

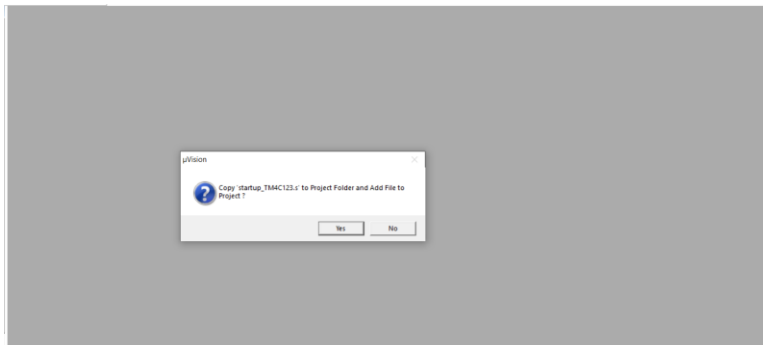


## Lab 5

### Introduction

To use the simulated ports of TivaC launchpad, follow the below steps:

1. Create new project.
2. Choose the target TM4C123GH6PM device.
3. Copy the start-up code of TM4C123GH6PM.



4. Remove instruction "IMPORT SystemInit"
5. Remove instruction "LDR R0, =SystemInit"
6. Remove instruction "BLX R0" at line 236 as shown below

```
startup_TM4C123.s
224 __Vectors_Size EQU __Vectors_End - __Vectors
225
226 AREA |.text|, CODE, READONLY
227
228 ; Reset Handler
229
230
231 Reset_Handler PROC
232 EXPORT Reset_Handler [WEAK]
233 IMPORT SystemInit
234 IMPORT __main
235 LDR R0, =SystemInit
236 BLX R0
237 LDR R0, =__main
238 BX R0
239 ENDP
240
241 ; Dummy Exception Handlers (infinite loops which can be modified)
242
243 NMI_Handler PROC
244 EXPORT NMI_Handler [WEAK]
245 B .
246 ENDP
247
248 HardFault_Handler\
249 PROC
250 EXPORT HardFault_Handler [WEAK]
251 B .
252 ENDP
253 MemManage_Handler\
254 PROC
```



7. Adjust the settings of the target by checking MicroLIB field in Code Generation options.

Options for Target 'Target 1'

Device: Texas Instruments TM4C123GH6PM

Xtal (MHz): 16.0

Operating system: None

System-Viewer File (.Sfr): TM4C123GH6PM.SFR

☐ Use Custom SVD File

**Code Generation**

☐ Use Cross-Module Optimization

☒ Use MicroLIB

☐ Big Endian

Floating Point Hardware: Use FPU

**Read/Only Memory Areas**

| default                             | off-chip | Start | Size    | Startup                          |
|-------------------------------------|----------|-------|---------|----------------------------------|
| <input type="checkbox"/>            | ROM1:    |       |         | <input type="radio"/>            |
| <input type="checkbox"/>            | ROM2:    |       |         | <input type="radio"/>            |
| <input type="checkbox"/>            | ROM3:    |       |         | <input type="radio"/>            |
| on-chip                             |          |       |         |                                  |
| <input checked="" type="checkbox"/> | IRAM1:   | 0x0   | 0x40000 | <input checked="" type="radio"/> |
| <input type="checkbox"/>            | IRAM2:   |       |         | <input type="radio"/>            |

**Read/Write Memory Areas**

| default                             | off-chip | Start      | Size   | NoInit                   |
|-------------------------------------|----------|------------|--------|--------------------------|
| <input type="checkbox"/>            | RAM1:    |            |        | <input type="checkbox"/> |
| <input type="checkbox"/>            | RAM2:    |            |        | <input type="checkbox"/> |
| <input type="checkbox"/>            | RAM3:    |            |        | <input type="checkbox"/> |
| on-chip                             |          |            |        |                          |
| <input checked="" type="checkbox"/> | IRAM1:   | 0x20000000 | 0x8000 | <input type="checkbox"/> |
| <input type="checkbox"/>            | IRAM2:   |            |        | <input type="checkbox"/> |

OK Cancel Defaults Help

8. Adjust the settings of the target by adding “-dLaunchPadDLL” in Parameter field to support the simulation in Keil 4.

Options for Target 'Target 1'

Device: Texas Instruments TM4C123GH6PM

☒ Use Simulator **Settings**

☐ Limit Speed to Real-Time

☒ Load Application at Startup ☒ Run to main()

Initialization File: ... Edit...

**Restore Debug Session Settings**

☒ Breakpoints ☒ Toolbox

☒ Watch Windows & Performance Analyzer

☒ Memory Display ☒ System Viewer

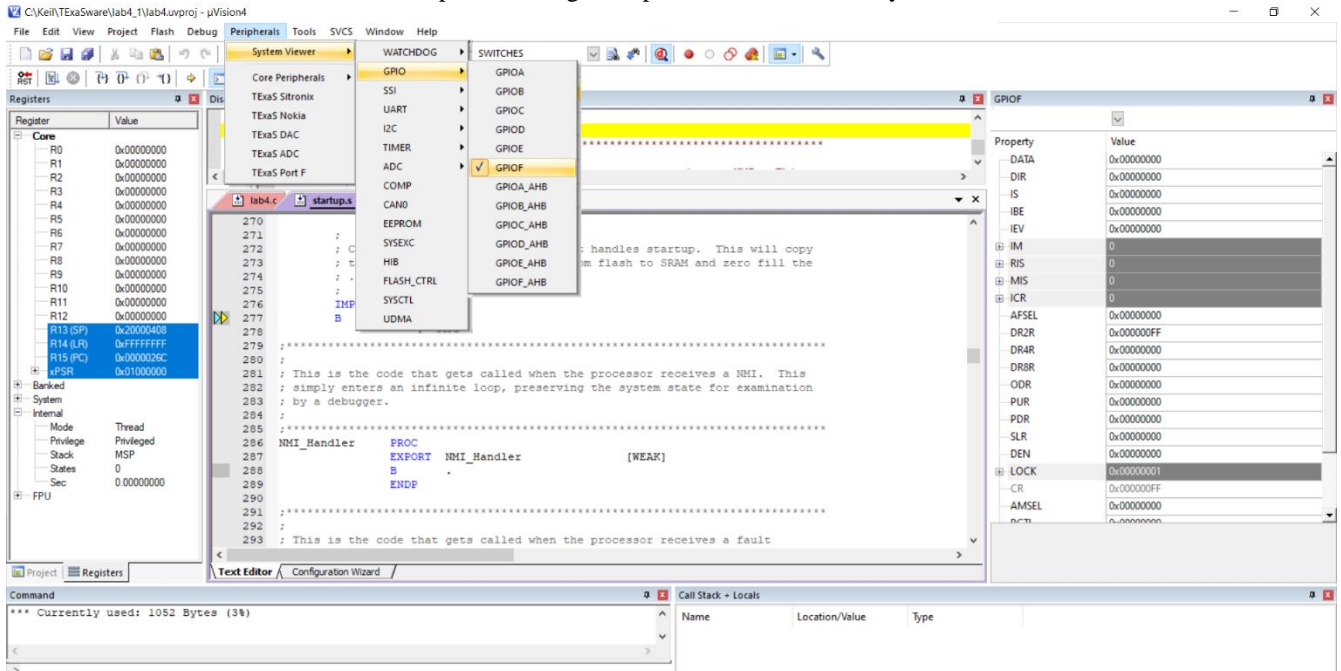
CPU DLL: SARMCM3.DLL Parameter: -MPU

Dialog DLL: DCM.DLL Parameter: -pCM4 -dLaunchPadDLL

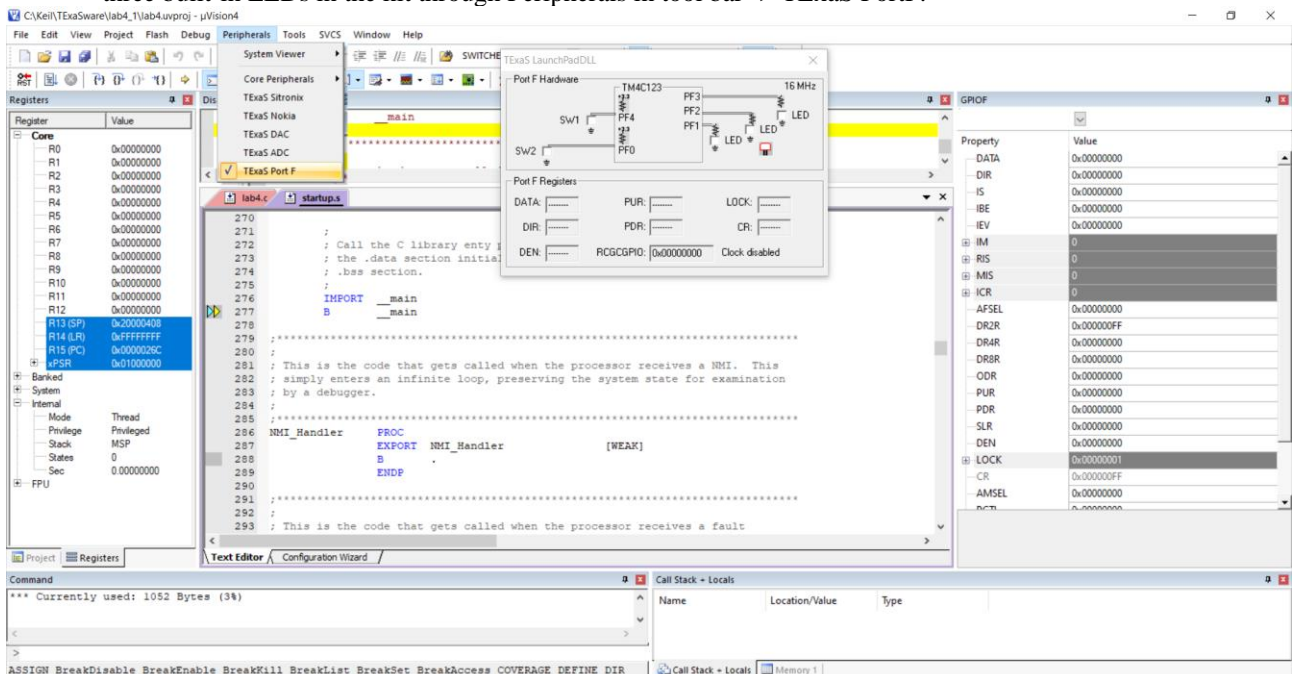
OK Cancel Defaults Help



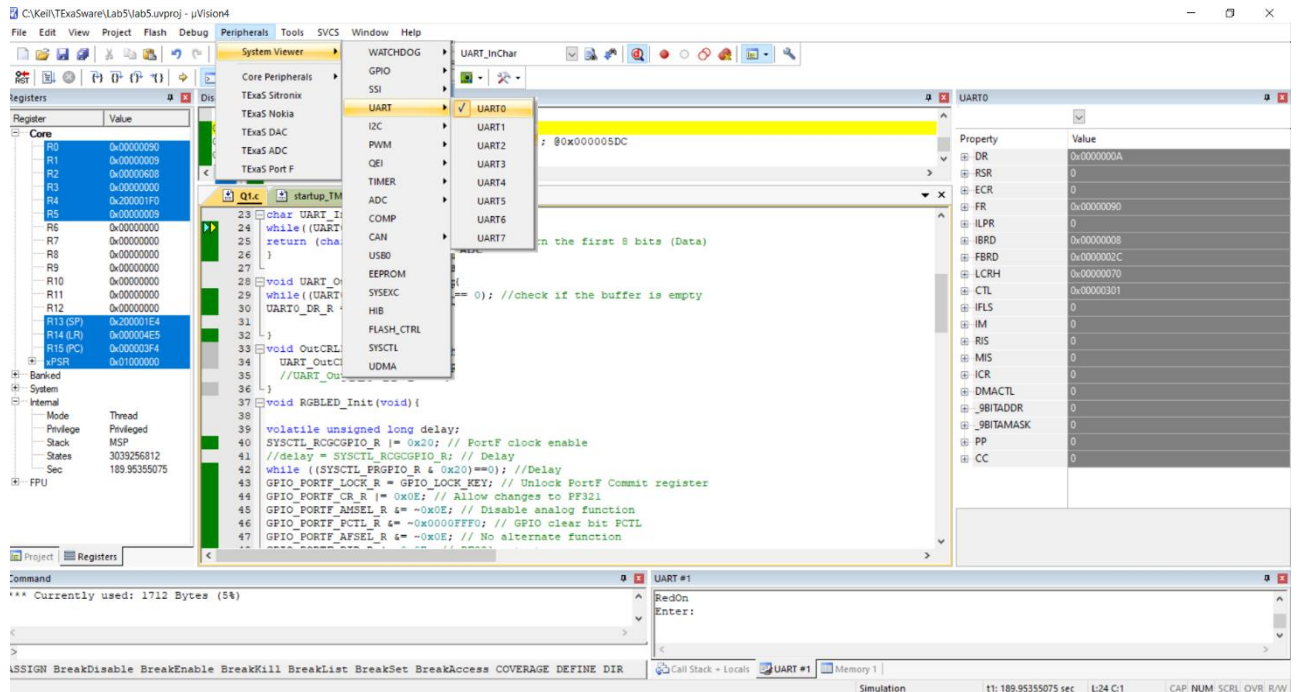
9. You can view the values of the port F through Peripherals in tool bar -> SystemViewer-> GPIO->GPIOF.



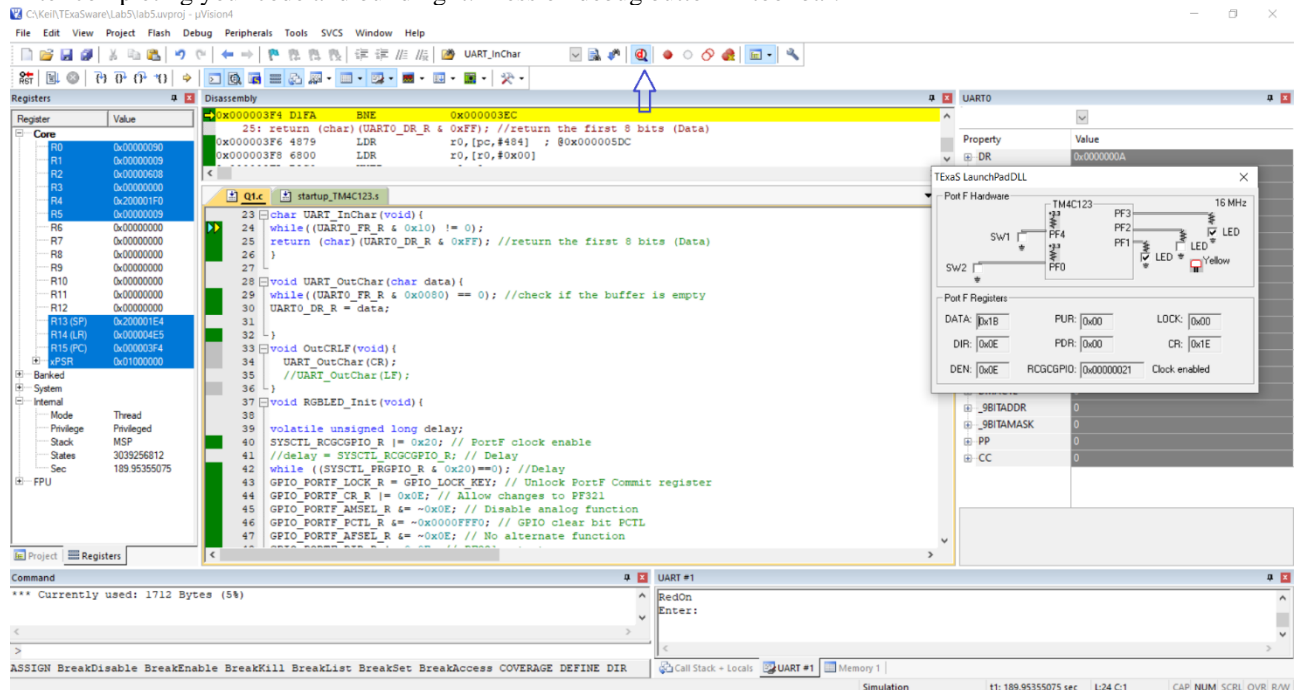
10. You can simulate the behavior of switches that are connected to port F in TivaC and check the behavior of the three built-in LEDs in the kit through Peripherals in tool bar -> TExaS PortF.



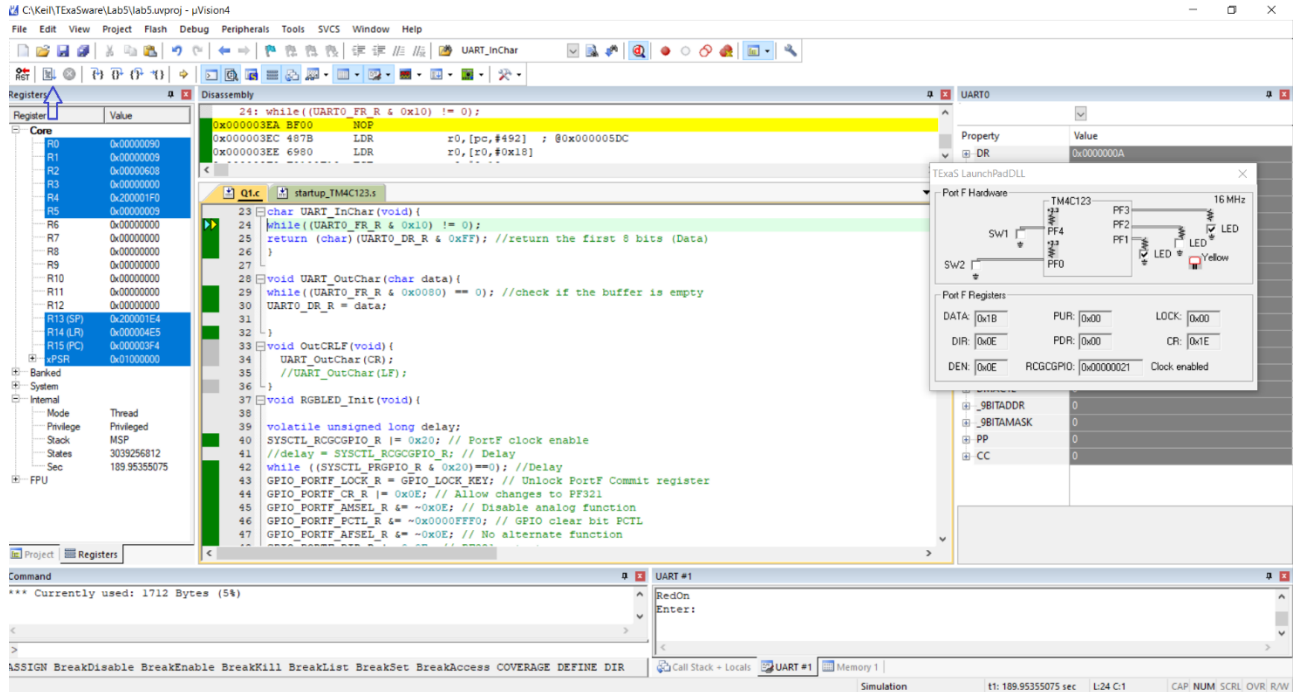
11. You can check the values of any UART registers through Peripherals in tool bar-> System Viewer-> UART->UART<x>.



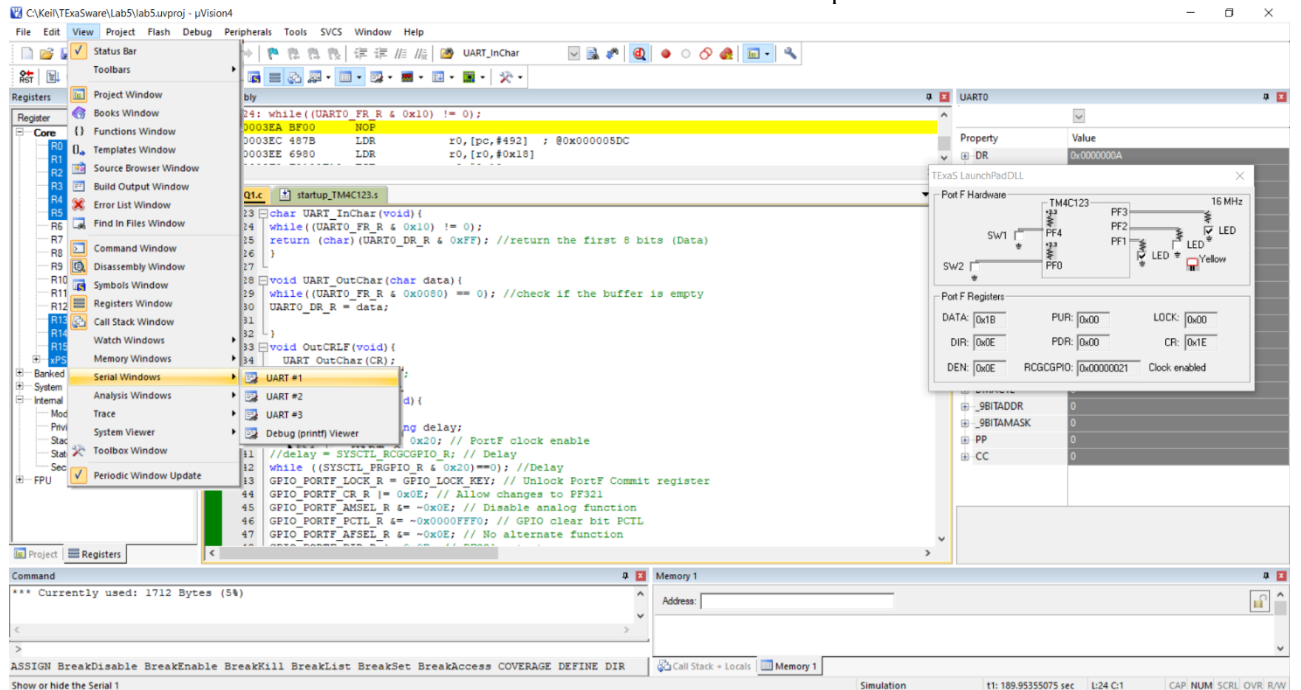
12. After completing your code and building it. Press on debug button in tool bar.



13. After pressing on debug button, press on run button in tool bar as shown below.



14. From view in tool bar -> choose Serial Window-> then choose UART#1 to open Keil terminal.





15. The UART#1 serial window will be appeared to provide you the capability to enter your inputs during running the program.

The screenshot displays the TI-RTOS IDE interface. The main window shows the disassembly of the UART initialization code. The code includes functions for UART initialization, data reception, and data transmission. The UART#1 serial window is open, showing the command prompt and the current state of the UART. The registers window shows the values of the registers, and the port hardware window shows the configuration of the port hardware.

Registers:

| Register | Value      |
|----------|------------|
| R0       | 0x00000000 |
| R1       | 0x00000000 |
| R2       | 0x00000000 |
| R3       | 0x00000000 |
| R4       | 0x00000000 |
| R5       | 0x00000000 |
| R6       | 0x00000000 |
| R7       | 0x00000000 |
| R8       | 0x00000000 |
| R9       | 0x00000000 |
| R10      | 0x00000000 |
| R11      | 0x00000000 |
| R12      | 0x00000000 |
| R13 (SP) | 0x200001E4 |
| R14 (LR) | 0x000004E5 |
| R15 (PC) | 0x000003F4 |
| LRSR     | 0x00000000 |

Disassembly:

```
24: while((UART0_FR_R & 0x10) != 0);  
0x000003EA BF00 NOP  
0x000003EC 487B LDR r0,[pc,#492] ; @0x000005DC  
0x000003EE 6980 LDR r0,[r0,#0x18]  
25: while((UART0_FR_R & 0x10) != 0);  
26: return (char)(UART0_DR_R & 0xFF); //return the first 8 bits (Data)  
27:  
28: void UART_OutChar(char data){  
29: while((UART0_FR_R & 0x0800) == 0); //check if the buffer is empty  
30: UART0_DR_R = data;  
31: }  
32:  
33: void OutCRLF(void){  
34: UART_OutChar(CR);  
35: //UART_OutChar(LF);  
36: }  
37: void RGBLED_Init(void){  
38:  
39: volatile unsigned long delay;  
40: SYSCTL_RCGCGPIO_R |= 0x20; // PortF clock enable  
41: //delay = SYSCTL_RCGCGPIO_R; // Delay  
42: while ((SYSCTL_PRGPIO_R & 0x20)==0); //Delay  
43: GPIO_PORTF_LOCK_R = GPIO_LOCK_KEY; // Unlock PortF Commit register  
44: GPIO_PORTF_CR_R |= 0x0E; // Allow changes to PF321  
45: GPIO_PORTF_AMSEL_R &= ~0x0E; // Disable analog function  
46: GPIO_PORTF_PCTL_R &= ~0x0000FFFF; // GPIO clear bit PCTL  
47: GPIO_PORTF_ASEL_R &= ~0x0E; // No alternate function  
48: GPIO_PORTF_SLR_R &= ~0x0E; // No alternate function
```

UART#1 serial window:

```
*** Currently used: 1712 Bytes (5%)  
RedOn  
Enter:  
<
```

16. Use the previous steps to help you out to simulate the following lab.



## Lab Exercise

Write an Embedded C program that receives commands through UART communication protocol to turn the LEDs on and off.

The commands are:

1. RedOn → Turn RED LED on
2. RedOff → Turn RED LED off
3. GreenOn → Turn Green LED on
4. GreenOff → Turn Green LED off
5. BlueOn → Turn Blue LED on
6. BlueOff → Turn Blue LED off
7. Anything else turn all the LEDs off

Your code should support the following assumptions:

1. Any command that turns on a specified LED should not affect the other LEDs.
2. Anything is sent except those commands should reset all the LEDs (reset the only pins that are connected to LEDs).
3. Any command that turns off a specified LED should not affect the other LEDs.
4. The new command is completed by pressing Enter.





## Lab Submission

Q2. Write Embedded C program that receives commands through UART communication protocol to do the following:

1. When sending "A", all the LEDs are turned off and the Red LED is turned on after 1 minute.
2. When sending "B", all the LEDs are turned off and the Blue LED is turned on after 0.5 minutes.
3. When sending "D", all the LEDs are turned off and the Green LED is turned on after 2 minutes.

Upon starting the program, all the LEDs should be turned off.

Check through the simulated Kit that the behavior of your code is correct.

For the lab submission, you should submit a pdf document contains the following.

1. Cover page that contains
  - a. your name,
  - b. your ID,
  - c. your department
2. Place snapshots to show the state of the LEDs.
3. The snapshots must show the values of the GPIOF registers such as (DATA, DIR, AFSEL, ... etc.), UART registers, and UART serial window when you verify your code on simulation level.
4. Place your code in the document.
5. Your document will be submitted on LMS.