CPE 233: Software assignment 3

Prof. Bridget Benson

Luis Gomez, Jared Rocha

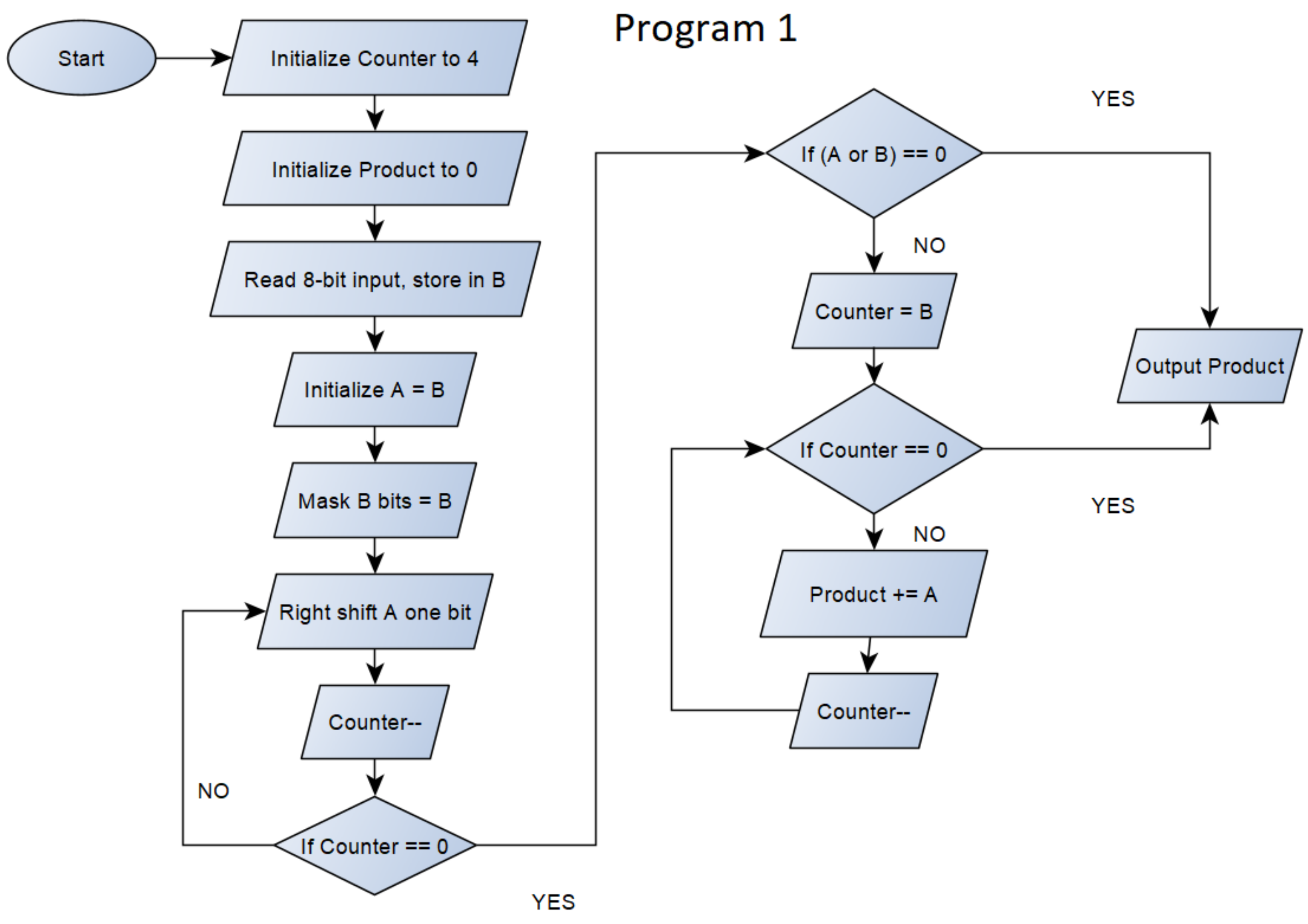
Behavior

In this assignment, we wrote two Assembly programs using the RAT simulator.

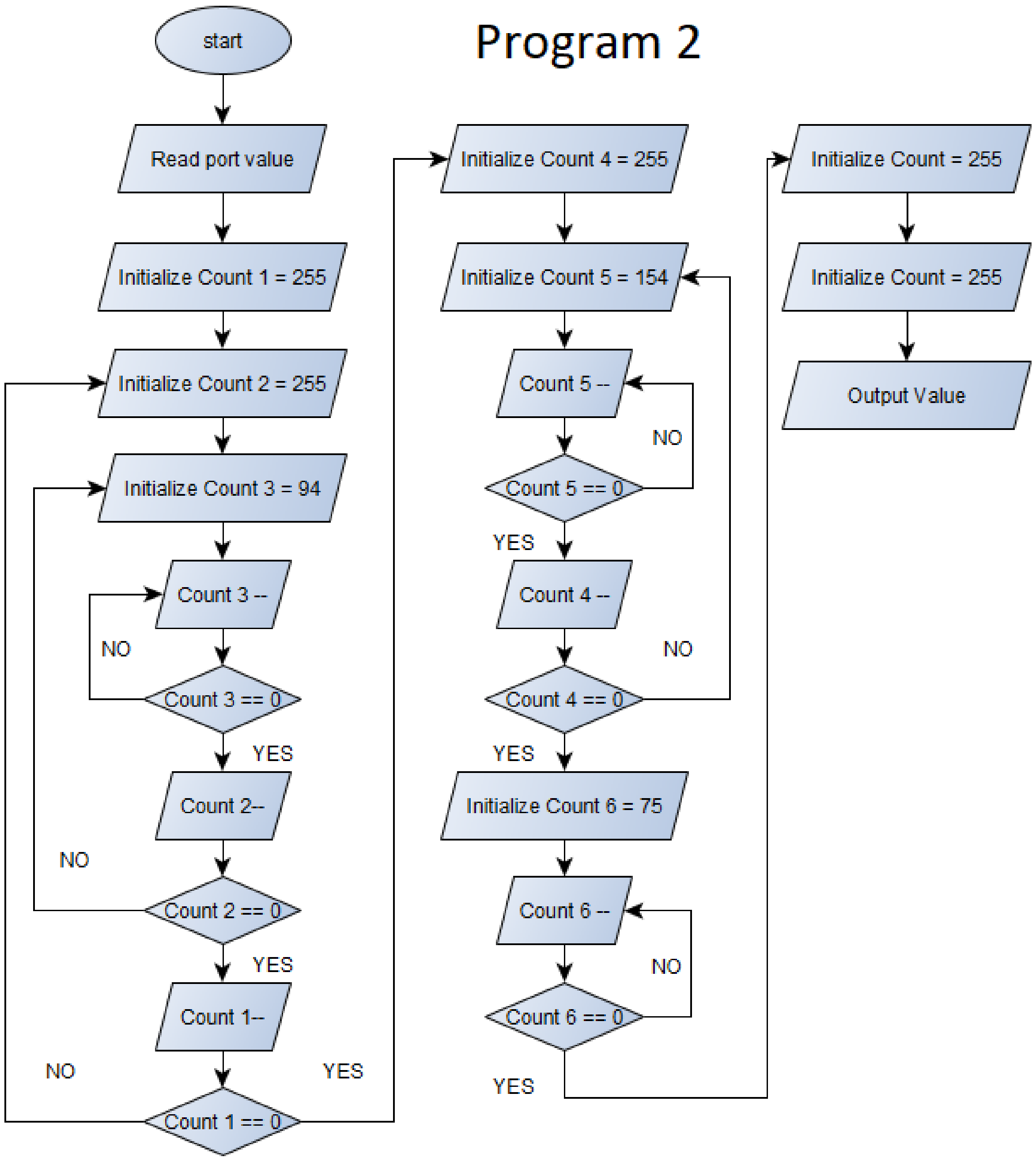
Program 1: A multiplier program that reads a single 8-bit value (port 0x9A), and outputs the product of A x B (port 0x42); where A is composed of the 4 most significant bits and B by the 4 least significant bits. Value B is generated by applying a mask to the 8-bit input to isolate the needed bits. Value A is generated by right-shifting the 8-bit value four times, clearing the carry with each iteration. When A & B are isolated, we check for zero and if not zero, we computer the product by adding A to a running total B times. The resulting product it output.

Program 2: A ½ second delay generator. This program reads a single 8-bit value (port 0x9A), generates a ½ second delay, and then outputs the value (port 0x42). The ½ second delay is generated through 3 sets of nested loops, details of which can be found in the verification section and in source code header.

Flowchart



Program 2



Verification

Program 1 Verification

Input	A	B	Expected Output	Output
0x0A	0	A	0 x 10 = 0	0x00
0xF0	F	0	15 x 0 = 0	0x00
0xFF	F	F	15 x 15 = 225	0xE1
0x44	4	4	4 x 4 = 16	0x10
0xC1	C	1	12 x 1 = 12	0x0C

Program 2 Verification

Let T = 40 ns, such that:

$T * (2 + [1 + 255(255 * (94 * 2) + 255 * 3) + 255 * 3] + [1 + 255(154 * 2) + 255 * 3] + [1 + 75 * 2])$ = .5 s

Source Code

PROGRAM 1: Multiplier

```
; Registers Used:
;     R0- counter variable
;     R2- input value, lower half of input (multiplier B)
;     R3- upper half of input, (multiplicand A)
;     R4- product (A * B)
;
; This program multiplies A * B, via iterations of addition

.EQU IN_PORT = 0x9A
.EQU OUT_PORT = 0x42
.EQU SIZE = 4

.CSEG
.ORG 0x01
                               ; SET-UP
     start:     MOV R0, SIZE    ; initialize counter
                MOV R4, 0x00    ; initialize product
                IN R2, IN_PORT  ; Read input value
                MOV R3, R2
     upper:     CLC             ; ISOLATE A LOOP
                LSR R3
                SUB R0, 1       ; counter--
                BRNE upper      ; END LOOP if R0 == 0
                CMP R4, R3      ; if A == 0, output 0
                BREQ end
                AND R2, 0x0F    ; ISOLATE B
                CMP R4, R2      ; if B == 0, output 0
                BREQ end
                MOV R0, R2      ; initialize multiplication counter
      mult:     ADD R4, R3      ; MULT LOOP
                SUB R0, 1       ; counter--
                BRNE mult       ; END LOOP if R0 == 0
       end:     OUT R4, OUT_PORT
```

PROGRAM 2: ½ Second delay

```
; Program which reads in a value, and generates a delay of .5 seconds, then
outputs value
;
; Given that each instruction ~ 40 ns, RAT MCU ~ 25MHZ. Let T = 40ns, such that:
;
;     T*{2+
;     [1+ 255(255*(94*2)+255* 3)+255*3]                (loop1(loop2(loop3)))
;     [1+ 255(154*2)+255* 3]                                (loop4(loop5))
;     [1+  75(2)]}                                          (loop6)
;     = .5 secs
;
; Registers Used:
;     R0- input value
;     R1- count 1, count 4, count 6
;     R2- count 2, count 5
;     R3- count 3

.EQU IN_PORT = 0x9A
.EQU OUT_PORT = 0x42
.EQU COUNT = 0xFF
.EQU COUNT_3 = 0x5E  ; 94
.EQU COUNT_5 = 0x9A  ; 154
.EQU COUNT_6 = 0x4B  ; 74

.CSEG
.ORG 0x01

              IN R0, IN_PORT        ; Start, read input
              MOV R1, COUNT         ; DELAY STARTS
     loop1:   MOV R2, COUNT         ; START LOOP 1
     loop2:   MOV R3, COUNT_3       ; START LOOP 2
     loop3:   SUB R3, 1             ; START LOOP 3, count3--
              BRNE loop3            ; END LOOP 3
              SUB R2, 1             ; count2--
              BRNE loop2            ; END LOOP 2
              SUB R1, 1             ; count1--
              BRNE loop1            ; END LOOP 1
              MOV R1, COUNT
     loop4:   MOV R2, COUNT_5       ; START LOOP 4
     loop5:   SUB R2, 1             ; START LOOP 5, count5--
              BRNE loop5            ; END LOOP 5
              SUB R1, 1             ; count4--
              BRNE loop4            ; END LOOP 4
              MOV R1, COUNT_6
     loop6:   SUB R1, 1             ; START LOOP ^, count6--
              BRNE loop6            ; END LOOP 6
              MOV R1, COUNT
              MOV R1, COUNT         ; DELAY ENDS
              OUT R0, OUT_PORT      ; end, output
```