FOX11 68HC11 Trainer Board

Getting Started Manual

Version 1.58 for Rev. F board

Table of Contents

The FOX11 trainer board, a low cost and high performance development board, provides real time emulation for the Motorola 68HC11microcontroller A and E families. It offers all useful features of the Motorola EVB board with the BUFFALO monitor and adds numerous enhancements at an extremely low cost. It combines a complete 68HC11 development system, advanced trainer, reliable 68HC 711E9, E20 programmer and a versatile SBC into a single package. For engineers, it serves as a WICE in-circuit emulator development system, Motorola EVB replacement, convenient prototype platform, and a low cost single board computer. For students, it acts as a user-friendly microcontroller trainer. It is as powerful as a high priced real-time in-circuit emulator, but it is as affordable as a low cost single board computer.

The FOX11 includes easy-to-use and user-friendly IDE software which runs under Windows® 95, 98, 2000, and XP. It offers fast file transfers, single-stepping, breakpoints, data watch for memory and registers, symbolic debugging compatibility with most assemblers and compilers, and user program termination with the <Esc> key.

Our exclusive ***phantom monitor***™ technology preserves all interrupt vectors including RESET. The monitor also preserves all on-chip RAM ($00-$1FF), EEPROM, and 30K external emulation RAM ($8800-$FFFF) available for user applications - there is no pre-empted chip memory.

Because many students are taught with the Motorola BUFFALO, we installed the BUFFALO monitor in U5 (27C256 EPROM). The board can be booted from the BUFFALO monitor, so students can use the board immediately before learning how to use the board under Wytec's phantom monitor.

The hardware includes:

- large solderless breadboard
- logic probe
- LCD module connector
- 4x4 keypad connector
- SPI port
- Speaker
- Potentiometer
- 8 LED status indicators for port B
- an 8 position DIP switch attached to port C
- 3 pushbutton switches
- dual RS232 ports
- a 26 pin male header for ports A, D, and E
- a 40-pin female connector for the address /data bus
- 60-pin EVB/EVBU-compatible female connector for all I/O ports

The package also includes a 9V 500mA wall plug-in power supply and a 6-foot DB9 cable.

The specification of the AC adapter is:

DC input:	110V
DC output:	9V
Current rating:	500mA
Type of plug:	2.1mm female barrier plug, center positive

The AC adapter is only available to the countries that use 110V.

WARNING: If more power is needed in a robot or other applications, the user should upgrade the AC adapter. Otherwise, the board could keep resetting itself when the VCC drops below 4.6V.
If your board sometimes resets by itself you need to upgrade your AC adapter to 9V 800mA or 9V 1A. Do not apply a DC voltage higher than 9V to this board.

People often use different terminology. In our product menus, "Download" means to transfer a file from the PC to the development board, while "Upload" means to transfer a file from the development board to the PC.

Through out the manual, **left click** means that you click the left button of the mouse and **right click** means that you click the right button of the mouse.

Install Ep2IDE software from CD:

If you have already installed the Ep2IDE for our WICE emulator or one of our EVBplus boards on your hard drive, you must rename the current folder from c:\Ep2IDE to c:\Ep2IDE_old before installing new software. The installation is automated by running "**SETUP.BAT**" on the CD. It will **OVERWRITE** the current folder if you don't rename it.

After the software is successfully installed, you can make a shortcut to AsmIDE.exe.

It's very important to make a shortcut so that its target location is C:\Ep2IDE, not c:\Windows\desktop or other locations. First, right click the Start button. Then, left click "Explorer". Left click on C:\Ep2IDE. Right click on AsmIDE.exe (an application program). Left click "Send to" and finally left click "Desktop" (do not click "COPY"). It will create an icon named "shortcut to AsmIDE" on the desktop. You can double check the target location by right clicking on the icon. Then, left click on "properties". You should see that the target location is C:\Ep2IDE. If you want to make a shortcut for AsmIDE on the Desktop, this is the correct way to do so. If you don't follow this method, your may have a problem running your program. Never drag the AsmIDE.exe to the desktop folder.

The default setting of AsmIDE for the FOX11 board is created in a text file named c:\Ep2IDE\AsmIDE.ini. In the future if you get lost with all the changes, you always can copy this file into the folder c:\Ep2IDE.

GETTING STARTED with BUFFALO Monitor

The Memory Map with BUFFALO monitor:

\$0000-\$00FF	On-chip 256 byte RAM for the 68HC11A1
\$0000-\$01FF	On-chip 512 byte RAM for the 68HC11E1
\$002D-\$00FF	On-chip RAM used by BUFFALO monitor
\$1000-\$103F	On-chip 64 control registers
\$1401	U1, 74HC273, Port F, output port for LCD, bit 3 is used to reset the board 0= normal, 1=reset
\$1403	U9, 74HC245, Port C, input port for DIP switch
\$1404	U10, 74HC273, Port B, output port for LEDs
\$2400-\$27FF	U6, 68B50
\$2400	UCTRL, USTAT
\$2401	UART
\$2800-\$2FFF	External device, /CS on pin40 of P4
\$6000-\$7FFF	If U4 is an 8K EEPROM (28C64), it's also duplicated at \$4000-\$5FFF
\$3000-\$7FFF	If U4 is a 32K EEPROM (28C256)
\$8000-\$DFFF	U3, RAM for user code or data
\$E000-\$FFFF	U5, BUFFALO monitor firmware

The important Jumper Settings:

J4	set for NCW by a cut-off trace (left side, No CodeWarrior)
J15	set for BUFL (middle position)
J17	set for NCW by a cut-off trace (left side, No CodeWarrior)
J20	set for DBUG (left side)
J21	jumper is installed by a cut-off trace for enabling single-step

Any attempt to write to the locations ($2200-$25FF) will have unpredictable results and must be handed with care in user programs.

Because most textbooks are written for the BUFFALO monitor, the FOX11board comes with both the BUFFALO monitor and the Wytec Phantom Monitor installed in U5. If you have worked with the BUFFALO monitor in the past, you can use the board right away. In fact, if you use the BUFFALO monitor with this board, the board becomes an ordinary 68HC11 EVB board just like many other EVBs on the market today, except it offers more on-board peripherals.

Before testing the board with the BUFFALO monitor, place a jumper in the middle position (labeled with 'BUFL' which stands for BUFFALO Monitor) on J15 and another jumper on J21 (labeled with 'TRACE'). To test the board, follow steps 1 through 6 below:

Step 1.

Plug the AC adapter into a wall outlet and plug the DC plug at the other end into the DC jack on the right side of the FOX11 board. During power up, the PB7 LED should blink twice and all other LEDs must be off. If this does not occur, turn over to the Questions & Answers section of the user manual.

Step 2.

Plug the DB9 male end of the cable into the DB9 connector P2 on the **upper right** corner of the FOX11 board. Plug the DB9 female end of the cable into the COM1 or COM2 port of your PC. The DB9 connector P3 on the lower right side of the board is the 68HC11 SCI port that can be used by a user's application program.

Step 3.

Press the reset button on the FOX11 board momentarily and the PB7 LED should blink twice. If this does not occur, turn over to the Questions & Answers section of the user manual.

Step 4.

To invoke the AsmIDE, right click the Start button. Then, left click "Explorer". Left click on C:\Ep2IDE and finally, double left click on AsmIDE.exe. If you have created a shortcut icon on the desktop, just double click the AsmIDE icon on the desktop.

The screen is divided into two windows. The top window is for editing your source code and the bottom window is shared by the **message window** and the **terminal window**.

Step 5:

You only need to use three commands from the AsmIDE for your 68HC11 development work. Use the File command to edit your source code, the Build->Assemble command to assemble your source code, and the Build->Download command to communicate with the FOX11 board.

In the View->Option->Terminal Window Options menu, set the COM port as 1 or 2 to match your PC COM port. Also, set the COM port options at 9600, N,8,1, and check the "enable the terminal window" box which will disable Wytec hc11 tools.

In the View->Option->Assembler menu, make sure that the chip family is 68HC11 and you can also see that the default assembler name for the hc11 is **myasm11.bat**, not as11.exe. This is the only change we have to make from the original AsmIDE. If you would like to use your own assembler, you can replace the myasm11.bat with the name of your new assembler.

If the assembler detects an error, it will show the error's line number in the file along with an error message. Go to the line to make a correction.

If the terminal options are set correctly, the jumper is installed on the BUFL position of J15, and the com port number is correct, you should see the following sign-on message every time the reset button is pressed. If you do not see this, the bottom window may be the message window. Click the terminal button in the bottom window to enable the terminal window display.

```
BUFFALO 3.47sc - Bit User Fast Friendly Aid to Logical Operation
```

Press the Enter key, you will get the BUFFALO monitor prompt
>

Step 6

All sample programs are debugged and tested for your convenience. They are located in the folder c:\Ep2IDE\Ex_BUFFL. Here are the steps to run your first sample program:

1. Click the File button to load test.asm from c:\Ep2IDE\Ex_BUFFL to view this program.

2. Click Build -> Assemble or click the assembler button on the toolbar to assemble your code and generate the test.s19 file. In order to display assembler messages, the bottom window is switched to the message window.

3. Click the terminal button of the bottom window to activate the terminal window and make sure that the BUFFALO monitor prompt ‘>’ is shown on the terminal window. If this is not the case, press the Enter key.

4. At the prompt ‘>’, type “**LOAD**” <Enter>.

5. Click Build->Download and select the file c:\Ep2IDE\Ex_BUFFL\test.s19 to download it.

6. After the download is done, type G D000 to run the program.

It will run the TEST program in real time. The program will test the switches, scan the keypad, send a message to the LCD display, adjust the LED brightness, generate music, and make LEDs PB0-PB7 act as a chaser. At first you can press the PC0 or PC1 pushbutton switches and see a change in the PB0 or PB1 LEDs. Then, pressing the PA0 switch will trigger the sound.

To stop the program, you have to press the reset button momentarily.

For more details on all of the sample programs, please read readme.txt in the c:\Ep2IDE\EX_BUFFL folder. All example programs are fully debugged, so the assembler won’t generate an error report. If you have an error in your program, you must correct it before an s19 file can be generated.

You can try to run a different example program later after you have finished reading this manual. You should always press the reset button before downloading a new program, because a new program may not work if an interrupt was enabled by a previous program.

Software development with the BUFFALO monitor:

We are using AsmIDE as a terminal program and the following instructions to create your source code. If you are using a different terminal program, the instructions may vary.

The steps to create your source code are as follows:

1. Click the **File** button to open an existing file or create a new file.

The memory locations from $00-$2C for the A family or $00-$2C and $100-$1FF for the E family are available as user DATA RAM. The BUFFALO monitor uses the RAM locations at $2D-$FF. The 24K memory locations from $8000 to $DFFF are available as user program CODE or DATA. In assembly language, you specify the starting address of your CODE by an ORG statement.

You can start the DATA RAM at address $00 with the statement ORG 0 followed by RAM variables, as shown by:

```
                ORG    0
TEMP:           RMB    1                ; reserve one byte of RAM for temp storage
```

```
XTEMP:      RMB   2           ; reserve two bytes of RAM for temp storage
```

If your program is small, say less than 4K, you can start your program at address $D000 with the statement ORG $D000 followed by your program, as shown by:

```
            ORG   $D000
            LDS   #$8FFF      ; initialize stack point
```

It will assemble your source program and generate hex code within 4K locations from $D000 to $DFFF. If your program is larger, you can change ORG $D000 to ORG $C000 or ORG $A000. You cannot use ORG $E000 or ORG $F000 because the BUFFALO monitor occupies the highest 8K locations ($E000-$FFFF).

Under the BUFFALO monitor, the FOX11 board cannot stop your program if it's hung in a loop. The only way to stop it is to reset the board. The problem with resetting the board is that you will not know where the 68HC11 was hung. It also leaves SWI instructions on all breakpoint addresses. You will have to re-download your s19 file all over again.

Here is a very simple program, but it's complete. It will flash the PB5 LED when it's running.

For a good programming practice, you should always place the LDS instruction in the first line of your code.

```
PB5:        EQU   $20               ; bit 5 of port B
PORTB:      EQU   $1404
FLS_RATE:   EQU   $8B00             ; change this number will change LED flash rate

            ORG   $D000
START:      LDS   #$8FFF
BACK:       CLR   PORTB             ; turn off the PB5 LED by resetting PB5=0
            JSR   DELAY
            LDAA  #PB5              ; turn on the PB5 LED by setting PB5=1
            STAA  PORTB
            JSR   DELAY
            JMP   BACK
DELAY:      LDY   #FLS_RATE
DLY:        DEY
            BNE   DLY
            RTS
            END
```

2. Save your file frequently while editing. If you are creating a new file and giving the file a name to save, enter the file name including the file extension, such as “test.asm”, not just “test”.

3. Click Build button-> Assemble, or click the assembler button on the toolbar to assemble your code and generate an s19 file. If the assembler detects an error, the error message will show the line numbers of your source code that caused the error. Beware that sometimes (not very often, but it does happen) the freeware assembler may indicate a wrong error line in the file and the actual line that caused the error may be one line off.

WARNING: The free assembler will generate an error on the BSET, BCLR, BRSET and BSCLR instructions if more than one comma is used in those instructions. For instance, BSET 0,X,$20 or BRSET 0,X,$20,$F000 is illegal, but BSET 0,X $20 or BESET 0,X $20 $F000 is OK. You have to use a space character to separate fields, not a comma.

4. Go to the line and correct the errors and go back to step 3 until there are no errors.

GETTING STARTED with Wytec Phantom Monitor and WBUG11

The FOX11 board is an ordinary EVB board with the BUFFALO monitor, but it's an easy-to-use and powerful In-circuit emulator type of development system when it's under the control of the Wytec Symbolic Debugger, WBUG11. If you don't use the WBUG11, you will not be able to exploit all of the features. Once you have used the WBUG11, you probably would not want to use the BUFFALO again.

The Memory Map with Wytec phantom monitor:

\$0000-\$00FF	On-chip 256 byte RAM for the 68HC11A1
\$0000-\$01FF	On-chip 512 byte RAM for the 68HC11E1
\$0800-\$0FFF	Wytec monitor firmware in U5
\$1800-\$1FFF	Wytec monitor firmware in U5
\$1000-\$103F	On-chip 64 control registers
\$1401	U1, 74HC273, Port F, output port for LCD, bit 3 is used to reset the board 0= normal, 1=reset
\$1403	U9, 74HC245, Port C, input port for DIP switch
\$1404	U10, 74HC273, Port B, output port for LEDs
\$2400-\$25FF	U6, 68B50
\$2400	UCTRL, USTAT
\$2401	UART
\$2800-\$2FFF	External device, /CS on pin40 of P4
\$6000-\$7FFF	If the U4 is an 8K EEPROM (28C64), it's duplicated at \$4000-\$5FFF
\$3000-\$7FFF	If the U4 is a 32K EEPROM (28C256)
\$8000-\$83FF	U3, RAM used by Wytec monitor
\$8400-\$FFFF	U3, RAM for user code or data

The important Jumper Settings:

J4	set for NCW by a cut-off trace (left side, No CodeWarrior)
J15	set for BUFL (middle position)
J17	set for NCW by a cut-off trace (left side, No CodeWarrior)
J20	set for DBUG (left side)
J21	jumper is installed by a cut-off trace for enabling single-step

Any attempt to write to the locations (\$2200-\$25FF, \$8000-\$83FF) will have unpredictable results and must be avoided in user programs.

Before testing the board with the Wytec monitor, place a jumper in the top position (labeled with 'WYTEC' which stands for Wytec Monitor) on J15. To test the board, follow steps 1 through 6 below:

Step 1.

Plug the AC adapter into a wall outlet and plug the DC plug at the other end into the DC jack on the lower right corner of the FOX11 board. During power up, the PB0 LED should blink 4 times and all other LEDs should be off. If this does not occur, turn over to the Questions & Answers section of the user manual.

Step 2.

Plug the DB9 male end of the cable into the DB9 connector P2 on the **upper right** corner of the FOX11 board. Plug the DB9 female end of the cable into the COM1 or COM2 port of your PC. The DB9 connector P3 on the middle of the right side of the board is the 68HC11 SCI port that can be used by a user's application program.

Step 3.

Press the reset button on the FOX11 board momentarily and the PB0 LED should now blink only twice. If it does not blink, turn over to the Questions & Answers section of the user manual.

Step 4.

To invoke AsmIDE, right click the Start button. Then, left click "Explorer". Left click on C:\Ep2IDE and double left click on AsmIDE.exe. If you have created a shortcut icon on the desktop, just double click the AsmIDE icon on the desktop.

Step 5:

The default settings of AsmIDE for the FOX11 board are created in a text file named c:\Ep2IDE\AsmIDE.ini. In the future if you get lost with all the changes, you can always copy this file into the folder c:\Ep2IDE.

In the View -> Options -> Wytec hc11 tools menu, you must check the 'Wytec Tools Enabled' box to use the Wytec Debugger software, WBUG11. You can use COM1 or COM2 port, but you do not have to change its baud rate, because the WBUG11 will set it at 38.4K for you. Please note that the default debugger startup batch name is **wbug11.exe**, not mydebug.bat. If you left click the assembler tab under the AsmIDE options menu, you also can see that the default assembler name for the hc11 is **myasm11.bat**, not as11.exe. These are two setup changes we have made from the original AsmIDE.

Step 6:

All sample programs are debugged and tested for your convenience and they are located in the folder c:\Ep2IDE\Ex_WYTEC. Here are the steps to run your first sample program:

1. Click the File button to load test.asm from c:\Ep2IDE\Ex_WYTEC to view the program.

2. Click Build -> Assemble or click the assembler button on the toolbar to assemble your code and generate the test.s19 file.

3. Click Build -> Wytec hc11 tools or click the debugger button on the toolbar that will bring up a small dialog window. You should see the file being downloaded is c:\Ep2IDE\Ex_WYTEC\test.s19 **(The current download s19 file name is always updated by the assembler).**

4. Now you **MUST** click the "Select this file" button, which will store the file name test.s19 into a text file named c:\Ep2IDE\Ep2IDE.txt. The debugger can automatically download the test.s19 by reading the Ep2IDE.txt when it is invoked. This is a very **important step** and you must do it when you want to debug a different program.

5. Click the Debugger button and the debugger will automatically download the test program TEST.s19. The "Programmer" button is used to program the on-chip EPROM of a 68HC711E9. You can ignore it for the time being.

WARNING: Check the Init register value shown on the top line of the screen before running your program. If you don't relocate your control register or on-chip RAM addresses, it should have a default value of $01. If the value is not $01, you should use Init command to change it back to $01, otherwise your program will crash.

6. At the prompt Ep6811>, enter "g \start" (the label is case sensitive) <Enter>.

It will run the TEST program in real time. The program will test the switches, scan the keypad, send a message to the LCD display, adjust the LED brightness, generate music, and make LEDs PB0-PB7 act as a chaser. At first, you can press the PC0 or PC1 pushbutton switches and see a change in the PB0 or PB1 LEDs. Then, pressing the PA0 switch will generate sound.

You can press the ESC key on the PC keyboard to stop the program. If you stop the program, the speaker may generate a clicking noise because the output comparator is still interrupting the 68HC11. To stop the clicking noise, you will need to press the reset button momentarily.

For more details on all sample programs, please read readme.txt in the folder c:\Ep2IDE\Ex_WYTEC.

All example programs are fully debugged, so the assembler won't generate an error. If you have an error in your program, you must correct it before an s19 file can be generated.

You can try to run a different example program later after you have finished reading this manual. You should always press the reset button before downloading a new program, because a new program may not work if an interrupt was enabled by a previous program.

Step 7:

To run another program, activate AsmIDE from the task bar at the bottom of the screen (this will minimize the debugger window). Click Build -> Wytec hc11 tools or click the debugger button on the toolbar that will bring up the Wytec hc11 tools dialog window. Use the browser to choose the s19 file you want to download, such as ex2.s19, then **click the "Select this file"** button and close the dialog window (Do not click the Debugger button, because the debugger is already invoked). Now activate **WBUG11 from the task bar** and press the F10 function key and R option. The ex2.s19 will be automatically downloaded. At the prompt Ep6811>, type "g \start" <Enter>.

Warning: Do not attempt to have more than one debug window open at any one time. If you do, you may lose communication with the board. Close the extra windows and then reset the board. Sometimes, if you lose communication, you may have to close AsmIDE, reset the board, and then reenter AsmIDE.

Software development with the Wytec's Phantom Monitor:

The steps to create your source code are as follows:

1. Click the **File** button to open an existing file or create a new file.

The memory locations from $00-$FF for the A family or $00-$1FF for the E family are available as user DATA RAM. The FOX11board does not use the RAM locations at $2D-$FF which are used by the BUFFALO monitor. The 30K memory locations from $8400 to $FFFF are available as user program CODE or DATA. In assembly language, you specify the starting address with an ORG statement.

You can start the DATA RAM at address $00 with the statement ORG 0 followed by RAM variables, as shown by:

```
            ORG   0
TEMP:       RMB   1           ; reserve one byte of RAM for temp storage
XTEMP:      RMB   2           ; reserve two bytes of RAM for temp storage
```

If your program is small, say less than 4K, you can start your program at address $F000 with the statement ORG $F000 followed by your program, as shown by:

```
            ORG   $F000
            LDS   #$FF        ; Initialize the stack point
```

It will assemble your source program and generate hex code within 4K locations from $F000 to $FFFF. If your program is larger, you can change the ORG $F000 to ORG $E000 or ORG $C000. You can use ORG $E000 or ORG $F000 because there is no BUFFALO monitor in the highest 8K locations ($E000-$FFFF)

After fully debugging your code, you do not have to relocate your code and re-assemble it. You can directly program the s19 file into a 68HC711E9. Your code is final for stand-alone operation after finishing your debugging session. It works just like a real time In-Circuit Emulator.

The FOX11 board can stop your program if it's hung in a loop. When you press the ESC key at the PC keyboard, it will interrupt the FOX11, but the BUFFALO monitor can't stop the program unless you reset the board. The problem with resetting the board is that you will not know where the 68HC11 was hung. It also leaves SWI instructions on all breakpoint addresses resulting in having to download your s19 file all over again.

Here is a very simple program, but it's complete. It will flash the PB5 LED when it's running. The program source code is similar to the tutor2.asm in the directory c:\Ep2IDE\Ex_WYTEC.

For a good programming practice, you should always place the LDS instruction in the first line of your code.

```
PB5:        EQU   $20           ; bit 5 of port B
PORTB:      EQU   $1404
FLS_RATE:   EQU   $8B00         ; change this number will change LED flash rate

            ORG   $F000
START:      LDS   #$FF          ; the top of A1 internal RAM
BACK:       CLR   PORTB         ; turn off the PB5 LED by resetting PB5=0
            JSR   DELAY
            LDAA  #PB5          ; turn on the PB5 LED by setting PB5=1
            STAA  PORTB
            JSR   DELAY
            JMP   BACK
DELAY:      LDY   #FLS_RATE
DLY:        DEY
            BNE   DLY
            RTS
            ORG   $FFFE
            FDB   START         ; reset vector
            END
```

2. Save your file frequently while editing. If you are creating a new file and giving the file a name to save, enter the file name including the file extension, such as "test.asm", not just "test".

NOTE: Since the debugger is written in DOS, **folder name and filename** should not be longer than 8 characters.

3. Click Build -> Assemble or click the assembler button on the toolbar to assemble your code. If your code has no errors, it will generate an s19 file, a listing file, and a symbol file. If the assembler detects an error, the error message will show the line numbers of your source code that caused error. Beware that sometimes (not very often, but it does happen) the freeware assembler may indicate a wrong error line in the file and the actual line that caused error may be one line off.

4. Correct the errors and go back to step 3, until there are no errors.

5. After your code is successfully assembled, you can click Build -> Wytec hc11 tools-> "select this file" button to update the file c:\Ep2IDE\Ep2IDE.txt for the debugger.

6. Close the dialog window and activate the **WBUG11 from the task bar**. Press the F10 function key and the R option. The debugger will automatically download 3 files, namely YOUR_FILENAME.S19, YOUR_FILENAME.SYM, and YOUR_FILENAME.PAR. The S19 file is the hex code. The SYM file is the symbol file, so you can use symbols in commands instead of hex numbers. The PAR file is the parameter file that includes the EEPROM programming enable/disable flag, INIT, TMSK2, OPTION, and BPROT register values. The PAR file also includes breakpoint addresses and memory display addresses.

The assembler makes a default PAR file. When you exit the debugger, the debugger will update it by saving the current settings. **The PAR file is automatically saved after you exit the debugging session.**

Also, the Write command of the debugger can create a PAR file. After invoking the Write command in the debugger, it will prompt you to enter a choice among U, S and P options. Choose the P option and give a full file name such as test.par. It will make a PAR file for you.

The most useful feature of the PAR file is its ability to remember the INIT register value. Suppose you relocate the 68HC11 control registers to a different memory block by changing the value of the INIT register in the beginning of your source program (the INIT register can only be changed in the first 64 E cycles in expanded mode), you have to use the Init command to change the INIT register to match that value before running your program under the WBUG11, otherwise your program will not run. When the PAR file is loaded, it will generate a system reset and force the 68HC11 to take the new INIT register value from the PAR file. For more information, see the Questions & Answers section of the user manual.

7. When the download is finished and you are prompted with Ep6811>. Next, the PC will reset the board. During the reset, there will be some communication between the PC and the board. If you enter the command too soon, it could disrupt the communication and you will get an error message. After this reset process, you can run your program by entering “go \start” where the start is the label of the starting address in your source code or enter G F000 if you know the starting address is $F000, but do not enter command too soon.

In the command line, **ALL NUMBERS ARE HEXADECIMAL** and the $ sign is not needed. Also you should notice that the label \start has the back slash in the front. **All hex numbers may be substituted by symbols starting with the backslash ‘\’.** For more information read the Command format section in the user manual.

8. During the debugging session, if you want to modify your source code, you don’t have to exit the debugger. You can activate AsmIDE to edit your code and save your new code with the same file name. Then click Build -> Assemble or click the assemble button on the toolbar to assemble the code. A new S19 file should be generated. Because you did not change the file name, you don’t have to **click the “Select this file”** button again. The only thing you have to do is to click the “**WBUG11**” button at the task bar at the **bottom line of the screen** to activate debugger window. Then enter the F10 key and the R option to download your files and start debugging again.

Single Board Computer application:

The Memory Map in Single Board Computer applications:

$0000-$00FF	On-chip 256 byte RAM for the 68HC11A1
$0000-$01FF	On-chip 512 byte RAM for the 68HC11E1
$0800-$0FFF	Wytec monitor in U5, some subroutines are callable from user programs
$1800-$1FFF	Wytec monitor in U5, some subroutines are callable from user programs
$1000-$103F	On-chip 64 control registers
$1401	U1, 74HC273, Port F, output port for LCD, bit 3 is used to reset the board 0= normal, 1=reset
$1403	U9, 74HC245, Port C, input port for DIP switch
$1404	U10, 74HC273, Port B, output port for LEDs
$2400-$25FF	U6, 68B50
$2400	UCTRL, USTAT
$2401	UART
$2800-$2FFF	External device, /CS on pin40 of P4

$8000-$AFFF	U3, RAM for user data
$E000-$FFFF	User code in U4 (8K EEPROM)
$B000-$FFFF	User code in U4 (32K EEPROM)

The important Jumper settings for SBC:

J4	set for NCW by a cut-off trace (left side, No CodeWarrior)
J15	set for BUFL (middle position)
J17	set for NCW by a cut-off trace (left side, No CodeWarrior)

Program external EEPROM in U4 and 512 bytes of internal EEPROM:

You can program EEPROM in U4 and 512 bytes of internal EEPROM ($B600-B7FF) under control of the Wytec debugger WBUG11. Use the Load command as described on page 10 of the user manual. The instructions will be shown on the screen when Load command is entered.
If you want to use the board as an SBC, you can program your application s19 file into U4. After programming the chip, move the jumper to the bottom position of J15 (located at the left side of the speaker). It will completely bypass both the Wytec's proprietary phantom monitor and the BUFFALO monitor. The address range of U4 is changed from $3000-$7FFF to $B000-$FFFF and the address range of U3 (32K RAM) is changed from $8000-$FFFF to $8000-$AFFF. The program code will auto start from U4 after reset or power up.

You can only program U4 under the control of Wytec debugger WBUG11 and your application s19 file must be assembled in the address range of $B000-$FFFF. **The two highest addresses $FFFE and $FFFF in your program must be loaded with the starting address of the program**. Here are the steps to program U4 with the file test.s19:

Before programming, the jumper on J15 must be placed on the "WYTEC" position and the jumper on J20 must be placed on the "DBUG" position (not the "PRG" position).

1. Move the jumper to the right side on J3 to enable EEPROM write.
2. Enter Load command and select the option E to download an s19 file into U4.
3. Enter test.s19 as the file name to be downloaded.
4. When the programming cycle starts, the PB0-PB7 LEDs will act as a chaser.
5. The programming is done when the LEDs stop chasing.
6. Move the jumper to the left side on J3 to write-protect U4.
7. Move the jumper on J15 to the bottom position (labeled with the 'SBC').
8. Press the reset button. Your application program should run.

Before programming you have to make sure that the jumper on J3 is on the right side to enable EEPROM write, otherwise the programming process will not continue. After programming, place the jumper on the left side of J3 to write-protect the EEPROM, otherwise the EEPROM could lose data quite easily after running a bad program during a debugging session.

The procedures to program the 512-byte internal EEPROM ($B600-B7FF) are the same as the above steps except for step 2. Select option B instead of option E after the Load command is entered.

The J3 is used to write-protect the external EEPROM U4 and it has nothing to do with the 512 bytes of internal EEPROM. The 512 bytes of internal EEPROM are enabled or disabled by the bit 0 of the Config register ($103F). To disable internal EEPROM, you program the Config register with value of $0C. To enable it, program the Config register with value of $0D.

The Config register is an EEPROM cell and it can be programmed in test mode. Here are steps to enable or disable 512 bytes of internal EEPROM

1. Make sure the jumper on J15 is in the top position for Wytec monitor
2. Press the reset button momentarily while holding down the PA0 switch to enter test mode. The PB0 LED should flash twice. The mode indicator 'EXP' on the top line of the screen should be changed to 'TST'.
3. Press the F8 function key and choose option 5.
4. Enter value $0C for Config register to disable internal EEPROM or $0D to enable EEPROM then <Enter>.
5. Press ESC to exit.

About AsmIDE:

AsmIDE is written by Eric Engler and it's simple and easy to use. It offers all the basic features that you need to learn 68HC11 programming with this board. Sometimes a simple IDE may not be a bad idea; at least students don't need to spend too much time to learn how to use the IDE, so they will have more time to focus on learning the 68HC11which may be their main objective of taking a course. If you spend a lot of time to master a bloated IDE and you don't use it for a while, you will probably forget how to use it anyway. So, why spend time on something that you are going to lose? On the contrary, a simple, easy to use IDE saves you time and will be much easier to remember how to use even after an extended period of time.

The most valuable asset of the FOX11 board is the WICE debugger, **WBUG11**. Some boards that you can buy today may have a bloated IDE, but you probably would get a BUFFALO type monitor for debugging.

Using your own assembler or editor for WBUG11:

If you would prefer to use your own assembler or editor instead of AsmIDE, you can use AsmIDE for launching the WBUG11. The full path name of the file that you work on must agree with the file name shown on the Wytec hc11 tools dialog window, otherwise the WBUG11 will not be able to locate your s19 file to be downloaded.

Make your own monitor:

If you want to make your own monitor to install a high language source level debugger, you can treat your monitor as an application and program it into U4 (20K EEPROM) according to the instructions shown on the previous page.

Using the board as a 68HC711E9 programmer:

If you need to program a 68HC711E9, you can use this board as a 68HC711E9 programmer, but only under control of the Wytec debugger WBUG11. You must provide a regulated 12 V DC, 30mA and also make sure that the program is fully debugged. To activate the programmer, click Build -> Wytec hc11 tools, or click the debugger button on the toolbar that will bring up a small dialog window. If the current downloaded s19 file is correctly shown in the dialog window, click the "**Select this file**" button and then click the "**programmer"** button to program the 68HC711E9. The programming is done in bootstrap mode and the programming instructions will be displayed on the screen step by step. The memory addresses range for the 68HC711E9 is from $D000 to $FFFF. If your S19 file contains addresses outside of this range, an error message will be generated and the chip will not be programmed. If your assembler or C compiler generates an s19 file with some extra RAM addresses, you must use an editor to delete those addresses.

During the programming, the data will be automatically verified.

The 52-pin PLCC socket is not built for a high volume production programmer. It will not last much longer than 100 insertions. When the contacts of the PLCC socket are worn, you can use a fine dental tool to pry up the contacts a little bit to extend its life considerably.

ON-BOARD HARDWARE

In expanded mode, Port B and port C of the 68HC11 are used for address and data buses. They are not available to user programs as I/O ports, but they are re-produced by U10, 74HC273 (for port B) and U9, 74HC245 (for port C and it's used as a 74HC244 for one direction only). The DDRC of the 68HC11 is ignored.

The addresses for port B and C are partially decoded because there are not enough inputs for U11, GAL16V8. The logic equations for port B and C are listed in CDROM\document\schematics\U11_logic.txt.

A0 is used to separate the port B and port C. All even numbers of addresses from $1400 to $17FF are assigned to port B, while all odd numbers are assigned to port C.
$1400, $1402, $1404, $1406 ... and $17FE are all valid addresses for port B. $1404 is chosen for port B because port B's real address is $1004.
$1401, $1403, $1405, $1407 ... and $17FF are all valid addresses for port C. $1403 is chosen for port C because port C's real address is $1003.

The LCD port (port F) is output only and the port C is input only, so they can share the same address. The A0 is the only address input needed to select 3 ports. $1401 is chosen for port F to make it looks like a different port, but addresses $1401 and $1403 are the same address because the partial decoding.

The re-produced port B is an output port and each port B line is monitored by a LED. The re-produced port C is an input port only and it is connected to an 8-position DIPswitch. The DIPswitch is connected to GND via eight 4.7K resistors, so it's not dead short to GND. When port C is driven by external circuits, the DIP switch setting is ignored, but it's better to leave all 8 DIP switches at upper positions.

In the 68HC11, the port B is also an output port only, but the port B latch is readable by the MCU. You cannot read the port B on the fox11 board, therefore the bset and bclr instructions won't work with port B on the FOX11. The way to get around is to use a RAM byte as an image of the port B, you can manipulate the RAM byte in any way you want, then output the RAM byte to port B.

So one instruction "bset portb,1' should be replaced by three instructions:

```
portb_image     rmb     1

                bset portb_image,1
                ldaa portb_image
                staa portb
```

The PA0 switch is used as a general purpose input switch, except during power up. During power up, pressing the RESET button momentarily while holding the PA0 switch will force the 68HC11 to enter test mode. In test mode, the configure register can be modified. PA4 and PA6 headers are the outputs of Output Comparators 4 and 2. They can be used to drive robot servos (Make sure that robot servos have their own power supply).

Port E is an 8-bit ADC or a general input port. The trimmer VR1 is connected to the PE7 input of the ADC port via the cut-off trace J5. The trace can be cut if PE7 must be used by a target circuit for a different purpose. PD2-PD5 and PE0-PE3 are used for keypad interface. If a keypad is not connected to the J6, PD2-PD5 and PE0-PE3 then can be use by user programs. If a keypad is used, PE4-PE7 are still available for A/D channels.

An on-board logic probe LED is connected to pin 33 of the female socket connector P4 and can be used to monitor high or low status at any point of the circuit as a logic probe. It can be connected to VCC for the power indicator if you add a jumper between pin 33 and pin 34.

The following are all I/O subroutines in Wytec's monitor that are callable from user's application programs:

```
                        ORG      $0800      ; FOX11 board I/O routines in Wytec monitor

RESERVE1:               RMB      3          ; reserved for future use
RESERVE2:               RMB      3          ; reserved for future use
GET_DATE:               RMB      3          ; gets current date from PTC
GET_TIME:               RMB      3          ; gets current time from PC
OUTSTRG00:              RMB      3          ; outputs a string terminated by 0
LCD_INI:                RMB      3          ; initializes the 16x2 LCD module
LCD_LINE1:              RMB      3          ; displays 16 char on the first line
LCD_LINE2:              RMB      3          ; displays 16 char on the second line
SEL_INST:               RMB      3          ; selects instruction before writing the LCD module
SEL_DATA:               RMB      3          ; selects data before writing the LCD module
WRT_PULSE:              RMB      3          ; generates a write pulse to the LCD module
```

The circuit is designed in such way that the value of all resistors and capacitors are not critical, they can be off -50% or +100%.

How to use the LCD port:

The LCD port is an 8-bit output port. Its primary usage is as an LCD display module. If the port is not used as an LCD display, it can be used as general-purpose output that can be accessed via the header J1. Note that LCD3 does not go to J1 but is used as software reset of the board.

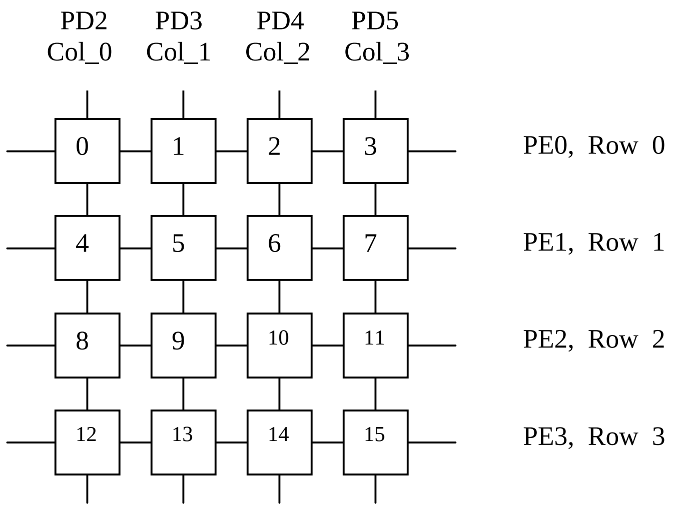
The pinouts of J1 are as follows:

Pin 1	GND	
Pin 2	VCC (5V)	
Pin 3	Via a 100 Ohm resistor to GND	
Pin 4	PF0	RS pin for LCD module
Pin 5	GND	
Pin 6	PF1	EN pin for LCD module
Pin 7	Not used	
Pin 8	Not used	
Pin 9	PF2	
Pin 10	PF3	
Pin 11	PF4	DB4 pin for LCD module
Pin 12	PF5	DB5 pin for LCD module
Pin 13	PF6	DB6 pin for LCD module
Pin 14	PF7	DB7 pin for LCD module
Pin 15	Via an 18 Ohm resistor to VCC	LED backlight for LCD module
Pin 16	GND	

Keypad interface via port D and port E:

The following signal definitions only apply to Wytec 4X4 membrane keypad.

Wytec 4X4 membrane keypad connections:



The pinouts of J6 for Wytec 4X4 membrane keypad are as follows:

Pin 1	GND	
Pin 2	PD2	connects COL0 of the keypad
Pin 3	PD3	connects COL1 of the keypad
Pin 4	PD4	connects COL2 of the keypad
Pin 5	PD5	connects COL3 of the keypad
Pin 6	PE0	connects ROW0 of the keypad
Pin 7	PE1	connects ROW1 of the keypad
Pin 8	PE2	connects ROW2 of the keypad
Pin 9	PE3	connects ROW3 of the keypad
Pin10	VCC	

An 8-pin male header is installed on pin 2 through pin 8. Pins 1 and 10 are not used.

PD2 connects COL0 of the keypad via pin 1 of the 8-pin keypad header J6
PD3 connects COL1 of the keypad via pin 2 of the 8-pin keypad header J6
PD4 connects COL2 of the keypad via pin 3 of the 8-pin keypad header J6
PD5 connects COL3 of the keypad via pin 4 of the 8-pin keypad header J6

PE0 connects ROW0 of the keypad via pin 5 of the 8-pin keypad header J6
PE1 connects ROW1 of the keypad via pin 6 of the 8-pin keypad header J6
PE2 connects ROW2 of the keypad via pin 7 of the 8-pin keypad header J6
PE3 connects ROW3 of the keypad via pin 8 of the 8-pin keypad header J6

The PE0-PE3 has a 100K pull-up resistor in each line.

The keypad scan routine sets PD5 low. It sets PD2, 3, and 4 high. It then tests PE0-PE3.
If no key is down, PE0-PE3 remain high.
If PE3 = low, key 15 is down.
If PE2 = low, key 11 is down.
If PE1 = low, key 7 is down.
If PE0 = low, key 3 is down.

The keypad scan routine then sets PD4 low. It sets PD2, 3, and 5 high. It then tests PE0-PE3.
If no key is down, PE0-PE3 remain high.
If PE3 = low, key 14 is down.
If PE2 = low, key 10 is down.
If PE1 = low, key 6 is down.
If PE0 = low, key 2 is down.

The keypad scan routine then sets PD3 low. It sets PD2, 4, and 5 high. It then tests PE0-PE3.
If no key is down, PE0-PE3 remain high.
If PE3 = low, key 13 is down.
If PE2 = low, key 9 is down.
If PE1 = low, key 5 is down.
If PE0 = low, key 1 is down.

The keypad scan routine then sets PD2 low. It sets PD3, 4, and 5 high. It then tests PE0-PE3.
If no key is down, PE0-PE3 remain high.
If PE3 = low, key 12 is down.
If PE2 = low, key 8 is down.
If PE1 = low, key 4 is down.
If PE0 = low, key 0 is down.

The 26 pin MCU I/O header, J16, is used to control a dedicated target board. Pin 1 through pin 20 are the same pinouts on the 20 pin MCU I/O header of the original EVBplus board. Pin 21 through pin 26 are additional I/O lines for the FOX11 board. If the user designs his target system by using only the I/O lines that are included in the 26 pin header, this FOX11 board with Wytec debugger is actually a real time In-Circuit Emulator. The user's code is emulated at exact addresses.

The pinouts of the 26 pin I/O header are as follows:

Pin 1	GND	Pin 2	VCC (5V)
Pin 3	PD5	Pin 4	/IRQ
Pin 5	PD4	Pin 6	PA7
Pin 7	PD3	Pin 8	PA6
Pin 9	PD2	Pin 10	PA5
Pin 11	PD1	Pin 12	PA4
Pin 13	PD0	Pin 14	PA3
Pin 15	PE1	Pin 16	PA2
Pin 17	PE0	Pin 18	PA1
Pin 19	/RESET	Pin 20	PA0
Pin 21	PE2	Pin 22	PE5
Pin 23	PE3	Pin 24	PE6
Pin 25	PE4	Pin 26	PE7

SPI port pinouts are as follows:

Pin 1	VCC (5V)	Pin 2	VCC (5V)
Pin 3	PF2 (LOAD)	Pin 4	PD2 (SPI DATA IN)
Pin 5	PF3 (STROBE)	Pin 6	PD3 (SPI DATA OUT from 68HC11)
Pin 7	not used	Pin 8	PD4 (CLOCK)
Pin 9	GND	Pin 10	GND

All on-board jumpers:

J1	LCD Port for an LCD module, 4-bit data interface.
J2	The jumper is installed to enables LCD backlight. Jumper can be removed to save power.
J3	U4 EEPROM write protect jumper. Place a jumper on the left side (It's labeled with 'DIS') to disable EEPROM programming (write-protect). Move the jumper to the right side (It's labeled with 'ENABLE') to enable EEPROM programming. This jumper has no any effect on 512 bytes of internal EEPROM.
J4	Code Warrior's Hi-Wave C source level debugger selector. The jumper is installed by a cut-off trace on the left side (NCW) for BUFFALO monitor or Wytec monitor operations. When jumper is on the left side (It's labeled with 'NCW' which stands for No CodeWarrior), the CodeWarrior monitor inside of U5 is disabled, but the BUFFALO monitor is enabled. When jumper is on the right side (It's labeled with 'CW' which stands for Code Warrior), it enables the CodeWarrior and disables the BUFFALO monitor. In order for the CodeWarrior to work, J15 must be set for the BUFFALO monitor.
J5	Connects VR1 trimmer pot to PE7 of the ADC, it's a cut-off trace.
J6	4 X 4 keypad interface
J7	Mode selector. No jumpers are installed. Install the MODEA jumper for single chip mode.
J8	Analog voltage reference selector When jumpers are on the left side, the on-board 5V DC is the reference voltage. When jumpers are on the right side, the user target provides the ADC reference voltage via pin 51 and pin 52 of P1 (they are labeled with UserVRL and UserVRH). It's connected to the on-board 5V DC reference voltage and GND via two cut-off traces.
J9	SPI connector
J10	68HC11 Clock selector. The jumper is on the upper position (labeled with 'INT') by a cut-off trace. When jumper is on the upper position (labeled with 'INT'), clock is provided by the on-board crystal. When jumper is on the lower position, (labeled with 'EXT'), user target provides an HC compatible clock source. The jumper is installed on the upper position by a cut-off trace
J11	Clock output. It's not installed. If it's installed, the clock output of the 68HC11(pin 8) can be a clock source of a user target board.
J12	The 68HC11's SCI receiver source selector Jumper on left side = SCI PD0 receives signal from your target system via pin 20 of P1 Jumper on right side = SCI PD0 receives signal from RS232 input of P3 (DB9 connector).
J13	Connects the 68HC11 SCI's PD1 to the RS232 output of P3 (DB9 connector). It's connected by a cut-off trace.
J14	Enables speaker. The speaker is driven by PA5, Output Comparator 3, through jumper J16. It's connected by a cut-off trace.
J15	Monitor selector. Place a jumper on the top position for Wytec monitor, the middle position for BUFFALO monitor and the bottom position for auto-starting program in U4.
J16	68HC11 ports A, D and E
J17	Code Warrior's Hi-Wave C source level debugger selector. The jumper is installed by a cut-off trace on the left side (NCW) for BUFFALO monitor or Wytec monitor operations. When jumper is on the left side (It's labeled with 'NCW' which stands for No CodeWarrior), the /IRQ of U6, 68B50, is connected to the /XIRQ of the 68HC11 for Wytec monitor. When jumper is on the right side (It's labeled with 'CW' which stands for CodeWarrior), the /IRQ of U6, 68B50, is connected to the /IRQ of the 68HC11 for Code Warrior's monitor. In order for the CodeWarrior to work, J15 must be set for BUFFALO monitor.
J18	PA4, OC4 output for servo application
J19	PA6, OC2 output for servo application
J20	The DBUG/PRG jumper is used for selecting an operating mode. When it's on the right side, the board is used **ONLY** for programming the 68HC711E9 chip. When it's on the left side, the board is used for debugging your code.
J21	Jumper is installed by a cut-off trace for enabling single-step PA3 is connected to XIRQ via a 1K resistor to enable single-stepping operation.

P1	60 pin female connector for I/O ports, EVB/EVBU compatible.
P2	RS232 port for development work, connects to the PC com port
P3	PS232 port of the SCI for user application. In order to use this port, the jumper on the J12 must be set at the right side labeled with ‘232’.
P4	40 pin female connector for address and data ports, Motorola 68HC11 EVM compatible

MODEA and MODEB jumpers: They are not used in debugging sessions. They can be used for programming the 68HC711E9 OTP part in bootstrap mode or debugging in single chip mode.

The P3 (DB9 female connector) is configured as a **DCE** device and it can be directly connected to your PC ‘s COM port.

Application Circuit Corner (ACC) and Solderless Breadboard:

The footprints of four popular 68HC11 applications are laid out underneath the solderless breadboard. The **ACC** consists of a DS1302 Real Time Clock with a battery backup, a DS1620 Digital Thermometer and Thermostat, a 12V DPDT relay, a L293D Motor driver, a RF transceiver, a 3.5mm jack, and a RS485 interface circuit. In the future we will offer each circuit in a separate kit for your experiments. With these application circuits, this board can easily be transformed into a complete platform for many useful applications.

All I/O signals are available on the 60 pin female connector P1. Use the breadboard for prototyping or solder components directly to the board if you remove the breadboard. The FOX11 board is not complete without a breadboard.

The schematic for the PC board is divided into sch1.pdf and sch2.pdf. The ACC schematic is in App_cir.pdf.

BUFFALO I/O routines.

Many 68HC11 books have example programs that access BUFFALO I/O routines. The BUFFALO I/O routines are located at $FFA0-$FFCF. For LCD interface, the following 3 routines are included in BUFFALO monitor version 3.47.

```
                    ORG      $FF70
lcd_ini:            rmb       3       ; initializes 16x2 LCD module
lcd_line1:          rmb       3       ; displays 16 char on first line
lcd_line2:          rmb       3       ; displays 16 char on second line
```

When using them with the Wytec debugger WBUG11, the BUFFALO I/O functions are duplicated at $0FA0-$0FCF.

Following are the BUFFALO I/O routines’ function description and jumper table:

ORG $0FA0

UPCASE	convert the character in A to uppercase
WCHEK	test the character in A for white space and returns with the Z bit set if A is a white space (Space, comma, tab)
DCHEK	test the character in A for white space and returns with the Z bit set if A is a carriage return or white space (Space, comma, tab)
INIT	initialize SCI, is not needed with the FOX11 board
INPUT	reads PC keyboard input
OUTPUT	writes the character in the A to CRT display
OUTLHLF	converts 4 most Significant Bit of A to ASCII and Writes it to CRT display

OUTRHLF	converts 4 most Significant Bit of A to ASCII and Writes it to CRT display
OUTA	output ASCII character in A to CRT display
OUT1BYT	converts the binary byte that is pointed to by X register to 2 ASCII bytes and write them to CRT display
OUT1BSP	it's OUT1BYT followed by sending a space to CRT display
OUT2BSP	converts the binary word (2 bytes) that is pointed to by X register to 4 ASCII bytes and write them followed a space to CRT display
OUTCRLF	write carriage return, line feed to CRT display.
OUTSTRG	it's OUTCRLF followed by writing the ASCII string that is pointed to by X register to CRT display and until character is $04
OUTSTRG0	it's OUTSTRG without writing leading carriage return & line feed to CRT display
INCHAR	waits for an ASCII character from keyboard and put it in accumulator A

Jump Table

$0FA0	UPCASE
$0FA3	WCHEK
$0FA6	DCHEK
$0FA9	INIT
$0FAC	INPUT
$0FAF	OUTPUT
$0FB2	OUTLHLF
$0FB5	OUTRHLF
$0FB8	OUTA
$0FBB	OUT1BYT
$0FBE	OUT1BSP
$0FC1	OUT2BSP
$0FC4	OUTCRLF
$0FC7	OUTSTRG
$0FCA	OUTSTRG0
$0FCD	INCHAR

We have added several I/O routines in the beginning of our Phantom Monitor.
Following are the Phantom Monitor's I/O routines' function description and jumper table starting at $0800:

```
                    ORG     $0800
```

RESERVE1	reserved for the future
RESERVE2	reserved for the future
GET_DATE	X register points to a 10 byte RAM block before calling this subroutine, it returns the date information of host PC in the format of MM-DD-YYYY.
GET_TIME	X register points to an 11 byte RAM block before calling this subroutine, it returns the time information of host PC in the format of HH:MM:SS AM or HH:MM:SS PM.
OUTSTRG00	It's OUTSTRG0 except the ending character is $00, instead of $04.
LCD_INI	initialize a 16x2 LCD display module
LCD_LINE1	displays 16 characters on the first line of a 16X2 LCD display module
LCD_LINE2	displays 16 characters on the second line of a 16X2 LCD display module
SEL_INST	selects instruction before writing a LCD module
SEL_DATA	selects data before writing a LCD module
WRT_PULSE	generates a write pulse for LCD module
SEVEN_SEGMENT:	converts Accu A to its segment pattern, bit 7 of the Accu A = Decimal Point of the 4 digit LED display module.

Jump Table

$0800	RS485_RECV
$0803	RS485_XMIT
$0806	GET_DATE
$0809	GET_TIME
$080C	OUTSTRG00
$080F	LCD_INI
$0812	LCD_LINE1
$0815	LCD_LINE2
$0818	SEL_INST
$081B	SEL_DATA
$081E	WRT_PULSE
$0821	SEVEN_SEGMENT

The subroutine SEVEN_SEGMENT is a conversion routine that converts a hex numbers to its seven segment pattern. The hex number range is from $00 to $20.

At entry, A= hex #	At exit, A=segment pattern	The letter or number to be formed by the segment pattern
A=$00	A=$3F	0
A=$01	A=$06	1
A=$02	A=$5B	2
A=$03	A=$4F	3
A=$04	A=$66	4
A=$05	A=$6D	5
A=$06	A=$7D	6
A=$07	A=$07	7
A=$08	A=$7F	8
A=$09	A=$6F	9
A=$0A	A=$77	A
A=$0B	A=$7C	B
A=$0C	A=$39	C
A=$0D	A=$5E	D
A=$0E	A=$79	E
A=$0F	A=$71	F
A=$10	A=$3D	G
A=$11	A=$76	H
A=$12	A=$74	h
A=$13	A=$1E	J
A=$14	A=$38	L
A=$15	A=$54	n
A=$16	A=$63	o
A=$17	A=$5C	o
A=$18	A=$73	P
A=$19	A=$50	r
A=$1A	A=$78	t
A=$1B	A=$3E	U
A=$1C	A=$1C	u
A=$1D	A=$6E	Y
A=$1E	A=$08	_
A=$1F	A=$40	--
A=$20	A=$00	blank

IMPORTANT NOTES

The following are some important notes that you should know and they may save you time:

1. Things to do if the board does not work.

Many little mistakes can cause a big problem, especially for new beginners. For instance, you may debug your code in expanded mode, but MODEB and MODEA are set for bootstrap mode. If the jumper on J10 is missing, the 68HC11 won't start.

Before troubleshooting the board, you must apply power to the board. Pressing the reset button, the LED should blink twice. If this does not blink, the board may not have 5V DC. Use a DMM to check voltages at the VCC test point. Sometimes it may be caused by a bad AC adapter or the AC adapter may not even be plugged in.

Sometimes the Config register has a wrong value, but the 68HC11 chip is still good.

To determine if the board malfunctions, you can restore the board jumper settings to the original default settings when you received the board. The default settings are as follows:

J4	Code Warrior's Hi-Wave C source level debugger selector. The jumper is installed by a cut-off trace on the left side (NCW) for BUFFALO monitor or Wytec monitor operations.
J7	Mode selector. No jumpers installed.
J10	Clock selector. The jumper is on the upper position (labeled with 'INT') by a cut-off trace, the clock is provided by the on-board crystal.
J12	HC11 SCI receiver source selector The jumper is on the right side (labeled with 'RS232')
J15	Monitor selector. It's in the middle position for the BUFFALO
J17	CodeWarrior's Hi-Wave C source level debugger selector. The jumper is installed by a cut-off trace on the left side (NCW) for BUFFALO monitor or Wytec monitor operations.
J20	The DBUG/RUN jumper is used for selecting an operating mode. It's on the left side for debugging mode.
J21	jumper is installed by a cut-off trace for enabling single-step

If all above settings are correct, when you press the reset button, the PB0 LED should blink twice. If it does not occur, you can try to enter test mode to determine if the 68HC11 chip is bad. Sometimes the Configuration register has a wrong value, but the 68HC11 chip is still good and will work in test mode. In test mode you can use the F8 function key to re-program the value of the Configuration register to $0D. If you cannot enter test mode, the 68HC11 is defective.
To enter test mode, you have to change the jumper on J15 to the top position for enabling Wytec monitor. Then press the reset button momentarily while holding down the PA0 switch to enter test mode. The PB0 LED should flash twice. If it does not flash, the 68HC11 is probably dead. The mode indicator 'EXP' on the top line of the screen should be changed to 'TST'.

The crystal is not soldered to the board and it's held by two machine pins. If you want to change the crystal, just unplug it and replace it with a new frequency, such as 9.8304 MHz. Don't cut a crystal's leads too short. If it's shorter than ¼", its metal case could short the two machine pins.

2. Always reset the board before downloading a new program.

If the previous application program that you ran was aborted, then you may need to reset the board before downloading a new application program. The reset action will disable the interrupt that was enabled by the previous application. If the interrupt was caused by a timer and is not disabled, the timer interrupt will continue even it’s not called for in your new application program. The result will be unpredictable.

3. Disconnect LCD module from the bottom of the main PCB first.

The LCD display module has 2 supporting nylon spacers. They fit tighter with the LCD module PCB than with the main PCB. So if you need to remove the LCD module from the board, the spacers should stay with the LCD module. That means you should use a pair of pliers to push up the spacers from the bottom of the main PCB to separate the spacers from the main PCB. Once the spacers are loose from the main PCB, you can easily unplug the LCD module.

4. Keep folder name and file name not to exceed 8 characters when working under Wytec Monitor.

The Wytec debugger is a DOS application. So keep folder and file names short, otherwise the debugger may not be able to find your application program.