MiniDRAGON+

Compact MC9S12DP256 Development Board

Getting Started Manual

Version 1.30 for rev. D board

Table of Contents

INTRODUCTION

The MiniDRAGON+ is a low-cost, feature-packed compact development board for the new Motorola MC9S12 microcontroller family. It is compatible with the Motorola 9S12DP256EVB board and other similar development boards on the market today, but it also incorporates many on-board peripherals that make this board a better value for users.

For engineers, it is a core module suitable for designers who want to rapidly develop and prototype new MC9S12 applications. For students, it is an advanced microcontroller trainer. Use it to build a solid foundation of microcontroller expertise and to create a real word application for a senior design project.

To help users get up and running, the MiniDRAGON+ board comes with many fully debugged, ready-to-run sample programs. The programs are ported from our EVBplus2 68HC11 development board and because the 68HC12 instructions are upward code compatible with the 68HC11 instructions, migrating from your 68HC11 projects into the new HCS12 world could not be any easier.

The state of the art MC9S12DP256 controller is the most powerful chip in the family and it is loaded with hardware features such as:

- 25 MHz bus speed
- 256K flash memory
- 12K SRAM
- 4K EEPROM

On-chip peripherals include:

- Dual SCIs
- Triple SPIs
- I2C
- Five CAN modules
- Dual 10-bit 8 channel ADCs
- Eight PWMs
- Eight 16-bit timers

It can also be used in the evaluation and development of all other family members, such as the A and D series.

The on-board hardware includes:

- BDM-in and BDM-out headers
- Solderless breadboard
- Speaker
- 7-segment LED display
- 2-postion DIP switch
- Two pushbuttons
- Potentiometer
- IR receiver
- RS485 interface
- 1 CAN port
- Dual SCIs
- LCD connector header
- 4X4 Keypad connector header
- Slide switch for Serial Monitor

All I/O lines in the male headers or female receptacles provide an easy access for your experiments.

The package also includes a 7.5V 300mA wall plug-in power supply, CD and a re-tractable 6-foot DB9 cable.

The specification of the AC adapter is:

DC input: 110V
DC output: 7.5V
Current rating: 300mA
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 may need to upgrade your AC adapter to 9V 500mA 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 MiniDRAGON+ software from CD:

The installation is automated by double clicking on the **SETUP.BAT** on the CD. It will create a folder c:\MiniDRAGON\examples and copy all example program files from the CD to c:\MiniDRAGON\examples

If the filename is only shown as **SETUP**, not **SETUP.BA**T, you should change a folder option of the explorer to show file extension. When a file's extension is hiding, sometime it is hard to know what it is. To have your files to be shown with extensions, click on the TOOL tab in explorer menu, then click on folder options, then click on view tab, finally un-check the item named 'Hide extensions for knowing file types'.

The AsmIDE is freeware, but if you would like to use it in the future, we encourage you donate $5.00 to Eric Engler at: http://www.geocities.com/englere_geo/ .

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:\MiniDRAGON, not c:\Windows\desktop or other locations. First, right click the Start button. Then, left click "Explorer". Left click on C:\MiniDRAGON. 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:\MiniDRAGON. 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 program may have a problem to run. Never drag the AsmIDE.exe to the desktop folder.

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

GETTING STARTED

To operate the MiniDRAGON+ board, follow steps1 through 6 below:

Step 1.

Plug the AC adapter into a wall outlet, and plug the DC female plug at the other end into the DC jack on the lower left side of the MiniDRAGON+ board. During power up, the small 7-segment LED should display a diagnostic code of E-4-3 momentarily. The letter E stands for EVB mode, the number 4 stands for single chip mode and the number 3 means that 2 switches (S3, S4) are open. Here is the code explanation:

First letter:	E = EVB mode, J = Jump to EEPROM, P = Pod and b = Bootloader mode.
Second number:	4 = Single chip mode, 6 = Narrow expended mode, 7 = Wide expended mode
Third number:	0 = S3 and S4 closed, 1 = S3 open and S4 closed, 2 = S3 closed and S4 open
	3= S3 and S4 open

If the code is other than E-4-3, check jumper setting on the 6-pin header J8 (MODE B and MODE A), jumper setting on the 3-pin header J9, the position of the slide switch S7 and jumper setting on the S3 and S4.

The default settings are listed on page 16.

Step 2.

Plug the RS232 cable into the R11 handset jack JK1at the upper left side of the MiniDRAGON+ board and plug the DB9 female end of the cable into the COM1 or COM2 port on your PC.

Step 3.

Press the reset button on the MiniDRAGON+ board, and the 7-segment LED should display E-4-3.

Step 4.

To invoke the AsmIDE, right click on the Start button and then left click on “Explorer”. Next, left click on C:\MiniDRAGON 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.

The AsmIDE is simple and very easy to use. You only need to use three commands from the AsmIDE for your 68HC12 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 download an s19 file to the MiniDRAGON+ 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.

In the View->Option->Assembler menu, make sure that the chip family is **68HC12**, not 68HC11. If you would like to use your own assembler, you can replace the as12.exe 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.

If the terminal options are set correctly, you should see the following prompt every time the reset button on the MiniDRAGON+ board is pressed. If you do not see this, the bottom window may be set for message window. Click the terminal button in the bottom window to enable the terminal window display.

```
D-Bug12 v4.0.0b29
Copyright 1996 - 2005 Motorola Semiconductor
For Commands type "Help"
>
```

SOFTWARE DEVELOPMENT

The MC9S12DP256 memory map in single chip mode is as follows:

$0000-$03FF Registers

$0400-$0FFF EEPROM

$1000-$3FFF RAM

$4000-$7FFF 16K fixed Flash

$8000-$BFFF 16K page window

$C000-$FFFF 16K fixed Flash

The steps to run your first sample program are as follows:

In order to run the test program, the jumper on J9 must be placed at right position and the slide switch S7 must be set at right position so the board will be booted in **EVB mode**.

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

2. Click the Build button to assemble code and generate the test.s19 file. This is how you normally generate an s19 file. You can omit this step, because the test.s19 is already on your hard disk.

3. Press the reset button on the board, you will see:

```
D-Bug12 v4.0.0b29
Copyright 1996 - 2005 Motorola Semiconductor
For Commands type "Help"
>
```

4. Type “LOAD” <Enter>.

5. Click the Build button. Select Download option and select the file ‘test.s19’ for downloading.
   Note: If the top window is not loaded with a file, clicking the Build button will prompt you with the “save changes” message. You can ignore that message and click the “No” answer.

6. After download is done, type “G 2000” <Enter> to run the test program.

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 the new program may not work if an interrupt was enabled by a previous program. In all example programs, the user program code locations are at $2000-$3FFF. The user data RAM locations are at $1000-$1FFF. The 64 RAM interrupt vector addresses are at $3E00-$3E7F. You must place your interrupt vectors at $3E00-$3E7F, because real interrupt vector addresses are taken by the bootloader, the bootloader will redirect interrupts to D-Bug12 monitor which will redirect them to the RAM interrupt vector addresses at $3E00-$3E7F.

The 64 RAM interrupt vector addresses (128 bytes of RAM) are assigned by the D-Bug12 monitor to different interrupt sources. The listing of interrupt sources is show on page 12 of the D-Bug12 V4.x reference guide **(DB12RG4.PDF)** on the CD.

The AN1280 was written for OLD HC12 family. If you happen to use printf routine with your old HC12 board you should be aware that I/O utility routines are moved to different addresses in D-Bug12 V4.x.

The new D-Bug12 V4.x is much different and much larger (about 60K) than old D-Bug12 V2.x.
The $C000-$EFFF are just a part of the monitor, In 16-bit S1 record they are $C000-$EFFF. In 24-bit S2 record, they are $FC000-FEFFF (ppage=$3F). Since the ppage register deals with the 16K window $8000-$BFFF the addresses $C000-$FFFF are not affected by the ppage. The other part of the monitor is at C0000-C87FF (16K window $8000-$BFFF when ppage=$30,$31 and $32). See details on page 20 of the app note AN2153 or page 71 of the D-Bug12 v4 reference guide on the CD. The address for the printf is $EE88, see page 79 of the D-Bug12 V4.x reference guide on the CD for addresses of other I/O routines.

The bootloader (**AN2153.PDF)**, the D-Bug12 reference guide (**DB12RG4.PDF**) and the MC9S12DP256 data book **(MC9SDP256.PDF**) are the most important documentation. They can be found on the folder named C:\MiniDRAGON\document after software installation. The HCS12 instruction set, register map and memory map can be found on page 26, 65 and 120 of the data book, respectively.

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 it can generate an s19 file.

Four operating modes:

The MC9S12DP256 on the MiniDRAGON+ board is pre-loaded with bootloader and D-Bug12 monitor firmware and it will operate in 4 different modes depending on the setting of the J9 and the S7. The current mode is indicated by the first letter on the 7-segment LED display during dower up or reset. During power up or reset, the MC9S12DP256 will read the PAD0 and PAD1 to decide which mode to boot up and the first letter of the 7-segment LED will display that mode.

Mode0:
EVB mode: PAD1=0 (SW7 is set at right side), PAD0=0 (jumper on J9 is at right side).
This is the standard debug environment running on the MC9S12DP256 for on-chip RAM or EEPORM based code development. Using an IDE program to view and modify registers and memory locations, set breakpoints, single step through program, assembly and disassembly code as you would in a BUFFALO monitor based Motorola 68HC11 EVB. It gives you 12K RAM and 3K EEPROM to develop and debug your code. You must place your interrupt vectors at $3E00-$3E7F, because real interrupt vector addresses are taken by the bootloader, the bootloader will redirect interrupts to the RAM interrupt vector addresses at $3E00-$3E7F.

After booting up this mode, you would see the following after reset:

```
D-Bug12 v4.0.0b29
Copyright 1996 - 2005 Motorola Semiconductor
For Commands type "Help"
>
```

Typing "HELP" then <Enter> will display a list of available commands.

In this mode, you cannot erase or program on-chip flash memory

Mode1:
Jump-to-EEPROM mode: PAD1=0 (SW7 is set at right side), PAD0=1 (jumper on J9 is at left side).
This mode enables the MC9S12DP256 to jump directly to the internal EEPROM at location $0400 upon reset. This mode makes the MC9S12DP256 a replacement for the old 68HC811E2 microcontroller, but it also gives you 3K EEPROM instead of 2K EEPROM with the 68HC811E2. The bus speed is 8MHz, one half of the crystal frequency by default, the PLL function must be initialized by user's code for a higher bus speed, because the D-Bug12 monitor firmware was bypassed.

Mode2:
POD mode: PAD1=1 (SW7 is set at left side), PAD0=0 (jumper on J9 is at right side).
In this POD mode the D-Bug12 firmware acts as a master to access all target MCU's resource on the target board (another MiniDRAGON+ or DRAGON12 board) via BDM port in a no-intrusive manner. It becomes a BDM that will have all features that a standard BDM has in debugging the target MCU and a programmer for programming the flash memory of the MCU on the target board (another MiniDRAGON+ or DRAGON12 board).

To use the master board as a programmer, you need a 6-pin ribbon cable to connect from the BDM OUT of the master board to the BDM IN of the target board (make sure that the polarity is correct, the pin 1of the BDM headers J4 and J5 is the lower left pin). You don't have to provide the power to both boards, only to provide power to the master board. Both boards should be set at single chip mode for the best result (the MODEB, MODEA are set for 0-0). The master board communicates to a PC COM port while the target board does not need to be connected to a PC COM port.

After booting up this mode, you would see the following after reset:

```
Can't Communicate With Target CPU

1.) Set Target Speed (48000 KHz)
2.) Reset Target
3.) Reattempt Communication
4.) Erase & Unsecure
?
```

You first must set the target speed with the choice 1). After enter the first choice, it will prompt you to enter the target speed. It's the crystal frequency, not the bus speed that is boosted up by the on-chip PLL. After a reset, before the PLL is enabled, the target MC9S12DP256 is running from the crystal frequency, not the PLL frequency. You enter 16000 for the target speed. After the correct speed is entered, the master will try to communicate with the target board. If it's not successful, you enter the choice 2) to reset the target board.

```
Can't Communicate With Target CPU

1.) Set Target Speed (4000 KHz)
2.) Reset Target
3.) Reattempt Communication
4.) Erase & Unsecure
? 1

Enter Target Crystal Frequency (kHz): 16000



Can't Communicate With Target CPU

1.) Set Target Speed (16000 KHz)
2.) Reset Target
3.) Reattempt Communication
4.) Erase & Unsecure
? 2
```

When the communication is established, you will see the following:
(Sometimes you need to reset the boards to make them communicate)

```
D-Bug12 v4.0.0b29
Copyright 1996 - 2005 Motorola Semiconductor
For Commands type "Help"

S>
```

You notice that the debug prompt is “S>” in the POD mode, not just a “>” in the EVB mode. The S> tells that this is the POD mode and the MC9S12DP256 on the target (slave board) is stopped. Sometimes the prompt could be an “R>” that means the target MCU is running. If you see the “R>”, just type “RESET” then <Enter> to reset it and it will come back with the “S>” prompt.

```
R>reset <Enter>
S>
```

In order to program the flash memory, you have to erase it by the FBULK command

```
S>fbulk <Enter>
S>
```

When the prompt “s>” returns, the FBULK command has already erased all of the flash memory contents of the target MC9S12DP256 including the bootloader. If it returns with a message “Flash or EEPROM Failed To Erase” the MC9S12DP256 is defective.

Now we are going to program the bootloader and the D-Bug12 into the flash memory of the target MC9S12DP256.
Before we actually program the flash memory, we must understand there are two different types of s-record file that can be generated by compilers and assemblers.

An s1-record uses a 16-bit starting address field while an s2-record uses a 24-bit starting address field.
An s1-record file looks like this:

```
S123FFA0F64CF650F654F658F65CF660F664F668F66CF670F674F678F67CF680F684F6883D
S123FFC0F68CF690F694F698F69CF6A0F6A4F6A8F6ACF6B0F6B4F6B8F6BCF6C0F6C4F6C81D
S123FFE0F6CCF6D0F6D4F6D8F6DCF6E0F6E4F6E8F6ECF6F0F6F4F6F8F6FCF700F704F00009
S9030000FC
```

An s2-record file looks like this:

```
S2240FEFA0DB70DB66DB5CDB52DB48DB3EDB34DB2ADB20DB16DB0CDB02DAF8DAEEDAE4DADA41
S2240FEFC0DAD0DAC6DABCDAB2DAA8DA9EDA94DA8ADA80DA76DA6CDDD0DA62DA58DA4EDA4494
S2240FEFE0DA02DA0ADA12DA1ADA22DA2ADA32DA3AD9FAD9F2D9AFD98AD9D5EF00EF00EF0039
S9030000FC
```

We are not going to explain the s-record format here. If you would like to know more on the subject, you can review the D-Bug12 reference guide on the CDROM (**BD12RG4.PDF**). It explains the subject in great details. Right now, all you need to know is that an s1-record file must be converted to an s2-record file before using the FLOAD command. The “FLOAD” command in the D-Bug12 is for downloading an s2 record file.

Our MiniDRAGON_boot_16.asm is modified from the Motorola’s BootDP256.asm. We added our modification in the beginning of the original source code and the s record file is generated by the AsmIDE and it’s an s1 record file, we converted it into an s2 record file by using the following commands:

```
Srccvt –m c0000 fffff 32 –of f0000 -o MiniDRAGON_boot_16.s29 MiniDRAGON_boot_16.s19
```

Now we type “FLOAD” <Enter> at the prompt. Click the Build button, select the Download option, and select the file **MiniDRAGON_boot_16.s29** located in the folder named “document”. You should see the following on the terminal window when programming is done (when the prompt “s>” appears):

```
S>fload <Enter>

**************************************************************************
S>
```

Now we are going to download the D-Bug12 monitor into the flash memory. We need to type “FLOAD” <Enter> at the prompt. Click the Build button, select the Download option, and select the file **db12dp256-16.s29** located in the folder named “document”. You should see the following on the terminal window when programming is done (when the prompt “s>” appears):

```
S>fload <Enter>

************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************************
************************************************************
S>
```

With the bootloader and the D-Bug12 programmed in the flash memory, the target board now becomes a true development board. That’s how we program the board before we ship it. Your MiniDRAGON+ board actually becomes a programmer. You can then repeat above steps as many times as you want. Just unplug the 6-pin BDM cable from the target board, and then plug it into a new target board to program its flash memory with these two files. You even don’t have to turn off the power while doing this.

The D-Bug12 monitor is an application program under the bootloader. If you program the D-Bug12 portion of flash memory with your application program, your program will run automatically in EVB mode after power up or reset.

As the same as in the mode1, the bus speed is 8MHz, one half of the crystal frequency by default. The PLL function must be initialized by your code for a higher bus speed, because the D-Bug12 monitor firmware is not in flash memory anymore.

Mode3:
BOOTLOADER mode: PAD1=1 (SW7 is set at left side), PAD0=1 (jumper on J9 is at left side).

This bootloader allows you to erase/program flash and erase EEPROM. It is mainly used to program the D-Bug12 monitor into flash memory or download a user’s fully debugged code into the D-Bug12 portion of flash memory. The latter allows the board to be operated in EVB mode and start your code every time the board is turned on or reset.
When you program your code into the D-Bug12 portion of flash memory, it wipes out the D-Bug12 monitor. You can restore it any time, just as if you were downloading another application program since the bootloader is not erased. You can erase and program the D-Bug12 monitor portion of the flash memory of the MC9S12DP256 on its own board, but not the bootloader portion. **The bootloader can only be erased by a BDM via the BDMin port.**

HCS912DP256 bootloader menu:

a) Erase Flash
b) Program Flash
c) Set Baud Rate
d) Erase EEPROM
?

The option a) will erase the D-Bug12 portion of the flash memory, not the bootloader itself.
The option b) will program the D-Bug12 portion of the flash memory, not the bootloader itself.
The file to be programmed into flash memory must be an s2-record file. If your assembler and compiler generates s1-record files only, you must convert an s1-record file to an s-2 record file before programming flash memory with the bootloader.
The option c) will set a new baud rate.
The option d) will erase all on-chip EEPROM.

To program flash memory with the D-Bug12 monitor:

1. Enter the option a to erase the D-Bug12 portion of flash memory. Wait until the bootloader menu re-appears after the flash memory is erased.
2. Enter the option b, the bootloader will wait for your file. **Do not type any thing on keyboard.**
3. Click the Build button, select the Download option, and select the file **db12dp256-16.s29** located in the folder named "document" for downloading. You should see the following on the screen:
   ****
   ****
   ****
   ****
   ****
4. Bootloader menu appears again after the programming is done.

How to use the MiniDRAGON+ board to develop or test your code? It depends if you use a BDM or not.

If you don't use a BDM, there are 3 approaches:

1. Use on-chip 12K RAM for code development in **EVB mode**.

   You can download your s19 file into the RAM and debug it with the D-Bug12 monitor in this mode. You must place your interrupt vectors at $3E00-$3E7F, because real interrupt vector addresses are taken by the bootloader. The bootloader and the D-Bug12 monitor will redirect interrupts to the RAM interrupt vector addresses at $3E00-$3E7F.

   Because the RAM will lose its contents after power off, you have to load your program every time after power-up. In the beginning of your program, you must initialize the interrupt vectors at $3E00-$3E7F.

2. Use on-chip 3K EEPROM for testing your code in **EVB mode**.

   If your program is small enough to fit into a 3K range, then you can download your code into the EEPROM. In this way, your program can be auto started from $0400 upon reset. You cannot set software breakpoint and single step in the EEPROM in EVB mode, so it makes sense to do development work in the RAM, when your code is completely debugged, then re-assemble or re-compile it at $0400 and download the final s19 file into the EEPROM for the auto start feature.

   Like the RAM-based development, your interrupt vectors are at $3E00-$3E7F. In the beginning of your program you must initialize the interrupt vectors at $3E00-$3E7F.

3. Use on-chip flash for testing your code in **BOOTLOADER mode**.

   In this mode, you download your program directly into the on-chip flash memory. You first erase the D-Bug12 monitor portion of flash memory, and then program that portion of the flash memory by downloading your application program code s19 file. Your program will replace the D-Bug12 monitor in the flash memory. The bootloader portion of the flash memory remains intact. To run your code, set J9 and S7 for EVB mode, then press the reset button. It usually runs the D-Bug12 monitor. Now, it will run your program. The flash memory is non-volatile like the EEPROM. Your code will run every time the board is turned on or reset.

   The bootloader redirects interrupts to $EF80-$EFFF. The D-BUG12 is not present and the interrupt vectors of your program are at $EF80-$EFFF. The addresses $EFFE and $EFFF contains starting address of program.

   In order to program the MC9S12DP256 flash memory, you must program an even number of bytes and begin on an even address boundary for each s-record. If any one s-record in the file contains an odd number of bytes or begins with an odd address, the flash memory cannot be programmed. If your assembler or compiler cannot generate the even format, you must use the Motorola s-record conversion utility **sreccvt.exe** to convert your odd format to the even format by using the following command line:

   Srccvt –m c0000 fffff 32 –of f0000 –o test.s29 test.s19

   It will create a new file named test.s29 that has even format and can be programmed into the flash memory. For your convenience, the sreccvt.exe is included in CDROM\document\Sreccvt-GUI.

If you ever use a BDM device with the MiniDRAGON+ board, there is a different method for debugging. The BDM stands for background debug mode. With a BDM device, you don't need the D-Bug12 monitor and the bootloader physically reside in flash memory of the MC9S12DP256. So, a BDM device allows the use of all on-chip 256K flash memory of the MC9S12DP256 for your program code, including the portion of the bootloader. You also can use the real interrupt vector addresses in your program. There aren't anymore pseudo RAM vector addresses.

When a BDM device is connected to the BDM IN of the MiniDRAGON+ board, it can interrogate the state of the MC9S12DP256 microcontroller in real-time without stopping the program. The BDM device also permits single stepping and setting hardware breakpoints in flash memory, so you can debug your code in flash memory.

This is a new product and you will get some software upgrade in the future. If you would like to get a free upgrade, you should send us an email once in a while to check out the status of our software

The web is the best source for getting more information about the HCS12. The Motorola web site has all documents and application notes that you need. The HC12 user group http://groups.yahoo.com/group/68HC12/ is a good place to ask a question and get a prompt answer from many HC12 gurus.

You also can visit our web site at: http://www.evbplus.com/these_sites.html to get links to different university web sites that offer course materials and lab assignments for the MiniDRAGON+ board.

All HCS12 boards on the market today are commodity products and are basically the same, because they are all using the Motorola bootloader and D-Bug12 monitor. If you are going to use a BDM to debug a HCS12 board, all boards will be exactly same because the BDM directly communicates with the MC9S12DP256 MCU. The information on our manual can apply to the boards from other manufacturers, and vice versa.

ON-BOARD HARDWARE

PH0-PH6 of port H are connected to a 7-segment LED display. The Decimal point of the display is connected to IR detector's output. When IR detector detects a 38 KHz IR light, the output of the detector will go low, the DP of the LED display will be lit.

The VR1 is connected to the AN07 input of the ADC port via J13 (SMD pads on the solder side), but the trace between the pads can be cut if AN07 must be used by target circuits.

U4, SN75176 (SMD chip on the solder side), converts the TTL signal from the SCI1 to RS485 differential signals and vice versa. The J1 is the RS485 connector.

PJ0 (pin 22) of the MC9S12DP256 is connected to control the direction of RS485 communication via J15 (SMD pads on the solder side), but the trace between the pads can be cut if PJ0 must be used by target circuits. If PJ0=0, the RS485 port, U9 DS75176, is set as the receiver port; If PJ0=1, the RS485 port, U9 DS75176, is set as the transmitter port.

PJ0=0 RS485 receiver port
PJ0=1 RS485 transmitter port

The speaker is connected to PT5 (pin 16) of the MC9S12DP256 via J15 (SMD pads on the solder side), but the trace between the pads can be cut if PT5 must be used by target circuits.

Two pushbuttons (S1 and S2) are connected to AN04 and AN03. The 2-pos DIP switch equivalent switches (S3 and S4) are connected to AN05 and AN06.
S1-----→ AN04
S2-----→ AN03
S3-----→ AN05
S4-----→ AN06
That's the way the PC board was laid out, the AN03 and AN04 should be swapped. Unfortunately we have to keep the way it is now.

The MiniDRAGON+ board was made before the serial monitor was released for the Code Warrior IDE, so it did not have an on-board RUN/LOAD switch required by the serial monitor. We added the slide switch S7 that is connected to the PAD01 on the rev. D board for working with the serial monitor and the Code Warrior IDE. In serial monitor mode, it selects RUN or LOAD mode. In D-Bug12 monitor mode, when the jumper on J9 is placed at right side, the S7 selects EVB mode or POD mode. When the jumper on J9 is placed at left side, the S7 selects JEE (Jump-to-EE) mode or BL (Bootloader) mode.

The crystal frequency is 16 MHz and usually it will result an 8 MHz bus speed, but on this board the MC9S12DP256's internal PLL brings up the bus speed to 24 MHz.

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

How to use port M:

Port M is an 8-bit bi-directional port. It's used for driving the LCD display module. If the port is not used for the LCD display, it can be used as a regular port M.

The pinouts of J21 are as follows:

Pin 1	GND	
Pin 2	VCC (5V)	
Pin 3	Via a 220 Ohm resistor to GND	
Pin 4	PM3	RS pin for LCD module
Pin 5	GND	
Pin 6	PM2	EN pin for LCD module
Pin 7	Not used	
Pin 8	Not used	
Pin 9	Not used	
Pin 10	Mot used	
Pin 11	PM4	DB4 pin for LCD module
Pin 12	PM5	DB5 pin for LCD module
Pin 13	PM6	DB6 pin for LCD module
Pin 14	PM7	DB7 pin for LCD module

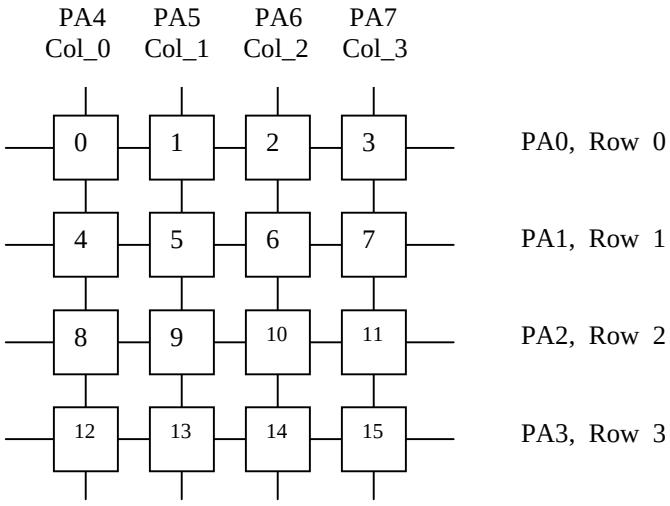
WARNING: Beware of pinouts of LCD modules. The pin 1 and pin 2 of some new modules are swapped. If you connect a wrong polarity LCD to the MiniDRAGON+ board, it could be damaged. So check the VCC and VSS polarity first before connecting your LCD module to the MiniDRAGON+ board.

How to use the keypad interface via port A:

Port A is an 8-bit bi-directional port. Its primary usage is for a 4X4 keypad interface. If the port is not used for the keypad, it can be used as a general-purpose I/O.
The PA0-PA7 has a weak internal pull-up resistor in each line.

There are two different keypads have been used with Wytec development boards.

The following signal definitions only apply to Grayhill 4X4 keypad.
Grayhill 4X4 keypad connections:



The pinouts of J22 for Grayhill 4X4 keypad is as follows:

Pin 1	GND	
Pin 2	PA0	connects ROW0 of the keypad
Pin 3	PA1	connects ROW1 of the keypad
Pin 4	PA2	connects ROW2 of the keypad
Pin 5	PA3	connects ROW3 of the keypad
Pin 6	PA4	connects COL0 of the keypad
Pin 7	PA5	connects COL1 of the keypad
Pin 8	PA6	connects COL2 of the keypad
Pin 9	PA7	connects COL3 of the keypad
Pin10	VCC	

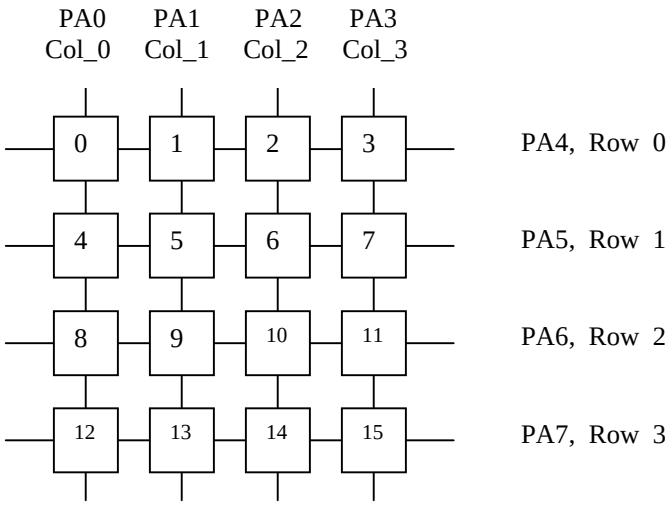
An 8-pin male header is installed on pin 2 through pin 8. Pins 1 and 10 are not installed.

Keypad interface

PA0 connects ROW0 of the keypad via pin 1 of the 8-pin keypad header
PA1 connects ROW1 of the keypad via pin 2 of the 8-pin keypad header
PA2 connects ROW2 of the keypad via pin 3 of the 8-pin keypad header
PA3 connects ROW3 of the keypad via pin 4 of the 8-pin keypad header

PA4 connects COL0 of the keypad via pin 5 of the 8-pin keypad header
PA5 connects COL1 of the keypad via pin 6 of the 8-pin keypad header
PA6 connects COL2 of the keypad via pin 7 of the 8-pin keypad header
PA7 connects COL3 of the keypad via pin 8 of the 8-pin keypad header

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



The pinouts of J22 for Wytec 4X4 membrane keypad is as follows:

Pin 1	GND	
Pin 2	PA0	connects COL0 of the keypad
Pin 3	PA1	connects COL1 of the keypad
Pin 4	PA2	connects COL2 of the keypad
Pin 5	PA3	connects COL3 of the keypad
Pin 6	PA4	connects ROW0 of the keypad
Pin 7	PA5	connects ROW1 of the keypad
Pin 8	PA6	connects ROW2 of the keypad
Pin 9	PA7	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 installed.

Keypad interface

PA0 connects COL0 of the keypad via pin 1 of the 8-pin keypad header
PA1 connects COL1 of the keypad via pin 2 of the 8-pin keypad header
PA2 connects COL2 of the keypad via pin 3 of the 8-pin keypad header
PA3 connects COL3 of the keypad via pin 4 of the 8-pin keypad header
PA4 connects ROW0 of the keypad via pin 5 of the 8-pin keypad header
PA5 connects ROW1 of the keypad via pin 6 of the 8-pin keypad header
PA6 connects ROW2 of the keypad via pin 7 of the 8-pin keypad header
PA7 connects ROW3 of the keypad via pin 8 of the 8-pin keypad header

The keypad scan routine sets PA3 low and PA0, PA1, and PA2 high. The routine then tests PA4-PA7.
If no key is down, PA4-PA7 remain high.
If PA7 = low, the key 15 is down.
If PA6 = low, the key 14 is down.
If PA5 = low, the key 13 is down.
If PA4 = low, the key 12 is down.

The keypad scan routine then sets PA2 low and PA0, PA1, and PA3 high. The routine then tests PA4-PA7
If no key is down, PA4-PA7 remain high.
If PA7 = low, the key 11 is down.
If PA6 = low, the key 10 is down.
If PA5 = low, the key 9 is down.
If PA4 = low, the key 8 is down.

The keypad scan routine then sets PA1 low and PA0, PA2, and PA3 high. The routine then tests PA4-PA7
If no key is down, PA4-PA7 remain high.
If PA7 = low, the key 7 is down.
If PA6 = low, the key 6 is down.
If PA5 = low, the key 5 is down.
If PA4 = low, the key 4 is down.

The keypad scan routine then sets PA0 low and PA1, PA2, and PA3 high. The routine then tests PA4-PA7
If no key is down, PA4-PA7 remain high.
If PA7 = low, the key 3 is down.
If PA6 = low, the key 2 is down.
If PA5 = low, the key 1 is down.
If PA4 = low, the key 0 is down.

All on-board jumpers:

J1 RS485 interface
J2 SCI1, 2nd SCI interface
J3 CAN0 interface
J4 BDM input, pin 1 is the lower left pin
J5 BDM output, pin 1 is the lower left pin
J6 External 9V DC input, it paralleled with the DC jack JK2. **Do not place a jumper on it.**
J7 Low Voltage Inhibit, remove this jumper for low voltage applications
J8 Mode B and Mode A selectors, it defaults at 0-0
J9 PAD0, it combines with switch S7 for selecting one of four operating modes (EVB, Jump-to-EEPROM, POD and BOOTLOADER).
J10 Enables 7-segment LED display. If the 7-segment LED display is not needed in an application, remove this jumper to turn off the display and save power.
J11 It's solder bridged (SMD pads on the solder side) to power the board by on-board 5V regulator. If an external 5V DC is used, the solder bridge must be removed to cut off the power from the on-board 5V regulator.
J12 It connects RS of CAN0 (U5) to VSS via SMD pads on the solder side.
J13 connects VR1 trimmer pot to the AN07 of the ADC via SMD pads on the solder side, but the trace between the pads can be cut if AN07 must be used by target circuits.

J14 MC9S12DP256 SCI1receiver source selector
When no solder bridge on all 3 pads, the SCI1 PS2 receives signal from your target system via pin 91 of the header H4
. When a solder bridge is on the left (the left pad and the middle pad are shorted), the SCI1 PS2 receives signal from J2, for RS232 input. The solder bridge on the J17 must be removed.
When a solder bridge is on the right (the right pad and the middle pad are shorted), the SCI1 PS2 receives signal from J1, for RS485 input. The solder bridge on the J17 must be removed.

J15 Enables speaker. The speaker is driven by PT5 (pin 16), Output Comparator 3, through this jumper (SMD pads on the solder side).
J16 RS485 direction control by PJ0 (pin22) through this jumper (SMD pads on the solder side).
J17 IR receiver output selector
When a solder bridge is on the upper position (labeled with 'SCI'), the MC9S12DP256's PS2 (RXD1) receives the data from IR receiver. The PS2 can be programmed as general I/O lines or the UART SCI1. On this position, the J14 must not have a solder bridge and pin 91 of the header H4 must not be connected to user's circuits, otherwise signal contention may occur. When the IR receiver is using the RXD1, no other I/O devices should use it.

When the solder bridge is on the lower position (labeled with 'PT3'), the MC9S12DP256's PT3 receives data from the IR receiver. On this position, the RXD1 is free to be used by other I/O devices.

J18 Connecting a terminating resistor for CAN0. Place a solder bridge on it at the last node in a network. If CAN0 is not used, it will save power consumption of the board by removing the solder bridge.
J19 Connecting a terminating resistor for RS485. Place a solder bridge on it at the last node in a network. If RS485 is not used, it will save power consumption of the board by removing the solder bridge.
J21 LCD port
J22 Keypad header
J23 External 5V input when solder bridge is removed on J11.
J24 20X2 right angle male header
J25 20X2 female socket connector.

Analog voltage reference VRH is connected to the on-board 5V VCC via the trace between the pin 83 and pin 84 of the header H3, but the trace can be cut if a more accurate analog reference voltage is needed.

Analog voltage reference VRL is connected to VSS via the trace between the pin 85 and pin 86 of the header H4, but the trace can be cut if a more accurate analog reference voltage is needed.

IMPORTANT NOTES

The following are some important notes that you should know and they may save your 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, if you want to run the board in single chip mode, but MODEB and MODEA are set for expanded mode, you know it won't work. If the jumper on J9 is missing, the board won't work in EVB mode.

Before troubleshooting the board, you must apply power to the board. When the board is powered, the small 7-segment LED should display the diagnostic code of E-4-3 momentarily. The letter E stands for EVB mode, the number 4 stands for single chip mode and the number 3 means that 2 switches (S3, S4) are open.

If the LED display did not show anything, check if the board has 5V DC. Use a DMM to check voltages at different points in the circuit to find a defective component. Sometimes it may be caused by a bad AC adapter or the AC adapter is not even plugged in. Also check if a jumper is installed on J10. Without the jumper, the LED display will be off.

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

S3	AN5 input	No jumper
S4	AN6 input	No jumper
J7	Low Voltage Inhibit	jumper is on
J8	Mode B and Mode A	both jumpers are at the right sides of the 6-pin header
J9	PAD0	jumper is on the right sides of the 3-pin header
S7	PAD1	switch is set at the right side
J10	Enables LED display	jumper is on
J11	VCC enable	It's solder bridged (SMD pads on the solder side)
J14	SCI1receiver source selector	solder bridge is on two left pads
J17	IR receiver output selector	no solder bridge is on pads (SMD pads on the solder side)
Y1	16 MHz crystal	Y1 is installed

If all above settings are correct and when you press the reset button, the 7-segment LED should display the diagnostic code of E-4-3 momentarily. If this does not occur, the bootloader could have been erased by a BDM. If the diagnostic code is displayed correctly, but the board does not communicate with the terminal, the EVB portion of flash memory could be erased or the com port number may not be set correctly by the terminal program. If the screen displays some garbled characters, the baud rate may not be set correctly. The D-Bug12 resets the baud rate to 9600 during power up, if you changed the baud rate, you must change the AsmIDE's baud rate to the same value.

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. In EVB mode, reset clears your pseudo RAM interrupt vectors.

When you develop code in RAM, it's a good idea to add instructions to initialize the RAM interrupt vectors in the very beginning of the program. If you don't do it and your application program uses interrupt, after you press the reset button, which will clear the interrupt vectors, your program may not run correctly since the interrupt vectors do not exist.

4. Operating mode changing is only effective after reset.

There are four operating modes that are selected by J9 and S7. The mode change won't be effective until you reset the board. So you always press the reset button after a mode change.

5. Learn HC12 programming by modifying HC11 code.

When you learn programming on a new microcontroller family, you usually would create more bugs in your code. Sometimes you really cannot tell if your code is bad or the logic is bad. Here you have many example programs to hack. These fully debugged example programs are written in 68HC11 code. You can gradually replace the HC11 code with its equivalent HC12 code. You can run the program to verify your HC12 code immediately after each small replacement. Step by step, after you complete your code migration from HC11 to HC12 for the test.asm. You will be a pretty good HC12 programmer.