EE234

Microprocessor Systems

Midterm Exam 2

Nov. 12, 2021. (2:10pm – 3pm)

Instructor: Dae Hyun Kim (daehyun@eecs.wsu.edu)

Name:

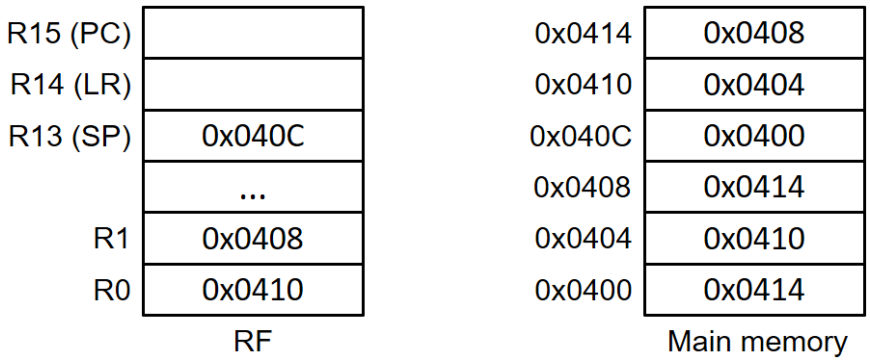
WSU ID:

Problem	Points	
1	10	
2	20	
3	40	
Total	70	

Problem #1 (Stack, 10 points)

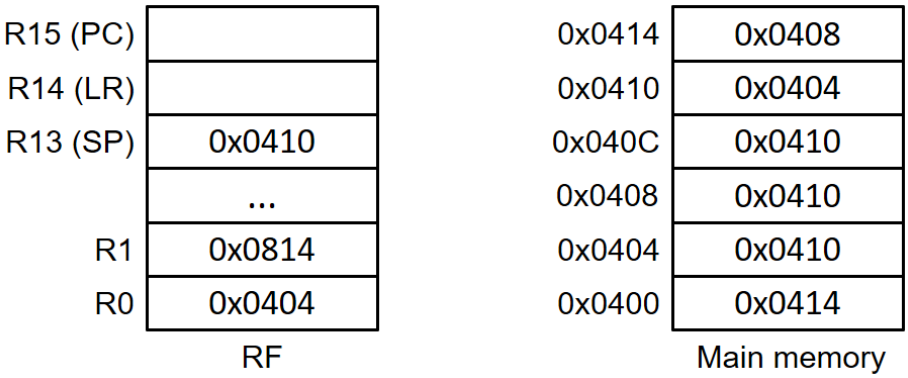
* Notice that we assume that SP points to the topmost available memory space.

The following shows the register file (RF) and the main memory map.



Fill in R0, R1, and the main memory map below after the following instructions are executed.

```
PUSH {R0}
PUSH {R0}
POP {R1}
POP {R0}
POP {R0}
ADD R1, R0, R1
```



Problem #2 (Stack, 20 points)

Answer the following questions for the assembly code shown below.

```
            main:
              MOV R0, #3
              BL run_1
Addr: A1      B end

            run_1:
              PUSH {LR}
              PUSH {R0}
              BL run_2
Addr: A2      POP {R0}
              ADD R1, R1, #2
              POP {LR}
              BX LR

            run_2:
              PUSH {LR}
              PUSH {R0}
              CMP R0, #1
              BEQ run_3
              SUB R0, R0, #1
              BL run_1
Addr: A3      ADD R1, R1, #1
              POP {R0}
              POP {LR}
              BX LR

            run_3:
              MOV R1, #3
              POP {R0}
              POP {LR}
              BX LR
            end:  // end of code
```

(1) R0 = 3, call run_1 (LR: A1)

(2) Stack {A1, 3}, call run_2 (LR: A2)

(3) Stack {A1, 3, A2, 3}, R0 != 1, so R0 = 2. call run_1 (LR: A3)

(4) Stack {A1, 3, A2, 3, A3, 2}, call run_2 (LR: A2)

(5) Stack {A1, 3, A2, 3, A3, 2, A2, 2}, R0 != 1, so R0 = 1. call run_1 (LR: A3)

(6) Stack {A1, 3, A2, 3, A3, 2, A2, 2, A3, 1}, call run_2 (LR: A2)

(7) Stack {A1, 3, A2, 3, A3, 2, A2, 2, A3, 1, A2, 1}, R0 == 1, so R1 = 3. Pop twice: R0 = 1, LR = A2. Stack {A1, 3, A2, 3, A3, 2, A2, 2, A3, 1}. Return.

(8) R0 = 1. R1 = R1 + 2 = 5. LR = A3. Stack {A1, 3, A2, 3, A3, 2, A2, 2}. Return.

(9) R1 += 1, so R1 = 6. Pop twice. R0 = 2, LR = A2. Stack {A1, 3, A2, 3, A3, 2}. Return.

(10) R0 = 2. R1 = R1 + 2, so R1 = 8. LR = A3. Stack {A1, 3, A2, 3}. Return.

(11) R1 += 1, so R1 = 9. Pop twice. R0 = 3, LR = A2. Stack {A1, 3}. Return.

(12) R0 = 3. R1 = R1 + 2, so R1 = 11. LR = A1. Stack {}. Return.

(13) end.

(1) (10 points) What is the value stored in R1 when the program ends? 11

(2) (10 points) How many times is the “PUSH {R14}” statement executed? 6

Problem #3 (Subroutines and Stack, 40 points)

```
main:
  MOV R0, #10
  BL run_1
  B end

run_1:
  PUSH {LR}
  CMP R0, #2
  BLT run_1_end
  SUB R0, R0, #1
  BL run_2
  POP {LR}
  BX LR
run_1_end
  MOV R1, #1
  POP {LR}
  BX LR

run_2:
  PUSH {LR}
  CMP R0, #2
  BLT run_2_end
  SUB R0, R0, #2
  BL run_1
  POP {LR}
  BX LR
run_2_end
  MOV R1, #2
  POP {LR}
  BX LR

end:  // end of code
```

Let's assume that the addresses of "BL run_1", the first "POP {LR}" in run_1, and the first "POP {LR}" in run_2 are A1, A2, and A3, respectively.

(1) R0 = 10, call run_1 (LR = A1)
(2) Stack {A1}, R0 = 9, call run_2
(3) Stack {A1, A2}, R0 = 7, call run_1
(4) Stack {A1, A2, A3}, R0 = 6, call run_2
(5) Stack {A1, A2, A3, A2}, R0 = 4, call run_1
(6) Stack {A1, A2, A3, A2, A3}, R0 = 3, call run_2
(7) Stack {A1, A2, A3, A2, A3, A2}, R0 = 1, call run_1
(8) Stack {A1, A2, A3, A2, A3, A2, A3}, **R1 = 1**, Stack {A1, A2, A3, A2, A3, A2}, LR = A3, return
(9) Stack {A1, A2, A3, A2, A3}, LR = A2, return
(10) Stack {A1, A2, A3, A2}, LR = A3, return
(11) Stack {A1, A2, A3}, LR = A2, return
(12) Stack {A1, A2}, LR = A3, return
(13) Stack {A1}, LR = A2, return
(14) Stack {}, LR = A1, return

(1) (10 points) What is the value stored in R1 when the program ends? 1

(2) (10 points) Will the code still work if the initial value of R0 ("MOV R0, #10") is very large (e.g., "LDR R0, =0xFFFFFFFF")? Explain if the code has any potential problem.

No, because the stack will overflow.

(3) (20 points) Rewrite the code (with or without subroutine calls) so that it can work without any problem for any initial value of R0 (assume that R0 has an unsigned value). Try to minimize the # lines (except address label lines) in your code. You can use R1-R12 for temporary registers.

If you code doesn't work, you will get max. 10 points. If you code works, you will get min. 10 points + extra points (# lines ≤15: +10 points, 16: +9 points, 17: +8 points, 18: +7 points, 19: +6 points, ≥20: +5 points)

For given R0, it performs the following function:

In function run_1, if R0 < 2, return 1, otherwise return run_2(R0-1).

In function run_2, if R0 < 2, return 2, otherwise return run_1(R0-2).

Thus, if R0 = 0 or 1, R1 = 1. If R0 = 2, R1 = 2. If R0 = 3, R1 = 1. If R0 = 4, R1 = 1. If R0 = 5, R1 = 2. If R0 = 6, R1 = 1. In sum,

if R0 = 3n+2, R1 = 2, otherwise (R0 = 3n or 3n+1), R1 = 1.

```
main:
  MOV R0, #10
  MOV R1, #1
main_loop:
  CMP R0, #0
  BEQ end
  SUB R0, R0, #1
  CMP R0, #0
  BEQ end
  SUB R0, R0, #1
  CMP R0, #0
  BEQ end_2
  SUB R0, R0, #1
  B main_loop
end_2:
  MOV R1, #2
end:  // end of code
```