CPE 233 Software Assignment 7

Interupts in Assembly

Report by:

Ethan Vosburg (evosburg@calpoly.edu)
Wyatt Tack (wtack@calpoly.edu)

March 6, 2024

Table of Contents

1 Flow Charts

1.1 Interrupt Flowchart

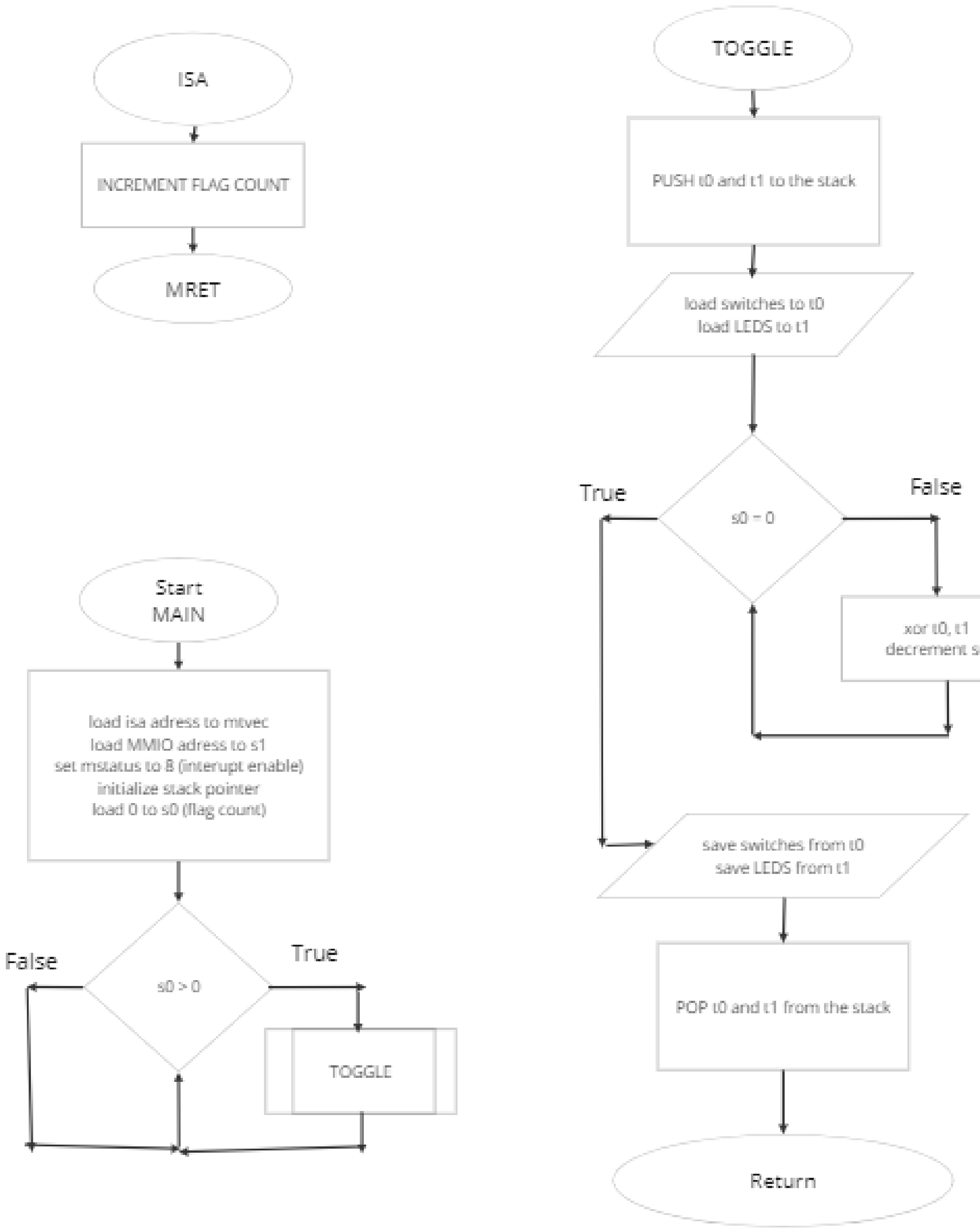

Figure 1: Interrupt Flowchart

1.2 Switch Store with Interrupt Flowchart

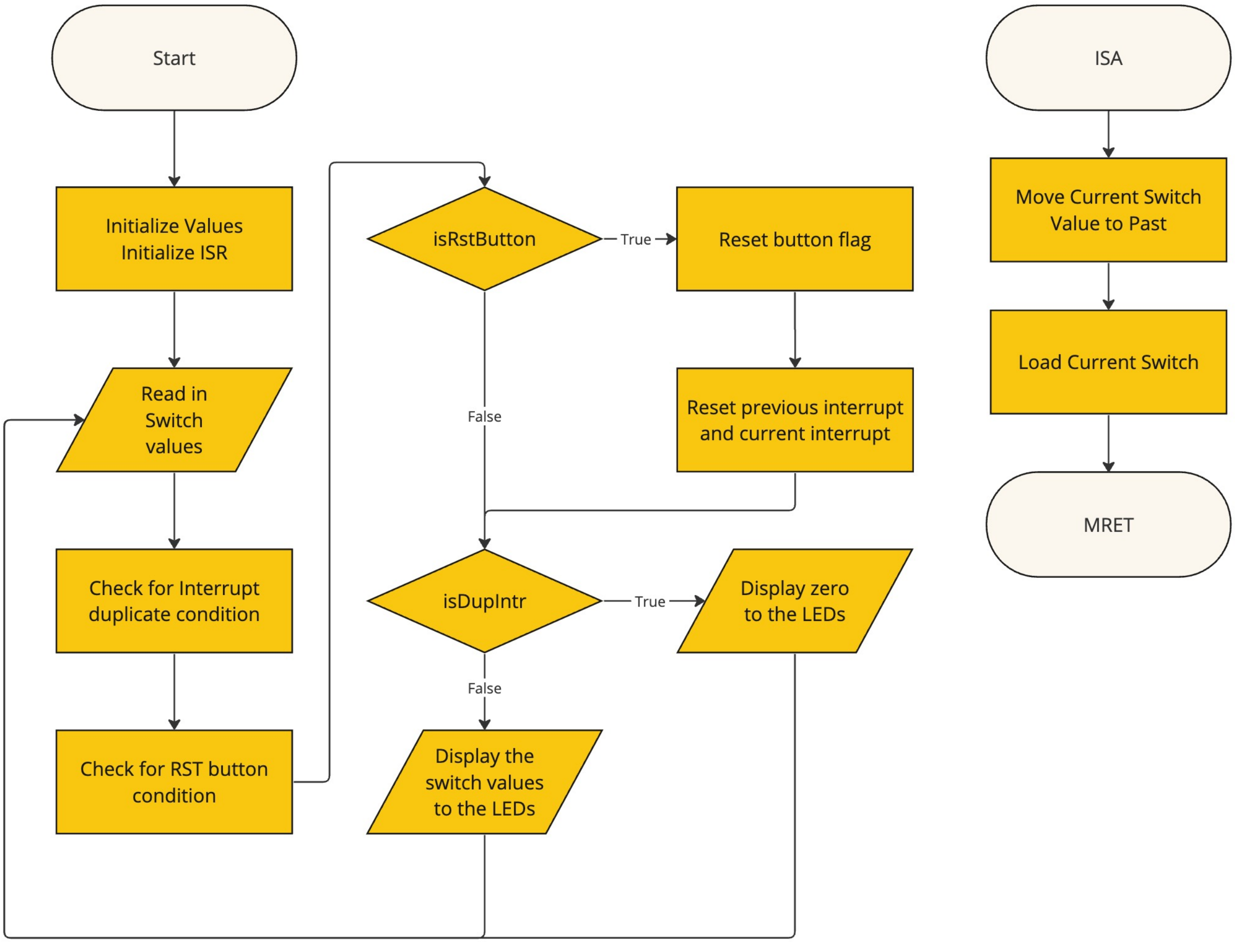

Figure 2: Switch Store with Interrupt Flowchart

2 Assembly Instructions

2.1 Interrupt

```
SETUP:
    la    t0, ISA
    csrrw x0, mtvec, t0   #set ISA prog add
    li    s1, 0x11000000  #set mmio
    li    t0, 8
    csrrw x0, mstatus, t0 #enable interrupts
    li    s0, 0           #set up flag count
    li    sp, 0x10000     #initialize the stacker
LOOP:
    beqz  s0, NOFLAG      #check for flag count
    call  TOGGLE
NOFLAG:
    j     LOOP
TOGGLE:
    addi  sp, sp, -8      #push t0, t1 to stack
    sw    t0, (sp)
    sw    t1, 4(sp)
    lw    t0, (s1)        #load switches
    lw    t1, 0x20(s1)    #load leds
FLAGLOOP:
    xor   t0, t0, t1      #toggle leds
    addi  s0, s0, -1      #decrement flag count
    bnez  s0, FLAGLOOP    #loop for all flag count

    sw    t0, 0x20(s1)    #store toggled leds
    lw    t0, (sp)        #pop t0, t1 from stack
    lw    t1, 4(sp)
    addi  sp, sp, 8
    ret
ISA:
    addi  s0, s0, 1       #increment flag count
    mret
```

Listing 1: Interrupt Implementation Code for Figure 1

2.2 Switch Store with Interrupt

```
# Author: Ethan Vosburg
# Date: 3/4/2024
# Description: This program reads switches to the LEDs and when an interrupt
# is triggered, it will check the value of the past two interrupts and change the
# Status of the LEDs if needed.

    lui     s0, 0x11000         # Load the address of the switches into s0
    li      s1, 0x0             # Intr curent value reg
    li      s2, 0x1             # Intr prev value reg
    la      t0, ISR
    csrrw   x0, mtvec, t0       # Load ISR address
    li      t0, 8
    csrrw   x0, mstatus, t0     # Allow interrupts

MAIN:
    lw      t0, 0(s0)           # Read switches

    # Check if the past two interrupts are the same
    xor     t1, s1, s2          # Check if past == pres
    seqz    t5, t1              # Flag if prev == past

    # Check if the reset button is pressed
    lw      t6, 0x200(s0)           # Get Button value rst if 1

    # Reset display if triggered
    beqz    t6, NOTRESETDISPLAY # Reset if button pushed
    li      t6, 0x0
    sw      zero, 0x200(s0)     # Reset Button
    li      t1, 8
    csrrw   zero, mstatus, t1   # Allow Interupts
    li      s1, 0x0             # Intr curent value reg
    li      s2, 0x1             # Intr prev value reg

NOTRESETDISPLAY:
    beqz    t5, DISPLAY         # Blank display if same
    li      t1, 0
    csrrw   zero, mstatus, t1   # Prevent interrupts
    sw      zero, 20(s0)        # Blankc LEDs
    j       MAIN


DISPLAY:
    sw      t0, 20(s0)          # Display switch values on LEDs
    j       MAIN


ISR:
    add     s2, s1, zero        # Move current val to past val
    lw      s1, 0(s0)           # Get current val
    mret
```

Listing 2: Switch Store with Interrupt Code for Figure 2

3 RARS Verification

3.1 Interrupts Verification

LEDS (0x20)	SWITCHES	INTERUPT	LEDS	Reasoning
0x0000_0000	0x0000_0000	0	0x0000_0000	Tests if nothing happens
0x0000_0000	0x1010_0101	1	0x1010_0101	Tests standard toggle
0x1010_0101	0x1111_1111	0	0x1010_0101	Tests for no toggle if no interrupt
0x1010_0101	0x1111_1111	1	0x0101_0101	Tests toggle for both on and off
0x0101_0101	0xffff_ffff	1	0xfefe_fefe	Tests toggle for all switches
0xfefe_fefe	0xfefe_fefe	1	0x0000_0000	Tests clear (toggle=leds)
0x0000_0000	0xffff_ffff	1	0xffff_ffff	Tests toggle high for all switches

Table 1: Interrupt Verification Table

3.2 Switch Store with Interrupt Verification

1. Inputting `0x0000aaaa` in for the switch vales and then triggering an interrupt and then triggering an interrupt again successfully displayed 0 on the LEDs. Then changing the value on the switches to `0x00005555` and triggering the interrupt again did not bring the LEDs back on. After using the reset button, the display came back on to the expected value.
2. Inputting `0x0005555` into the switches displayed the value on the screen successfully. Then changing the switches to `0x00001234` updated the display. After triggering an interrupt, nothing changed. Then changing the switch values to `0x00004321` updated the display and then pressing the interrupt again did nothing. This is expected behavior
3. Performed the same steps at 1 but end verified the program went back to regular operation but loading random values `0x00002ef7`, then `0x0000e8bb`, and finally `0x00001111`.
4. Input a bunch of random values showing the the program worked as intended with out the interrupt.