



Faculty of Engineering

Ain Shams University

CSE 211s: Introduction to Embedded Systems

REPORT (4)

Name: Habiba El-Sayed Mowafy

Sec.: 1

Program: CSE

ID: 2100792

Question 2:

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.

C code:

```
#include "tm4c123gh6pm.h"
#include <string.h>
#include <stdio.h>
#define CR 0x0D

void UART_Init (void) {
    SYSTCTL_RCGCUART_R |= 0x0001;
    SYSTCTL_RCGCGPIO_R |= 0x0001;
    UART0_CTL_R &= ~0x0001;
    UART0_IBRD_R = 0x68;
    UART0_FBRD_R = 0xB;
    UART0_LCRH_R |= 0x0070;
    UART0_CTL_R &= 0x0301;
    GPIO_PORTA_AFSEL_R |= 0x03;
    GPIO_PORTA_PCTL_R = (GPIO_PORTA_PCTL_R & 0xFFFFFFF0) + 0x00000011;
    GPIO_PORTA_DEN_R |= 0x03;
    GPIO_PORTA_AMSEL_R &= ~0x03;
}

void RGB_Init(void) {
    SYSTCTL_RCGCGPIO_R |= 0x20; //enable clk for port F
    while ((SYSTCTL_PRGPIO_R & 0x20) == 0); //check for the clk delay
    GPIO_PORTF_LOCK_R = GPIO_LOCK_KEY; //break the lock of port F
    GPIO_PORTF_CR_R |= 0x0E; //enable making changes in leds pins (PF1,PF2&PF3)
    GPIO_PORTF_AMSEL_R &= ~0x0E; //disable analog mode
    GPIO_PORTF_AFSEL_R &= ~0x0E; //disable using alternate functions (use the pins
as GPIO)
    GPIO_PORTF_PCTL_R &= ~0x0000FFF0; //clear the bits corresponfing to the 3 pins
    GPIO_PORTF_DEN_R |= 0x0E; //enable using the pins in digital mode
    GPIO_PORTF_DIR_R |= 0x0E; //set the bits corresponding to the pins to 1 to use
them as output
    GPIO_PORTF_DATA_R &= ~0x0E; //initialize the leds to be off
}
```

```

void SysTick_wait (){
    NVIC_ST_CTRL_R = 0x00;
    NVIC_ST_RELOAD_R = 16000 -1;
    NVIC_ST_CURRENT_R = 0x00;
    NVIC_ST_CTRL_R = 0x05;
    while ((NVIC_ST_CTRL_R & 0x00010000) == 0); //wait for count flag
}

void delay (int time){
    int i;
    for (i = 0; i <= time; i++){
        SysTick_wait ();
    }
}

char UART_INchar (void){
    while ((UART0_FR_R & 0x0010) != 0);
    return (char) (UART0_DR_R & 0x00FF);
}

void UART_OUTchar (unsigned char data){
    while ((UART0_FR_R & 0x0020) != 0);
    UART0_DR_R = data;
}

void getCommand (char * command, int len){
    char ch;
    int i;
    for (i=0; i<len; i++){
        ch = UART_INchar ();
        if (ch != CR){
            command [i] = ch;
            UART_OUTchar (command [i]);
        }
        else if (ch == CR || i == len) break;
    }
}

void UART_OutString (char *pt){
    while (*pt){
        UART_OUTchar (*pt);
        pt++;
    }
}

void RGB_Output (unsigned char data){
    GPIO_PORTF_DATA_R |= data;
}

void RGB_CLR (unsigned char data){
    GPIO_PORTF_DATA_R &= ~data;
}

```

```

int main (){
    const int len = 9;
    char command [len] = {0};
    RGB_Init();
    UART_Init();
    RGB_CLR (0x0E);

    while (1){
        UART_OutString ("Enter a character: ");
        getCommand (command, len);

        if(strcmp(command, "A") == 0){
            RGB_CLR (0x0E);
            delay (60000);
            RGB_Output (0x02);
            memset(command, 0, len);
        }
        else if(strcmp(command, "B") == 0){
            RGB_CLR (0x0E);
            delay (30000);
            RGB_Output (0x04);
            memset(command, 0, len);
        }
        else if(strcmp(command, "D") == 0){
            RGB_CLR (0x0E);
            delay (120000);
            RGB_Output (0x08);
            memset(command, 0, len);
        }

        UART_OutString ("\n");
    }
}

```

SnapShots:

Program Started, all LEDs are OFF

Registers

Register	Value
R0	0x0000054
R1	0x2000020
R2	0x0000000
R3	0x0000068
R4	0x000006A
R5	0x000006A
R6	0x0000000
R7	0x0000000
R8	0x0000000
R9	0x0000000
R10	0x0000000
R11	0x0000000
R12	0x0000000
R13 (SP)	0x2000020
R14 (LR)	0x0000068
R15 (PC)	0x0000054
xPSR	0x6100000

Disassembly

```
81: int main () {
82:     const int len = 9;
83:     char command[len] = {0};
84:     RGB_Init();
85:     UART_Init();
86:     RGB_CLR (0x0E);
87:
88:     while (1) {
89:         UART_OutString ("Enter a character: ");
90:         getCommand (command, len);
91:
92:         if (strcmp (command, "A") == 0) {
93:             RGB_CLR (0x0E);
94:             SysTick_wait (60);
95:             RGB_Output (0x02);
96:         }
97:     }
98: }
```

GPIOF

Property	Value
DATA	0x0000011
DIR	0x000000E
IS	0x0000000
IBE	0x0000000
IEV	0x0000000
IM	0
RIS	0
MIS	0
ICR	0
AFSEL	0x0000000
DR2R	0x000000F
DR4R	0x0000000
DR8R	0x0000000
ODR	0x0000000
PUR	0x0000000
PDR	0x0000000

Command

*** Currently used: 1696 Bytes (5%)

UART #1

Enter a character:

UART #0

Enter a character:

Registers

Register	Value
R0	0x0000054
R1	0x2000020
R2	0x0000000
R3	0x0000068
R4	0x000006A
R5	0x000006A
R6	0x0000000
R7	0x0000000
R8	0x0000000
R9	0x0000000
R10	0x0000000
R11	0x0000000
R12	0x0000000
R13 (SP)	0x2000020
R14 (LR)	0x0000068
R15 (PC)	0x0000054
xPSR	0x6100000

Disassembly

```
81: int main () {
82:     const int len = 9;
83:     char command[len] = {0};
84:     RGB_Init();
85:     UART_Init();
86:     RGB_CLR (0x0E);
87:
88:     while (1) {
89:         UART_OutString ("Enter a character: ");
90:         getCommand (command, len);
91:
92:         if (strcmp (command, "A") == 0) {
93:             RGB_CLR (0x0E);
94:             SysTick_wait (60);
95:             RGB_Output (0x02);
96:         }
97:     }
98: }
```

GPIOF

Property	Value
DATA	0x0000011
DIR	0x000000E
IS	0x0000000
IBE	0x0000000
IEV	0x0000000
IM	0
RIS	0
MIS	0
ICR	0
AFSEL	0x0000000
DR2R	0x000000F
DR4R	0x0000000
DR8R	0x0000000
ODR	0x0000000
PUR	0x0000000
PDR	0x0000000

Command

*** Currently used: 1696 Bytes (5%)

UART #1

Enter a character:

UART #0

Enter a character:

Sending "A"

This screenshot shows the TI-RTOS IDE with the following components:

- Registers:** A list of registers with their current values. R0 is 0x0000054, R1 is 0x2000020, and xPSR is 0x6100000.
- Disassembly:** The assembly code for the main function. The instruction at address 0x00000542 is highlighted: `0x00000542 B50E PUSH {r1-r3,}`.
- Port F Hardware:** A diagram of the TM4C123 microcontroller showing the connection of SW1 to PF4, SW2 to PF0, and two LEDs (one green, one red) to PF3 and PF2 respectively. A 16 MHz clock is also shown.
- Port F Registers:** A table showing the current values of the Port F registers: DATA (0x13), DIR (0x0E), DEN (0x0E), PUR (0x00), PDR (0x00), RCGCGPIO (0x00000021), LOCK (0x00), and CR (0x1E). The clock is enabled.
- GPIOF:** A table showing the current values of the GPIOF registers: DATA (0x00000013), DIR (0x0000000E), IS (0x00000000), IBE (0x00000000), IEV (0x00000000), IM (0), RIS (0), MIS (0), ICR (0), AFSEL (0x00000000), DR2R (0x000000FF), DR4R (0x00000000), DR8R (0x00000000), ODR (0x00000000), PUR (0x00000000), and PDR (0x00000000).
- Command:** A text box showing the command "Enter a character: A".
- UART #1:** A text box showing the command "Enter a character: A".

This screenshot shows the TI-RTOS IDE with the following components:

- Registers:** A list of registers with their current values. R0 is 0x0000054, R1 is 0x2000020, and xPSR is 0x6100000.
- Disassembly:** The assembly code for the main function. The instruction at address 0x00000542 is highlighted: `0x00000542 B50E PUSH {r1-r3,}`.
- Port F Hardware:** A diagram of the TM4C123 microcontroller showing the connection of SW1 to PF4, SW2 to PF0, and two LEDs (one green, one red) to PF3 and PF2 respectively. A 16 MHz clock is also shown.
- Port F Registers:** A table showing the current values of the Port F registers: DATA (0x13), DIR (0x0E), DEN (0x0E), PUR (0x00), PDR (0x00), RCGCGPIO (0x00000021), LOCK (0x00), and CR (0x1E). The clock is enabled.
- GPIOF:** A table showing the current values of the GPIOF registers: DATA (0x00000020), RSR (0), ECR (0), FR (0x00000090), ILPR (0), IBRD (0x00000068), FBRD (0x00000008), LCRH (0x00000070), CTL (0), IFLS (0), IM (0), RIS (0), MIS (0), ICR (0), DMACTL (0), and _9BITADDR (0).
- Command:** A text box showing the command "Enter a character: A".
- UART #1:** A text box showing the command "Enter a character: A".

Sending "B"

The screenshot shows the TI-RTOS IDE with the following components:

- Registers:** A list of registers with their current values. R0 is 0x00000054, R1 is 0x20000020, R2 is 0x00000000, R3 is 0x00000068, R4 is 0x0000006A, R5 is 0x0000006A, R6 is 0x00000000, R7 is 0x00000000, R8 is 0x00000000, R9 is 0x00000000, R10 is 0x00000000, R11 is 0x00000000, R12 is 0x00000000, R13 (SP) is 0x20000020, R14 (LR) is 0x00000068, R15 (PC) is 0x00000054, and xPSR is 0x61000000.
- Disassembly:** The main function is shown with assembly instructions. The current instruction is `0x00000542 B50E PUSH {r1-r3, ...}`. The code includes a loop that waits for a character 'A' and then sends 'B' via UART.
- GPIOF:** A table of GPIOF register properties and values.
- UART #1:** A window showing the UART output, which currently displays "Enter a character: D", "Enter a character: B", and "Enter a character:".

Property	Value
DATA	0x00000015
DIR	0x0000000E
IS	0x00000000
IBE	0x00000000
IEV	0x00000000
IM	0
RIS	0
MIS	0
ICR	0
AFSEL	0x00000000
DR2R	0x000000FF
DR4R	0x00000000
DR8R	0x00000000
ODR	0x00000000
PUR	0x00000000
PDR	0x00000000

Property	Value
DR	0x00000020
RSR	0
ECR	0
FR	0x00000090
ILPR	0
IBRD	0x00000068
FBRD	0x00000008
LCRH	0x00000070
CTL	0
IFLS	0
IM	0
RIS	0
MIS	0
ICR	0
DMACR	0
_9BITADDR	0

The screenshot shows the TI-RTOS IDE with the following components:

- Registers:** A list of registers with their current values. R0 is 0x00000054, R1 is 0x20000020, R2 is 0x00000000, R3 is 0x00000068, R4 is 0x0000006A, R5 is 0x0000006A, R6 is 0x00000000, R7 is 0x00000000, R8 is 0x00000000, R9 is 0x00000000, R10 is 0x00000000, R11 is 0x00000000, R12 is 0x00000000, R13 (SP) is 0x20000020, R14 (LR) is 0x00000068, R15 (PC) is 0x00000054, and xPSR is 0x61000000.
- Disassembly:** The main function is shown with assembly instructions. The current instruction is `0x00000542 B50E PUSH {r1-r3, ...}`. The code includes a loop that waits for a character 'A' and then sends 'B' via UART.
- UART0:** A table of UART0 register properties and values.
- UART #1:** A window showing the UART output, which currently displays "Enter a character: D", "Enter a character: B", and "Enter a character:".

Property	Value
DR	0x00000020
RSR	0
ECR	0
FR	0x00000090
ILPR	0
IBRD	0x00000068
FBRD	0x00000008
LCRH	0x00000070
CTL	0
IFLS	0
IM	0
RIS	0
MIS	0
ICR	0
DMACR	0
_9BITADDR	0

Sending "D"

The screenshot displays the IDE interface for a Texas Instruments LaunchPad. The main window shows the disassembly of the program, with the following code visible:

```
81: int main () {  
82:     const int len = 9; FUSH (r1-r3,  
83:     NOP  
84:     void RGB_CLR (unsigned char data) {  
85:         GPIO_PORTF_DATA_R &= ~data;  
86:     }  
87:     int main () {  
88:         const int len = 9;  
89:         char command [len] = {0};  
90:         RGB_Init ();  
91:         UART_Init ();  
92:         RGB_CLR (0x0E);  
93:         while (1) {  
94:             UART_OutString ("Enter a character: ");  
95:             getCommand (command, len);  
96:             if (strcmp (command, "A") == 0) {  
97:                 RGB_CLR (0x0E);  
98:                 SysTick_wait (60);  
99:                 RGB_OutPut (0x02);  
100:             }  
101:         }  
102:     }  
103: }
```

The GPIOF register is shown with the following properties:

Property	Value
DATA	0x00000019
DIR	0x0000000E
IS	0x00000000
IBE	0x00000000
IEV	0x00000000
IM	0
RIS	0
MIS	0
ICR	0
AFSEL	0x00000000
DR2R	0x000000FF
DR4R	0x00000000
DR8R	0x00000000
ODR	0x00000000
PUR	0x00000000
PDR	0x00000000

The Command window shows the following input:

```
*** Currently used: 1696 Bytes (5%)  
Enter a character: A  
Enter a character: D  
Enter a character:
```

The screenshot displays the IDE interface for a Texas Instruments LaunchPad, showing the disassembly of the program. The main window shows the disassembly of the program, with the following code visible:

```
81: int main () {  
82:     const int len = 9; FUSH (r1-r3,  
83:     NOP  
84:     void RGB_CLR (unsigned char data) {  
85:         GPIO_PORTF_DATA_R &= ~data;  
86:     }  
87:     int main () {  
88:         const int len = 9;  
89:         char command [len] = {0};  
90:         RGB_Init ();  
91:         UART_Init ();  
92:         RGB_CLR (0x0E);  
93:         while (1) {  
94:             UART_OutString ("Enter a character: ");  
95:             getCommand (command, len);  
96:             if (strcmp (command, "A") == 0) {  
97:                 RGB_CLR (0x0E);  
98:                 SysTick_wait (60);  
99:                 RGB_OutPut (0x02);  
100:             }  
101:         }  
102:     }  
103: }
```

The UART0 register is shown with the following properties:

Property	Value
DR	0x00000020
RSR	0
ECR	0
FR	0x00000090
ILPR	0
IBRD	0x00000068
FBRD	0x00000008
LCRH	0x00000070
CTL	0
IFLS	0
IM	0
RIS	0
MIS	0
ICR	0
DMACTL	0
_9BITADDR	0

The Command window shows the following input:

```
*** Currently used: 1696 Bytes (5%)  
Enter a character: A  
Enter a character: D  
Enter a character:
```

Video showing the LEDs with timer changing:

[Click here](#)