EMBSYS 110 Assignment 3

LED HSM

1 Goal

The goal of this assignment is to implement an LED pattern generator using HSM (Hierarchical State Machine). This approach allows the source code (implementation) to be directly traceable to the statechart (design) which can in turn be traced back to the requirements (specifications).

See Session 2.1. Model-Based Software Development of the following publication:

Automatic code generation for instrument flight software. Kiri L. Wagstaff, Edward Benowitz, Dj Byrne, Ken Peters, Garth Watney. Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2007.

(http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.126.548&rep=rep1&type=pdf)

2 Setup

1. *Carefully* unplug any extension modules from your Nucleo board. They are not required and may hinder your view of the USER LED. Besides, the USER LED pin (PA.5) is shared with the SPI CLK pin of the LCD module, and therefore they cannot be used together.
2. Download the compressed project file (platform-stm32f401-nucleo_assignment3.tgz) from the Assignment 3 course site.

   Place the downloaded tgz file in your VM under **~/Projects/stm32**.
3. Move your existing project folder to a backup folder, e.g.

   mv platform-stm32f401-nucleo platform-stm32f401-nucleo.bak

   Note – You may want to pick another backup folder name if the one shown above already exists.
4. Decompress the tgz file with:

   tar xvzf platform-stm32f401-nucleo_assignment3.tgz

   The project folder will be expanded to **~/Projects/stm32/platform-stm32f401-nucleo**.
5. Launch Eclipse. Hit F5 to refresh the project content.

   Or you can right-click on the project in Project Explorer and then click "Refresh".

6. Clean and rebuild the project. Download it to the board and make sure it runs.

   In minicom, type the command "`led ?`" and verify you see the following help text:

```
22562 CONSOLE_UART2> led ?
[Commands]
test            Test function
on              Start a pattern
off             Stop a pattern
?               List commands
```

   Type the command "`led on 0`" and verify that the USER LED does NOT blink.

7. The statechart was created with the free tools named **UMLet**. You can download the latest version (14.3) with this link:

   https://www.umlet.com/download/umlet_14_3/umlet-standalone-14.3.0.zip

   (or on this page: http://umlet.com/changes.htm. Be careful NOT to click on those advertisement Download links!)

   The stand-alone version is more stable. It runs on Java, so it should work in your VM which has Java installed.

   Note – UMLet is to be installed (decompressed) in your VM but not on your host machine. Once installed, CD to the installation folder (e.g. ~/Projects/Umlet) and type "`./umlet.sh &`" to launch it.

3 Tasks

1. Imagine the following requirements are handed to you about an LED module:

   (a) Upon a pattern request event, the module should show the specified LED pattern.

   (b) If the pattern request includes a "repeated" flag, the module should show the specified LED pattern repeatedly.

   (c) Upon an *off* request event, the module should turn off the LED.

2. Think about the above requirements. Could you find any gaps in the specifications?

3. Even for a single LED, specifying its behaviors is not as simple as it appears. Now review the provided statechart in **src/UserLed/UserLed.uxf**. Does it tell us more about how the LED module should behave?

4. Copy and paste the two LED patterns you added in the previous assignment to **src/UserLed/LedPattern.cpp** (appending them after the two sample patterns). Remember to update the total count to 4.

5. Implement the statechart in **src/UserLed/UserLed.cpp** and **src/UserLed/UserLed.h**.

   The header file already contains declarations for all the state functions, internal events, timers, member variables required by the statechart design. Refer to the washing machine example for the coding rules.

6. Test your code with the console command:

   led on <pattern index from 0 to 3 > <0 for non-repeating; 1 for repeating>

   (e.g. "led on 2 1" to show pattern 2 repeatedly.)

   led off

   See how you can start a new pattern or stop it at any time. For example you can stop a pattern in the middle of a cycle. You can change the pattern before a cycle has finished. This design is reactive and responsive, which is the essence of real-time systems.

4 Submission

The due date is 4/27 Monday 11:59pm. Please submit:

1. A zip file containing the source code in **src/UserLed**.
2. A log file capturing the output of the UART console. You can turn on capturing in minicom by:

   *sudo minicom **-C log.txt***

   Your log should at least contain the output of the following commands:

   a) led on 2 0 (shows pattern 2 once)

   b) led on 3 1 (shows pattern 3 repeatedly)

   c) led off (stops pattern 3 after about 5 cycles)

Upload to the usual Canvas assignment upload location.