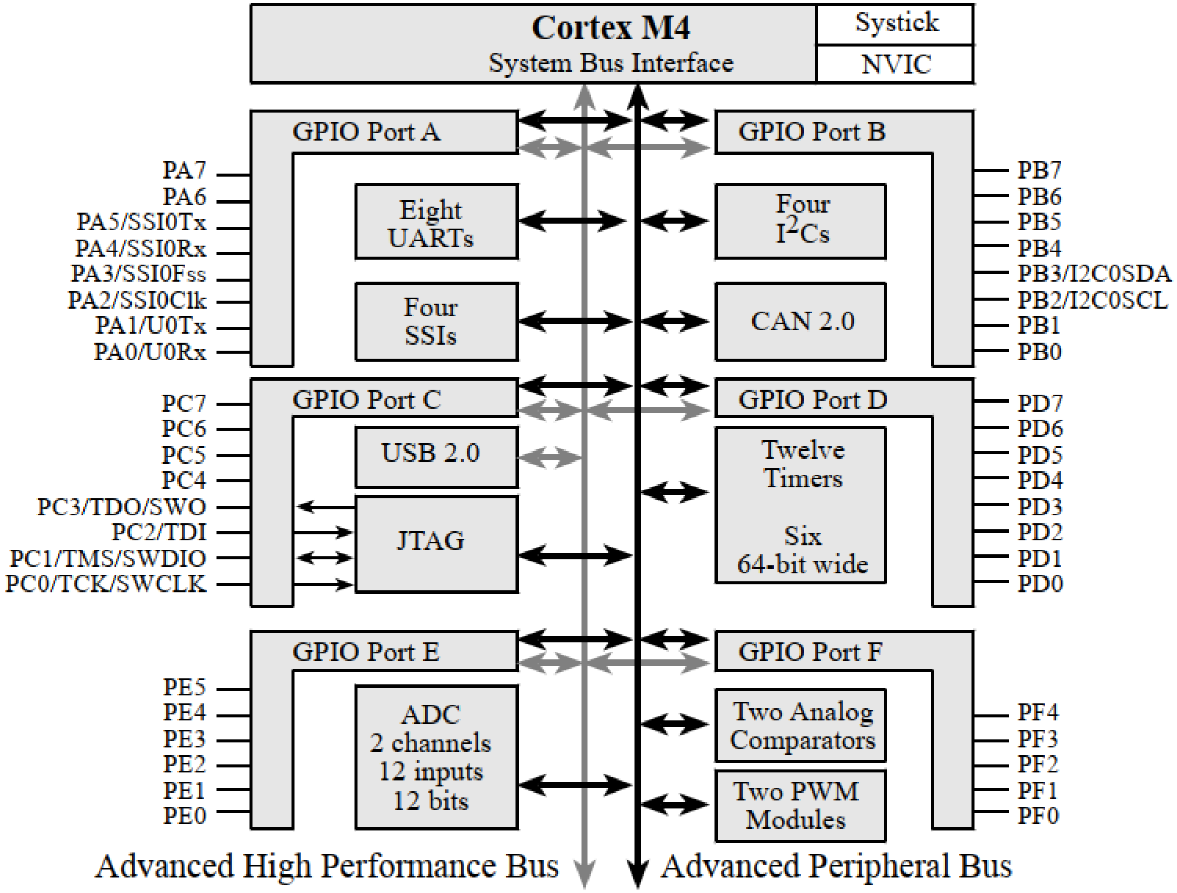
CSE 211: Introduction to Embedded Systems

Section 5

I/O port pins for TM4C123GH6PM microcontrollers.



Switches and LEDs on Tiva C Board

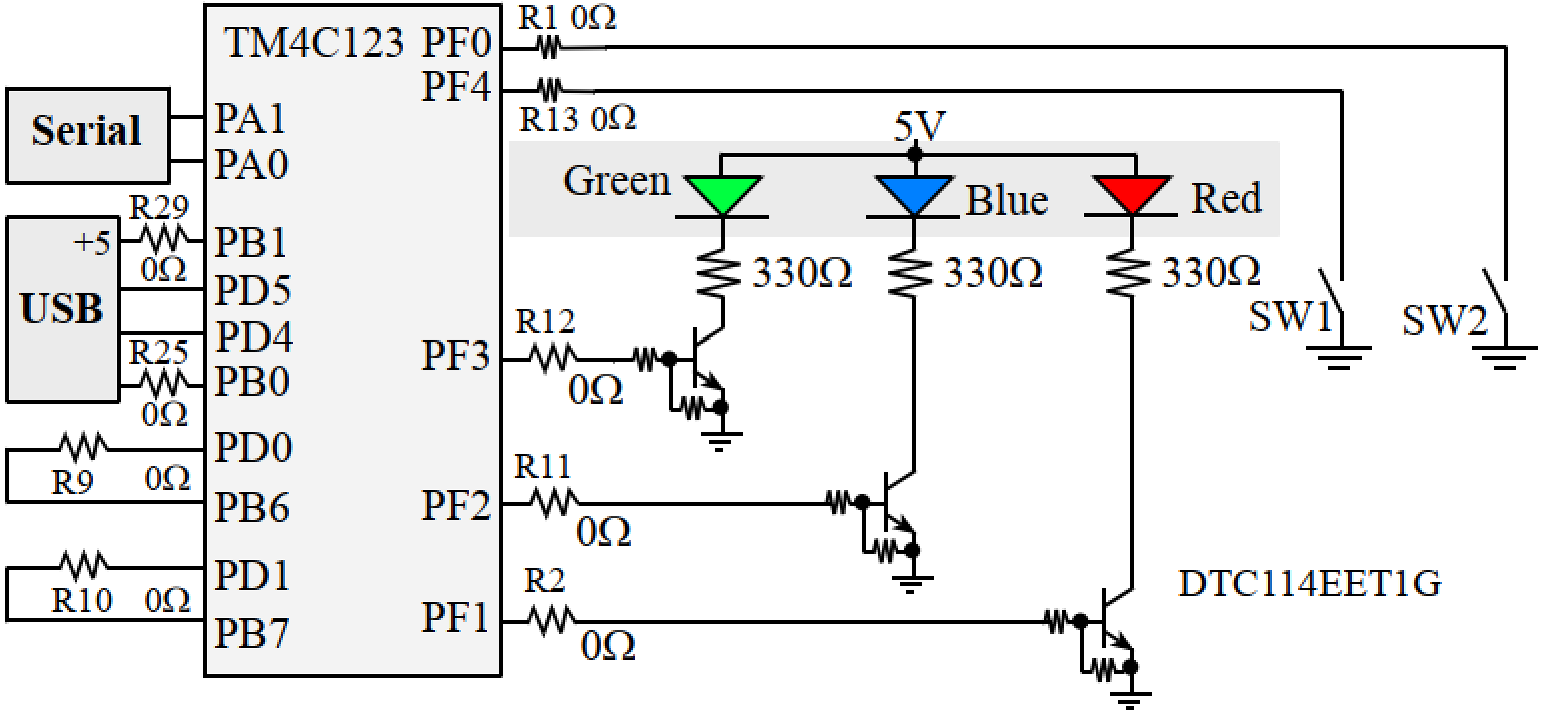
TIVAC LaunchPad has

- Two build-in switches:

  1. SW 1 (PF4)
  2. SW 2(PF0)

- Three LEDS:

  1.Red(PF1)
  2.Blue(PF2)
  3.Green(PF3)



Switch Interfacing

- The switches are negative logic and will require activation of the internal pull-up resistors.

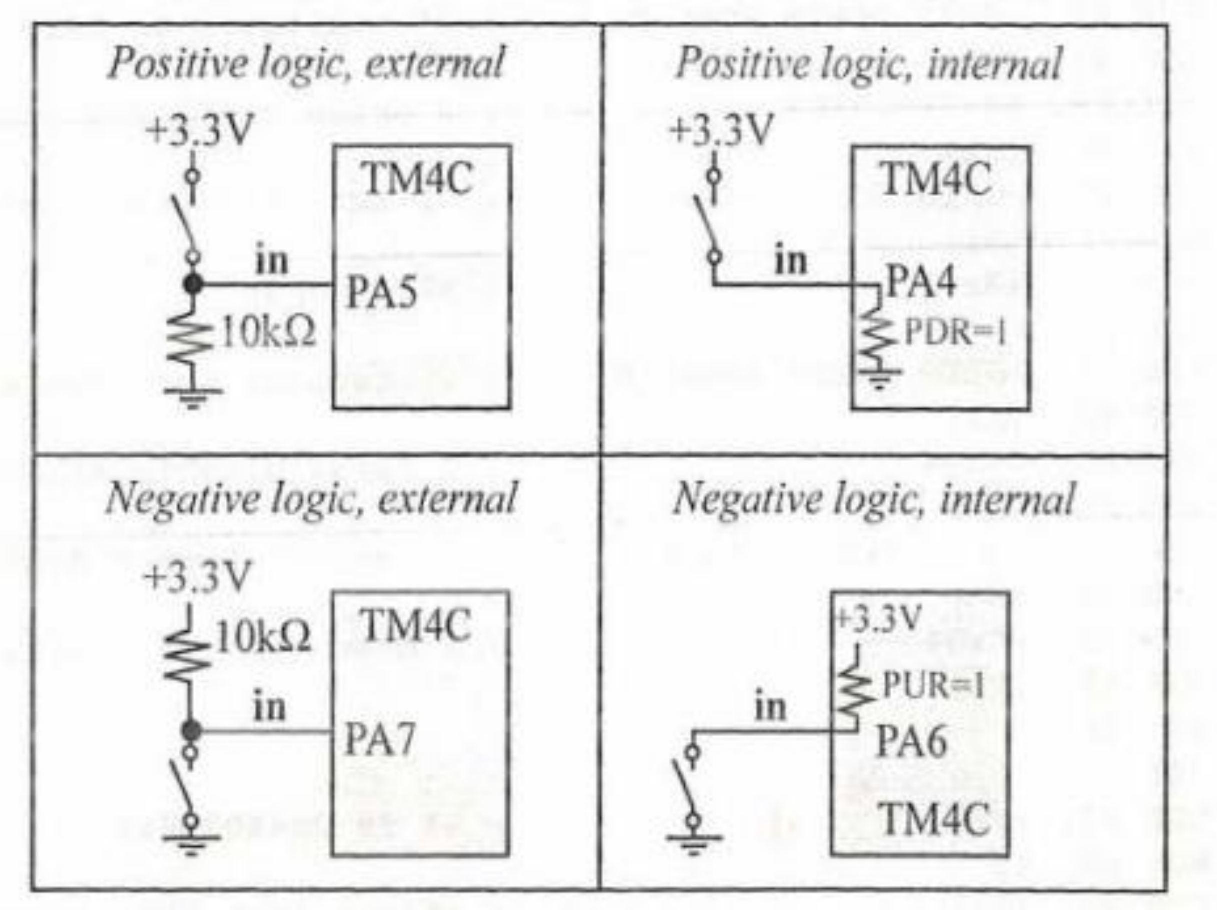
- You will set bits 0 and 4in GPIO_PORTF_PUR_R register.

- The LED interfaces on PF3 – PF1 are positive logic.

- To use the LED, make the PF3 – PF1 pins an output.

- To activate the red color, output a one to PF1.

- The blue color is on PF2, and the green color is controlled by PF3.



Switch Debouncing

- Mechanical switches may bounce when changing state
- We debounce the switch using time delay

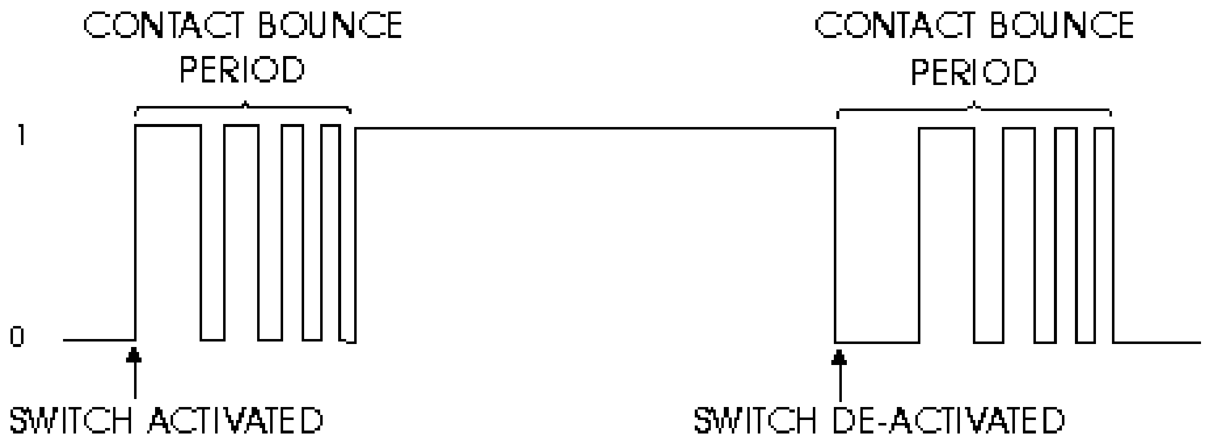
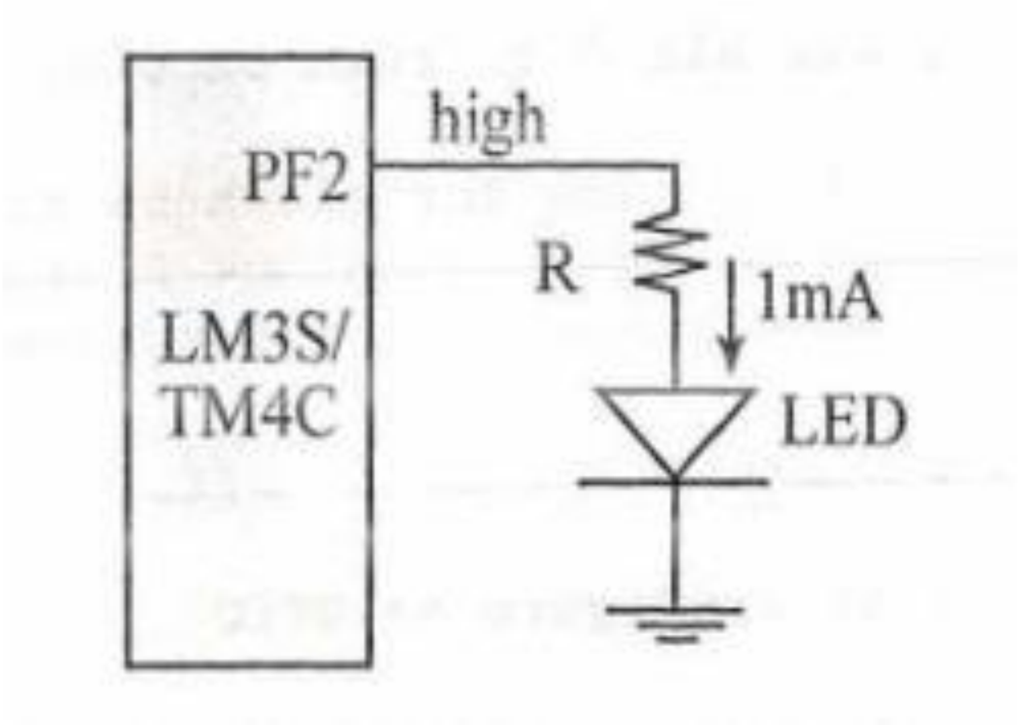
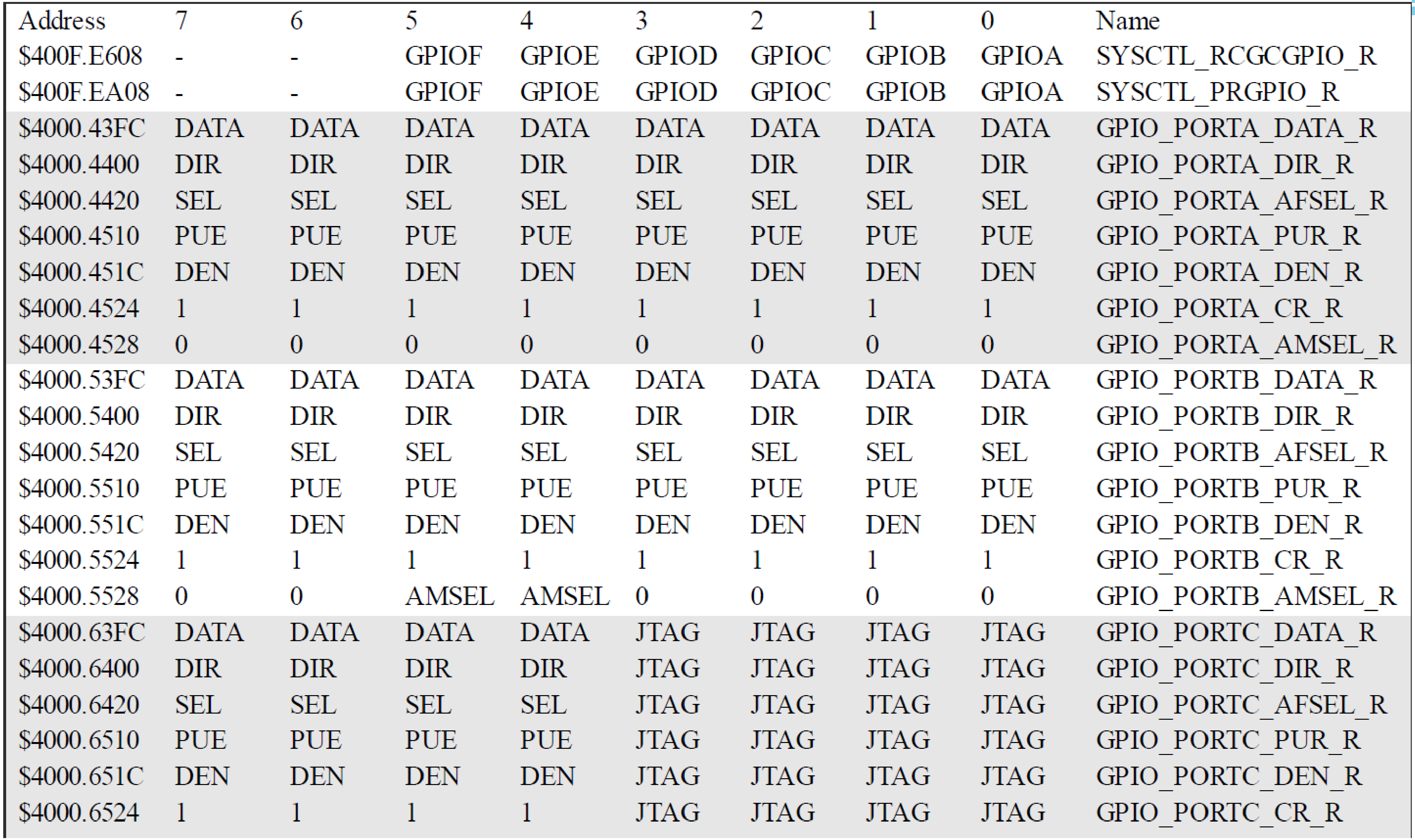


Figure 1

LED Interfacing





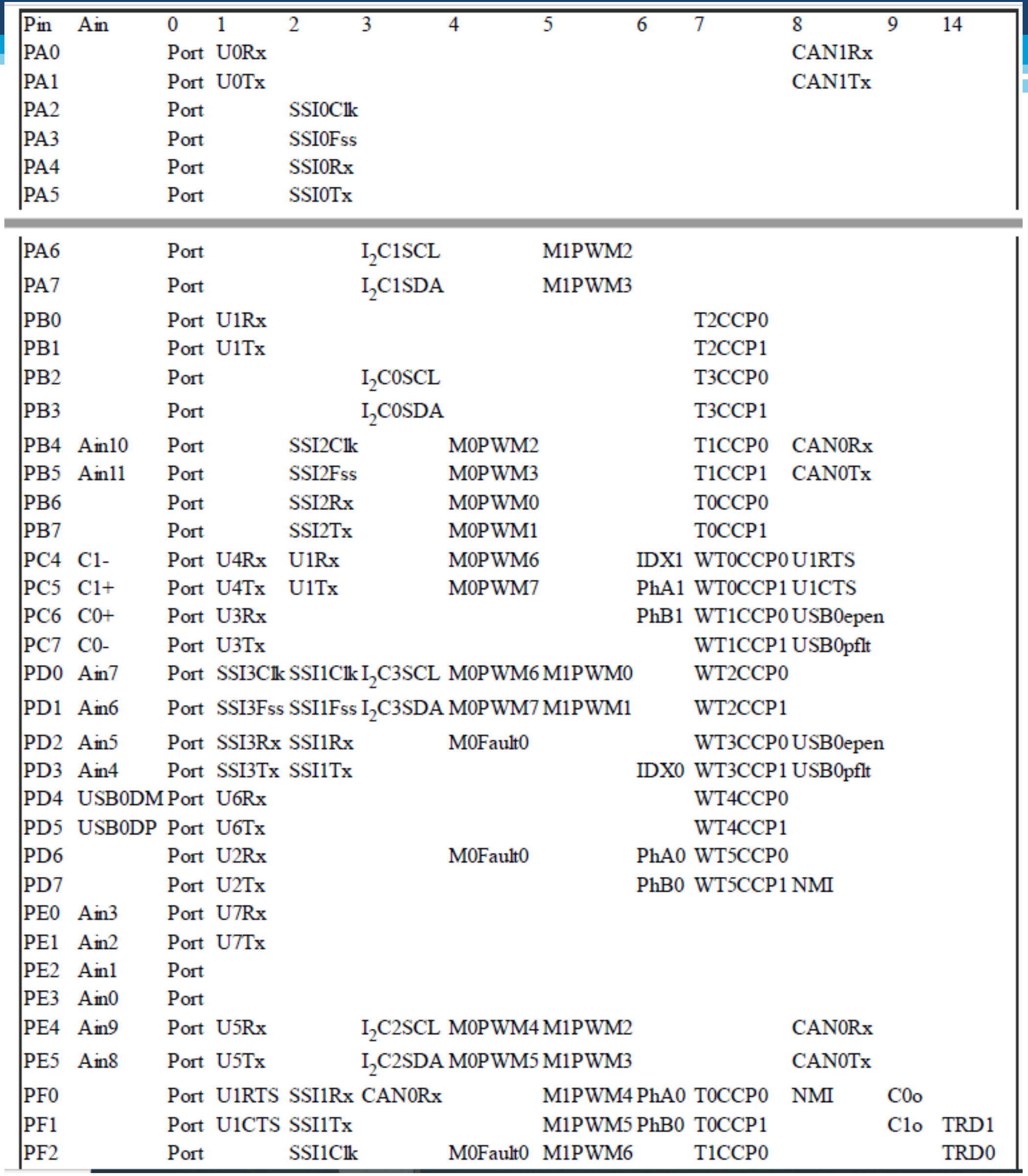
Address	7	6	5	4	3	2	1	0	Name
$400F.E608	-	-	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	SYSCTL_RCGCGPIO_R
$400F.EA08	-	-	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	SYSCTL_PRGPIO_R
$4000.43FC	DATA	DATA	DATA	DATA	DATA	DATA	DATA	DATA	GPIO_PORTA_DATA_R
$4000.4400	DIR	DIR	DIR	DIR	DIR	DIR	DIR	DIR	GPIO_PORTA_DIR_R
$4000.4420	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	GPIO_PORTA_AFSEL_R
$4000.4510	PUE	PUE	PUE	PUE	PUE	PUE	PUE	PUE	GPIO_PORTA_PUR_R
$4000.451C	DEN	DEN	DEN	DEN	DEN	DEN	DEN	DEN	GPIO_PORTA_DEN_R
$4000.4524	1	1	1	1	1	1	1	1	GPIO_PORTA_CR_R
$4000.4528	0	0	0	0	0	0	0	0	GPIO_PORTA_AMSEL_R
$4000.53FC	DATA	DATA	DATA	DATA	DATA	DATA	DATA	DATA	GPIO_PORTB_DATA_R
$4000.5400	DIR	DIR	DIR	DIR	DIR	DIR	DIR	DIR	GPIO_PORTB_DIR_R
$4000.5420	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	GPIO_PORTB_AFSEL_R
$4000.5510	PUE	PUE	PUE	PUE	PUE	PUE	PUE	PUE	GPIO_PORTB_PUR_R
$4000.551C	DEN	DEN	DEN	DEN	DEN	DEN	DEN	DEN	GPIO_PORTB_DEN_R
$4000.5524	1	1	1	1	1	1	1	1	GPIO_PORTB_CR_R
$4000.5528	0	0	AMSEL	AMSEL	0	0	0	0	GPIO_PORTB_AMSEL_R
$4000.63FC	DATA	DATA	DATA	DATA	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_DATA_R
$4000.6400	DIR	DIR	DIR	DIR	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_DIR_R
$4000.6420	SEL	SEL	SEL	SEL	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_AFSEL_R
$4000.6510	PUE	PUE	PUE	PUE	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_PUR_R
$4000.651C	DEN	DEN	DEN	DEN	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_DEN_R
$4000.6524	1	1	1	1	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_CR_R

Registers of Ports

- **GPIO_PORTF_DIR_R**: it sets the direction register to specify which pins are input and which are output.
- **GPIO_PORTF_AFSEL_R**: to activate the alternate functions.
- **GPIO_PORTF_DEN_R**: to use a pin as a digital input or output.
- **GPIO_PORTF_AMSEL_R**: To use a pin as an analog input.
- **SYSCTL_RCGCGPIO_R**: Each of the ports has a clock, which can be separately enabled by writing to it.
- **SYSCTL_PRGPIO_R**: Because it takes time for the clock to stabilize, we will wait for its status bit in the PRGPIO to be true.

Configurations of Ports

- PCTL values
  - Each pin also has four bits in the **PCTL** register, which we set to specify the alternative function for that pin (0 means regular I/O port).



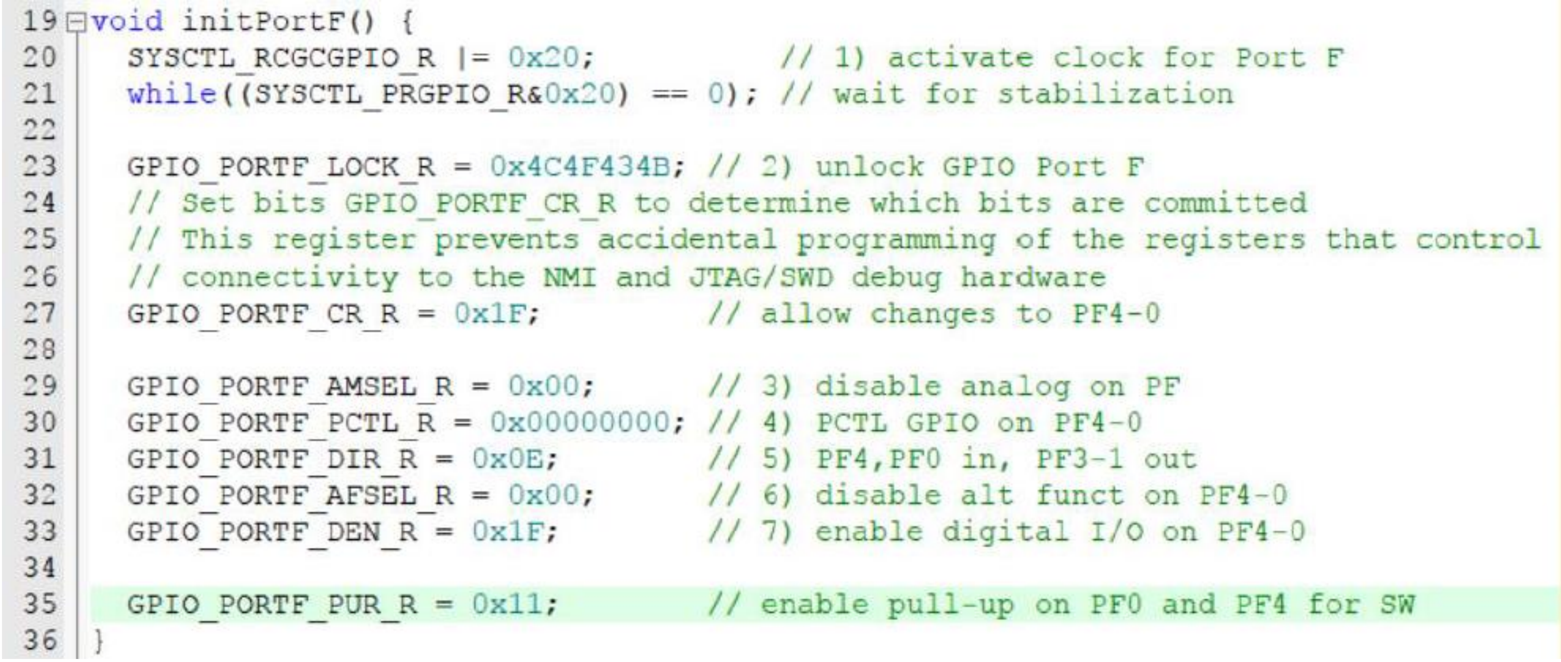
Pin	Ain	0	1	2	3	4	5	6	7	8	9	14
PA0		Port	U0Rx							CAN1Rx		
PA1		Port	U0Tx							CAN1Tx		
PA2		Port		SSI0Clk								
PA3		Port		SSI0Fss								
PA4		Port		SSI0Rx								
PA5		Port		SSI0Tx								
PA6		Port			I_2C1SCL		M1PWM2					
PA7		Port			I_2C1SDA		M1PWM3					
PB0		Port	U1Rx						T2CCP0			
PB1		Port	U1Tx						T2CCP1			
PB2		Port			I_2C0SCL				T3CCP0			
PB3		Port			I_2C0SDA				T3CCP1			
PB4	Ain10	Port		SSI2Clk		M0PWM2			T1CCP0	CAN0Rx		
PB5	Ain11	Port		SSI2Fss		M0PWM3			T1CCP1	CAN0Tx		
PB6		Port		SSI2Rx		M0PWM0			T0CCP0			
PB7		Port		SSI2Tx		M0PWM1			T0CCP1			
PC4	C1-	Port	U4Rx	U1Rx		M0PWM6		IDX1	WT0CCP0	U1RTS		
PC5	C1+	Port	U4Tx	U1Tx		M0PWM7		PhA1	WT0CCP1	U1CTS		
PC6	C0+	Port	U3Rx					PhB1	WT1CCP0	USB0epen		
PC7	C0-	Port	U3Tx						WT1CCP1	USB0pflt		
PD0	Ain7	Port	SSI3Clk	SSI1Clk	I_2C3SCL	M0PWM6	M1PWM0		WT2CCP0			
PD1	Ain6	Port	SSI3Fss	SSI1Fss	I_2C3SDA	M0PWM7	M1PWM1		WT2CCP1			
PD2	Ain5	Port	SSI3Rx	SSI1Rx		M0Fault0			WT3CCP0	USB0epen		
PD3	Ain4	Port	SSI3Tx	SSI1Tx				IDX0	WT3CCP1	USB0pflt		
PD4	USB0DM	Port	U6Rx						WT4CCP0			
PD5	USB0DP	Port	U6Tx						WT4CCP1			
PD6		Port	U2Rx			M0Fault0		PhA0	WT5CCP0			
PD7		Port	U2Tx					PhB0	WT5CCP1	NMI		
PE0	Ain3	Port	U7Rx									
PE1	Ain2	Port	U7Tx									
PE2	Ain1	Port										
PE3	Ain0	Port										
PE4	Ain9	Port	U5Rx		I_2C2SCL	M0PWM4	M1PWM2			CAN0Rx		
PE5	Ain8	Port	U5Tx		I_2C2SDA	M0PWM5	M1PWM3			CAN0Tx		
PF0		Port	U1RTS	SSI1Rx	CAN0Rx		M1PWM4	PhA0	T0CCP0	NMI	C0o	
PF1		Port	U1CTS	SSI1Tx			M1PWM5	PhB0	T0CCP1		C1o	TRD1
PF2		Port		SSI1Clk		M0Fault0	M1PWM6		T1CCP0			TRD0

GPIO Activation

- To activate a GPIO port for digital I/O, we need to do 7 steps
- 1. Activate the clock RCGCGPIO and wait for its status bit in PRGPIO
- 2. (optional) Unlock pins PD7 and PF0
- 3. Disable the analog function AMSEL_R
- 4. Disable alternate function AFSEL_R
- 5. Enable digital port DEN_R
- 6. Clear PCTL_R to select digital function (4 bits/pin)
- 7. Set direction register DIR_R (0 for In, 1 for out)

Activation Example

- Code Example to initialize Port F for regular digital I/O. Bits 0 and 4 as input, Bits 1 -3 as output (Tiva C: Switches and LEDs)



```
void initPortF() {
  SYSCTL_RCGCGPIO_R |= 0x20;          // 1) activate clock for Port F
  while((SYSCTL_PRGPIO_R&0x20) == 0); // wait for stabilization

  GPIO_PORTF_LOCK_R = 0x4C4F434B; // 2) unlock GPIO Port F
  // Set bits GPIO_PORTF_CR_R to determine which bits are committed
  // This register prevents accidental programming of the registers that control
  // connectivity to the NMI and JTAG/SWD debug hardware
  GPIO_PORTF_CR_R = 0x1F;         // allow changes to PF4-0

  GPIO_PORTF_AMSEL_R = 0x00;      // 3) disable analog on PF
  GPIO_PORTF_PCTL_R = 0x00000000; // 4) PCTL GPIO on PF4-0
  GPIO_PORTF_DIR_R = 0x0E;        // 5) PF4,PF0 in, PF3-1 out
  GPIO_PORTF_AFSEL_R = 0x00;      // 6) disable alt funct on PF4-0
  GPIO_PORTF_DEN_R = 0x1F;        // 7) enable digital I/O on PF4-0

  GPIO_PORTF_PUR_R = 0x11;        // enable pull-up on PF0 and PF4 for SW
}
```

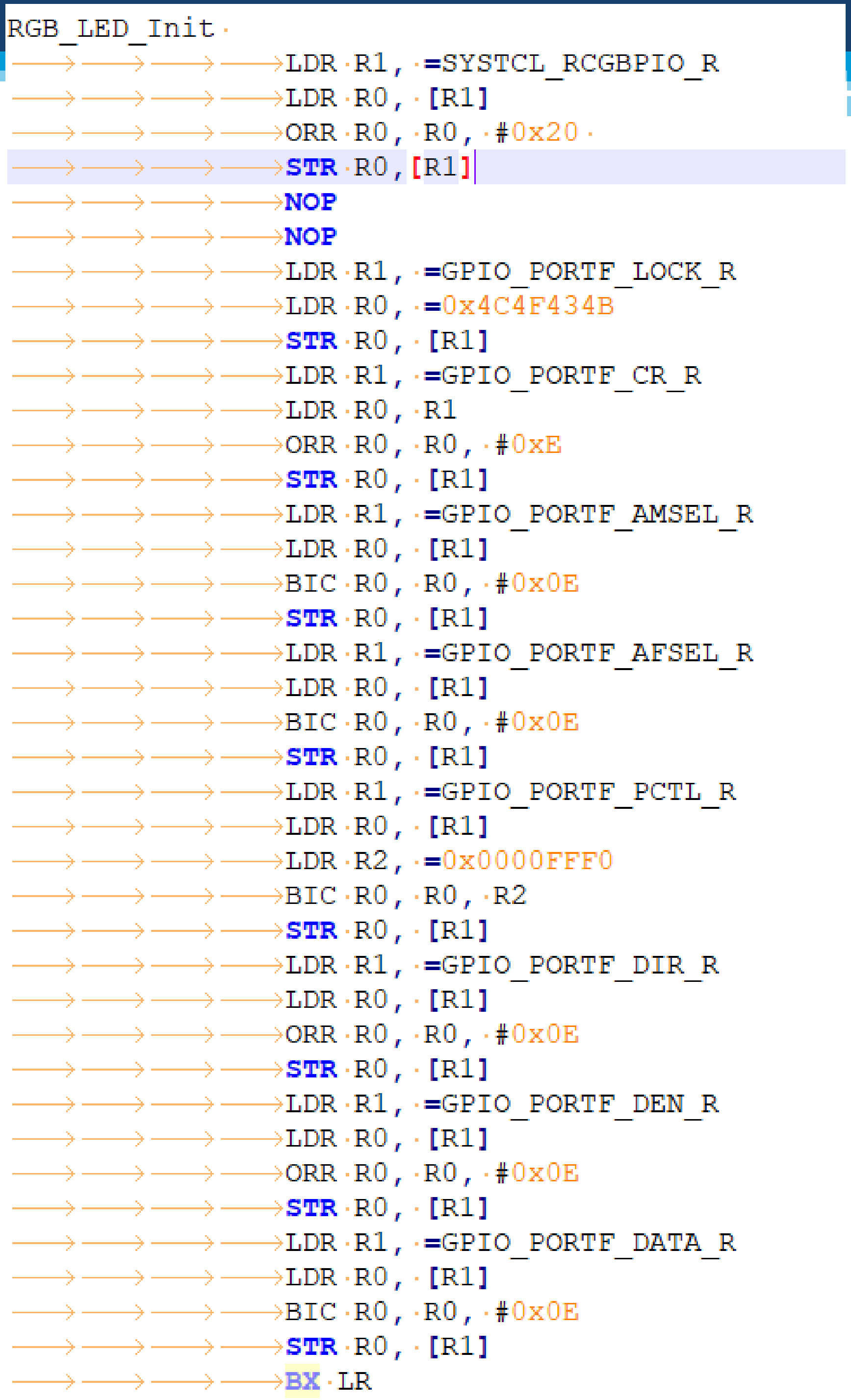
Setting UART in port E

- Use UART7 on pins PE0 and PE1
- Set bits 1,0 in the DEN register (enable digital)
- Clear bits 1,0 in the AMSEL register (disable analog)
- Write a 0001,0001 to bits 7–0 in the PCTL register (enable UART7 functionality)
- Set bits 1,0 in the AFSEL register (enable alternate function)

Sheet 4

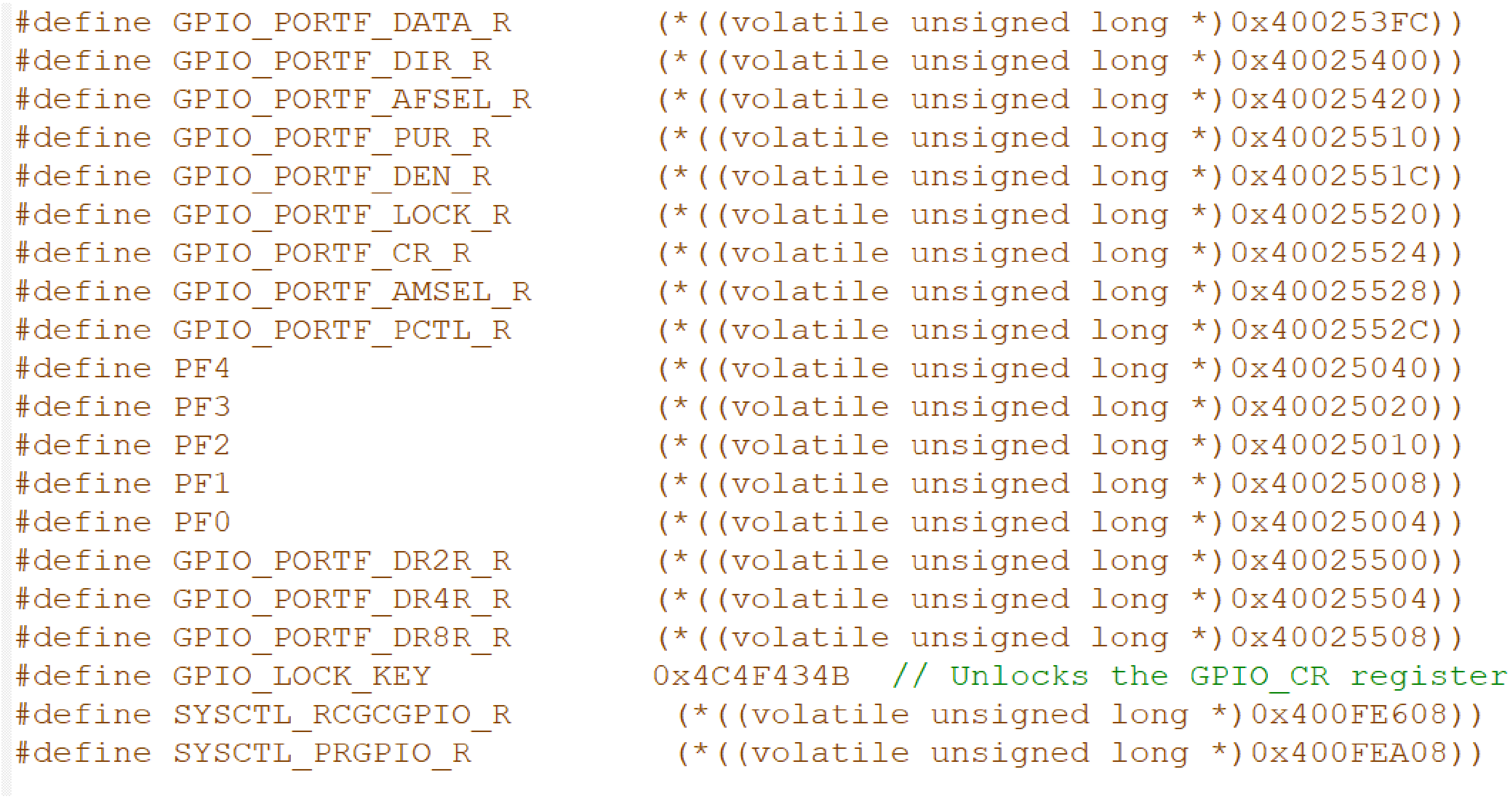
Write an assembly /C function that initializes port F pins 1, 2, and 3 as Digital Output with initial zero values.

Assembly



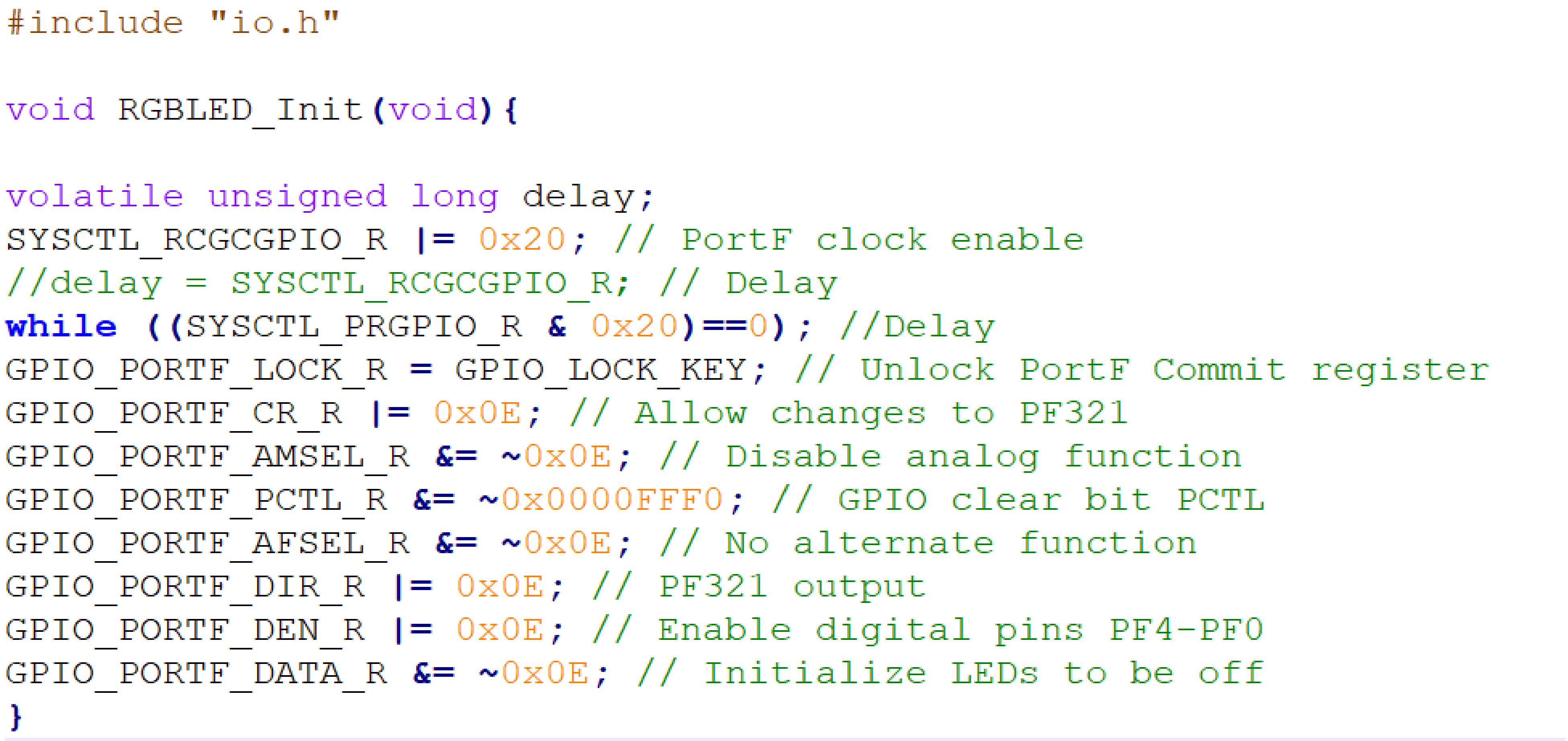
```
RGB_LED_Init
				LDR R1, =SYSTCL_RCGBPIO_R
				LDR R0, [R1]
				ORR R0, R0, #0x20
				STR R0,[R1]
				NOP
				NOP
				LDR R1, =GPIO_PORTF_LOCK_R
				LDR R0, =0x4C4F434B
				STR R0, [R1]
				LDR R1, =GPIO_PORTF_CR_R
				LDR R0, R1
				ORR R0, R0, #0xE
				STR R0, [R1]
				LDR R1, =GPIO_PORTF_AMSEL_R
				LDR R0, [R1]
				BIC R0, R0, #0x0E
				STR R0, [R1]
				LDR R1, =GPIO_PORTF_AFSEL_R
				LDR R0, [R1]
				BIC R0, R0, #0x0E
				STR R0, [R1]
				LDR R1, =GPIO_PORTF_PCTL_R
				LDR R0, [R1]
				LDR R2, =0x0000FFF0
				BIC R0, R0, R2
				STR R0, [R1]
				LDR R1, =GPIO_PORTF_DIR_R
				LDR R0, [R1]
				ORR R0, R0, #0x0E
				STR R0, [R1]
				LDR R1, =GPIO_PORTF_DEN_R
				LDR R0, [R1]
				ORR R0, R0, #0x0E
				STR R0, [R1]
				LDR R1, =GPIO_PORTF_DATA_R
				LDR R0, [R1]
				BIC R0, R0, #0x0E
				STR R0, [R1]
				BX LR
```

Header File (io.h)



```
#define GPIO_PORTF_DATA_R       (*((volatile unsigned long *)0x400253FC))
#define GPIO_PORTF_DIR_R        (*((volatile unsigned long *)0x40025400))
#define GPIO_PORTF_AFSEL_R      (*((volatile unsigned long *)0x40025420))
#define GPIO_PORTF_PUR_R        (*((volatile unsigned long *)0x40025510))
#define GPIO_PORTF_DEN_R        (*((volatile unsigned long *)0x4002551C))
#define GPIO_PORTF_LOCK_R       (*((volatile unsigned long *)0x40025520))
#define GPIO_PORTF_CR_R         (*((volatile unsigned long *)0x40025524))
#define GPIO_PORTF_AMSEL_R      (*((volatile unsigned long *)0x40025528))
#define GPIO_PORTF_PCTL_R       (*((volatile unsigned long *)0x4002552C))
#define PF4                     (*((volatile unsigned long *)0x40025040))
#define PF3                     (*((volatile unsigned long *)0x40025020))
#define PF2                     (*((volatile unsigned long *)0x40025010))
#define PF1                     (*((volatile unsigned long *)0x40025008))
#define PF0                     (*((volatile unsigned long *)0x40025004))
#define GPIO_PORTF_DR2R_R       (*((volatile unsigned long *)0x40025500))
#define GPIO_PORTF_DR4R_R       (*((volatile unsigned long *)0x40025504))
#define GPIO_PORTF_DR8R_R       (*((volatile unsigned long *)0x40025508))
#define GPIO_LOCK_KEY           0x4C4F434B  // Unlocks the GPIO_CR register
#define SYSCTL_RCGCGPIO_R        (*((volatile unsigned long *)0x400FE608))
#define SYSCTL_PRGPIO_R          (*((volatile unsigned long *)0x400FEA08))
```

C-Function

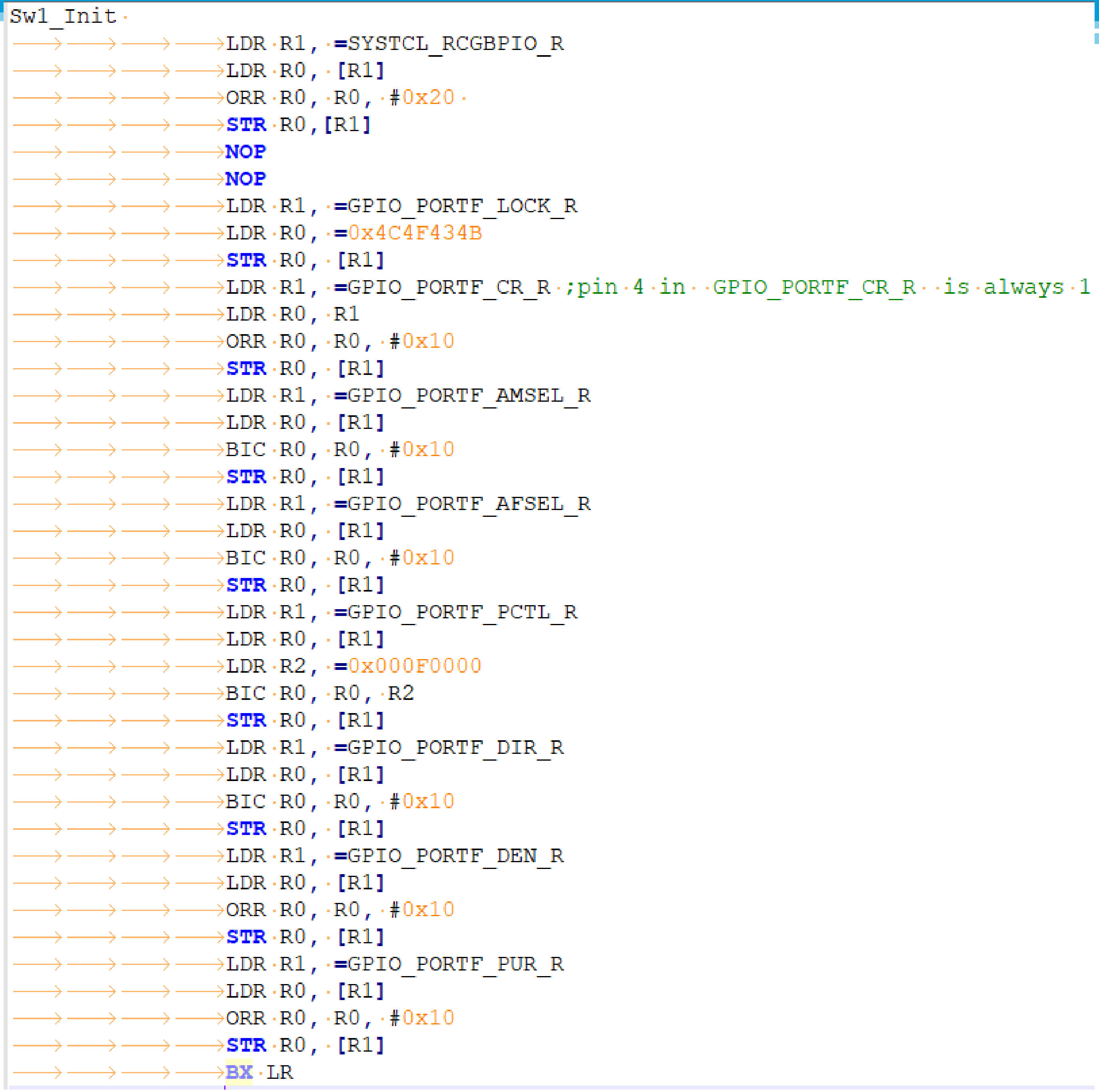


```
#include "io.h"

void RGBLED_Init(void){

volatile unsigned long delay;
SYSCTL_RCGCGPIO_R |= 0x20; // PortF clock enable
//delay = SYSCTL_RCGCGPIO_R; // Delay
while ((SYSCTL_PRGPIO_R & 0x20)==0); //Delay
GPIO_PORTF_LOCK_R = GPIO_LOCK_KEY; // Unlock PortF Commit register
GPIO_PORTF_CR_R |= 0x0E; // Allow changes to PF321
GPIO_PORTF_AMSEL_R &= ~0x0E; // Disable analog function
GPIO_PORTF_PCTL_R &= ~0x0000FFF0; // GPIO clear bit PCTL
GPIO_PORTF_AFSEL_R &= ~0x0E; // No alternate function
GPIO_PORTF_DIR_R |= 0x0E; // PF321 output
GPIO_PORTF_DEN_R |= 0x0E; // Enable digital pins PF4-PF0
GPIO_PORTF_DATA_R &= ~0x0E; // Initialize LEDs to be off
}
```

Sheet 4

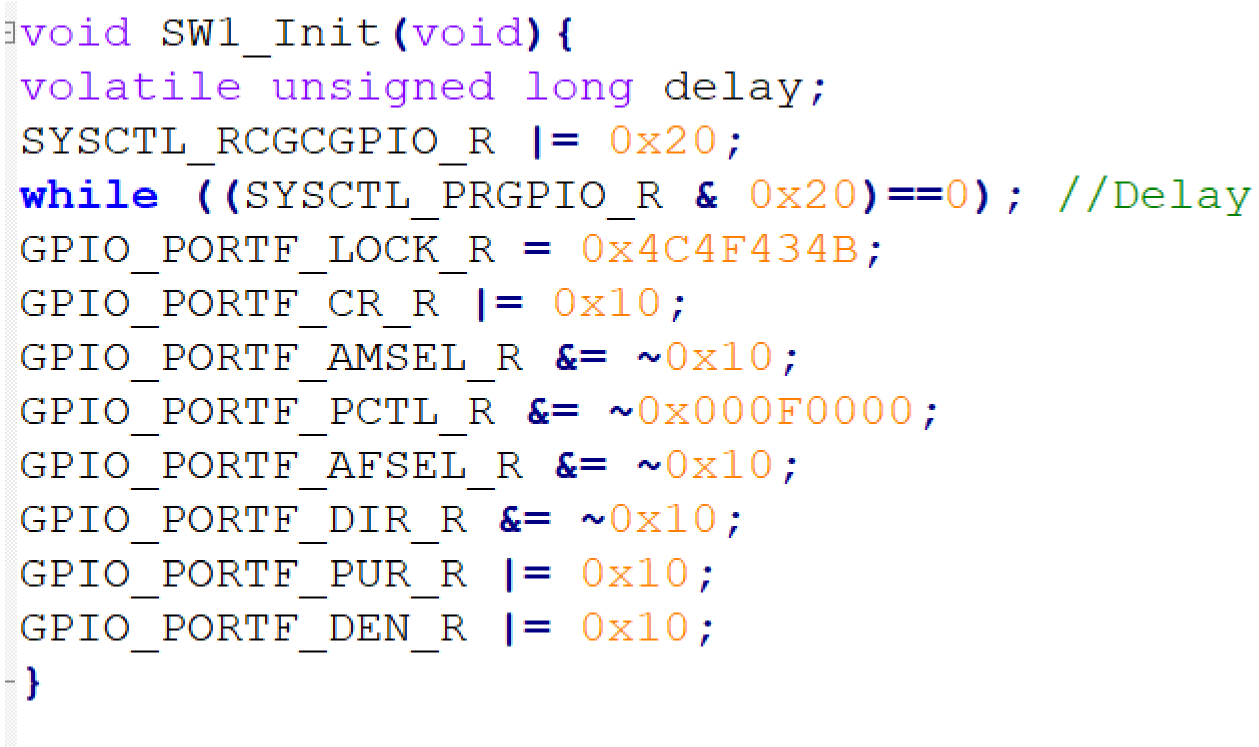
- Write an assembly/C function that initializes port F pin 4 as Digital Input that will be connected to a switch.

Assembly



```
Sw1_Init
                LDR R1, =SYSTCL_RCGBPIO_R
                LDR R0, [R1]
                ORR R0, R0, #0x20
                STR R0,[R1]
                NOP
                NOP
                LDR R1, =GPIO_PORTF_LOCK_R
                LDR R0, =0x4C4F434B
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_CR_R ;pin 4 in  GPIO_PORTF_CR_R  is always 1
                LDR R0, R1
                ORR R0, R0, #0x10
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_AMSEL_R
                LDR R0, [R1]
                BIC R0, R0, #0x10
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_AFSEL_R
                LDR R0, [R1]
                BIC R0, R0, #0x10
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_PCTL_R
                LDR R0, [R1]
                LDR R2, =0x000F0000
                BIC R0, R0, R2
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_DIR_R
                LDR R0, [R1]
                BIC R0, R0, #0x10
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_DEN_R
                LDR R0, [R1]
                ORR R0, R0, #0x10
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_PUR_R
                LDR R0, [R1]
                ORR R0, R0, #0x10
                STR R0, [R1]
                BX LR
```

C-Function

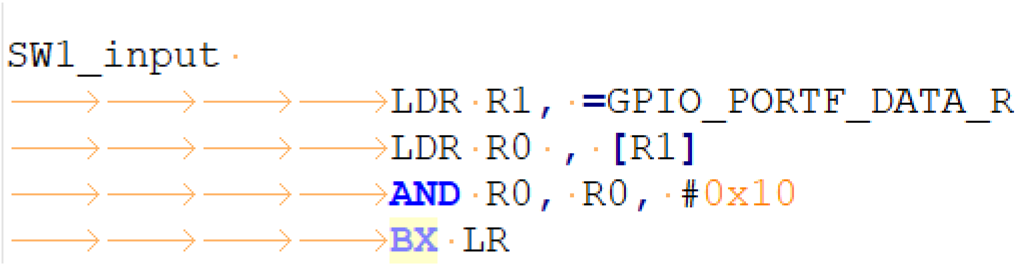


```c
void SW1_Init(void){
volatile unsigned long delay;
SYSCTL_RCGCGPIO_R |= 0x20;
while ((SYSCTL_PRGPIO_R & 0x20)==0); //Delay
GPIO_PORTF_LOCK_R = 0x4C4F434B;
GPIO_PORTF_CR_R |= 0x10;
GPIO_PORTF_AMSEL_R &= ~0x10;
GPIO_PORTF_PCTL_R &= ~0x000F0000;
GPIO_PORTF_AFSEL_R &= ~0x10;
GPIO_PORTF_DIR_R &= ~0x10;
GPIO_PORTF_PUR_R |= 0x10;
GPIO_PORTF_DEN_R |= 0x10;
}
```

Sheet 4

- Write an assembly /C function that reads PORTF pin4.

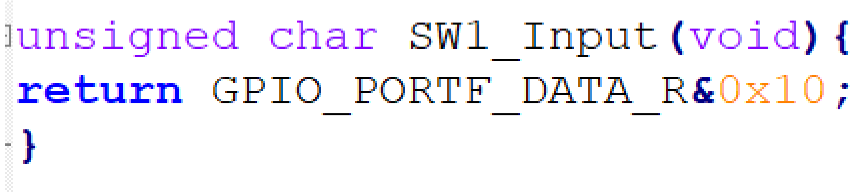
Assembly



```
SW1_input
				LDR R1, =GPIO_PORTF_DATA_R
				LDR R0 , [R1]
				AND R0, R0, #0x10
				BX LR
```

C-Function

- Write an C function that reads PORTF pin4.

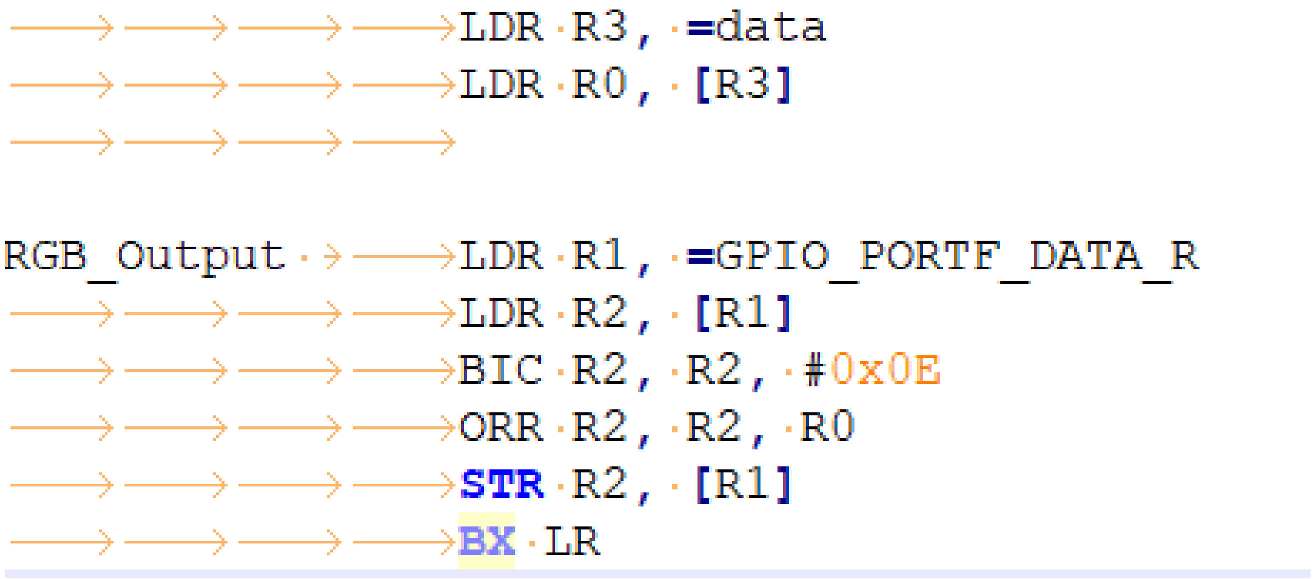


```
unsigned char SW1_Input(void){
return GPIO_PORTF_DATA_R&0x10;
}
```

Sheet 4

- Write an C function that clears pin1, pin2, and pin3 then update the mentioned pins with new values of data in PORTF.

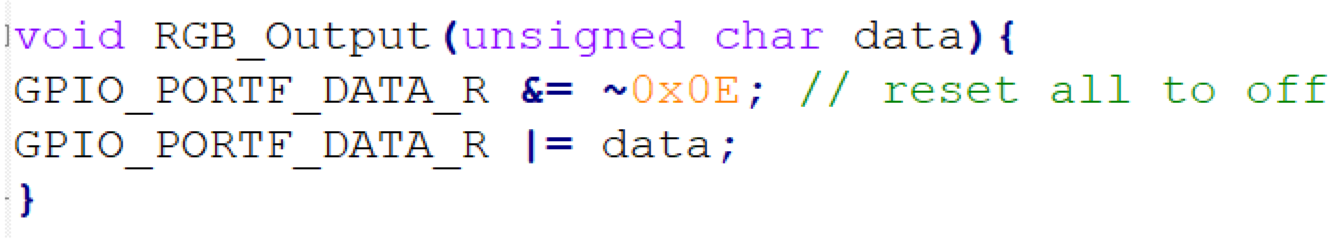
Assembly



```
                LDR R3, =data
                LDR R0, [R3]


RGB_Output      LDR R1, =GPIO_PORTF_DATA_R
                LDR R2, [R1]
                BIC R2, R2, #0x0E
                ORR R2, R2, R0
                STR R2, [R1]
                BX LR
```

C-Function

- Write an C function that clears pin1, pin2, and pin3 then update the mentioned pins with new values of data in PORTF.

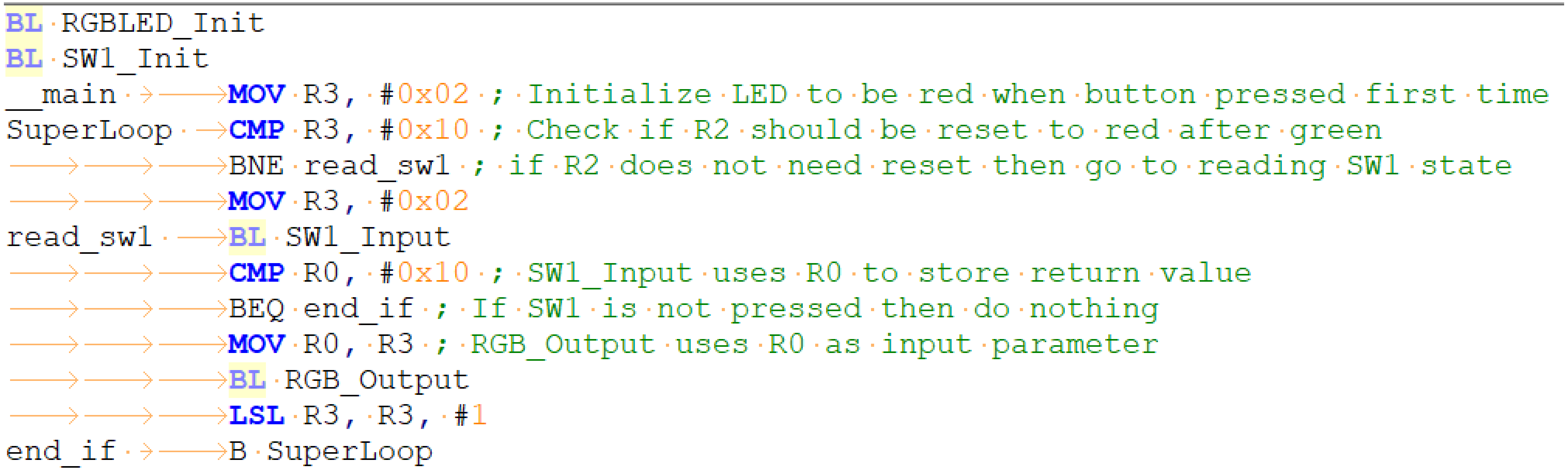


```
void RGB_Output(unsigned char data){
GPIO_PORTF_DATA_R &= ~0x0E; // reset all to off
GPIO_PORTF_DATA_R |= data;
}
```

Sheet 4

- In Tiva C, PF4 is connected to a push button and PF1, PF2, and PF3 are connected to an RGB LED. PF1 is red, PF2 is blue, and PF3 is green. Write assembly application that uses the init functions developed in previous questions that reads input from the switch and when it is pressed for the first time the red LED should be turned on then when pressed a second time turn off the red LED and turn on the blue LED then when pressed a third time turn off the blue LED and turn on the green LED then when pressed a fourth time turn off the green LED and turn on again the red LED and then repeat the cycle.

Assembly



```
BL RGBLED_Init
BL SW1_Init
__main      MOV R3, #0x02 ; Initialize LED to be red when button pressed first time
SuperLoop   CMP R3, #0x10 ; Check if R2 should be reset to red after green
            BNE read_sw1 ; if R2 does not need reset then go to reading SW1 state
            MOV R3, #0x02
read_sw1    BL SW1_Input
            CMP R0, #0x10 ; SW1_Input uses R0 to store return value
            BEQ end_if ; If SW1 is not pressed then do nothing
            MOV R0, R3 ; RGB_Output uses R0 as input parameter
            BL RGB_Output
            LSL R3, R3, #1
end_if      B SuperLoop
```

Thank You