CMD711-EX

Development Board

xiom
anufacturing

717 Lingco Dr., Suite 209 • Richardson, TX 75081 • (972) 994-9676 FAX (972) 994-9170
email: Gary@axman.com • web: http://www.axman.com

CONTENTS

GETTING STARTED

The Axiom CMD711-EX single board computer is a fully assembled, fully functional development system for the Motorola 68HC11E Microcontroller series, complete with wall plug style power supply and serial cable. To get started quickly, perform the following test now to make sure everything is working correctly:

1. Install the software on your PC by running the SETUP.EXE program on the disk.
2. Connect one end of the supplied serial cable to a free COM port on your PC. Connect the other end of the cable to the DB-9 Serial COM1 port on the CMD711-EX board.
3. Apply power to the board by plugging in the wall plug power supply that came with the system.
4. Run the programming utility installed from the disk. You can use either the Windows version (AXIDE) or DOS version (AX11E) of the utilities. Do not run both at the same time.
5. If you're using the DOS program:
   - Change to the directory containing the utility software and execute the program: **AX11E.EXE**.
   - Select the PC COM port that you're using.
   - From the main menu select "Terminal Window".
6. If you're using the Windows program:
   - Double click the AXIDE icon to start the program.
   - Select the PC COM port that you're using from the Options window.
7. Press then release the RESET button on the CMD711-EX board now.
8. If everything is working properly, you should see the buffalo monitor prompt in the terminal window. Your board is now ready to use!

If you do not see the monitor/debugger message prompt, or the text is garbage, see the **TROUBLESHOOTING** section of this manual.

NOTE that buffalo monitor is loaded at the factory and the jumpers set to run the program when the board is shipped. If you did not receive the board from Axiom, or if it has been used previously, the Buffalo Monitor program may have been overwritten. You can re-program it easily from the file named BUF34E.S19. located on the disk.

DEVELOPMENT PHILOSOPHY

The CMD711-EX board is memory mapped to emulate the Motorola 68HC11E9 (12K) and 68HC11E20 (20K) memory maps for program storage. Custom memory configurations can be produced at the factory.

Software development on the Axiom CMD711-EX development board can be performed using the Buffalo Debug Monitor utility, located in the re-programmable external EEPROM in U5, to create or assist in creating your program stored in RAM. During this "debug" mode your program should locate itself above the internal register block, for example $2000 (see the **Memory Map** section for details).

After satisfactory operation running under the monitor environment, your program can easily be written to the external EEPROM Program Memory in U5 by relocating it to start at the beginning of EEPROM memory (which is usually $D000) then selecting Program from the menu. When programming is complete, you simply turn off the mode switches so your program will run automatically when the board is powered on or RESET is applied.

Using the external EEPROM this way is very convenient since you can re-write the same memory chips over and over many times without any special UV erasing, which would require chip removal.

To return to "debug" mode under the Monitor you must re-program the monitor utility in EEPROM. The file name is `BUF34E.S19`.

Programming this file will re-install the Buffalo Monitor starting at $E000 and will program the reset vector address ($FFFE - $FFFF) to point to $E000. A common mistake is to forget to program this reset vector in your own code when running it from EEPROM.

If you want to distribute your software in a board you have designed, you can program it into Internal (onchip) EPROM, or OTP, in U1 by selecting Internal Program Memory from the menu instead of External Program Memory. The micro in U1 can then be removed and replaced with a blank one.

SOFTWARE

There are many useful programs on the software disk that make developing your projects easier. You can download the latest update to this disk free at any time from our web page at: **http://www.axman.com**.

The main programming interface to the CMD711-EX board is included as both a DOS program named AX11E.EXE and a Windows program named AXIDE.EXE. The windows program is a 32-bit application that requires Windows 95 or greater. You can use the DOS utilities standalone or in a DOS window under Windows 3.1.

Both versions of the software communicate serially with the boards COM1 port and include a Terminal window for interfacing with other programs running on the board, such as the buffalo monitor/debugger program.

Most communications programs will also work with the CMD711-EX board. Even the Terminal program in Windows will do an adequate job. Communications settings should be set to 9600 baud, 1 start, 1 stop, 8 data, no parity.

See the **README** file for a complete listing of all programs and files on the disk.

TUTORIAL

This section should help you get started with the specifics of the CMD711-EX development process. Be sure to read the rest of this manual as well as the documentation on the disk if you need further information.

The following sections take you through the complete development cycle of a simple "hello world" program. Since the methods for using the DOS version of the software are almost identical to using the Windows version, only the Windows version is used in this tutorial. You should however be able to substitute the DOS version of the programming software AX11E.EXE and follow this tutorial without much difficulty.

Creating source code

The first step in any software development process is creating source code. You can write source code for the CMD711-EX board using any language that compiles to 68HC11 instructions. Included on the software disk is a free Assembler and also a freeware Basic compiler written by Karl Lunt. Inexpensive or free compilers are also available for C and Forth languages.

You should write your source code using a standard ASCII text editor. Many powerful code editors are available or you can use the free EDIT or NOTEPAD programs that come with your OS. Once your source code is written and saved to a file, you can assemble or compile it to a Motorola S-Record (hex) format. This type of output file usually has a .MOT or .HEX or .S19 file extension and is in a format that can be read by the programming utilities to be programmed into the CMD711-EX board.

IT IS IMPORTANT to understand your development board's use of Memory and Addressing when writing source code so you can locate your code at valid addresses. For example, when in "debug" mode, you should put your Program CODE in External RAM. In assembly language, you do this with ORG statements in your source code. Any lines following an ORG statement will begin at that ORG location, which is the first number following the word ORG, for example: ORG $2000.

You must start your DATA (or variables) in a RAM location unused by your program, for example: ORG $1040. When finished debugging, you must change these ORG statements so that your program is moved to a valid EEPROM area - somewhere after hex D000. Do this by putting an ORG $D000 in front of your Program CODE. Data may remain where it is or be moved down to internal RAM starting at ORG $0000. You must also program the STACK register somewhere at the top of your available RAM, for example hex 1FF. Do this with this instruction as the first instruction in your program code: LDS #$01FF.

A look at the example programs on the disk can make all of this clearer. If you're using a Compiler instead of an assembler, consult the compiler documentation for methods used to locate your code and data.

Using the Programming Utilities

For this tutorial we'll use a sample assembly language program in the EXAMPLES sub-directory called HELLO.ASM. This small program sends a text string to your PC serial port.

1. If you haven't done so already, verify that the development board is connected and operating properly by following the steps under GETTING STARTED.
2. From the Menu, select Assemble.
3. Change to the Examples directory and double-click the source file HELLO.ASM. This will launch a DOS window and execute the assembler to assemble the hello program.
4. If any errors are found they are displayed in this window.

You should now have a new file called **HELLO.S19** (a Motorola hex object file) in the EXAMPLES directory. The HELLO program is memory mapped to EXTERNAL RAM, starting at address $2400. Follow these steps to load and execute the program using the Buffalo Monitor program:

1. Make sure all MODE switches are OFF and apply power to the board or press RESET to get a monitor prompt in the terminal window.
2. Hit ENTER to get a > prompt then type LOAD T and press ENTER. This will prepare buffalo to receive a program.
3. Press the Upload button.
4. Select the file named HELLO.S19 in the EXAMPLES sub-directory.
5. The program will be sent to the development board through the serial port. When finished uploading, press the ENTER key. If you are familiar with hex record format, you can see the first S1 record starts at $2400.
6. Type CALL 2400 and press ENTER.
   This tells the monitor to execute the subroutine at address $2400, which is the start of our test program.

If everything is working properly you should see the message "Hello World" echoed back to your terminal screen and, since we return at then end of the hello program, a status message and lines containing the internal register status displayed by the monitor. You are then returned to the monitor prompt.

If you do not get this message try going through this tutorial once more then, if still no go, see the **TROUBLESHOOTING** section. If the HELLO.ASM file has been modified you'll may need to re-load it from the disk or change it back for this to work.

You can modify the hello program to display other strings or do anything you want. The procedures for assembling your code, uploading it to the board and executing are the same.

The Buffalo monitor has many powerful features such as breakpoints, assembly/disassembly, memory dump and modify and program trace. Type HELP at the prompt for a listing of commands.

Programming the External EEPROM

When finished with program development you'll probably want to write your program to EEPROM so that it executes automatically when you apply power to the board.

1. Use a text editor to modify HELLO.ASM. Change the start of your program, **ROMBS**, to **$E000** which is near the beginning of the EEPROM in U5.
2. Remove the comment * character before the first line after **START**. This will initialize the stack pointer which is necessary when running outside of buffalo but should not be done while running under buffalo since buffalo must handle the stack.
3. Add a comment * character before the first RTS instruction at the end of the main loop so that the program will perform an endless loop when finished, instead of returning.

4. Re-Assemble the HELLO.ASM program as described previously.

5. Select “Program” from the toolbar and when prompted for a file name, select **HELLO.S19** from the examples directory and choose External Program Memory from the options.

6. Follow the instructions on screen. When finished programming, be sure to reset the mode jumpers.

7. Cycle power or press the RESET button on the board. The program should now start automatically.

To return to development mode, repeat the above steps starting with number 5, substituting BUF34E.S19 instead of HELLO.S19.

Programming Internal EPROM

If you wish to distribute your software on-chip (either in EPROM or OTP versions of the micro.), you can program it into U1 using the same method described above, just select Configure and enable the ROMON bit then select Internal Program Memory in step 5. This will put your program in the EPROM on the microcontroller. This may be either in OTP form or UV erasable (windowed) form.

NOTE: A programming adapter can be purchased from Axiom Mfg. to provide programming for different 68HC11 package styles or for easier insertion and removal.

USING THE BUFFALO MONITOR

The debug monitor program supplied for this board is called BUFFALO (Bit User Fast Friendly Aid to Logical Operations). This is a freeware command driven debugger originally written by Motorola that has been modified slightly to work with this development board. This software provides an embedded means for uploading your application software to RAM and executing it in a controlled environment using software breakpoints, trace and memory monitoring features.

The monitor program communicates via the HC11 serial port (SCI) interface. It is executed from EEPROM (external to the MCU). When you first receive your board, the monitor is programmed into the external EEPROM at locations $E000 - $FFFF.

IMPORTANT: If you program external memory you will probably over-write whatever program is currently in memory at the time, including the monitor program, and that program will no longer start on reset. To restore the Buffalo Monitor, you must re-program the EEPROM, using the programming utility software supplied, with the file named BUF34E.S19 which is on the software disk.

While writing your software to debug under the monitor, be aware that the monitor program itself uses internal RAM located at $0036-$00FF and the control registers located at $1000-$103F. The monitor program also uses Output Compare 5 (OC5) and XIRQ for the TRACE instruction, so OC5 and XIRQ should not be used in any of your routines being traced.

OPERATING INSTRUCTIONS

To execute the monitor software (or any application running from EEPROM) the MODE switches must be in normal run mode state, usually all OFF. If you have the serial cable connected, an open com terminal window and the Buffalo Monitor in EEPROM, pressing the RESET button will display a message prompt as follows:

```
BUFFALO 3.4AX - AXIOM VERSION of MOTOROLA MONITOR / DEBUGGER
```

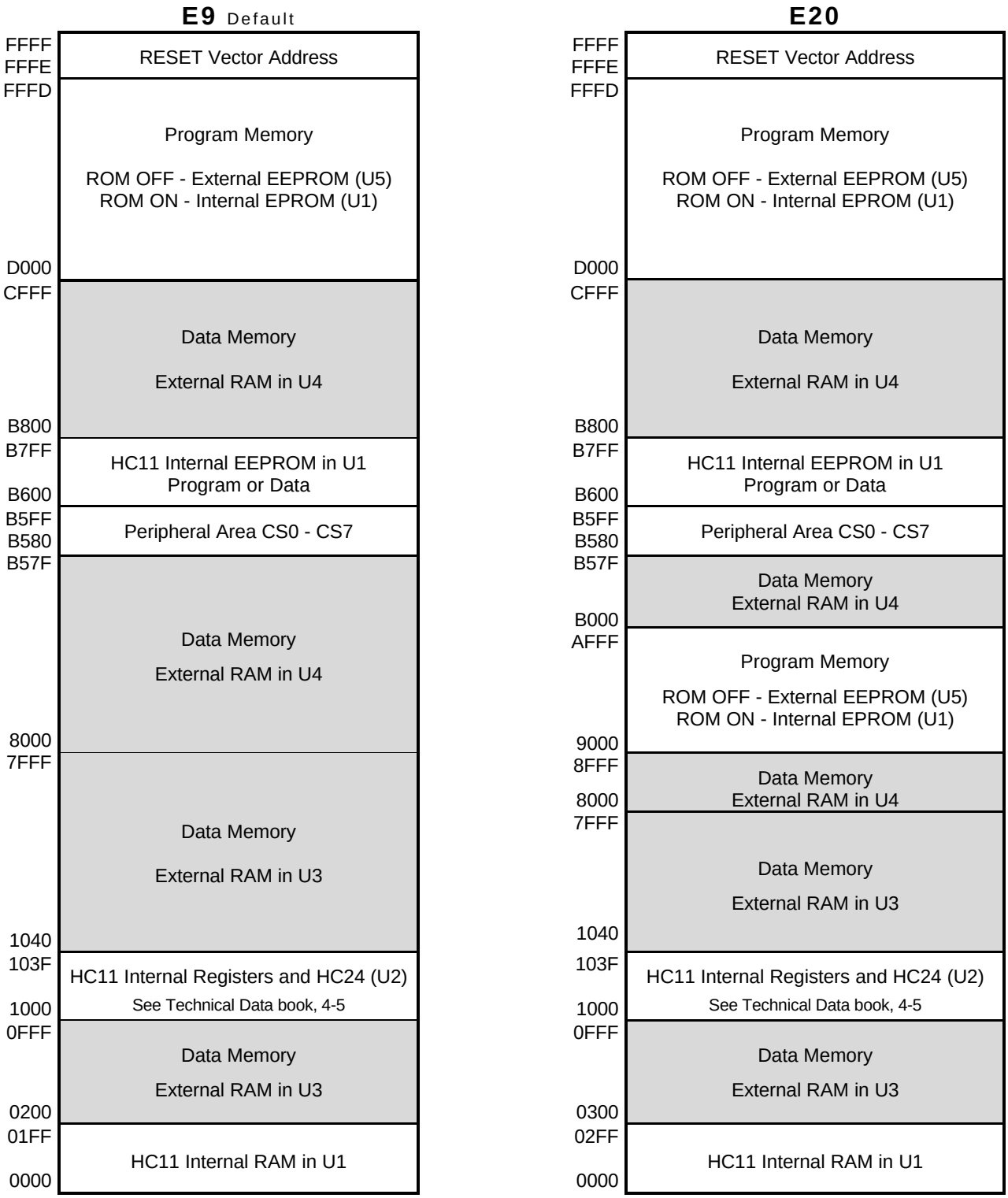
Press ENTER to continue, this will bring up a prompt, which is a > sign. You can now begin to enter commands, upload code or view memory contents.

Type HELP and press enter to get a quick list of commands available. For complete documentation of all the commands available with the monitor and how to use them, see the online help or the file on the disk called `buffalo.txt`.

MEMORY MAP

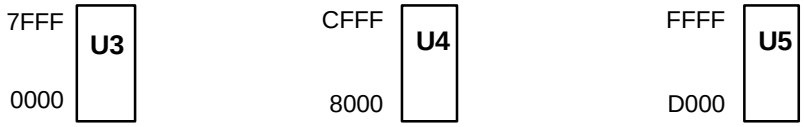
Following is the **DEFAULT EXPANDED MODE** memory map for this development board. It is identical to the HC11 E series memory maps shown in section 4 of the technical data book. See this section for other non-expanded modes - single chip, special test and bootstrap.

In default (E9) non-programming mode, all Mode Switches are off and the ROMON config. bit is disabled (OFF).



ADDRESS DECODING

IC sockets **U3** and **U4** hold 32k RAM chips for external data storage and for debugging programs. **U5** can accommodate EEPROM or ROM but is primarily used for program storage using an EEPROM. Jumpers JP1 through JP7 and Mode Switches 4-5 determine how U5, U6, and U7 are used.



Address decoding is accomplished on the board using a GAL22V10 (U7) programmable logic device. Address lines A<8:15>, AS (address strobe), R/W (read/write), and E (clock) are processed to provide the memory control signals as shown below by default. Custom configurations, differing from that shown below, are also possible. Contact the factory for assistance in redefining the memory map if required.

OE Output enable to U3, U4, U5 and BUS_PORT

WE Write enable to U3 direct and to U4 and U5 through jumper JP4 and Mode Switch 3 respectively.

M1 Chip select to U3 active from 0 to 32k, 0000 - 7FFF.

M2 Chip select to U4 active for the 24k between 8000 and DFFF, with the exception of B580 through B7FF inclusive. B580 to B5FF is used by both P and CS0-7. B600 to B7FF is taken by the HC11 internal EEPROM, if enabled.

M3 Chip select to U7 active for the 8k between D000 and FFFF.
NOTE**:** Mode switches 4-5 effect the memory map for M2 and M3.

P Peripheral Access CS0 - CS7. B580 through B5FF.

All of these signals are active low. Signal line M2 is also connected to the BUS_PORT expansion connector allowing M2 to work in conjunction with the CS and Address lines to implement off board, page banked memory. When M2 is used in this manner, U6 must be removed from the board.

Peripheral Access 'P' is used in conjunction with A<4:6>, and AS to generate CS<0:7>. Each of these eight chip selects controls sixteen bytes in the memory map from B580 through B5FF. CS<7:0> are brought out to the BUS_PORT where they can be used to control peripherals external to the development board. See the Memory Map for further clarification. CS7 is used for the LCD_PORT.

Single Chip Mode

To operate the HC11 in "True Single Chip" mode, Mode Switch position 1 should be ON and the BUS_EN jumper installed to disable bus operation. In this mode BUS_PORT D0-7 are HC11 port C and A8-15 are HC11 port B. In expanded mode the HC24 Port Replacement Unit (U2) replaces these ports on the MCU_PORT connector.

HARDWARE

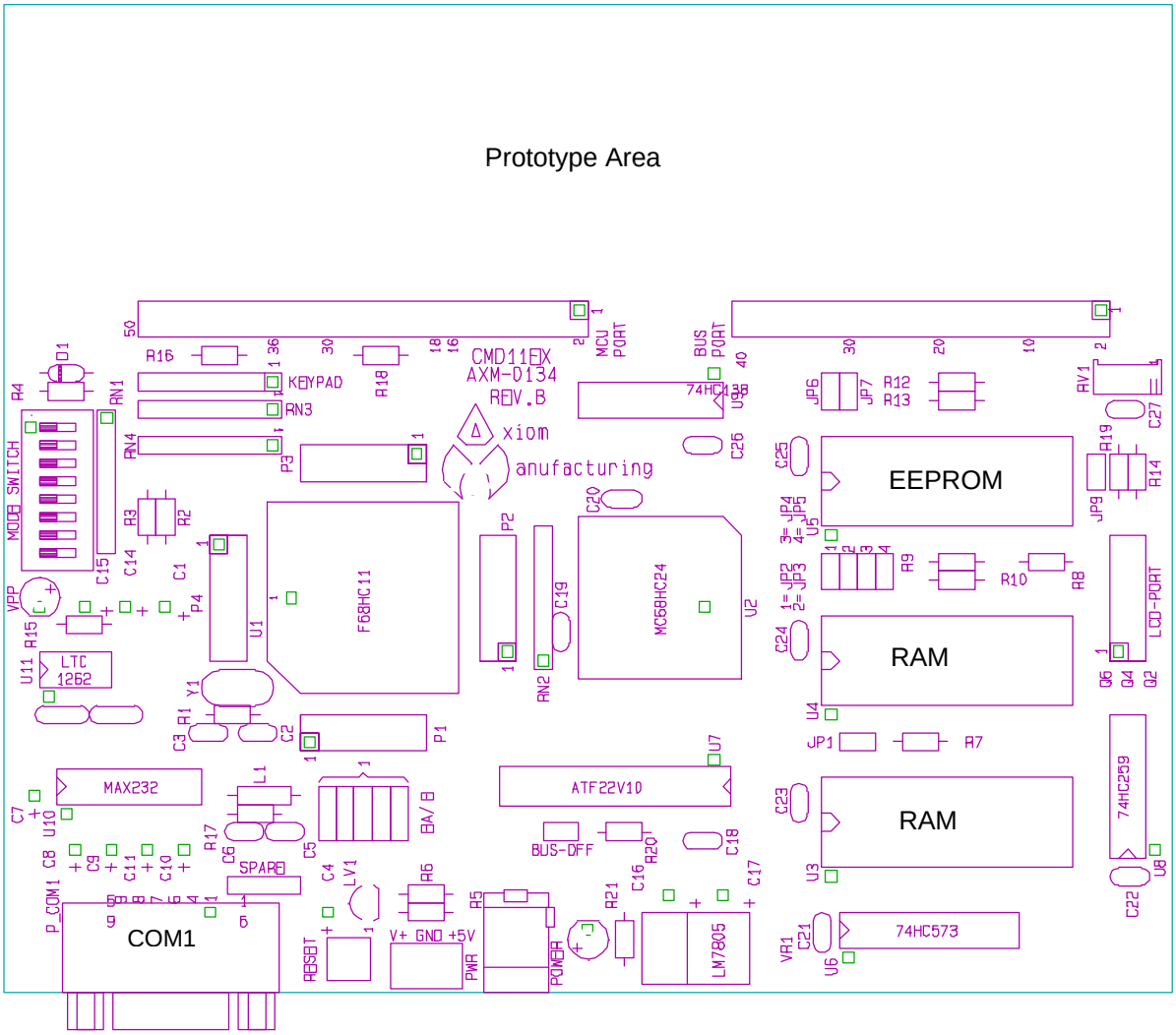
SPECIFICATIONS

Oscillator	9.8304 MHz
E Clock	2.4576 MHz
Operating temperature	0°C to +70°C
Power requirement	Nominal 9V, +8 to +25V @ 100 ma.
Programming Voltage	+12V @ 30 ma. supplied with mode switch 8 ON

Devices Supported

MC68HC11E0FN
MC68HC11E1FN
MC68HC11E8FN
MC68HC11E9FN
MC68HC711E9FN / FS
MC68HC11E20FN
MC68HC711E20FN / FS
MC68HC11EA9FN
MC68HC711EA9FN / FS

Additional devices supported with optional adapters. Contact Axiom Mfg. for availability.



JUMPERS AND SWITCHES

JP1	ON enables A13 to U3. Normally this should never be removed, except when using an 8K device in U3.
JP2	ON enables A14 to pin 1 of U4 (RAM or EEPROM)
JP3	ON enables A14 to pin 27 of U4 (EPROM only). Not installed if JP4 is ON.
JP4	ON write enables U4 pin 27 (RAM or EEPROM). Not installed if JP3 is ON.
JP5	ON enables A13 to pin 26 of U4. Removed for installing an 8k device.
JP6	ON enables A14 to pin 1 of U5 (EEPROM)
JP7	ON enables A14 to pin 27 of U5 (EPROM)
JP9	ON synchronizes the CS0 - CS7 chip selects with the E CLOCK, which allows for simple latches (374 or 245 for example) to be used with CS0 - CS7. OFF synchronizes the chip selects with valid address decoding. Control signals OE & WE should be used with peripherals connected to the chip selects in this mode.
BUS_EN	ON disables the HC24 Port Replacement unit (U2) and all external memory devices and bus operation. This is "true single chip" mode with Mode Switch position 1 ON.

EA/E Jumpers

This block of 5 jumpers allow configuring for the EX series (default) or EA type HC11 micro-controllers. You can use these jumpers to reconfigure the EX pin connections of the 52-pin PLC device to support the EA series parts. See the schematic for more information.

MODE SWITCH

Normally the MODE SWITCH should be set to **ALL OFF** which is normal running mode. The individual switch settings are as follows:

1	ON configures the HC11 MODA to Ground or 0 volts. See table.
2	ON configures the HC11 MODB to Ground or 0 volts. See table.
3	ON Write Enables External Program memory (U5).
4	ON Enables 20K program space in the external memory map.
5	ON Disables the Peripheral Chip Selects
6	ON Enables monitor TRACE operation using PA3 / XIRQ.
7	ON Enables Internal Memory Programming using Vpp / XIRQ.
8	ON Turns on Vpp Programming Voltage

HC11 MODE Table

1	2	Mode
on	on	Bootstrap / utility
on	off	True Single Chip
off	off	Normal Expanded

RESET SWITCH

Reset is provided by an 8054 low voltage detector when Vdd falls below approximately 4.4 volts. Manual reset is provided by the RESET push button located next to the COM1 port on the board.

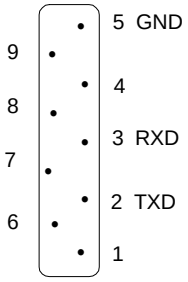
PORTS and CONNECTORS

SERIAL PORT

The onboard serial port COM1 is a simple, three wire asynchronous serial interface with hard wired Clear to Send (CTS) and Data Terminal Ready (DTR). It is driven by the HC11 internal SCI port using I/O pins PD0 and PD1. These two logic level signals are coupled through a RS232 level shifter to the COM1 connector.

COM1 is the default serial interface for the Buffalo Monitor and programming software.

COM1 DB9S Style Connector



- Cut-Away jumpers between the following pins:
  4 → 1 and 6 (DTR/DSR/DCD)
  7 → 8 (RTS/CTS)
- COM1 is set to connect directly to a PC serial port with a straight through type cable (supplied).

KEYPAD Connector

The KEYPAD connector is an eight position connector that implements 4 bits of Port D and 4 bits of Port E as a passive keypad interface. This interface is implemented as a software keyscan. Pins PD2 - PD5 are columns which are Set High to read the row conditions on PE0 - PE3 which are Active High. See the file **KEYLCD-E.ASM** on the software disk for an example program using this keypad connector.

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
D2	D3	D4	D5	E0	E1	E2	E3

LCD_PORT Connector

The LCD Display interface is connected to the data bus and memory mapped to locations $B5F0 through $B5F3. Addresses $B5F0 and $B5F1 are the Command and Data registers respectfully. The LCD interface supports all OPTREX™ DMC series displays up to 80 characters and provides the most common pinout. Power, ground and Vee are also available at the LCD_PORT connector.

See the file **KEYLCD-E.ASM** on the software disk for an example program using this LCD connector.

+5V	**2**	**1**	GND
A0	**4**	**3**	Vee
/LCDCS	**6**	**5**	/RW
D1	**8**	**7**	D0
D3	**10**	**9**	D2
D5	**12**	**11**	D4
D7	**14**	**13**	D6

B5F0 LCD Command Register
B5F1 LCD Data Register

SPARE Connector

The Spare connector supplies 2 channels of RS232 serial translation for developer use as follows:

4	Receive Data TTL level
3	Transmit Data TTL level
2	Transmit Output RS232 level
1	Receive Input RS232 level

P1, P2, P3 and P4 Headers

This set of 4, 14-pin connectors can be used to connect an external Programming Adapter or an Emulator Interface to the board. See the Technical Data Book for detailed information on all of these lines.

P1

XTAL	1	2	D0
D1	3	4	D2
D3	5	6	D4
D5	7	8	D6
D7	9	10	/ RESET
/ XIRQ	11	12	/ IRQ
PD0 / RXD0	13	14	GND

P3

PA0	1	2	A15
A14	3	4	A13
A12	5	6	A11
A10	7	8	A9
A8	9	10	PE0
PE4	11	12	PE1
PE5	13	14	GND

P2

PD1 / TXD0	1	2	PIN22
PIN23	3	4	PIN24
PIN25	5	6	+5V
PA7	7	8	PA6
PA5	9	10	PA4
PA3	11	12	PA2
PA1	13	14	GND

P4

PE2	1	2	PE6
PE3	3	4	PE7
V_{RL}	5	6	V_{RH}
GND	7	8	PIN2
MODA	9	10	/AS
ECLK	11	12	/RW
EXTAL	13	14	GND

BUS_PORT and MCU_PORT Connectors

The **BUS_PORT** supports off-board memory and peripheral devices. The **MCU_PORT** provides access to the peripheral features and I/O lines of the HC11. See the Technical Data Book for detailed information on all of these lines.

BUS_PORT

GND	1	2	D3
D2	3	4	D4
D1	5	6	D5
D0	7	8	D6
A0	9	10	D7
A1	11	12	A2
A10	13	14	A3
/OE	15	16	A4
A11	17	18	A5
A9	19	20	A6
A8	21	22	A7
A12	23	24	A13
A14	25	26	A15
/WE	27	28	CS0
CS1	29	30	CS9
CS3	31	32	CS4
CS5	33	34	/IRQ
+5V	35	36	M2
ECLK	37	38	CS6
GND	39	40	CS7
GND	41	42	/ RESET

MCU_PORT

+5V	1	2	+5V
PC0	3	4	PC1
PC2	5	6	PC3
PC4	7	8	PC5
PC6	9	10	PC7
STRA	11	12	STRB
PB6	13	14	PB7
PB4	15	16	PB5
PB2	17	18	PB3
PB0	19	20	PB1
PA0	21	22	PA1
PA2	23	24	PA3
PA4	25	26	PA5
PA6	27	28	PA7
PE1	29	30	PE0
PE3	31	32	PE2
PE5	33	34	PE4
PE7	35	36	PE6
PD0 / RXD0	37	38	PD1 / TXD0
PD2 / SI	39	40	PD3 / SO
PD4 / SCLK	41	42	PD5 / SEL0
V_{RL}	43	44	V_{RH}
/AS	45	46	MODA
/XIRQ	47	48	MODB
GND	49	50	GND

I/O POINTS

The following connector points on the board do not have connectors attached but are provided for easy access.

COM1 Points

These points 9 8 7 6 4 1 located next to the DB-9 serial port connector COM1, can be used to connect the COM1 lines that are unused by the DB-9 connector to provide hardware handshaking back to the PC or other connected device. The board is shipped with all points (except 9) connected by cut-away traces on the bottom of the board.

Q2, Q4 and Q6 Points

These points, located near the LCD port, are provided for decoding peripheral control signals. They are all Active High, and can be used to connect larger LCD panels or simple 373 style latches for example.

Point	Address
Q2	B5F4
Q4	B5F8
Q6	B5FC

MISCELLANEOUS

Power Jack

The required 9V operating power can be supplied by inserting the "barrel" type connector of the supplied wall plug adapter into the power connector on the board, located near the serial port.

External power and ground points, beside the power connector, can also be used as alternate input power and ground to the board OR to draw external power and ground from the board for external devices.

LED Indicators

GREEN	Power ON
RED	VPP Programming Voltage ON

Monitor Trace Switch

Mode Switch 6 is used by the included Buffalo Monitor for Trace and Single Step functions. These functions can be enabled by setting Mode Switch 6 to the ON position. This will connect the PA3 Output Compare 5 pin to the XIRQ line for nonmaskable interrupt service to implement software Trace and Single Step.

Exercise caution when using this mode that no other connections are made to the XIRQ or PA5 I/O pins of the 68HC11.

A/D Reference

The VRH and VRL lines from the HC11 are connected to +5v through R3 and Ground through R2. These two surface mount resistors are on the bottom (solder) side of the circuit board. The resistors are identified on the silk screen by their reference designators and the dashed line box that surrounds them. The appropriate resistor(s) need to be removed in order to apply an external reference to the VRH and/or VRL inputs.

TROUBLESHOOTING

The CMD711-EX board is fully tested and operational before shipping. If it fails to function properly, inspect the board for obvious physical damage first. Ensure that all socketed IC devices are properly seated in the their sockets.

Verify that your PC communications port is working by substituting a known good serial device, or by doing a loop back diagnostic. Verify that no other devices are conflicting with your PC serial port (such as a mouse, modem, etc.).

Check your hardware configuration jumpers and switches. Verify the power source. You should measure 9 volts between GND and +9V test points on the board. If no voltage is found, verify the wallplug connections to 115V AC outlet and power connector. Disconnect all external connections to the board except for COM1 to the PC and the wall plug. Follow these steps in the order given:

Hardware Troubleshooting Steps

1. Visual Inspection

2. Verify that all MODE switches are switched OFF.

3. Verify power by checking for +9 volts between GND and +9V test point pads. The Green LED should be ON.

4. Verify the Monitor EEPROM for proper installation (no bent pins) and proper jumper settings for the device used.

5. Check the Configuration Byte. Normally ROMON should be disabled (OFF). If accessing Internal EPROM then ROMON should be ON.

6. Disconnect any peripheral devices including the LCD and Keypad.

7. Make sure that the RESET line is not being held low.

8. Verify the presence of a 9.8MHz sine wave on the crystal if possible.

Please check off these steps and list any others you have performed before calling so we can better help you.

Tips and Suggestions

Following are a number of tips, suggestions and answers to common questions that will solve most problems users have with the AX11E development system. This information is also available in the online help under Troubleshooting, which will have the most complete and updated information. There also may be a newer version of the software available. You can download the latest software from the Support section of our web page at:

www.axman.com

Programming Utilities

- If you're using External EEPROM (default) the ROMON bit in the Configuration Byte must be DISABLED (off). If you're using Internal (onchip) EPROM the ROMON bit must be ENABLED (on).
- If you're trying to program or read memory, make sure all the MODE switches are set properly and the Configuration Byte is programmed to the proper value. Usually, this means Mode Switches 1 & 2 on for external programming, 1,2,7 & 8 on for internal memory.
- Always turn off VPP after programming (switch 8). This will prevent possible damage to the micro.
- If you're trying to execute a program, for example running the Debug Monitor from the Terminal window, the MODE Switches must be reset to normal mode. Usually this means all switches OFF, however check the MODE Switch section if you're not sure.
- Be certain that the data cable you're using is bi-directional and is connected securely to both the PC and the board. Also, make sure you are using the correct serial port.
- Make sure the correct power is supplied to the board. You should only use a 9 volt, 300 mA adapter or power supply. If you're using a power strip, make sure it is turned on.
- Make sure you load your code to an address space that actually exists. See the Memory Map if you're not sure.
- If you're running in a multi-tasking environment (such as Windows™) close all programs in the background to be certain no serial conflict occurs.
- If the Assembler menu option doesn't work properly in the DOS utilities you can modify it's operation by editing the file DO_ASM.BAT in the AX11E directory. This can also be done from the Options menu.

Code Execution

- Always remember to move the MODE switches back to their correct positions after programming memory.
- Make sure the ROMON configuration bit is set properly - see above.
- If you programmed your code into EEPROM memory over the Debug Monitor you will have to re-program the monitor software from the S19 file on the software disk to use it again.
- After programming your application into memory, if it still doesn't run, check the HC11 reset vector located at FFFEh - FFFFh. These 2 bytes contain the address where execution will begin when the unit is powered on.
- If your program worked previously under the debug monitor but does not work after programming it, check all the hardware registers you are using for proper configuration. The monitor initializes some of these registers for it's own use - such as the serial port, stack pointer, timers, etc. You must initialize any used peripheral registers in your own code.