Getting Started for Advanced Users and Developers

Setting up the Imote2 Development Environment for TinyOS 1.x and Installing the ISHMP Services Toolsuite

Illinois Structural Health Monitoring Project

January, 2011

ILLINOIS STRUCTURAL HEALTH MONITORING PROJECT
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
OPEN SYSTEMS LABORATORY & SMART STRUCTURES TECHNOLOGY LABORATORY

Table of Contents

Introduction

This document will guide you through the process of installing and configuring the development environment for the Imote2 wireless smart sensor produced by MEMSIC (http://memsic.com) and installing and configuring the Illinois Structural Health Monitoring Project (ISHMP) Toolsuite. The programming environment for the Imote2 is based on TinyOS 1.x (http://www.tinyos.net) using Cygwin (http://sourceware.org/cygwin) on a Windows platform. While some programming knowledge is required to create new Imote2 programs, this guide assumes no such background and should be suitable for end-users of Imote2 applications as well as application developers. The ISHMP Toolsuite, which will be installed onto the Imote2s, comprises open source middleware services, sample applications, and tools to monitor structures with a dense network of Imote2 smart sensors.

Following the instructions in this guide will help the user to:

- installing the software needed to work with the Imote2
- connecting the Imote2 to the computer
- interfacing with the Imote2
- Install the ISHMP Toolsuite.
- Run a test application using an Imote2 and an attached sensor board.

Following the steps outlined in this guide takes anywhere from two hours – if you have a fast internet connection and encounter no problems – to a full day's work. Reading and following the instructions carefully will help the user to install the environment and Toolsuite quickly while trying to move through the instructions hastily will likely cause the user to miss a small step that will delay the installation as they try to figure out what went wrong.

The current version of these instructions includes a new initial step that was created to address some of the problems that users have encountered when installing the Imote2 environment. This step also introduces the various programs that will be installed and the hardware that will be required. Every computer has different settings and permissions that may cause installation problems members of the Illinois SHM Project have not encountered. The Illinois SHM Project therefore cannot make any guarantees that following these instructions will lead to successful installation of the Imote2 environment or the ISHMP Toolsuite. However, using a computer that runs Windows 7 or Windows XP with the latest updates provided by Microsoft will almost guarantee success. Users of Windows Vista may have to follow a few additional steps as outlined in the instructions. The Illinois SHM Project does not recommend using a Mac computer and does not provide additional instructions on installing the environment for them.

The Illinois SHMP Toolsuite software is open source, but not public domain. The Toolsuite's license (http://shm.cs.uiuc.edu/license.html), a copy of which is included at the end of this document and in the software distribution, allows for unlimited use and modification of the software for non-commercial purposes.

The Illinois SHM Project frequently issues new versions of the software that increases its functionality and improves its performance. To install a new version of the software on to a computer already equipped with the Imote2 environment, follow at least Steps 12 to 14 of this guide. It is advisable to complete Steps 15 and 16 as well just to confirm that the new installation was successful.

This guide was originally created by Kirill Mechitov for the Illinois SHM Project, and is derived from the MEMSIC Imote2 install guide. It has since been updated by Shin Ae Jang, Jennifer Rice, Sung-Han Sim, Hongki Jo, Robin Kim, Brian Schertz, Ryan Kent Giles, and Nicholas Wierschem.

Comments and questions:

If you have questions about the software or this guide, or run into problems, please join us on the Imote2 discussion forum: http://vibration.shef.ac.uk/imote2_forum.

Step 1 - Before You Begin

Users from around the globe have installed the Imote2 environment and the ISHMP Toolsuite on their computers using these instructions. Their feedback has led to the creation of this step that will hopefully provide guidance in selecting hardware and software that will make the installation process smoother.

1.1 *Necessary Equipment*

In order to complete and test the installation of the Imote2 environment and the ISHMP Toolsuite, the following hardware is required:

- Computer (All Steps)
- IPR2400 Imote2 (Steps 8,9,11,14-16)
- USB to mini cable (Steps 8-11,14-16)
- IIB2400 interface board (Steps 10-11,14-16)
- Sensor board (choose one of the following) (Steps 15-16)
    - ITS400C sensor board (A or B)
    - ISM400 (formerly SHM-A) sensor board

If you are installing the software for a situation where the Imote2 and interface board will be provided later, you can complete Steps 1-8 and 12-13 in preparation. In addition to the hardware mentioned above, a good internet connection is required to download the required software. Some downloads can take more than an hour on a slow connection so it is recommended that your connection be as fast as possible.

1.2 *Sensor Board Selection*

As noted in Step 1.1, the ISHMP Toolsuite can support two types of sensor boards. The first is the MEMSIC ITS400 (http://www.memsic.com/support/documentation/wireless-sensor-networks/category/7-datasheets.html?download=137%3Aits400) in either of the formats, A or B, that it has been marketed in. The Illinois SHM Project has also developed a sensor board, the ISM400, which is produced by MEMSIC. This board was designed specifically to obtain the high quality data needed in performing SHM on real structures. More information about this sensor board is available on the ISHMP website (http://shm.cs.uiuc.edu/hardware.html) in the ISM400: Datasheet and User's Guide.

1.3 *Computer Selection*

As noted in the introduction, the Illinois SHM Project recommends installing the Imote2 environment on a computer that runs Windows XP. Any desktop or laptop will be sufficient as long as one USB port is available. Older computers might have problems communicating with the Imote2. At times the Imote2 will try and send data back to the computer at a higher baud rate than the computer can handle. Though the Imote2 for Structural Health Monitoring User's Guide available at http://shm.cs.uiuc.edu/AdvancedUser.html provides some trouble shooting tips for resolving this problem, it can be avoided by ensuring the computer has a relatively fast processor with ample RAM.

Those who choose to use a machine with Windows Vista may have problems installing the software due to the way this operating system handles default ownership permissions. If you encounter a problem during installation (usually during Step 5 – Installing the NesC Compiler) you will have to manually make yourself the owner of the folder and give yourself full read, write, and execute permissions. Also, Windows 7 may have problems detecting imote2s when configure an application (during Step 7- Testing the setup). You will have to change your account to administrator to see if that allows you to detect the imote2.

The Illinois SHM Project does not recommend using a Mac and has no tips or instructions on installing the Imote2 environment on these computers.

1.4 *User and Computer Name*

The Imote2 environment centers on the Cygwin software package. Cygwin provides an UNIX-like interface within the Windows operating system. During installation, Cygwin will create a home directory that is located in a folder named for the current Windows user. For example, if the Windows user name were “ISHMP” then Cygwin will create a home folder called “ISHMP”. User names that contain spaces, dashes, slashes or other non-alphanumeric characters will cause problems in the Cygwin environment. It is recommended that you change both the user name and computer name so as not to contain the problematic characters.

If you have already installed Cygwin and have a two word user name, then you need to add a line to your `.bashrc` file. Adding a line to the `.bashrc` file is explained in Step 4 – Configuring the TinyOS 1.x Tree for Imote2s. For example if your user name were “Illinois SHM”, then you would add the following line:

- `export HOME="/home/Illinois\ SHM"`

In this instance the space character has been preceded by a backslash. This should rectify the problems of Cygwin not recognizing your home directory during the installation process.

1.5 *Text Editor*

Some of the steps involved in installation require the use of a text editor. Though Windows comes with Notepad and WordPad preinstalled, we find that these are not suitable for editing the files required to install the Imote2 environment. The reason is that they will save the file in a text format suited for the DOS environment that Windows uses. Cygwin however operates in a UNIX environment and has a different text format. We recommend using Programmer’s Notepad, a useful text editor that can be downloaded from http://www.pnotepad.org/ for free. This program will save the file in the proper format and also is helpful when programming additional features for the Imote2.

If you do not wish to install Programmer’s Notepad or use VIM or EMACS within the Cygwin environment, there is a command that will convert DOS text into UNIX text. The command to type into Cygwin is `dos2unix` *`filename`*. This must be done every time after the file has been changed using a DOS based text editor.

1.6 *Web Browser*

Recently, one of the files downloaded during the installation process has not been downloading properly when the user has been using Internet Explorer. Other web browsers such as Firefox and Chrome are known to work properly. The Illinois SHM Project therefore recommends using one of these browsers to download the necessary files.

Step 2 - Installing Cygwin

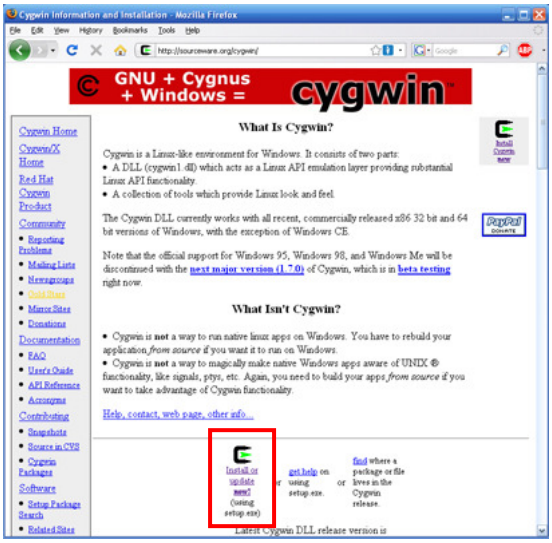
Cygwin is a software environment that provides UNIX-like functionality on a Windows PC. Many of the same applications commonly used in UNIX (and Linux) systems can be compiled for Cygwin and run in Windows. More importantly, TinyOS requires Cygwin to work on a Windows system. Cygwin is free software released under the GNU General Public License.

If you have previously installed Cygwin, run `setup.exe` again and check that you have the packages listed below installed. Also note that your directory might be different from those mentioned below.

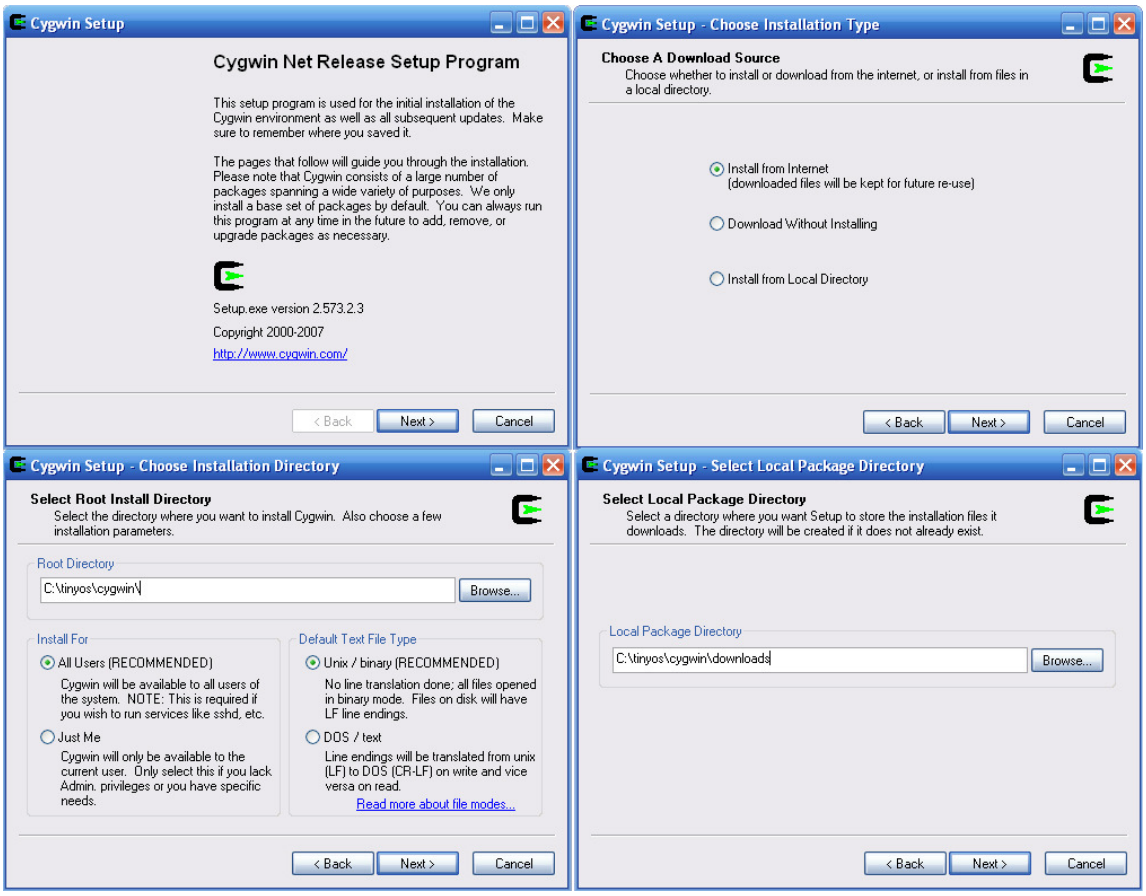
2.1 *Cygwin Installation*

If you do not already have Cygwin installed:

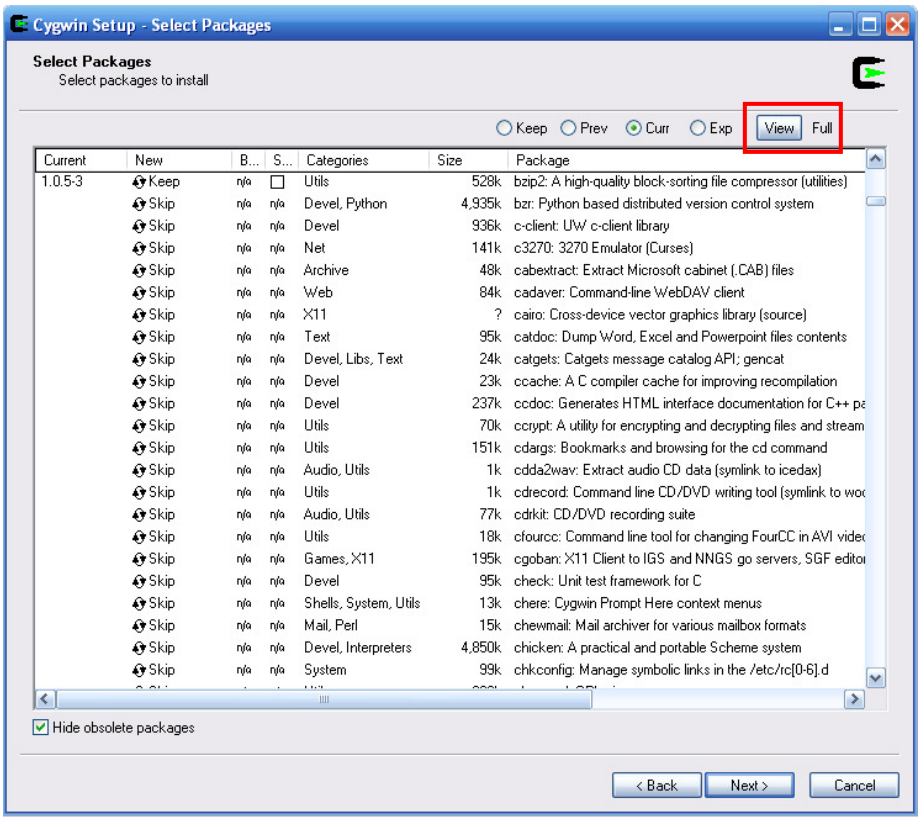
- Download the latest version from Sourceware (http://sourceware.org/cygwin/)



- Install Cygwin at `C:\tinyos\cygwin`
- Select the options "install for all users" and "use UNIX file types".
- Select a directory to download the packages (e.g., `C:\tinyos\cygwin\downloads`)



- Use a “Direct Connection” to connect to the internet.
- Select a repository to download packages from. Any repository will do, but the closer it is geographically to you the faster the download will be.
- On the list of packages, click on the "View" button once, so that the text to its right reads "Full".
- Maximize the window so it is easy to see all the columns.



- Select the following packages from the list by clicking on the "Skip" word on that package's line once. If you see a version number instead of "Skip", do not click it; the package is already selected. The packages are listed in alphabetical order.
    - autoconf
    - automake
    - binutils
    - ctags
    - cvs
    - cvsutils
    - diffutils
    - gcc
    - gdb
    - libintl2
    - make
    - rpm
    - unzip
    - util-linux
    - zip
- If you failed to install a package, you can run Cygwin's `setup.exe` file again to install any missing packages. In fact, it is recommended that you copy the file `setup.exe` to the `C:\tinyos\cygwin` directory for future use.

- **Note:** Typical Windows shortcuts like CRTL+C or CTRL+V for cutting and pasting text will not work in the Cygwin window. To paste text that you have cut from this guide or other sources, follow the following instructions:
  - Right click on the icon on the top left of the Cygwin window.
  - Scroll down to `Edit`
  - Select `Paste`.

2.2 *Home Directory*

During installation, Cygwin created a "home directory" for you. The "home directory" is the directory you are in automatically when you open a Cygwin shell by clicking on the Cygwin icon on the desktop. In Windows, this directory is C:\tinyos\cygwin\home\username, where "username" is your Windows user name. If your user name has two words or any non alphanumeric characters, please refer to Step 1.4 before proceeding.

In Cygwin, you can get to the home directory by using the command `cd` (for change directory). All of the following commands are equivalent and will take you to the home directory:

- `cd`
- `cd ~`
- `cd ~username`
- `cd $HOME`
- `cd /home/username`
- `cd /cygdrive/c/tinyos/cygwin/home/username`

When you see "~" or "home directory" in this guide, it refers to this directory.

Step 3 - Downloading the TinyOS 1.x Source Code

TinyOS is the operating system used by Imote2s and many other smart sensor platforms. Though a 2.x version is available, the ISHMP uses the TinyOS 1.x and only this version should be installed as the ISHMP software is not compatible with the 2.x version. The latest version of the TinyOS 1.x source code is available from a CVS repository at SourceForge. These instructions will install TinyOS 1.x in the directory `~/opt/tinyos-1.x`. From your Cygwin shell, run the following commands:

- `mkdir /opt`
- `chmod go+w /opt`
- `cd /opt`
- `touch ~/.cvspass`
- `cvs -z3 -d:pserver:anonymous@tinyos.cvs.sourceforge.net:/cvsroot/tinyos co -P tinyos-1.x` (Note: This command is all one line)

This last command will download the TinyOS 1.x tree to your computer. The download will take anywhere from 10 minutes to 1 hour depending on your internet speed. When the download is complete, run the following commands:

- `mv tinyos-1.x tinyos-cvs`

- `ln –s tinyos-cvs tinyos-1.x`
- `ln –s tinyos-1.x tinyos`

The last two lines set up symbolic links called `/opt/tinyos` and `/opt/tinyos-1.x` to point to the TinyOS source tree. Symbolic links are similar to, but not the same as, shortcuts. This step must be completed in Cygwin as shortcuts created in Windows are not treated the same as a symbolic link. Since some applications require changes to the core files found in the source tree, you can keep multiple copies of the tree and switch back and forth between the versions by updating only the tinyos-1.x symbolic link by entering the command:

- `ln –s tinyos-newtree tinyos-1.x.`

Step 4 - Configuring the TinyOS 1.x Tree for Imote2s

In order to make the tree downloaded in Step 3 operable, a few modifications need to be made to the TinyOS 1.x tree.

4.1 *Create Symbolic Links*

The files needed to operate the Imote2 are found in the `pxa27x` and `imote2` subdirectories in `/opt/tinyos/beta/platform`. In order to use them properly, links are needed to these directories in the main TinyOS platform directory. Create symbolic links to the platform files in the main TinyOS platform directory by running the following commands in a Cygwin shell:

- `cd /opt/tinyos/tos/platform`
- `ln –s ../../beta/platform/imote2`
- `ln –s ../../beta/platform/pxa27x`

4.2 *Change File Permissions:*

To run the `iMoteConsole` application, its file permissions need to be changed to make it executable. To do so, run the following command:

- `chmod a+x /opt/tinyos/contrib/imote2/tools/bin/iMoteConsole.exe`

4.3 *Edit the .bashrc File*

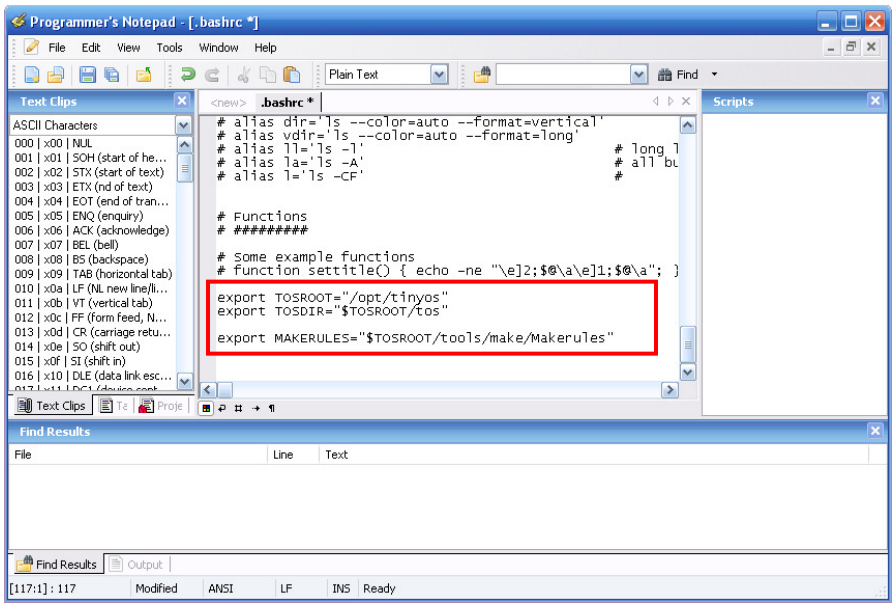
Cygwin read the `.bashrc` file every time it is opened to allow the user to declare certain environmental variables. To make using TinyOS easier in Cygwin, we will create a few variables that point to various directories and files. For Cygwin to recognize the changes made to the `.bashrc` file, all Cygwin windows must be closed and the program restarted. Therefore make sure to restart Cygwin after making any changes to your `.bashrc` file. To make the necessary additions, open the `.bashrc` file in your home directory (`~/.bashrc`) with a text editor (see Step 1.5), and add the following lines to the end to set up the TOSDIR and TOSROOT variables:

- `export TOSROOT="/opt/tinyos"`
- `export TOSDIR="$TOSROOT/tos"`

To set up the new `make` utility to be used by your `Makefile` when compiling programs, also add the following line after the previous two:

- `export MAKERULES="$TOSROOT/tools/make/Makerules"`

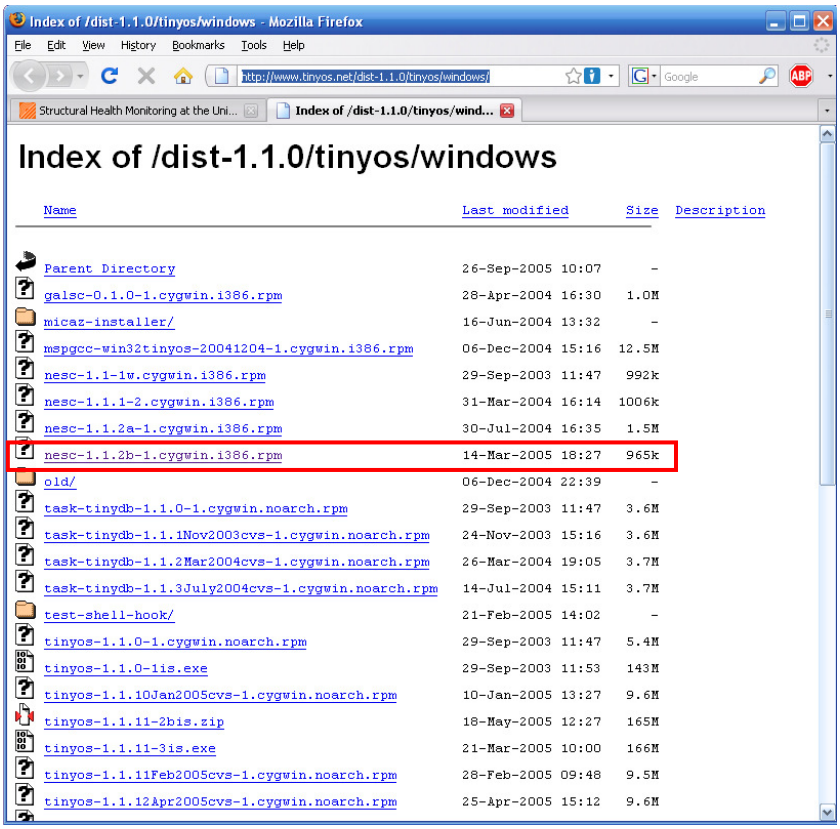
Save the file and then close all Cygwin windows and restart the program.



Step 5 - Installing the NesC Compiler

The NesC compiler translates applications and TinyOS platform files from the NesC language in which they are written to the plain C language. Programming the motes is much easier in NesC than doing it directly in C. To install the NesC compiler:

- Download the compiler from the TinyOS website (http://www.tinyos.net/dist-1.1.0/tinyos/windows/) into your home directory. Make sure you get the latest version available (`nesc-1.xxxx.cygwin.i386.rpm`) where `xxxx` is the highest number.

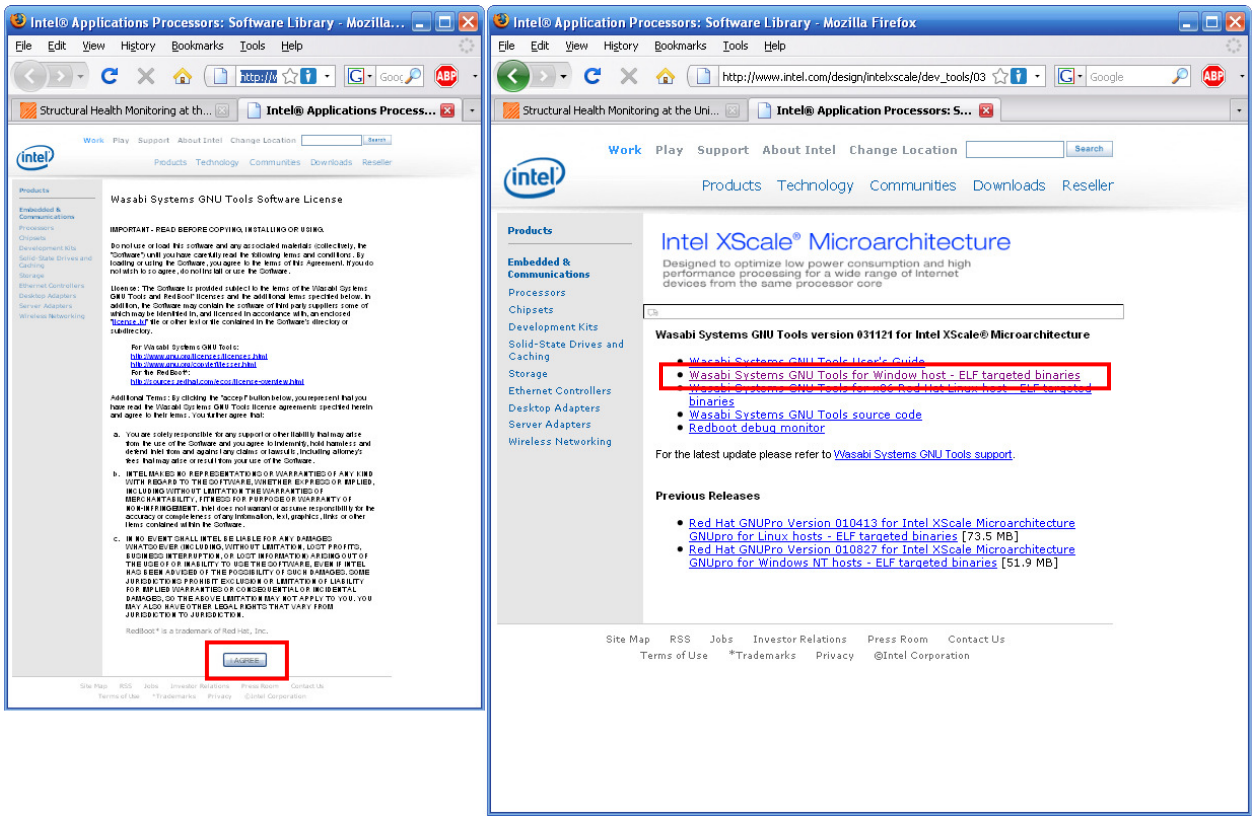


- Install the rpm package by executing the following command in a Cygwin window:
  - `rpm --ignoreos -ivh ~/nesc-1.xxxx.cygwin.i386.rpm`
- Note: It is on this step that Vista users will likely need to change the permissions of the `/opt` folder so that you are the owner and have read, write, and execute permissions (See Step 1.3).

Step 6 - Installing the Wasabi tool suite

The Wasabi tool suite from Intel contains the compiler and related tools needed to create binary images of Imote2 applications from the C code created by the NesC compiler. To install it:

- Download the wasabi tool suite from the Intel website (http://www.intel.com/design/intelxscale/dev_tools/031121/)



- Extract the package to `C:\tinyos\cygwin`, overwriting existing files if prompted.
- From a Cygwin window, run
  - `ln -s /bin/true /usr/local/bin/dwarf2bd`
- Unfortunately, Intel hard-coded a path to drive `E:` in the files. If you have a CD or a removable drive with that letter, either put a disk in that drive (any non-blank data CD or DVD will do), remove the device from the system, or disable the device by going to the `Start Menu -> Control Panel -> System -> Hardware -> Device Manager`. For a graphic example, see Step 10.2.
- Restart the Cygwin shell and check whether the compiler works by doing the following:
  - Create a text file `test.c` in your home directory using a text editor (See Step 1.5) with the following contents followed by a blank line:
    - `int main(int argc, char **argv) { return 0; }`
  - Run the following command
    - `xscale-elf-gcc -o test.exe test.c`
  - If the previous command created the file `test.exe` in your home directory and no errors are given, the Wasabi tool suite installation is successful. You can check if the file is present by using the command `ls` (for "**ls**t") while in your home directory.
  - You can now safely remove the two files, as they are quite useless by running the following command:
    - `rm -f test.c test.exe`

Incidentally, you can use `rm` (for "**rm**ove") to delete files in Cygwin, `rmdir` to delete empty directories, and `rm -fr` to delete directories and all files and subdirectories they contain.

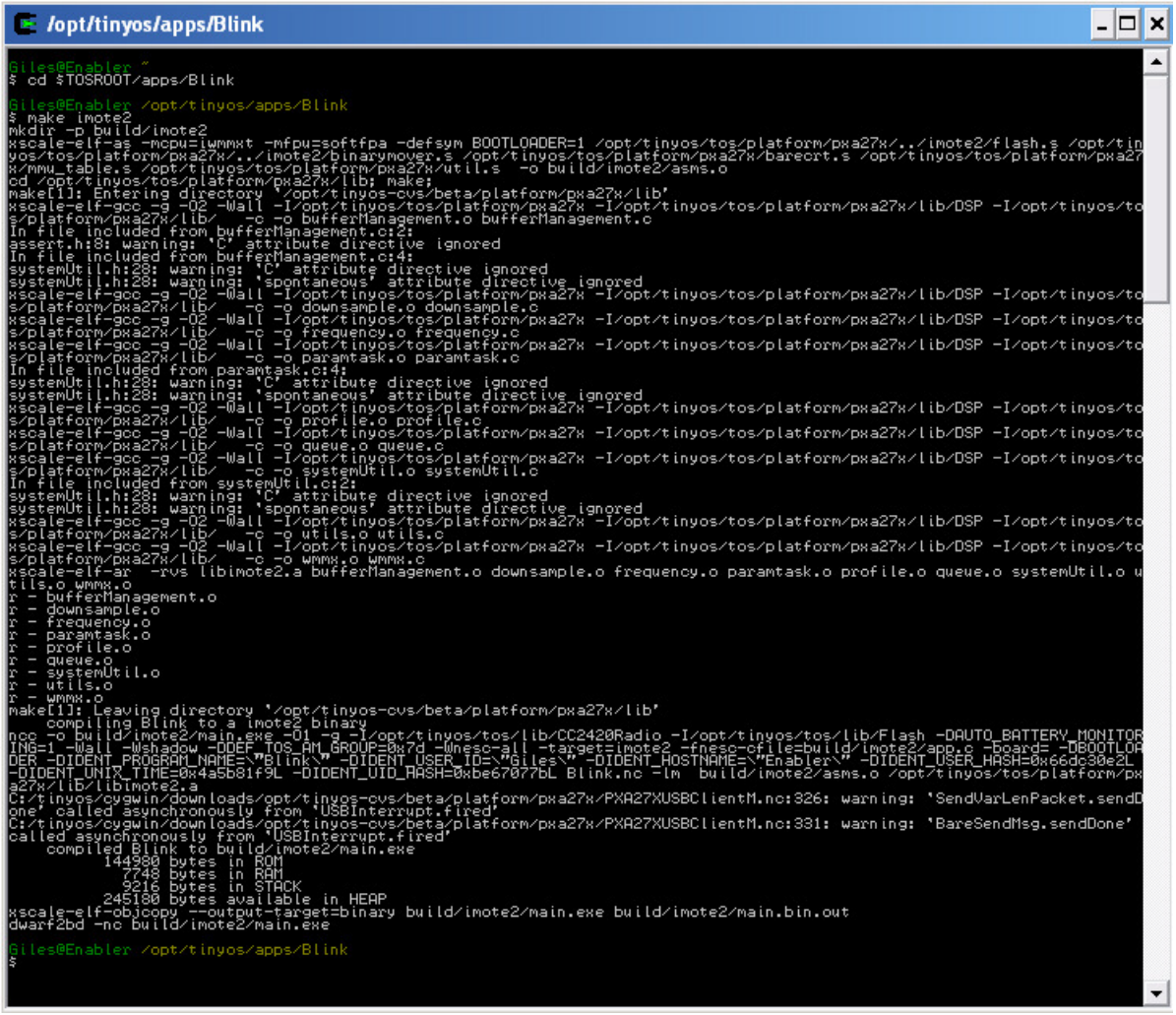
However, be aware that these commands do not send the files to a recycle bin and therefore the files cannot be recovered once the command is issued!

Step 7 - Testing the setup

To test the installation setup up to this point, we will compile a test application for the Imote2 without actually installing it on the Imote2 at this time. In a Cygwin shell do the following:

- `cd $TOSROOT/apps/Blink`
- `make imote2`

This `make` command should execute successfully if your setup is working properly. Any "warnings" can be safely ignored, but "errors" indicate something is wrong with your installation. If you get any "errors" recheck your environment variables, installed Cygwin packages, etc. as installed in previous steps. There will be a lot of output onto the Cygwin window. When compiling any application, the output to the screen may pause at some point. Be patient; the compilation is continuing even though the output to the screen has paused. Compilation is finished when the command prompt returns. The figure below shows a successful installation with warnings but no errors.



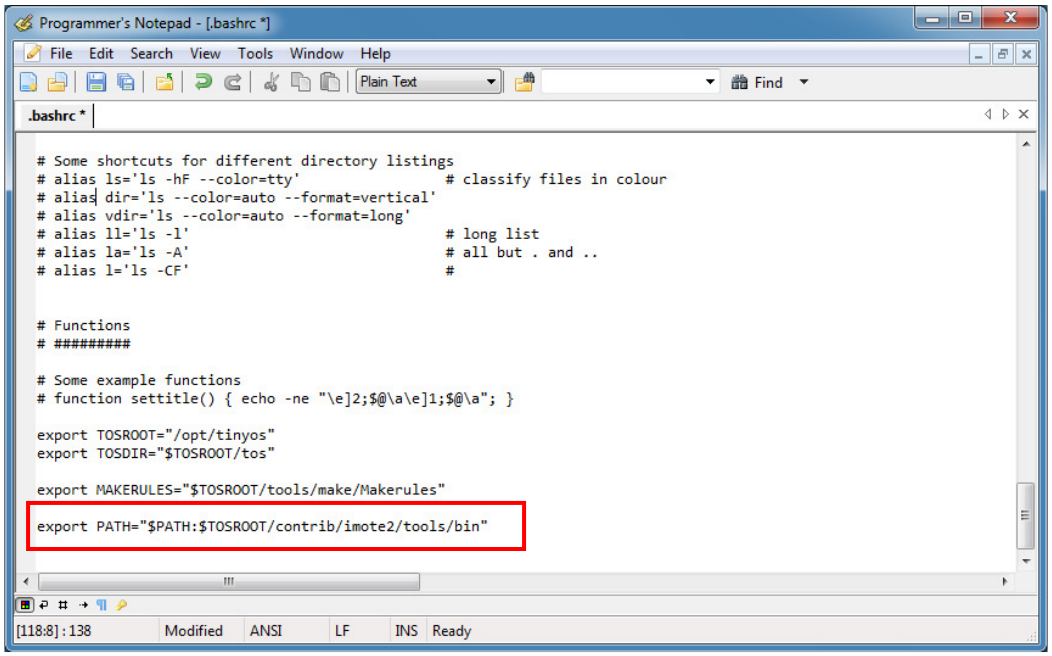
Step 8 - Installing Imote2 Applications with the USB loader

MEMSIC IPR2400 Imote2s (http://memsic.com/support/documentation/wireless-sensor-networks/category/7-datasheets.html?download=134%3Aimote2), as shown below, come pre-loaded with a TinyOS 1.x compatible USB loader. This application called `USBLoaderHost` and located in `$TOSROOT/contrib/imote2/tools/bin`, serves to upload program images to the Imote2.

8.1 *Add USBLoaderHost Location to PATH*

We will add the location of the `USBLoaderHost` application to the `PATH`, so we do not have to type out that long path name each time. To do so:

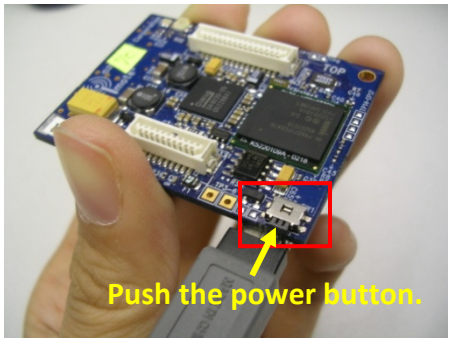
- Open `~/.bashrc` with a text editor (see Step 1.5).
- From Step 8.1 you should already have an "`export PATH="..."`" line in the `.bashrc` file. Add the `$TOSROOT/contrib/imote2/tools/bin` path following that one on the same line and preceded by ":", like so:
  - `export PATH="$PATH:$TOSROOT/contrib/imote2/tools/bin"`
- Restart Cygwin.



8.2 *Install Blink Test Application*

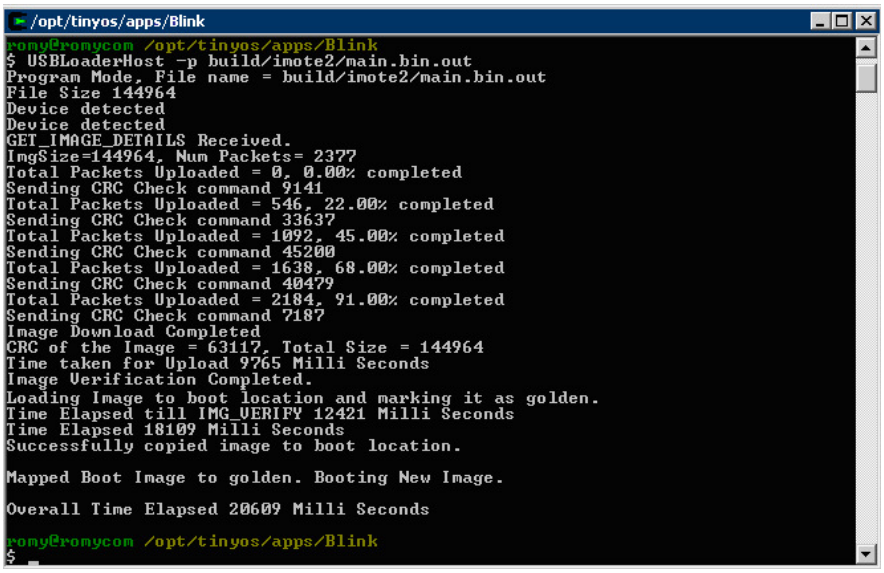
In order to install your TinyOS 1.x applications on the Imoet2, follow the steps in this section. In order to guide you through the steps once, we will install Blink. Blink is a small program that will have the red LED on the Imote2 blink once per second. To install Blink on the Imote2:

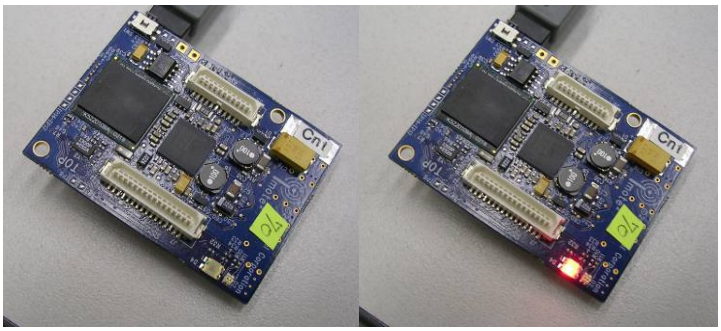
- Compile Blink if you haven't already by doing the following in Cygwin:
  - `cd $TOSROOT/apps/Blink`
  - `make imote2`
- Plug a USB cable into the PC and directly into a turned off Imote2 (NOT into the Imote2 interface board, if you have one).
- Once the turned off Imote2 is connected to the computer, push the power button on the Imote2 to turn it on. The power button is the little button right above the USB connector.
  - If this is the first time you have plugged in this particular Imote2:
    - Let Windows install the USB HID driver.
    - Unplug and then re-plug the Imote2 into the computer
    - Push the power button again.



- Install the application with the USB loader:
  - `USBLoaderHost -p build/imote2/main.bin.out`

- This command will download the application to the Imote2 using the USB connection. After the download is done, the Imote2 should begin to blink the red LED at 1Hz.

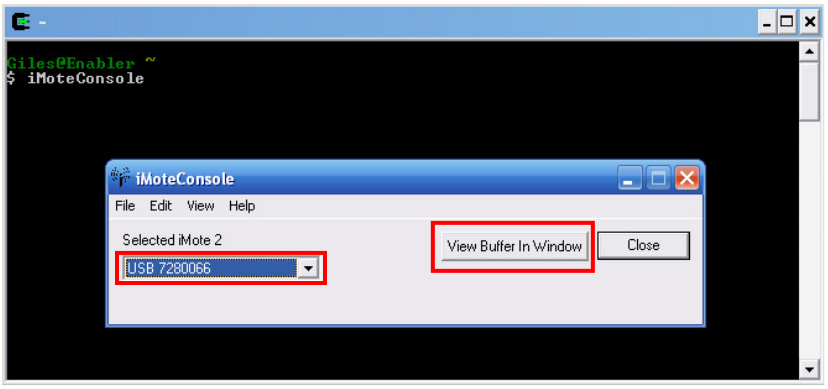




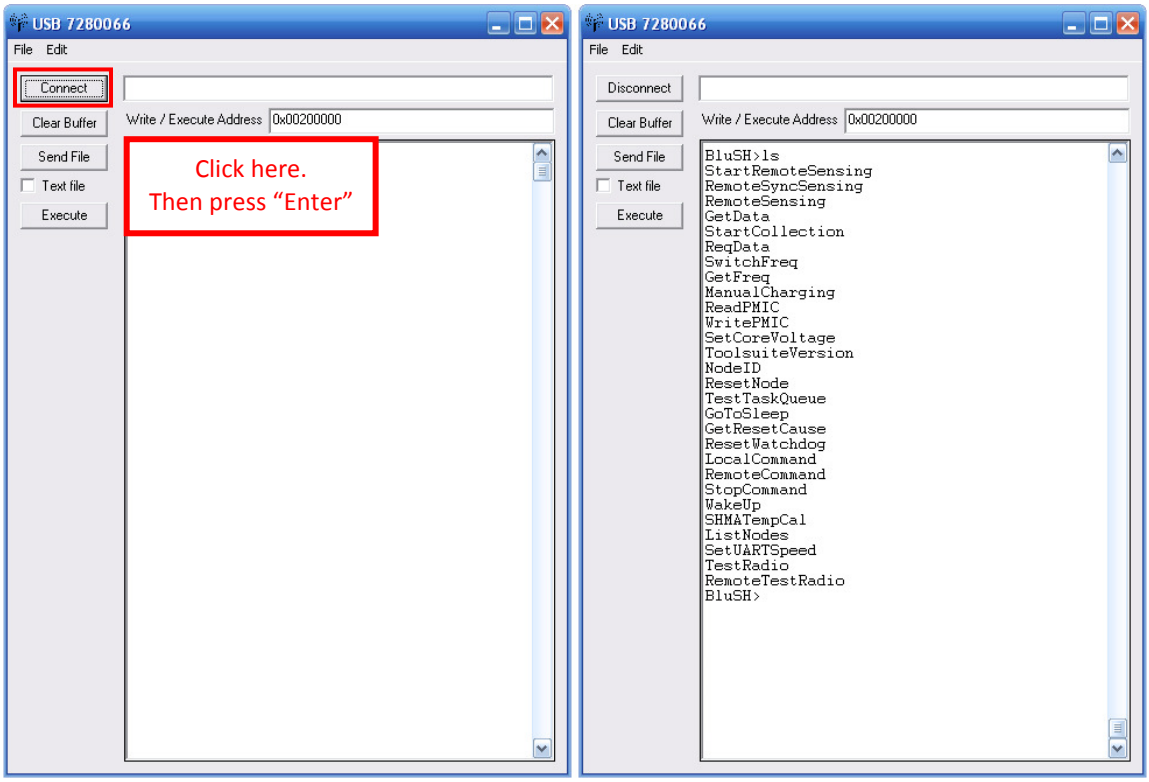
Step 9 - Communicating with the Imote2 via USB

The BluSH shell is a program that allows the user to interface with the Imote2. It allows you to run some simple commands on the Imote2 and to get brief output in text format. The BluSH shell is accessed using the `iMoteConsole` application.

- Plug a USB cable into the PC and directly into a turned off Imote2. Turn the Imote2 on.
- In a Cygwin window, run `iMoteConsole`.
- From the dropdown box select "USB xxx", where xxx is a series of letter and numbers.
- Click "View Buffer in Window" and a new window will appear.



- In the new window click Connect; then click on the big text area.
- Press "Enter" once or twice. You should see a "`BluSH>`" prompt where you can enter commands.
- Type `ls` and press "Enter" to see a list of available commands. Your list of available commands may vary depending on what is already installed on your Imote2.



- For example, you can run `NodeID` to print out the hexadecimal address of the Imote2 you are using.

NEVER EVER use the "Send File" or "Execute" buttons on the `iMoteConsole`! Doing so can potentially overwrite your Imote2's USB boot loader. As a result, you will not be able to program your Imote2 via USB. The Imote2 will still be programmable, but you will have to use a much more complicated J-Tag programming setup. The ISHMP does not have instructions on setting this alternative up once you have overwritten the USB loader.

Step 10 - Installing Imote2 Interface Board Drivers

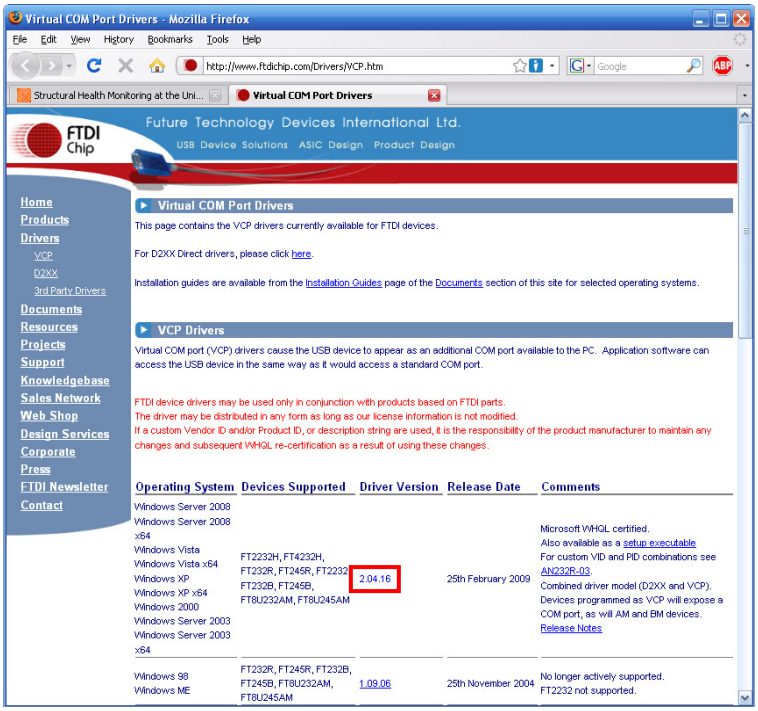
The IIB2400 interface board (http://www.xbow.com/Products/productdetails.aspx?sid=262), as shown below, is required to do this and the next step. If you do not have a IIB2400 interface board, you have finished the installation process.

The Imote2 interface board (IIB) contains a dual port FTDI chip, which provides two serial port (UART) interfaces over a USB connection to your PC. This makes interacting with the Imote2 much easier (and more portable to non-Windows platforms) than the method in the previous step.

10.1 *Install the FTDI Drivers*

In order to access the board, you must first install the FTDI Drivers. Windows does not automatically recognize the necessary drivers as it does with the Imote2. To install the drivers:

- Before connecting the IIB to your PC, download the drivers from the FTDI website (http://www.ftdichip.com/Drivers/VCP.htm).



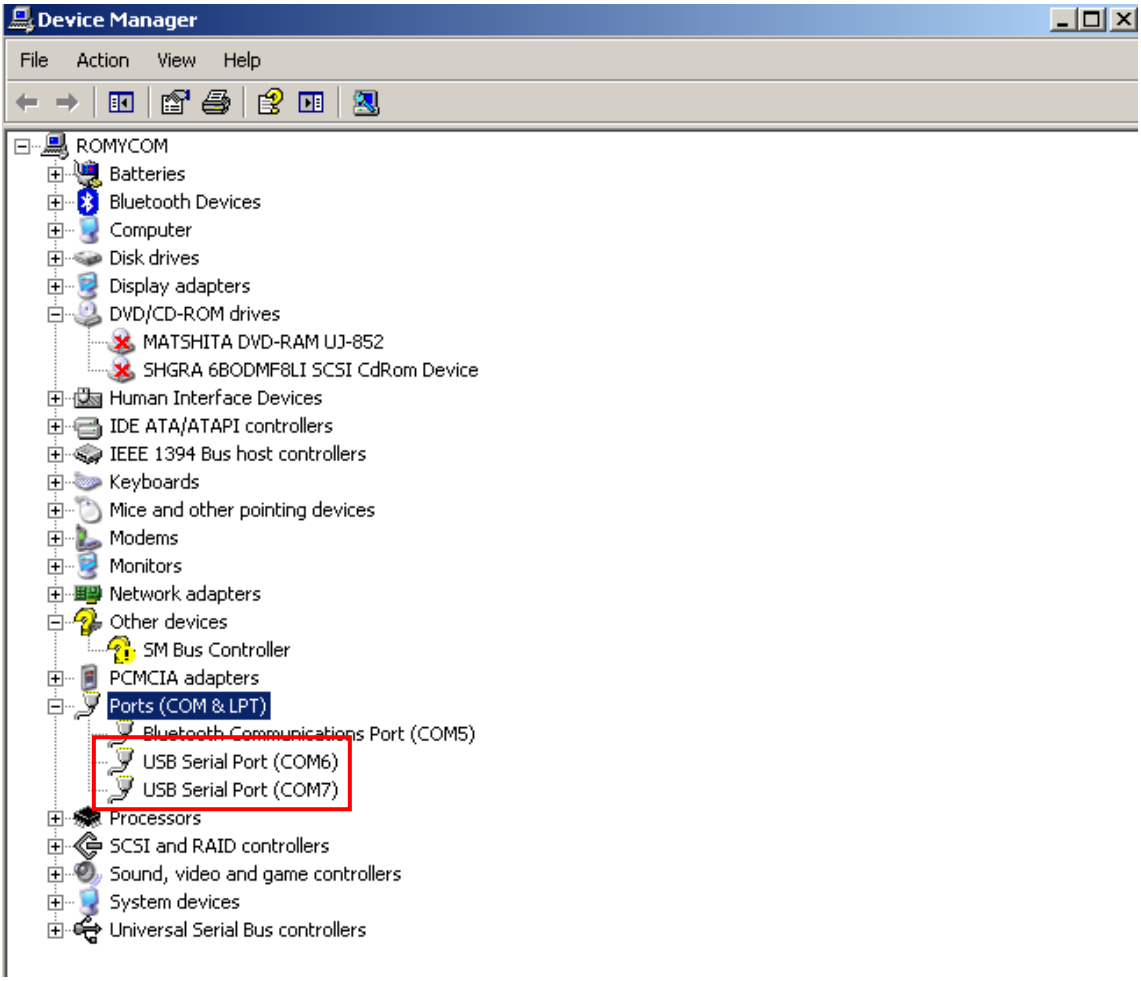
- Unzip the driver archive to a folder on your PC.
- Connect the IIB to your computer via a USB cable. When Windows prompts you for a driver:
    - Follow the installation instructions.

- When asked where to look for the driver, specify the folder where you just unzipped the archive.
- Two USB serial ports should be installed.

10.2 *Determine the IIB Ports*

Now that the drivers and ports have been installed, you need to determine the port numbers for each of the new ports:

- Open the Windows Control Panel. Select "System"; choose the "Hardware" tab; click on "Device Manager".
- Expand the "Ports (COM & LPT)" category. You should see something like:
  - USB Serial Port (COM6)
  - USB Serial Port (COM7)
- If you cannot see in Device Manager or access the COM ports after installing the drivers, reboot the computer.
- Write down these values (COM6 and COM7 in this example). This guide will refer to these later as "data UART" and "debug UART," respectively. You will also need to know these identifiers to interact with the Imote2.
- **Note**: if you have multiple IIBs, each new board you connect will create its own set of COM ports so keep track of which board is which!



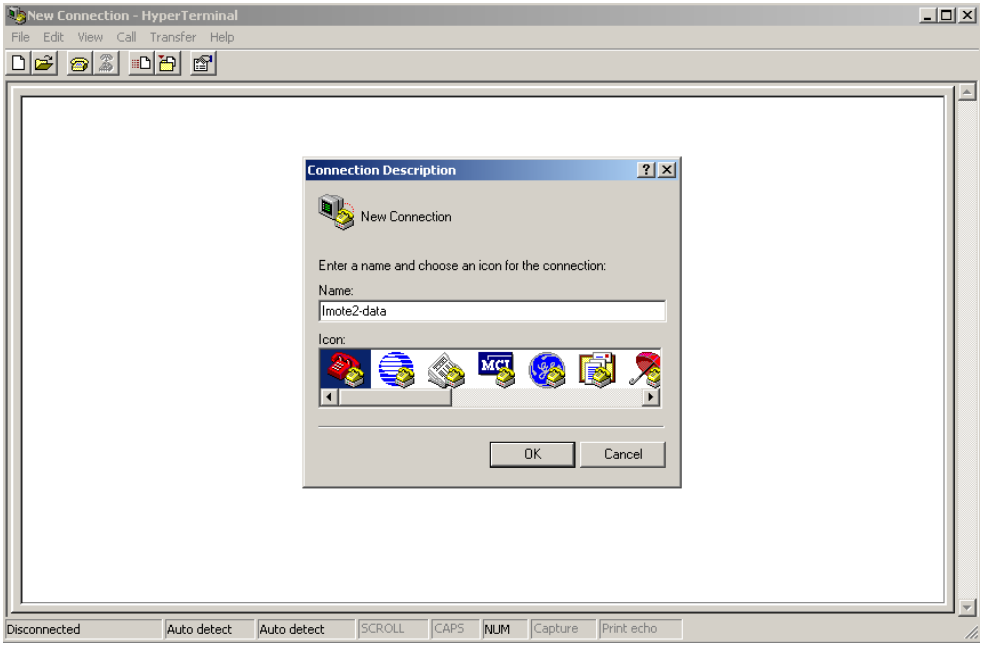
Step 11 - Communicating with the Imote2 via the IIB (Optional for Window XPs)

The following step is not necessary for using the Imote2 with the ISHMP Toolsuite that is available on the ISHMP website. However, if you will be creating additional applications not currently available in the ISHMP Toolsuite, creating the HyperTerminal connections in this step are helpful during the debugging process. However, Windows 7 no longer supports the Hyper Terminal. If you have Windows 7, you may skip this step now and complete it later with no problems if you decide you need the feature later.

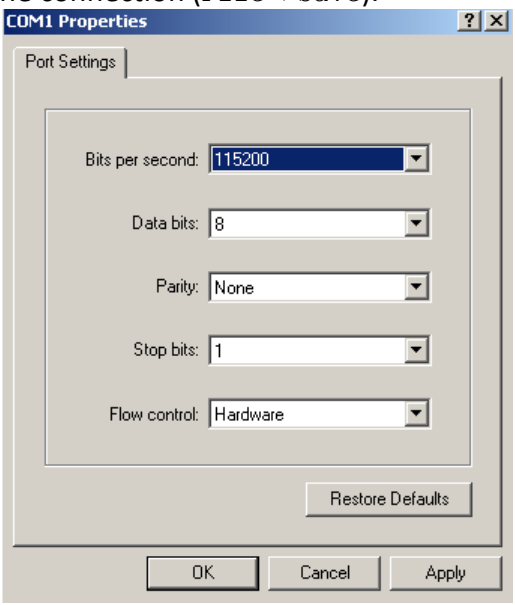
This step will help you to use Windows' HyperTerminal to connect to the Imote2 through the interface board:

- First you will need to connect the Imote2, IIB, and the computer.
  - Plug the turned-off Imote2 programmed with the Blink application into the IIB.
  - Connect the USB cable from your computer to the IIB (NOT directly to the Imote2 as you did for programming in Step 8).
  - Push the power button on the Imote2. The red LED should start to blink as before.

- Now, set up two HyperTerminal connections (*Note:* Windows 7 does not include this application by default.):
  - Go to `Start->Programs->Accessories->Communication->HyperTerminal`.



- Create a new connection named "Imote2-data":
  - Select the data UART port from Step 10.2.
  - For the port settings, enter: `921600, 8, None, 1, Hardware`.
  - Save the connection (`File->Save`).



- Create a new connection named "Imote2-debug":
  - Create a new connection (`File->New Connection`)
  - Select the debug UART port from Step 10.2.
  - For port settings enter: `115200, 8, None, 1, None`.
  - Save the connection (`File->Save`).
- Close HyperTerminal.

    - Now you should be able to start both connections by going to `Start->Programs->Accessories->Communication->HyperTerminal folder->Imote2-*.ht`.
- Finally, we can interface with the BluSH shell of the Imote2 through the debug UART by doing the following:
    - Open `Imote2-debug.ht` in HyperTerminal.
    - Press "Enter" once or twice. You should see a "`BluSH>`" prompt.
- The data UART can be accessed in a similar manner and allows applications to transfer data back and forth between the PC and the Imote2. It is only available through the interface board and not if the Imote2 is connected directly to the PC via the USB. However, note that the data UART is not used by the Blink application and will therefore not show any data output.

Step 12 - Installing the ISHMP Toolsuite

The Illinois SHM Project maintains the latest software version of its Toolsuite on its website (http://shm.cs.uiuc.edu/software.html). If you are installing a newer version of the Toolsuite you should make sure you replace the entire SHM tree and not just individual files. Both first time installers and version updaters should do the following:

- Download the latest version of the Toolsuite from the Illinois Structural Health Monitoring Project website (http://shm.cs.uiuc.edu/software.html).
- Extract the `shm-xxxx.zip` archive to a convenient location in Cygwin, for example the directory `/opt/shm`.

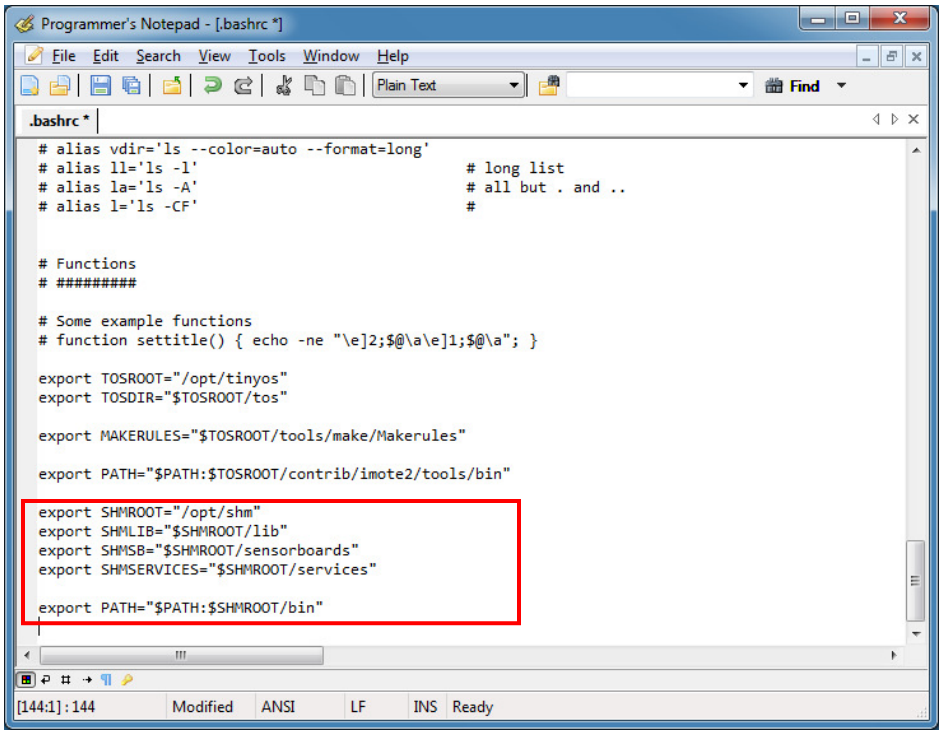
If you followed the guide so far, the `/opt/shm` directory will be located at `C:\tinyos\cygwin\opt\shm`. If you chose a different location during your installation, use that location in place of `/opt/shm` and `C:\tinyos\cygwin\opt\shm` for the remainder of this guide.

Step 13 - Setting up the ISHMP Toolsuite

Once the files have been placed in the SHM folder, you need to tell your Cygwin installation where they are located and then compile them so they are operational.

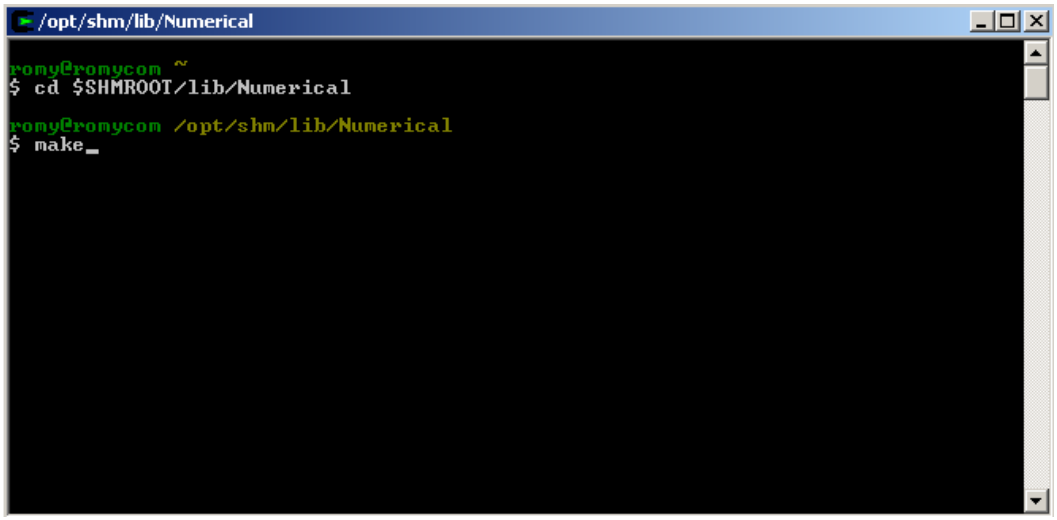
13.1 *Edit .bashrc File*

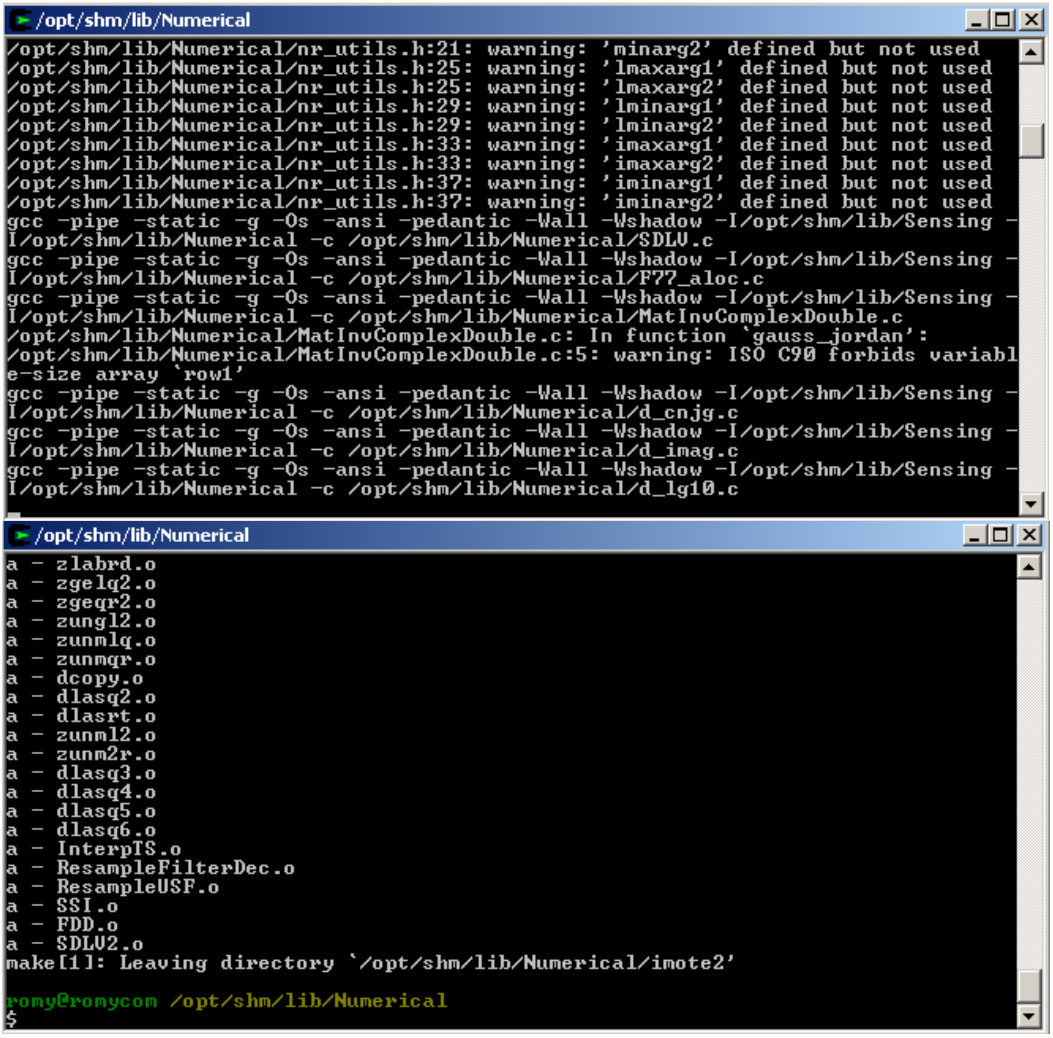
- Open your `~/.bashrc` file in `/home/username` directory with a text editor (see Step 1.5).
- Add the following environment variables to the end of that file (see screenshot on the next page):
    - `export SHMROOT="/opt/shm"`
    - `export SHMLIB="$SHMROOT/lib"`
    - `export SHMSB="$SHMROOT/sensorboards"`
    - `export SHMSERVICES="$SHMROOT/services"`
    - `export PATH="$PATH:$SHMROOT/bin"`
- Restart Cygwin.



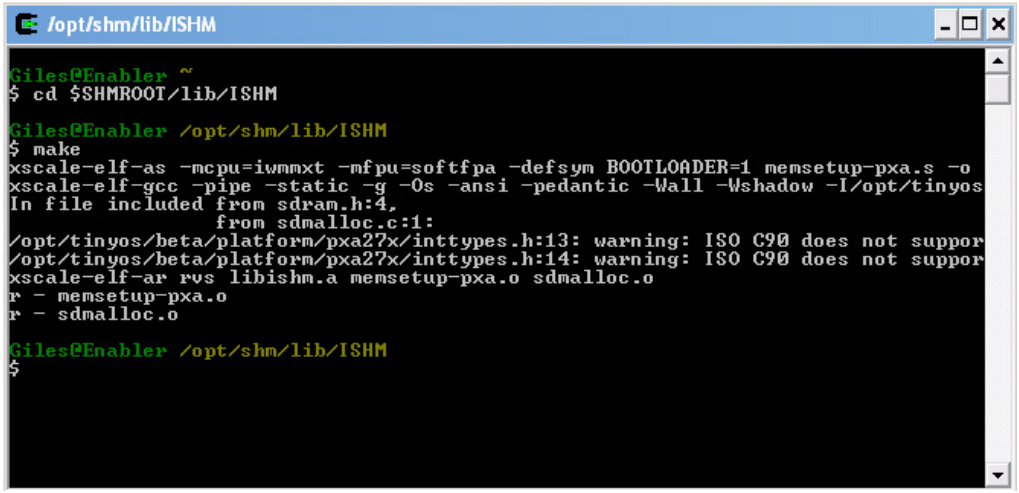
13.2 *Compile the ISHMP Libraries (optional)*

- *Note:* This step will be performed automatically for you the first time you compile an ISHMP application, or after you have modified any of the numerical library source files in `$SHMROOT/lib/Numerical`.
- Compile the numerical library for Imote2 and PC, ignoring any warnings during compilation:
  - `cd $SHMROOT/lib/Numerical`
  - `make`





- In order to enable memory allocation to the 32 MB SDRAM on the Imote2, the ISHM library needs to be compiled. Doing so will allow the sensors to record longer data records. Compile the ISHM library ignoring any warnings displayed during compilation:
  - `cd $SHMROOT/lib/ISHM`
  - `make`



Step 14 - Initializing Flash Constants

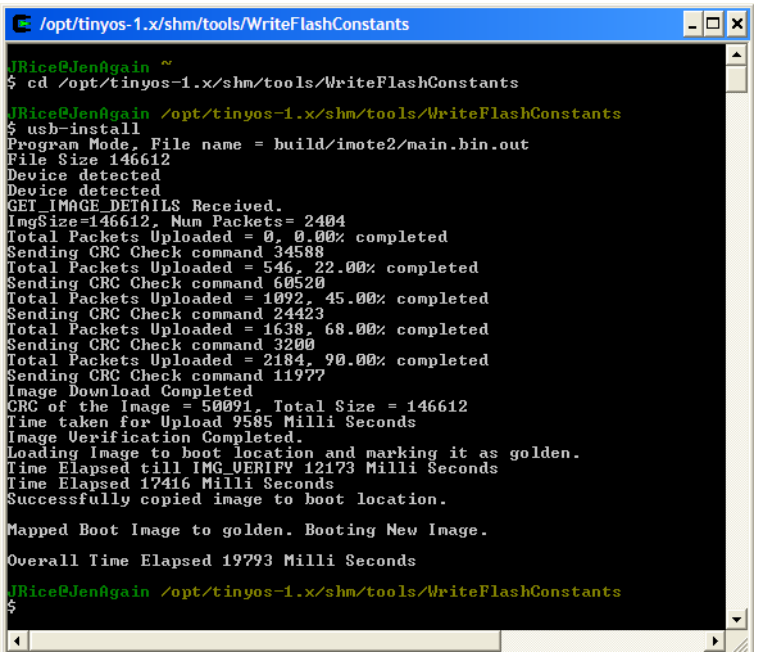
Many of the applications in the ISHMP Toolsuite utilize constants that are stored in Flash (non-volatile memory). Prior to using the ISHMP Toolsuite you must initialize each Imote2 with default Flash constants. You will not need to repeat this step unless you need to modify the Flash constants, or if they are updated in a future version of the Toolsuite.

Compiling and installing the *WriteFlashConstants* application to initialize the Flash constants will also confirm that the installation and configuration of the Illinois SHMP Toolsuite has been successful.

14.1 *Compile and Install WriteFlashConstants*

- Compile the application:
  - `cd $SHMROOT/tools/WriteFlashConstants`
  - `make imote2`

- *Note:* Unless you completed the optional Step 13.2, the first compilation process may take a long time and produce additional output. Subsequent compilations of Imote2 applications will be much quicker.
- Install the application on a mote:
  - Connect a USB cable from the PC directly to an Imote2.
  - Use the USB loader to install the application. The ISHMP Toolsuite provides an easier command for doing this. From the *WriteFlashConstants* application directory, run:
    - `make imote2 reinstall`
  - Repeat steps above to all imote2s.



14.2 *The WriteFlashConstants Utility*

Once installed, the *WriteFlashConstants* application will automatically check the constants loaded on the mode, and update them if they differ from the default ones. The LED on the Imote2 will turn red for a few seconds if the constants do not match. The constants will be written, and in a few seconds (during which time the LED will be blue/green) then the Imote2 will reboot. After the Imote2 reboots, the LED will show green, and it is ready to be programmed with other applications.

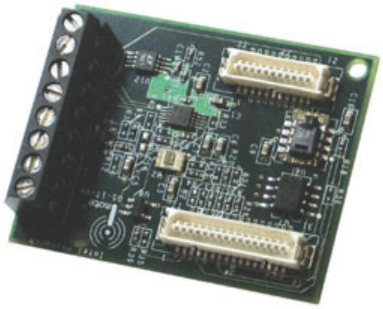
Step 15 - Compiling a Test Application

Compiling and installing a test application will confirm that the installation and configuration of the Illinois SHM Toolsuite has been successful. We will use the *RemoteSensing* application to take acceleration data using a sensor board.

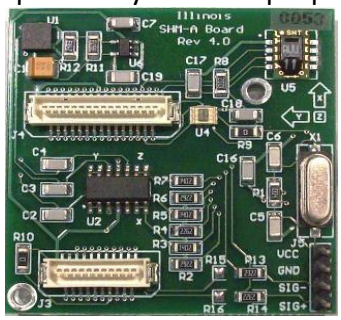
15.1 *Select a Sensor Board*

- The ISHM Toolsuite supports two different sensor boards:

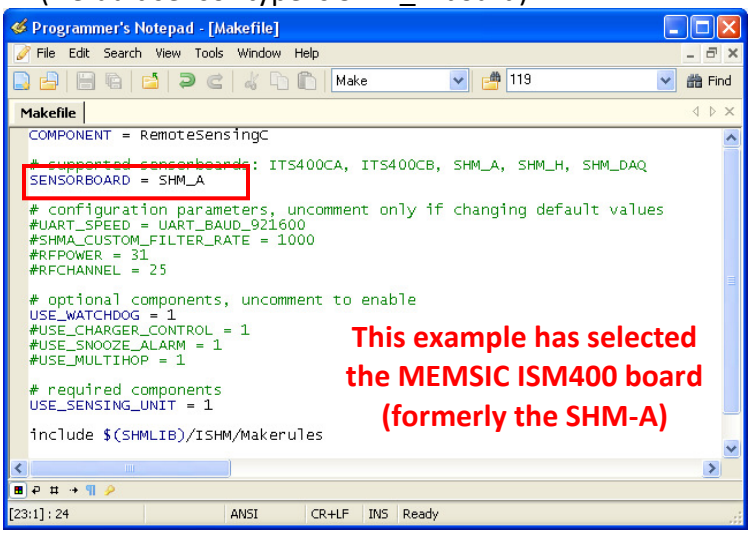
- Both versions of the Intel ITS400C sensor board (http://www.xbow.com/Products/productdetails.aspx?sid=261)



- The MEMSIC ISM400 sensor board (formerly the SHM-A) developed at the University of Illinois specifically for SHM purposes.

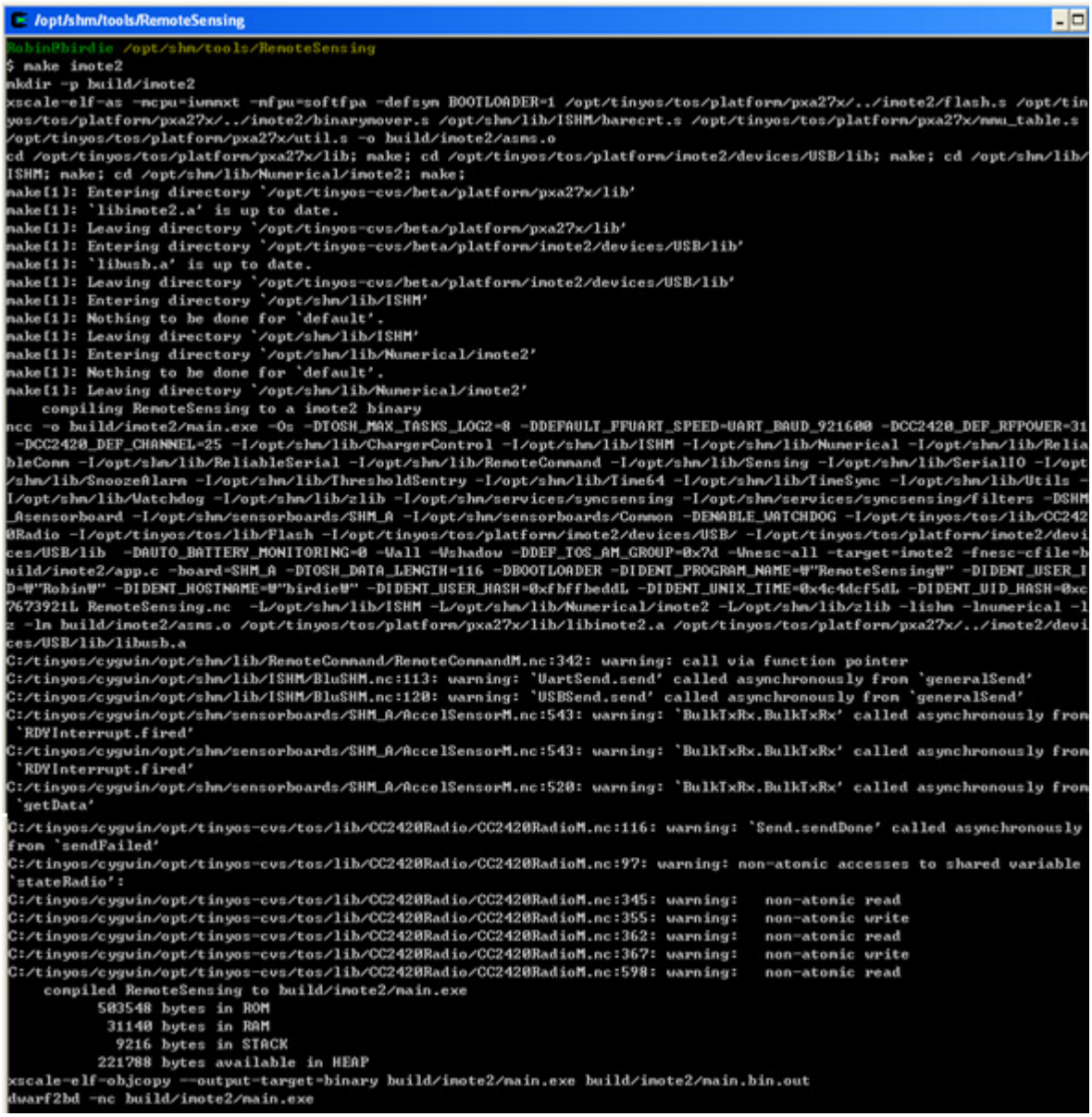


- Open `/opt/shm/tools/RemoteSensing/Makefile` with a text editor (see Step 1.5).
- Select which sensor board to use by typing one of the sensorboard identifiers at the "`SENSORBOARD = ...`" line.
  - Lines starting with `#` in the `Makefile` are considered to be comments and are ignored when the computer compiles the program.
  - For example, if we use the ITS400CA sensor board:
    - `SENSORBOARD = ITS400CA`
      (Default sensor type is SHM_A board)

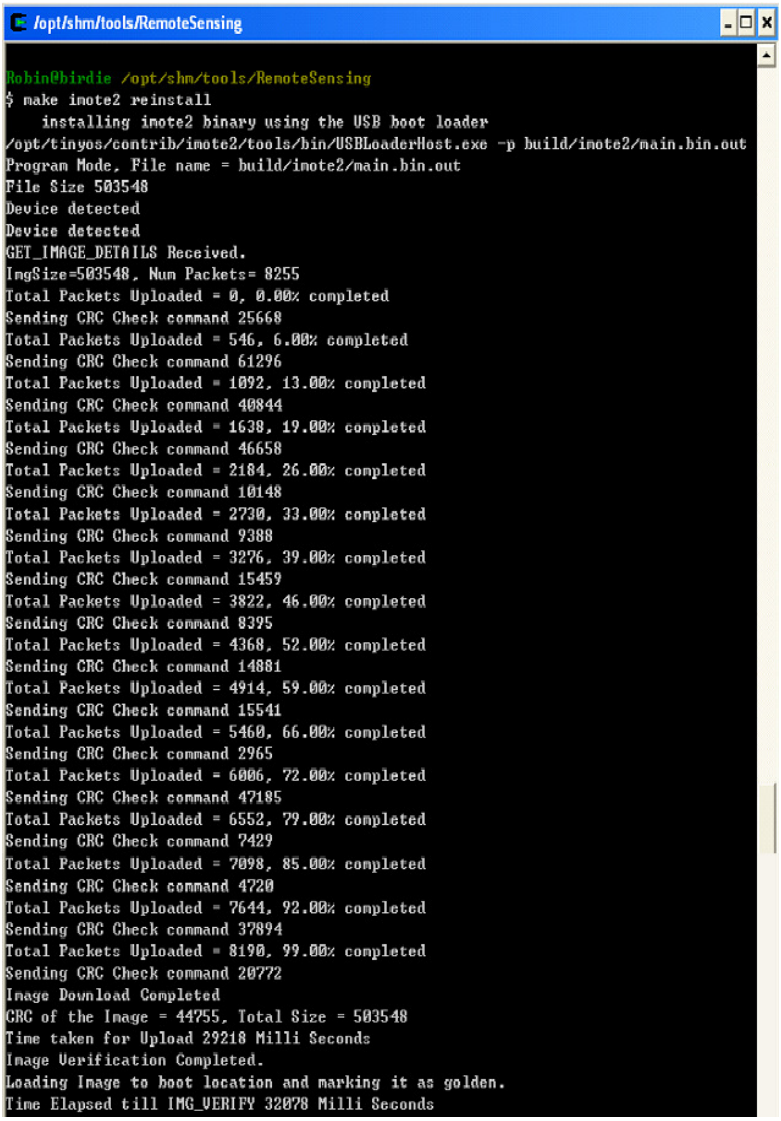


15.2 *Compile and Install RemoteSensing*

- Compile the application:
  - `cd $SHMROOT/tools/RemoteSensing`
  - `make imote2`



- Install the application on each mote:
  - Connect a USB cable from the PC directly to an Imote2, with the sensor board attached.
  - Use the USB loader to install the application. The ISHM Toolsuite provides an easier command for doing this. From the *RemoteSensing* application directory, run:
    - `make imote2 reinstall`
  - Repeat steps above to all imote2s.



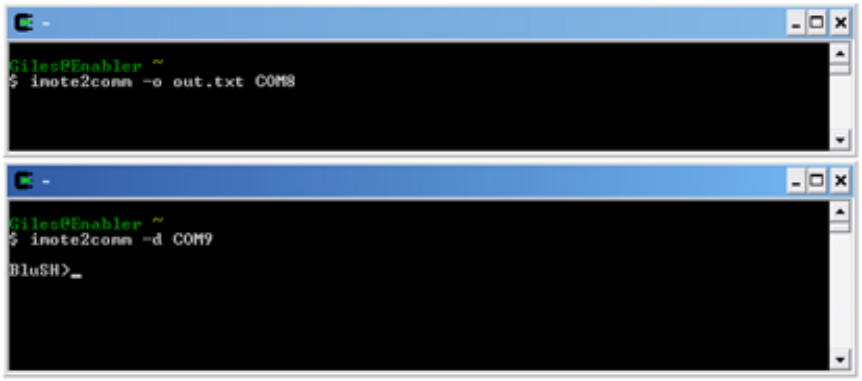
Step 16 - Running a test application on the Imote2

We will now use the `imote2comm` tool to interact with the *RemoteSensing* application running on the Imote2. The `imote2comm` tool is a command-line replacement for Windows HyperTerminal with a few extra features.

16.1 *Connect to the Imote2:*

- Prepare an imote2, a debug board and a usb cable to set a 'Gateway node'
- Prepare an imote2, a battery board and a sensor board to form one 'Leaf node'. You can prepare as many leaf node set as you want.
- Plug the Imote2 with the sensor board attached into an IIB2400 interface board (http://www.xbow.com/Products/productdetails.aspx?sid=262).
- Connect the USB cable from the PC to the interface board (NOT directly to the Imote2).

- Open two Cygwin windows.
- In the first Cygwin shell, run:
  - `imote2comm -o out.txt COMx`
    - "`-o out.txt`" writes any data received over the data UART to a text file called `out.txt`. A copy of the output will also be displayed in the Cygwin window. This parameter is optional.
    - `COMx` should be the data UART (replace "x" with the correct port number for this Imote2).
    - **Note:** Including the option "`-n`" before the "`-o`" will prevent the measured data from being printed on the screen. This will improve the data output speed considerably.
- In the second window, run:
  - `imote2comm -d COMy`
    - "`-d`" specifies that this is the debug UART.
    - `COMy` should be the debug UART (replace "y" with the correct port number for this Imote2).
  - Press <enter> a couple of times until you receive the "`BluSH>`" prompt.

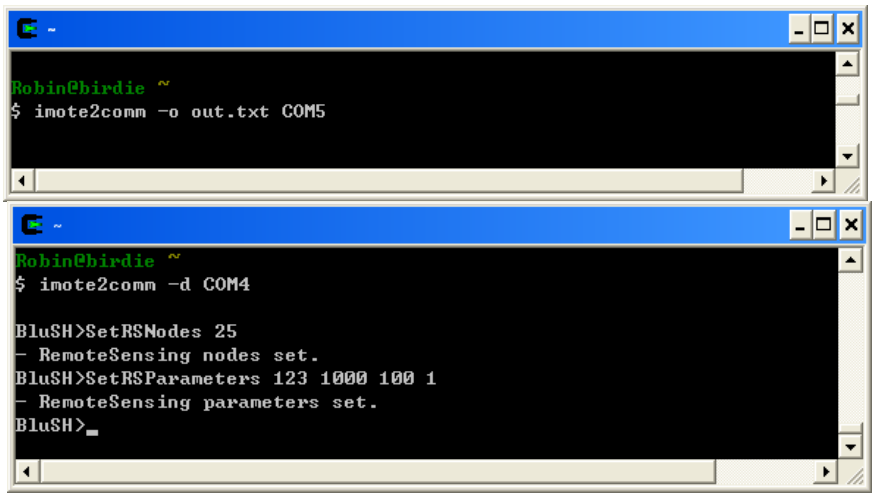


16.2 *Run the application:*

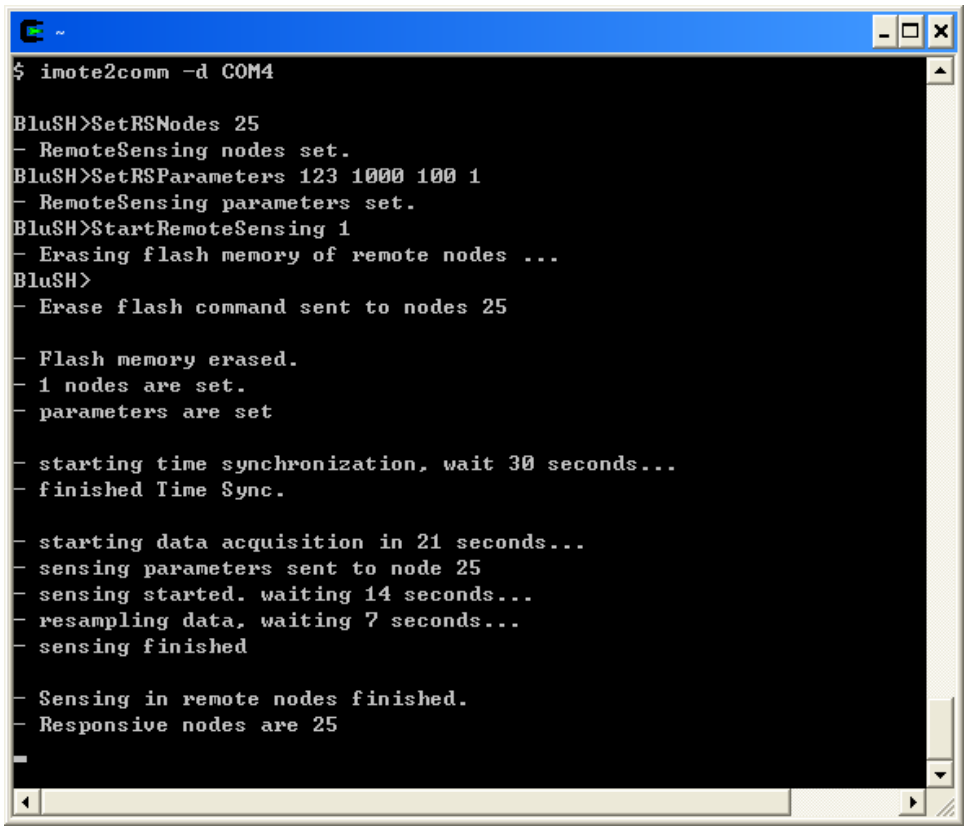
- Run `ls` to see a list of available commands. It should contain `SetRSNodes`, `SetRSParameters`, `StartRemoteSensing` and `RetrieveData` among others.
- Run `SetRSNodes <nodeId>` to initialize the `RemoteSensing` application.
- Run `SetRSParameters 123 1000 100 1`. This will set up the program to:

- Acquire data from channels 1, 2, and 3 (the accelerometer in the *x*, *y*, and *z* directions respectively). You can acquire any combination of channels by inputting the desired combination instead of `123`. For example, to get data from only channels 1 and 3, input `SetRSParameters 13 1000 100 1`.
- Requesting 1000 samples.
- Input 1 for network time synchronization. If you prepared multiple leafnodes, synchronization is recommended. Whereas for single leaf node setup, you can initiate network without time synchronization by replacing fourth input `1` to `0`. For example, `SetRSParameters 123 1000 100 0`.
- At the sampling rate of 100Hz. You may have to change this according to the sampling rates supported by the sensor board you are using.
  - ITS400CA: 280, 560, 1120, and 4480 Hz
  - ITS400CB: 40, 160, 640, and 2560 Hz
  - MEMSIC ISM400: 25, 50, 100, and 280 Hz

- No data collection has happened yet; these steps have just performed the setup. The program should return confirmation that the channels are prepared.

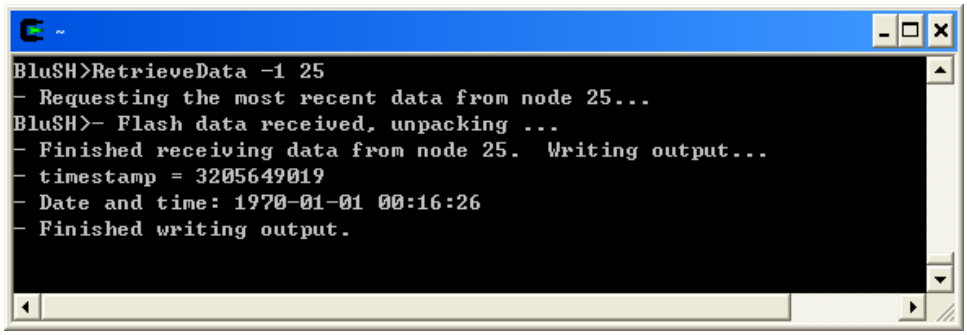


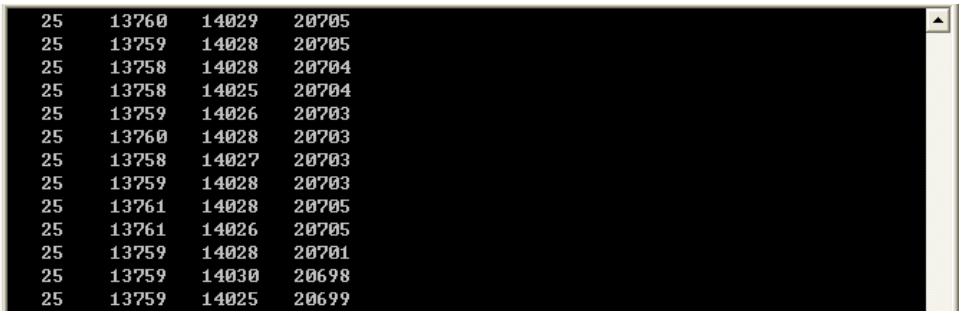
- Run `StartRemoteSensing 1` to actually start acquiring data.
  - Input `StartRemoteSensing 1` will erase the first memory block and store. Input `'0'` will store data in the next available block.
  - Some debug output will be printed in the first Cygwin window.
  - Actual sensing will start after you see "`Starting data acquisition in 21 seconds…` " line in the second Cygwin window.
  - Sensing is done when you see "`Responsive nodes are <nodeId>`"line in the second Cygwin window.



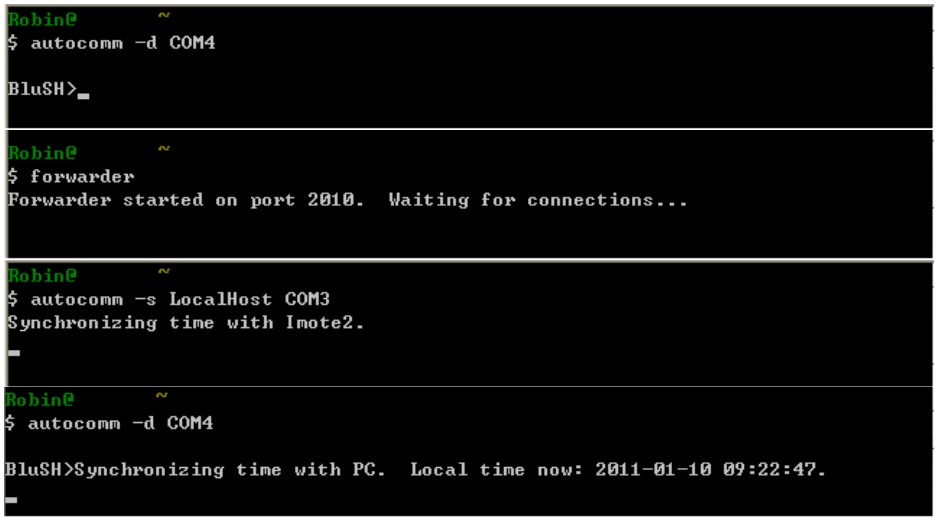
- Run `RetrieveData -1 <nodeId> [nodeId] [nodeId] …`
  - Input `-1` to retrieve the most recent data set.
  - Subsequent debug output may be delayed until all of the data is printed in the other Cygwin window. You may have to push `Enter` once in debug Cygwin window to get the `BluSH` prompt again.
  - Date and time will be different with your date and time on your computer.

  Note: If multiple data sets are stored in flash memory, they can be retrieved by replacing '-1' to a data set index (1 through 9). Note that the maximum number of data sets can be stored in flash memory is nine; if the tenth data set is collected with clearmetaData = 0, it will overwrite the first data set.



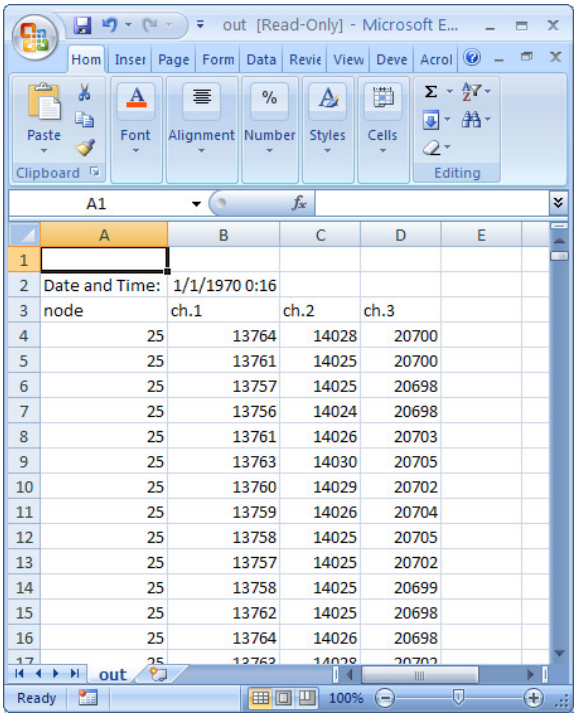


- Synchronizing base station computer time and the gate way (optional)
  - Collected data sets are stored persistently in flash memory on the leaf nodes. A leaf node can hold up to 9 sets of data block stored successively.
  - `Retrieve -1 [nodeId]` retrieves the latest data set, but the user also can retrieve older data sets by synchronizing the base station computer and the gateway.
  - Also, when a node fails to collect data in a network, and the gateway attempt to retrieve all the data set in a network, data may not be the synchronized flash data set. User can tell if the retrieved data sets are actually synchronized, by timestamps in output file
  - Open a new Cygwin window (keep "`autocomm -d COMy`" opened in earlier steps)
  - Run `forwarder`
    - Debug logs "`Forwarder started on port … Waiting for connections…`" will appear
  - Open another Cygwin window
  - Run "`autocomm -s LocalHost Comx`"
    - Debug log "`Synchoronizing time with imote2`" will appear
    - Synchronized time will appear in Debug window connected earlier with "`autocomm -d COMy`"
    - At the same time, a debug message "`New client connection established`. "will appear in the forwarder Cygwin window.



16.3 *Data Output*

- Press "Ctrl-C" in both Cygwin windows to close the `imote2comm` application.
- The output in "`out.txt`" located in your home directory will contain timestamps and sensor data in tab-separated columns that are suitable for importing into Excel or Matlab.
- The columns in the output file are:
  - `node` - the node address of the Imote2 sensor.
  - `ch.` *n* - sensor data for accelerometer channel *n* in floating point format.



node	ch.1	ch.2	ch.3
25	13764	14028	20700
25	13761	14025	20700
25	13757	14025	20698
25	13756	14024	20698
25	13761	14026	20703
25	13763	14030	20705
25	13760	14029	20702
25	13759	14026	20704
25	13758	14025	20705
25	13757	14025	20702
25	13758	14025	20699
25	13762	14025	20698
25	13764	14026	20698

Conclusion

Congratulations, if you followed the steps in this document you should now be able to compile and install programs on the Imote2, interact with most Imote2 TinyOS applications, and compile and use all the applications available in the ISHMP Toolsuite. To help you understand each application function in the ISHMP Toolsuite and what commands need to be executed to run each application, the Illinois SHM Project has included a readme.txt file in each application folder. To become familiar with the available applications, read through the readme.txt files in each application folder. To help you use the Toolsuite for SHM installations, the Illinois SHM Project has prepared an additional guide entitled Imote2 for Structural Health Monitoring: User's Guide. This user's guide is available at http://shm.cs.uiuc.edu/AdvancedUser.html on the ISHMP website

Software License

Unless otherwise noted, the files which comprise the Structural Health Monitoring software toolsuite (SHM) are subject to the following restrictions.

The SHM software is NOT in the public domain. However, it is freely available without fee for education, research, and non-profit purposes. By obtaining copies of this and other files that comprise the SHM software, you, the Licensee, agree to abide by the following conditions and understandings with respect to the copyrighted software:

1. The software is copyrighted in the name of the Board of Trustees of the University of Illinois (UI), and ownership of the software remains with the UI.
2. Permission to use, copy, and modify this software and its documentation for education, research, and non-profit purposes is hereby granted to the Licensee, provided that the copyright notice, the original author's names and unit identification, and this permission notice appear on all such copies, and that no charge be made for such copies. Any entity desiring permission to incorporate this software into commercial products should contact:

   Professor Gul A. Agha agha@cs.uiuc.edu
   University of Illinois at Urbana-Champaign
   Department of Computer Science
   2104 Siebel Center
   201 North Goodwin Avenue
   Urbana, Illinois 61801
   USA
3. Licensee may not use the name, logo, or any other symbol of the UI nor the names of any of its employees nor any adaptation thereof in advertising or publicity pertaining to the software without specific prior written approval of the UI.
4. THE UI MAKES NO REPRESENTATIONS ABOUT THE SUITABILITY OF THE SOFTWARE FOR ANY PURPOSE. IT IS PROVIDED "AS IS" WITHOUT EXPRESS OR IMPLIED WARRANTY.
5. The UI shall not be liable for any damages suffered by Licensee from the use of this software.
6. The software was developed under agreements between the UI and the Federal Government which entitles the Government to certain rights.

Developed by:

Open Systems Lab
University of Illinois at Urbana-Champaign
Department of Computer Science
201 North Goodwin Avenue
Urbana, IL 61801
http://osl.cs.uiuc.edu/

and

Smart Structures Technology Laboratory
University of Illinois at Urbana-Champaign
Department of Civil and Environmental Engineering
205 North Matthews Avenue
Urbana, IL 61801
http://sstl.cee.uiuc.edu/

Send comments to: agha@cs.uiuc.edu

Principal Investigators:
Gul A. Agha (agha@cs.uiuc.edu)
B.F. Spencer, Jr. (bfs@uiuc.edu)

This work was supported in part by:
NSF grants CMS 06-00433 and CNS 05-09321
NCASSR grant N00014-04-1-0562
DARPA grant F33615-01C-1907

For inquiries, please contact:

Professor B.F. Spencer, Jr.
bfs@illinois.edu
University of Illinois at Urbana-Champaign
Department of Civil and Environmental Engineering
2213 Newmark Civil Engineering Laboratory, MC-250
205 North Mathews Ave
Urbana, IL 61801
USA

Or visit:

http://vibration.shef.ac.uk/imote2_forum
http://shm.cs.uiuc.edu