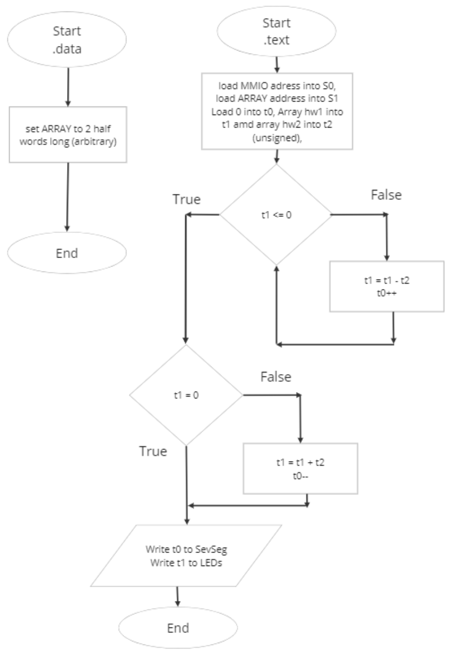
CPE 233 SW 5

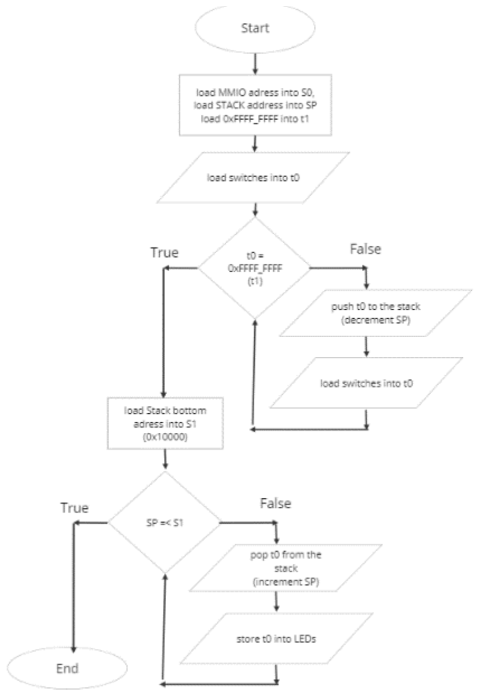
Wyatt Tack and Thomas Hong

1. Flowcharts :

   Part 1 :


Start
.data
set ARRAY to 2 half words long (arbitrary)
End
Start
.text
load MMIO adress into S0, load ARRAY address into S1 Load 0 into t0, Array hw1 into t1 amd array hw2 into t2 (unsigned),
True
t1 <= 0
False
t1 = t1 - t2
t0++
False
t1 = 0
True
t1 = t1 + t2
t0--
Write t0 to SevSeg
Write t1 to LEDs
End

Part 2 :


Start
load MMIO adress into S0, load STACK address into SP load 0xFFFF_FFFF into t1
load switches into t0
True
t0 = 0xFFFF_FFFF (t1)
False
push t0 to the stack (decrement SP)
load switches into t0
load Stack bottom adress into S1 (0x10000)
True
SP =< S1
False
pop t0 from the stack (increment SP)
store t0 into LEDs
End

2. Table 1: Verification Part 1:

HW1	HW2	Quotient	Remainder	Reasoning
12 **0x000c**	**3** **0x0003**	**4** **0x0004**	**0** **0x0000**	Standard Division without remainder
256 **0x100**	**4** **0x0004**	**64** **0x0040**	**0** **0x0000**	‘
65535 **0xffff**	**15** **0x000f**	**4369** **0x1111**	**0** **0x0000**	‘ at highest possible HW1
482 **0x01e2**	**13** **0x000d**	**37** **0x0025**	**1** **0x0001**	Standard Division with remainder
18 **0x0012**	**22** **0x0016**	**0** **0x0000**	**18** **0x0012**	‘ fractional division
1 **0x0001**	**0** **0x0000**	**Program never terminates**		Tests overflow from dividing by 0

Table 2: Verification Part 2:

Switches (First to last)	LEDS (First to last)	Reasoning
0x0000_000a 0x0000_000b 0x0000_000c 0x0000_000d 0x0000_000e 0x0000_000f 0x0000_0001 0x0000_0002 0x0000_0003 0xffff_ffff	0x0000_0003 0x0000_0002 0x0000_0001 0x0000_000f 0x0000_000e 0x0000_000d 0x0000_000c 0x0000_000b 0x0000_000a	Tests standard operation, outputs reverse.
0xffff_ffff	(Nothing)	Tests no stack input
0x1010_b2b2 0xffff_fffe 0x0000_0000 0x7fff_ffff 0x2020_c3c3 0xffff_ffff	0x2020_c3c3 0x7fff_ffff 0x0000_0000 0xffff_fffe 0x1010_b2b2	Tests values close to 0xffff_ffff to see if triggers stack popping

3. Figure 1: Assembly Code Part 1:

```
.data
ARRAY:.half 19, 3

.text
      li  s0,0x11000000 #MMIO ADDRESS
      la  s1,  ARRAY
      li t0, 0
      lhu t1, (s1) #first value in ARRAY
      lhu t2, 0x2(s1) #second value in ARRAY
SUB:  ble t1, x0, DIV #count how many times
      sub t1, t1, t2 #hw1 can be subtracted by hw2
      addi t0, t0, 1
      j SUB
DIV:  beq t1, x0 QTNT  #if remainder, remove 1 from sub
      add t1, t1, t2   #count and send to remainder
      addi t0, t0, -1
QTNT: sw t0, 0x40(s0) #store quotient to sevseg
      sw t1, 0x20(s0) #store remainder to LEDs
```

Figure 2: Assembly Code Part 2:

```
      li s0,  0x11000000 #MMIO ADDRESS
      li sp, 0x10000 #Stack Pointer
      li t1, 0xffffffff #evaluating constant
      lw t0, (s0) #first load
PUSH: beq t0, t1 LOADED #loop until switches is all high
      addi sp, sp, -4 #push switches to stack
      sw t0, (sp)
      lw t0, (s0) #load new switches
      j PUSH
LOADED:    li s1, 0x10000
POP:  beq sp, s1, END
      lw t0, (sp)#pop t0 from stack
      addi sp, sp, 4
      sw t0, 0x20(s0) #store t0 in LEDs
      j POP
END:  nop
#program cannot store more than 10239 itterations of switches
```