IRISWSNLAB

IRIS MOTE MODULE KIT PROJECT

Embedded Systems Project

Agnese Salutari, agnese.salutari@student.univaq.it

IRISWSNLAB PROJECT OVERVIEW

In IrisWSNLab Project, I performed some of the tutorials proposed by TinyOS at the following link:

http://tinyos.stanford.edu/tinyos-wiki/index.php/Main_Page

- 1.1
- 1.2
- 1.3
- 1.4
- 1.5
- 1.6
- 1.7
- 1.8
- 1.11
- 1.13
- 1.15

1.5.1 and 1.9 pages have been removed. I'm going to talk about useful commands and problems/solutions I found during these tutorials (other material and explanations are part of the tutorials themselves).

ENVIRONMENT

In addition to the Iris Kit (WSN hardware modules), I used:

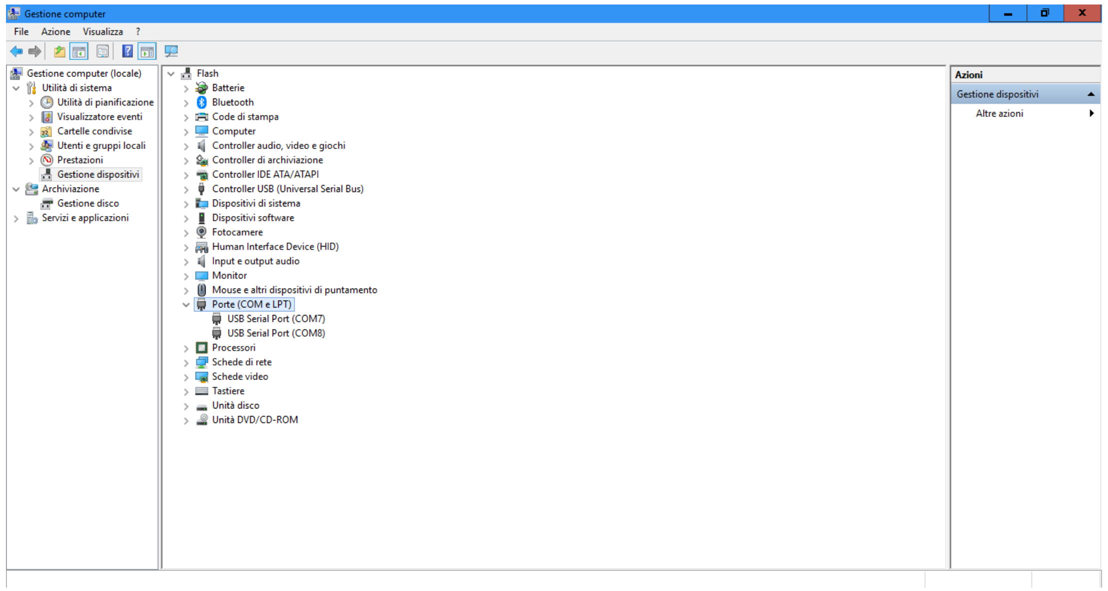
- A Debian 7 virtual machine (running on Windows 10) provided with TinyOS 2.1.2 and tutorials code (you can find it at tutorials link).
- VM Ware Workstation Player, a Virtualization Environment (to run the virtual machine):
  - https://www.vmware.com
- MIB520 drivers for Windows:
  - http://www.ftdichip.com/Drivers/VCP.htm

MIB520 DRIVERS

To install MIB520 drivers on Windows 10:

- I plugged the MIB520 programming board (the only one provided with a USB port) to one of my PC USB ports:

  As you can see in Device Management, the device is detected as 2 COM ports.
- I download the executable file from the link (Windows denied me doing the manual installation) and I run it.



Gestione computer (locale)
- Utilità di sistema
  - Utilità di pianificazione
  - Visualizzatore eventi
  - Cartelle condivise
  - Utenti e gruppi locali
  - Prestazioni
  - Gestione dispositivi
- Archiviazione
  - Gestione disco
- Servizi e applicazioni

Flash
- Batterie
- Bluetooth
- Code di stampa
- Computer
- Controller audio, video e giochi
- Controller di archiviazione
- Controller IDE ATA/ATAPI
- Controller USB (Universal Serial Bus)
- Dispositivi di sistema
- Dispositivi software
- Fotocamere
- Human Interface Device (HID)
- Input e output audio
- Monitor
- Mouse e altri dispositivi di puntamento
- Porte (COM e LPT)
  - USB Serial Port (COM7)
  - USB Serial Port (COM8)
- Processori
- Schede di rete
- Schede video
- Tastiere
- Unità disco
- Unità DVD/CD-ROM

TINYOS COMMANDS – COMPILING AND PROGRAMMING

To compile an application for an Iris Mote (a WSN sensor node), go inside the project folder and give the following command:

- make iris

To compile and program an Iris Mote, plug it to the programming board, connect the board to the PC, then go inside the project folder and give the following command:

- make iris install.[<moteID>] mib510,/dev/ttyUSBx

  Mote ID is the unique identifier you can assign to the mote (or it will be provided with the default ID).

  x is the USB port number the programming board is connected to.

To directly program an Iris Mote (without compiling it), the command is instead:

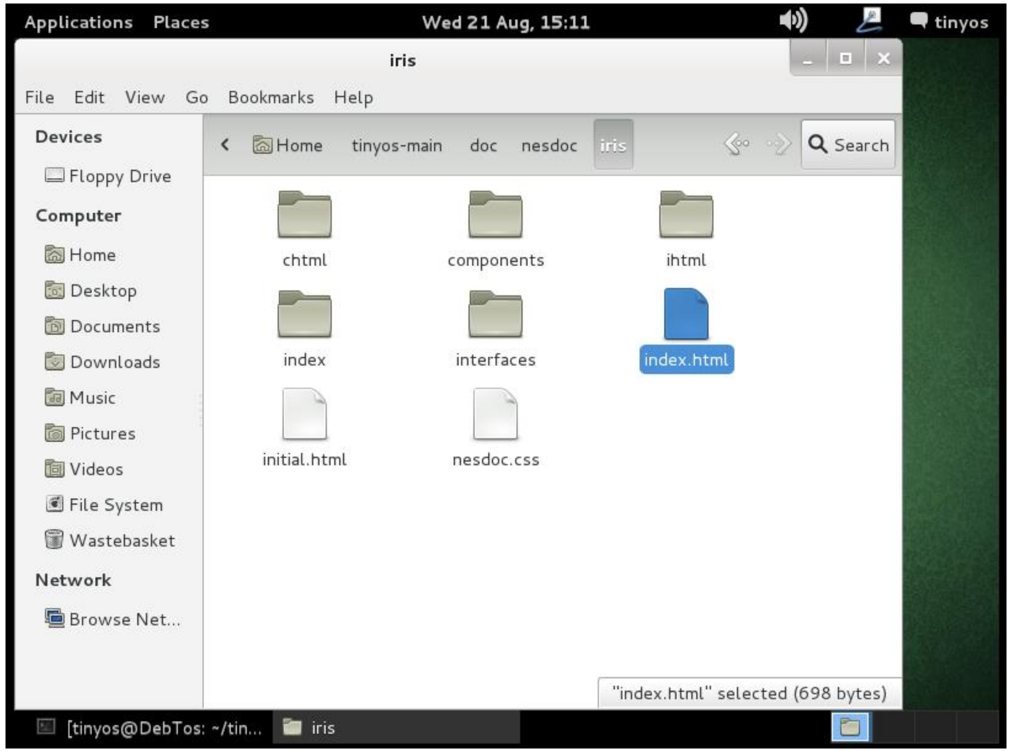
- make iris reinstall[.<moteID>] mib510,/dev/ttyUSBx

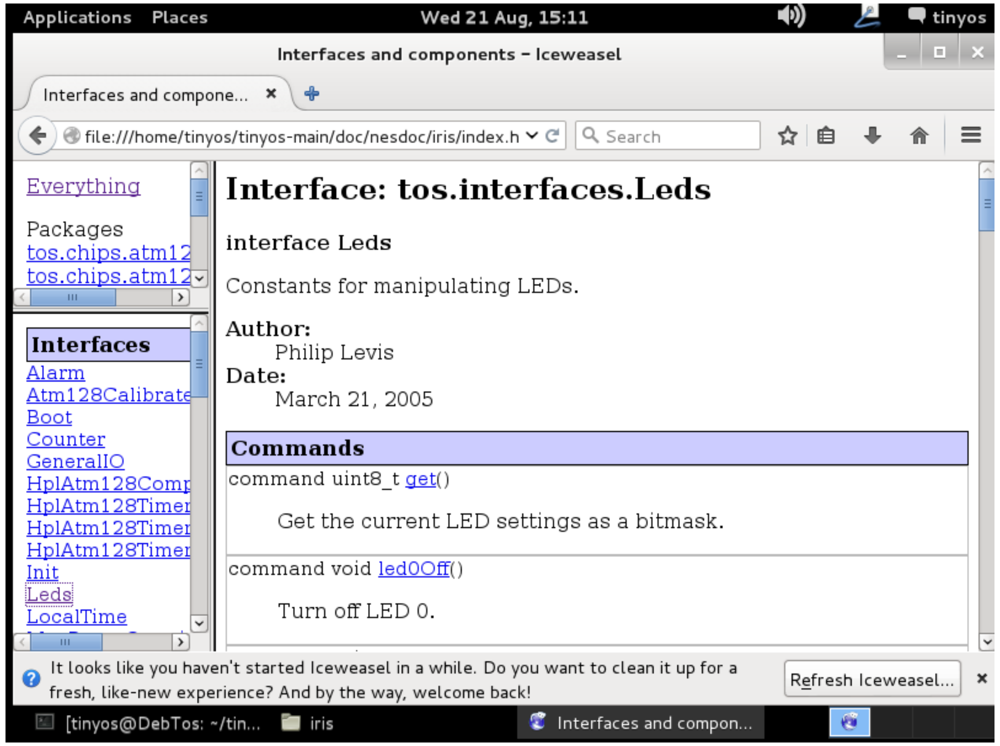
TINYOS COMMANDS – REVERSE ENGINEERING

To see how your components are made and wired together, you can use reverse engineering by the command:

- make iris docs

The result is an html file (inside doc folder).





Interface: tos.interfaces.Leds

interface Leds

Constants for manipulating LEDs.

Author:
Philip Levis
Date:
March 21, 2005

Commands

command uint8_t get()

Get the current LED settings as a bitmask.

command void led0Off()

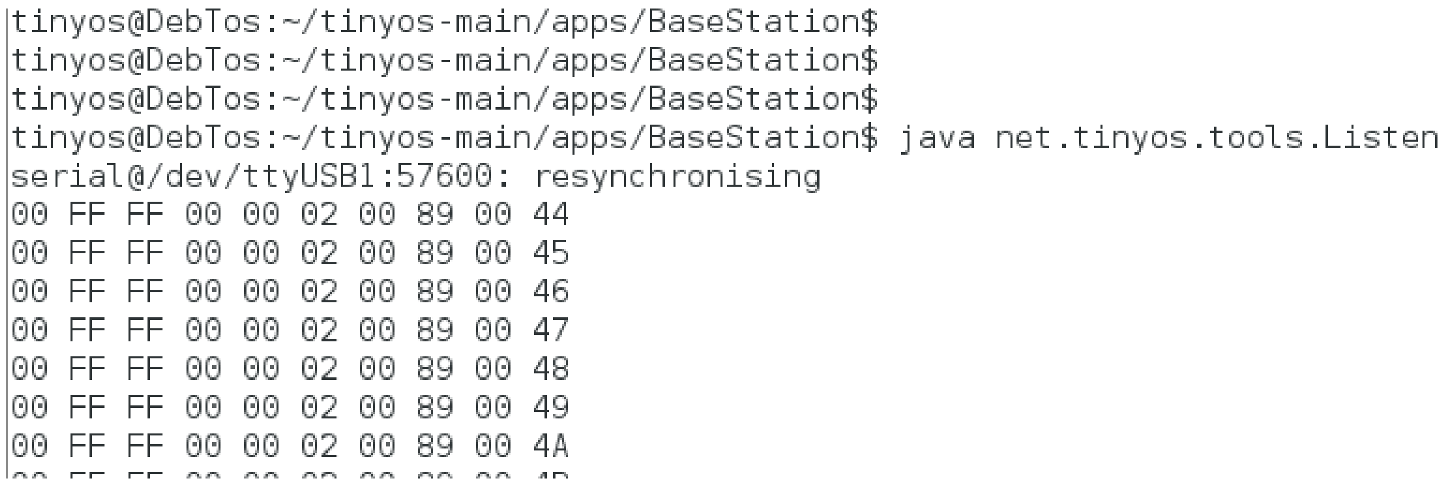
Turn off LED 0.

It looks like you haven't started Iceweasel in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

TINYOS COMMANDS – LISTENING TO A SERIAL PORT

If you installed an application that sends data to the PC on one ore more motes and the BaseStation app (that receives data) on the mote connected to the PC, you can see these data by giving the following command:

- java net.tinyos.tools.Listen -comm serial@/dev/ttyUSBx:iris



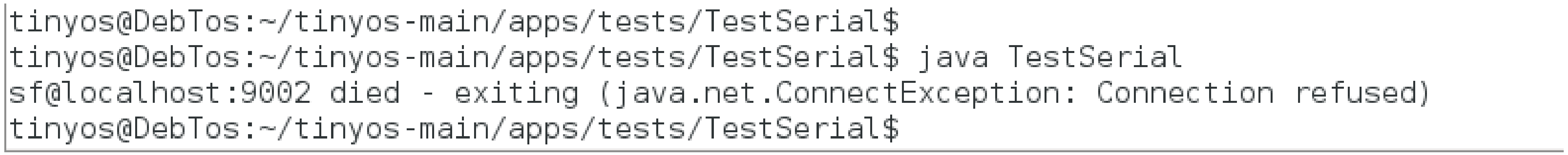
```
tinyos@DebTos:~/tinyos-main/apps/BaseStation$
tinyos@DebTos:~/tinyos-main/apps/BaseStation$
tinyos@DebTos:~/tinyos-main/apps/BaseStation$
tinyos@DebTos:~/tinyos-main/apps/BaseStation$ java net.tinyos.tools.Listen
serial@/dev/ttyUSB1:57600: resynchronising
00 FF FF 00 00 02 00 89 00 44
00 FF FF 00 00 02 00 89 00 45
00 FF FF 00 00 02 00 89 00 46
00 FF FF 00 00 02 00 89 00 47
00 FF FF 00 00 02 00 89 00 48
00 FF FF 00 00 02 00 89 00 49
00 FF FF 00 00 02 00 89 00 4A
```

To allow more than one application to listen to the motes, you have to use the Serial Forwarder:

- java net.tinyos.sf.SerialForwarder -comm serial@/dev/ttyUSBx:iris

TINYOS COMMANDS – LISTENING TO A SERIAL PORT

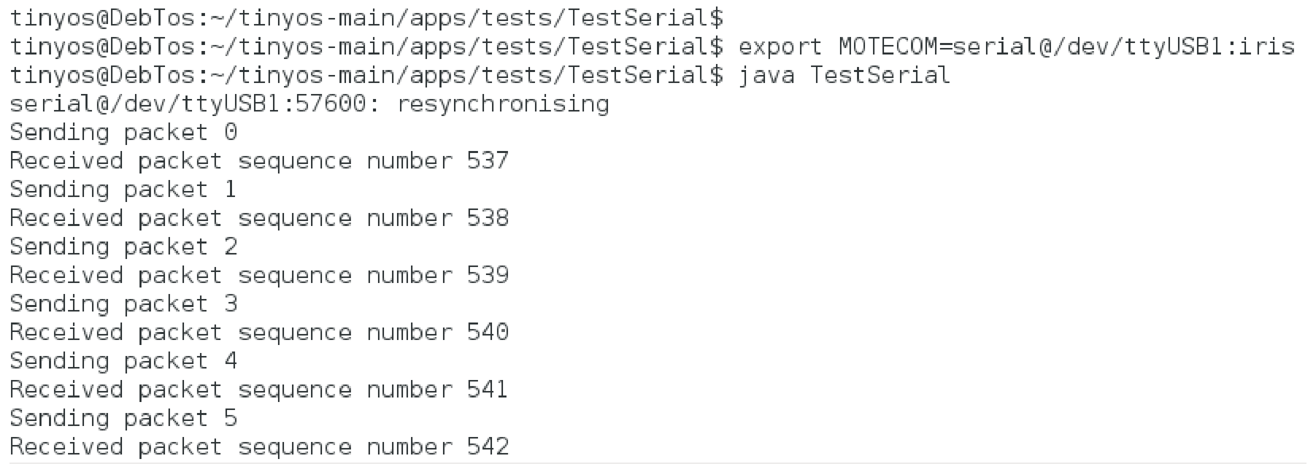
Pay attention: if you forget to add the –comm option, an error will occur:



```
tinyos@DebTos:~/tinyos-main/apps/tests/TestSerial$
tinyos@DebTos:~/tinyos-main/apps/tests/TestSerial$ java TestSerial
sf@localhost:9002 died - exiting (java.net.ConnectException: Connection refused)
tinyos@DebTos:~/tinyos-main/apps/tests/TestSerial$
```

To avoid writing the –comm parameter every time, you can give (once) the following command:

- export MOTECOM=serial@/dev/ttyUSBx:iris



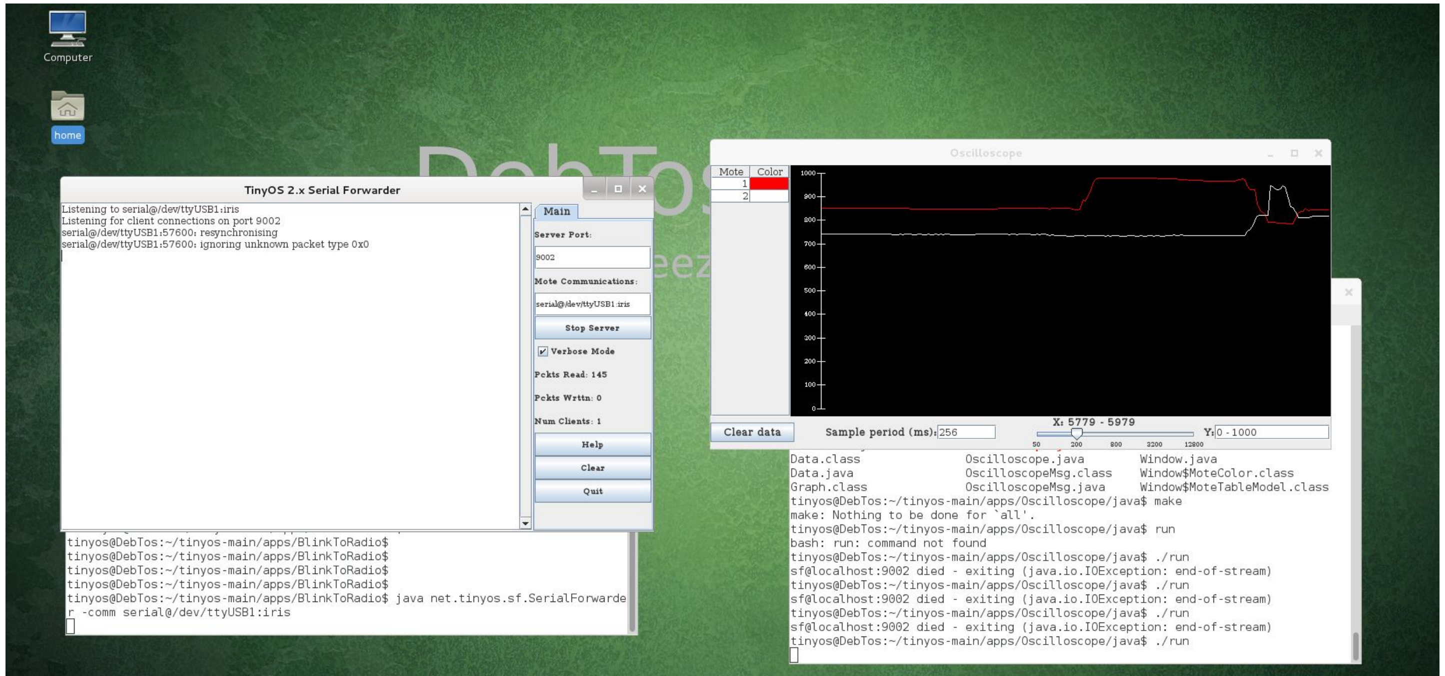
```
tinyos@DebTos:~/tinyos-main/apps/tests/TestSerial$
tinyos@DebTos:~/tinyos-main/apps/tests/TestSerial$ export MOTECOM=serial@/dev/ttyUSB1:iris
tinyos@DebTos:~/tinyos-main/apps/tests/TestSerial$ java TestSerial
serial@/dev/ttyUSB1:57600: resynchronising
Sending packet 0
Received packet sequence number 537
Sending packet 1
Received packet sequence number 538
Sending packet 2
Received packet sequence number 539
Sending packet 3
Received packet sequence number 540
Sending packet 4
Received packet sequence number 541
Sending packet 5
Received packet sequence number 542
```

TINYOS COMMANDS – LISTENING TO A SERIAL PORT

There are some TinyOS applications allowing you to see incoming data via a GUI, like Oscilloscope, you can use within the Serial Forwadrer.

In order to use MDA100CB sensor board (having plugged it on the motes, programmed with a proper application) to take light sensor data (shown by Oscilloscope), I gave the following command:

- SENSORBOARD=mda100 make iris



```
Listening to serial@/dev/ttyUSB1:iris
Listening for client connections on port 9002
serial@/dev/ttyUSB1:57600: resynchronising
serial@/dev/ttyUSB1:57600: ignoring unknown packet type 0x0
```

Server Port: 9002

Mote Communications: serial@/dev/ttyUSB1:iris

[✓] Verbose Mode

Pckts Read: 145

Pckts Wrttn: 0

Num Clients: 1

```
tinyos@DebTos:~/tinyos-main/apps/BlinkToRadio$
tinyos@DebTos:~/tinyos-main/apps/BlinkToRadio$
tinyos@DebTos:~/tinyos-main/apps/BlinkToRadio$
tinyos@DebTos:~/tinyos-main/apps/BlinkToRadio$
tinyos@DebTos:~/tinyos-main/apps/BlinkToRadio$ java net.tinyos.sf.SerialForwarde
r -comm serial@/dev/ttyUSB1:iris
```

Mote	Color
1	🟥
2	

Sample period (ms): 256

X: 5779 - 5979

Y: 0 - 1000

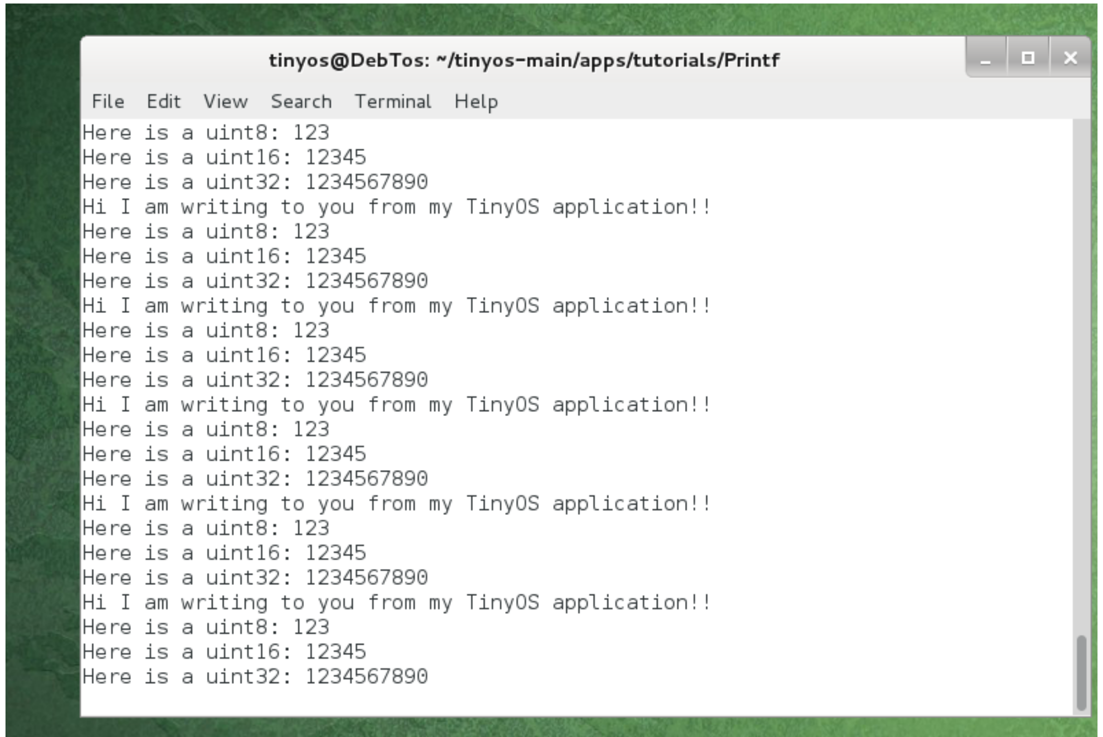
```
Data.class                Oscilloscope.java         Window.java
Data.java                 OscilloscopeMsg.class     Window$MoteColor.class
Graph.class               OscilloscopeMsg.java      Window$MoteTableModel.class
tinyos@DebTos:~/tinyos-main/apps/Oscilloscope/java$ make
make: Nothing to be done for `all'.
tinyos@DebTos:~/tinyos-main/apps/Oscilloscope/java$ run
bash: run: command not found
tinyos@DebTos:~/tinyos-main/apps/Oscilloscope/java$ ./run
sf@localhost:9002 died - exiting (java.io.IOException: end-of-stream)
tinyos@DebTos:~/tinyos-main/apps/Oscilloscope/java$ ./run
sf@localhost:9002 died - exiting (java.io.IOException: end-of-stream)
tinyos@DebTos:~/tinyos-main/apps/Oscilloscope/java$ ./run
sf@localhost:9002 died - exiting (java.io.IOException: end-of-stream)
tinyos@DebTos:~/tinyos-main/apps/Oscilloscope/java$ ./run
```

TINYOS COMMANDS – DEBUGGING

If you need an easy debugging, you can use TinyOS Printf library to show text messages in your application.

After installing the application on a mote, you can see these messages via the following command:

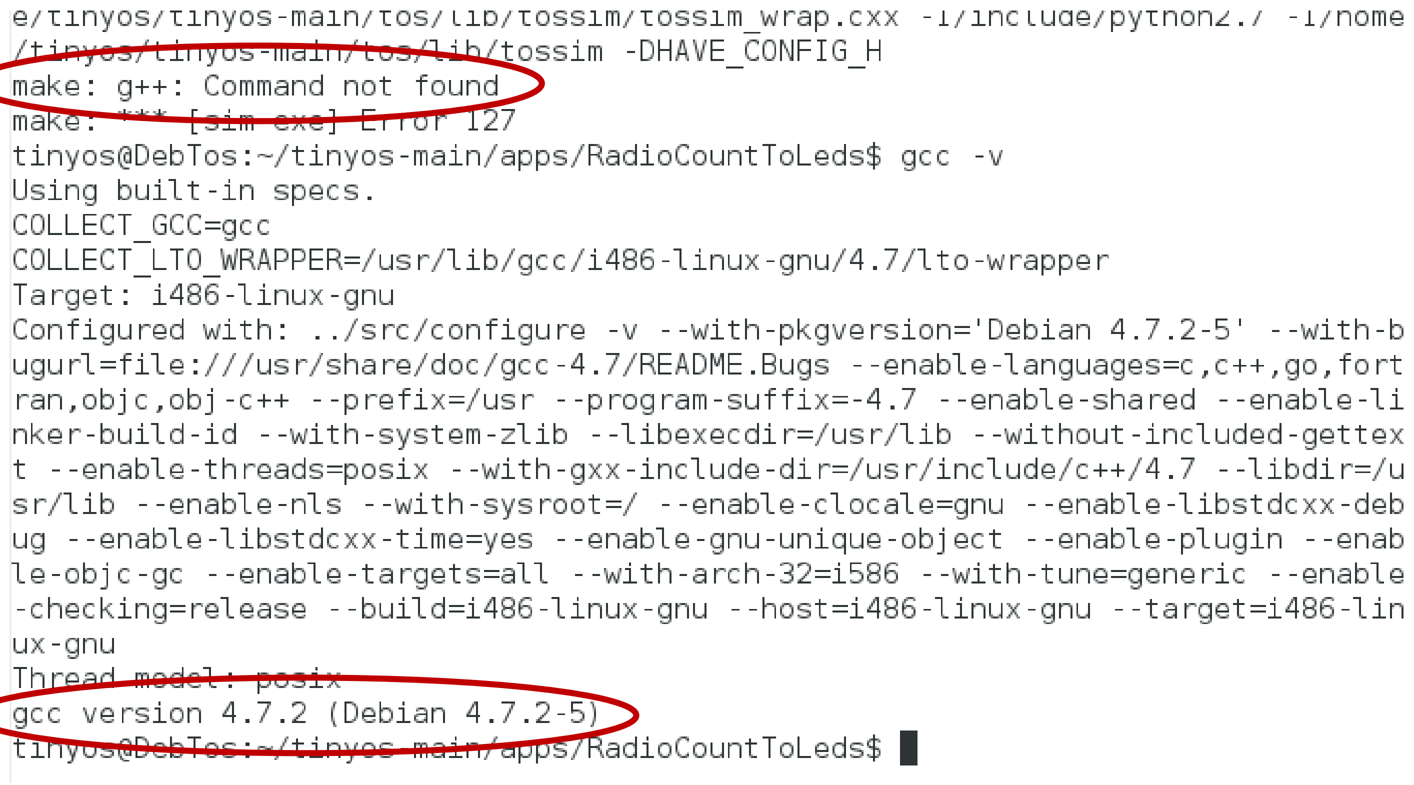
- java net.tinyos.tools.PrintfClient -comm serial@/dev/ttyUSBx:iris


tinyos@DebTos: ~/tinyos-main/apps/tutorials/Printf
File Edit View Search Terminal Help

```
Here is a uint8: 123
Here is a uint16: 12345
Here is a uint32: 1234567890
Hi I am writing to you from my TinyOS application!!
Here is a uint8: 123
Here is a uint16: 12345
Here is a uint32: 1234567890
Hi I am writing to you from my TinyOS application!!
Here is a uint8: 123
Here is a uint16: 12345
Here is a uint32: 1234567890
Hi I am writing to you from my TinyOS application!!
Here is a uint8: 123
Here is a uint16: 12345
Here is a uint32: 1234567890
Hi I am writing to you from my TinyOS application!!
Here is a uint8: 123
Here is a uint16: 12345
Here is a uint32: 1234567890
Hi I am writing to you from my TinyOS application!!
Here is a uint8: 123
Here is a uint16: 12345
Here is a uint32: 1234567890
```

TINYOS COMMANDS – TOSSIM TUTORIAL ERROR

During Tossim tutorial, the following strange error occurred:



```
e/tinyos/tinyos-main/tos/lib/tossim/tossim_wrap.cxx -I/include/python2.7 -I/home
/tinyos/tinyos-main/tos/lib/tossim -DHAVE_CONFIG_H
make: g++: Command not found
make: *** [sim-exe] Error 127
tinyos@DebTos:~/tinyos-main/apps/RadioCountToLeds$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/i486-linux-gnu/4.7/lto-wrapper
Target: i486-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Debian 4.7.2-5' --with-b
ugurl=file:///usr/share/doc/gcc-4.7/README.Bugs --enable-languages=c,c++,go,fort
ran,objc,obj-c++ --prefix=/usr --program-suffix=-4.7 --enable-shared --enable-li
nker-build-id --with-system-zlib --libexecdir=/usr/lib --without-included-gettex
t --enable-threads=posix --with-gxx-include-dir=/usr/include/c++/4.7 --libdir=/u
sr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --enable-libstdcxx-deb
ug --enable-libstdcxx-time=yes --enable-gnu-unique-object --enable-plugin --enab
le-objc-gc --enable-targets=all --with-arch-32=i586 --with-tune=generic --enable
-checking=release --build=i486-linux-gnu --host=i486-linux-gnu --target=i486-lin
ux-gnu
Thread model: posix
gcc version 4.7.2 (Debian 4.7.2-5)
tinyos@DebTos:~/tinyos-main/apps/RadioCountToLeds$ █
```

THANKS FOR YOUR ATTENTION