The University of Alabama in Huntsville
Electrical and Computer Engineering Department
CPE 221 01
Test 2 Solution
Spring 2019

This test is closed book, closed notes. You may use a calculator. You should have the 6 page ARM Instruction Reference. Please check to make sure you have all 5 pages of the test. You must show your work to receive full credit.

1. (1 point) _Concrete_ RTL describes what can be done in a single clock cycle.
2. (1 point) A _microarchitecture_ represents the organization of the computer in terms of registers, functional units, and buses.
3. (1 point) Microprogrammed control is the alternative to _hardwired_ control.
4. (1 point) True/False _False_ The amount of time it takes to execute a single instruction will decrease as a result of pipelining.
5. (1 point) The _scope_ of a variable defines the range of its visibility or accessibility within a program.
6. (10 points) A processor executes an instruction in the following eight stages. The time required by each stage in picoseconds (1,000 ps = 1 ns) is:

F	Fetch	180 ps
D	Decode	200 ps
O	Register Read Operands	120 ps
E1	Execute 1	220 ps
E2	Execute 2	220 ps
MR	Memory Read	400 ps
MW	Memory Write	400 ps
WB	Result Writeback to Register	120 ps

   a. (5 points) What is the time taken to fully execute an instruction assuming that this structure is pipelined in eight stages and that there is an additional 15 ps per stage due to the pipeline latches?
   8 * (max(F, D, O, E1, E2, MR, MW, WB) + latch) = 8 * (max(180, 200, 120, 220, 220, 400, 400, 120) + 15) ps = 8 * (400 + 15) = 8 * 415 = 3320 ps

   b. (5 points) What is the time to execute an instruction if the processor is not pipelined?
   1 * (F + D + O + E1 + E2 + MR + MW + WB) = 1 * (180 + 200 + 120 + 220 + 220 + 400 + 400 + 120) ps = 1860 ps

7. (20 points) Write the code to implement the expression A = (B – C/D)/(E + F*G) on 3-, 2-, 1-, and 0-address machines. Do not rearrange the expression. In accordance with programming language practice, computing the expression should not change the values of its operands. When using a 0-address machine, the order used is SOS op TOS, where SOS is second on stack and TOS is top of stack.

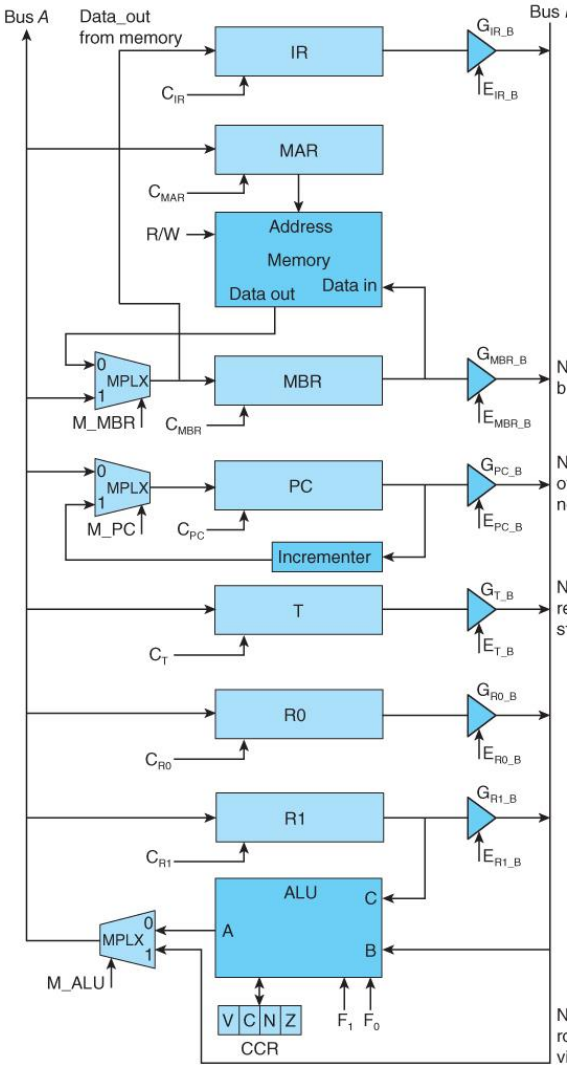
3-address	2-address	1-address	0-address
DIV A, C, D	LOAD A, B	LDA F	PUSH B
SUB A, B, A	LOAD T, C	MPY G	PUSH C
MPY T, F, G	DIV T, D	ADD E	PUSH D
ADD T, T, E	SUB A, T	STA T	DIV
DIV A, A, T	LOAD T, F	LDA C	SUB
	MPY T, G	DIV D	PUSH F
	ADD T, E	STA A	PUSH G
	DIV A, T	LDA B	MPY
		SUB A	PUSH E
		DIV T	ADD
		STA A	DIV
			POP A

8. (20 points) Complete the ARM assembly language program below so that it implements the following C++ statements.

```
;
;       This program adds two arrays, element by element into a third
;       array. It also writes a 0 into an array called sign if the sum
;       sum is negative and a 1 into the array called sign if the sum
;       is positive.
;
;       const int size = 10;
;       int x[size] = {100, 3, -1, 2, 4, 4, 2, -1, 3, 100};
;       int y[size] = {-53, 247, 95, -7, 481, 91, -33, 1500, 29, -83};
;       int z[size];
;       int sign[size];
;       int i;
;       for (i = 0; i < size; i++)
;       {
;         z[i] = x[i] + y[i];
;         if (z[i] < 0)
;           sign[i] = 0;
;         else
;           sign[i] = 1;
;       }
```

```
        AREA    PROB_8, CODE, READEXECUTE
        ENTRY
        ADR     r0, x
        ADR     r1, y
        ADR     r2, z
        LDR     r3, size
        LDR     r4, i
        ADR     r5, sign
loop    CMP     r4, r3
        BGE     done
        LDR     r6, [r0, r4, LSL #2]
        LDR     r7, [r1, r4, LSL #2]
        ADDS    r6, r6, r7
        STR     r6, [r2, r4, LSL #2]
        MOVGE   r7, #1
        MOVLT   r7, #0
        STR     r7, [r5, r4, LSL #2]
        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
sign    SPACE   40
i       DCD     0
size    DCD     10
        END
```

9. (20 points) For the architecture shown, write the sequence of signals and control actions necessary to execute the instruction `STRI (P), R0, R1`, that stores the sum of R0 and R1 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. The actions of this instruction are described by the abstract RTL `M[M[P]] ← R0 + R1`.



F_1	F_0	Operation
0	0	A = B′
0	1	A = B
1	0	A = B + C
1	1	A = B + 1

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

10. (10 points) For the figure of Problem 9, write the sequence of signals and control actions necessary to implement the fetch cycle.

Cycle	Concrete RTL	Signals
1	MAR ← PC	EPC_B, F = 01, M_ALU = 0, CMAR
2	IR ← M[MAR], PC ← PC + 1	READ, M_MBR = 0, CIR, CPC, M_PC = 1

11. (15 points) A RISC processor executes the following code. There are data dependencies but no internal forwarding. A source operand cannot be used until it has been written. Assume that the first instruction begins executing in the first cycle.

a. (6 points)) Assuming a four-stage pipeline (fetch (IF), operand fetch (OF), execute (E), operand write (W)), what registers are being read during the eighth clock cycle and what register is being written? **Reading r3, r6, Writing r1**
The dependences that exist are:
a-b, a-e, b-d, b-f, c-f

	1	2	3	4	5	6	7	8	9	10	11	12
(a) MUL r0, r1, r2	IF	OF	E	W								
(b) ADD r3, r1, r0		IF	OF	OF	OF	E	W					
(c) LDR r1, [r2]			IF	IF	IF	OF	E	W				
(d) ADD r5, r3, r6						IF	OF	OF	E	W		
(e) ADD r6, r0, r7							IF	IF	OF	E	W	
(f) STR r3, [r1]									IF	OF	E	W

b. (9 points) Assuming a six-stage pipeline: fetch (F), register read (O), execute (E), memory read (MR), memory write (MW), register write (WB), how long will it take to execute the entire sequence? **18 cycles**

	1	2	3	4	5	6	7	8	9	10	11
MUL r0, r1, r2	F	O	E	MR	MW	WB					
ADD r3, r1, r0		F	O	O	O	O	O	E	MR	MW	WB
LDR r1, [r2]			F	F	F	F	F	O	E	MR	MW
ADD r5, r3, r6								F			
ADD r6, r0, r7											
STR r3, [r1]											

	12	13	14	15	16	17	18	19	20	21	22
MUL r0, r1, r2											
ADD r3, r1, r0											
LDR r1, [r2]	WB										
ADD r5, r3, r6	O	E	MR	MW	WB						
ADD r6, r0, r7	F	O	E	MR	MW	WB					
STR r3, [r1]		F	O	E	MR	MW	WB				