Porting TinyOS 1.x Code to TinyOS 2.0

Author: Tahir Azim and Philip Levis
Date: October 26 2006

Note

This document provides a few important points that describe major steps required for porting TinyOS 1.x code to TinyOS 2.0. It is based on Tahir Azim's experience porting Beacon Vector Routing (BVR[1]) from TinyOS 1.x to T2. This document is not a complete porting guide, but the hope is that it will provide some help or guidance.

1. Porting Points

As these observations come from porting a network protocol, they are rather protocol-centric and do not consider other abstractions such as storage. We hope to add such points in the future.

1. SUCCESS was a non-zero error code in TinyOS 1.x, while FAIL was non-zero. So any "if blocks" of the following form need to be changed:

```
if (call Packet...) {
    //SUCCESS!: do this...
}
```

In TinyOS 2.x, SUCCESS is equal to a zero error code, while other error codes are non-zero. So calls like this should be changed to make sure they test the result for equality with SUCCESS:

```
if (call Packet... () == SUCCESS ) {
      //SUCCESS!: do this...
  }
```

2. The "init()" and "start/stop()" methods in StdControl have been separated in TinyOS 2.x. The init() method is now part of the "Init" interface. Therefore all modules implementing StdControl should now implement Init also. Modules wired to the StdControl interface of a module should also wire to its Init interface.
3. The nx_bool data type should be replaced by nx_uint8_t.
4. Radios need to be started manually using SplitControl.start() and SplitControl.stop() at the beginning of the simulation. In

TinyOS 1.x, this was assumed to be done automatically by TOSSIM/TinyOS.

5. Packets are now an abstract data type (ADT) in TinyOS 2.x. Therefore, destination addresses from packets can no longer be obtained by using "msg -> addr". Instead the AMPacket.destination() method of the AMPacket interface should be used for this purpose. AMSenderC or AMReceiverC can be used to wire the AMPacket interface.
6. Similarly, in order to get a pointer to the payload of received message_t structures, and to get the payload lengths and maximum payload lengths of message_t structures, the Packet interface is used. This can also be wired to an AMSenderC or AMReceiverC component. Similarly, instead of using "msg->strength" to get the strength of a received signal, CC2420Packet.getRssi(msg) can be used. The CC2420Packet interface can be wired to CC2420ActiveMessageC.
7. Communication interfaces are very similar but require straightforward porting. SendMsg and ReceiveMsg interfaces (wherever used or provided by various modules) should be replaced by AMSend and Receive interfaces. At the lowest layer of the communication stack, AMSend and Receive interfaces should be wired to AMSenderC and AMReceiverC.
8. Where a module that previously provided SendMsg is changed to provide AMSend, extra methods have to be added that are part of the AMSend signature. These include the cancel, getPayload and maxPayloadLength methods. The Packet interface wired to AMSenderC can generally be used to implement these methods.
9. TOS_UART_ADDRESS no longer exists. Use an SerialAMSenderC component when you would like to send to the serial port.
10. TOS_LOCAL_ADDRESS no longer exists. There is now a distinction between the local node's ID (which is TOS_NODE_ID) and the active message address. The active message address of a communication interface can be obtained through the AMPacket.localAddress() command. By default, node ID and AM address are the same. TOS_NODE_ID is bound at compile-time, while an interface's AM address can be changed at runtime.
11. Calls such as Leds.greenToggle(), Leds.yellowToggle() etc need to be replaced by Leds.led1Toggle(), Leds.led2Toggle() etc.
12. You should no longer use "#ifdef PLATFORM_PC" to separate pieces of code that are to run only on the 'pc' target. Instead, "#ifdef TOSSIM" is used to identify blocks of code that should be run only in TOSSIM.
13. dbg messages no longer use one of the debug modes of the form, DBG_* as their first argument. Instead, they should be

replaced with strings identifying the sources from where the messages originated.

2. Author's Address

Tahir Azim
358 Gates Hall
Computer Systems Laboratory
Stanford University
Stanford, CA 94305

email - tazim@cs.stanford.edu

Philip Levis
358 Gates Hall
Computer Systems Laboratory
Stanford University
Stanford, CA 94305

phone - +1 650 725 9046

email - pal@cs.stanford.edu

3. Citations

[1] Rodrigo Fonseca, David Culler, Sylvia Ratnasamy, Scott Shenker, and Ion Stoica. "Beacon Vector Routing: Scalable Point-to-Point Routing in Wireless Sensornets." In Proceedings of the Second USENIX/ACM Symposium on Network Systems Design and Implementation (NSDI 2005).