The University of Alabama in Huntsville
Electrical and Computer Engineering Department
CPE 221 01
Sample Final Exam Solution
Spring 2018

This test is closed book, closed notes. You may use a calculator. You should have the ARM reference packet. You must show your work to receive full credit.

1. (1 point) A _tristate buffer_ is used to control the access to a bus from the output of a register.
2. (1 point) _SRAM_ uses cross-coupled transistors to store one bit of memory.
3. (1 point) _DRAM_ requires refreshing.
4. (1 point) The principle of _spatial_ locality states that items close to an item referenced are likely to be referenced in the near future.
5. (1 point) (True or False) _False_ A two-way set associative cache has two sets.
6. (3 points) In an ARM computer, r2 contains a value of --3621 in decimal. What is the binary value of r1 after this instruction is executed?

   MVN r1, r2

```
           0     E
16        14     2
16       226     5
16      3621
3621 is         0000 0000 0000 0000 0000 1110 0010 0101
r2 is -3621, or 1111 1111 1111 1111 1111 0001 1101 1011
r1 is           0000 0000 0000 0000 0000 1110 0010 0100 or 0x0000 0E24
```

7. (3 points) In an ARM computer, r2 contains a value of 1698 in decimal. What is the binary value of r1 after this instruction is executed?

   NEG r1, r2

```
           0     6
16         6     A
16       106     2
16      1698
1698 is  0000 0000 0000 0000 0000 0110 1010 0010
-1698 is 1111 1111 1111 1111 1111 1001 0101 1110 and the value of r1 is
0xFFFF F95E
```

8. (2 points) In an ARM computer, r2 contains a value of -3621 in decimal while r3 contains a value of 1698 in decimal. What is the binary value of r1 after this instruction is executed?

OR r1, r2, r3

```
-3621     1111 1111 1111 1111 1111 0001 1101 1011
1698      0000 0000 0000 0000 0000 0110 1010 0010
R1        1111 1111 1111 1111 1111 0111 1111 1011 or 0xFFFF F7FB
```

9. (6 points) A RISC processor executes the following code. There are data dependencies. A source operand cannot be used until it has been written.

```
    LDR    r2, [r4]
    STR    r6, [r2]
```

Assuming a five-stage pipeline (fetch (IF), operand fetch (OF), execute (E), memory access (M), and register write (W)), how many extra cycles are required to ensure that the correct value of r2 is available for the STR instruction?

	1	2	3	4	5	6	7	8	9
LDR r2, [r4]	IF	OF	E	M	W				
STR r6, [r2		IF	OF	OF	OF	OF	E	M	W

The STR is in OF for four cycles, three extra cycles.

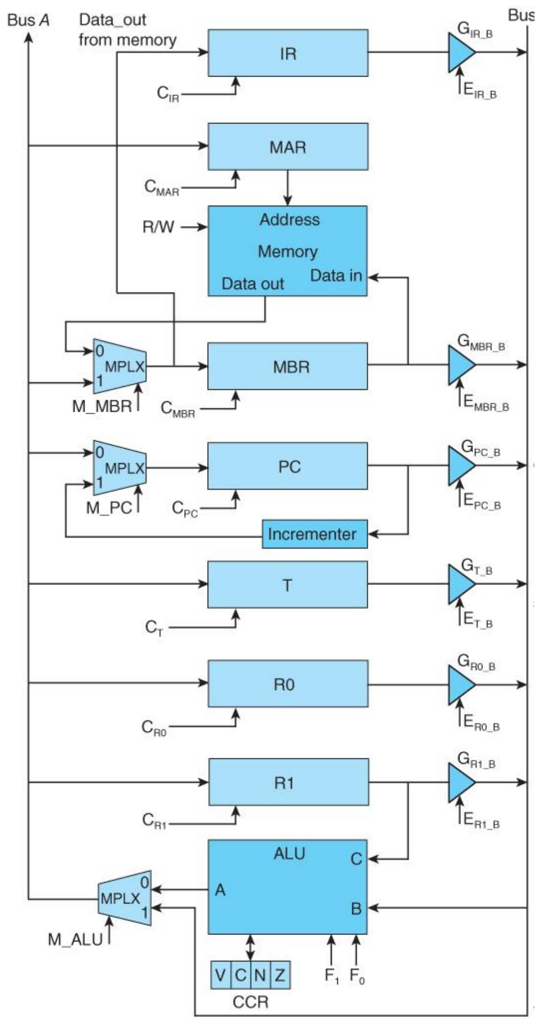
10. (15 points) (a) (9 points) What are the values of the following registers when the program executes "B loop" for the sixth time? Answer in decimal.

r5: _4_ r6:_91_ r2: _164__

(b) (6 points) Consider the STRGT and STRLE instructions that immediately precede the sixth execution of "B loop". What value is written by the STRGT instruction and what value is written by the STRLE instruction? Answer in decimal. STRGT _not executed in this iteration as the comparison of r5 and r6 yields less than_ STRLE _91 is stored at address 0x0000 00A0_

```
        AREA    PROB_11, CODE, READONLY
        ENTRY
        ADR     r0, x
        ADR     r1, y
        ADR     r2, z
        LDR     r3, size
        LDR     r4, i
loop    CMP     r4, r3
        BPL     done
        LDR     r5, [r0], #4
        LDR     r6, [r1], #4
        CMP     r5, r6
        STRGT   r5, [r2], #4
        STRLE   r6, [r2], #4
        ADD     r4, r4, #1
        B       loop
done    B       done
x       DCD     100, 3, -1, 2, 4, 4, 2, -1, 3, 100
y       DCD     -53, 247, 95, -7, 481, 91, -33, -1500, 29, -83
z       SPACE    40
i       DCD     0
size    DCD     10
        END
```

11. (15 points) For the architecture shown, write the concrete RTL and the sequence of signals and control actions necessary to execute the instruction `STRI (P), R0, R1`, that stores the sum of D0 and D1 in the memory location pointed to by the contents of the memory location P. Assume that the address P is in the instruction register IR. Abstract RTL: `M[M[P]] ← R0 + R1`.



F_1	F_0	Operation
0	0	A = B + 1
0	1	A = B – 1
1	0	A = B + C
1	1	A = B - C

Cycle	Concrete RTL	Signals
1	**MAR ← IR**	**E_{IR_B}, M_ALU = 1, C_{MAR}**
2	**MBR ← M[MAR]**	**Read, M_MBR = 0, C_{MBR}**
3	**MAR ← MBR**	**E_{MBR_B}, M_ALU = 1, C_{MAR}**
4	**MBR ← R1 + R0**	**E_{R0_B}, F = 10, M_ALU = 0, M_MBR = 1, C_{MBR}**
5	**M[MAR] ← MBR**	**Write**

12. (10 points) A certain memory system has a 1024 MB main memory and a 64 MB cache. Blocks are 4 words and each word is 64 bits. Show the fields in a memory address if the cache is 16-way set associative.

64 MB x 1 word/8 bytes x 1 block/4 words x 1 set/16 lines = 128 K sets
Byte offset = 3 bits
Block offset = 2 bits
Index = 17 bits
1024 MB main memory has 30 bits, leaving 8 bits of tag

13. (6 points) If you want to build a 2^{48} word, 64-bits-per-word memory and the only parts you have available to you are static RAM chips that contain 2^{40} 8 bit words each. (a) (2 points) How many rows are required? (b) (2 points) How many columns are required? (c) (2 points) How many chips in all?

a. $2^{48}/2^{40} = 2^8 = 256$
b. 64/8 = 8
c. 256 x 8 = 2048

14. (20 points) Complete the ARM assembly language program below so that it implements the following C++ statements. You must store the maximum value found in the array in the memory location pointed to by the label `max`.

```
int max;
int size = 10;
int nums[10] = {5, 3, -1, 2, 4, 37, -100, 13, -5, 0};
max = nums[0];
for (i = 1; i < size; i++)
   if (nums[i] > max)
      max = nums[i];
;
;      This program finds the maximum value in an array
;
      AREA ARRAY_MAX, CODE
      ENTRY
      ADR    r10, array          ; pointer to first element of array
      LDR    r1, size            ; holds size of array
      LDR    r2, [r10]           ; max = num[0]
      MOV    r4, #1
loop  SUBS   r5, r4, r1          ; Check to see whether index < size.
      BPL    done                ; If not, done.
      LDR    r7, [r10, #4]!      ; r7 = array[i], starting with 1
      SUBS   r8, r7, r2
      BMI    next                ; if minus, array[i] < max
      MOV    r2, r7              ; else max = array[i]
next  ADD    r4, r4, #1          ; i++
      B      loop
done  STR    r2, max             ; store max value in r2 at label max
stop  B      stop
size  DCD    10
max   SPACE  4
array DCD    5, 3, -1, 2, 4, 37, -100, 13, -5, 0
      END
```

15. (15 points) Consider the following ARM program. Trace the stack activity, including all changes to the stack pointer and to the contents of the stack. Clearly indicate the value of the sp.

```
0          MOV    sp, #0x00000000
4          B      main
8    swap  SUB    sp, sp, #4
12         LDR    r1, [sp, #8]
16         LDR    r2, [r1]
20         STR    r2, [sp]
24         LDR    r0, [sp, #4]
28         LDR    r3, [r0]
32         STR    r3, [r1]
36         LDR    r3, [sp]
40         STR    r3, [r0]
44         ADD    sp, sp, #4
48         MOV    pc, lr
52   main  SUB    sp, sp, #8
56         ADR    r6, x
60         STR    r6, [sp, #4]
64         ADR    r6, y
68         STR    r6, [sp]
72         BL     swap
76         ADD    sp, sp, #8
80   stop  B      stop
84   x     DCD    2
88   y     DCD    3
```

Address	Value
FFFF FFF0	
FFFF FFF4	
FFFF FFF8	
FFFF FFFC	

Instruction: 0 MOV sp, #0x00000000, sp = 0

Address	Value
FFFF FFF0	
FFFF FFF4	
FFFF FFF8	
FFFF FFFC	

Instruction: 52 main SUB sp, sp, #8

Address	Value
FFFF FFF0	
FFFF FFF4	
FFFF FFF8	
FFFF FFFC	84

Instruction: 60 STR r6, [sp, #4]

Address	Value
FFFF FFF0	
FFFF FFF4	
FFFF FFF8	88
FFFF FFFC	84

Instruction: 68 STR r6, [sp]

Address	Value
FFFF FFF0	
FFFF FFF4	
FFFF FFF8	88
FFFF FFFC	84

Instruction: 8 swap SUB sp, sp, #4

Address	Value
FFFF FFF0	
FFFF FFF4	2
FFFF FFF8	88
FFFF FFFC	84

Instruction: 20 STR r2, [sp]

Address	Value
FFFF FFF0	
FFFF FFF4	2
FFFF FFF8	88
FFFF FFFC	84

Instruction: 44 ADD sp, sp, #4

Address	Value
FFFF FFF0	
FFFF FFF4	2
FFFF FFF8	88
FFFF FFFC	84

Instruction: 76 ADD sp, sp, #8, sp = 0

Address	Value
FFFF FFF0	
FFFF FFF4	
FFFF FFF8	
FFFF FFFC	

Instruction: ______________________________