Project #1 Verification

Assembly Programming, LEDs, and Switches  
Cooper Brotherton  
Instructor: Dr. Michael Jo  
ECE230-03

# Requirements

* Project implemented in assembly
* LED2 shall be initially *off* on system start
* While LED2 is *off*, upon press and release of S1, LED2 shall begin blinking at a rate of 1Hz
  + Switch S1 shall be software-debounced on press and release
  + LED2 shall turn *on* within 10ms of the release of S1
  + LED2 shall blink at 50% duty cycle (*on* for 500ms and *off* for 500ms)
* While LED2 is blinking, upon press of S1, LED2 shall turn *off* and stop blinking
  + LED2 shall turn *off* within 1ms of S1 press
  + System shall wait until S1 has been released before returning to a state of detecting a new press and release of S1

# Advanced Requirements

* For all requirements, accuracy shall be within ±10μs
* Initially on system start, the red LED of LED2 shall be the *active* LED whose state is toggled by S1
* While LED2 is blinking, upon press of S2, the *active* LED shall toggle in the following cyclic pattern: red → green → blue → red → etc.
  + The newly *active* LED shall turn *on* within 1ms of the press of S2, but not prior to the former *active* LED turning *off*
  + The *active* LED shall toggle only once per press and release of S2
  + Switch S2 shall be software-debounced on press and release
  + The system shall be in a paused state between the press and release of S2

# Test plan

The following details a plan for testing the specifications

|  |  |  |
| --- | --- | --- |
| Test | Procedure | Pass/Fail Criteria |
| 1 | On start, verify LED2 is off | LED2 is off at t=0 |
| 2 | On start, verify the active LED is red | The red LED activates on the first press and release of S1 |
| 3 | Verify LED2 turns on within spec | LED2 turns on within 0<t<10.01ms after S1 is released |
| 4 | Verify S1 and S2 are debounced on both press and release | S1 and S2 have software debouncing on every press and release |
| 5 | Use an oscilloscope to verify LED2 has a frequency of 1Hz and a 50% duty cycle | LED2 toggles every 499.99<t<500.01ms |
| 6 | Verify LED2 turns off and stops blinking after S1 is pressed within spec | LED2 turns off and stops blinking within 0<t<1.01ms after the press of S1 |
| 7 | Verify the system waits until S1 has been released before waiting for a new S1 press | The system waits until S1 has been released before detecting a new S1 press and release |
| 8 | Verify S2 toggles which LED is active and toggles in the correct cyclic pattern | S2 toggles the active LED and goes red → green → blue → red |
| 9 | Verify the next active LED turns on after S2 is pressed within spec | The former active LED turns off and the next active LED turns on within 0<t<1.01ms after the press of S2 |
| 10 | Verify the active LED toggles only once per S2 press and release | The active LED changes only once when S2 is pressed and released |
| 11 | Verify the system is paused while S2 is being held | The LED does not blink nor react to switch inputs while S2 is being held |

# Verification

## Test 1

On start, LED2 is completely off.

**Meets criteria**.

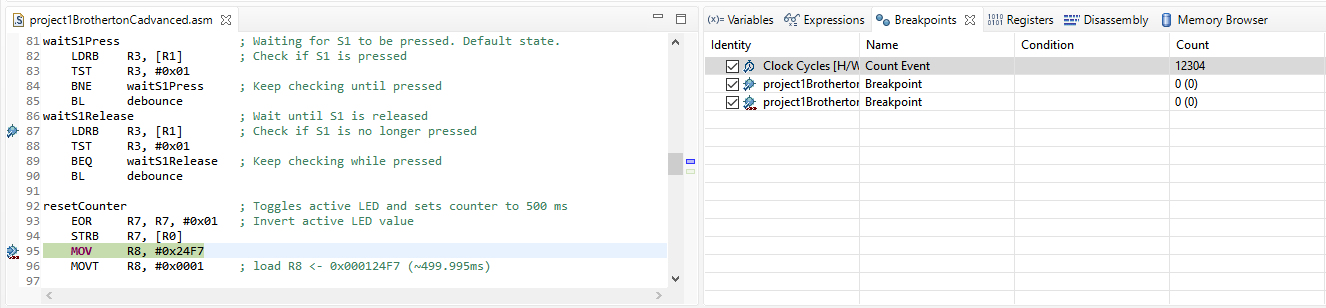
## Test 2

When S1 is pressed and released, the red LED begins blinking.

**Meets criteria.**

## Test 3

The following code analysis depicts the code between S1 being released (breakpoint at line 47) and the LED turning on (breakpoint at line 95). By looking at the clock cycle counter, it can be determined how long it took.



The following calculation shows how long the process took given that the MSP432P401R clock rate is 3MHz:

**Meets criteria**.

## Test4

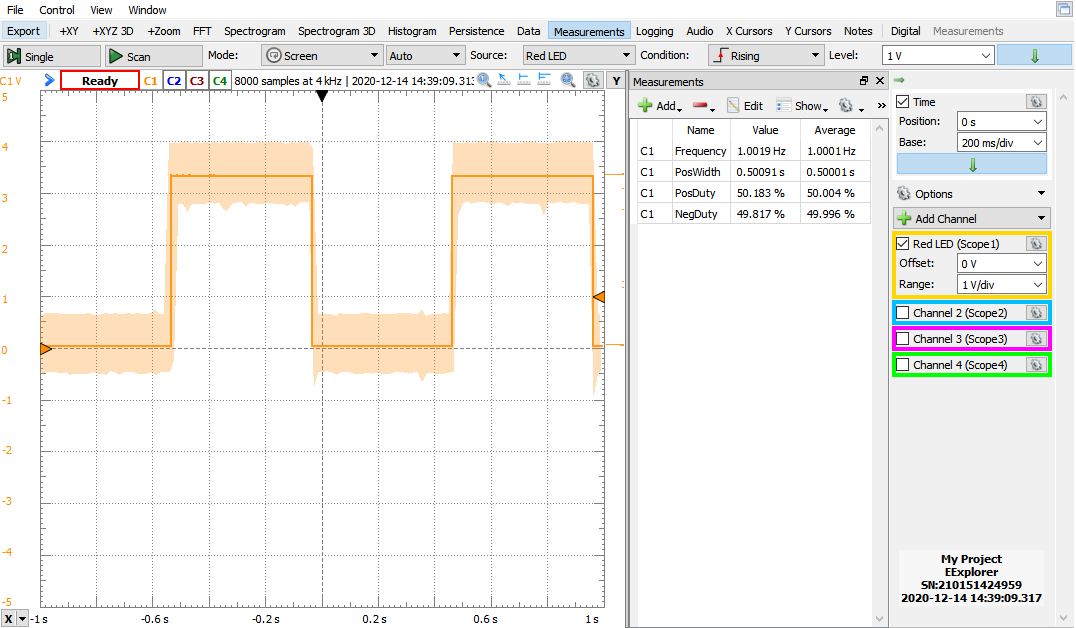
The following code displays the part of the program with the state machine that checks for switches being pushed.

|  |  |
| --- | --- |
|  | After every single switch press or release and before any check for switch input (lines 85, 90, 118, 123, 130, 135), the software branches and links to a procedure which causes a delay and acts as a software debounce. |

**Meets criteria.**

## Test 5

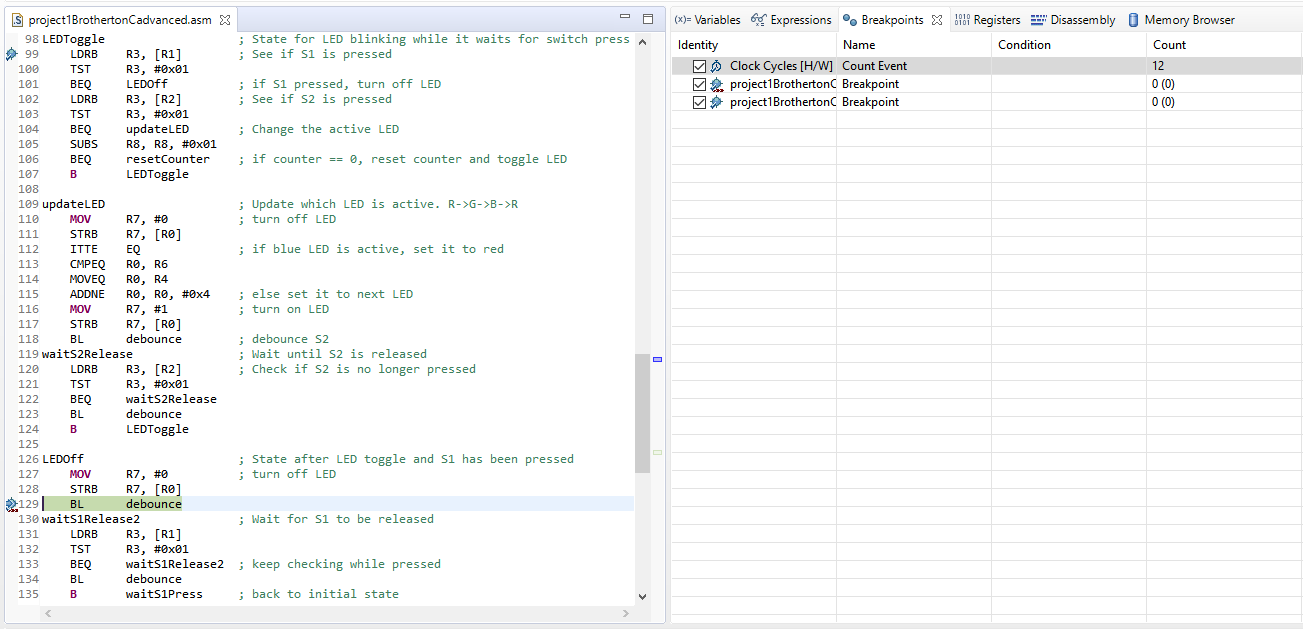
In order to verify the LED follows the blinking pattern, an oscilloscope was used. The following is a snapshot of the oscilloscope and the LED output. Due to slight inconsistencies, how long the LED blinks (PosWidth) and the frequency varies. However, on average the LED blinks for 500.01ms and nearly has a 50% duty cycle.



**Meets criteria**.

## Test 6

The following code analysis depicts the code between S1 being pressed (breakpoint at line 99) and the LED turning off (breakpoint at line 131). By looking at the clock cycle counter, it can be determined how long it took.

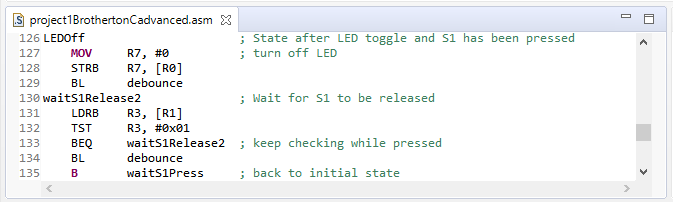


The following calculation shows how long the process took given that the MSP432P401R clock rate is 3MHz:

**Meets criteria**.

## Test 7

The following code shows that the program will wait until S1 is released until it will detect another S1 press.



**Meets criteria.**

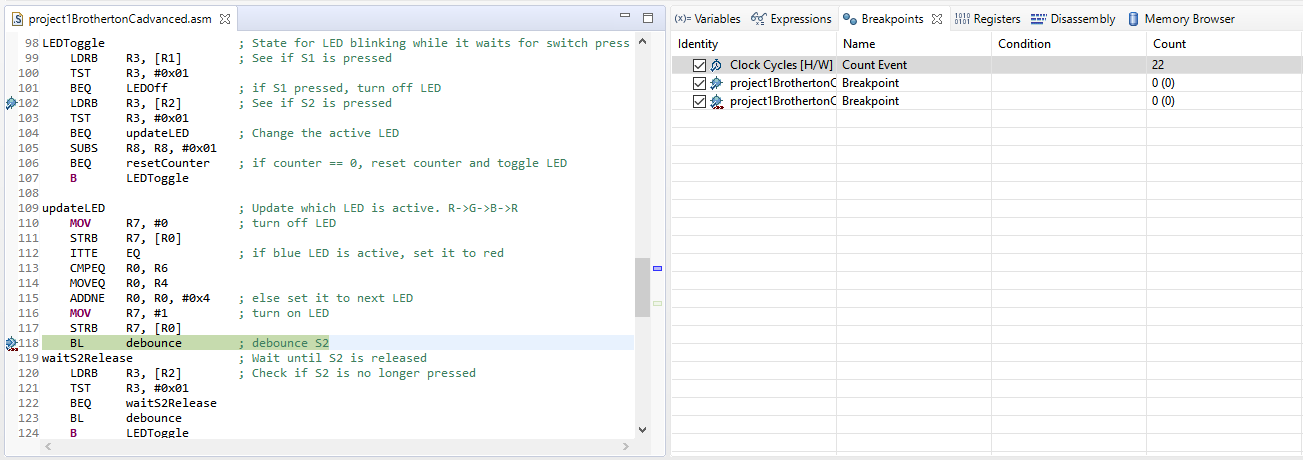
## Test 8

When S2 is pressed while LED2 is in the blinking cycle, the system cycles between the red, green, and blue LED.

**Meets criteria.**

## Test 9

The following code analysis depicts the code between S2 being pressed (breakpoint at line 102) and the former active LED turning off and the new active current LED turning on (breakpoint line 114). By looking at the clock cycle counter, it can be determined how long it took.



The following calculation shows how long the process took given that the MSP432P401R clock rate is 3MHz:

**Meets criteria**.

## TEst 10

The LED only changes once per press of S2. Regardless if S2 is pressed or held.

**Meets criteria.**

## Test 11

While S2 is being held, the active LED remains on and the S1 does not change the system.

**Meets criteria.**

# Conclusion

All the tests met the criteria for both the basic and advanced requirements.

# Demo Link

The following a YouTube link demonstrating the project:

<https://www.youtube.com/watch?v=uPxH2E80UFQ>