EE234

Microprocessor Systems

Midterm Exam

Oct. 18, 2024. (2:10pm – 3pm)

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

Name:

WSU ID:

Problem	Points	
1	10	
2	20	
3	20	
4	20	
5	20	
Total	90	

Problem #1 (Bit manipulation, 10 points)

R# is an 8-bit register. The data stored in R# is treated as an unsigned binary number. R1 has an input data x in the range of $0 \leq x \leq 63$. The following two instructions perform an arithmetic operation. Explain what it does (i.e., briefly explain the meaning of the data stored in R2 in terms of arithmetic operations) or draw a graph of (R1 vs. R2). Here, “arithmetic” means something like addition, subtraction, multiplication, division (quotient), division (remainder), square root, transcendental functions, etc. Ignore overflow/underflow exceptions in the operations.

```
AND R2, R1, #0xFC

MOV R2, R2, LSL #2
```

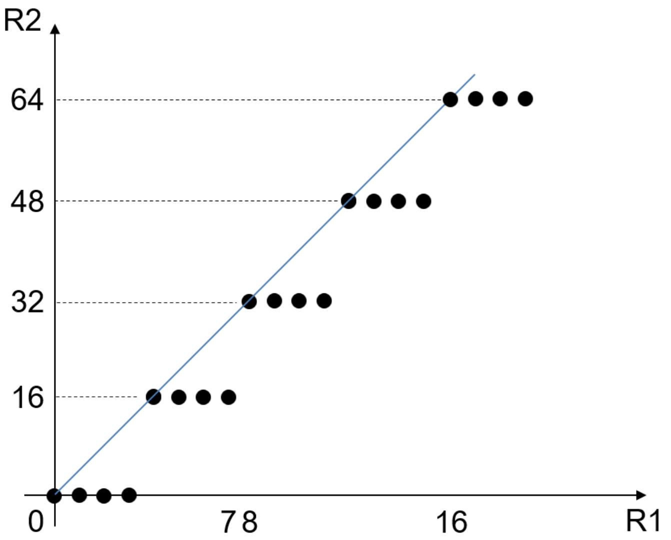
X in R1 = $00x_5x_4x_3x_2x_1x_0$.

R2 after AND = $00x_5x_4x_3x_200$.

Y in R2 after MOV = $x_5x_4x_3x_20000$.

Thus, $Y = 4 \cdot \{X - (X\ mod\ 4)\}$.

$$R_1 = 00x_5x_4x_3x_2x_1x_0$$
$$R_2 = x_5x_4x_3x_20000$$



Problem #2 (ARM assembly, 20 points)

```
main:
   MOV R6, #0
   MOV R1, #1
loop1:
   CMP R1, #6
   BGT loop1_end
   ADD R2, R1, #1
loop2:
   CMP R2, #6
   BGT loop2_end
   MOV R3, #0
   MOV R4, #1
loop3:
   CMP R4, R2
   BGT loop3_end
   ADD R3, R3, R1
   ADD R4, R4, #1
   B loop3
loop3_end:
   ADD R6, R6, R3
   ADD R2, R2, #1
   B loop2
loop2_end:
   ADD R1, R1, #1
   B loop1
loop1_end:
```

What is the value of the data stored in R6 when the above program ends?

```
int s = 0;

for ( int j = 1 ; j <= 6 ; j++ ) {
  for ( int k = j+1 ; k <= 6 ; k++ ) {
    t = 0;
    for ( m = 1 ; m <= k ; m++ )
      t = t + j;

    // t = j*k
    s = s + t;  // s = s + (j*k)
  }
}
```

s: R6
j: R1
k: R2
t: R3
m: R4

s = (1*2 + 1*3 + … + 1*6) + (2*3 + … + 2*6) + … + (4*5 + 4*6) + (5*6) = 1*20 + 2*18 + 3*15 + 4*11 + 5*6 = 20 + 36 + 45 + 44 + 30 = 175.

Problem #3 (ARM assembly, 20 points)

Translate the following C code into an assembly code.

```
int a, b, c, d;
…

while ( (a != 10) && (b < 5) && (c > 7) ) {
   b++;

   while ( (d > 6) || (d < 12) ) {
      c++;
      d--;
   }

   if ( a <= b ) {
      break;
   }
}
```

- Use the assembly instructions listed in the last page only.
- a is in R0, b is in R1, c is in R2, and d is in R3.
- The exit point (the end of the if statement) could be just an address label.

```
wh_1:
    CMP R0, #10
    BNE wh_2
    B wh_end
wh_2:
    CMP R1, #5
    BLT wh_3
    B wh_end
wh_3:
    CMP R2, #7
    BGT wh_body
    B wh_end
wh_body:
    ADD R1, R1, #1
wh_4
    CMP R3, #6
    BGT wh_body2
    CMP R3, #12
    BLT wh_body2
    B if
wh_body2:
    ADD R2, R2, #1
    SUB R3, R3, #1
    B wh_4
if:
    CMP R0, R1
    BLE wh_end
    B wh_1
wh_end:
```

Problem #4 (ARM assembly, 20 points)

```
main:                   // Addresses
  MOV R0, #0            // 0x00
  MOV R1, #1            // 0x04
label1:
  CMP R1, #11           // 0x08
  BGE end               // 0x0C
  BL sub1               // 0x10
  BL sub2               // 0x14
  ADD R1, R1, #1        // 0x18
  B label1              // 0x1C
sub1:
  MOV R2, #0            // 0x20
  MOV R3, #0            // 0x24
label2:
  CMP R2, R1            // 0x28
  BGE sub1_end          // 0x2C
  ADD R3, R3, R1        // 0x30
  ADD R2, R2, #1        // 0x34
  B label2              // 0x38
sub1_end:
  BX LR                 // 0x3C
sub2:
  ADD R0, R0, R3        // 0x40
  BX LR                 // 0x44
end:
  …                     // 0x48
```

The address column shows the addresses of the instructions.

1. What is the value of the data stored in R0 when the above program ends?

(Hint: Translate the code into a C code, and then analyze it.)

R0	R1	R2	R3	LR
0	1	0	0	
1	2	1	1	
		0	0	
		1	2	
1+4	3	2	4	
		0	0	
		1	3	
		2	6	

1+4+9	4	3	9	

Answer: 1+4+9+…+100 = 1^2 + 2^2 + 3^2 + … + 10^2 = 385.

2. What is the value of R14 (LR) when the above program ends?

0x18

Problem #5 (ARM assembly, 20 points)

Write an assembly code to compute x^4. Assume x is an unsigned integer and stored in R0. Store x^4 in R1. Use the assembly instructions listed in the last page only.

```
// R0 = x
// R1 = x^4
// R2 = t
// R3 = k

  MOV R2, #0
  MOV R3, #0
loop1:
  CMP R3, R0
  BGE loop1_end
  ADD R2, R2, R0
  ADD R3, R3, #1
  B loop1
loop1_end:
  MOV R1, #0
  MOV R3, #0
loop2:
  CMP R3, R2
  BGE loop2_end
  ADD R1, R1, R2
  ADD R3, R3, #1
  B loop2
loop2_end:
```