on the screen page at which the
                picture should start.  This position is calculated as:
                        (ROW*80) + (COLUMN/8).
                Row must be an even number between 0 and 134. Column must
                be between 0 and 512 and divisible by 8.
```

WHITE_PCT -- A value between 0 and 100 that indicates the approximate percent of the current picture that is white.

KEY_VALUE -- At the beginning of each FRAMEGRAB this variable is set to zero. If during the FRAMEGRAB a key is pressed, then the ASCII value of the key is placed in the LSB of key_value.

For custom applications you may wish to increase or decrease the size of the buffers used to create pictures of each of the supported types. The table below shows the byte requirements for the 3 buffers BITMAP, BITMAPB, and WORKMAP when using each picture type. As shipped, WORKMAP is set to 10240 bytes, BITMAP 10240 bytes and BITMAPB 4096 bytes. With such a configuration the BASIC program MEYE allows single array pcitures for all 6 picture types, but supports 2 array mode for only the first three picture types (64 row pictures).

BUFFER SIZE REQUIREMENTS FOR EACH PICTURE TYPE

PICTURE TYPE	BUFFER NAME	BYTES WHEN USING 1 ARRAY	BYTES WHEN USING 2 ARRAYS
128 x 64 (B/W)	WORKMAP	1024	2048
512 x 64 (B/W)	BITMAP	1024	2048
	WORKMAP	4096	8192
512 x 64 (GREY)	BITMAP	1024	2048
	BITMAPB	1024	2048
	WORKMAP	4096	8192
512 x 128 (B/W)	BITMAP	4096	*
	WORKMAP	8192	*
640 x 128 (B/W)	BITMAP	4096	*
	WORKMAP	10240	*
640 x 128 (GREY)	BITMAP	10240	*
	BITMAPB	4096	*
	WORKMAP	10240	*

* 2-array mode is not supported by this software when working with the x128 pictures. Firstly, buffer space required was prohibitive for users with less than 192K of memory. Secondly, a separate set of routines to handle the enhancement of the top array (only the bottom array is used in 1-array mode) would be required because pixel placement is a mirror image of what is done in the bottom array. In other words, in the top array on even rows you would do what was done on odd rows in the bottom array except that the bytes that are moved one row away stay where they are and the bytes that normally stay where they are move one row away.

NOTE: RS-232 users who are referring to this code to develop code for other computers should be aware that the data lines for the IBM interface have been flipped to accomodate the IBM PC's method of graphics display. All status, control, and data registers associated with the MicronEye interface on the IBM version are exactly backwards of the status, control, and data registers employed by the RS-232 interface. In other words, when the IBM writes a C0 hex (11000000 binary) to the control register, RS-232 MicronEye users will want to write a 03 hex (00000011 binary).

By the same token data received from the MicronEye is backwards. In the IBM the most significant bit of each byte received corresponds to the leftmost of the eight pixels in the image. In the RS-232 the least significant bit of each byte corresponds to the leftmost pixel. The Apple II and Commodore 64 versions of the MicronEye are similar in this respect to the RS-232 version. The TRS-80 Color Computer interface to the MicronEye uses the same bit orientation as the IBM PC.

SUBTTL MAIN SUBROUTINES AND DATA DECLARATIONS

```
                                        PAGE
0000                            CSEG    SEGMENT PARA PUBLIC
                                        ASSUME  CS:CSEG,DS:CSEG,ES:NOTHING
                                ;
0000                            GETPICT PROC    FAR             ;get picture(s) from the MicronEye
0000  E9 61EC R                         JMP     GETPIC
0003  CA 0006                   GPIC_X: RET     6
0006                            GETPICT ENDP
                                ;
0006                            PARMCLC PROC    FAR             ;calculate parameters based on pictype,
0006  E9 626E R                         JMP     PCALC           ;  pics_per_screen,exit_at_eof, and
0009  CA 0008                   PARM_X: RET     8               ;  expose_time.
000C                            PARMCLC ENDP
                                ;
000C                            PICDUMP PROC    FAR             ;prints picture on screen. uses bitmap
000C  E8 6303 R                         CALL    SCRDUMP         ;  image rather than screen as source.
000F  CB                                RET
0010                            PICDUMP ENDP
                                ;
0010                            SCRXFER PROC    FAR             ;moves bitmap picture at WORKMAP to screen
0010  E8 638E R                         CALL    MOVESCR
0013  CB                                RET
0014                            SCRXFER ENDP
                                ;
0014  0400                      MAPBYTES        DW      1024
0016  0000                      KEY_VALUE       DW      0
0018  0000                      PICTYPE         DW      0
001A  0000                      SCREEN_START    DW      0
001C  0040                      SCR_ROWCT       DW      64
001E  0010                      SCR_COLCT       DW      16
0020  0001                      ARRAYCT         DW      1
0022  28A0 [                    WORKMAP         DB      10400 DUP (?)   ;use 16000 for 2-array pictypes 3-5
               ??
                    ]

28C2  28A0 [                    BITMAP          DB      10400 DUP (?)   ;use 16000 for 2-array pictype 5
               ??
                    ]

5162  1000 [                    BITMAPB         DB      4096  DUP (?)   ;use 8192 for 2-array pictype 2 & 5
               ??
                    ]

                                ;
6162  0318                      CONTROL         DW      318H            ;camera control port
6164  0318                      STATUS          DW      318H            ;camera status port
6166  0319                      DATAIN          DW      319H            ;camera data-from port
6168  0319                      DATAOUT         DW      319H            ;camera data-to port
616A  0000                      EXITEOF         DW      0
616C  012C                      EXPOSE_TIME     DW      300
616E  0000                      WHITE_PCT       DW      0
6170  1B                        COMMAND         DB      1BH
6171  ????                      ROWCTR          DW      ?
6173  ????                      COLCTR          DW      ?
6175  00                        BITCT           DB      0
6176  0000                      WORDCT          DW      0
6178  ????                      WHITECT         DW      ?
617A  0400                      TOTBYTES        DW      1024
                                ;
617C  0400 0400 0400 1000       TOT_TAB         DW      1024,1024,1024,4096,4096,4096
      1000 1000
6188  0040 0040 0040 0080       ROWCT_TAB       DW      64,64,64,128,128,128
      0080 0080
6194  0010 0040 0040 0040       COLCT_TAB       DW      16,64,64,64,80,80
      0050 0050
61A0  0400 1000 1000 2000       BYTES_TAB       DW      1024,4096,4096,8192,10240,10240
      2800 2800
61AC  001B 001B 001B 001F       CMD_TAB         DW      1BH,1BH,1BH,1FH,1FH,1FH
      001F 001F
                                ;
61B8  0000                      PAT             DW      0
61BA  00 05 0F 00 50 55         DPAT            DB      0,5,0FH,0,50H,55H,5FH,0,0F0H,0F5H,0FFH
      5F 00 F0 F5 FF
61C5  00 0F F0 FF               CPAT            DB      0,0FH,0F0H,0FFH
61C9  1B 41 08 0D FF            GRSETUP         DB      1BH,41H,8H,0DH,0FFH     ; 8/72" line spacing
61CE  1B 32 0D FF               TEXTSET         DB      1BH,32H,0DH,0FFH        ; 6 lines/in spacing
61D2  0D 0A 1B 4B               GLIN            DB      0DH,0AH,1BH,4BH         ; 60 dots/in graphics
61D6  0000 FFFF                 GRLEN           DW      0H,0FFFFH
61DA  6246 R 6247 R 624A R      PIC             DW      PICA,PICB,PICC,PICD,PICE,PICF
```

```
      6253 R 6256 R 625C R
61E6  673A R 674A R           SOAKPTR      DW      SOAK,GREYSOAK
61EA  ????                    MAPADR       DW      ?
                              ;
                                      SUBTTL  GETPIC ROUTINE  ;get picture from the MicronEye
```



```
                                      PAGE    +
61EC                          GETPIC  PROC    NEAR            ;does everything necessary to get picture
61EC  55                              PUSH    BP              ;preserve stack ptr in BP
61ED  8B EC                           MOV     BP,SP
61EF  1E                              PUSH    DS              ;save calling environment
61F0  06                              PUSH    ES
61F1  8C D8                           MOV     AX,DS           ;set up ES to point to data segment of
61F3  8E C0                           MOV     ES,AX           ;  calling program.
61F5  8C C8                           MOV     AX,CS           ;set up DS to be the same as the code segment
61F7  8E D8                           MOV     DS,AX
61F9  8B 7E 0A                        MOV     DI,10[BP]       ;get SCREEN_START address from stack
61FC  26: 8B 05                       MOV     AX,ES:[DI]      ;get SCREEN_START value
61FF  A3 001A R                       MOV     SCREEN_START,AX
6202  E8 6715 R                       CALL    ACIACLR         ;initialize comm link
6205                          RESTART:
6205  E8 63D2 R                       CALL    FRAMEGRAB       ;get picture from MicronEye
6208  8B 1E 0018 R                    MOV     BX,PICTYPE      ;select picture formatting routine based
620C  D1 E3                           SHL     BX,1            ;  on pictype
620E  FF 97 61DA R                    CALL    WORD PTR PIC[BX]
6212  E8 638E R                       CALL    MOVESCR         ;moved formatted picture to screen
6215  F7 06 616A R 0001               TEST    EXITEOF,1       ;if exit_at_eof set then prepare to exit
621B  75 07                           JNZ     DONE
621D  83 3E 0016 R 00                 CMP     KEY_VALUE,0
6222  74 E1                           JE      RESTART         ;otherwise go get another picture
6224                          DONE:
6224  B8 0064                         MOV     AX,100          ;calc white_pct = 100 * white_ct / totbytes
6227  F7 26 6178 R                    MUL     WHITECT
622B  F7 36 617A R                    DIV     TOTBYTES
622F  A3 616E R                       MOV     WHITE_PCT,AX
6232  8B 1E 0016 R                    MOV     BX,KEY_VALUE
6236  8B 7E 06                        MOV     DI,6[BP]        ;get address of key_value variable
6239  8B 76 08                        MOV     SI,8[BP]        ;get address of white_pct variable in
623C  07                              POP     ES              ;  calling program
623D  1F                              POP     DS              ;restore environment before returning
623E  89 04                           MOV     [SI],AX         ;(note that DS of calling program is being used
6240  89 1D                           MOV     [DI],BX         ; to assign value to WHITE_PCT and KEY_VALUE
6242  5D                              POP     BP              ; of calling programs)
6243  E9 0003 R                       JMP     GPIC_X
                              ;
6246  C3                      PICA:   RET                     ;no formatting needed for 128 x 64 B&W picture
6247  E9 6474 R               PICB:   JMP     SMOOTHB         ;format routine for 512 x 64 smoothed picture
624A  E8 674A R               PICC:   CALL    GREYSOAK        ;format routine for 512 x 64 grey picture
624D  E8 63C9 R                       CALL    GREYGRAB
6250  E9 65BD R                       JMP     GREYADD
6253  E9 64AE R               PICD:   JMP     ENHANCE         ;format routine for 512 x 128 B&W picture
6256  E8 669B R               PICE:   CALL    ENHANCE2        ;format routine for 512 x 128 smoothed picture
6259  E9 6609 R                       JMP     FILLIN2
625C  E8 674A R               PICF:   CALL    GREYSOAK        ;format routine for 512 x 128 grey picture
625F  E8 63C9 R                       CALL    GREYGRAB
6262  E8 669B R                       CALL    ENHANCE2
6265  E8 668F R                       CALL    ENHANCEG
6268  E8 66F3 R                       CALL    GREYADD2
626B  E9 6609 R                       JMP     FILLIN2
626E                          GETPIC  ENDP
                              ;
                                      SUBTTL  PARMCALC -- Determine picture characteristics
```

```
                                               PAGE
                                        ;
626E                                    PCALC   PROC    NEAR
626E  55                                        PUSH    BP                  ;save environment
626F  8B EC                                     MOV     BP,SP
6271  1E                                        PUSH    DS
6272  06                                        PUSH    ES
6273  8C D8                                     MOV     AX,DS               ;set up ES to point to calling routine's
6275  8E C0                                     MOV     ES,AX               ;  data segment
6277  8C C8                                     MOV     AX,CS               ;set up DS to be the same as this routine's
6279  8E D8                                     MOV     DS,AX               ;  code segment
627B  8B 7E 0C                                  MOV     DI,12[BP]           ;get pictype off stack
627E  26: 8B 1D                                 MOV     BX,ES:[DI]
6281  89 1E 0018 R                              MOV     PICTYPE,BX
6285  8B 7E 0A                                  MOV     DI,10[BP]           ;get arrayct off stack
6288  26: 8B 05                                 MOV     AX,ES:[DI]
628B  A3 0020 R                                 MOV     ARRAYCT,AX
628E  83 3E 0018 R 02                           CMP     PICTYPE,2           ;if we have chosen pictypes 3-5 then only
6293  7E 06                                     JLE     NOTWO               ;  allow 1 array mode
6295  C7 06 0020 R 0001                         MOV     ARRAYCT,1
629B  8B 7E 08                          notwo:  MOV     DI,8[BP]            ;get exiteof off stack
629E  26: 8B 05                                 MOV     AX,ES:[DI]
62A1  A3 616A R                                 MOV     EXITEOF,AX
62A4  8B 7E 06                                  MOV     DI,6[BP]            ;get exposetime off stack
62A7  26: 8B 05                                 MOV     AX,ES:[DI]
62AA  A3 616C R                                 MOV     EXPOSE_TIME,AX
62AD  D1 E3                                     SHL     BX,1                ;word-adjust BX for table lookups
62AF  8B 87 6194 R                              MOV     AX,COLCT_TAB[BX]
62B3  A3 001E R                                 MOV     SCR_COLCT,AX        ;get rowct for this pictype
62B6  8B 87 61AC R                              MOV     AX,CMD_TAB[BX]
62BA  A2 6170 R                                 MOV     COMMAND,AL          ;get MicronEye command byte for this pictype
62BD  8B 87 6188 R                              MOV     AX,ROWCT_TAB[BX];get column bytes ct for this pictype
62C1  8B 8F 61A0 R                              MOV     CX,BYTES_TAB[BX];get clipped pic size in bytes for this pictype
62C5  8B 97 617C R                              MOV     DX,TOT_TAB[BX]      ;get total bytes for this pictype
62C9  8B DA                                     MOV     BX,DX
62CB  8B 16 0020 R                              MOV     DX,ARRAYCT
62CF  4A                                        DEC     DX
62D0  81 E2 0001                                AND     DX,1
62D4  74 1C                                     JZ      NO2ARRAY            ;if arrayct = 2 then
62D6  80 06 6170 R 20                           ADD     COMMAND,20H         ;  alter command byte for 2 arrays
62DB  D1 E3                                     SHL     BX,1                ;  double total bytes
62DD  D1 E0                                     SHL     AX,1                ;  double row ct
62DF  D1 E1                                     SHL     CX,1                ;  double clipped pic size
62E1  3D 00C8                                   CMP     AX,200              ;  max rowct is 200
62E4  7E 03                                     JLE     ROWOK
62E6  B8 00C8                                   MOV     AX,200
62E9  81 F9 3E80                        rowok:  CMP     CX,16000            ;  clipped pic size max is 16000
62ED  7E 03                                     JLE     NO2ARRAY
62EF  B9 3E80                                   MOV     CX,16000
62F2                                    NO2ARRAY:
62F2  A3 001C R                                 MOV     SCR_ROWCT,AX        ;transfer parameters from registers
62F5  89 0E 0014 R                              MOV     MAPBYTES,CX
62F9  89 1E 617A R                              MOV     TOTBYTES,BX
62FD  07                                        POP     ES                  ;restore environment
62FE  1F                                        POP     DS
62FF  5D                                        POP     BP
6300  E9 0009 R                                 JMP     PARM_X              ;return to PARMCALC shell (doing things this
6303                                    PCALC   ENDP                        ; forces the needed FAR return)
                                        ;
                                                SUBTTL  SCREENDUMP ROUTINE -- prints image in WORKMAP buffer
```

```
                                              PAGE
6303                                SCRDUMP PROC    NEAR
6303  1E                                    PUSH    DS                ;preserve calling environment
6304  8C C8                                 MOV     AX,CS             ;set up data segment pointer
6306  8E D8                                 MOV     DS,AX
6308  B9 61C9 R                             MOV     CX,OFFSET GRSETUP
630B  E8 637C R                             CALL    PUTSTR            ;set up line spacing, etc for graphics dump
630E  8B 16 001C R                          MOV     DX,SCR_ROWCT      ;  the length of each graphic bitstream is 2x
6312  D1 E2                                 SHL     DX,1              ;  the rowct.  This value is inserted at GRLEN
6314  89 16 61D6 R                          MOV     GRLEN,DX          ;  which is part of the GLIN string which puts
6318  A1 001E R                             MOV     AX,SCR_COLCT      ;  the printer into graphics mode
631B  A3 6173 R                             MOV     COLCTR,AX         ;setup COLCTR which also serves as current col
631E  8B 16 001C R                  nxtset: MOV     DX,SCR_ROWCT      ;reinitialize ROWCTR at the start of each col
6322  89 16 6171 R                          MOV     ROWCTR,DX
6326  FF 0E 6173 R                          DEC     COLCTR            ;point to the next col to print (last to first)
632A  B4 0B                                 MOV     AH,0BH            ;check for keypress
632C  CD 21                                 INT     21H
632E  3C 00                                 CMP     AL,0
6330  74 07                                 JE      NOCHR             ;if keypress then get char and exit
6332  B4 08                                 MOV     AH,8
6334  CD 21                                 INT     21H
6336  EB 3A 90                              JMP     EARLYX
6339  B9 61D2 R                     nochr:  MOV     CX,OFFSET GLIN    ;tell printer to receive the next GRLEN bytes
633C  E8 637C R                             CALL    PUTSTR            ;  as a graphics bitstream
633F  BE 0022 R                             MOV     SI,OFFSET WORKMAP
6342  03 36 6173 R                          ADD     SI,COLCTR         ;calculate location of the first byte for col
6346  8A 24                         nxtbyt: MOV     AH,[SI]           ;get the byte and complement it to avoid
6348  F6 D4                                 NOT     AH                ;  printing a negative of the picture
634A  B9 0008                               MOV     CX,8              ;next we need to invert the bits in the byte
634D  D0 E4                         flipit: SHL     AH,1              ;  because we are printing the picture from
634F  D0 DA                                 RCR     DL,1              ;  right to left rather than the normal left
6351  E2 FA                                 LOOP    FLIPIT            ;  to right
6353  E8 6377 R                             CALL    PUTBYTE           ;print the byte twice to obtain the proper
6356  E8 6377 R                             CALL    PUTBYTE           ;  aspect ratio
6359  03 36 001E R                          ADD     SI,SCR_COLCT      ;get the byte for the next row
635D  FF 0E 6171 R                          DEC     ROWCTR            ;repeat until done with column (actually
6361  75 E3                                 JNZ     NXTBYT            ;  printing 8 columns at a time)
6363  83 3E 6173 R 00                       CMP     COLCTR,0          ;go onto next row unless done with
6368  75 B4                                 JNE     NXTSET            ;  entire picture
636A  B9 61CE R                     tset:   MOV     CX,OFFSET TEXTSET
636D  E8 637C R                             CALL    PUTSTR            ;restore printer to normal operation
6370  1F                                    POP     DS                ;restore calling environment
6371  C3                                    RET
6372  E8 675A R                     earlyx: CALL    BEEP              ;we sound the bell to signal early end of
6375  EB F3                                 JMP     TSET              ;  printing caused by keypress
                                    ;
6377                                PUTBYTE:
6377  B4 05                                 MOV     AH,5              ;This routine sends the character
6379  CD 21                                 INT     21H               ;  in DL to the printer
637B  C3                                    RET
                                    ;
637C                                PUTSTR:
637C  87 CE                                 XCHG    CX,SI             ;This routine prints a string whose
637E  8A 14                         nput:   MOV     DL,[SI]           ;  starting address is in CX and is
6380  80 FA FF                              CMP     DL,0FFH           ;  terminated by 0FFH
6383  74 06                                 JE      ENDSTR
6385  E8 6377 R                             CALL    PUTBYTE
6388  46                                    INC     SI
6389  EB F3                                 JMP     NPUT
638B  8B F1                         endstr: MOV     SI,CX             ;Restore orginal value of SI
638D  C3                                    RET
638E                                SCRDUMP ENDP
                                    ;
                                            SUBTTL  MOVESCR -- Move picture to graphics screen
```

```
                                        PAGE    +
638E                            MOVESCR PROC    NEAR
638E  1E                                PUSH    DS              ;save calling environment just
638F  06                                PUSH    ES              ;  in case this call is coming
6390  8C C8                             MOV     AX,CS           ;  from BASIC
6392  8E D8                             MOV     DS,AX           ;set up proper data segment ptr
6394  BE 0022 R                         MOV     SI,OFFSET WORKMAP
6397  FC                                CLD                     ;make sure direction flag is forward
6398  8B 3E 001A R                      MOV     DI,SCREEN_START ;point destination register
639C  B8 B800                           MOV     AX,0B800H       ;  at desired offset on the
639F  8E C0                             MOV     ES,AX           ;  graphics screen
63A1  8B 16 001C R                      MOV     DX,SCR_ROWCT    ;DX is set to the number of rows/2
63A5  D1 EA                             SHR     DX,1            ;  because we handle two rows at a time
63A7  8B 1E 001E R                      MOV     BX,SCR_COLCT    ;BX is set to the number of bytes/row
63AB  8B CB                     nxtrow: MOV     CX,BX           ;CX is the number of words per row
63AD  D1 E9                             SHR     CX,1
63AF  F3/ A5                            REP     MOVSW           ;move the entire even row
63B1  2B FB                             SUB     DI,BX           ;set up for odd row which is 2000H away
63B3  81 C7 2000                        ADD     DI,2000H
63B7  8B CB                             MOV     CX,BX
63B9  D1 E9                             SHR     CX,1
63BB  F3/ A5                            REP     MOVSW           ;move the entire odd row
63BD  81 EF 1FB0                        SUB     DI,1FB0H        ;set up for next even row
63C1  2B FB                             SUB     DI,BX
63C3  4A                                DEC     DX              ;continue until we've done every row
63C4  75 E5                             JNZ     NXTROW
63C6  07                                POP     ES
63C7  1F                                POP     DS
63C8  C3                                RET
63C9                            MOVESCR ENDP
                                ;
                                        SUBTTL  FRAMEGRABBER ROUTINES -- gets image from MicronEye
```

```
                                    PAGE    +
                                    ASSUME ES:CSEG
63C9                        GREYGRAB PROC   NEAR
63C9  BF 5162 R                     MOV     DI,OFFSET BITMAPB
63CC  BB 0002                       MOV     BX,2            ;this is an alternate entry for the
63CF  EB 1A 90                      JMP     FG2             ; framegrab routine which sends the image
63D2                        GREYGRAB ENDP                   ; to the BITMAPB buffer and uses GREYSOAK
                            ;                               ; to make for a shorter than normal soaktime
63D2                        FRAMEGRAB PROC  NEAR
63D2  C7 06 0016 R 0000             MOV     KEY_VALUE,0     ;zero keyvalue
63D8  BB 0000                       MOV     BX,0
63DB  83 3E 0018 R 00               CMP     PICTYPE,0       ;all pictures but pictype 0 framegrab
63E0  75 06                         JNE     SETBMP          ; to the BITMAP buffer which grabs to
63E2  BF 0022 R                     MOV     DI,OFFSET WORKMAP
63E5  EB 04 90                      JMP     FG2             ; the WORKMAP buffer.  This allows the
63E8  BF 28C2 R             SETBMP: MOV     DI,OFFSET BITMAP; formatted picture to always end up in
63EB  8A 26 6170 R          fg2:    MOV     AH,COMMAND      ;tell MicronEye to soak w/o send
63EF  89 3E 61EA R                  MOV     MAPADR,DI
63F3  80 CC C0                      OR      AH,0C0H
63F6  E8 6720 R                     CALL    SENDCMD
63F9  FF 97 61E6 R                  CALL    WORD PTR SOAKPTR[BX]    ;soak for specified expose time
63FD  8B DF                         MOV     BX,DI           ;save start address of buffer for compare
63FF  E8 6464 R                     CALL    KEYCHK
6402  E8 6768 R                     CALL    INTOFF          ;disable interrupts during grab
6405  8A 26 6170 R                  MOV     AH,COMMAND      ;tell MicronEye to send picture (w/o soak)
6409  E8 6720 R                     CALL    SENDCMD
640C  8B 16 6164 R                  MOV     DX,STATUS
6410  8C D8                         MOV     AX,DS           ;equate extra segment and data segment
6412  8E C0                         MOV     ES,AX
6414  C7 06 6178 R 0000             MOV     WHITECT, 0      ;zero whitect
641A  FC                            CLD                     ;set direction reg to forward movement
641B  B9 000F               nfbyt:  MOV     CX,15           ;set up timeout register for byte receipt
641E  EC                    rechk:  IN      AL,DX           ;if byte not available after 15
641F  D0 E0                         SHL     AL,1            ; checks then we assume the MicronEye is
6421  73 0D                         JNC     DNCHK           ; done sending
6423  42                            INC     DX              ;when byte has come we point DX to
6424  EC                            IN      AL,DX           ; DATAIN and get the byte and then
6425  4A                            DEC     DX              ; repoint DX at the status register
6426  AA                            STOSB                   ;put the byte in the buffer
6427  D0 E0                         SHL     AL,1            ;increment whitect if high bit of byte
6429  83 16 6178 R 00               ADC     WHITECT, 0      ; was white
642E  EB EB                         JMP     NFBYT           ;go back and try and get another byte
6430  E2 EC                 dnchk:  LOOP    RECHK           ;this is the timeout decrementer
6432  8B C7                         MOV     AX,DI           ;if we have timed out then we check and
6434  2B C3                         SUB     AX,BX           ; see if we got as many bytes as we had
6436  3D 0014                       CMP     AX,20           ; hoped to get
6439  7F 03                         JG      NTO             ;if not enough bytes received then we
643B  E8 675A R                     CALL    BEEP            ; beep to show our disgust
643E  E8 6761 R             nto:    CALL    INTON           ;reenable interrupts
6441  8A 26 6170 R                  MOV     AH,COMMAND      ;tell MicronEye to refresh w/o send
6445  80 CC 80                      OR      AH,80H
6448  E8 6720 R                     CALL    SENDCMD
644B  80 3E 6170 R 3F               CMP     COMMAND,3FH     ;if we use two-arrays and the 256 x 128 picture
6450  75 0E                         JNE     NTA             ; we have a small problem.  The descramble due
6452  8B 3E 61EA R                  MOV     DI,MAPADR       ; to topology acts on the top array exactly
6456  8B F7                         MOV     SI,DI           ; opposite to what we are used to.  So even
6458  83 C6 20                      ADD     SI,32           ; rows need to be processed like odd and
645B  B9 0800                       MOV     CX,2048         ; vice-versa.  the quick and dirty fix is to
645E  F3/ A5                        REP     MOVSW           ; scoot the entire top array up one row.
6460  E8 6464 R             nta:    CALL    KEYCHK
6463  C3                            RET
                            ;
6464  B4 01                 keychk: MOV     AH,1            ;if key is available from keyboard buffer
6466  CD 16                         INT     16H             ; then get the key, put in key_value
6468  74 09                         JZ      NOKEY
646A  B4 00                         MOV     AH,0
646C  CD 16                         INT     16H
646E  A3 0016 R                     MOV     KEY_VALUE,AX
6471  EB F1                         JMP     KEYCHK
6473  C3                    nokey:  RET
                            ;
6474                        FRAMEGRAB ENDP
                            ;
                                    SUBTTL  EXPANSION ROUTINE for 512 x 64 picture
```

```
                                    PAGE
6474                        SMOOTHB PROC    NEAR
6474  06                            PUSH.   ES
6475  9C                            PUSHF
6476  FC                            CLD
6477  8C D8                         MOV     AX,DS
6479  8E C0                         MOV     ES,AX
647B  BF 0022 R                     MOV     DI,OFFSET WORKMAP
647E  BE 28C2 R                     MOV     SI,OFFSET BITMAP
6481  8B 1E 617A R                  MOV     BX,TOTBYTES     ;do a word at a time to
6485  D1 EB                         SHR     BX,1            ; speed things up
6487  89 1E 6176 R                  MOV     WORDCT,BX
648B  AD                    nwordc: LODSW                   ;spread out byte 4x
648C  86 C4                         XCHG    AL,AH
648E  8B D0                         MOV     DX,AX
6490  B9 0008                       MOV     CX,8            ;set up bit ctr (2 bits/pass)
6493  BB 0000               nbitc:  MOV     BX,0            ;this stretches the image from a
6496  D1 E2                         SHL     DX,1            ; 128 x 64 image to a 512 x 64 image
6498  D1 D3                         RCL     BX,1            ; which goes a long way to correct the
649A  D1 E2                         SHL     DX,1            ; aspect ratio.
649C  D1 D3                         RCL     BX,1
649E  8A 87 61C5 R                  MOV     AL,CPAT[BX]
64A2  AA                            STOSB
64A3  E2 EE                         LOOP    nbitc
64A5  FF 0E 6176 R                  DEC     WORDCT
64A9  75 E0                         JNZ     NWORDC
64AB  9D                            POPF
64AC  07                            POP     ES
64AD  C3                            RET
64AE                        SMOOTHB ENDP
                            ;
                                    SUBTTL  ENHANCE ROUTINE for 512 x 128 picture
```

ENHANCE ROUTINE for 512 x 128 picture

```
                                        PAGE
64AE                            ENHANCE PROC    NEAR            ;this routine takes the 256 x 128 image from
64AE  FC                                CLD                     ;  the MicronEye and converts it to a 512 x
64AF  E8 65A3 R                         CALL    CLEARW          ;  128 image with properly placed pixels.
64B2  BE 28C2 R                         MOV     SI,OFFSET BITMAP;zero out WORKMAP buffer, point SI (source
64B5  BF 0022 R                         MOV     DI,OFFSET WORKMAP ;index) to the 256 x 128 array, and DI to
64B8  A1 001C R                         MOV     AX,SCR_ROWCT    ;  the 512 x 128 array
64BB  D1 E8                             SHR     AX,1            ;set up rowctr to be half the number of rows
64BD  A3 6171 R                         MOV     ROWCTR,AX       ;  because we process in even/odd row pairs
64C0  C7 06 6173 R 0020         newrow: MOV     COLCTR,32       ;set up colctr to 32 bytes/row to do even row
64C6  AC                        evrow:  LODSB                   ;get byte from source and increment source idx
64C7  BB 0000                           MOV     BX,0            ;zero BX and DX-- BX will catch the odd bits
64CA  8B D3                             MOV     DX,BX           ;  and DX will catch the even bits
64CC  D0 E0                             SHL     AL,1            ;shift high-order bit (7) into BX
64CE  D1 D3                             RCL     BX,1
64D0  D0 E0                             SHL     AL,1            ;shift bit 6 into DX
64D2  D1 D2                             RCL     DX,1
64D4  B9 0003                           MOV     CX,3            ;set up to shift other 6 bits-- 3 odd, 3 even
64D7                            nevbyt:
64D7  D1 E3                             SHL     BX,1            ;the even bits go into the BX register such
64D9  D1 E3                             SHL     BX,1            ;  that the final bit pattern is:
64DB  D1 E3                             SHL     BX,1            ;  000x000x000x000x where the x's correspond to
64DD  D0 E0                             SHL     AL,1            ;     7   5   3   1 bit positions.
64DF  D1 D3                             RCL     BX,1            ;
64E1  D1 E2                             SHL     DX,1            ;the odd bits go into the DX register such
64E3  D1 E2                             SHL     DX,1            ;  that the final bit pattern is:
64E5  D1 E2                             SHL     DX,1            ;  00x000x000x000x0
64E7  D0 E0                             SHL     AL,1            ;    6   4   2   0  bit positions
64E9  D1 D2                             RCL     DX,1
64EB  E2 EA                             LOOP    NEVBYT          ;
64ED  D1 E2                             SHL     DX,1            ;
64EF  86 F2                             XCHG    DH,DL           ;
64F1  86 FB                             XCHG    BH,BL           ;
64F3  09 15                             OR      [DI],DX         ;after we flip sex on BX and DX to get bytes in
64F5  09 5D 40                          OR      64[DI],BX       ;  proper order, we OR the pattern with the
64F8  83 C7 02                          ADD     DI,2            ;  bits already at the destination of the two
64FB  FF 0E 6173 R                      DEC     COLCTR          ;  words.
64FF  75 C5                             JNZ     EVROW           ;
6501  C7 06 6173 R 0020                 MOV     COLCTR,32       ;after completing the even row we go onto
6507  AC                        oddrow: LODSB                   ;  the next row which is an odd row
6508  BB 0000                           MOV     BX,0            ;we get the byte and set up the BX and DX
650B  8B D3                             MOV     DX,BX           ;  registers as before.  This time however
650D  D0 E0                             SHL     AL,1            ;  we are going to use a slightly different
650F  D1 D3                             RCL     BX,1            ;  bit pattern to get things into their
6511  D0 E0                             SHL     AL,1            ;  proper places
6513  D1 D2                             RCL     DX,1
6515  B9 0003                           MOV     CX,3
6518                            nodbyt:
6518  D1 E3                             SHL     BX,1            ;the BX register should end up with the
651A  D1 E3                             SHL     BX,1            ;  following bit pattern:
651C  D1 E3                             SHL     BX,1            ;  x000x000x000x000
651E  D0 E0                             SHL     AL,1            ;  7   5   3   1
6520  D1 D3                             RCL     BX,1
6522  D1 E2                             SHL     DX,1            ;the DX register should end up with the
6524  D1 E2                             SHL     DX,1            ;  following bit pattern:
6526  D1 E2                             SHL     DX,1            ;  00000x000x000x00 0x00
6528  D0 E0                             SHL     AL,1            ;       6   4   2    0
652A  D1 D2                             RCL     DX,1            ;since not all the pattern fits into DX we do
652C  E2 EA                             LOOP    NODBYT          ;  some fancy footwork
652E  D1 E3                             SHL     BX,1            ;the net result of all of this is that we
6530  D1 E3                             SHL     BX,1            ;  have some bit patterns that can now be put
6532  D1 E3                             SHL     BX,1            ;  into the output array.
6534  D1 EA                             SHR     DX,1
6536  9F                                LAHF
6537  D1 EA                             SHR     DX,1
6539  86 F2                             XCHG    DH,DL
653B  86 FB                             XCHG    BH,BL
653D  09 1D                             OR      [DI],BX
653F  09 55 40                          OR      64[DI],DX
6542  BB 0000                           MOV     BX,0            ;you can see that after we have done this for
6545  9E                                SAHF                    ;  every row in the array, 1/2 the bits in the
6546  D1 DB                             RCR     BX,1            ;  512 x 128 array (except for some of the edge
6548  D1 EB                             SHR     BX,1            ;  pixels) will be filled in.
654A  08 7D 42                          OR      66[DI],BH
654D  83 C7 02                          ADD     DI,2
6550  FF 0E 6173 R                      DEC     COLCTR
6554  75 B1                             JNZ     ODDROW
6556  FF 0E 6171 R                      DEC     ROWCTR          ;This is the fill-in algorithm mentioned
655A  75 44                             JNZ     NEWRWJ          ;  in the manual. There are probably other
```

```
655C  C7 06 61B8 R 6666       FILLIN: MOV    PAT,6666H          ;  approaches towards creating a 512 x 128
6562  BF 0022 R                       MOV    DI,OFFSET WORKMAP ; array form the 256 x 128 image from the
6565  BE 0020                 filn:   MOV    SI,32              ;  Microneye.
6568  8B 05                   filp:   MOV    AX,[DI]
656A  8B 9D 0080                      MOV    BX,128[DI]
656E  8B 4D 40                        MOV    CX,64[DI]
6571  8B D1                           MOV    DX,CX
6573  D1 E1                           SHL    CX,1
6575  D1 EA                           SHR    DX,1
6577  0B CA                           OR     CX,DX
6579  8B D1                           MOV    DX,CX
657B  23 D0                           AND    DX,AX
657D  23 C3                           AND    AX,BX
657F  23 CB                           AND    CX,BX
6581  0B C1                           OR     AX,CX
6583  0B C2                           OR     AX,DX
6585  23 06 61B8 R                    AND    AX,PAT
6589  09 45 40                        OR     64[DI],AX
658C  83 C7 02                        ADD    DI,2
658F  4E                              DEC    SI
6590  75 D6                           JNZ    filp
6592  B8 FFFF                         MOV    AX,0FFFFH
6595  31 06 61B8 R                    XOR    PAT,AX
6599  81 FF 1FC2 R                    CMP    DI,8096+OFFSET WORKMAP
659D  7C C6                           JL     filn
659F  C3                              RET
65A0  E9 64C0 R               NEWRWJ: JMP    NEWROW
65A3                          ENHANCE ENDP
                              ;
65A3                          CLEARW  PROC   NEAR               ;the CLEARW routine zeroes the WORKMAP buffer
65A3  BF 0022 R                       MOV    DI,OFFSET WORKMAP
65A6  EB 04 90                        JMP    CW                 ;the CLEARG routine zeroes the BITMAP buffer
65A9  BF 28C2 R               CLEARG: MOV    DI,OFFSET BITMAP
65AC  FC                      cw:     CLD                       ;note that only an area corresponding
65AD  8B 0E 0014 R                    MOV    CX,MAPBYTES        ;  in size to the current picture type
65B1  D1 E9                           SHR    CX,1               ;  is cleared
65B3  8C D8                           MOV    AX,DS              ;set up the ES register to point to the
65B5  8E C0                           MOV    ES,AX              ;  data segment
65B7  B8 0000                         MOV    AX,0               ;select the value to be replicated thru memory
65BA  F3/ AB                          REP    STOSW              ;this command singlehandedly zeroes the desired
65BC  C3                              RET                       ;  memory area
65BD                          CLEARW  ENDP
                              ;
                                      SUBTTL GREYADD ROUTINE for 512 x 64 image
```

```
                                        PAGE
65BD                            GREYADD PROC    NEAR
65BD  FC                                CLD                     ;set forward direction
65BE  BE 28C2 R                         MOV     SI,OFFSET BITMAP  ;use BITMAP as source and WORKMAP as
65C1  BF 0022 R                         MOV     DI,OFFSET WORKMAP ;the destination
65C4  8B 1E 617A R                      MOV     BX,TOTBYTES
65C8  D1 EB                             SHR     BX,1            ;set up count of words to be processed
65CA  89 1E 6176 R                      MOV     WORDCT,BX
65CE  B9 0008                   nword:  MOV     CX,8            ;set up loop ctr, we do 2 bits per pass
65D1  8B 04                             MOV     AX,[SI]         ;word from first image in AX
65D3  8B 94 28A0                        MOV     DX,WORD PTR (OFFSET BITMAPB - OFFSET BITMAP)[SI]
65D7  86 C4                             XCHG    AL,AH           ;corresponding word from second image in DX
65D9  86 D6                             XCHG    DL,DH           ;realign byte sex for proper shifting
65DB  BB 0000                   nbitd:  MOV     BX,0            ;BX is going to be built up to contain a
65DE  D1 E0                             SHL     AX,1            ; ptr to the proper pattern to represent
65E0  80 D3 00                          ADC     BL,0            ; 2 sets of 4-bit black, white, or grey blobs
65E3  D1 E2                             SHL     DX,1            ; depending on the values of corresponding
65E5  80 D3 00                          ADC     BL,0            ; pixels in the two images
65E8  D1 E3                             SHL     BX,1            ;this is done by shifting a pixel out AX and
65EA  D1 E3                             SHL     BX,1            ; DX and putting the sum in DX creating the
65EC  D1 E0                             SHL     AX,1            ; first 4-bit blob's ptr (0=black, 1=grey,
65EE  80 D3 00                          ADC     BL,0            ; 2=white); we shift the sum over two bits,
65F1  D1 E2                             SHL     DX,1            ; take the next pixel out of AX and DX, sum
65F3  80 D3 00                          ADC     BL,0            ; them and put that sum in BX as well
65F6  8A BF 61BA R                      MOV     BH,DPAT[BX]     ;the result is a ptr into DPAT that will return
65FA  88 3D                             MOV     [DI],BH         ; a double blob that is then put into the
65FC  47                                INC     DI              ; destination buffer
65FD  E2 DC                             LOOP    NBITD
65FF  83 C6 02                          ADD     SI,2
6602  FF 0E 6176 R                      DEC     WORDCT
6606  75 C6                             JNZ     NWORD
6608  C3                                RET
6609                            GREYADD ENDP
                                ;
                                        SUBTTL  FILLIN ROUTINE FOR 640 x 128 picture
```

```
                                                PAGE
6609                                    FILLIN2 PROC    NEAR
6609  BE 0022 R                                 MOV     SI,OFFSET WORKMAP
660C  8B 1E 001C R                              MOV     BX,SCR_ROWCT    ;as discussed in the topology section and
6610  D1 EB                                     SHR     BX,1            ;  elsewhere, the 256 x 128 array needs to
6612  89 1E 6171 R                              MOV     ROWCTR,BX       ;  be descrambled and filled in.  This takes
6616  8B 16 001E R                              MOV     DX,SCR_COLCT    ;  care of the fillin when expanding the
661A  D1 EA                                     SHR     DX,1            ;  256 x 128 to 1024 x 128 (clipped to
661C  89 16 6173 R                              MOV     COLCTR,DX       ;  640 x 128 because of screen limitations.
6620  8B FA                                     MOV     DI,DX
6622  8B 04                             nebyt:  MOV     AX,[SI]         ;to do the fillin we look at the bytes one
6624  8B 5C 50                                  MOV     BX,80[SI]       ;  row back, one row forward, and the bit to
6627  8B CB                                     MOV     CX,BX           ;  the side of each 'hole'.
6629  D1 E3                                     SHL     BX,1
662B  D1 E3                                     SHL     BX,1
662D  D1 E9                                     SHR     CX,1
662F  D1 E9                                     SHR     CX,1
6631  0B D9                                     OR      BX,CX
6633  8B 8C 00A0                                MOV     CX,160[SI]      ;
6637  8B D1                                     MOV     DX,CX
6639  23 C8                                     AND     CX,AX
663B  23 C3                                     AND     AX,BX
663D  23 DA                                     AND     BX,DX
663F  0B C3                                     OR      AX,BX
6641  0B C1                                     OR      AX,CX
6643  25 3C3C                                   AND     AX,3C3CH        ;
6646  09 44 50                                  OR      80[SI],AX       ;
6649  83 C6 02                                  ADD     SI,2
664C  4F                                        DEC     DI
664D  75 D3                                     JNZ     NEBYT
664F  8B 3E 6173 R                              MOV     DI,COLCTR
6653  8B 04                             nobyt:  MOV     AX,[SI]         ;when we work with the odd rows we do exactly
6655  8B 5C 50                                  MOV     BX,80[SI]       ;  row back, one row forward, and the bit to
6658  86 FB                                     XCHG    BH,BL
665A  8B CB                                     MOV     CX,BX           ;  the side of each 'hole'.
665C  D1 E3                                     SHL     BX,1
665E  D1 E3                                     SHL     BX,1
6660  D1 E9                                     SHR     CX,1
6662  D1 E9                                     SHR     CX,1
6664  0B D9                                     OR      BX,CX
6666  86 FB                                     XCHG    BH,BL
6668  8B 8C 00A0                                MOV     CX,160[SI]      ;
666C  8B D1                                     MOV     DX,CX
666E  23 C8                                     AND     CX,AX
6670  23 C3                                     AND     AX,BX
6672  23 DA                                     AND     BX,DX
6674  0B C3                                     OR      AX,BX
6676  0B C1                                     OR      AX,CX
6678  25 C3C3                                   AND     AX,0C3C3H       ;
667B  09 44 50                                  OR      80[SI],AX       ;
667E  83 C6 02                                  ADD     SI,2
6681  4F                                        DEC     DI
6682  75 CF                                     JNZ     NOBYT
6684  8B 3E 6173 R                              MOV     DI,COLCTR
6688  FF 0E 6171 R                              DEC     ROWCTR
668C  75 94                                     JNZ     NEBYT
668E  C3                                        RET
668F                                    FILLIN2 ENDP
                                        ;
                                                SUBTTL  ENHANCE ROUTINE for 640 x 128 picture
```

```
                                             PAGE
668F                             ENHANCEG PROC    NEAR
668F  E8 65A9 R                           CALL    CLEARG
6692  BE 5162 R                           MOV     SI,OFFSET BITMAPB
6695  BF 28C2 R                           MOV     DI,OFFSET BITMAP
6698  EB 0A 90                            JMP     ENTRYB
669B                             ENHANCEG ENDP
                                 ;
669B                             ENHANCE2 PROC    NEAR
669B  E8 65A3 R                           CALL    CLEARW
669E  BE 28C2 R                           MOV     SI,OFFSET BITMAP
66A1  BF 0022 R                           MOV     DI,OFFSET WORKMAP
66A4                             ENTRYB:
66A4  FC                                  CLD
66A5  8B 1E 001C R                        MOV     BX,SCR_ROWCT
66A9  D1 EB                               SHR     BX,1
66AB  B2 0A                               MOV     DL,10
66AD  8A F2                      nrow2:   MOV     DH,DL
66AF  AD                         nbe2:    LODSW
66B0  86 C4                               XCHG    AL,AH
66B2  B9 0008                             MOV     CX,8
66B5  D1 E0                      nxt2:    SHL     AX,1
66B7  73 04                               JNC     NOCA
66B9  80 4D 50 03                         OR      BYTE PTR 80[DI],3
66BD  D1 E0                      noca:    SHL     AX,1
66BF  73 03                               JNC     NOCB
66C1  80 0D 0C                            OR      BYTE PTR [DI],0CH
66C4  47                         nocb:    INC     DI
66C5  E2 EE                               LOOP    NXT2
66C7  FE CE                               DEC     DH
66C9  75 E4                               JNZ     NBE2
66CB  8A F2                               MOV     DH,DL
66CD  83 C6 0C                            ADD     SI,12           ;skip to start of next row
66D0  AD                         nb02:    LODSW
66D1  86 C4                               XCHG    AL,AH
66D3  B9 0008                             MOV     CX,8
66D6  D1 E0                      nxt3:    SHL     AX,1
66D8  73 03                               JNC     NOCC
66DA  80 0D C0                            OR      BYTE PTR [DI],0C0H
66DD  D1 E0                      nocc:    SHL     AX,1
66DF  73 04                               JNC     NOCD
66E1  80 4D 51 30                         OR      BYTE PTR 81[DI],30H
66E5  47                         nocd:    INC     DI
66E6  E2 EE                               LOOP    NXT3
66E8  FE CE                               DEC     DH
66EA  75 E4                               JNZ     NB02
66EC  83 C6 0C                            ADD     SI,12
66EF  4B                                  DEC     BX
66F0  75 BB                               JNZ     NROW2
66F2  C3                                  RET
66F3                             ENHANCE2 ENDP
                                 ;
                                          SUBTTL  GREYADD ROUTINE for 640 x 128 picture
```

```
                                    PAGE
66F3                        GREYADD2 PROC NEAR
66F3  FC                            CLD
66F4  8C D8                         MOV     AX,DS           ;this routine is very simplistic
66F6  8E C0                         MOV     ES,AX           ;we takes the images produced by the two
66F8  BE 28C2 R                     MOV     SI,OFFSET BITMAP;  different exposure times and alternately
66FB  BF 0022 R                     MOV     DI,OFFSET WORKMAP; use the bits from one image with the
66FE  8B 1E 0014 R                  MOV     BX,MAPBYTES     ;  bits from the other image
6702  D1 EB                         SHR     BX,1
6704  AD                    ngrey:  LODSW
6705  8B 0D                         MOV     CX,[DI]
6707  25 5555                       AND     AX,5555H
670A  81 E1 AAAA                    AND     CX,0AAAAH
670E  0B C1                         OR      AX,CX
6710  AB                            STOSW
6711  4B                            DEC     BX
6712  75 F0                         JNZ     NGREY
6714  C3                            RET
6715                        GREYADD2 ENDP
                            ;
6715                        ACIACLR PROC    NEAR
6715  8B 16 6162 R                  MOV     DX,CONTROL
6719  B0 C0                         MOV     AL,0C0H                 ;send master reset to camera
671B  EE                            OUT     DX,AL
671C  B0 28                         MOV     AL,28H                  ;send camera protocol of
671E  EE                            OUT     DX,AL                   ;  1 start, 8 data, 1 stop bits
671F  C3                            RET
6720                        ACIACLR ENDP
                            ;
6720                        SENDCMD PROC    NEAR
6720  B9 03E8                       MOV     CX,1000
6723  8B 16 6164 R          scmd:   MOV     DX,STATUS
6727  EC                            IN      AL,DX                   ;get status of camera
6728  A8 40                         TEST    AL,40H                  ;see if command can be sent
672A  75 06                         JNZ     sok
672C  E2 F5                         LOOP    scmd                    ;loop until ready or timeout
672E  E8 675A R                     CALL    BEEP
6731  C3                            RET
6732  8A C4                 sok:    MOV     AL,AH                   ;set up command
6734  8B 16 6168 R                  MOV     DX,DATAOUT
6738  EE                            OUT     DX,AL                   ;send camera command
6739  C3                            RET
673A                        SENDCMD ENDP
                            ;
673A                        SOAK    PROC    NEAR
673A  8B 0E 616C R                  MOV     CX,EXPOSE_TIME  ;soaktime = number msec delay
673E  E3 09                         JCXZ    NOSOAK
6740  51                    S1:     PUSH    CX
6741  B9 0106                       MOV     CX,262          ;set up loop for 1 msec
6744  E2 FE                 S2:     LOOP    S2
6746  59                            POP     CX
6747  E2 F7                         LOOP    S1
6749  C3                    NOSOAK: RET
674A                        SOAK    ENDP
                            ;
674A                        GREYSOAK PROC NEAR
674A  8B 0E 616C R                  MOV     CX,EXPOSE_TIME
674E  E3 09                         JCXZ    NGSOAK
6750  51                    G1:     PUSH    CX
6751  B9 00DE                       MOV     CX,222
6754  E2 FE                 G2:     LOOP    G2
6756  59                            POP     CX
6757  E2 F7                         LOOP    G1
6759  C3                    ngsoak: RET
675A                        GREYSOAK ENDP
                            ;
675A                        BEEP    PROC    NEAR
675A  B4 02                         MOV     AH,2            ; DOS call to sound bell
675C  B2 07                         MOV     DL,7
675E  CD 21                         INT     21H
6760  C3                            RET
6761                        BEEP    ENDP
                            ;
6761                        INTON   PROC    NEAR            ;enable kbd/timer interrupts
6761  E4 21                         IN      AL,21H
6763  24 FC                         AND     AL,0FCH
6765  E6 21                         OUT     21H,AL
6767  C3                            RET
6768                        INTON   ENDP
```

```
                          ;
6768                      INTOFF  PROC   NEAR        ;disable kbd/timer interrupts
6768  E4 21                       IN     AL,21H
676A  0C 03                       OR     AL,3
676C  E6 21                       OUT    21H,AL
676E  C3                          RET
676F                      INTOFF  ENDP
                          ;
676F                      CSEG    ENDS
                                  END
```

Segments and groups:

N a m e	Size	align	combine class
CSEG	676F	PARA	PUBLIC

Symbols:

N a m e	Type	Value	Attr	
ACIACLR.	N PROC	6715	CSEG	Length =000B
ARRAYCT.	L WORD	0020	CSEG	
BEEP	N PROC	675A	CSEG	Length =0007
BITCT.	L BYTE	6175	CSEG	
BITMAP	L BYTE	28C2	CSEG	Length =28A0
BITMAPB.	L BYTE	5162	CSEG	Length =1000
BYTES_TAB.	L WORD	61A0	CSEG	
CLEARG	L NEAR	65A9	CSEG	
CLEARW	N PROC	65A3	CSEG	Length =001A
CMD_TAB.	L WORD	61AC	CSEG	
COLCTR	L WORD	6173	CSEG	
COLCT_TAB.	L WORD	6194	CSEG	
COMMAND.	L BYTE	6170	CSEG	
CONTROL.	L WORD	6162	CSEG	
CPAT	L BYTE	61C5	CSEG	
CW	L NEAR	65AC	CSEG	
DATAIN	L WORD	6166	CSEG	
DATAOUT.	L WORD	6168	CSEG	
DNCHK.	L NEAR	6430	CSEG	
DONE	L NEAR	6224	CSEG	
DPAT	L BYTE	61BA	CSEG	
EARLYX	L NEAR	6372	CSEG	
ENDSTR	L NEAR	638B	CSEG	
ENHANCE.	N PROC	64AE	CSEG	Length =00F5
ENHANCE2	N PROC	669B	CSEG	Length =0058
ENHANCEG	N PROC	668F	CSEG	Length =000C
ENTRYB	L NEAR	66A4	CSEG	
EVROW.	L NEAR	64C6	CSEG	
EXITEOF.	L WORD	616A	CSEG	
EXPOSE_TIME.	L WORD	616C	CSEG	
FG2.	L NEAR	63EB	CSEG	
FILLIN	L NEAR	655C	CSEG	
FILLIN2.	N PROC	6609	CSEG	Length =0086
FILN	L NEAR	6565	CSEG	
FILP	L NEAR	6568	CSEG	
FLIPIT	L NEAR	634D	CSEG	
FRAMEGRAB.	N PROC	63D2	CSEG	Length =00A2
G1	L NEAR	6750	CSEG	
G2	L NEAR	6754	CSEG	
GETPIC	N PROC	61EC	CSEG	Length =00B2
GETPICT.	F PROC	0000	CSEG	Length =0006
GLIN	L BYTE	61D2	CSEG	
GPIC_X	L NEAR	0003	CSEG	
GREYADD.	N PROC	65BD	CSEG	Length =004C
GREYADD2	N PROC	66F3	CSEG	Length =0022
GREYGRAB	N PROC	63C9	CSEG	Length =0009
GREYSOAK	N PROC	674A	CSEG	Length =0010
GRLEN.	L WORD	61D6	CSEG	
GRSETUP.	L BYTE	61C9	CSEG	
INTOFF	N PROC	6768	CSEG	Length =0007
INTON.	N PROC	6761	CSEG	Length =0007
KEYCHK	L NEAR	6464	CSEG	
KEY_VALUE.	L WORD	0016	CSEG	
MAPADR	L WORD	61EA	CSEG	
MAPBYTES	L WORD	0014	CSEG	
MOVESCR.	N PROC	638E	CSEG	Length =003B
NBO2	L NEAR	66D0	CSEG	
NBE2	L NEAR	66AF	CSEG	
NBITC.	L NEAR	6493	CSEG	
NBITD.	L NEAR	65DB	CSEG	
NEBYT.	L NEAR	6622	CSEG	
NEVBYT	L NEAR	64D7	CSEG	
NEWROW	L NEAR	64C0	CSEG	
NEWRWJ	L NEAR	65A0	CSEG	
NFBYT.	L NEAR	641B	CSEG	
NGREY.	L NEAR	6704	CSEG	
NGSOAK	L NEAR	6759	CSEG	
NO2ARRAY	L NEAR	62F2	CSEG	
NOBYT.	L NEAR	6653	CSEG	

```
NOCA . . . . . . . . . . . . . . .      L NEAR  66BD    CSEG
NOCB . . . . . . . . . . . . . . .      L NEAR  66C4    CSEG
NOCC . . . . . . . . . . . . . . .      L NEAR  66DD    CSEG
NOCD . . . . . . . . . . . . . . .      L NEAR  66E5    CSEG
NOCHR. . . . . . . . . . . . . . .      L NEAR  6339    CSEG
NODBYT . . . . . . . . . . . . . .      L NEAR  6518    CSEG
NOKEY. . . . . . . . . . . . . . .      L NEAR  6473    CSEG
NOSOAK . . . . . . . . . . . . . .      L NEAR  6749    CSEG
NOTWO. . . . . . . . . . . . . . .      L NEAR  629B    CSEG
NPUT . . . . . . . . . . . . . . .      L NEAR  637E    CSEG
NROW2. . . . . . . . . . . . . . .      L NEAR  66AD    CSEG
NTA. . . . . . . . . . . . . . . .      L NEAR  6460    CSEG
NTO. . . . . . . . . . . . . . . .      L NEAR  643E    CSEG
NWORD. . . . . . . . . . . . . . .      L NEAR  65CE    CSEG
NWORDC . . . . . . . . . . . . . .      L NEAR  648B    CSEG
NXT2 . . . . . . . . . . . . . . .      L NEAR  66B5    CSEG
NXT3 . . . . . . . . . . . . . . .      L NEAR  66D6    CSEG
NXTBYT . . . . . . . . . . . . . .      L NEAR  6346    CSEG
NXTROW . . . . . . . . . . . . . .      L NEAR  63AB    CSEG
NXTSET . . . . . . . . . . . . . .      L NEAR  631E    CSEG
ODDROW . . . . . . . . . . . . . .      L NEAR  6507    CSEG
PARMCLC. . . . . . . . . . . . . .      F PROC  0006    CSEG    Length =0006
PARM_X . . . . . . . . . . . . . .      L NEAR  0009    CSEG
PAT. . . . . . . . . . . . . . . .      L WORD  61B8    CSEG
PCALC. . . . . . . . . . . . . . .      N PROC  626E    CSEG    Length =0095
PIC. . . . . . . . . . . . . . . .      L WORD  61DA    CSEG
PICA . . . . . . . . . . . . . . .      L NEAR  6246    CSEG
PICB . . . . . . . . . . . . . . .      L NEAR  6247    CSEG
PICC . . . . . . . . . . . . . . .      L NEAR  624A    CSEG
PICD . . . . . . . . . . . . . . .      L NEAR  6253    CSEG
PICDUMP. . . . . . . . . . . . . .      F PROC  000C    CSEG    Length =0004
PICE . . . . . . . . . . . . . . .      L NEAR  6256    CSEG
PICF . . . . . . . . . . . . . . .      L NEAR  625C    CSEG
PICTYPE. . . . . . . . . . . . . .      L WORD  0018    CSEG
PUTBYTE. . . . . . . . . . . . . .      L NEAR  6377    CSEG
PUTSTR . . . . . . . . . . . . . .      L NEAR  637C    CSEG
RECHK. . . . . . . . . . . . . . .      L NEAR  641E    CSEG
RESTART. . . . . . . . . . . . . .      L NEAR  6205    CSEG
ROWCTR . . . . . . . . . . . . . .      L WORD  6171    CSEG
ROWCT_TAB. . . . . . . . . . . . .      L WORD  6188    CSEG
ROWOK. . . . . . . . . . . . . . .      L NEAR  62E9    CSEG
S1 . . . . . . . . . . . . . . . .      L NEAR  6740    CSEG
S2 . . . . . . . . . . . . . . . .      L NEAR  6744    CSEG
SCMD . . . . . . . . . . . . . . .      L NEAR  6723    CSEG
SCRDUMP. . . . . . . . . . . . . .      N PROC  6303    CSEG    Length =008B
SCREEN_START . . . . . . . . . . .      L WORD  001A    CSEG
SCRXFER. . . . . . . . . . . . . .      F PROC  0010    CSEG    Length =0004
SCR_COLCT. . . . . . . . . . . . .      L WORD  001E    CSEG
SCR_ROWCT. . . . . . . . . . . . .      L WORD  001C    CSEG
SENDCMD. . . . . . . . . . . . . .      N PROC  6720    CSEG    Length =001A
SETBMP . . . . . . . . . . . . . .      L NEAR  63E8    CSEG
SMOOTHB. . . . . . . . . . . . . .      N PROC  6474    CSEG    Length =003A
SOAK . . . . . . . . . . . . . . .      N PROC  673A    CSEG    Length =0010
SOAKPTR. . . . . . . . . . . . . .      L WORD  61E6    CSEG
SOK. . . . . . . . . . . . . . . .      L NEAR  6732    CSEG
STATUS . . . . . . . . . . . . . .      L WORD  6164    CSEG
TEXTSET. . . . . . . . . . . . . .      L BYTE  61CE    CSEG
TOTBYTES . . . . . . . . . . . . .      L WORD  617A    CSEG
TOT_TAB. . . . . . . . . . . . . .      L WORD  617C    CSEG
TSET . . . . . . . . . . . . . . .      L NEAR  636A    CSEG
WHITECT. . . . . . . . . . . . . .      L WORD  617B    CSEG
WHITE_PCT. . . . . . . . . . . . .      L WORD  616E    CSEG
WORDCT . . . . . . . . . . . . . .      L WORD  6176    CSEG
WORKMAP. . . . . . . . . . . . . .      L BYTE  0022    CSEG    Length =28A0

Warning Severe
Errors  Errors
0       0
```

APPENDIX F

GUIDE TO OPTICS SELECTION AND LIGHTING TECHNIQUES

F.1 LIGHTING CONSIDERATIONS FOR THE IS32 OPTICRAM

The IS32 OpticRAM lends itself to profiling scenes and component parts by imaging the dimension to be measured onto a matrix of light sensors where each light sensor is equal to some distance in physical space.

The MicronEye Camera needs a high contrast scene in order to image the object into the IS32. Unlike a TV camera which can respond to shades of gray, the IS32 is a digital chip where each picture element makes a black/white judgement based on an arbitrary light level used as a threshold (trip light level). Portions of the scene that are lighter than the threshold level will be judged as white while portions of the scene darker than the threshold level will be judged as black.

For example, if the trip light level is made lighter, then a new slice of the scene would be captured around that light threshold. One can look at shades of gray as planes of binary light level slices. One example: 64 gray scales means 64 binary light level slices.

The trip light level can be changed in one of three ways:

1. Changing the exposure time.

2. Changing the f-stop on the lens.

3. Changing the light on the scenes itself.

Doubling the exposure time is the same as opening the f-stop by one stop, (changing the f-stop to the next smaller number), or in other words doubling the amount of light. Contrast can now be defined as a minimum difference between adjacent slices. Example: In taking 64 gray scale slices there is normally only one slice where the adjacent slice is of a minimum difference.

High contrast means that there are more than two adjacent slices that are about the same (usually three or more adjacent slices are about the same).

F.1.1 FRONT LIGHTING

Front lit scenes, where the camera is on the same side of the scene as the light source or ambient light, usually is low in contrast. In this situation extreme care in setting up uniform lighting on the scene is necessary and the optimum trip light level needs to be used. Front lighting requires a multiple diffused light source such that the contrast in the scene is increased. If defects or points of interest are to be emphasized, side lighting such that the defects or points of interest cast a shadow, or increase in spectral energy (reflection) will usually point out the defects.

To set up a front lit scene, normally one or more flood lamps (outdoor flood lamps purchased from a local hardware store are adequate) are arranged around the scene far enough away so that there are no shadows. Then the f-stop, focus and lamps are adjusted for maximum contrast and focus. Adjust the focus where the smallest part of the scene has the most detail. The depth of focus (the distance the scene can move in relation to the camera and still be in focus) is increased at higher f-stops. Increase the amount of light and/or the integration time to optimize the result.

A trade-off of lighting, integration time, f-stop and scene-to-camera positioning (also lens selection) is necessary to optimize the result. Due to light falling off (at a slope of cos*cos*cos*cos) from the center of the lens going to the edges of the lens, the periphery of a scene takes more light for a uniform trip light threshold to capture the scene.

F.1.2 BACK LIGHTING

For a backlit scene, the light comes from behind the scene so that the object being viewed is shadowed into the camera. Backlighting the object, for maximum contrast will give the best repeatable results. Backlighting is recommended if the camera is used to measure the object or certain aspects of the object and/or for part recognition since the trip light level can move a large amount without degrading the results.

The backlit light source must be large enough so that the camera, without the object in the field of view will see a uniform amount of light. This is normally accomplished by using several flood lamps and shining the flood lamps onto a diffused surface (ground glass, or diffused white plastic, or frosted mylar), such that a uniform light source is created. Placing the object between the diffused surface and the camera will shadow the object into the camera with maximum contrast. Adjust the f-stop to the maximum value that the amount of light and integration time will allow. NOTE: For non-contact measurement of the objects' size, the magnification changes in relation to its distance from the camera to the object.

In selecting a lens, the magnification change as the object moves in the Z axis must be considered. The farther the lens is from the object the less the size changes as the object moves in the Z axis. The equation that relates the Z axis motion of the object to the change in lens-to-object distance is:

Z = change in object motion to/from the camera
L= Lens to object distance;

% area change = 200 * (Z/L + Z*Z/L*L)

For example, if the Z axis motion is 1/2 inch and the lens to object distance is 20 inches, then the change in size of the scene, as the computer sees it, is 5.25% in area. In comparing the MicronEye camera, lighting and processing, to other industrial systems that do gray scale processing, where lighting is not a dominant factor, there is usually a 300 to 1 cost trade-off. Placing more emphasis to correct the lighting so that a single threshold can be used produces a saving of 300 times.

F.1.3 ILLUMINATION SOURCES

Some of the common illumination sources are tungsten, quartz halogen, quartz iodine, fluorescent, and mercury or xenon arc lamps, as well as various flash lamps, lasers and LED sources. The common ways to configure these sources are: 1) illumination of the scene, 2) backlighting (shadowing) of the scene or 3) a combination of both, depending on the type of information desired from the camera. See figure F-1.

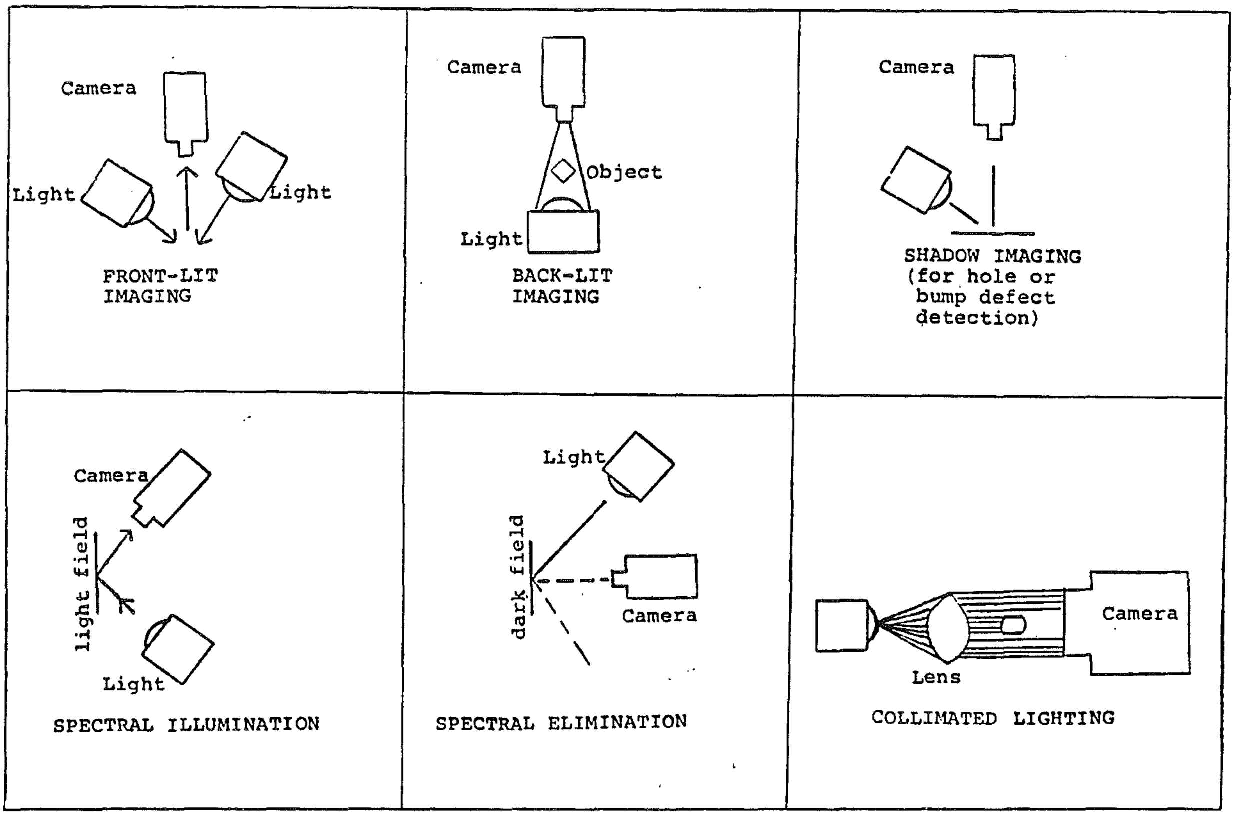


Figure F-1. Illumination Techniques

The light intensity required by the image sensor must be well defined in order to have even illumination of the scene, since the camera uses a common threshold for the entire scene, calling it light or dark. Only a small portion of light from the light source, via the scene, actually ends up in the sensor. Therefore, in choosing a suitable light source, such factors as even illumination versus threshold, f-stop and magnification of the lens, and the surface of the object (light or dark, diffused or specular) must be considered. Certain sections of the object may require spotlights to create an even illumination where a meaningful threshold scene can be produced. The amount of light coming through the lens is increasingly attenuated as the angle between the center of the lens going to the edge of the lens increases.

F.2 OPTICS

The MicronEye comes standard with a C-mount lens. Special applications may require the use of other lenses or filters which are not of the C-mount variety. C-mount adaptors are available for the more common lens types discussed below.

F.2.1 LENS TYPES

Three common lens types are the C-mount series, U-mount series, and L-mount series.

F.2.1.1 The C-mount Lens - The C-mount has a flange focal distance of 17.526mm (.690"). The flange focal distance is the distance from the lens mounting flange to the convergence point of all parallel rays entering the lens when the lens is focused at infinity. The C-mount lens is the work horse of the TV camera world.

Its format is designed for performance over the diagonal of a standard television camera videcon. This lens was selected by Micron because of its popularity and ease of availability. The mounting thread characterics are: 1" diameter, 32 threads/inch (machinist thread information 1"-32um2A).

Generally, this lens is an excellent choice for the OpticRAM. However, due to geometric distortion and field angle characteristics, short focal length lenses should be evaluated as to suitability for metrology (measurement) imaging. For instance, an 8.5mm focal length lens should not be used with an image sensor greater than 1/8" in length (the OpticRAM is .174") if the application involves metrology. Also, the majority of lenses should not be used wide open because of the light falloff characteristics.

The lens-to-OpticRAM distance has been established by using the flange focal distance dimension for fixed focal length lenses (non-adjustable focus). For close-ups, lens extenders will be required. The lens extender is used behind the lens to increase the lens to OpticRAM distance.

Spacer Lens (in mm) = Focal Length / Magnification

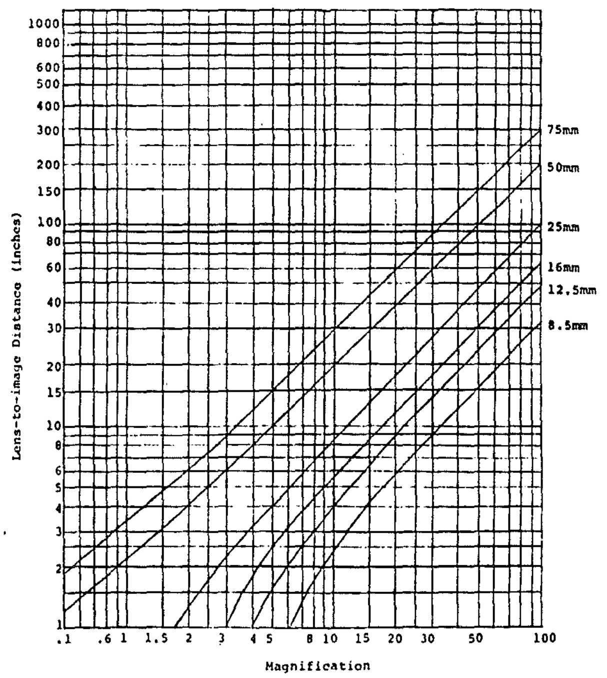
For a given lens, as magnification increases the distance between lens and focal plane decreases. Figure F-2 contains graphs of object distance versus magnification for common C-, U-, and L-mount lenses. These charts are a useful "ballpark" guide for lens focal length selection.

F.2.1.2 The U-mount Lens - The U-mount lens is a focusable lens having a flange focal distance of 46.52mm (1.7913"). The characteristic of the mounting threads is M42x1. This lens was primarily designed for 35mm photography applications. A C-mount to U-mount adapter can be purchased from most camera stores.

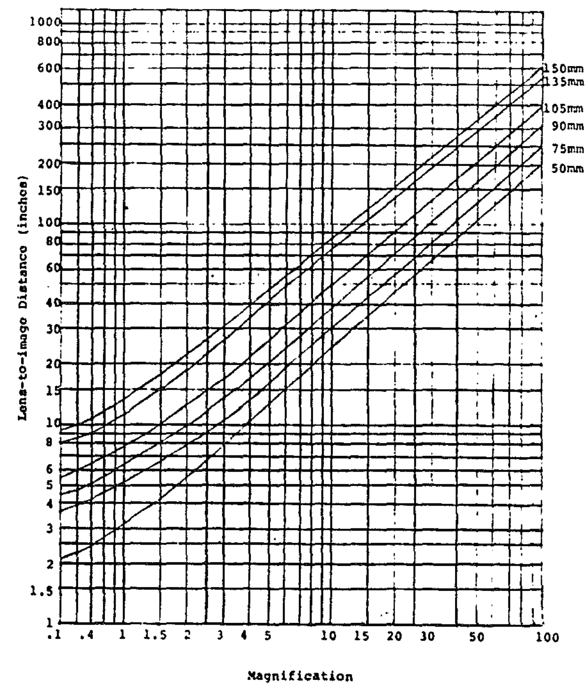
F.2.1.3 The L-mount Lens - The L-mount lens is a fixed-focus flat-field lens designed for committed industrial applications. This lens was originally designed for photographic enlargers. The flange focal distance is a function of the specification of each lens selected.

F.2.1.4 Microscope Lenses - There are standard microscope lenses available. These are to be used in applications where a magnification of less than one is required. However, a microscope lens to C-mount adapter in most cases needs to be individually designed because generally long lens extenders are needed.

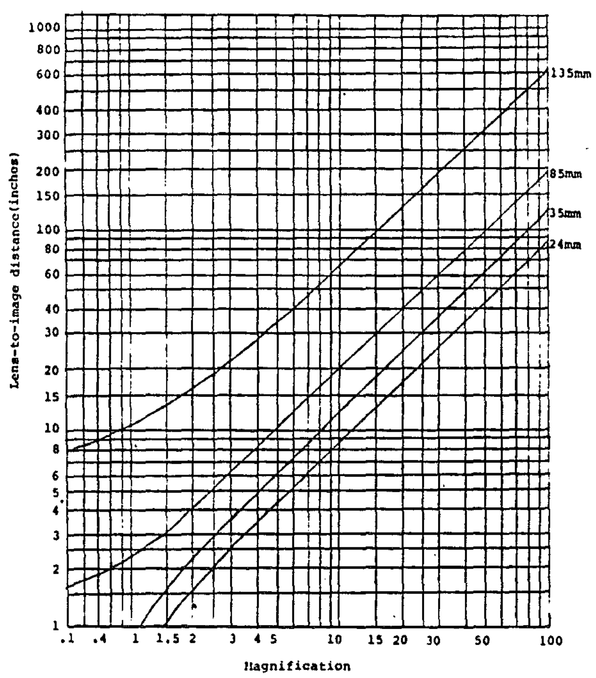
Lens Focal Length Selection
(C-mount)



Lens Focal Length Selection
(L-mount)



Lens Focal Length Selection
(U-mount)



DEPTH OF FIELD VERSUS MAGNIFICATION

(Element pitch = .3 mils)

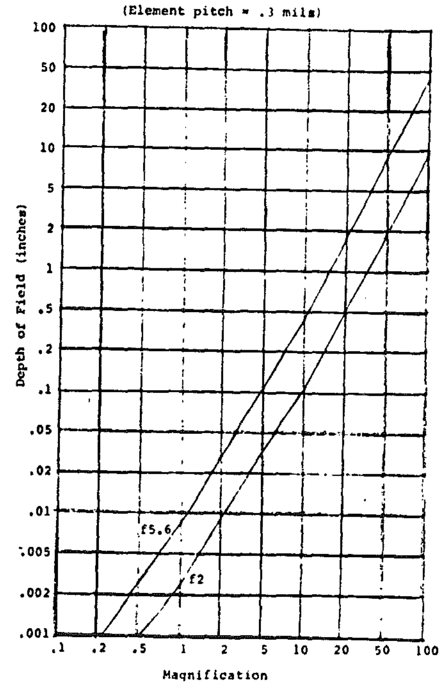


Figure F-2

F.2.2 TERMS AND DEFINITIONS

ARRAY SIZE: The physical size of the OpticRAM array from the 1st to the last pixel. The size can be looked at from many points of view. Care must be exercised in how the scene is projected onto the array via the optics.

Example: From the 1st pixel to the last pixel the column size = 174.016 mils and the row size of either section = 34.52 mils. The row dimension of the total array (of both arrays plus space pixels) = 73.764 mils.

FIELD OF VIEW (FOV): The maximum image dimension plus an allowance for alignment and part variation.

FOCAL LENGTH (F)> Type of lens, defined in millimeters. The present lens that is shipped with the camera is a 16mm C-mount lens.

F-STOP: The opening of the iris on the lens is calibrated in f-stops. Each higher number requires twice the light on the object for the same amount of light falling on the array.

LENS TO IMAGE DISTANCE (S'): The distance from the lens to the image (scene).

LENS TO OPTICRAM DISTANCE (S): The distance from the shoulder of the lens mount to the surface of the integrated circuit inside the OpticRAM package (plane of best focus). A lens extender may be required for objects that are closer to the lens than the normal lens design dictates.

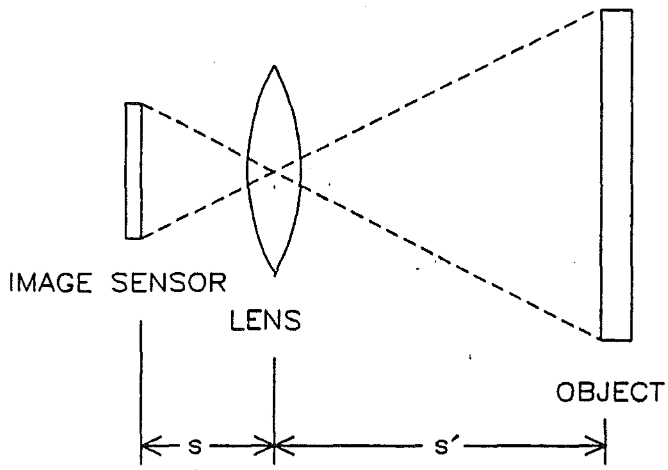
MAGNIFICATION (M): A camera lens is a transformation device that will make the image projection onto the array either smaller or larger depending on the lens and the distance away from the lens. The ratio of the object's true size to the size of the projection on the array is called the magnification.

PIXEL COUNT: A count of the number of pixel pitches that an aspect of the image traverses on the array, directly proportional to the magnification. In image space each pixel pitch represents a minimum resolution (image resolution).

RESOLUTION: The smallest size that is of interest in the field of view of the camera. The resolution is pixel pitch times the magnification.

Z AXIS CHANGE: The change in the distance between the camera and the object. As the distance between the scene and the camera decreases, the image projected onto the OpticRAM gets bigger, and therefore covers more pixels. As the distance between the scene and camera increases, the image gets smaller. If the distance between the camera and the scene is closer than the lens will focus, a spacer can be inserted between the lens and the camera to extend the focus range, or a different lens may be used to enhance the focus. The spacer length formula is used to determine the size of the spacer needed.

F.2.3 USEFUL EQUATIONS



M (magnification)	= Image Field of View / OpticRAM Size
	$= S'/S$
	$= F/(S-F)$
	$= (S'-F)/F$
F (Focal Length)	$= S'/(M+1)$
	$= (S*M)/(M+1)$
	$= (S+S')/(M+2+(1/M))$
	$= (M*(S+S'))/(M+1)$
S' (lens to object distance)	$= S*M$
	$= F*(M+1)$
	$= (S*F)/(S-F)$

Note: All dimensions are in millimeters.

Figure F-3. Simple Lens Equations.

METRIC CONVERSIONS:

1 INCH	= 25.4 Millimeters
1 INCH	= 2.54 Centimeters
1 FOOT	= 304.8 Millimeters
1 FOOT	= .3048 Meters
1 YARD	= .9144 Meters
1 Millisecond	= .001 seconds (msec)
1 Microsecond	= .000001 seconds (usec)

The resolution in the scene is dependent on the pixel pitch times the magnification. However, since the row pitch and the column pitch are different, this will correspond to a different magnification in the XY plane. Care must be exercised in selecting the dominant pitch.

PERCENT OF MAGNIFICATION CHANGE PER IMAGE AXIS = (Z/S') * 100

As the scene moves towards the camera, each scene axis gets bigger. As the scene moves away from the camera, each scene axis gets smaller. This equation relates the total Z axis motion (to and away motion of the scene as related to the camera) to on edge change providing the scene is still in focus.

F.2.4 LENS SELECTION CONSIDERATIONS

The selection of a lens requires the consideration of many parameters such as lighting, edge sharpness of the scene, Z axis motion of the scene, and distance from the camera to the scene. The lens provides a projection of the scene into the OpticRAM. This means if the lens is not selected properly or is misadjusted (out of focus, etc.) the information that the OpticRAM sees will not adequatly represent the scene, (for the threshold data slice of the scene will not represent the scene). One will be hard pressed to interpret what the camera is looking at. The choice of a lens in terms of focal length and field of view are directly affected by restrictions which may exist on the working distance of the camera. For example, a room size may restrict the camera from moving back far enough to have the scene in focus or fully captured.

The least resolvable element or increment in a measurement system may be the dominant factor, implying that more than one camera may be required in the system. In our system, with a built in threshold sensing technique, the resolution is equivalent to one pixel. The scene resolution is the pixel pitch times the object magnification.

Accuracy is the degree of exactness to which the measurement can be made. Under controlled conditions, accuracy can equal the resolution. When measuring the distance between two edges of an image, the accuracy is equivalent to one element per edge under conditions of having a sharp optical image of the object's edge. If lesser accuracy occurs, it is usually due to an unsharp edge, created by poor contrast between the object and the background, or due to dynamic aspects of object movements and integration time. However, by averaging one edge (or edges), the accuracy can be finer than the object resolution.

The following example discusses how one would select each component part for the camera and system configuration:

A disk is to be measured for its diameter on a translucent conveyor. The conveyor speed is 15 feet per minute. The disk size is .2 inches (with .02 inches of variation) with a height variation of 40 mils. This includes the conveyor thickness variation and vibration. NOTE: The limit tolerance in relation to the nominal size is .02 inches. However, the measurement of the part may require 10 times better resolution than the limit requires, 1% in this case.

In this example we will look at two ways to implement the solution. One solution is using a strobe light while the other solution is to analyze the motion of the part as it relates to the array. Figure F-4 describles the disk on the conveyor.

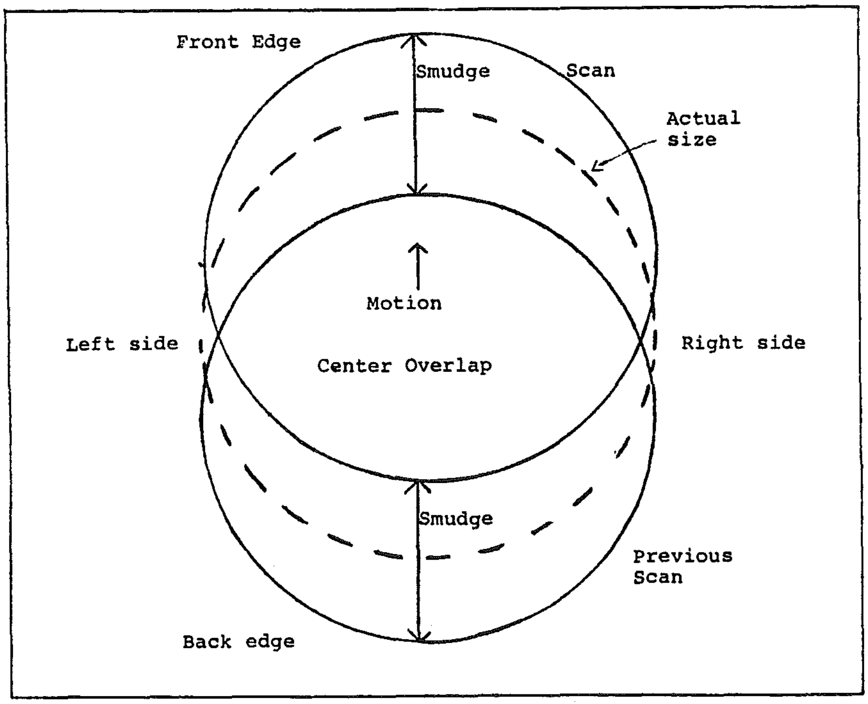


Figure F-4. Dynamics of Sample Problem

F.2.4.1 SOLUTION 1 - The Field of View (FOV) is .2 + (.02x2) = .240 inches. This gives a tolerance of the maximum disk size with .01 inches on top and bottom for location variation.

Calculate the magnification using the row dimension of 129 elements (34.52 mils). This is the dominant dimension in this case since the diameter of the disk need to be contained within the field of view of the camera. The column dimension of 514 elements is 174.016 mils.

M = .240 inches / .03452 inches = 7.0 MAGNIFICATION USING THE ROW AXIS

Resolution in the Row Axis = 7.0 x .26772 mils = 1.87 mils static resolution. Resolution in the column axis = 7.0 x.33858 mils = 2.37 mils static resolution. However, the 174.016 mils column axis times 7.0 = 1.218 inches. The FOV window at a magnification of 7.0 in space is .2416 inches by 1.218 inches. This gives a lot of space for the disk to move around, yet it can still be accurately measured.

This means that if we project the OpticRAM array into the object plane, each row axis pixel will have a pitch of 1.87 mils and each column axis pixel will have a pitch of 2.37 mils.

Using the chart for C-mount lenses (Figure F-2) for a magnification of 7.0, the lens to image distance for different lenses could be:

12.5	mm	=	2.5"
16	mm	=	3.75"
25	mm	=	5.75"
50	mm	=	14"
75	mm	=	22"

To find the image distance, find 7.0 on the magnification axis. Follow it until it intersects the lens types and read off the walues of the working distance on the other axis.

The disk height variation of 40 mils creates a change of dimension (magnification change). The percent of dimensional change is related to the height variation, divided by the lens-to-object distance times 100 ((Z/C')*100). If a lens extender is required, the extender length can be calculated by dividing the lens focal length by the required magnification. Units are in millimeters. The resulting image will focus when the lens focus control is set in its mid-point position. The following lenses can all be used to give a magnification of 7.0:

Focal Lens Selection		Lens to Image Distance	% Deviation Versus Z Change	Spacer Length
12.5 mm	=	2.5"	1.6 %	1.8 mm
16 mm	=	3.75"	1.1 %	2.3 mm
25 mm	=	5.75"	.67%	3.75mm
50 mm	=	11 "	.28%	7.1 mm
75 mm	=	17 "	.18%	10.7 mm

The 75mm lens will provide the least amount of magnification distortion. If there is enough physical space, then selecting the 75mm lens with a 10.7mm extender ring places the camera ajd lens 22" above the disk conveyor.

The dynamic property of the system is the smudge. As the part passes the field of view of the camera, the edge of the part is smudged across several pixels as the camera integrates the light entering the camera. Since the part is traveling at 15 ft. per minute, what must the integration time be so that only one pixel will be smudged? Converting feet per minute to inches per second =

15 ft/min * 12 inches/ft * 1 min/60 sec = 3 inches/sec

As calulated before, 1 pixel of the row dimension = 1.87 mils. This means that for each frame scan the part can only move 1.87 mils per scan and since the part travels at 3 inches/second, then:
.00187"/scan * sec/3" = .000623 sec/scan = 623 microsec/scan

This is clearly too fast for the camera, which can operate at only 4 scans per second. What is the solution? At each scan, the disk moves:

3 inches/sec * .25 seconds/scan = .75 inches/scan

The part is only .24 inches in diameter. This means for every scan, the part can move approximately four times its diameter through the field of view of the camera. The solution is to place a photo transistor looking across the conveyor to an LED. As the disk blocks the LED light to the photo transistor, it triggers a strobe light that is mounted below the translucent conveyor. Select a strobe light with a flash of peak energy shorter than 613 microseconds. The setup is shown in Figure F-5.

As the strobe light flashes, it also triggers the software that brings in the camera data. The camera integration time is directly linked with the part pitch. However, care must be taken so that the integration time does not exceed where the ambient light or dark current rises above the camera threshold. If the conveyor stops or no parts come down the conveyor, this fact must be sent to the software where it will input data from the camera and throw it away (dummy read) to refresh the pixels to keep the camera in the alert condition. By having a photo-transistor that precedes the strobe photo-transistor, the first photo-transistor does a dummy read. This arms the camera and after the flash the camera will contain the correct data. A strobe light is an effective tool to freeze action in dynamic situations. However, in many situations a strobe light may not be required.

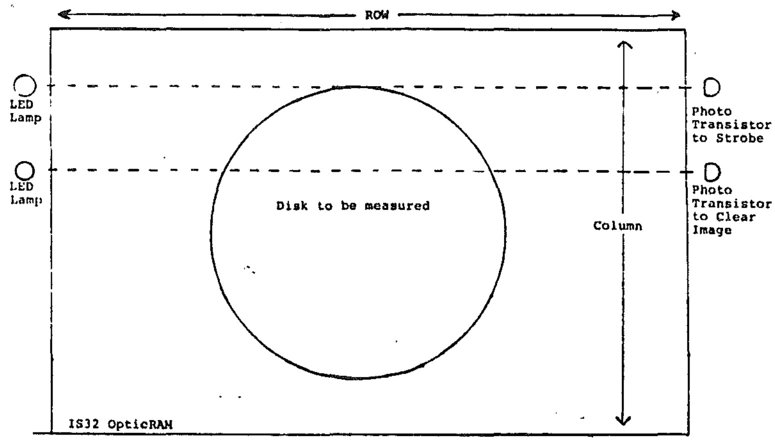


Figure F-5. Triggering Camera Based on Part Location

F.2.4.2 SOLUTION 2 - This solution shows how to approach the problem without using a strobe light. Assume that an incandescent light is used to backlight the part, and the OpticRAM is operated at 120 frames per second, which translates to 8.33 msec/frame. The part is still moving at 3 inches/sec, as we calculated in the previous solution.

Calculate the distance over which the disk is smudged:

.083 sec/scan * 3 inches/sec = .025 inches/scan smudge

From scan to scan, the part moves .025 inches. Therefore, the field of view needs to be the size of the part (.24") plus 2x the smudge to allow for the smudge of the leading and trailing edges. Dividing the FOV (.29") by the row dimension (.03452) we are able to calculate a magnification constant of 8.4.

Assuming that a 75mm lens was selected gives a distance of 26" from camera to scene and a deviation of .15 percent of Z-axis magnification change with a spacer of 8.9mm. The row axis resolution is determined by the product of .268 mils * 8.4 giving 2.25 mils. The column axis resolution is the product of .33858 mils * 8.4 giving 2.84 mils.

This means that each edge has a gradient (in this case) of 12 pixel smudge motion. See figure F-6. If the threshold is centered to the midpoint of the light amplitude, the 12 pixels that are smudged will go to 6 pixels (actual edge) on each side. The actual size can be realized by either changing the intensity of the lamp via a fixed threshold or by changing the threshold and holding the intensity of the lamp constant. However, since size is directly related to light versus threshold levels, the lamp output needs to be accurately stablilized.

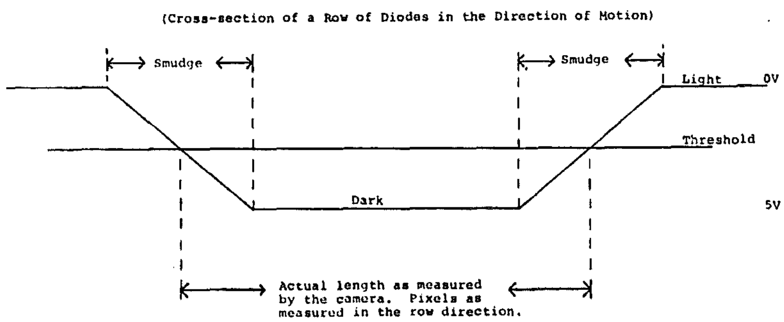


Figure F-6. Length Measurement of a Moving Object

We have talked so far about what happens to the middle of the part, now we need to talk about what happens at the left or right edge of the part in a dynamic situation. (Refer back to Figure F-4)

Assuming the right most edge or left most edge covers a pixel, the question is, for what duration is the pixel covered? Assume from scan to scan that the disk moves .025 inches. Using the formula for a chord of a circle (Figure F-7), we need to determine the error at point A and point B.

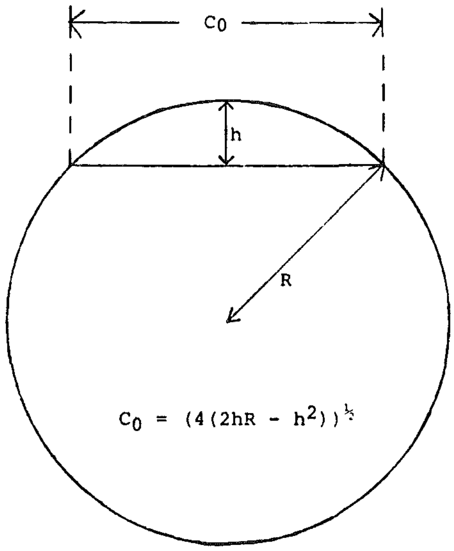


Figure F-7. Chord of a Circle Equation

From earlier calculations, the pixel width in the column axis of 2.84 mils, with .1 inch radius is:

Co = SQRT(4(2*.00284*.1 - .00284²)) = .0476

.0476" * scan/.025" = 190% of the time the A and B pixels are dark suggesting that the error at points A and B is neglible. (190% of the time is an awkward way of saying that the disk travels only about half the distance between points A and B in one scan period. When the percentage exceeds 60%, we can say for certain that the left/right edge pixel represents the part. Motion is always a problem even in static situations because between the camera and the scene there is vibration which may require careful attention to detail.

Once data is captured either by a strobe lamp or by back lighting (shadowing) and stored in the computer memory, statistical averaging is then done in order to improve the data. EXAMPLE: Using the formula to find how many row pixels should come dark at the same time at the entry and exit. The row resolution is .00225 inches per pixel. Using the formula for a chord of a circle:

Co = SQRT(4(2*.00225 * .1 - .00225*.00225)) = .042 mils

.042 mils * pixel/.00225 mils = 18 pixels

This indicates that if 18 pixels are averaged at the max/min points then the resolution and accuracy can be increased by a value of:

SQRT(number of pixels) / 2

Find the midpoint of the circle, then average the 10 pixels on either side (20 pixels)

SQRT(20)/2 = 4.5/2 = approx 2

This suggests a half a digit increase in accuracy.

1.84 mils/.2 diam * 100 = .92% + .23% for Z axis Motion = 1.15%

1.15% / 2 = .575% resolution (after calibration).

From disk to disk, one should be able to resolve each disk to about .6% The design goal was 1%. If it is desired, an out of round figure of merit can also be calculated:

area = pi * R*R

circumference = 2 * pi * R

area/circumference = R/2

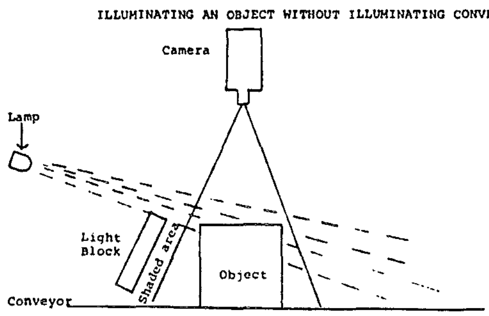
Adding the area pixels and dividing by the edge pixels, should give a number close to half the radius pixel as a ratio. The ratio should hold. If it does not, this is an indication of out of roundness. One can also sort parts for rough out-of-round tolerances.

F.3 OTHER CONSIDERATIONS

Since backlighting is a problem on most conveyors then using a structured light may be the solution.

In general, arbitrary lighting of the environment is not acceptable because it can result in low-contrast images, specular reflections, shadowing, and extraneous details. A well-designed lighting system illuminates the scene so that the complexity of the resulting image is minimized, while the information required for inspection or manipulation is enhanced.

Once the data of the scene is in memory, further algorithms can be employed to extract useful feature data, such as: modeling the algorithm of an object to extract the following features: area, parameters, centroid, ratio of minimum to maximum moment of inertia, axis of least moment of intertia, diagonal length of a bonding rectangle, and simple dimensional measurements at key points that can resolve a problem.



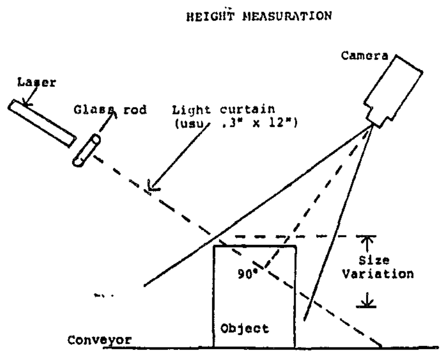


Figure F-8. Other Useful Lighting Techniques

APPENDIX G

HARDWARE DESCRIPTION

G.1 TIMING GENERATION CIRCUIT

This circuit generates the timing signals for the operation of the MicronEye. A CMOS oscillator circuit generates the basic clock signal. This signal is divided down to produce the various possible baud rates and the timing signals which drive the IS32. The baud clock signals sequence the Interrupt Generator and the Transmitter circuit.

The oscillator circuit consists of a CMOS inverter, a crystal, two resistors and two capacitors. It generates a 4.9152 Mhz signal which is buffered by an inverter (A4,pin2). This frequency is divided in half by a D flip-flop at A3-5, and again at A3,pin9. Both outputs lead to baud rate selection pads. Flip-flop output A3,pin9 also connects to the clock input at B5,pin10. IC B5 does successive frequency divide-by-twos. The various outputs lead to other baud rate selection pads. Pads 5 through 8 are baud Clock signals. One of these baud clocks is used in the transmitter and Interrupt Generator circuit. Pads 1 through 4 are clock signals that are 16 times higher in frequency than the baud clocks. One of these 16x clock signals is used in the receiver circuit.

The output of B5,pin7 drives the Optic RAM timing circuitry which generates RAS, CAS and R/W (read/write). The outputs of inverters A4,pin4 and A4,pin6 are identical. A4,pin4 drives the RAS input to the Optic Ram, and is buffered separately because it is required to drive its signal through the ribbon cable if a Bullet MicronEye is used. A4,pin 6 is identical to the RAS signal, but it is used as inputs to other camera circuitry and is labeled RAS'.

When the camera is not in an Interrupt mode (i.e., is not transmitting data from the OpticRAM), CAS and R/W are disabled. The signal INT is low and INT/ (The "/" after a signal name indicates the complement of the signal.) is high, so the AND gate driving CAS remains low and the OR gate driving R/W remains high.

During an Interrupt cycle, INT goes high and INT/ goes low, enabling CAS and R/W. RAS' goes low with RAS which latches the Row address into the OpticRAM. RAS' passes through a delay line consisting of 2 inverters and an RC network, and then causes CAS to go low, latching the Column Address into the OpticRAM. At this time the R/W signal is still high, so the accessed pixel is read out. After another delay period, R/W goes low, which causes the OpticRAM to write data into the accessed cell. The addressing circuitry presents the proper data on the Data In pin to make sure that 5 volts is written back into the cell.

When RAS' goes high, the Interrupt cycle is terminated and CAS and R/W are disabled.

G.2 COMMAND RECEIVER CIRCUIT

G.2.1 General Description

The serial command line carries the camera commands from the computer to the camera. This data enters the command receiver circuit one bit at a time. The first bit to arrive is the start bit, followed by 8 data bits and then the stop bit. The start bit enables the input shift register and starts the shift register clock. The clock is initially low. When it goes high, the start bit, which is a high, is latched into the first of eight data positions in the shift register. When the clock goes low, the first data bit arrives at the shift register input. On the rising edge of the clock, the shift register "shifts" the high start bit from position 1 to position 2, and shifts the first data bit from the shift register input, into position 1. As each successive bit arrives, each one is shifted into the shift register on the rising clock edge.

When the start bit finally shifts into position 8, the camera has received all of the command information., The first six data bits are transferred from the shift register into a latch (memory) called the Command Register. The clock is disabled and the shift register is cleared. Now the six camera command bits are in the Command Register and the receiver is ready to get another command.

G.2.2 Circuit Description

The start bit from the computer appears as a high level at the output of the inverter at G1-12. The rising edge of this start bit clocks flip-flop F1-9 to the high state. This line clears the reset on IC's F2 and F4. F2 is a shift register and F4 is used as a divide-by-16 counter. F4's input is a clock whose frequency is 16 times greater than the baud rate (16x clock). After eight clock cycles, the counter output (F4-11) goes high, shifting the start bit into position 1 (F2-3) of the shift register. 8 clock cycles later, the shift register clock at F4-11 goes low and the first data bit arrives. 8 clock cycles later F4-11 goes high, shifting the data bit into position 2 (F2-3), and the start bit into position 1 (F2-4). This process continues until the start bit reaches position 8 (F2-13). The high start bit causes a low at the flip-flop RESET input (F1-13). This causes the flip-flop Q/ output (F1-8) to go high, latching the serial register data into the Command Register, F3. At the same time, the flip-flop Q output (F1-9) goes low, resetting F2 and F4.

G.3 ADDRESS REGISTERS

This circuit latches the Row, Column and Refresh pointers for the OpticRAM addressing.

G.3.1 General Description

Address registers C4 and C3 hold the RAS and CAS addresses, respectively. These registers are enabled only when the camera is to fetch and transmit a single bit of information from the OpticRAM. This fetch operation is initiated by the INT signal going high, and is called an Interrupt cycle. An Interrupt cycle is started on the rising edge of RAS' and is ended on the next rising edge of RAS'.

When the camera is not in an Interrupt cycle, the Refresh Register, C2, is active. This register increments the Row Address from 0 to 255, thus performing a refresh operation on the OpticRAM.

All three Registers have tri-state outputs and only one register is active at any one time. The selected register drives its data onto a common bus called the Present Address bus. The Present Address passes through the descramble and soak circuitry, to the OpticRAM, where it is used to select a Row or Column. The Present Address bus also connects to the Address Circuit, where a value of 0, 1 or 2 is added to the Present Address value.

The resulting sum is driven out of the adder onto the Next Address bus. This bus connects to the inputs of each of the Address Registers. The value on the Next Address bus is latched into the selected Address Register and then that Register is disabled.

G.3.2 Circuit Description

When the MicronEye is not in an Interrupt mode, the INT signal is low and the INT/ signal is high. this forces the Enable inputs (active low) to C3 and C4 to remain high. When RAS' and Td go high and INT is high, the NAND gate output at A1-3 is low, enabling C2. C2 drives its data onto the Present Address bus. The data propagates to the OpticRAM and to the Adder circuit. The Adder circuit adds a 1 to the value on the Present Address bus and drives the sum onto the Next Address bus where it appears at the inputs to C2. When RAS' goes low, the descrambled Present Address is latched into the OpticRAM, and the output of A1-3 goes high, clocking the value on the Next Address bus into C2 and turning off the outputs.

During an Interrupt cycle, INT/ is low, so C2 is disabled. The rising edge of RAS' initiates the Interrupt cycle, so initially RAS' (and Td) and INT will be high, driving the NAND gate A1-8 low and enabling the Row Register, C4. C4 drives its value onto the Present Address bus. Some value, either 0,1 or 2 is added to it in the Adder and the sum is placed on the Next Address bus. When RAS' goes low, the Next Address value is latched into the Row Register, the Row Register outputs are disabled and the Column Registers outputs are enabled. The data from the Column Register, C3, is driven onto the Present Address bus, through the Adder Circuit (where it may be incremented) and onto the Next Address bus. It also propagates to the OpticRAM where it is latched when CAS goes low. When RAS' goes high, the value on the Next Address bus is latched into the Column Register and it's output drivers are disabled.

The Array Selection circuit determines whether one or both arrays are transmitted. If 2ARRAY/ is high, the output of the OR gate (B4-11) is always high and the Row Register value (C4) will never be less than 128. Thus, only the second array (rows 128 to 255) will be addressed. If 2ARRAY/ is low, however, the OR gate will appear transparent and the value on the Next Address bus line D7 will drive onto C4. This means all addresses from 0 to 255 will be selected and both arrays will be transmitted.

G.4 ADDRESS DESCRAMBLE, SOAK/, AND DIN/DOUT CIRCUI

G.4.1 Address Descramble

The internal circuitry in the OpticRAM scrambles the Row and Column Address values when accessing a cell. The Address Descramble circuit reverses the OpticRAM scramble. It transforms the Data from the Address Registers into a new address, which the OpticRAM decodes to access the desired pixel.

The circuit consists of 2 inverters, 3 Exclusive-OR's and a multiplexor (D2). The invertors and Exclusive-ORs provide the descramble function on the Row and Column addresses. The multiplexor selects between the descrambled Row and Column address' at the appropriate time and drives the address to the OpticRAM. The multiplexor uses RAS' to determine which address is selected. If RAS' is high at the multiplexor SELECT input (D2-1), the B inputs, which are the descrambled Row Addressinputs, are selected. When RAS' is low, the A inputs, or descrambled Column Address inputs, are selected. The descramble truth-table is available in the IS32 data sheet.

G.4.2 SOAK/

The purpose of the SOAK/ circuit is to prevent the refresh from reaching the OpticRAM. The OpticRAM is light sensitive only when it is not being refreshed. When INT is low (which is when the Refresh Register is active) and SOAK/ is low, the output of the NOR gate, B3-13, is high. This sets the multiplexor Enable input (D2-15) high and drives the multiplexor outputs low. The high NOR gate output at B3-13 also forces a low at the inverter output E3-8, which forces the outputs of the four AND gates (D4-3,6,8,11) low. Thus, the OpticRAM address inputs remain low, and the refresh function is performed only on address 0, i.e., only Row 0 gets refreshed.

When SOAK/ goes high, the multiplexor and AND gate outputs are enabled and the refresh addresses reach the OpticRAM and the entire chip is refreshed, making it insensitive to light. The SOAK/ command can be thought of as an electronic shutter control.

G.4.3 Din/Dout Circuit

This circuit controls the input to the OpticRAM Din (Data In) pin and also detects when a cell in the OpticRAM has been "exposed" to the low state.

For a cell to be light sensitive, it must be initially charged to +5 volts. This is done by writing data into the cells. Due to the operation of the OpticRAM internal circuitry, a logic "1" must be written into all cells with row addresses between 0 and 127, and a logic "0" must be written into all cells with row addresses between 128 and 255. The most significant row address bit, Q7, is latched (during interrupt cycles) by flip-flop E4 on the falling edge of RAS'. When the row address is between 0 and 127, row address bit Q7 is a 0, and when the row address is between 128 and 255, row address bit Q7 is a 1. The inverting output of flip-flop E4 (E4-8) is connected to the Data In pin on the IS32. Thus, the proper data will be presented to the OpticRAM to write each cell to +5 volts.

The Exclusive-OR gate (E2-8,9,10) compares the data out of the OpticRAM with the data tht was read into it. Notice that the input to the Exclusive-OR gate at E2-8 is the complement of the value at the Din pin. Thus, if the OpticRAM cell being read out is still high, the two flip-flop outputs, E4-8 and E4-9, will be at opposite levels and the output of the Exclusive-OR (E4-10), will be high. Conversely, if the cell has been exposed to the low state, the two inputs to the Exclusive-OR will be the same and it's output, E4-10, will be low. The output of E4-10 propagates to the Transmitter circuit, where it is latched and transmitted to the computer.

G.5 TRANSMITTER AND INTERRUPT GENERATOR CIRCUIT

This circuit transmits the serial information, inserting start and stop bits where appropriate, and generates the INT and INT/ signals for fetching pixel information.

G.5.1 General Description

At the heart of this circuit is the ripple Counter, D1. D1 is enabled when the MicronEye has been commanded to transmit data. It inhibits the Interrupt circuit when start and stop bits are being transmitted, and enables the Interrupt circuit when it is transmitting data. The Transmitter is clocked by the baud clock. On each baud clock cycle, only one start, stop or data bit is transmitted.

The Interrupt Generator is enabled by both the ripple counter (D1) and the baud clock, but the Interrupt cycle is clocked by RAS'. Remember the purpose of the Interrupt cycle is to fetch a single pixel for transmission, and only one pixel can be tansmitted on each baud clock cycle. The rising edge of the baud clock enables the Interrupt circuit. The next rising edge of RAS initiates the Interrupt cycle, causing a pixel to be read from the OpticRAM. The INT/ signal feeds back into the Interrupt circuit, resetting the Interrupt enable. When RAS' goes high again, the Interrupt cycle is terminated. The next rising edge of the baud clock will enable the Interrupt circuit again (unless a start or stop bit is to be transmitted). Thus, only one pixel is transmitted during each baud clock cycle.

The WIDEPIX circuit is used to help compensate for the 2.5 to 1 aspect ratio of the OpticRAM. If the optic data is displayed on a screen with a 1 to 1 aspect ratio, the image will appear to be squeezed in the horizontal direction. The WIDEPIX circuit helps compensate for this by causing each pixel to be transmitted twice, doubling the width of the image. The circuit is enabled when the MicronEye is transmitting and the WIDEPIX command bit is high. This causes the flip-flop output A2-5 to toggle on every baud clock cycle.

This flip-flop inhibits the Interrupt cycle on alternate baud clock cycles. During baud clock cycles in which the Interrupt is inhibited, the pixel from the previous Interrupt cycle is transmitted again.

G.5.2 Circuit Description

When the MicronEye is not in a Transmit mode, the XMIT signal is low, driving the ripple counter RESET input high (01-15). This puts the ripple counter in a reset state in which output Q0 (D1-3) is high. The high on Q0 drives the RESET input at E5-1, low and the flip-flop Q/ output E5-6, high. E5-6 is the data transmission line to the computer. The high level of Q0 (D1-3) also drives the flip-flop data input (E5-12) high (let's assume LINE is low). This prevents any Interrupt cycles from occuring.

When the MicronEye receives a Transmit command, XMIT goes high, XMIT/ goes low and the ripple counter D1 is enabled. D1 is clocked by the rising edge of the BAUD clock. The first clock causes Q0 (D1-3) to go low and Q1 (D1-2) to go high. This sets the transmit line E5-6 low, representing the start bit. The first clock also forces a high at flip-flop data input, E5-12. The baud clock is delayed through an RC network (R3 and C2) and now clocks the high input at flip-flop E5-12 to the output at E5-9. This forces a high on the input of the Interrupt flip-flop, A2-12. When RAS' goes high at the flip-flop clock input A2-11, it initiates the Interrupt cycle. INT goes high and INT/ goes low. INT/ is an input to the AND gate. B2-1 and forces the flip-flop RESET inputs (E5-13) low. This forces A2-12 low, so on the next rising edge of RAS', the Interrupt cycle is terminated. INT/ going high clears the RESET at e%-13 and another interrupt will occur when the baud clock goes high again.

When the WIDEPIX bit is set high, the RESET input at A2-1 is high, enabling the flip-flop. The output toggles on each interrupt request and inhibits every other interrupt cycle by bringing the RESET input A2-13 low.

The LINE and LINE/ signals indicate that the Column Address Register has reached terminal count. These signals inhibit further interrupts from occurring during data bit transmissions, so the value of the last accessed data bit is repeated to complete the current byte transmission. This guarantees that the next byte transmitted contains information from the next row, i.e., no single byte will contain information from two rows. When the stop bit is to be transmitted, LINE at E1-5 causes an Interrupt Request and LINE/ at A1-4 ensures that the Interrupt flip-flop is enabled. This "dummy" interrupt is used to increment the Row Address Register. The pixel that is accessed during this cycle is blanked by the transmission of the Stop bit.

G.6 ADDER AND END-OF-FRAME CIRCUIT

This circuit adds the proper increments to the Row, Column and Refresh Registers and generates signals indicating End-of-Line and End-of-Frame in the OpticRAM.

G.6.1 General Desription

When any of the Address Registers drive a value onto the Present Address bus, the Adder circuit receives this value, adds a 0, 1 or 2 to it (depending on the control inputs) and drives the sum onto the Next Address bus. The control lines are RAS', LINE, ALTBIT and INT. When the Refresh Register is active, the INT line causes a "1" to be added each cycle. During interrupt cycles, the Row and Column Registers are active. The Adder sequences these registers through the OpticRAM in a column-fast mode, i.e., the Adder adds a "zero" to the Row Address and a "one" to the Column Address until the end of the column (End-of-Line) is reached. the Adder then adds a 1 to both the Row and Column, thus incrementing the Row Register and resetting the Column Register to zero.

The ALTBIT input simply adds another "1" to the value on the Present Address bus during Interrupt cycles,, thus the Row and Column Reigsters are incremented by 2 rather than 1.

G.6.2 Circuit Description

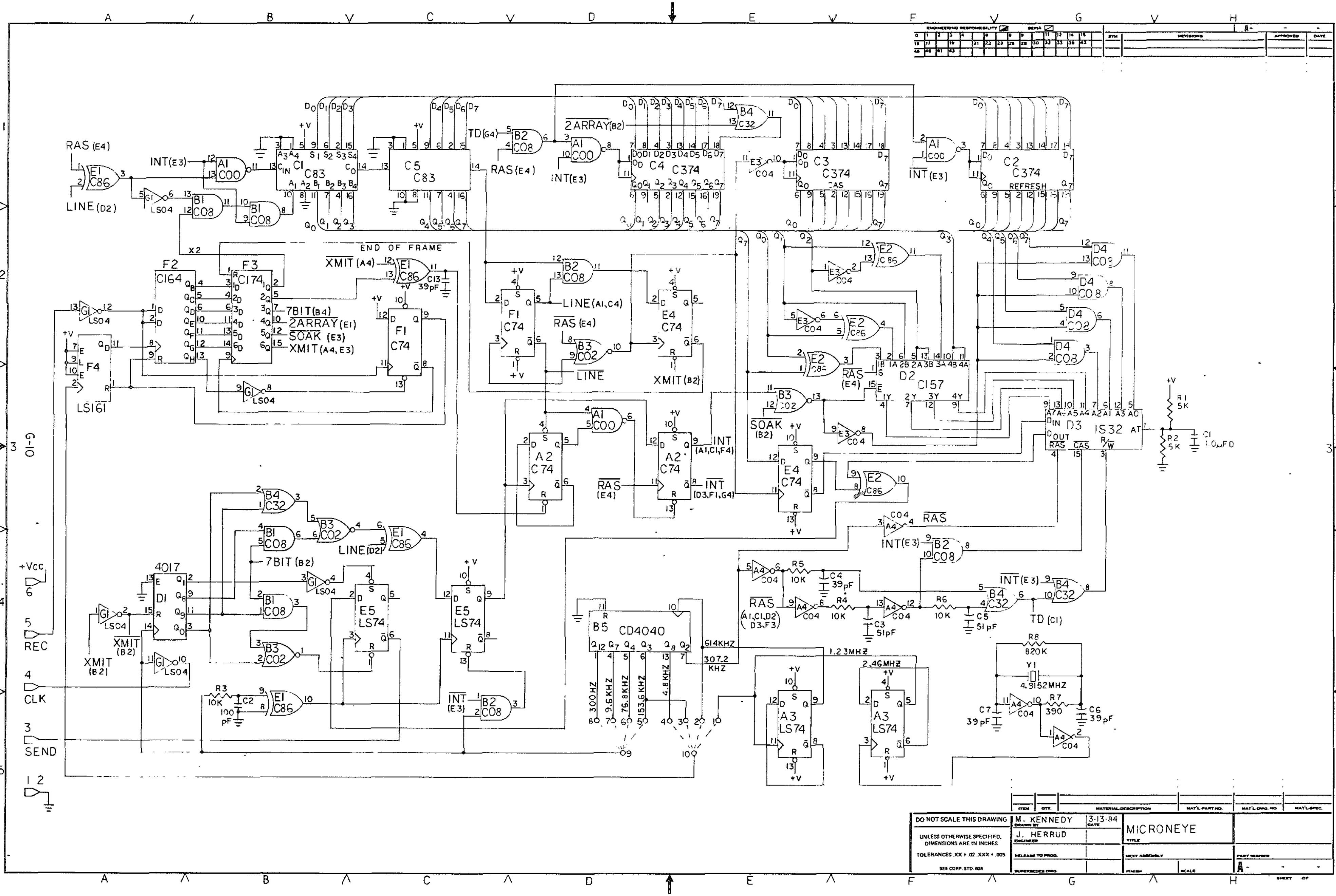
During Refresh cycles, the INT signal is low, forcing the Carry In input to the Adder (C1-13) to be high. Thus, a value of "1" is added to the value on the Present Address bus on each Refresh cycle.

During Interrupt cycles, the INT signal is high. Let's assume LINE and ALTBIT are low. For the first half of the Interrupt cycle, the Row Register is active and RAS' is high, forcing the Carry-In input of the Adder to be low. A zero is added to the Present Address value, so the Row Register address remains unchanged.

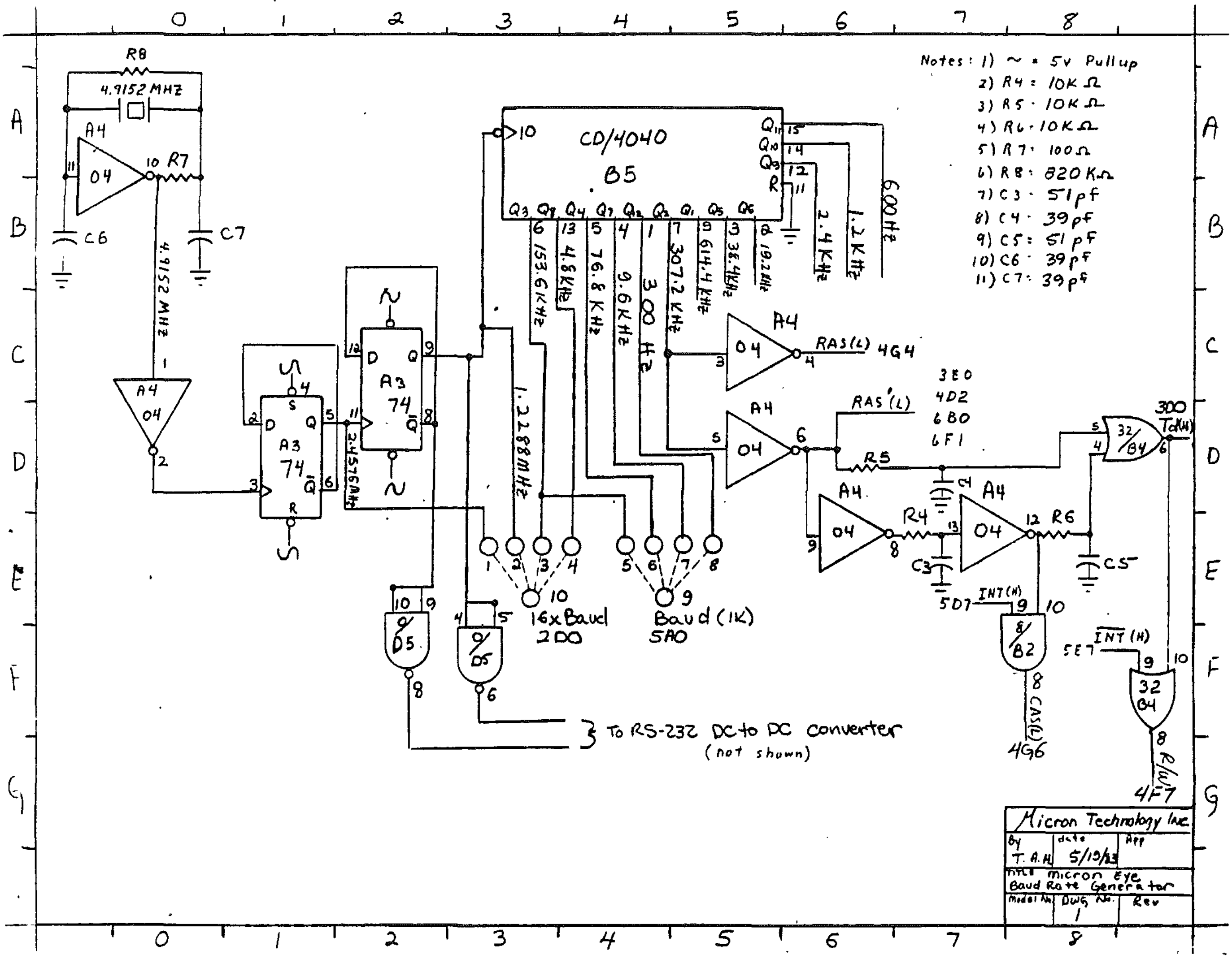
When RAS goes low, the Column Register is active and a high is driven onto the Adders Carry-In input. A "1" is added to the Present Address bus and the incremented value is stored back into the Column Register. Thus, the Registers count down the columns in the same row.

When the last cell is acessed, the Column Address is at the Adder's terminal count of 255, setting the carry-out signal high. (The Column Register is incremented to zero). The high Carry-Out signal is latched by the rising edge of INT/ at F1-2, and forces the outputs, LINE and LINE/ (F1-5 and 6) to the asserted state. These signals cause the next Interrupt cycle to occur during the transmission of the next stop bit. The LINE input to the Exclusive-OR at E1-2, reverses the effect of RAS' on the Adders' Carry-In input. Thus, a "1" is added to the Row Register and a "0" is added to the Column Register. The pixel that is accessed during this Interrupt is blanked by the stop bit transmission. At the start of the next Interrupt cycle (when RAS' goes high), LINE and LINE/ are reset and the circuit sequences down this next row.

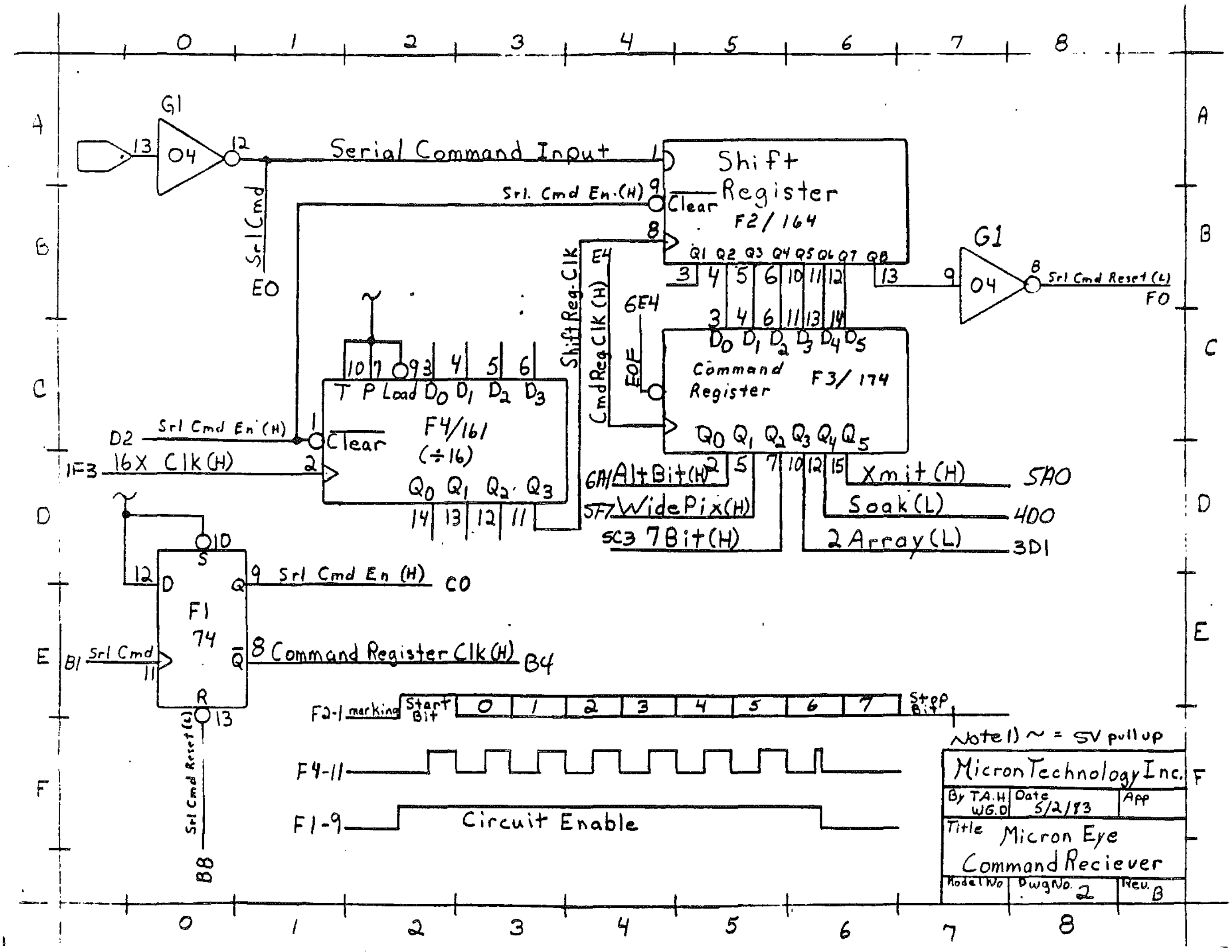
Let's assume the last pixel in the OpticRAM has been accessed and LINE has been set. The Column Register has been incremented to zero and the Row Register is at terminal count (255). The next Interrupt cycle forces the Row Register to drive its value of 255 onto the Present Address bus and to the Adder. The Adder adds a "1" to it and drives a value of zero onto the Next Address bus and also sets the Carry-Out (C5-14) high. The Carry-Out and LINE signals force the output of the AND gate (B2-11) high, thus setting the flip-flop input (E4-2) high also. When RAS' goes low, the NOR gate (B3-10) goes high, clocking E4-3. The Q/ output of the flip-flop (E4-6) goes low. This is the End-of-Frame signal. The EOF is connected to the reset input of the Command Register, so a low on the EOF line resets all of the command lines to zero. The XMIT command line is connected to the flip-flop reset (E4-1), so when XMIT goes low, flip-flop E4-1 is reset and the EOF signal is reset high. Note that the Row and Column Registers both now hold a value of zero.


DO NOT SCALE THIS DRAWING
UNLESS OTHERWISE SPECIFIED, DIMENSIONS ARE IN INCHES
M. KENNEDY
3-13-84
J. HERRUD
MICRONEYE

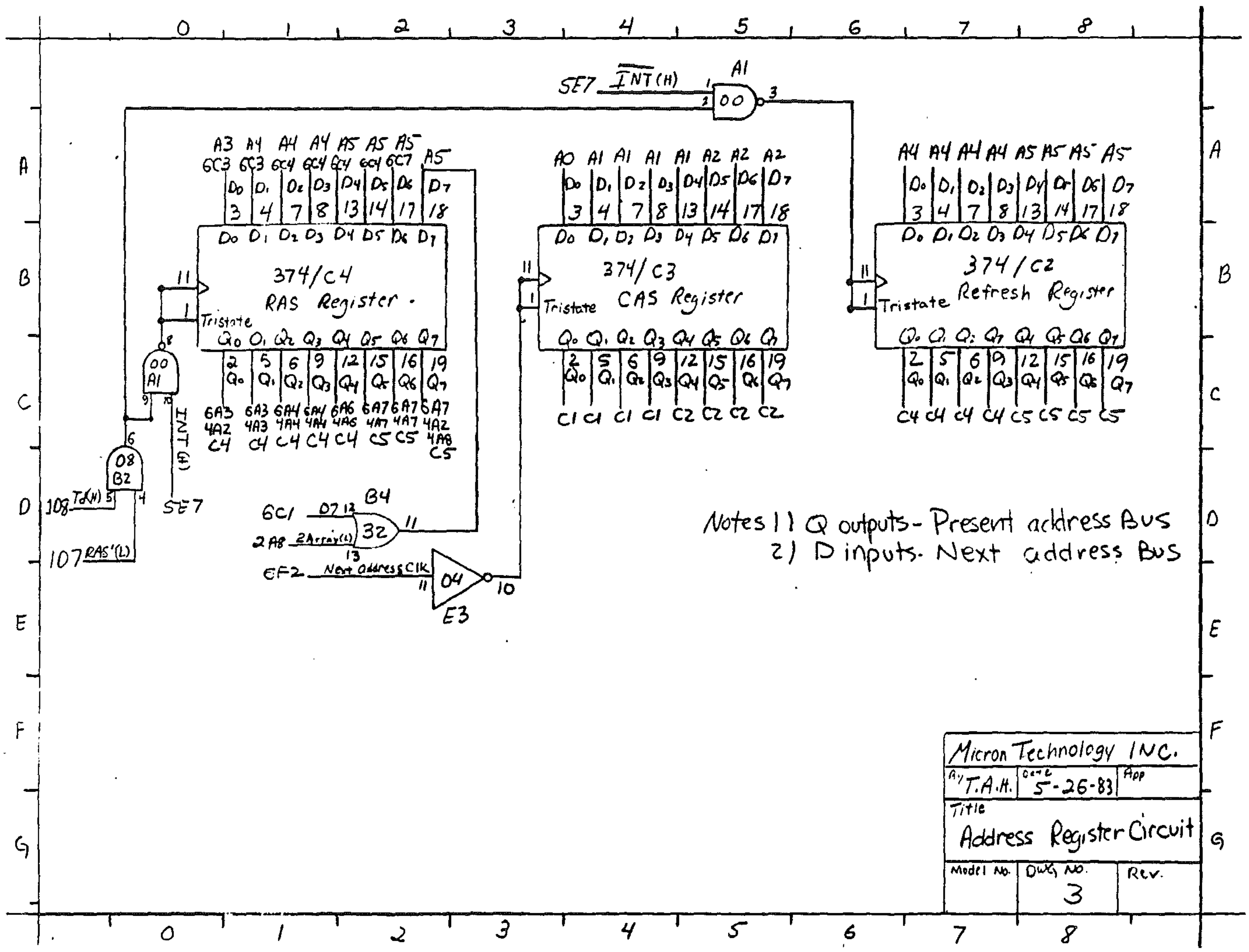
TIMING GENERATION CIRCUIT



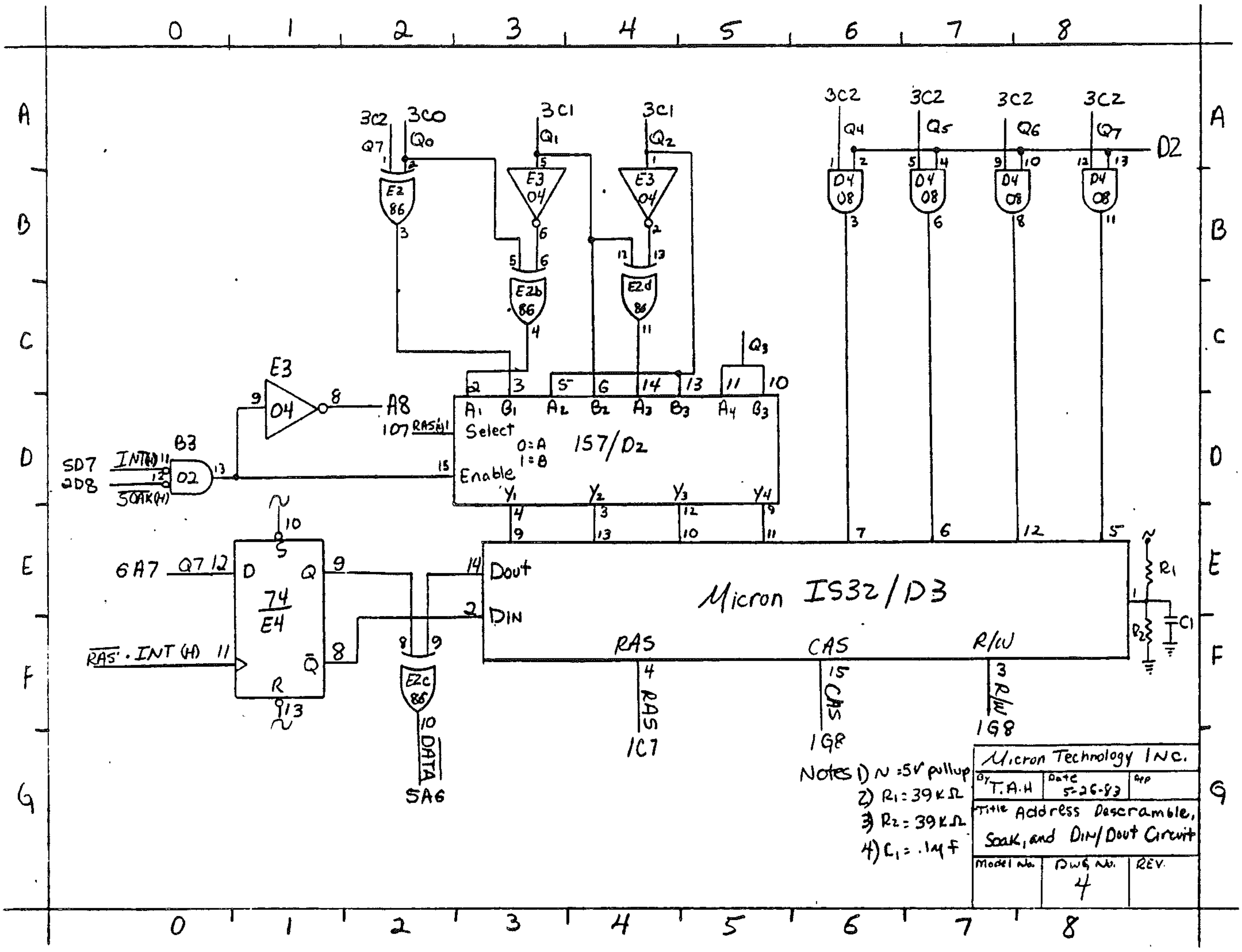
COMMAND RECEIVER CIRCUIT



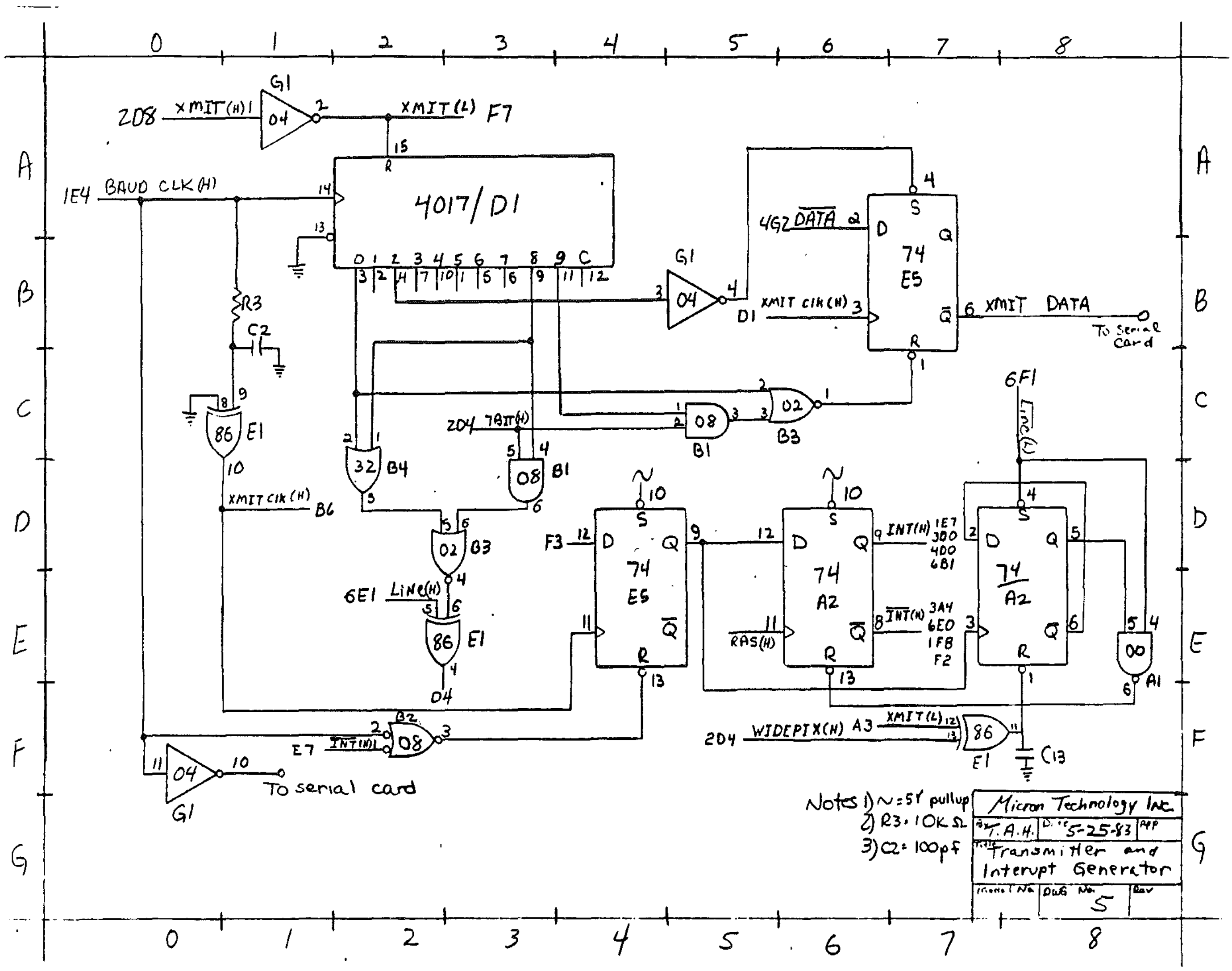
ADDRESS REGISTER CIRCUIT



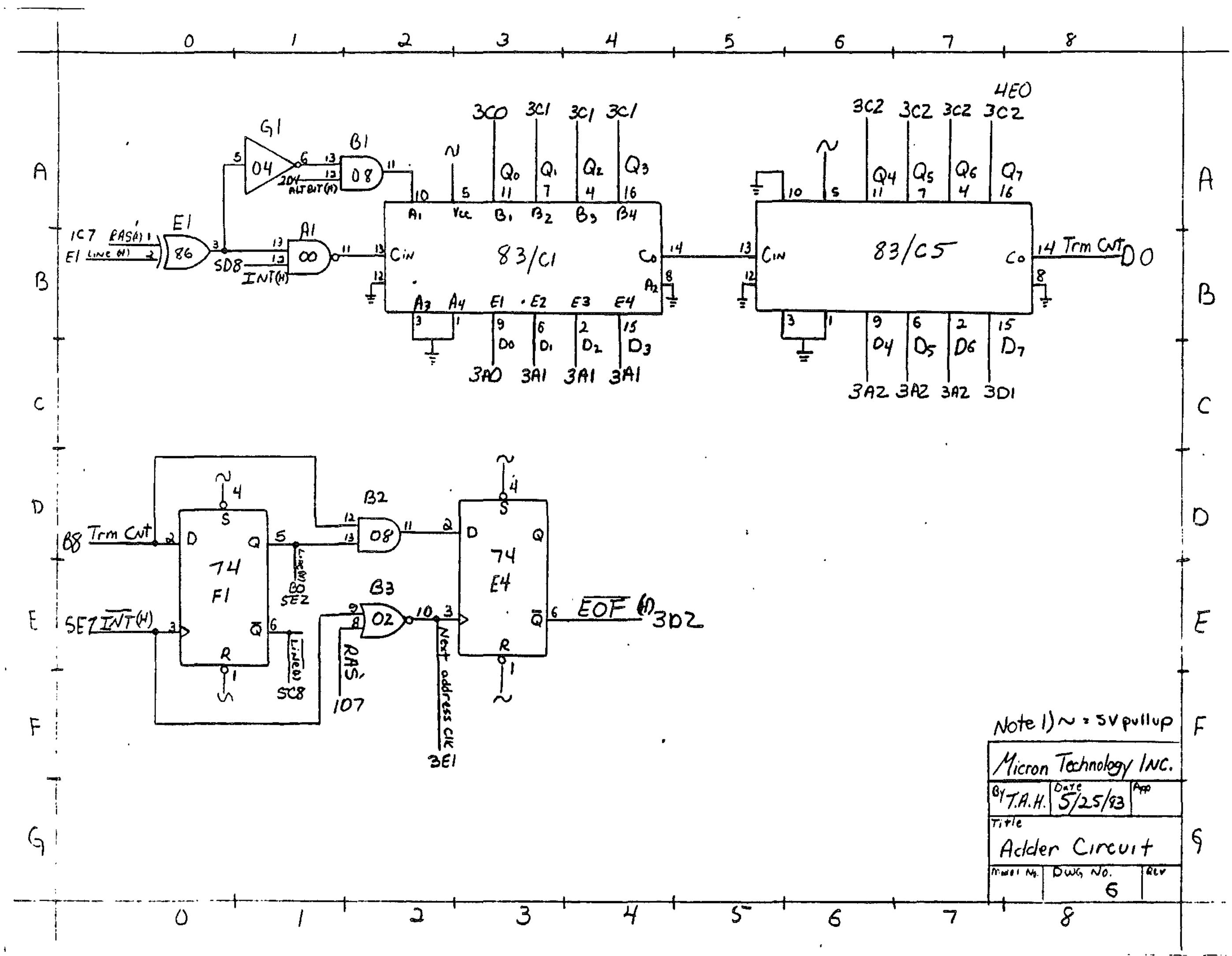
ADDRESS DESCRAMBLE, SOAK, DIN/DOUT CIRCUIT



TRANSMITTER AND INTERUPT GENERATOR CIRCUIT



ADDER AND END-OF-FRAME CIRCUIT



APPENDIX H

MICRONEYE APPLICATIONS SUBROUTINE LIBRARY DEVELOPMENT ROUTINES

MicronEye Applications Subroutine Library for Apple II Computers

ASSEMBLY LANGUAGE ROUTINES

GENERAL INFORMATION - Memory areas used are: $4000-$5FFF for hires graphics memory; $6000-$7FFF for the bitmap image; $8000-$8FFF for run-length-encode and temporary workspace for 512 x 128 images; $9000-$9670 for assembly language routines; $300-$305 for variables common to BASIC and assembly language; and $307-$30C for variables local to the assembly language routines. Load the routines with the command **BLOAD MEYEAPP,A$9000**. Specify **HIMEM: 16384** as the first instruction in a BASIC program to protect memory area above $4000. If the user has additional subroutines being loaded below $4000 and wants to insure the area is not overwritten by BASIC, HIMEM: should be set accordingly. Because these routines run into DOS buffer space, it is a good practice to use **PRINT CHR$(4);"MAXFILES 2"** at the start of the program. These routines (or possibly the MicronEye) do not always work properly when 80-column mode is active on the Apple IIe.

INIT - (CALL 36864). Reset ACIA; Set defaults for initial MicronEye setup. Everything but the slot number is initialized to some value by **INIT**. The slot number must be POKE'd prior to calling **INIT** because **INIT** calculates the slot address from the current slot number.

GETPIC - (CALL 36867). Gets a picture of the current exposure time and picture type and puts it into the bitmap area.

GETPIC7 - (CALL 36870). Puts picture directly to graphics screen using current exposure time and picture type. Picture types have the same dimensions as in **GETPIC** but the picture quality for types 1 and 2 will be somewhat worse because no enhancements are performed.

PUTSCREEN - (CALL 36873). Takes the image currently in the bitmap area and maps it onto the page 2 of hires graphics. The image will not display unless bitmap mode has been invoked. The routine masks off the MSB of each byte to avoid going from 8-bit to 7-bit graphics. When using the **GETPIC7** routine the **PUTSCREEN** routine does not need to be used.

BITMODE - (CALL 36876). Puts the computer into hires graphics mode without clearing the graphics screen. Use HGR2 from BASIC to enter hires graphics and clear the graphics screen. In this mode the normal text screen will not be visible.

TEXTMODE - Use the BASIC **TEXT** command to return from **BITMODE**.

RLE - (CALL 36879). Run-length encodes the image currently in the bitmap area and puts the encode into the encode area. Each row is terminated with an $FF and the image is terminated with two bytes of $FE. Picture types 0 and 2 use byte-length encoding while picture type 1 uses word-length encoding. Picture type 1 terminates each row with an $FFFF. Briefly stated, run-length encoding is a series of pairs of column locations in a row where the image changes from the background to foreground color and back. A row encoded with the bytes $44, $66, $80, $83, $FF means that columns $0-$43 of the row are the background color, $44-$65 are the foreground color, $66-$7F are the background color, $80-$82 are the foreground color, and $83 to the end of the row are the background color.

USER VARIABLES FOR SUBROUTINE CONTROL

SLOTNO (768) - Slot number containing the MicronEye. Must be a value between 1 and 7. This value must be set prior to calling the **INIT** routine. Failure to set to the proper slot number will cause the Apple to crash or hang.

PICTYPE (769) - Current picture type. 0 is a 128 x 64 bit image. 1 is a 512 x 128 bit image. 2 is a 256 x 64 bit image. **INIT** defaults the picture type to 0.

EXPTIME (770-771) - Exposure time in milliseconds. Exposure time divided by 256 goes into 771 and the remainder goes into 770. **INIT** defaults the exposure time to 256 milliseconds.

RLEBACK (773) - Run-length encode background color. 0-black, $FF-white. The **INIT** routine initializes the background color to white ($FF).

This variable is also used to determine the color that row 0 and columns 0 and 1 of every row are set to when **ENHANCE** is performed. This is necessary because some of the edge pixels lack enough neighbors to enhance. Therefore, the entire row or column is set to the background color.

ROWTAB (774-775) Location containing the address of a table that contains the starting addresses of the first 128 rows of the hires screen (page 2). From BASIC, to access the **N**th byte of row **M** on the hires screen, the following statements would be used:

```
10 RTAB = PEEK(774) + (256*(PEEK(775))
20 ROWM = RTAB + (2*M)
30 RADR = PEEK(ROWM) + (256*(PEEK(ROWM+1))
40 BYTE = PEEK(RADR+N)
```

In an assembly language environment, **ROWTAB** contains the address of the **ROWPTR** table. This is helpful when the user has his own set of assembler routines to complement the MicronEye Applications subroutine library and would like to access the **ROWPTR** table. To access the **N**th byte (7 pixels per byte) in the **M**th row, the user could do the following:

```
LDA   ROWTAB       ;Move value to ROWTAB to a zero-page location
STA   $06
LDA   ROWTAB+1
STA   $07
LDA   M            ;Multiply the desired row by two to obtain the
ASL                ; offset into the ROWPTR table of row M.
TAY
LDA   ($06),Y      ;Move the starting address of row M into
STA   $08          ; zero-page locations $08-$09.
INY
LDA   ($06),Y
STA   $09
LDY   N            ;Get the desired pixel into the accumulator.
LDA   ($08),Y
```

MicronEye Applications Subroutine Library for the IBM PC Computer

GENERAL INFORMATION - The MicronEye applications subroutines for the IBM PC support parameter passing and are designed to be compatible with Microsoft and Lattice 'C' compilers. To use these subroutines with other compilers or interpreters will require an understanding of how parameters are passed in the language being used.

INITPORT - *acia_status = **initport**(status_port);*
This routine initializes the communications link with the MicronEye. The *status_port* will always be a hexidecimal 318 unless the user has rejumpered the MicronEye interface board to another port address. The value returned by this routine is the status of the ACIA chip (serial communications chip) after initialization. Refer the the MicronEye owner's manual for a definition of ACIA status values.

GETPIC - *numbytes = **getpic**(bitmap,pictype,expose_time);*
This routine gets an image from the MicronEye of the specified type and exposure time. *Bitmap* is a pointer the byte array to be receiving the image. *Pictype* designates the picture type. Type 0 is a 128 x 64 image and type 1 is a 256 x 128 image. The 256 x 128 image is unenhanced and the **ENHANCE** or **MEDRES** routine is normally called subsequent to the **GETPIC** call to enhance the image. *Expose_time* is the time in milliseconds to expose the picture. *Numbytes* is returned from the call and is the number of bytes received during the transmission.

MEDRES - ***medres**(source,dest);*
This routine enhances a 256 x 128 image by simply eliminating the misplaced pixels. The resultant image is a 256 x 64 array. Both *source* and *dest* are char pointers and may optionally point to the same location.

ENHANCE - ***enhance**(source,dest);*
This routine enhances a 256 x 128 image by relocating the misplaced cells to their proper location. The resultant image is a 512 x 128 image. Fillin of the 'holes' is not performed automatically. Use the **FILLIN** routine after the **ENHANCE** routine to get a filled-in 512 x 128 image. *Source* and *dest* should not overlap.

FILLIN - *fillin(source);*
This routine fills in the image created by the **ENHANCE** routine. This routine should be used prior to using the **RLEROW** routine for the run-length encode routine to operate properly. The **SHOWPIC** routine will accept either a filled or unfilled image for display.

SHOWPIC - ***showpic****(startrow,startcol,rowct,colct,bitmap);*
This routine displays an image on the graphics screen. *Bitmap* is a char pointer to the start of the image. *Startrow* is an even integer between 0 and 199 specifying the row in the graphics screen that will contain the first row of the image. *Startcol* is an integer between 0 and 639 that specifies the column in the graphics screen that will contain the first column of the image. (The value is reduced to the nearest column divisible by 8.) *Rowct* is the number of rows in the image. *Colct* is the number of columns in the image.

RLEROW - *new_rleptr =* ***rlerow****(rowlen,eyeptr,rleptr);*
This routine run-length encodes a row of the image using word-length encoding. *Rowlen* is an integer and is the number of bytes to be encoded. This is normally the number of pixels per row divided by 8. *Eyeptr* is a pointer to the byte in the image where the encoding is to commence. *Rleptr* is a pointer to the next available word in the encode array. This is where the routine will place the encoded row. *New_rleptr* is a pointer returned by the routine which points to the next available word in the encode table after execution of **RLEROW**. Briefly stated, run-length encoding is a series of pairs of column locations in a row where the image changes from the background to foreground color and back. A row encoded with the bytes $44, $66, $80, $83, $FF means that columns $0-$43 of the row are the background color, $44-$65 are the foreground color, $66-$7F are the background color, $80-$82 are the foreground color, and $83 to the end of the row are the background color.

RLEROB - *new_rleptr =* ***rlerob****(rowlen,eyeptr,rleptr);*
This routine is identical to the **RLEROW** routine with the exception that byte-lenth encoding is used. This implies that a row larger than 255 pixels will not properly encode. This also requires that *rleptr* and *new_rleptr* be defined as char pointers rather than pointers to integers.

MicronEye Applications Subroutine Library for the Commodore 64 Computer

ASSEMBLY LANGUAGE ROUTINES

GENERAL INFORMATION - Memory usage is as follows: Hires screen is located at $E000 underneath the kernal ROM; screen color memory is located from $C000-$C3FF; assembly language routines reside from $4900-$4FFF and are loaded with **LOAD "MEYE.BIN",8,1**; 8K is required for the bitmap image and is defaulted by **INIT** to reside at $6000; 4K is required for temporary storage when working with picture types other than 0 and is defaulted by **INIT** to reside at $5000; For run-length encoding memory space must be designated and is defaulted by **INIT** to use the same area as the temporary storage area; Memory locations 679-687 is set aside for variables common to BASIC and assembly language; Memory locations 688-703 are reserved for variables local to the assembly language routines.

INIT - (SYS 18688). Reset ACIA; Set defaults for initial MicronEye setup as follows: display color for light pixels is white; display color for dark pixels is black; picture type is 0 (128 x 64); exposure time is 256 milliseconds; run-length encode area begins at $5000 and extends for 4K; temporary storage for enhanced image manipulation begins at $5000; the bitmap image begins at $6000; the background color for run-length encoding is white; and MEMTOP (highest memory available to BASIC is set to $5000.

GETPIC - (SYS 18691). Get a picture of the current exposure time and picture type and put it into the bitmap area.

PUTSCREEN - (SYS 18694). Takes the image currently in the bitmap area and maps it onto the graphics screen. The image will not display unless bitmap mode has been invoked.

BITMODE - (SYS 18700). Puts the computer into hires graphics mode without clearing the graphics screen. Use **GRESET** to clear the graphics screen. In this mode the normal text screen will not be visible.

TEXTMODE - (SYS 18703). Returns the computer to text mode. This routine should always be called prior to exiting the program or whenever text needs to be displayed. Text and graphics cannot be displayed at the same time.

RLE - (SYS 18697). Run-length encodes the image currently in the bitmap area and puts the encode into the encode area. Each row is terminated with an $FF and the image is terminated with two bytes of $FE. Picture types 0 and 2 use byte-length encoding while picture type 1 uses word-length encoding. Picture type 1 terminates each row with an $FFFF. Briefly stated, run-length encoding is a series of pairs of column locations in a row where the image changes from the background to foreground color and back. A row encoded with the bytes $44, $66, $80, $83, $FF means that columns $0-$43 of the row are the background color, $44-$65 are the foreground color, $66-$7F are the background color, $80-$82 are the foreground color, and $83 to the end of the row are the background color.

GRESET - (SYS 18706). Clears the graphics screen. Useful when changing from a large-sized picture to a smaller one.

CHANGEHUE - (SYS 18709). Changes the display color used for dark and light pixels based on the value of the **BWCOL** variable.

EXIT - (SYS 18712). Resets MEMTOP to the original value prior to the call to **INIT**.

USER VARIABLES FOR SUBROUTINE CONTROL

PICTYPE (679) - Current picture type. 0 is a 128 x 64 bit image. 1 is a 512 x 128 bit image. 2 is a 256 x 64 bit image. **INIT** defaults the picture type to 0.

EXPTIME (680-681) - Exposure time in milliseconds. Exposure time divided by 256 goes into 681 and the remainder goes into 680. **INIT** defaults the exposure time to 256 milliseconds.

BITMAP (682) - Memory page at which to locate the bitmap image. **INIT** defaults this to $60.

RLEMAP (683) - Memory page at which to locate the run-length encode. **INIT** defaults this to $50.

TMPMAP (684) - Memory page at which to locate the temporary storage area. **INIT** defaults this to $50.

RLESIZE (685) - Number of pages to reserve for the run-length encode. **INIT** defaults this to 4K (16 pages).

BWCOL (686) - Display colors to be used for dark and light pixels. The light pixel color belongs in the upper 4 bits of the byte and the dark pixel color belongs in the lower 4 bits of the byte. The numeric representation for each color is as follows: 0-black, 1-white, 2-red, 3-cyan, 4-purple, 5-green, 6-blue, 7-yellow, 8-orange, 9-brown, 10-light red, 11-gray 1, 12-gray 2, 13-light green, 14-light blue, 15-gray 3. The **INIT** routine uses white (1) for light pixels and black (0) for dark pixels.

BKCOL (687) - Run-length encode background color. 0-black, $FF-white. The **INIT** routine initializes the background color to white ($FF).

WARRANTY

If a MicronEye fails to perform properly due to a defect in workmanship or material within ninety (90) days from date of purchase, Micron will repair or replace it free of charge. Should this product require service during this warranty period, return the product to Micron at the following address, transportation charges prepaid:

MICRON TECHNOLOGY, INC.
2805 E. Columbia Road
Boise, Idaho 83706

Attach to the MicronEye your name, address, telephone number, a description of the problem and proof of date of retail purchase. This warranty does not apply to defects caused by unreasonable use.

A $35.00 restocking fee will be charged for units returned for any reason other than for replacement of defective parts. No unit may be returned for a refund after thirty (30) days from purchase.

THE FOREGOING IS IN LIEU OF ALL OTHER WARRANTIES EXPRESSED OR IMPLIED. MICRON TECHNOLOGY, INC. NEITHER ASSUMES NOR AUTHORIZES ANY PERSON TO ASSUME FOR IT ANY OTHER OBLIGATION OR LIABILITY IN CONNECTION WITH THE SALE OF THIS PRODUCT. IN NO EVENT SHALL MICRON TECHNOLOGY, INC. OR ITS DEALERS BE LIABLE FOR SPECIAL OR CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF THIS PRODUCT.

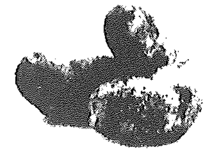
WARRANTY REGISTRATION

To receive full warranty protection, YOU MUST RETURN THIS REGISTRATION WITHIN 10 DAYS AFTER DATE OF PURCHASE to Micron Technology, Inc., Attn: Sales and Marketing Department, 2805 East Columbia Road, Boise, Idaho 83706.

Purchaser's Name ______________________________
Address ______________________________
City ____________ State ____________ Zip ________
Telephone Number ______________________________
Place of Purchase ______________________ Date ________
City ____________ State ____________ Zip ________
Serial Number ______________________________

MICRON
TECHNOLOGY INCORPORATED

2805 East Columbia Road
Boise, Idaho 83706
(208) 383-4000
TWX 910-970-5973



MICRON TECHNOLOGY manufactures high quality semiconductors in beautiful Southwestern Idaho. Micron's products reflect its continuing emphasis on quality products competitively priced. On-going research and development projects are geared to the continued supply of unique innovative products that are easy to use and complimentary in a wide variety of applications.

We're building our reputation on innovation.