CMD912x

Development Board for Motorola 68HC912 MCU's
68HC912D60/ DG128/ DT128 and MC9S12DP256

CONTENTS

GETTING STARTED

The Axiom CMD912X single board computer is a fully assembled, fully functional development system for the Motorola 68HC912D60/ DG128/ DT128 and MC9S12DP256 microcontrollers, complete with wall plug power supply and serial cable. Support software for this development board is provided for Windows 95/98 and NT operating systems.

Follow the steps in this section to get started quickly and verify everything is working correctly.

Installing the Software

1. Insert the Axiom 68HC12 support CD in your PC. If the setup program does not start, run the file called "SETUP.EXE" on the disk.

2. Follow the instructions on screen to install the support software onto your PC.
    You should at minimum install the AxIDE for Windows software.

3. The programming utility "AxIDE" requires you to specify your board. You should select "912-xxx" version of your development board.

Board Startup

Follow these steps to connect and power on the board for the default Monitor operation. This assumes you're using the provided AxIDE utility (installed in the previous section) or a similar communications terminal program on your PC. If you're using a different terminal program than the one provided, set it's parameters to 9600 baud, N,8,1.

1. Set the CMD912x board **CONFIG SWITCH** as follows:

1	2	3	4	5
ON	**ON**	**OFF**	OFF	OFF

2. Set the PM12Dxx module **MODE SWITCH** as follows:

Module	1	2	3	4	5
PM12D60	**ON**	**ON**	**OFF**	x	x
PM12DG128	**ON**	**ON**	**OFF**		
PM12DT128	**ON**	**ON**	**OFF**		
PM12DP256	**ON**	**ON**	**ON *1**	**OFF**	**OFF**

    Note 1: DP256 mask set 1K79X and earlier requires Expanded Wide Emulation Mode.

3. Verify CMD912x board **COM-SWITCH** position **1** is **ON**.

4. Connect one end of the supplied 9-pin serial cable to a free COM port on your PC.
    Connect the other end of the cable to the COM1 port on the CMD912X board.

5. Apply power to the board by plugging in the power adapter that came with the system.

6. If everything is working properly, you should see a message to "PRESS KEY TO START MONITOR…" in your terminal window. Press the ENTER key and you should see:

```
Axiom MON12 - HC12 Monitor / Debugger
> _
```

7. Your board is now ready to use! If you do not see this message prompt, or if the text is garbage, see the **TROUBLESHOOTING** section at the end of this manual.

Support Software

There are many programs and documents on the included HC12 support CD you can use with the CMD912X board.   You should install what you want from the main menu then browse the disk and copy what you like to your hard drive.

At minimum, you should install the AxIDE program.   This flash programming utility communicates with the board via its COM1 port and the supplied serial cable. This program also includes a simple terminal for interfacing with other programs running on the CMD912X and information from your own programs that send output to the serial port.

Also on the disk are free assemblers, the GNU C/C++ compiler tools for HC12, example source code, and other useful software. The introductory tutorial in this manual uses the free AS12 assembler integrated into the AxIDE program. This is a simple assembler with limited capability. For a more powerful assembly tool, install the Motorola MCUez program from the CD. This will allow you to used PAGED memory in your application.

Software Development

Software development on the CMD912x can be performed using either the Mon12 monitor utility installed in EPROM (sockets U6/U7), a third party debugger (NoICE, CodeWarrior, etc.) or a Background Debug Module (BDM) connected to the PM12xx Module BDM PORT connector. Any of these tools can be used to assist in creating and debugging your program stored in RAM (see **Memory Map**).

After satisfactory operation running under a debugger, your program can be written to Internal Flash Memory by changing the PM12xx Module MODE SWITCH settings and programming it using one of the included programming utilities. Your program may l then run automatically whenever the board is powered on or RESET is applied.

Option switches on the board allow for easy transition from operating monitor or debugger and user code.

TUTORIAL

This section was written to help you get started developing software with the CMD912X board. 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, which sends the string "Hello World" to the serial port.

Creating source code

You can write source code for the CMD912X board using any language that compiles to Motorola 68HC12 instructions. Included on the software disk is a free Assembler.

You can write your source code using any ASCII text editor. You can use the free EDIT or NOTEPAD programs that come with your computer. 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, .HEX or .S19 file extension and is in a format that can be read by the programming utilities and programmed into the CMD912X board.

It's 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 locate the code 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 $4400`. You must start your DATA (or variables) in a RAM location unused by your program, for example: `ORG $4000`.

In "debug mode" you'll be using a debugger utility (Mon12, NoICE, etc) which will handle both interrupts (reset, timers, etc) and the STACK. When finished debugging, you must add code to your application to handle the STACK and Interrupt vector initialization.   Set the stack somewhere at the top of your available RAM, for example $3FFE, in assembly this would be `LDS #$3FFE`.   Also define the RESET vector address, $FFFE, at the end of your program. For example:

```
ORG $FFFE
FDB START ; where START is the beginning label of your program
```

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, data and stack.

Assembling source code

An example program called "HELLO.ASM" is provided under the **\EXAMPLES\912x** directory of the CD and if you installed AxIDE, under that programs **\EXAMPLE** directory. You must use the example for the PM Module you have installed on the CMD912x board. The PM Label is located beside the microcontroller.   For example:

```
\EXAMPLE\HC12D60\HELLO.ASM     Example program for the PM12D60
\EXAMPLE\HC12D128\HELLO.ASM    Example program for the PM12D128
\EXAMPLE\HC12D256\HELLO.ASM    Example program for the PM12D256
```

You can assemble your source code using command line tools under a DOS prompt by typing:

```
AS12 HELLO.ASM –LHELLO
```

Most compilers and assemblers allow many command line options so using a MAKE utility or batch file is recommended if you use this method. Run AS12 without any arguments to see all the options, or see the AS12.TXT file on the disk.

The programming utility AxIDE provided with this board contains a simple interface to this assembler. Use it by selecting "Build" from its menu. This will prompt you for the file to be assembled.   **NOTE**: You must select your board from the pull down menu first, or it may not build correctly.

DO NOT use long path names (> 8 characters). The free assembler is an old DOS tool that does not recognize them.

If there are no errors in your source code, 2 output files will be created:

HELLO.S19	a Motorola S-Record file that can be programmed into memory
HELLO.LST	a common listing file which shows the relationship between source and output

The listing file is especially helpful to look at when debugging your program. If your program has errors, they will be displayed and no output will be generated, otherwise the listing file will be displayed.

If you prefer a windows integrated programming environment, try the Motorola MCU-EZ tools. Refer to the MCU-EZ documentation on the disk for more information.

Also, a port for the free GNU C compiler and tools for the HC12 is available on the CD under \Shareware and also online at www.gnu-m68hc11.org.

Running your application

After creating a Motorola S-Record file you can "upload" it to the development board for a test run. The provided example "HELLO.ASM" was created to run from RAM so you can use the Mon12 Monitor to test it without programming it into Flash.

If you haven't done so already, verify that the CMD912X board is connected and operating properly by following the steps under "GETTING STARTED" until you see the Mon12 prompt, then follow these steps to run your program:

1. Press and release the RESET button on the CMD912X board. You should see the PRESS ANY KEY message. Hit the return key ↵ to get the monitor prompt.

2. Type **LOAD** ↵
   This will prepare Mon12 to receive a program.

3. Select Upload and when prompted for a file name select your assembled program file in s-record format that was created in the previous section called: **HELLO.S19**
   Your program will be sent to the board thru the serial port.

4. When finished loading you will see the > prompt again. Type **GO 4400** ↵
   This tells Mon12 to execute the program at address $4400, which is the start of our test program.

5. If everything is working properly you should see the message "Hello World" echoed back to your terminal screen. Press RESET to return to the monitor.

6. If you do not get this message, see the **TROUBLESHOOTING** section in this manual

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 it remain the same. Mon12 has many features such as breakpoints, assembly/disassembly, memory dump and modify and program trace. Type HELP at the Mon12 prompt for a listing of commands or consult the Mon12 documentation on the disk for more information.

For a more powerful debugger with many advanced features such as source level debugging, you can use the NoICE debugger software. A full featured demo version is provided on the CD, which you can use to get started. **NOTE**: To use this program instead of Mon12 you must simply move the CONFIG SWITCH position 3 ON, RESET the board, and run the NoICE software. See the help documentation in this program for more information.

NoICE monitor is not installed on Monitor versions R2 and earlier.

Programming Flash EEPROM

After debugging, you can program your application into Flash Memory so it executes automatically when you apply power to the board as follows:

1. Make a backup copy of HELLO.ASM then use a text editor to modify it.

2. Change the ORG location for the program start to the internal flash if needed.

3. Remove the comment ; character before one of the following lines to initialize the stack pointer which is necessary when running outside of a debugger:

```
LDS      #$3FFE   ; DG/DP128 or DP256 – initialize…
LDS      #$7FE    ; D60 - initialize the stack pointer
```

4. Remove the comment ; character from before the following 2 lines at the end, to set the reset vector to go to the beginning of the program (the label START) when powered on:

```
org    $fffe         reset vector
       fdb    START
```

5. Re-Assemble HELLO.ASM as described in the "Assembling Source Code" section.

6. Select **Program** from the AxIDE menu and follow the message prompts. When prompted for a file name, enter the new HELLO.S19 file.

7. Press the RESET button on the board before clicking OK. When prompted to Erase, choose Yes.

8. When finished programming, REMOVE POWER then set the MODE SWITCH positions 1 and 2 OFF. Note: turn position 3 off also for DP256 version.

9. Re-Apply Power to the board. Your new program should start automatically and the "Hello World" prompt should be displayed in the terminal window.

To return to the Mon12 monitor program, set the MODE SWITCH positions 1 and 2 back ON then press RESET. Note: MODE switch 3 also must be ON for DP256 1K79 and earlier mask sets.

BDM OPERATION

The CMD912X board will emulate supported HC12 device internal flash memory in external ram. This feature allows BDM (Background Debug Modules) such as the AX-BDM12 to load and control the execution of code being developed without the necessity of the internal flash memory being programmed many times during the development process. This feature improves updating time and allows the use of may software breakpoints instead of being limited to only 2 hardware breakpoints.

Operation Notes for BDM use:

1)   CMD912x CONFIG SWITCH should be set 1 ON, 2 OFF, 3 OFF, and 4 ON. Position 5 should be Off unless the DG/DT128 Fixed Page is desired to be emulated in ram. These settings provide external ram and PRU operation with the ECS enabled for Paging emulation.

2)   PM12xxx MODE SWITCH is usually set for positions 1 and 2 off to select Single Chip Mode. This setting forces the BDM Monitor in the HC12 active immediately after Reset with the BDM connected which is desired for most BDM pods. The BDM can then load a script to set the desired operating mode and configuration, see next note. The DP256 version should have Mode Switch position 3 ON for correct operation of the MODC select buffer during RESET.

3)   The BDM initialization of the HC12 should set the correct operating MODE (Expanded Wide for memory access) and enable port emulation for the PRU to operate. The EME, EMK, LSTRB, RW, IVIS, ROMEN and Stretch configuration bits should be set for proper memory map and PRU operation. The external RAM does not require any cycle stretch for accesses up to 25MHz E clock. The Axiom support CD contains sample set-up macros for the AX-BDM12.

MEMORY MAPS

Following is the memory map for the CMD912X development board and the various Microcontroller PM Modules that it supports. Consult your MCU technical reference manual on the CD for internal memory map details for the processor.

PM12DP256 Memory Map

	Expanded Wide Mode	Special (BDM) Expanded Wide Mode	Single Chip Mode
FFFF	**CONFIG** ON ON OFF (1 2 3) **MODE** ON ON ON **External EPROM** U5/6 (Mon12)	**CONFIG** ON OFF (1 2 3) **MODE** OFF OFF ON (see BDM notes)	**CONFIG** OFF OFF OFF OFF (1 2 3 4) **MODE** OFF OFF OFF **On-Chip Flash Memory**
C000			
BFFF	**External RAM** U3/4	**External RAM** U3/4	Flash Page 8000 – C000
8000			
4000			
3FFF	**On-Chip RAM** 3E00 – 3FFF used by Mon12		
1000			
FFF	**On-Chip EEPROM**		
800			
7FF	**Internal Registers** See your MCU Technical Reference Manual		
000			

NOTE: the DP256 does not provide a Register Following Peripheral Area.

PM12DG128 / PM12DT128 Memory Map

	Expanded Wide Mode	Special (BDM) Expanded Wide Mode	Single Chip Mode

FFFF

Expanded Wide Mode

	1	**2**
CONFIG	ON	ON
MODE	ON	ON

External EPROM
U5/6 (Mon12)

Special (BDM) Expanded Wide Mode

	1	**2**
CONFIG	ON	OFF
MODE	OFF	OFF

(see BDM notes)

Single Chip Mode

	1	**2**	**3**	**4**
CONFIG	OFF	OFF	OFF	OFF
MODE	OFF	OFF		

On-Chip Flash Memory

C000

BFFF

External RAM
U3/4

External RAM
U3/4

Flash Page
8000 – C000

8000

External RAM
U3/4

4000

3FFF

On-Chip RAM

3E00 – 3FFF used by Mon12

2000

1FFF

Reserved

1000

FFF

On-Chip EEPROM

800

7FF

Peripheral Area

Unused = **400-7BF**	CS5 = **7D0-7DF**	CS2 = **7A0-7AF**
LCD / CS7 = **7F0-7FF**	CS4 = **7C0-7CF**	CS1 = **790-79F**
CS6 = **7E0-7EF**	CS3 = **7B0-7BF**	CS0 = **780-78F**

400

3FF

Internal Registers
See your MCU Technical Reference Manual

000

1. The Peripheral Area (A00-BFF) is set to Narrow (8-bit) data width by the debug utilities. If using this memory, you must also do this in your software when booting from flash as follows:

```
MOVW  #$0CF0,PEAR
MOVB  #$73,MISC      ; Flash on, p-sel stretch = 3
```

PM12D60 Memory Map

Expanded Wide Mode	Special (BDM) Expanded Wide Mode	Single Chip Mode
CONFIG ON ON **MODE** ON ON **External EPROM** U5/6 (Mon12)	**CONFIG** ON OFF **MODE** OFF OFF	**CONFIG** OFF OFF **MODE** OFF OFF
External RAM U3/4	**External RAM** U3/4	**Internal Flash Memory** On-Chip

FFFF

C000

BFFF

1000

FFF

C00

HC12 Internal EEPROM On-Chip

BFF

Peripheral Area - see note 2 below

Unused = **A00-B7F**	CS5 = **BD0-BDF**	CS2 = **BA0-BAF**
LCD / CS7 = **BF0-BFF**	CS4 = **BC0-BCF**	CS1 = **B90-B9F**
CS6 = **BE0-BEF**	CS3 = **BB0-BBF**	CS0 = **B80-B8F**

A00

9FF

Internal Registers - see note 1 below See 68HC912D60 Technical Reference Manual

800

7FF

Internal RAM On-Chip

000

2. The Internal Register base address is relocated from $000 to $800 on startup by the debug utilities (Mon12 and NoICE). To preserve this memory map, you must also do this in your software when booting from flash. To do this, load register $11 with $08 for example: `MOVB  #08,$11    ; post-reset location of INITRG`

3. The Peripheral Area (A00-BFF) is set to Narrow (8-bit) data width by the debug utilities. If using this memory, you must also do this in your software when booting from flash as follows:

```
MOVW  #$0CF0,PEAR
MOVB  #$73,MISC      ; Flash on, p-sel stretch = 3
```

OPTION SWITCHES

CONFIG SWITCH

The CMD912X board is shipped from the manufacturer with the following default CONFIG SWITCH settings:

1	**2**	3	4	5
ON	**ON**	OFF	OFF	OFF

The 5 position CONFIG SWITCH provides an easy method of configuring the CMD912X board memory operation. Following are the configuration switch descriptions:

CONFIG SWITCH	OPERATION when in ON position
1	EXT – External Memory and PRU enable [1]
2	MON – Monitor Memory enable [2]
3	MON SEL - Select NOICE Debug kernel
4	ECS Enable - Enable ECS (Emulation Chip Select) signal to ram for paging emulation on Devices larger than 60K
5	FPAGE Enable - Enable DG/DT128 Fixed Page at $4000 hex

[1] Enables memory bus operation for access to board memory and PRU. Expanded Wide bus mode must be enabled for proper operation.

[2] Enables monitor EPROM's in memory map at 0xC000 – FFFF hex if CONFIG SWITCH position 1 is also on. When in off position memory space is SRAM for BDM use.

COM SWITCH

The 6 position COM SWITCH on the CMD912x Board provides an easy method of connecting or isolating the HC12 SCI and CAN channel RXD pins from the provided on-board transceivers. The HC12 SCI channels are connected to RS232 transceivers and the first 4 CAN channels are connected to 1M baud CAN transceivers. To apply the RXD pins on the channels for other user applications requires that the transceiver driver be removed from the HC12 pin. User may then apply signals to the respective pins at the MCU PORT connector without driver conflict. Please note that the on-board monitor(s) require HC12 SCI channel 0 for user interface.

COM SWITCH	HC12 Connection in the ON position
1	SCI0 RXD to COM1 RS232 Transceiver [1]
2	SCI1 RXD to COM2 RS232 Transceiver
3	CAN0 RXD to CAN1 Port Transceiver
4	CAN1 RXD to CAN2 Port Transceiver
5	CAN2 RXD to CAN3 Port Transceiver
6	CAN3 RXD to CAN4 Port Transceiver

[1] Must be ON if using on board Monitor firmware.

MODE SWITCH

The 3 to 5 position MODE SWITCH on the PM12xxx Module provides an easy method of configuring the HC12 operating Mode and Options from RESET.

NOTE: Expanded Narrow Mode is not available on this board, Expanded Wide operation is required due to the memory application and that any expanded bus operation requires both HC12 I/O ports A and B in any case.

Following are the Mode Switch selections and descriptions:

PM12D60

Default: 1 and 2 ON, 3 - 5 OFF.

MODE SWITCH	SELECTION OPTION	OFF POSITION	ON POSITION
1 and 2	Operating Mode	Single Chip Mode	Expanded Wide Mode
3	Port H pull device	Port H Pull-Down	Port H Pull-Up
4	Port G pull device	Port G Pull-Down	Port G Pull-Up
5	Oscillator Select	Ext. 16Mhz Clock	User applied crystal

PM12DG128 / PM12DT128

Default: 1 and 2 ON, 3 OFF

MODE SWITCH	SELECTION OPTION	OFF POSITION	ON POSITION
1 and 2	Operating Mode	Single Chip Mode	Expanded Wide Mode
3	Oscillator Select	Ext. 16Mhz Clock	User applied crystal

PM12DP256

Default: 1 - 3 ON, 4 and 5 OFF

MODE SWITCH	SELECTION OPTION	OFF POSITION	ON POSITION
1 and 2	Operating Mode	Single Chip Mode	Expanded Wide Mode
3	MODC Selection	Normal Modes	Special / Emulation Mode
4	ROMON Select	Internal Flash ON	Internal Flash OFF
5	Oscillator Select	Ext. 16Mhz Clock	User applied crystal

PORTS AND CONNECTORS

PRU PORT

The Port Replacement Unit (PRU) provides simulation of the HC12 bus and control ports A, B, E, and K so expanded memory can be used for single-chip application development. PRU operation is enabled with external memory by CONFIG Switch 1 ON. The PRU also provides control of the external memories and peripherals on the CMD912x board. Care should be taken not to violate PRU operation constraints or user code operation could be rendered inoperable until a RESET is performed.

PRU Ports A, B, E, and K are simulated ports due to the different drive characteristics of the PRU logic device. Following are the characteristic differences between the HC12 and PRU ports:

1) The PRU will drive ports to TTL levels with 24ma of source current. Greater than the HC12.

2) The PRU provides pull-up resistance of 47K ohms to +5V is applied to all PRU ports. The HC12 allows this to be optioned on input ports.

3) The PRU will only drive outputs high to +4V. The PRU port pull-up resistors raise the output level to +5V with minimal drive.

4) PRU port inputs will indicate logic high if not driven due to the pull-up resistors.

The PRU will provide HC12 internal resource memory mapping support with constraints. The user should be cautious to stay within the bounds of the constraints for proper operation of the board. PRU constraints:

1) The HC12 R/W, LSTRB, and ECLK signals must be enabled for correct operation of the PRU.

2) The HC12 MODE register must have the IVIS, EME, and EMK bits enabled for correct PRU operation.

3) The IVIS bit in the HC12 MODE register must be enabled prior to any HC12 internal resource map changes from default locations.

4) HC12 internal Ram block (INITRM register) cannot be moved above $4000 hex.

5) HC12 internal Register block (INITRG register) cannot be moved above $8000 hex and is treated as a 2K byte memory space.

6) HC12 internal EEprom block (INITEE register) is treated as a 4K Byte memory space.

PRU PORT CONNECTOR

+5V	**1**	**2**	+5V
XPB6	**3**	**4**	XPB7
XPB4	**5**	**6**	XPB5
XPB2	**7**	**8**	XPB3
XPB0	**9**	**10**	XPB1
XPA6	**11**	**12**	XPA7
XPA4	**13**	**14**	XPA5
XPA2	**15**	**16**	XPA3
XPA0	**17**	**18**	XPA1
XPE6	**19**	**20**	XPE7
XPE4	**21**	**22**	XPE5
XPE2	**23**	**24**	XPE3
PE0	**25**	**26**	PE1
XPK4	**27**	**28**	XPK7
XPK2	**29**	**30**	XPK5
XPK0	**31**	**32**	XPK3
GND	**33**	**34**	XPK1

Note: PE0 and PE1 are the same signals as the HC12 Port E.

MCU_PORT 1

+5V	**1**	**2**	GND
PP0	**3**	**4**	PP1
PP2	**5**	**6**	PP3
PP4	**7**	**8**	PP5
PP6	**9**	**10**	PP7
PX0	**11**	**12**	PX1
PX2	**13**	**14**	PX3
PX4	**15**	**16**	PX5
PX6	**17**	**18**	PX7
PS6	**19**	**20**	PS7
PS4	**21**	**22**	PS5
PS2	**23**	**24**	PS3
PS0	**25**	**26**	PS1
PT0	**27**	**28**	PT1
PT2	**29**	**30**	PT3
PT4	**31**	**32**	PT5
PT6	**33**	**34**	PT7
PG/PJ0	**35**	**36**	PG/PJ1
PG/PJ2	**37**	**38**	PG/PJ3
PG/PJ4	**39**	**40**	PG/PJ5
PG/PJ6	**41**	**42**	PG/PJ7
PH6	**43**	**44**	PH7
PH4	**45**	**46**	PH5
PH2	**47**	**48**	PH3
PH0	**49**	**50**	PH1

The **MCU_PORT1** provides access to the peripheral features and I/O lines of the HC12. Note:

1) Not all I/O Ports are provided by all HC12 MCUs.

2) The PX0 - PX7 port is the CAN channel port and additional I/O provided by the HC12 devices associated with that port. Some devices designate this port as PCAN, PIB, ect.

3) The PG/PJx ports provide either the HC12 port G or HC12 port J depending on device installed.

MCU_PORT 2

PK0	**1**	**2**	PK1
PK2	**3**	**4**	PK3
PK4	**5**	**6**	PK5
PB0/D0	**7**	**8**	PK7/ECS
PB2/D2	**9**	**10**	PB1/D1
PB4/D4	**11**	**12**	PB3/D3
PB6/D6	**13**	**14**	PB5/D5
PE0/XIRQ*	**15**	**16**	PB7/D7
PE6/MODB	**17**	**18**	PE3/LSTRB*
A14	**19**	**20**	PE5/MODA
A15	**21**	**22**	PE7
A16	**23**	**24**	A18
A17	**25**	**26**	A19

The **MCU_PORT 2** provides access to the Expanded Bus and I/O lines of the HC12. Note:

1) Not all I/O Ports are provided by all HC12 MCUs.

2) The A14 - A19 address signals are provided by the PRU. The A16 - A19 signals are derived from the HC12 PK0 - PK5 signals when emulating internal flash paging operation.

BUS_PORT

GND	**1**	**2**	D11
D10	**3**	**4**	D12
D9	**5**	**6**	D13
D8	**7**	**8**	D14
A0	**9**	**10**	D15
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
/ WE	**25**	**26**	CS0
CS1	**27**	**28**	CS2
CS3	**29**	**30**	CS4
CS5	**31**	**32**	IRQ
+5V	**33**	**34**	/P-SEL
/RW	**35**	**36**	CS6
E	**37**	**38**	CS7
GND	**39**	**40**	/ RESET

The **BUS_PORT** supports off-board memory devices.

D8 - D15 High Byte Data Bus in Wide Expanded Mode and Peripheral 8 bit data bus. Port A in Single Chip Mode.

A0 – A13 Memory Addresses 0 to 13.

/OE Memory Output Enable signal, Active Low. Valid with ECLK and R/W high.

CS0 – CS7 Peripheral chip selects, 16 bytes each, see memory maps for location, 8 bit access (narrow bus).

/WE Memory Write Enable signal, Active Low. Valid with ECLK high and R/W low.

IRQ HC12 IRQ (PE1) Interrupt Input.

/RW HC12 Read/Write (PE2) control signal.

E HC12 ECLK (PE4) bus clock signal. Stretch should be enabled in software.

/P-SEL Selects Peripheral area, register following space, 8 bits wide.

/RESET HC12 active low RESET signal.

ANALOG PORT

PAD0	**1 2**	PAD8
PAD1	**3 4**	PAD9
PAD2	**5 6**	PAD10
PAD3	**7 8**	PAD11
PAD4	**9 10**	PAD12
PAD5	**11 12**	PAD13
PAD6	**13 14**	PAD14
PAD7	**15 16**	PAD15
VRL0	**17 18**	VRH0
VRL1	**19 20**	VRH1

The **ANALOG** port provides access to the Port AD0 and Port AD1 Analog-to-Digital input lines.

PAD0 – PAD7 HC12 Port AD0-15 is an input port or A/D Converter inputs.

VRH / VRL HC12 A/D Converter Reference Pins. See A/D Reference Section. To provide an external reference voltage, R3,4,10 and 32 may need to be removed. See schematic.

COM1

	1	
TXD0	**2**	**6**
RXD0	**3**	**7**
	4	**8**
GND	**5**	**9**

The **COM-1** port has a Female DB9 connector that interfaces to the HC12 internal SCI0 serial port. It uses a simple 2 wire asynchronous serial interface and is translated to RS232 signaling levels.

COM2

	1	**2**
TXD1	**3**	**4**
RXD1	**5**	**6**
	7	**8**
GND	**9**	**10**

The **COM-2** has a Female DB9 connector that interfaces to the HC12 internal SCI1 serial port. It uses a simple 2 wire asynchronous serial interface and is translated to RS232 signaling levels.
JP1 may be used to reverse RS232 RX and TX signals to the COM2 connector.

NOTE:

1) COM1 and 2 connector Pins 1, 4, and 6 are connected for default handshake standards.

2) COM1 and 2 connector Pins 7 and 8 are connected for default handshake standards.

3) Handshake pins are provided access pads behind the COM connectors for user application and can be easily isolated from each other on the bottom of the CMD912x board.

4) SCI0:PS0/RXD0 and SCI1:PS2/RXD1 signals can be isolated from the RS232 transceiver by turning COM Switch positions 1 and 2 OFF respectfully.

5) SCI0:PS1/TXD0 and SCI1:PS3/TXD1 signals can be isolated from the RS232 transceiver by removing resistors R10 and R11 respectfully from the bottom of the CMD912x board.

CAN1 - 4 PORTS

These ports provide the CAN Bus input and output. Each port has a CAN Transceiver (Philips PCA82C250) capable of up to 1M Baud data rate. Not all HC12 devices support all of the channels, refer to the device data for capability. Each transceiver receive output has a COM Switch position associated with it. For proper operation the COM Switch should be turned on for each CAN channel that is used for CAN communication. The switch provides isolation for the HC12 I/O port if the CAN operation is not supported by the device or desired by the user. All HC12 I/O ports are available at the Port Headers for other I/O applications.

CAN 1 - 4 Port Connections

1	GND	The CAN1-4 connector provides an interface to the MSCAN12
2	CAN-H	channels 0 - 3 on the microcontroller.
3	CAN-L	
4	+5V	

CAN BUS TRANSMIT ENABLE

Each CAN port transceiver transmit driver is enabled for maximum drive and minimum slew rate by default. The drive and slew rate may be adjusted by changing the value of RC10, RC20, RC30, or RC40 for each CAN 1 - 4 port respectfully (see PCA82C250 data sheet for more information). These 1206 size SMT resistors are located on the bottom of the CMD912x board near the respective CAN port.

CAN Bus transceiver transmit enable control can be applied to each CAN 1 - 4 Port by the RS1 - 4 tie pads respectfully. The user should select an available HC12 I/O port to perform the transmit enable function and connect it from the MCU_PORTx pin to RSx pad as required. **The RC10, RC20, RC30, or RC40 resistors must be removed from the respective CAN port to apply transmit enable control.** The transmit enable signal to the CAN transceivers is active logic low.

CAN BUS TERMINATION

Each CAN port has a set of 1206 SMT size termination resistors on the bottom of the CMD912x board that are not installed at the factory. The termination resistors provide optional bias and termination impedance for the CAN bus connected to the CAN 1 - 4 ports. Type of wire media, data rate, length of wire, and number of CAN bus nodes can all effect the requirement or value of the termination for the CAN bus. User should refer to particular application for termination requirements.

CAN-H Bias Resistors: RC11, RC21, RC31, and RC41 provide bias to ground potential for CAN 1- 4 ports respectfully.

CAN-L Bias Resistors: RC13, RC23, RC33, and RC43 provide bias to +5V potential for CAN 1- 4 ports respectfully.

CAN Termination Resistors: RC12, RC22, RC32, and RC42 provide termination between CAN-H and CAN-L signals for CAN 1- 4 ports respectfully.

LCD_PORT

The LCD_PORT interface is connected to the data bus and memory mapped into the Register Following memory area of the HC12. Note that the DP256 does not support the LCD Port due to no Register following area is available. Refer to the PM12xxx board memory map for LCd Port address location. For the standard display, the base address of the LCD Port is the Command register and the Base+1 address is the display Data register.

The interface supports all OPTREX™ DMC series displays in 8 bit bus mode with up to 80 characters and provides the most common pinout for a dual row rear mounted display connector. Power, ground, and Vee are also available at this connector.

+5V	2	1	GND
A0	4	3	LCD-Vee
LCD1	6	5	/RW
D9	8	7	D8
D11	10	9	D10
D13	12	11	D12
D15	14	13	D14

Command Register: LCD PORT Base+0
Data Register: LCD PORT Base +1

LCD-Vee is supplied by U16 and is adjusted by the CONTRAST Potentiometer (adjustable resistor).

See the file **KLCD12Dx.ASM** for an example program using this LCD connector.

J3

LCD3	2	1	LCD2
	4	3	LCD4

Additional lines can be used as enables for larger character panels and are mapped as:

LCD2 = Base +$4 & $5   LCD4 = Base + $C & $D
LCD3 = Base +$8 & $9

Note: These selects can also be used for peripheral controls.

KEYPAD

1	PH0
2	PH1
3	PH2
4	PH3
5	PH4
6	PH5
7	PH6
8	PH7

The KEYPAD connector is a passive 8-pin connector that can be used to connect a 4 x 4 matrix (16 key) keypad device. The connector is mapped to HC12 I/O port H. This interface is implemented as a software keyscan. Pins PH0-3 are used as column drivers which are active high outputs. Pins PH4-7 are used for row input and will read high when their row is high.

See the file **KLCD12Dx.ASM** for an example program using this connector.

TB1 and J6 Power

The TB1 and J6 connectors provide power input to the board or if J6 is used for input, TB1 maybe used to source additional circuitry. The J6 power jack accepts a standard 2.0 ~ 2.1mm center barrel plug connector (positive voltage center) to provide the +VIN supply of +7 to +20 VDC @ 300ma minimum (+9VDC nominal). TB1 provides access to the +VIN, GND (power ground), and +5V power supplies. The CMD912x power supply will provide 500ma of +5V for user application. +VIN input power should only be applied by J6 or TB1, not both or a supply conflict may occur and the CMD912x board could be damaged.

BDM PORT

The BDM port on the PM12xxx module is a 6 pin header compatible with the Motorola Background Debug Mode (BDM) Pod. This allows the connection of a background debugger for software development, programming and debugging in real-time without using HC12 I/O resources.

BGND

1	**2**
3	**4**
5	**6**

GND
/RESET
+5V

See the HC12 Technical Reference Manual for complete documentation of the BDM.

A Background Debug Module is available from the manufacturer.

TROUBLESHOOTING

The CMD912X 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 IC devices in sockets are properly seated. Verify the communications setup as described under GETTING STARTED and see the **Tips and Suggestions** sections following for more information.

The most common problems are improperly configured communications parameters, and attempting to use the wrong COM port.

1. Verify that your communications port is working by substituting a known good serial device or by doing a loop back diagnostic.

2. Verify the MODE, CONFIG, and COM switch settings are correct.

3. Verify the power source. You should measure approximately 9 volts between the GND and +VIN connections on the TB1 power connector with the standard power supply provided.

4. If no voltage is found, verify the wall plug connections to 115VAC outlet and the power connector.

5. Verify the logic power source. You should measure +5 volts between the GND and +5V connections on the TB1 power connector. If the +VIN supply is good and this supply is not +5V, immediately disconnect power from the board. Contact support@axman.com by email for instructions and provide board name and problem.

6. Disconnect all external connections to the board except for COM1 to the PC and the wall plug.

7. Make sure that the RESET line is not being held low.
    Check for this by measuring the RESET pin on P4 for +5V.

8. Verify the presence of a 16MHz square wave at the EXTAL pin or 8MHz E clock signal if possible.

9. Contact support@axman.com by email for further assistance. Provide board name and describe problem.

Tips and Suggestions

Following are a number of tips, suggestions and answers to common questions that will solve many problems users have with the CMD912X development system. You can download the latest software from the Support section of our web page at:

www.axman.com

Utilities

- If you're trying to program memory or start the utilities, make sure all jumpers and CONFIG SWITCH settings are correct.

- 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. The CONFIG switch changes the memory map.

- If debugging under Mon12, make sure you're not over-writing RAM used by it.

- 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.

Code Execution

- Make sure the CONFIG SWITCH is set for the proper mode.

- CONFIG switch 3 must be ON to access the external bus (LCD display, etc) even if executing code from Internal Flash memory.

- Under Mon12, breakpoints may not be acknowledged if you use the CALL command. You should use one of the GO command instead.

- Check the HC12 reset vector located at FFFE - FFFF. These 2 bytes contain the address where execution will begin when the unit is powered on.

- When running your code stand-alone, you must initialize ALL peripherals used by the micro, including the Stack, Serial Port, Reset and Interrupt vectors etc.

- You must either reset the COP watchdog timer in the main loop of your code or disable it when not running under Mon12 or BDM mode. The micro enables this by default and if you don't handle it your code will reset every couple of ms.

TABLES

TABLE 1. LCD Command Codes

Command codes are used for LCD setup and control of character and cursor position. All command codes are written to LCD panel address $B5F0. The BUSY flag (bit 7) should be tested before any command updates to verify that any previous command is completed. A read of the command address $B5F0 will return the BUSY flag status and the current display character location address.

Command	Code	Delay
Clear Display, Cursor to Home	$01	1.65ms
Cursor to Home	$02	1.65ms
Entry Mode:		
Cursor Decrement, Shift off	$04	40us
Cursor Decrement, Shift on	$05	40us
Cursor Increment, Shift off	$06	40us
Cursor Increment, Shift on	$07	40us
Display Control:		
Display, Cursor, and Cursor Blink off	$08	40us
Display on, Cursor and Cursor Blink off	$0C	40us
Display and Cursor on, Cursor Blink off	$0E	40us
Display, Cursor, and Cursor Blink on	$0F	40us
Cursor / Display Shift: (nondestructive move)		
Cursor shift left	$10	40us
Cursor shift right	$14	40us
Display shift left	$18	40us
Display shift right	$1C	40us
Display Function (default 2x40 size)	$3C	40us
Character Generator Ram Address set	$40-$7F	40us
Display Ram Address and set cursor location	$80- $FF	40us

TABLE 2. LCD Character Codes

$20	Space	$2D	-	$3A	:	$47	G	$54	T	$61	a	$6E	n	$7B	{
$21	!	$2E	.	$3B	;	$48	H	$55	U	$62	b	$6F	o	$7C		
$22	"	$2F	/	$3C	{	$49	I	$56	V	$63	c	$70	p	$7D	}
$23	#	$30	0	$3D	=	$4A	J	$57	W	$64	d	$71	q	$7E	>
$24	$	$31	1	$3E	}	$4B	K	$58	X	$65	e	$72	r	$7F	<
$25	%	$32	2	$3F	?	$4C	L	$59	Y	$66	f	$73	s		
$26	&	$33	3	$40	Time	$4D	M	$5A	Z	$67	g	$74	t		
$27	'	$34	4	$41	A	$4E	N	$5B	[	$68	h	$75	u		
$28	(	$35	5	$42	B	$4F	O	$5C	Yen	$69	i	$76	v		
$29	)	$36	6	$43	C	$50	P	$5D	]	$6A	j	$77	w		
$2A	*	$37	7	$44	D	$51	Q	$5E	^	$6B	k	$78	x		
$2B	+	$38	8	$45	E	$52	R	$5F	_	$6C	l	$79	y		
$2C	,	$39	9	$46	F	$53	S	$60	`	$6D	m	$7A	z		

TABLE 3. Mon12 Monitor Commands

`BF <StartAddress> <EndAddress> [<data>]`	Fill memory with data
`BR [<Address>]`	Set/Display user breakpoints
`BULK`	Erase entire on-chip EEPROM contents
`CALL [<Address>]`	Call user subroutine at <Address>
`G [<Address>]`	Begin/continue execution of user code
`HELP`	Display the Mon12 command summary
`LOAD [P]`	Load S-Records into memory, P = Paged S2
`MD <StartAddress> [<EndAddress>]`	Memory Display Bytes
`MM <StartAddress>`	Modify Memory Bytes
`   <CR>`	Examine/Modify next location
`   </> or <=>`	Examine/Modify same location
`   <^> or <->`	Examine/Modify previous location
`   <.>`	Exit Modify Memory command
`MOVE <StartAddress> <EndAddress> <DestAddress>`	Move a block of memory
`RD`	Display all CPU registers
`RM`	Modify CPU Register Contents
`STOPAT <Address>`	Trace until address
`T  [<count>]`	Trace <count> instructions

1. Mon12 uses internal RAM space from $600 - $7FF for the D60 and $3E00 - $3FFF for the DG128 and DP256. DO NOT use this space in your program if debugging under Mon12.

2. Register space is located starting at $0000 for the DG128 and DP256, D60 is at $800.

3. Mon12 will not trace into interrupts. To trace an interrupt service set a breakpoint in the service routine and then trace.

TABLE 4. I/O Signal Connection Reference

PM12D60	PM12DG/DT128	PM12DP256	J1/P1 J2/P2	CMD912x	OTHER
PA0 / D8	PA0 / D8	PA0 / D8	J2/P2 - 60	BUS PORT - 7	LCD PORT - 7
PA1 / D9	PA1 / D9	PA1 / D9	J2/P2 - 59	BUS PORT - 5	LCD PORT - 8
PA2 / D10	PA2 / D10	PA2 / D10	J2/P2 - 58	BUS PORT - 3	LCD PORT - 9
PA3 / D11	PA3 / D11	PA3 / D11	J2/P2 - 57	BUS PORT - 2	LCD PORT - 10
PA4 / D12	PA4 / D12	PA4 / D12	J2/P2 - 56	BUS PORT - 4	LCD PORT - 11
PA5 / D13	PA5 / D13	PA5 / D13	J2/P2 - 55	BUS PORT - 6	LCD PORT - 12
PA6 / D14	PA6 / D14	PA6 / D14	J2/P2 - 54	BUS PORT - 8	LCD PORT - 13
PA7 / D15	PA7 / D15	PA7 / D15	J2/P2 - 53	BUS PORT - 10	LCD PORT - 14
PB0 / D0	PB0 / D0	PB0 / D0	J1/P1 - 30	MCU PORT2 - 7	
PB1 / D1	PB1 / D1	PB1 / D1	J1/P1 - 29	MCU PORT2 - 10	
PB2 / D2	PB2 / D2	PB2 / D2	J1/P1 - 32	MCU PORT2 - 9	
PB3 / D3	PB3 / D3	PB3 / D3	J1/P1 - 31	MCU PORT2 - 12	
PB4 / D4	PB4 / D4	PB4 / D4	J1/P1 - 34	MCU PORT2 - 11	
PB5 / D5	PB5 / D5	PB5 / D5	J1/P1 - 33	MCU PORT2 - 14	
PB6 / D6	PB6 / D6	PB6 / D6	J1/P1 - 36	MCU PORT2 - 13	
PB7 / D7	PB7 / D7	PB7 / D7	J1/P1 - 35	MCU PORT2 - 16	
PE0 / XIRQ*	PE0 / XIRQ*	PE0 / XIRQ*	J1/P1 - 60	MCU PORT2 - 15	PRU PORT - 25
PE1 / IRQ*	PE1 / IRQ*	PE1 / IRQ*	J1/P1 - 59	BUS PORT - 32	PRU PORT - 26
PE2 / RW	PE2 / RW	PE2 / RW	J1/P1 - 58	BUS PORT - 35	LCD PORT - 5
PE3 / LSTRB*	PE3 / LSTRB*	PE3 / LSTRB*	J1/P1 - 57	MCU PORT2 - 18	
PE4 / ECLK	PE4 / ECLK	PE4 / ECLK	J1/P1 - 44	BUS PORT - 37	
PE5 / MODA	PE5 / MODA	PE5 / MODA	J1/P1 - 43	MCU PORT2 - 20	
PE6 / MODB	PE6 / MODB	PE6 / MODB	J1/P1 - 42	MCU PORT2 - 17	
PE7 / DBE	PE7 / DBE	PE7 /	J1/P1 - 41	MCU PORT2 - 22	
PG0	PJ0	PJ0	J1/P1 - 26	MCU PORT1 - 35	
PG1	PJ1	PJ1	J1/P1 - 25	MCU PORT1 - 36	
PG2	PJ2		J1/P1 - 24	MCU PORT1 - 37	
PG3	PJ3		J1/P1 - 23	MCU PORT1 - 38	
PG4	PJ4		J1/P1 - 14	MCU PORT1 - 39	
PG5	PJ5		J1/P1 - 13	MCU PORT1 - 40	
PG6	PJ6	PJ6	J1/P1 - 12	MCU PORT1 - 41	
PG7	PJ7	PJ7	J1/P1 - 11	MCU PORT1 - 42	
PH0	PH0	PH0	J1/P1 - 56	MCU PORT1 - 49	KEYPAD - 1
PH1	PH1	PH1	J1/P1 - 55	MCU PORT1 - 50	KEYPAD - 2
PH2	PH2	PH2	J1/P1 - 54	MCU PORT1 - 47	KEYPAD - 3
PH3	PH3	PH3	J1/P1 - 53	MCU PORT1 - 48	KEYPAD - 4
PH4	PH4	PH4	J1/P1 - 40	MCU PORT1 - 45	KEYPAD - 5
PH5	PH5	PH5	J1/P1 - 39	MCU PORT1 - 46	KEYPAD - 6
PH6	PH6	PH6	J1/P1 - 38	MCU PORT1 - 43	KEYPAD - 7
PH7	PH7	PH7	J1/P1 - 37	MCU PORT1 - 44	KEYPAD - 8
	PK0	PK0	J1/P1 - 6	MCU PORT2 - 1	PRU
	PK1	PK1	J1/P1 - 5	MCU PORT2 - 2	PRU
	PK2	PK2	J1/P1 - 4	MCU PORT2 - 3	PRU
	PK3	PK3	J1/P1 - 3	MCU PORT2 - 4	PRU
		PK4	J1/P1 - 22	MCU PORT2 - 5	PRU
		PK5	J1/P1 - 21	MCU PORT2 - 6	PRU
	PK7/ECS*	PK7/ECS*	J2/P2 - 10	MCU PORT2 - 8	PRU
PS0/RXD0	PS0/RXD0	PS0/RXD0	J2/P2 - 30	MCU PORT1 - 25	COM 1
PS1/TXD0	PS1/TXD0	PS1/TXD0	J2/P2 - 29	MCU PORT1 - 26	COM 1
PS2/RXD1	PS2/RXD1	PS2/RXD1	J2/P2 - 28	MCU PORT1 - 23	COM 2
PS3/TXD1	PS3/TXD1	PS3/TXD1	J2/P2 - 27	MCU PORT1 - 24	COM 2

PM12D60	PM12DG/DT128	PM12DP256	J1/P1 J2/P2	CMD912x	OTHER
PS4/MIOS	PS4/MIOS	PS4/MIOS	J2/P2 - 26	MCU PORT1 - 21	
PS5/MOIS	PS5/MOIS	PS5/MOIS	J2/P2 - 25	MCU PORT1 - 22	
PS6/SCK	PS6/SCK	PS6/SCK	J2/P2 - 24	MCU PORT1 - 19	
PS7/SS*	PS7/SS*	PS7/SS*	J2/P2 - 23	MCU PORT1 - 20	
PT0	PT0	PT0	J1/P1 - 8	MCU PORT1 - 27	
PT1	PT1	PT1	J1/P1 - 7	MCU PORT1 - 28	
PT2	PT2	PT2	J1/P1 - 10	MCU PORT1 - 29	
PT3	PT3	PT3	J1/P1 - 9	MCU PORT1 - 30	
PT4	PT4	PT4	J1/P1 - 18	MCU PORT1 - 31	
PT5	PT5	PT5	J1/P1 - 17	MCU PORT1 - 32	
PT6	PT6	PT6	J1/P1 - 20	MCU PORT1 - 33	
PT7	PT7	PT7	J1/P1 - 19	MCU PORT1 - 34	
RXCAN0	RXCAN0	PM0/RXCAN0	J2/P2 - 14	MCU PORT1 - 11	CAN 1
TXCAN0	TXCAN0	PM1/TXCAN0	J2/P2 - 13	MCU PORT1 - 12	CAN 1
PCAN2	RXCAN1	PM2/RXCAN1	J2/P2 - 16	MCU PORT1 - 13	CAN 2
PCAN3	TXCAN1	PM3/TXCAN1	J2/P2 - 15	MCU PORT1 - 14	CAN 2
PCAN4	PIB4	PM4/RXCAN2	J2/P2 - 18	MCU PORT1 - 15	CAN 3
PCAN5	PIB5	PM5/TXCAN2	J2/P2 - 17	MCU PORT1 - 16	CAN 3
PCAN6	PIB6	PM6/RXCAN3	J2/P2 - 20	MCU PORT1 - 17	CAN 4
PCAN7	PIB7	PM7/TXCAN3	J2/P2 - 19	MCU PORT1 - 18	CAN 4
PP0/PWM0	PP0/PWM0	PP0/PWM0	J2/P2 - 2	MCU PORT1 - 3	
PP1/PWM1	PP1/PWM1	PP1/PWM1	J2/P2 - 1	MCU PORT1 - 4	
PP2/PWM2	PP2/PWM2	PP2/PWM2	J2/P2 - 4	MCU PORT1 - 5	
PP3/PWM3	PP3/PWM3	PP3/PWM3	J2/P2 - 3	MCU PORT1 - 6	
		PP4/PWM4	J2/P2 - 6	MCU PORT1 - 7	
		PP5/PWM5	J2/P2 - 5	MCU PORT1 - 8	
		PP6/PWM6	J2/P2 - 8	MCU PORT1 - 9	
		PP7/PWM7	J2/P2 - 7	MCU PORT1 - 10	
BGND	BGND	BGND / MODC	J1/P1 - 27		BDM PORT
RESET*	RESET*	RESET*	J1/P1 - 47	BUS PORT - 40	BDM PORT
PAD0 / AN0	PAD0 / AN0	PAD0 / AN0	J2/P2 - 47	ANALOG PORT - 1	
PAD1 / AN1	PAD1 / AN1	PAD1 / AN1	J2/P2 - 45	ANALOG PORT - 3	
PAD2 / AN2	PAD2 / AN2	PAD2 / AN2	J2/P2 - 43	ANALOG PORT - 5	
PAD3 / AN3	PAD3 / AN3	PAD3 / AN3	J2/P2 - 41	ANALOG PORT - 7	
PAD4 / AN4	PAD4 / AN4	PAD4 / AN4	J2/P2 - 39	ANALOG PORT - 9	
PAD5 / AN5	PAD5 / AN5	PAD5 / AN5	J2/P2 - 37	ANALOG PORT - 11	
PAD6 / AN6	PAD6 / AN6	PAD6 / AN6	J2/P2 - 35	ANALOG PORT - 13	
PAD7 / AN7	PAD7 / AN7	PAD7 / AN7	J2/P2 - 33	ANALOG PORT - 15	
PAD10 / AN10	PAD10 / AN10	PAD8 / AN8	J2/P2 - 48	ANALOG PORT - 2	
PAD11 / AN11	PAD11 / AN11	PAD9 / AN9	J2/P2 - 46	ANALOG PORT - 4	
PAD12 / AN12	PAD12 / AN12	PAD10 / AN10	J2/P2 - 44	ANALOG PORT - 6	
PAD13 / AN13	PAD13 / AN13	PAD11 / AN11	J2/P2 - 42	ANALOG PORT - 8	
PAD14 / AN14	PAD14 / AN14	PAD12 / AN12	J2/P2 - 40	ANALOG PORT - 10	
PAD15 / AN15	PAD15 / AN15	PAD13 / AN13	J2/P2 - 38	ANALOG PORT - 12	
PAD16 / AN16	PAD16 / AN16	PAD14 / AN14	J2/P2 - 36	ANALOG PORT - 14	
PAD17 / AN17	PAD17 / AN17	PAD15 / AN15	J2/P2 - 34	ANALOG PORT - 16	
VRH0	VRH0		J2/P2 - 50	ANALOG PORT - 18	
VRL0	VRL0		J2/P2 - 49	ANALOG PORT - 17	
VRH1	VRH1	VRH1	J2/P2 - 32	ANALOG PORT - 20	
VRL1	VRL1	VRL1	J2/P2 - 31	ANALOG PORT - 19	

Free Manuals Download Website

http://myh66.com

http://usermanuals.us

http://www.somanuals.com

http://www.4manuals.cc

http://www.manual-lib.com

http://www.404manual.com

http://www.luxmanual.com

http://aubethermostatmanual.com

Golf course search by state

http://golfingnear.com

Email search by domain

http://emailbydomain.com

Auto manuals search

http://auto.somanuals.com

TV manuals search

http://tv.somanuals.com