Roee Landesman
Amir Hashemizad

SW#8: Using the Interrupts in Assembly Language Programming

Behavior Description

One:

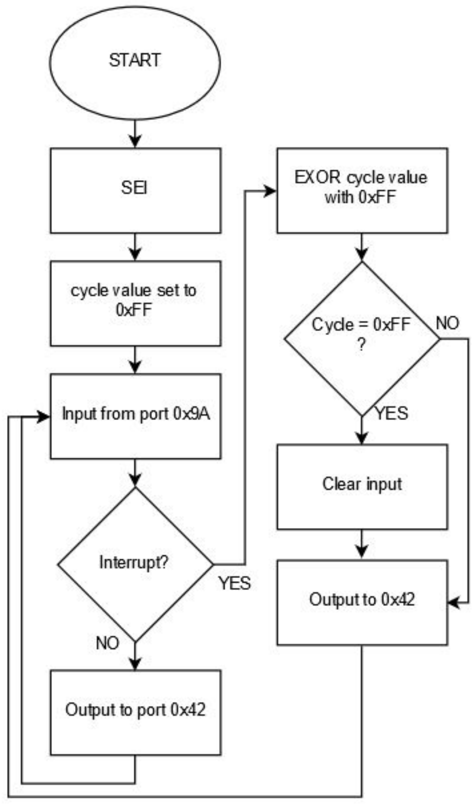
First a register is loaded with the hex value 0xFF used to check how many cycles have been run through. The main loop inputs from the switches and outputs before looping back to main. If an interrupt is pressed, 0xFF is exor'ed with 0xFF to flip it to zero. This value is compared with the original value to determine which of the two possible states it is, zero or FF. if it is zero, it outputs to the leds, if not it clears the value before outputting to the leds. The the program loops back to main.

Two:

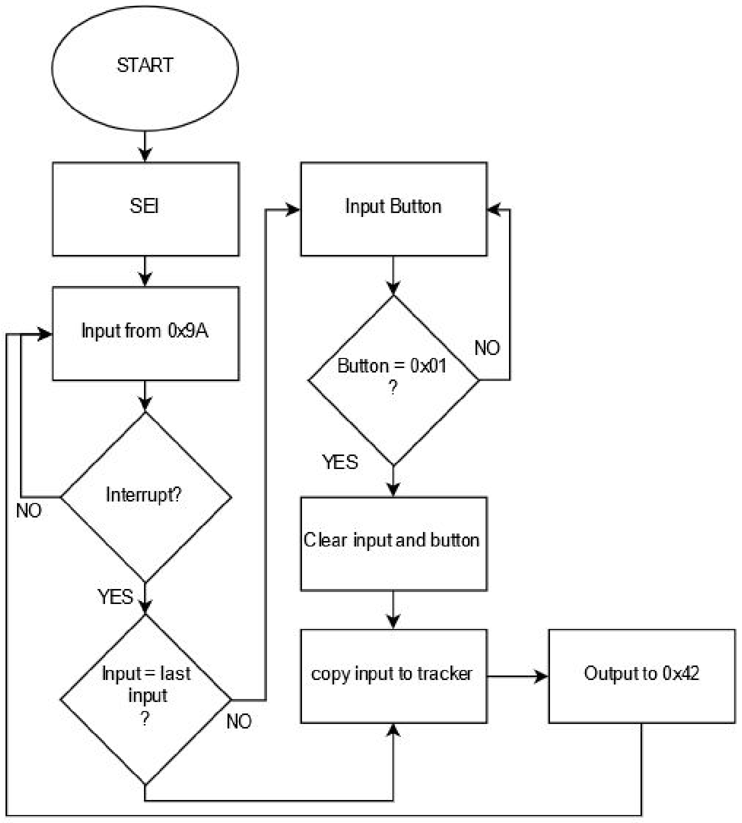
First a register is initialized at zero to serve as a baseline to be compared to later. Then in the main loop only an input and a branch back to main are utilized. In the interrupt triggered process, the input is compared to the first initialized register. If they are not equal, the input is copied to the first register to be compared later and then outputted. If they are equal, then a button value is input in an endless loop which s only escaped to the output process if the button value is 0x01.

Flow Chart

1)



2)



Verification by Simulation Results

1)

	1	2	3	4
Switch	0x02	0x11	0x11	0xFF
Interrupt	NO	YES	YES	YES
Output	0x02	0x11	0x00	0xFF

2)

	1	2	3	4
Switches	0x10	0x10	0x11	0x11
Interrupt	NO	YES	YES	YES
Button	N/A	N/A	0x00	0x01
Stuck in Loop	N/A	N/A	YES	NO
Output	0x00	0x11	N/A	0x00

Assembly Source Code

1)

```
;- Programmers: Amir Hashemizad and Roee Landesman
;- Date: 02-23-18
;-
;- This program blinks LEDS depending on the most recent switches
;- pressed, outputting on port 0x42, and turns them on or off
;- depending on  if the interrupts
;--------------------------------------------------------------------
;--------------------------------------------------------------------
;- Constants
;--------------------------------------------------------------------
.EQU SWITCHES = 0x9A
.EQU LEDS = 0x42

.CSEG
.ORG 0x10                       ; data starts here

          SEI                   ; init interrupts
          MOV R0, 0xFF          ; set cycle value to 0xFF
MAIN:     IN R10, SWITCHES           ; input from port 0x9A
          OUT R10, LEDS         ; output to port 0x42
          BRN MAIN              ; loop to main
                                ; interrupt service routine below
ISR:      EXOR R0, 0xFF         ; exor input with 0xFF
          CMP R0, 0xFF          ; compare cycle value to 0xFF
          BRNE OUTP             ; if cycle value is zero, output
          MOV R10, 0x00         ; clear input if otherwise
OUTP:     OUT R10, LEDS         ; output input
          RETIE                 ; return


.ORG 0x3FF                      ; vector starts here
VECTOR:    BRN ISR              ; branch to the ISR
```

2)

```
;- Programmers: Amir Hashemizad and Roee Landesman
;- Date: 02-23-18
;-
;- This program outputs the most recent values on the switches to
leds
;- on port 0x42 when there are interrupts. If the same value is
loaded
```

```
;- twice the program stops outputting until a button is pressed.
;---------------------------------------------------------------------
;---------------------------------------------------------------------
;- Constants
;---------------------------------------------------------------------
.EQU SWITCHES = 0x9A
.EQU BUTTON = 0x9B
.EQU LEDS = 0x42

.CSEG
.ORG 0x10                        ; data starts here

          SEI                    ; init interrupts
          MOV R11, 0x00          ; init instance tracker
MAIN:     IN R10, SWITCHES            ; input to R10
          BRN MAIN               ; loop to main
                                 ; interrupt service routine below
ISR:      CMP R10, R11           ; compare input with instance tracker
          BRNE OUTP              ; if tracker is the same as input,
                                 ; output
BN:       IN R12, BUTTON         ; input from button
          CMP R12, 0x01          ; compare button with 0x01
          BRNE BN                ; input button again if not equal
          MOV R10, 0x00          ; clear switch input
          MOV R12, 0x00          ; clear button input

OUTP:      MOV R11, R10          ; copy input to tracker
          OUT R10, LEDS          ; output input
          RETIE                  ; return


.ORG 0x3FF                       ; vector starts here
VECTOR:    BRN ISR               ; branch to the ISR
```