On 09/12/2013 We learned how to compile and run programs on Telosb Mote

The program we tested today can be found at http://tinyos.stanford.edu/tinyos-wiki/index.php/Modules_and_the_TinyOS_Execution_Model

Where are all the sample apps?

/opt/tinyos-main/apps

Other important directories worth checking out..

tos/chips - drivers for individual chips
tos/interfacees - standard interfaces for the core components
tos/platforms - collection of drivers for platforms - note that the telosb drivers are split in both the telosa and telosb folder
tos/lib - library, interesting components are the timer and serialprintf
tos/types - header files for many of the core components. TinyError contains the error codes for error_t.

How to compile the code?

make telosb - for compiling the program
make telosb install - for quick compile and install
make teloab install bsl,/dev/ttyUSB0 - for installation on specific device

motelist – to list all the motes connected to the VM.

Sometimes the /dev/ttyUSBx port can get stuck and it is not possible to program the telosb. To fix the problem run picocom as described below and exit it again. This should reset the serial port.

Exercise 1 Due Thursday Sep 19 before class starts. Upload all modified source files and answers to Blackboard.

A) In the VM image there is a folder /home/cs450/cs450 which is also a git repository. If you do a 'git pull' you should get a new folder 'tosUartEcho'. This application should help you use printf over the serial port.

   If you compile and upload it, you can communicate with the telosb using picocom:
   picocom -b 115200 /dev/ttyUSB0
   (use Ctrl+a and then Ctrl+q to exit picocom.)

   Picocom should now echo key presses through the telosb and back to the screen at 115200 bits per second.
   To compile tosUartEcho you can simply follow the same instructions of compiling as above.

i) Take the screenshot of the screen and name it tosUartEcho.jpg or tosUartEcho.png
ii) Check the Makefile and source files and tell us what is needed to include Printf.
iii) Next, decouple the async event by posting a task - similar to the tutorial with the leds.
iv) The <enter> key only does a carriage return, '\r', not a newline. Catch the '\r' character and replace it with a "\n\r" when echoed back.
v) In the tos/platforms/telosa/chips/ folder there are three sensor drivers: s1087, s10871, sht11. Look at the s10871 (light sensor) and wire it into the tosUartEcho application, using the key presses to trigger sensor readings. Hint: you only need to use the Read<uint16_t> interface. Use printf to show the readings in picocom.

Extra: Look up other sensors and try to add them into the above application.

Programming in C

B) In /home/cs450/cs450 you will find a C application folder **cUartEcho** .

i) Compile and install the program using “make install” command for C applications.
ii) cUartEcho is using the slow 32kHz crystal so the UART speed is only 9600 bps. Adjust your call to picocom accordingly.
iii) Open UartEcho9600.c and note that function sig_UART1RX_VECTOR (void) is called whenever a character is received by the Mote and it transfers it back to the sender (**picocom**).
iv) Change the application such that program echo backs case converted alphabets from lower case to upper case and upper case to lower case. Hint: Look up the ASCII table and compare the numerical values.

C) In /home/cs450/cs450 you will find a C application folder **cBlink**
i) Compile and install the program using “make install” command for C applications.
ii) Open cBlink.c and note that function sig_TIMERAO_VECTOR (void) is called by timer to toggle the LEDs.
iii) Now change the program to Toggle only one LED.
iv) Now change the timer time by adjusting #define OFFSET. Can you tell us what does the initial value of OFFSET (which sets the timer) to 32768 signify in terms of hardware specification and how many milliseconds the timer is set to?

D) Modify cUartEcho such that an
i) RED LED Blinks when an upper case alphabet is received.
ii) GREEN LED Blinks when a lower case alphabet is received.
iii) **BLUE LED Blinks when a non-alphabet character is received.**