EE234

Microprocessor Systems

Final Exam

Dec. 12, 2022. (1:10pm – 4:00pm)

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

Name:

WSU ID:

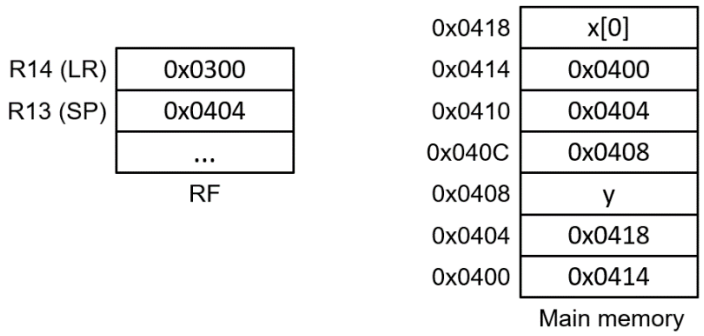
Problem	Points	
1	20	
2	20	
3	30	
4	30	
5	30	
Total	130	

Problem #1 (1D Array, 20 points)

We have two arrays as follows:

- int x[100]: Static array.
- int* y = new int[100]: Dynamic array.

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



Translate the following assembly code into a C code (hint: it is a “for” loop).

```
  MOV R0, #0
if:
  CMP R0, #34
  BLT loop
  B end
loop:
  MUL R1, R0, #3
  MUL R1, R1, #4
  ADD R1, R1, #20
  ADD R1, R1, SP
  LDR R2, [R1]
  ADD R2, R2, #3
  LDR R3, [SP, #4]
  ADD R4, R0, #1
  MUL R4, R4, #4
  ADD R4, R3, R4
  STR R2, [R4]
  ADD R0, R0, #1
  B if
end:
```

Let R0=k. Then, k=0. If k < 34, we are done. Otherwise, we go through the loop and at the end, we increase k by 1 (k++). This becomes a for loop.

for (int k = 0 ; k < 34 ; k++) {}

Now, R1 = SP + 20 + 4*3*k, so it is &(x[3*k]). R2 = x[3k]+3.

R3 = [SP+4] = y = &(y[0]).

R4 = y + 4*(k+1) = &(y[k+1]).

Thus, y[k+1] = x[3k] + 3.

The answer is

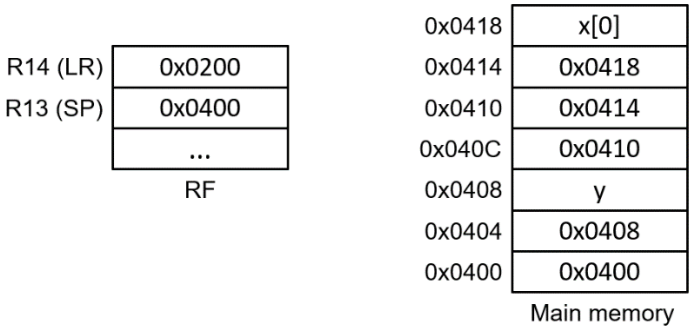
```
for ( int k = 0 ; k < 34 ; k++ ) {
  y[k+1] = x[3*k] + 3;
}
```

Problem #2 (1D Array, 20 points)

We have an array as follows:

- int* y = new int[100]: Dynamic array.

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



Write an assembly code for the following C code. R0-R12 are all available. You can use registers for “int a” and “int k”.

```
int a = y[0];

for ( int k = 0 ; k < 99 ; k++ ) {
  y[k] = y[k+1];
}

y[99] = a;
```

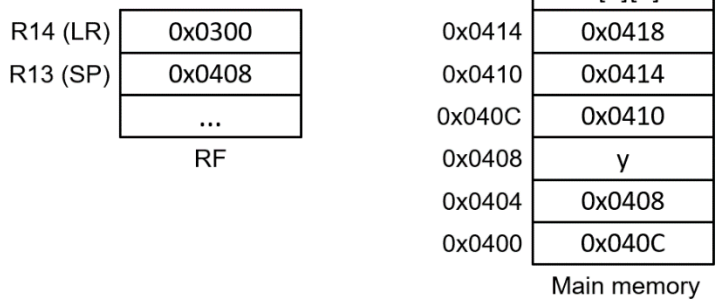
```
  LDR R7, [SP, #8]
  LDR R7, [R7]
  MOV R0, #0
for:
  CMP R0, #99
  BGE for_end
  LDR R1, [SP, #8]
  ADD R2, R0, #1
  MUL R2, R2, #4
  ADD R2, R1, R2
  LDR R2, [R2]
  MUL R3, R0, #4
  ADD R3, R1, R3
  STR R2, [R3]
  ADD R0, R0, #1
  B for
for_end:
  MOV R0, #99
  MUL R0, R0, #4
  ADD R1, R1, R0
  STR R7, [R1]
```

Problem #3 (2D Array, 30 points)

We have two arrays as follows:

- int x[10][10]: Static array.
- int** y = new int*[10]
- for (int k = 0 ; k < 10 ; k++)
    - y[k] = new int[10];

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



Write an assembly code for the following C code. R0-R12 are all available. You can use registers for “int k” and “int p”. This code converts the rows (or columns) of y to columns (or rows).

```
for ( int k = 0 ; k < 10 ; k++ ) {
  for ( int p = 0 ; p < 10 ; p++ ) {
    x[p][k] = y[k][p];
  }
}
```

```
  MOV R0, #0
for1:
  CMP R0, #10
  BGE for1_end
  MOV R1, #0
for2:
  CMP R1, #10
  BGE for2_end
loop:
  LDR R2, [SP]
  MUL R3, R0, #4
  ADD R3, R2, R3
  LDR R2, [R3]
  MUL R3, R1, #4
  ADD R3, R2, R3
  LDR R3, [R3]
  MUL R4, R1, #10
  MUL R4, R4, #4
  ADD R4, R4, #16
  ADD R4, R4, SP
  MUL R5, R0, #4
  ADD R4, R4, R5
  STR R3, [R4]
  ADD R1, R1, #1
  B for2
for2_end:
  ADD R0, R0, #1
  B for1
for1_end:
```

Problem #4 (2D Array, 30 points)

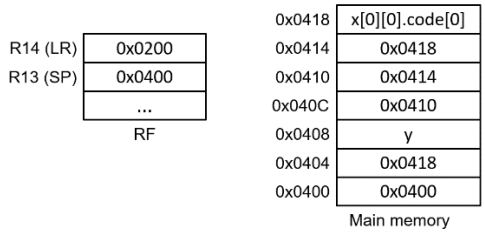
The following shows a structure definition.

```
struct Airport {
  int code[4];
};
```

We have two arrays as follows:

- Airport x[10][10]: Static array.
- Airport** y = new Airport*[10];
- for (int k = 0 ; k < 10 ; k++)
    - y[k] = new Airport[10];

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



Write an assembly code for the following C code. R0-R12 are all available. You can use registers for “int k” and “int p”.

```
for ( int k = 0 ; k < 10 ; k++ ) {
  for ( int p = 0 ; p < 10 ; p++ ) {
    y[k][p].code[2] = x[k][p].code[3];
  }
}
```

```
  MOV R0, #0
for1:
  CMP R0, #10
  BGE for1_end
  MOV R1, #0
for2:
  CMP R1, #10
  BGE for2_end
loop:
  ADD R2, SP, #24
  MUL R3, R0, #160
  ADD R2, R2, R3
  MUL R3, R1, #16
  ADD R2, R2, R3
  ADD R2, R2, #12
  LDR R2, [R2]
  LDR R3, [SP, #8]
  MUL R4, R0, #4
  ADD R3, R3, R4
  LDR R3, [R3]
  MUL R4, R1, #16
  ADD R3, R3, R4
  ADD R3, R3, #8
  STR R2, [R3]
  ADD R1, R1, #1
  B for2
for2_end:
  ADD R0, R0, #1
  B for1
for1_end:
```

Problem #5 (30 points)

See the following C code.

```
int p[5][20];

int* x = new int[100];

int** y = new int*[5];

for ( int k = 0 ; k < 5 ; k++ ) {
  y[k] = &(x[20*k]);
}
```

We want to copy the 2-D array “y” to the 2-D array “p”, but we will use “x” for “y” as follows:

```
for ( int k = 0 ; k < 100 ; k++ ) {
  p[A][B] = x[k];
}
```

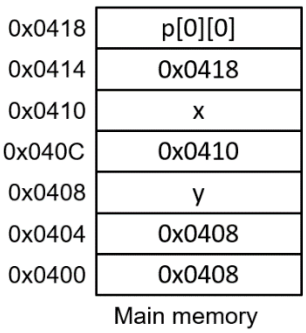
(1) Express “A” in “p[A][B]” as a function of “k” and some constants. (5 points)

A = k / 20

(2) Express “B” in “p[A][B]” as a function of “k” and some constants. (5 points)

B = k % 20

(3) Write an assembly code for the above code copying the array using “x”. R0-R12 are all available. You can use registers for “int k”. Use the following RF and main memory map. (20 points)




```
  MOV R0, #0
for:
  CMP R0, #100
  BGE end
  LDR R1, [SP, #16]
  MUL R2, R0, #4
  ADD R1, R1, R2
  LDR R1, [R1]
  ADD R2, SP, #24
  MUL R3, R0, #4
  ADD R2, R2, R3
  STR R1, [R2]
  ADD R0, R0, #1
  B for
end:
```