Networked Embedded Systems [NES]

Hello World

Gurjashan Pannu

13.12.2019

1 Introduction

In this exercise sheet, we will help you get familiar with the system (a *Moteiv Tmote Sky* sensor mote running the *Contiki* OS) that we will be working with during the semester.

1. First, prepare your environment for building firmware based on Contiki and build a small example firmware image.
   (a) Set up your environment, so that the *msp430* cross compiler and Contiki sources can be found. For the university computer pools, we are providing a shell script that you can have your shell read (i.e., *source*). To do this, open a new shell and run, e.g., the following shell command: "`source /upb/groups/fg-ccs/public/share/nes/2018w/env.sh`". Note that you will need to execute this again for each shell you open, so you might want to put this line in your shell's (interactive, non-login) startup file (e.g., `~/.bashrc`).
   (b) Download and unpack the example firmware's source code from PANDA[1]. We recommend choosing "`~/src/nes/hello`" as the destination directory.
   (c) Change to the destination directory of item 1b which holds the example source code, e.g., by running "`cd ~/src/nes/hello`".
   (d) We provide a `Makefile` that simplifies building the firmware. Familiarize yourself with the way this file works.
   (e) Build the firmware for the small test program, e.g., by running "`make hello.sky`".
2. Next, run the firmware in the *Cooja* emulator included with Contiki.
   (a) Run Cooja, e.g., by typing "`make simulation`". Make sure to maximize the window now.
   (b) Create a new simulation by choosing "File > New simulation...". The default parameters are fine.
   (c) Create a new mote type by choosing "Motes > Add motes > Create new mote type > Sky mote...". Browse to the destination directory of item 1b which holds the sample firmware. Choose "hello.c" as the source file to build the firmware from and click "Compile". You should see the compilation process complete successfully. Hit "Create" to finish creating the new mote type. A window will open asking whether you want to instantiate the new mote type.
   (d) Instantiate one of the newly-created mote types by clicking "Add motes". You should see the new mote appear in the "Network" pane (with the number "1" superimposed on it).

[1] see `http://panda.upb.de/`

(e) In the "Simulation control" pane, set the "Speed limit" to 100 %, then click on "Start". You should see the "Timeline" pane animating and, shortly after, output appearing in the "Mote output" pane (`... Hello World!`).

(f) Right click on mote 1 in the "Network" pane and select "Show LEDs on Sky 1". You should see a new "Mote Interface Viewer (Sky 1)" pane appear with 3 different colored circles representing 3 LEDs on the sensor mote. Familiarize yourself with other interface viewers (button, serial port, etc.)

(g) Right click on mote 1 in the "Network" pane and select "Click button on Sky 1". You should see a light sensor reading appear in the "Mote output" pane (`Light sensor value: ...`) and 3 LEDs in item 2f change the colors.

3. Next, run the same firmware on the sensor mote.

(a) Sign out one of our sensor motes.

(b) Familiarize yourself with the mote. Compare its design with that shown on the lecture slides. Find the *reset* button and the *user* configurable button.

(c) Plug our mote (tightly) into a USB port of your machine. Check your kernel log to see if the mote's USB/serial converter has registered with the USB, e.g., by running `dmesg` and checking for log lines like "`new full-speed USB device`", followed by an indication like "`FTDI USB Serial Device converter now attached to ttyUSB0`".

(d) Change to the destination directory of item 1b which holds the example simulation. Make sure that Contiki detects your node by running `make motelist`. You should see a line containing the device's name just mentioned (e.g., `/dev/ttyUSB0`).

(e) Upload the firmware to the sensor mote by running "`make hello.upload`".

(f) Connect your terminal to the running firmware's stdin/stdout by running "`make login`". Carefully press the *user* button on the mote (you should hear a soft *click*). Just like in item 2g, you should see the firmware output a light sensor reading and the LEDs on the mote change the colors. If you signed out a mote with a light sensor, the reading value is non-random. Close the connection by pressing `Ctrl+C`.

(g) Return the sensor mote you signed out so that it may be used by one of your colleagues.

This concludes our "Hello World" exercises.

Contact

Gurjashan Pannu <pannu@ccs-labs.org>

Course Website: http://www.ccs-labs.org/teaching/nes/2019w/