Temple University

College of Engineering

Department of Electrical and Computer Engineering (ECE)

Student Lab Report Cover Page

Course Number : 3613

Course Section : 002

Experiment # : Lab # 5

Student Name (print) : Robert Bara

TUid# : 915614617

Date : 9/30/2020

Grade : ______ **/100**

TA Name : Sung Choi

ACTIVITIES:

Activity 1 .

Write the assembly code using the Stack Pointer (SP) to store the given numbers in the assigned memory locations and retrieve the numbers from the memory locations. The result of arithmetic operations using the retrieved value must output through the I/O ports (total 50 points).

- **CODE with full comments:** Specify each part in the code

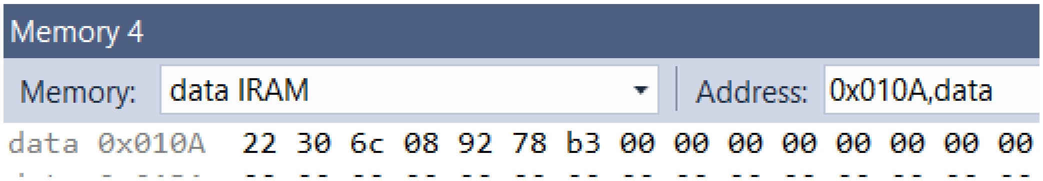
```
//Robert Bara Lab 5
//Activity 1
//Part I Stack Pointer
LDI R16,HIGH(0x0110)          ;Set high byte of address for SP
OUT SPH,R16
LDI R16,LOW(0X0110)                   ;Set low byte of address for SP
OUT SPL,R16
//initalize the ports as output
LDI R29, $00
LDI R30, $FF
OUT DDRA, R30
OUT DDRB, R30
OUT DDRC, R30
OUT DDRD, R30
nop

//Loading the values to be pushed into the stack
LDI R16,$B3
LDI R17,$78
LDI R18, $92
LDI R19, $08
LDI R20, $6C
LDI R21, $30
LDI R22, $22
nop

//Pushing values into the STACK
PUSH R16
PUSH R17
PUSH R18
PUSH R19
PUSH R20
PUSH R21
PUSH R22
nop

//Part II Popping values off of the STACK to be stored in GPRs and performing Arithmetic Operations
//Starting from the top of the stack
POP R22
POP R21
```

```
ADD R21, R22 ;Operation $30+$22, stored in R21
IN R0, SREG

POP R20
POP R19
SUB     R19,R20 ;Operation $08-$6C, stored in R19
IN R1, SREG

POP R18
POP R17
POP R16
ADD R16, R17 ;Operation $B3+$78, and stored in R16
IN R2, SREG

NOP
//PART III CHECKING IF CARRY BIT OF STATUS REGISTER IS 1, DETERMINING IF PORTD WILL BE EQUAL TO 1 OR 0
OUT PORTA, R21
SBRC R0,0
OUT PORTD, R30
SBRS R0, 0
OUT PORTD, R29
NOP

OUT PORTB, R19
SBRC R1,0
OUT PORTD, R30
SBRS R1, 0
OUT PORTD, R29
NOP

OUT PORTC, R16
SBRC R2,0
OUT PORTD, R30
SBRS R2, 0
OUT PORTD, R29
NOP
//UPDATING R16 TO FINISH PART II'S OPERATION
SUB R16,R18 ;Operation $B3+$78-92, and stored in R16
IN R3, SREG
NOP
OUT PORTC, R16
SBRC R3,0
OUT PORTD, R30
SBRS R3, 0
OUT PORTD, R29
NOP
```

- **RESULT:** Show the screenshots for the required outputs from each part.

Part I. Store the given numbers in the memory locations using the Stack Pointer (SP) and PUSH instruction. (10 points) Screenshot with the description.

Screenshot – Contents of the memory locations from 0x010A to 0x0110 after PUSH



Memory: data IRAM

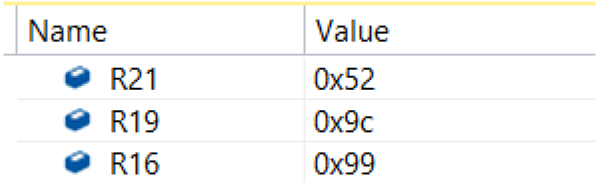
Address: 0x010A,data

```
data 0x010A  22 30 6c 08 92 78 b3 00 00 00 00 00 00 00 00
```

[Figure 1. Contents of Memory Locations using Stack Pointer and Push instructions from 0x010A to 0x0110]

Part II. Load the numbers in the memory locations using the Stack Pointer (SP) and POP instruction to GPRs and use them to do arithmetic operations. (20 points)

Screenshot – Results of the Arithmetic Operations shown in GPRs



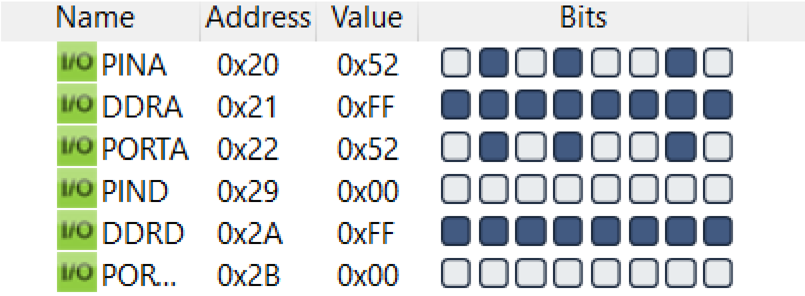
Name	Value
R21	0x52
R19	0x9c
R16	0x99

[Figure 2. GPR results after popping back into GPR and performing arithmetic operations]

Part III. Send the arithmetic operations results to the designated I/O ports and generate the required output according to the carry flag value to the specified I/O port. (20 points)

Screenshot – Results of the Arithmetic Operations shown in PORTA and PORTD

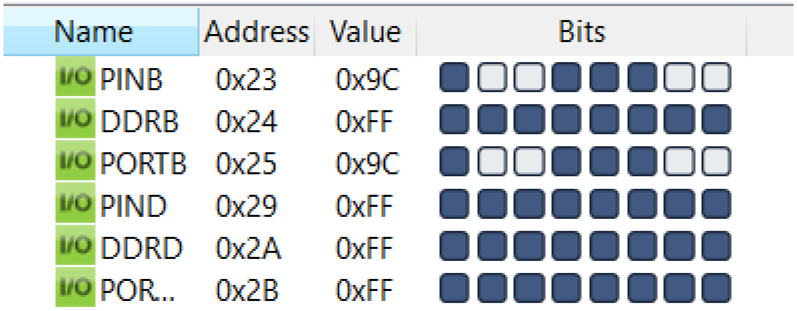
(1)



Name	Address	Value	Bits
PINA	0x20	0x52	□■□■□□■□
DDRA	0x21	0xFF	■■■■■■■■
PORTA	0x22	0x52	□■□■□□■□
PIND	0x29	0x00	□□□□□□□□
DDRD	0x2A	0xFF	■■■■■■■■
POR..	0x2B	0x00	□□□□□□□□

[Figure 3A. Result of arithmetic operations, PORTA, PORTD]

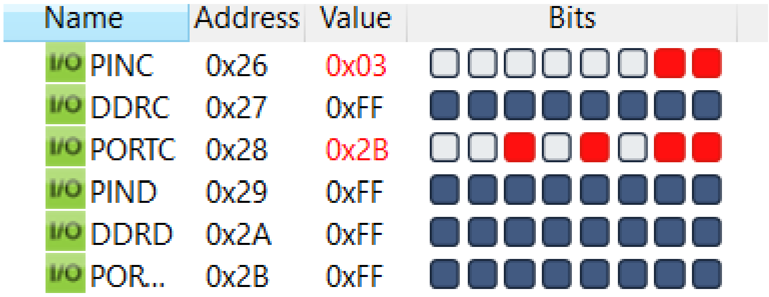
(2)



Name	Address	Value	Bits
PINB	0x23	0x9C	■□□■■■□□
DDRB	0x24	0xFF	■■■■■■■■
PORTB	0x25	0x9C	■□□■■■□□
PIND	0x29	0xFF	■■■■■■■■
DDRD	0x2A	0xFF	■■■■■■■■
POR...	0x2B	0xFF	■■■■■■■■

[Figure 3B. Result of arithmetic operations, PORTB, PORTD]

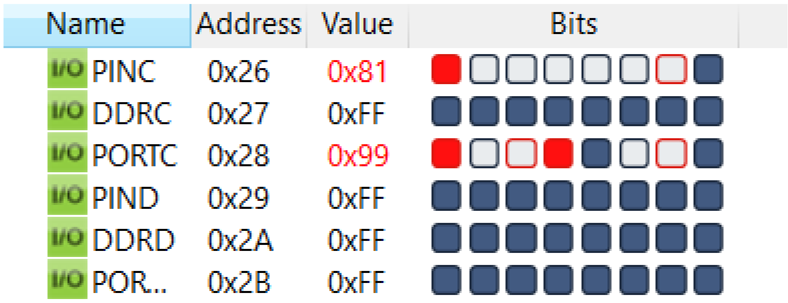
(3)



Name	Address	Value	Bits
PINC	0x26	0x03	□□□□□□🟥🟥
DDRC	0x27	0xFF	■■■■■■■■
PORTC	0x28	0x2B	□□🟥□🟥□🟥🟥
PIND	0x29	0xFF	■■■■■■■■
DDRD	0x2A	0xFF	■■■■■■■■
POR...	0x2B	0xFF	■■■■■■■■

[Figure 3C. Result of arithmetic operations, PORTC, PORTD]

(4)



Name	Address	Value	Bits
PINC	0x26	0x81	🟥□□□□□□■
DDRC	0x27	0xFF	■■■■■■■■
PORTC	0x28	0x99	🟥□□🟥■□□■
PIND	0x29	0xFF	■■■■■■■■
DDRD	0x2A	0xFF	■■■■■■■■
POR...	0x2B	0xFF	■■■■■■■■

[Figure 3D. Result of arithmetic operations, PORTC, PORTD]

Activity 2.

Write assembly codes to use subroutine and call to make output through the I/O ports and the memory locations using directives (total 50 pts).

- **CODE with full comments:** Specify each part in the code

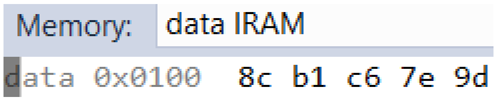
```
//Activity 2 Robert Bara
//PART 1: ASSIGNING DIRECTIVES TO MEMORY LOCATIONS FOR STORING
ACTIVITY_2:
.EQU NUM1=0X0100
.EQU NUM2=0X0101
.EQU NUM3=0X0102
.EQU NUM4=0X0103
.EQU NUM5=0X0104
//LOADING NUMBERS AND PERFOMING ARITHMETIC OPERATIONS
LDI R16,$8C ;num1
LDI R17,$9A
LDI R18,$17
ADD R17,R18 ;num2
LDI R19,0B11000110
LDI R20, 126
LDI R21, $F2
LDI R22, $55
SUB R21,R22
//STORING THE NUMBERS/RESULTS IN THE DESIGNATED MEMORY
STS NUM1,R16
STS NUM2, R17
STS NUM3, R19
STS NUM4, R20
STS NUM5,R21
NOP
//PART II
LDI R30, $FF
OUT DDRA, R30
LOOP:
OUT PORTA,R16 //For some reason I was having trouble with Atmel where it kept glitching so for num 4 and num 5, I just updated the registers on this line to get the screen shots
CALL TIME_DELAY
OUT PORTA,R17
CALL TIME_DELAY
OUT PORTA, R19
CALL TIME_DELAY
OUT PORTA, R20
CALL TIME_DELAY
OUT PORTA, R21
RJMP LOOP
RJMP ACTIVITY_2
NOP

//0.05 second => number of loops: (32*20*250)
```

```
TIME_DELAY:  LDI R20,32     //outer loop control value
          L1: LDI R21, 20          //mid loop control value
          L2: LDI R22, 25                    //inner loop control value
          L3:       NOP
                    NOP
                              DEC R22
                              BRNE L3
                    DEC R21
                    BRNE L2
          DEC R20
          BRNE L1
RET                 //return to MAIN
```

- **RESULT:** Show the screenshots for the required outputs from each part.

Part I. Store the given numbers and the results of arithmetic operation to the assigned memory locations using directives. **(20 points)**

Screenshot – Show the contents of the memory address, from 0x0100 to 0x0104



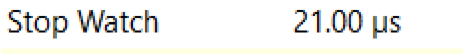
```
Memory:  data IRAM
data 0x0100   8c b1 c6 7e 9d
```

[Figure 4: Stored contents into memory using directives]

Part II. Show the numbers output to the assigned I/O ports with a time delay. Make the binary output of each number/result through PORTA with a 0.05-second time delay**.** Create the **subroutine** with the label, '**time_delay**,' to make 0.05 second. Call the subroutine for each PORTA output **(30 points)**

****Result 1. Screenshot – stopwatch view for the time_delay subroutine**

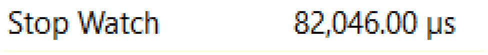
1) Show the stopwatch in the Studio 7 before calling the <time_delay> subroutine



Stop Watch   21.00 µs

[Figure 5: Stopwatch before calling TIME_DELAY]

2) Show the stopwatch in the Studio 7 after return from the <time_delay> subroutine

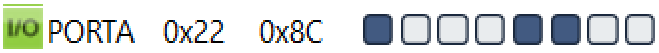


Stop Watch   82,046.00 µs

[Figure 5A: Stopwatch after TIME_DELAY returns to main]

****Result 2. Screenshot – PORTA outputs**

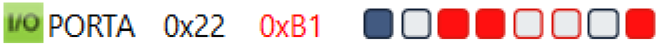
1) PORTA for Num1



I/O PORTA 0x22 0x8C ■□□□■■□□

[Figure 6A: Num1 in binary stored in PortA]

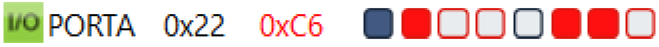
2) PORTA for Num2



I/O PORTA 0x22 0xB1 ■□■■□□□■

[Figure 6B: Num2 in binary stored in PortA]

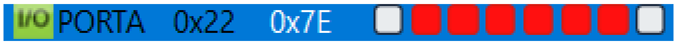
3) PORTA for Num3



I/O PORTA 0x22 0xC6 ■■□□□■■□

[Figure 6C: Num3 in binary stored in PortA]

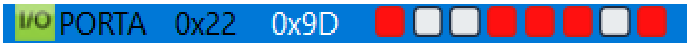
4) PORTA for Num4



I/O PORTA 0x22 0x7E □■■■■■■□

[Figure 6D: Num4 in binary stored in PortA]

5) PORTA for Num5



I/O PORTA 0x22 0x9D ■□□■■■□■

[Figure 6E: Num5 in binary stored in PortA]

ECE3613 Processor System Laboratory Rubric
Lab #: 4
Section: 001 / 002
Name: _________________________

Activity	**Part**	**Task**	**Full Points**	**Earned Points**	**Comment**
1	I	PUSH	10		
	II	POP & ALU	20		
	III	I/O	20		
Subtotal			50		
2	I	Directives & Memory Storing	20		
	II	I/O, Condition and Time Delay	30		
Subtotal			50		
Total			100		