LAB 3-C

COUNTER

OBJECTIVES:

- To examine the I/O port operation using a simulator.
- To trace through a CALL subroutine using a simulator.

MATERIAL:

- Atmel Studio
- https://lcgamboa.github.io/js/picsimlab.html?../picsimlab_examples/ (Simulator)

WEB SITES:

- www.microchip.com for Atmel Studio Software

ACTIVITY 1

Test the operation of an "up-counter" from $00 - $FF on picsimlab. Connect the LEDs to each pin of the PORTD and perform the following steps:

a) Assemble the following code:

```
;========Press your up-counter code here==========



;================================================
DELAY:  LDI     R21, 32
DL1:    LDI     R22, 200
DL2:    LDI     R23, 250
DL3:    NOP
        NOP
        DEC     R23
        BRNE    DL3
        DEC     R22
        BRNE    DL2
        DEC     R21
        BRNE    DL1
        RET
```

b) Upload the hex file to the picsimlab.

LAB 3-C

COUNTER

c) Observe the LEDs counting up from $00 to $FF.

The LEDs attached to PORTD will visually represent an 8-bit binary counter, starting at 0x00 and increasing sequentially until they reach 0xFF. Once the maximum value is hit, the counter automatically resets to zero and repeats the cycle.

To adjust how fast the LEDs count, you must change the value in register R21 inside the delay subroutine. Since R21 is currently set to 32, increasing this number will slow down the count, while decreasing it will make the binary increment appear faster.

d) Change the time delay in between the counts by changing the value of R21 register (or increase/decrease the number of NOPs) but make sure the time delay is long enough that you can observe the LEDs counting up. Create a hex file then upload to picsimlab to see what differences.

Code:

```
.INCLUDE "m328pdef.inc"

.ORG 0x0000

 RJMP MAIN


 ; 1. Initialize Stack Pointer (Essential for CALL/RET)

 LDI R16, LOW(RAMEND)

 OUT SPL, R16

 LDI R16, HIGH(RAMEND)

 OUT SPH, R16


 LDI R16, 0xFF

 OUT DDRD, R16


 LDI R16, 0x00


LOOP:
```

LAB 3-C

COUNTER

```
 OUT PORTD, R16 ; Display counter on LEDs
 RCALL DELAY ; Call delay
 INC R16 ; Increment counter
 RJMP LOOP ; Repeat (R16 naturally wraps from $FF to $00)
;===============================================
DELAY:
   LDI R21, 32
DL1:
   LDI R22, 200
DL2:
   LDI R23, 250
DL3:
   NOP
   NOP
   NOP    ; Added extra NOP
   NOP    ; Added extra NOP
   DEC R23
   BRNE DL3
   DEC R22
   BRNE DL2
   DEC R21
   BRNE DL1
   RET
```

ACTIVITY 2

In Activity 1, the maximum count was $FF (or 255). Modify the above program to set maximum count to 10.

LAB 3-C

COUNTER

e) Upload the hex file into the AVR simulator.

f) Observe the LEDs counting up from $00 to $09 (00001001 binary) continuously.

g) Change the maximum count to the value of your age and observe the LED counting up to that number.

```
.INCLUDE "m328pdef.inc"

.ORG 0x0000

 RJMP MAIN


MAIN:

 ; Initialize Stack Pointer

 LDI R16, LOW(RAMEND)

 OUT SPL, R16

 LDI R16, HIGH(RAMEND)

 OUT SPH, R16


 ; Set PORTD as Output

 LDI R16, 0xFF

 OUT DDRD, R16


 ; Initialize Counter

 LDI R16, 0x00


LOOP:

 OUT PORTD, R16 ; Display current value

 RCALL DELAY
```

LAB 3-C

COUNTER

```
INC R16        ; Increment value

; --- MODIFIED SECTION ---
CPI R16, 20    ; Compare R16 with 20 (instead of 10)
BRNE LOOP      ; If R16 != 20, continue looping
; ------------------------

; If R16 == 20, reset to 0
LDI R16, 0
RJMP LOOP

; DELAY subroutine remains the same
DELAY:
 LDI R21, 32
DL1:
 LDI R22, 200
DL2:
 LDI R23, 250
DL3:
 NOP
 NOP
 DEC R23
 BRNE DL3
 DEC R22
 BRNE DL2
 DEC R21
```

LAB 3-C

COUNTER

BRNE DL1

RET

ACTIVITY 3

For this activity, connect the DIP switches to pins of port B. Now, use the DIP switches on picsimlab to set the maximum count for an up-counter instead of using constant as Activity 1 and 2.

a) Upload the hex file into the AVR simulator.

b) Observe the LEDs counting up from 00 to the value set by the 8 switches continuously.

Pattern is binary counting from 00 up to the switch value. When LEDs match the switch pattern, it reset to 00. It is a continuous loop so it won't stop.

Answer question

1) What is the maximum count for register R20?

   255 or $FF

2) In this Lab, which port was used to display the count? Which one was used to set the maximum count? Can we use one port for both (inputting the maximum count and displaying the count)?

   PORTD served as the output to visualize the count on LEDs, while PORTB functioned as the input to define the count limit through DIP switches. Under this specific setup, a single port cannot handle both input and output duties at once. This is because each port's pins must be explicitly designated as either an input (by writing a 0) or an output (by writing a 1) within its corresponding Data Direction Register (DDRx) at any given moment.

3) In this Lab, we used BRNE (Branch Not Equal) instruction. Explain how it works.

   The BRNE (Branch if Not Equal) instruction operates by monitoring the Zero Flag (Z) within the Status Register (SREG). If a previous calculation or comparison results in a non-zero value, the Z flag remains 0, triggering the program to jump to a designated label. Conversely, if the result is exactly zero, the Z flag is set to 1, causing the CPU to bypass the jump and continue with the subsequent line of code.