Assignment 1:

The purpose of the first assignment is to get aquatint with the kit (*Figure 1*), and to write your first program.

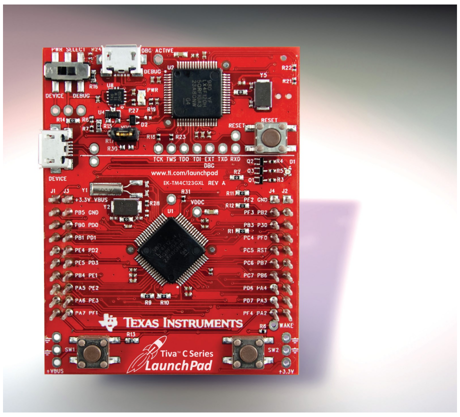


Figure 1. The Kit.

Assignment:
Write a program for the kit.
The program must implement a binary counter (0-7) with the value shown at the RGB –LED of the kit, giving 8 different colours in a given sequence.
The counter must be able to count up and down.
The counter must advance one step whenever the <SW1> button is pushed.
The direction (up/down) must be toggled when the <SW1> button is double clicked.
A continuous press at <SW1> for more than 2 seconds must set the counter in AUTO MODE. In AUTO MODE the counter will automatically advance one step (up or down) every 200 millisecond.
Any push to the <SW1> button while the counter is in AUTO MODE must return the counter the normal state.

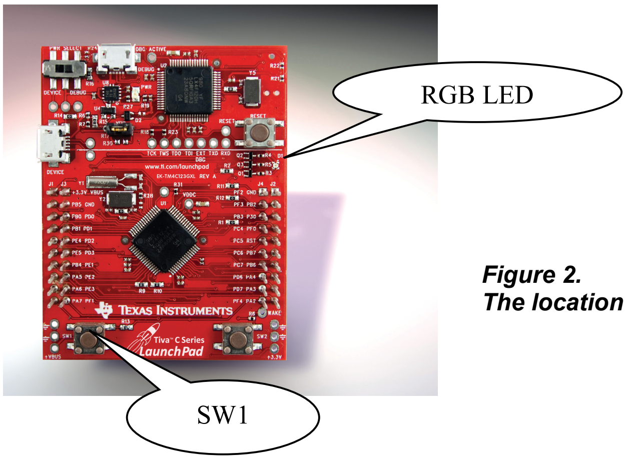


Figure 2.
The location of the RGB-LED and SW1.

RGB-LED:

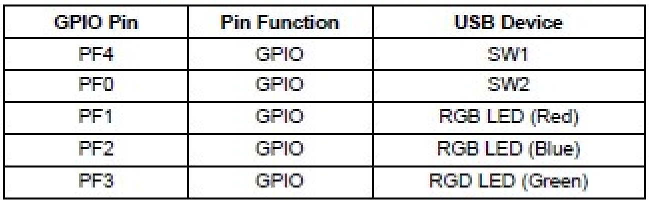
Showing the binary counter on the RGB-LED will generate the following colors.

Counter value	*bit 2 red PF1*	*bit 1 blue PF2*	*bit 0 green PF3*	*color*		
0	0	0	0	LEDs turned off		
1	0	0	1	Green		
2	0	1	0	Blue		
3	0	1	1	Cyan		
4	1	0	0	Red		
5	1	0	1	Yellow		
6	1	1	0	Magenta		
7	1	1	1	White		

I/O:
In order to fulfill assignment 1, you need to access the <SW1> button and the RGB-LED (red, green and blue.

The Port layout of the kit can be seen in “Tiva™ C Series TM4C123G LaunchPad Evaluation Board User's Guide” at page 9, which can be found at Black Board Datasheets/The kit.

Table 2-2. User Switches and RGB LED Signals



GPIO Pin	Pin Function	USB Device
PF4	GPIO	SW1
PF0	GPIO	SW2
PF1	GPIO	RGB LED (Red)
PF2	GPIO	RGB LED (Blue)
PF3	GPIO	RGD LED (Green)

The <SW1> button is mapped to the General Purpose I/O (GPIO) port F, bit 4, and the RGB_LED is mapped to the General Purpose I/O (GPIO) port F, bit 1-3.

Now it is time to consult the Tiva™ TM4C123GH6PM Microcontroller data sheet. In the chapter about “General-Purpose Input/Outputs (GPIOs)” you will find information of how to access the registers related to a given GPIO.

In the subchapter: “Register Map”, you will find the following information.

GPIO	GPIO base address
GPIO Port G	0x4002.5000

The most important registers are shown in the next table.

Offset	Name	**Type**	**Reset**	**Description**
0x000	GPIODATA	R/W	0x0000 0000	GPIO Data
0x400	GPIODIR	R/W	0x0000 0000	GPIO Direction
0x510	GPIOPUR	R/W	-	GPIO Pull-Up Select
0x51C	GPIODEN	R/W	-	GPIO Digital Enable

You can find more detail about the individual registers in the subchapter “Register Descriptions”

If you create a new project according to the note “Create a C project in Code Composer”, then you can use the register names defined in the “tm4c123gh6pm.h” file to access the registers. Note the register names defined in the “tm4c123gh6pm.h” file are unfortunately not the same as those in the datasheet, but you can use the address of a register to find the register in the “tm4c123gh6pm.h” file.

Remember to enable the clock for the F GPIO port.

Timing:

You might also want to use a hardware timer to time the AUTO MODE. Here you have the choice of the SysTick Timer or a General Purpose Timer. You can get inspiration to the use of the SysTick timer from the example: systick.c, which can be found in the CODE folder at BB. The code in systick.c sets up the SysTick timer to generate an interrupt every 5 mS.

Enjoy, see you next week
Morten