CST 347 – Lab 1 (Intro to FreeRTOS and PIC Starter Kit
Spring 2013, Instructors: Claude Kansaku & Troy Scevers

Procedure:

1) Create a /cst347/labs folder.

2) Create a /cst347/labs/FreeRTOS folder.

3) Create a /cst347/labs/lab 1 folder.

4) From the FreeRTOS installation on the p: drive, copy the /CST347/FreeRTOS/FreeRTOSV7.4.0/FreeRTOS/License and the /CST347/FreeRTOS/FreeRTOSV7.4.0/FreeRTOS/Source folders to the cst347/labs/FreeRTOS folder.

5) Copy the /src and the /include folders provided by the instructor to the lab 1 folder. In your /cst347/labs/lab 1/ folder you should have the following files:
/cst347/labs/lab 1/src/main.c and
/cst347/labs/lab 1/include/FreeRTOSConfig.h.

6) In summary, your folder structure should now be:
/cst347/labs/FreeRTOS/License
/cst347/labs/FreeRTOS/Source
/cst347/labs/lab 1/include
/cst347/labs/lab 1/src

7) Using the MPLAB X wizard, create a project with the following properties:
   1. Microchip Embedded – Standalone Project
   2. Family: 32-bit MCUs (PIC32) – Device: PIC32MX360F512L (or x795x)
   4. Microchip Starter Kits: SKDE PIC32
   6. XC32 (v1.xx)
   7. Project Name: lab 1

   Project Location: z:\cst347\labs\lab 1
   Project Folder: z:\cst347\labs\lab 1 (Delete the default ".X" naming extension)

8) In the Source Files logical folder, Add Existing Item…\cst347\labs\lab 1\src\main.c. **IF YOU ARE USING THE x795X Ethernet SK**, uncomment the #pragma config PWP = OFF /*, UPLLEN = OFF, FSRSSEL = PRIORITY_7 */ line to include the UPLLEN and FSRSSEL parameters.

9) In the Source Files logical folder, create a FreeRTOS logical folder.

10) In the Source Files\FreeRTOS logical folder, Add Existing Item…files:
\cst347\labs\FreeRTOS\src\list.c
\cst347\labs\FreeRTOS\src\queue.c

\cst347\labs\FreeRTOS\src\tasks.c
\cst347\labs\FreeRTOS\src\portable\MemMang\heap_2.c
\cst347\labs\FreeRTOS\src\portable\MPLAB\PIC32MX\port.c
\cst347\labs\FreeRTOS\src\portable\MPLAB\PIC32MX\port_asm.S

11) In the Header Files logical folder, Add Existing Item…
\cst347\labs\lab 1\include\ FreeRTOSConfig.h.

12) In the Header Files logical folder, create a FreeRTOS logical folder.

13) In the Header Files\FreeRTOS logical folder, Add Existing Item…files:
\cst347\labs\FreeRTOS\Source\include\croutine.h
\cst347\labs\FreeRTOS\Source\include\FreeRTOS.h
\cst347\labs\FreeRTOS\Source\include\list.h
\cst347\labs\FreeRTOS\Source\include\portable.h
\cst347\labs\FreeRTOS\Source\include\projdefs.h
\cst347\labs\FreeRTOS\Source\include\queue.h
\cst347\labs\FreeRTOS\Source\include\semphr.h
\cst347\labs\FreeRTOS\Source\include\task.h

14) Open lab 1 Project Properties using the File menu or the Dashboard "wrench" icon. Select the XC32 (Global Options) – xc32-as option on the left. Add the following path to the "Preprocessor Include directories" option field: (Double-click to manually enter path)
./include

15) Open lab 1 Project Properties. Select the XC32 (Global Options) – xc32-gcc option on the left. Add the following paths to the "Include directories" option field: (Double-click to manually enter paths)
./include
../FreeRTOS/Source/include
../FreeRTOS/Source/portable/MPLAB/PIC32MX

16) Create a Backup Copy of This project for later reference

---

1) You will now modify this demo project to do the following:

2) Create a new set of files called led.c and led.h and add them into your project (These should be stored in the include and src directories of your project)

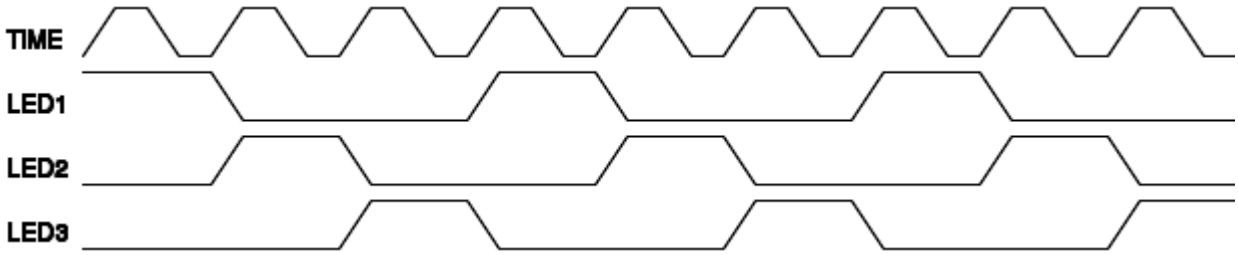
3) In the led files write an LED "driver" function call *unsigned int led_drive(int lednum, int cmd, int param)* that will perform the following functionality:
   a. *lednum* is defined as follows:
      i. 0 = LED1 (RD0)
      ii. 1 = LED2 (RD1)
      iii. 2 = LED3 (RD2)

b. *cmd* is defined as follows:
    i. 0 = Read an LED state (returns 0 for OFF and 1 for ON)
    ii. 1 = Set an LED state to OFF or ON (returns 0, uses *param* to set value as defined below)
    iii. 2 = TOGGLE the current LED state (returns 0)
c. Only *cmd = 1* uses *param* as follows: (*param* is a don't care for all other cases)
    i. 0 = OFF
    ii. 1 = ON

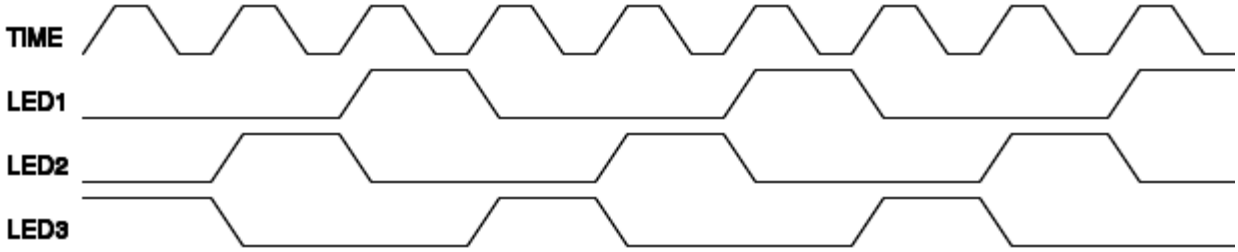
4) Create another set of files called tasks.c and tasks.h and add them to your project (These should be stored in the include and src directories of your project)

5) You will also have to modify your main.c file to include this new header file tasks.h

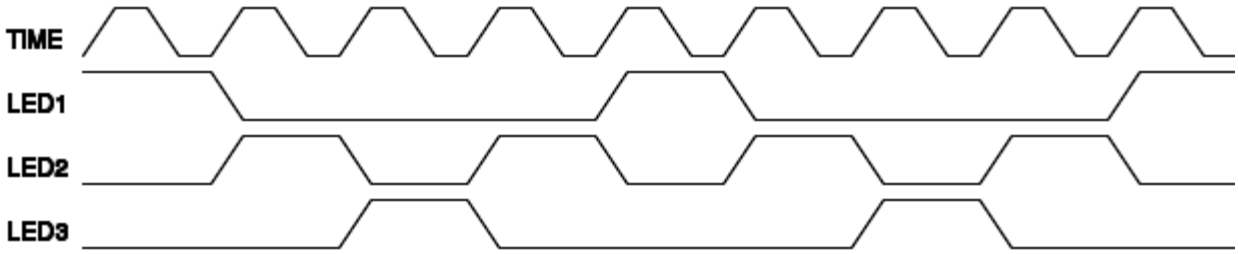
6) Write a task that uses your *led_drive()* to perform three different LED light patterns as follows:
    a. Strobe down continuously



    b. Strobe up continuously



    c. Bounce up and down continuously



The task will use two parameters to set the LED ON/OFF time and pattern. You will create the parameter structure similar to the example provided.

7) Demonstrate each of these patterns to the Lab Instructor