

# Simple Tutorial for Imote2 on selected topics

January 2007

In this tutorial you can find some of the problems that you might face while starting with Imote2 and the solutions that I have come up with. It is a simple beginner's tutorial on some selected topics. Note that Imote2s are compatible with the regular TinyOS environment and except some differences they are configured and programmed in the same way.

Good luck...

Panos Kampanakis  
Graduate Student - Cyber Defense Lab  
North Carolina State Univ.  
email: pan\_kamp@ncsu.edu  
url: <http://www4.ncsu.edu/~ptkampan>

## Table of Contents

How to install and run a simple program on Imote2 in TinyOS.....	3
How to print data back from the sensor to the serial port for debugging .....	5
How to send data back from the sensor to the serial port for debugging using the Serial Forwarder.....	6
How to get the time in an Imote2.....	7
How to change the core voltage and frequency of the Imote2 .....	8

## How to install and run a simple program on Imote2 in TinyOS

The following instructions are mostly part of the instructions provided in a manual titled "Install Software for Programming Imote2" in 27 January 2006 by Jianping Wang. There is also Intel Corporation Imote2 group's manual which is posted by Lama Nachman. You can download these instructions too; they are very helpful and more explanatory.

### PRODUCTS USED:

I am just including exactly the set of steps that I had to take to make my system work in simple wording. I had an Optiplex Dell computer with A7 BIOS, although this doesn't make much of a difference. I was using WindowsXP SP2 which also can't be very significant if someone else is using other Windows systems. For the installation I used the products that came with the Intel's Development Kit. I used the JTAG dongle, a bus connecting the dongle with the debug board, the debug board and a cable (included in the Kit) connecting it to a computer via USB port. Also, I attached an Imote2 to the debug board and connected the dongle to my computer using a parallel cable.

### BEFORE STARTING:

Before starting please make sure that you get into your systems BIOS and set the parallel port to be in ECP mode, with address 0x378 and DMA off.

You have to make a directory called nordheim in drive C:\

have to make a directory called imote2 in drive C:\

### THE PROCEDURE:

- First you need to install TinyOS. This is done very easily in Windows. The TinyOS webpage <http://www.tinyos.net/download.html#snapshot> provides all the necessary instructions. What you really have to do is to download and install TinyOS 1.1.11 or 1.1.0 as described in [http://www.tinyos.net/windows-1\\_1\\_0.html](http://www.tinyos.net/windows-1_1_0.html) and then upgrade to TinyOS 1.1.15 by downloading an rpm file as described in <http://www.tinyos.net/dist-1.1.0/snapshot-1.1.15Dec2005cvs/doc/install-snapshots.html>. You are done, you have a working TinyOS installed in your Windows system, with Cygwin and everything installed.

- Now that you have TinyOS or even if you already had it, you have to update to the latest CVS version. This is done simply by making sure that you have a functional fast Internet connection and opening a Cygwin prompt. There you write

```
$ cvs -  
d:pserver:anonymous@tinyos.cvs.sourceforge.net:/cvsroot/tinyos  
login
```

When it asks for PASSWORD you press [enter]

```
$ cvs -z3 -  
d:pserver:anonymous@tinyos.cvs.sourceforge.net:/cvsroot/tinyos  
co -P tinyos-1.x
```

Now, let it update to the latest CVS. **Note** that it will take some time.

- Now, you can update the nesC (though, it might not be necessary). Go to <http://www.tinyos.net/dist-1.1.0/tinyos/windows/> and download the latest nesC .rpm file. To install it you just have to go to a Cygwin prompt and do

```
$ rpm --ignoreos -ivh *.rpm
```

- Then you have to do three simple things in Cygwin

```
$ cd /opt/tinyos-1.x/tos/platform  
$ ln -s /opt/tinyos-1.x/beta/platform/imote2 imote2  
$ ln -s /opt/tinyos-1.x/beta/platform/pxa27x pxa27x
```

- Then you have to add the following line into the file /etc/profile:

```
export MAKERULES=$TOSDIR/../tools/make/Makerules
```

- Now we have to download the Wasabi tools. Go to

[http://www.intel.com/design/intelxscale/dev\\_tools/031121/](http://www.intel.com/design/intelxscale/dev_tools/031121/), press Agree and download the ones for Windows. Decompress the zip file into a new directory wasabi that is in the same directory with /opt.

Add the following line into the file /etc/profile:

```
PATH= /wasabi/usr/local/bin:$PATH
```

- Now we need the Software Development Tool Suite for Intel XScale Microarchitecture. Go to <http://www.intel.com/cd/software/products/asm-na/eng/225396.htm> and press the [Free Evaluation Software](#) link. Then follow what it is asked and download it. **Note** that it is only for 30days evaluation version. Receive the email with the serial number and license and then install it. It is supposed to ask for the license etc during the installation but mine didn't ask for any while I was installing it in NCSU (strange!). The files are usually installed in C:\Program Files\Intel\SDT2.0.1

- Following download the configuration files (sent by Lama) from

<http://www.cs.virginia.edu/~jwang/> > Working Materials > download imote2\_install.zip

Unzip it and execute the following tasks in Cygwin:

\* copy imote2.fcf and imote2.xsf from the imote2\_install.zip to /c/imote2 that you created in the beginning

```
* $ cd /c/nordheim
```

```
* $ ln -s /c/Program\ Files/Intel/SDT2.0.1/xflash xflash
```

\* Copy boards.ini (rename boards.in from the zipped file to boards.ini) and imote2.xdb into directory /c/Program\ Files/Intel/SDT2.0.1/xflash.

**Note** that I was using the Intel JTAG Dongle provided in the Kit, so I also had to do the following: add the line

```
export PXA27X_JTAG_DEV=""INTEL(R) JTAG CABLE""
```

in the file /etc/profile. If you are using Macraigor Raven JTAG dongle, it is supposed to work without doing anything.

- Finally, we are ready to test the Blink program and see how it works. Connect the debug board with the dongle using the bus. Connect your computer with the dongle using a parallel cable and the debug board with the computer via USB using one of the cables in the Kit. Also, attach an Imote2 on the board. Be careful, the switch SW3 on the sensor should be set to MOTE, not CDLP. Press the Reset button on the corner of the sensor.

Then open a Cygwin prompt and do

```
$ cd /opt/tinyos-1.x/apps/Blink
```

```
$ make install imote2 debug
```

It will take a little but hopefully your sensor will be blinking in the end. Just to verify that it is working you can change the color of the LED in the BlinkM.nc file to green, and reload the Blink program to the sensor. If it is blinking green you are done.

**Note** that there will be some warnings while compiling the Blink program, inexplicable though. I have been told that if you try TinyOS 2.0 there are no warnings.

## How to print data back from the sensor to the serial port for debugging

Most of the times when you are writing code for sensors, or whenever you write code in general, you want to be able to follow the execution of the program and be able to debug it. In order to do so, you can use the trace command in order to printout to the serial port some values and some messages from inside your code.

For example, in the Blink program described above, if you wan to print a value temp whenever the LED blinks you can add the line

```
trace(DBG_USR1, "My temp is %d\n\r",t) ;
```

right after the LED blinking (line “call Leds.redToggle();”) is called.

Then you can install the Blink program again on your mote following the procedure described earlier.

If you have installed the programming board successfully, it will create two “USB serial port” instances, you can find which COM ports were mapped by going to Control Panel > System> Hardware> Device Manager. The ports will be listed under “Ports (COM & LPT)”. If you don’t see these listed, then the USB drivers aren’t installed properly. Try reconnecting the debug board on your computer (note that I didn’t need to install any additional drivers for WindowsXP SP2).

The second USB serial port listed is the one dedicated for the console and will be used for receiving the debugging messages. A simple demonstration of how this is done can be seen in [my BlinkI](#) application.

Then, you can use any HyperTerminal Application, like Terminal.exe in order to make sure that data is sent to the serial port. Start the program. Choose the second com port listed and configure it to 115200, 8, none, 1, none. Press “Connect” and you will be able to see the debugging message you defined above in the Blink program.

## How to send data back from the sensor to the serial port for debugging using the Serial Forwarder

In Imote2s, it is also possible to send messages to the serial port using the SendMsg interface. It is done exactly the same way as in the micazs in TinyOS. The difference is that when compiling and installing the application for the Imote2 you do not define the port to which the messages will be sent. Though, they will be by default sent to the first of the two COM ports mapped to the USB connected to the debug board.

The call

```
call Mesg.send(TOS_UART_ADDR, sizeof(mssg), &report);
```

is the one that will be used for sending the packet. A simple example of how this is done can be seen in [my BlinkI](#) application.

**Note** that, after you send data to the port you can use the serial forwarder to forward the data to another port and then receive and process them from this port using a customized program. Instructions of how to configure and use the serial forwarder can be found in [TinyOS tutorial](#) and An Liu's TinyECC [README](#) file.

Though, for Imote2 you have to always remember that the baudrate is 115200 and the serial port is the first COM port of the two mapped to the debug board. Also, there is a change that has to be performed in `/opt/tiny-1.x/tools/java/net/tinyos/platforms.properties` file. You have to add the line

```
imote2=micaz,5,115200
```

because micaz and Imote2 are using the same modules for sending data to the serial port.

Then you can run the Serial Forwarder java tools with the baudrate mentioned and forward some data.

## How to get the time in an Imote2

In Imote2s, sometimes we need to get the time at a certain point of our application execution and compare it with the time at another point. In micazs this was done using the SysTime interface.

In Imote2, it is done very similarly using the interface SysTime64 located in /opt/tinyos-1.x/ beta/platform/pxa27x/ and calling the function

```
call SysTime64.getTime32();
```

A simple demonstration of how this is done can be seen in [my BlinkI](#) application.

## How to change the core voltage and frequency of the Imote2

Imote2s are supposed to have a frequency of 13MHz-416MHz, maybe up to 520MHz. So there must be a way to change it. Intel claims that you can use the Blush command prompt to change it, but currently I couldn't get it to work. So, I tried to do it using the interface provided for Imote2. **Note** that at the time that this manual was compiled they supported only 13MHz (default) and 104MHz.

The frequency can be set using the interface DVFS that is located in the `/opt/tinyos-1.x/beta/platform/imote2`. In order to do so you have to call the function

```
call DVFS.SwitchCoreFreq(x,x);
```

where x is the frequency you want to set in MHz.

Though, you have to remember that if for example you want to set the frequency to 104MHz the core voltage has to be changed to 950mV too. In order to change it, you will use the interface PMIC that is in `/opt/tinyos-1.x/beta/platform/imote2`. You have to call

```
call PMIC.setCoreVoltage(y);
```

where y is the voltage in mV, starting from 850mV in increments of 25.

Hopefully, Intel will support other frequencies in the future too. A simple demonstration of how to implement the above can be found in [my BlinkI](#) application.