FreeEMS Workstation HowTo

Andy Goss

07/16/2012

This document assumes you are starting with a blank hard drive.
This means NO OPERATING SYSTEM.

Make sure you start with Chapter 1, Step 1 and do not skip any steps.
They are in this order for a reason!

The purpose of this document is to describe how to setup an Intel or AMD PC with 1 Tb HDD to Dual Boot Windows and Debian Testing (Wheezy) Linux for the average computer user. The author assumes that the user has installed Windows at least one or more times as instructions for the exact process of installing Windows will not be shown.

Please read through the ENTIRE document before beginning the installation process to make sure you fully understand each step of the process. The author is not responsible for any actions resulting from you the user following the instruction contained in this HowTo. This is a free world with no strings attached and YOU are responsible for YOUR actions.

To quote Morpheus from the movie "The Matrix": "This is your last chance. After this, there is no turning back. You take the blue pill - the story ends, you wake up in your bed and believe whatever you want to believe. You take the red pill - you stay in Wonderland and I show you how deep the rabbit-hole goes."

I am only your guide, you must make the decision to take the red pill or blue pill...
So if you decide to take the red pill and follow the white rabbit to the land of penguins...

Contents

Chapter 1

Step 1: Install Windows and Apps

1. Install Windows on half of the HDD space by creating an 500 Gb partition during the Windows installation. This can be either Windows 32bit or Windows 64bit. I won't go into detail of how to install Windows, if you aren't familiar with the install process try http://www.google.com. After Windows has been installed and updated, continue to the next step.

2. We need to install CodeWarrior so that we can load the Serial Monitor program onto the Freescale mc9s12xdp512 microprocessor. We really aren't interested in the CodeWarrior IDE as a development platform, but we need the HiWave Debugger to flash the Serial Monitor program onto the blank MCU and it is included in the CodeWarrior download and installation process.
   Download and install CodeWarrior for HCS12(X) Microcontrollers (Classic) IDE :
   `http://cache.freescale.com/lgfiles/devsuites/HC12/CW_HC12_v5.1_SPECIAL.exe`

3. Download PE Micro's free Unsecure-12 utility for the HCS12(X) MCUs in case you get the Freescale MCU stuck in a Secured state. You can download it here after you register for a free account:
   `https://www.pemicro.com/login.cfm?from_url=/downloads/download_file.cfm?download_id=16`

4. To use a USBDM/TBDML debug module (USBDM is preferred) with the HiWave debugger then you can clone my github repository which contains the complete information on how to build one and program it along with the Windows 32-bit driver that works with WinXP, Vista and Windows.

   - Here is a link to my repository:
     `https://github.com/DeuceEFI/TBDML-12f-GPL` `https://github.com/DeuceEFI/USBDM`
   - Once you plug in the USBDM/TBDML device into the USB port on your computer, Windows will ask for the location of the driver.
   - The Windows 32bit driver for the USBDM is located in the `USBDM Driver Installer` directory
   - The Windows 32bit driver for the TBDML is located in the `bin_tbdml_win_driver_11` directory

5. Search for and create a shortcut on your Desktop for hiwave.exe

   - Open the shortcut to hiwave.exe that you just created.
   - The defualt path to hiwave.exe on a 32bit system is:
     `C:\Program Files\Freescale\CWS12v5.1\Prog\hiwave.exe`

- The defualt path to hiwave.exe on a 64bit system is:
  `C:\Program Files(x86)\Freescale\CWS12v5.1\Prog\hiwave.exe`
- Click on Connection then click on Load..., then click on the Yes button
- For Processor, select HC12.
- For Connection, select TBDML.
- For the Set Derivative, select MC9S12XDP512.
- Now your HiWave debugger is setup to use the TBDML debug module with the Freescale MCU.

6. **Follow these instructions to setup Unsecure-12 in the HiWave debugger to perform the Unsecure task.**

   - Next click on **TBDML HCS12** then click on **Command Files**
     All the files below should be located in:
     `C:\Program Files\Freescale\CWS12v5.1\(CodeWarrior_Examples)\HCS12X\Banked-Data\cmd\`
   - Click on the **Startup** tab, then click the **Browse...** button and locate the **P&E_ICD_Startup.cmd** file, click the **Open** button,
     then click the checkbox for **Enable Command File**.
   - Click on the **Reset** tab, then click the **Browse...** button and locate the **P&E_ICD_Reset.cmd** file, click **Open** button,
     then click the checkbox for **Enable Command File**.
   - Click on the **Preload** tab, then click the **Browse...** button and locate the **P&E_ICD_Preload.cmd** file, click **Open** button,
     then click the checkbox for **Enable Command File**.
   - Click on the **Postload** tab, then click the **Browse...** button and locate the **P&E_ICD_Postload.cmd** file, click **Open** button,
     then click the checkbox for **Enable Command File**.
   - Click on the **Vppon** tab, then click the **Browse...** button and locate the **P&E_ICD_Vppon.cmd** file, click **Open** button,
     then click the checkbox for **Enable Command File**.
   - Click on the **Vppoff** tab, then click the **Browse...** button and locate the **P&E_ICD_Vppoff.cmd** file, click **Open** button,
     then click the checkbox for **Enable Command File**.
   - Click on the **Unsecure** tab, then click the **Browse...** button and locate the **P&E_ICD_Erase_unsecure_hcs12.cmd** file, click **Open** button,
     then click the checkbox for **Enable Command File**.
   - Click on the **OK** button to save your changes.

Chapter 2

Step 2: Download and create Linux boot media

In this step you will need to download and create the appropriate boot media for Debian Testing (Wheezy) 32 bit. You have a couple of options here, you can either go the route of Method #1: (download a .iso file and make a CD to boot from) or you can follow Method #2 (use unetbootin to make a bootable USB Thumb drive).

1. Download a disk image .iso file of your Linux distribution of choice, here are a few links:

   Debian Testing 32bit CD images:
   `http://cdimage.debian.org/cdimage/weekly-builds/i386/iso-cd/`

   Debian Testing 32bit DVD images:
   `http://cdimage.debian.org/cdimage/weekly-builds/i386/iso-dvd/`

   With any of the above .iso files use the CD burning software of your choice to make a bootable CD-ROM to launch the installer. Or you can also use unetbootin as in Method #2 and follow these directions:
   `http://unetbootin.sourceforge.net/#other`
   to create a bootable USB Thumb drive from the .iso file that you downloaded.

2. Download unetbootin from:
   `http://unetbootin.sourceforge.net/unetbootin-windows-latest.exe`
   Run the unetbootin-windows-latest.exe file while the computer is running from its Windows partition and follow the instructions at:
   `http://unetbootin.sourceforge.net/#install`
   to create a bootable Debian 32bit USB Thumb drive.

Chapter 3

Install Linux

Insert the boot media you created in Chapter 2 to install Debian Testing (Wheezy) on the remaining 500 Gb HDD space. Now reboot your computer making sure to choose the boot media you just created.

1. To reduce reduce the number of steps, this example begins at the step where disk detection starts after you have gone through the basic steps of the graphical installer. Select Manual. Click Continue.
2. Here, the installer should show either 1 NTFS partition for Windows XP or the 2 NTFS partitions for (Windows Vista or Windows 7) . As you can see, there is free, or unallocated space. Select FREE SPACE, then click Continue.
3. Scroll to Create a new partition. Continue.
4. By default the /boot partition on Debian Testing is allocated a disk space of about 250 MB. Most Linux distributions use 500 MB. I think 1000 MB will do just fine for my system. Continue.
5. If you are installing Debian to dual boot with Windows XP, you do not need to follow any further steps until step 49 so you can skip to step 49 now. If you are installing Debian to dual boot with Windows Vista or Windows 7, continue to the next step.
6. In Linux, you cannot create more than four primary partitions. And since we already have two primary partitions (for Windows Vista or Windows 7), the installer will by default want to create the first and subsequent Debian partitions as logical partitions. The partition numbers will, therefore, start at 5. The boot partition, which is the first one we are creating, will be /dev/sda5. Note this carefully, because GRUB will be installed on this partition for reasons that I will explain when we get to that step. Click Continue.
7. Click on Logical. Click Continue.
8. Click on Beginning. Click Continue.
9. On this next Partition Disks screen, select the "Use as:" "Ext3 journaling file system" and "Mount point:" "/boot" for the new partition. Then scroll to Done setting up the partition. Click Continue.
10. With /boot created. you can now create other partitions. For this example, I am going to create LVM partitions.

11. Select the free space and click Continue.
12. Click on Create a new partition. Click Continue.
13. Since this partition will be used as a Physical Volume for LVM, it is okay to use all available space on the disk. Click Continue.
14. Click on "Logical". Click Continue.
15. Click on Use as. Continue.
16. Select physical volume for LVM. If you prefer installing the system on an encrypted LVM (recommended), select physical volume for encryption. Scroll to Done setting up the partition. Click Continue.
17. With the PV created, scroll to Configure the Logical Volume Manager to create the Volume Group (VG). Click Continue.
18. At the "Write the changes to disks and configure LVM" prompt answer "Yes". Note the partition number of the boot partition (partition #5). That piece of information will come in handy several steps ahead. Click Continue.
19. Scroll to Create volume group. Click Continue.
20. Give the Volume Group a name. Any name will do. The shorter the better. Capitalization is optional. I called mine "HU", Click Continue.
21. These are all the partitions on the disk. Which one will be made a member of the VG? Only a Physical Volume qualifies to be included in a VG. In this example, that will be /dev/sda6, the Physical Volume that was created earlier. Select /dev/sda6 and Click Continue.
22. The Volume Group has been created.
23. The next task is to create Logical Volumes, which are the equivalents of disk partitions. For this example, only two Logical Volumes will be created: / and swap. You can create more if you like, but for a desktop system, those two should be all you need to create during installation.
24. Click on "Create logical volume" and click Continue.
25. The first step in creating a Logical Volume (LV), is to identify the Volume Group the disk space for the LV will be taken from. Click on "HU" and click Continue.
26. The first LV will be for /. Like the VGs name, the names for the LVs can be anything you feel comfortable with. For ease of management, it is better to use a name that reminds you of the mount point of the LV). I like to name the LV that will be used for / as "root" and "swap" for that which will be used for swap.
27. Name the Logical Volume "root". Click Continue.
28. How much disk space should be allocated to this "root" LV? I used 494 GB for the / partition. Click Continue.
29. Click on "Create logical volume" and click Continue.

30. Name the Logical Volume "swap". Click Continue.

31. How much disk space should be allocated to this "swap" LV? I used the remainder of the FREE SPACE (14 GB) for the swap partition. Click Continue.

32. Scroll to Finish. Click Continue.

33. The last task in creating LVs is to specify a file system and mount points. So, select the first (largest) LV, and click Continue.

34. Scroll to Use as to specify a file system for the / partition. The default on Debian 6 is "Ext3 journaling file system". You can stick with that or choose any other available journaling file system. Click Continue.

35. Scroll to Mount point and choose the mount point of "/" for the LV. Then scroll to Done setting up the partition. Click Continue.

36. Select the second (smallest) LV, and Click Continue.

37. Scroll to Use as to specify "swap". Click Continue.

38. Scroll to Done setting up the partition. Continue.

39. Scroll to Finish partitioning and write changes to disk. Click Continue.

40. Debian 6/Ubuntu 11 uses GRUB 2 as the bootloader, and will attempt to install it on the Master Boot Record (MBR) of the hard disk. Note, however, that installing GRUB in the MBR when you are attempting to dual-boot with Windows will overwrite Windows boot files. This is not the recommended approach. It is better to install it in the boot partition of the Debian installation. So, select "No" and click Continue.

41. All the steps involved in this tutorial are important, but this one is especially so. Remember the partition number of the boot partition? For this example, the boot partition is /dev/sda5. The partition number is, therefore, 5. The device to specify for installing GRUB, in GRUBs syntax, is (hd0,5). The 0 is the first hard disk and the 5, the partition number. Click Continue.

42. Continue with the rest of the installation. When completed, the system will reboot into Windows. This is expected.

43. The final task is to add an entry for Debian Wheezy in Windows boot menu. The easiest tool I know of to do that is EasyBCD, a free application from NeoSmart Technologies. You may download the latest version here:
`http://neosmart.net/dl.php?id=1`

44. Install EasyBCD and launch it.

45. This is the main tab of EasyBCD. It shows the only entry in the boot menu. Click on Add New Entry.

46. On the Add New Entry tab, click on the Linux/BSD tab in the upper part of the window. Debian 6 uses GRUB 2, so select GRUB 2 from the Type menu. Edit the Name field to Debian, then click on the Add Entry button. Click on Edit boot Menu.

47. Here are the two entries. Windows is the default. You can change the order if you want to. Exit EasyBCD and reboot.

48. To be sure that it works, boot into Debian. It should prompt you to boot either Windows Vista or Windows 7 if you followed all the steps above. Note that if you attempt to boot into Debian, you will next encounter the GRUB menu. If you were able to boot into Windows Vista or Windows 7 then you are done with this chapter, continue to chapter 4 to install the Linux libraries and programs.

49. For a dual boot Debian and Windows XP when the installation prompts you to install the GRUB bootloader, allow it to be installed in the Master Boot Record (MBR).

50. Reboot your computer when the Debian install is complete.

51. Notice that when the computer restarts you are presented with the GRUB bootloader and only the Debian choices are available. Let it boot into the default Debian instance and we will add Windows XP to the boot loader in the next step.

52. Open a terminal and type the following

    a) `$ sudo grub-mkdevicemap`

    b) `$ sudo update-grub2`

53. Reboot your computer.

54. Notice that when the computer restarts you are presented with the GRUB bootloader and now the Debian choices are available along with Microsoft Windows XP.

55. Your installation of both operating systems is now complete. At this point you should test and make sure that you can boot into both operating systems before continuing.

Chapter 4

Install Linux Libraries and Programs

1. Boot into Debian.
2. Open a Root Terminal, edit `/etc/apt/sources.list` and add:
   For 32bit OS:
   ```
   # FreeEMS Toolchain
   deb http://debs.diyefi.org css-debs binary
   ```
3. Open a Terminal window and type:
   `$ sudo gedit /etc/sudoers`
   and add your user under root.
4. In the same Terminal window, type:
   `sudo gedit /etc/group` and add your username to:
   `tty, dialout, sudo, operator, users, iocard and staff`
   You may not have all of these, only add your username after the groups that exist on your system.
5. These changes will require a reboot to assign the permissions, so do that now.
6. Open a Terminal window and update apt's database and install upgraded system files:
   ```
   $ sudo apt-get update
   $ sudo apt-get upgrade
   ```
7. Install the tools necessary for MegaTunix:
   In the same Terminal window, type:
   ```
   $ sudo apt-get install pkg-config libtool intltool libgtkglext1-dev g++ gcc flex
   $ sudo apt-get install bison glade libglade2-dev make git-core gdb automake1.9
   ```
8. Install the FreeEMS Toolchain so you can compile the firmware:
   In the same Terminal window, type:
   ```
   $ sudo apt-get install freeems-toolchain cutecom
   ```
9. Install linux-headers:
   In the same Terminal window, type:
   ```
   $ sudo apt-get install linux-headers-2.6.32-5-686
   ```

Chapter 5

Install Essential Programs

1. Install QT4
   Open a Terminal window and type:
   `$ sudo apt-get update && sudo apt-get install libqt4-dev libqt4-opengl-dev qt4-qmake libqwt-dev libqt4-declarative libqjson-dev`

2. Now the system should be ready to compile emstudio, MegaTunix, FreeEMS, FreeEMS-Loader and qserialport!

3. Create a git directory in your home directory:
   In the same Terminal window, type:
   `$ mkdir ~/git`

4. Change to your new git directory:
   In the same Terminal window, type:
   `$ cd ~/git`

5. Clone the git repositories for qserialport, freeems-loader, MegaTunix and freeems-vanilla
   In the same Terminal window, type:
   ```
   $ git clone git://gitorious.org/inbiza-labs/qserialport.git
   $ git clone git://github.com/seank/freeems-loader.git
   $ git clone git://github.com/djandruczyk/MegaTunix.git
   $ git clone git://github.com/malcom2073/emstudio.git
   $ git clone git://github.com/fredcooke/freeems-vanilla.git
   ```

   This creates a directory within the git directory named after each repository such as:
   ```
   ~/git/qserialport
   ~/git/freeems-loader
   ~/git/MegaTunix
   ~/git/emstudio
   ~/git/freeems-vanilla
   ```

6. Start with qserialport since we need it for SeanK's freeems-loader to communicate with the serial port:
   In the same Terminal window, type:
   ```
   $ cd ~/git/qserialport
   $ ./configure
   ```

```
$ make
$ sudo make install
$ sudo /sbin/ldconfig
```

Chapter 6

FreeEMS-Loader

FreeEMS-Loader is the program you will use to load the freeems-vanilla firmware onto your FreeEMS based engine management system, so here are the instructions of how to compile and install this program.

1. Make sure your freeems-loader git clone is up to date by excuting the following:
   Open a Terminal window and type:
   `$ cd ~/git/freeems-loader`
   `$ git pull`

2. Next compile freeems-loader:
   In the same Terminal window, type:
   `$ cd ~/git/freeems-loader/src`
   `$ qmake && make all`

3. Next test run freeems-loader:
   In the same Terminal window, type:
   `$ ./Freeems-Loader`

4. Create a Gnome menu item for FreeEMS-Loader:
   Click on `System -> Preferences -> Main Menu`
   Click on the "New Menu" button and call the new menu "FreeEMS".

5. Create an application called "FreeEMS-Loader":
   Click on "FreeEMS" and then click on the "New Item" button,
   The Type is "Application", for the Name type in "FreeEMS-Loader" and browse for `~/git/freeems-loader/src/Freeems-Loader` , then click OK.

Chapter 7

Install emstudio

emstudio is the tuning program you will use to make changes to your FreeEMS version 0.2.0 based engine management system so here are the instructions on how to install and run this program.

1. Make sure your emstudio git clone is up to date by excuting the following:
   Open a Terminal window and type:
   ```
   $ cd  ~/git/emstudio
   $ git pull
   ```
2. Now compile emstudio:
   In the same Terminal window, type:
   ```
   $ cd  ~/git/emstudio
   $ qmake
   $ make
   ```
3. Now run emstudio:
   In the same Terminal window, type:
   ```
   $ cd  ~/git/emstudio
   $ ./emstudio
   ```

Chapter 8

Install MegaTunix

MegaTunix is the tuning program you will use to make changes to your FreeEMS version 0.1.0 based engine management system so here are the instructions on how to install this program.

1. Make sure your MegaTunix git clone is up to date by excuting the following:
   Open a Terminal window and type:
   ```
   $ cd  ~/git/MegaTunix
   $ git pull
   ```
2. Now compile MegaTunix:
   In the same Terminal window, type:
   ```
   $ cd  ~/git/MegaTunix
   $ ./autogen.sh
   $ make
   $ sudo make install
   $ sudo /sbin/ldconfig
   ```

Chapter 9

FreeEMS Firmware

In this chapter are the instructions to build the freeems-vanilla firmware with specific instructions for the Deuce Coupe build.

1. Make sure your freeems-vanilla git clone is up to date by excuting the following:
   Open a Terminal window and type:
   ```
   $ cd  ~/git/freeems-vanilla
   $ git pull
   ```
2. Build the FreeEMS Firmware:
   In the same Terminal window and type:
   ```
   $ cd  ~/git/freeems-vanilla/src
   $ CLIFLAGS="-D DEUCECOUPE=1" make clean s19
   ```
3. Your newly built FreeEMS firmware can be found in:
   ```
   $ cd  ~/git/freeems-vanilla/src/firmware
   ```
4. freeems.serial.monitor.s19 is located in:
   ```
   $ cd  ~/git/freeems-vanilla/lib
   ```

Chapter 10

Set User Permission to USB Device

In this chapter we will setup your USB TBDML debug module so that your normal user can use it without using the **sudo** command.

1. Install libusb:
   `$ sudo apt-get install libusb-0.1.4 libusb-1.0-0 libusb-1.0-0-dev libusb-dev`
2. To get libusb devices to run without being root:

   a) Plug in your USB device and run:
   `$ dmesg | tail`
   to find IDVendor and IDProduct of your USB device.
   Example output:
   `[49182.484111] usb 2-4:  New USB device found, idVendor=0425, idProduct=1000`

   b) Modify /etc/udev/udev.conf and ensure that it at least has following settings:
   ```
   $ sudo gedit /etc/udev/udev.conf
   udev_root="/dev/"
   udev_db="/dev/.udevdb"
   # udev_rules="/etc/udev/rules.d/"
   # udev_permissions="/etc/udev/permissions.d/"
   default_mode="0660"
   udev_log="err"
   ```
3. Create USB udev rule:
   ```
   $ sudo cat > /etc/udev/rules.d/23-usb.rules << "EOF"
   # Set group ownership for raw USB devices
   # This is for the TBDML
   ACTION=="add", SUBSYSTEM=="usb", SYSFS{idVendor}=="0425", SYSFS{idProduct}=="1000",
   ENV{DEVTYPE}=="usb_device", GROUP="dialout", MODE="0660", SYMLINK+="tbdml-%k"
   #This is for the Saleae Logic Analyzer
   ACTION=="add", SUBSYSTEM=="usb", SYSFS{idVendor}=="0925", SYSFS{idProduct}=="3881",
   ENV{DEVTYPE}=="usb_device", GROUP="dialout", MODE="0660", SYMLINK+="saleae-%k"
   EOF
   ```

4. Comment out the rule in /lib/udev/rules.d/95-upower-wup.rules
   ```
   $ gedit /lib/udev/rules.d/95-upower-wup.rules
   Now add a # to the beginning of the line that reads like the one below:
   # SUBSYSTEM=="tty", SUBSYSTEMS=="usb", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001
   ATTRS{serial}=="A80", ENV{UPOWER_VENDOR}="Watts Up, Inc.", ENV{UPOWER_PRODUCT}="Watts
   Up Pro", ENV{UP_MONITOR_TYPE}="wup"
   Save your change
   ```

5. Kill and Respawn udev deamon and run the newly created rules:
   ```
   $ sudo skill udevd
   $ sudo /sbin/udevd -d
   ```

Chapter 11

hcs12mem

In this chapter we will download and test hcs12mem a Freescale HCS12 flashing utility.

hcs12mem is only to test to make sure you can communicate with the HCS12 core. You will not be using this utility for flashing anything to the MCU under Linux.

1. Download hcs12mem from:
   `http://sourceforge.net/projects/hcs12mem/files/latest/download`
   I have not been able to get hcs12mem to program the freeems.serial.monitor.s19 file onto a blank HC12S(X) MCU as of the writing of this how-to. This is considered a work in progress.
2. Install hcs12mem-1.4.1-linux.deb with GDebi Package Installer. This file will be located in `~/Downloads/`
3. Copy the Freescale mc9s12xdp512.dat file to the library for hcs12mem:
   `$ sudo cp ~/git/freeems-vanilla/bin/mc9s12xdp512.dat /usr/local/share/hcs12¬ mem/mc9s12xdp512.dat`
4. Connect the TBDML USB programmer to the USB port of your PC and connect the BDM cable to the BDM connector on the Freescale Microprocessor (MCU). The TBDML adapter is only used to program the Serial Monitor program onto the Freescale Microprocessor so that we can communicate with either the FreeEMS-Loader or the MegaTunix-Loader.
5. Test hcs12mem operation to query the Freescale MCU:
   `$ hcs12mem -i tbdml -t mc9s12xdp512`

   You should see output like this:

```
andy@DeuceEMS: /hcs12mem$ hcs12mem -i tbdml -t mc9s12xdp512
hcs12mem:  Freescale S12 MCU memory loader V1.4.1 (C) 2005-2007 Michal Konieczny
<mk@cml.mfk.net.pl>

target info <single chip MC9S12XDP512 MCU>
target mcu <MC9S12XDP512> family <S12X>
USB device USB version <1.1> VID <0x0425> PID <0x1000> device release <0.0.1>
USB device manufacturer <Freescale> product <TurboBDM Light v0.1>
USB device config <#1> power <bus> max current <300mA>
```

```
TBDML hardware version <1.0> firmware version <0.3>
TBDML detected target frequency <16.000 MHz>
BDM status <enabled,active,secured> clock <bus>
S12X part id <0xC410> family <XD> memory <512kB> mask <1.0>
S12X part security <unsecured> backdoor key <enabled>
S12X register space <2kB> address range <0x0000-0x07FF>
S12X RAM size <32kB> space <12kB> address range <0x1000-0x3FFF>
S12X FLASH module <FTX512K4> size <512kB> state <enabled> ROMHM <no>
S12X FLASH block <0> protection all <off> high area <2kB> low area <off>
S12X FLASH block <1> protection all <off> high area <2kB> low area <off>
S12X FLASH block <2> protection all <off> high area <2kB> low area <off>
S12X FLASH block <3> protection all <off> high area <2kB> low area <off>
```

6. Test Erasing the entire Flash memory to default the Freescale MCU:
   `$ hcs12mem -i tbdml -t mc9s12xdp512 flash-erase-unsecure`
   ***If you run this command and the CPU goes into Secured mode. Boot into Windows and run HiWave Debugger and Unsecure the CPU.

Chapter 12

Build a BDM

I have used a TBDML in the past to connect the MCU to the HiWave Debugger, I have included a link to my github repository where you can find the information to make your own as well as the driver for Windows XP 32-bit, Vista 32-bit and Windows 7 32-bit editions.

I recommend using a USBDM to connect the MCU to the HiWave Debugger, I have included a link to my github repository where you can find the information to make your own as well as the driver for Windows XP 32-bit.

1. Clone my USBDM repository from https://github.com/DeuceEFI/USBDM (recommended)
2. Clone my TBDML-12f-GPL repository from https://github.com/DeuceEFI/TBDML-12f-GPL

3. Follow the instructions contained in the file "USBDM$_{I}nstructions.md"tobuildaUSBDMandinstallthe32-bitdriver$.Follow the instructions in Daniel Malk's instructional PDF to build a TBDML and install the included 32-

Chapter 13

Flashing the Freescale MCU

In this chapter we will learn how to flash the Serial Monitor (SM) program onto your Freescale Microcontroller Unit.

4. Boot into Windows.

2. Connect the USBDM/TBDML to your computer.

3. Run the hiwave.exe shortcut you created in Chapter 1, Step 1, Instruction 5.

4. Program the Freescale MCU with the Serial Monitor program by:

   a) Click on **TDBML HCS12**

   b) Click on **Unsecure**

   c) Click on **TDBML HCS12**

   d) Click on **Flash**

   e) Click on **Load**

   f) Choose **freeems.serial.monitor.s19**, click on Open.

   g) Click on Close.

5. Reset the Freescale MCU to Normal Mode:

   a) Click on **TDBML HCS12**

   b) Click on **Reset to Normal Mode**

   c) Exit HiWave Debugger.

Chapter 14

Test Serial Monitor

In this chapter we will learn how to test the Serial Monitor to make sure it is working properly.

1. Boot back into Debian OR Ubuntu.
2. Open **cutecom**, click on the **Hex Output** check box, to the right of **Send file...** button, make sure **Plain** is selected and in the other box make sure **LF line end** is selected.
   a) Set it up to use device /dev/ttyF0 or /dev/ttyS0 or /dev/ttyUSB0 (which ever you are using)
   b) The Serial Monitor uses 115200 baud, 8 data bits, 1 stop bit, no parity.
   c) Connect a serial cable to the FreeEMS MCU.
   d) Click **Open Device**, click on the **Send file...** button and select the file **serial.mon.reset**
   e) It should respond back with: e0 08 3e
   f) This means that the Serial Monitor program is running.
   g) Close out of **cutecom**.

Chapter 15

Load FreeEMS Firmware

In this chapter we will learn how to load the freeems-vanilla firmware onto your Freescale MCU using Sean's Freeems-Loader. You will use this procedure to load any firmware files that you compile in the future onto the Freescale MCU.

1. To load the FreeEMS firmware onto your MCU, open FreeEMS-Loader and do the following:
   a) Change the Run/Load switch to Load on the either the Technological Arts board and press the RESET button OR install the shorting jumper on the Load/Run header on the Jaguar PCB and power cycle the PCB to tell the Serial Monitor program to accept an incoming file.
   b) In FreeEMS-Loader, click on the **Connect** button.
   c) Click on the **Open File** button, select the FreeEMS .s19 firmware file you want to load and click on the **Open** button.
   d) Click on the **Load** button. It will now load the file onto the MCU.
   e) Click on the **Close/Rst** button.
   f) Close FreeEMS-Loader and remember to change the Load/Run switch back to Run on the TA board and press the RESET button or remove the Load/Run shorting jumper on the Jaguar PCB and power cycle the PCB.

Chapter 16

Using git

In this chapter we will learn how to use git.

1. The following is a list of useful git commands:
   a) `git clone git://github.com/user_name/repository_name.git` -this clones the repository to your computer
   b) `git pull` -this retrieves the latest commit changes from the remote repository to your computer
   c) `git commit -a` -this allows you edit the commit message before pushing the changes to the repository on the Internet
   d) `git push` -this uploads your changes to the repository on the Internet
   e) `git branch` -this shows the branches in the repository along with indicating the current branch with an
   f) `git rev-parse HEAD | cut -c1-10` -this shows the first 10 digits of the commit hash
   g) `git diff --stat` -this shows the changes you have made to the repository
   h) `git branch` -this shows the branches in the repository along with indicating the current branch with an
   i) `git checkout dev` -this changes the working branch to the **dev** branch in the repository
   j) `git fetch` -Fetches named heads or tags from one or more other repositories, along with the objects necessary to complete them.