1	/4
2	/6
3	/9
4	/11

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

6.004 Computation Structures
Spring 2015

Quiz #2: March 20, 2015

Name	*Athena login name*	*Score*

Michael	*Philippe*	*Miriam*	*Ciara*	*Louis*
☐ WF 10, 34-302	☐ WF 12, 34-301	☐ WF 2, 34-301	☐ WF 12, 34-302	☐ WF 2, 34-302
☐ WF 11, 34-302	☐ WF 1, 34-301	☐ WF 3, 34-301	☐ WF 1, 34-302	☐ WF 3, 34-302

Please enter your name and Athena login name in the spaces above. Enter your answers in the spaces provided after each question. You can use the extra white space and the backs of the pages for scratch work.

Problem 1. Beta Assembly (4 points)

For the Beta instruction sequence shown below, indicate the values in the specified registers after the sequence has been executed starting at location 0. Execution terminates when the HALT() instruction is reached. Assume that all registers have been initialized to 0 before execution begins.

Remember that even though the Beta reads and writes 32-bit words from memory, all addresses are *byte addresses*, i.e., the addresses of successive words in memory differ by 4.

A summary of Beta instructions is attached at the end of the quiz.

```
    . = 0
    LD(r31, X, r0)
    CMPLE(r0, r31, r1)
    BNE(r1, L1, r1)
    ADDC(r31, 17, r2)
    BEQ(r31, L2, r31)
L1: SRAC(r0, 4, r2)
L2: HALT()

    . = 0x1CE8
X:  LONG(0x87654321)
```

Value left in R0? 0x 87654321

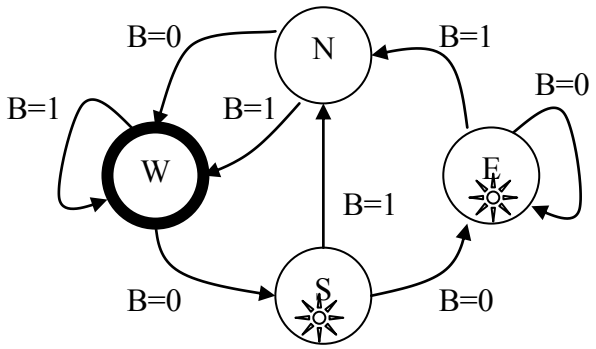
Value left in R1? 0x C

Value left in R2? 0x F8765432

Value UASM assigns to L1: 0x 14

Problem 2. Finite-state Machines (6 points)

Below is a state transition diagram for a 4-state FSM with a single binary input B. The FSM has single output – a light that is "on" when the FSM is in states "E" or "S". The starting state, "W", is marked by the heavy circle.



(A) (1 Point) Does this FSM have a set or sets of equivalent states that can be merged to yield an equivalent FSM with fewer states?

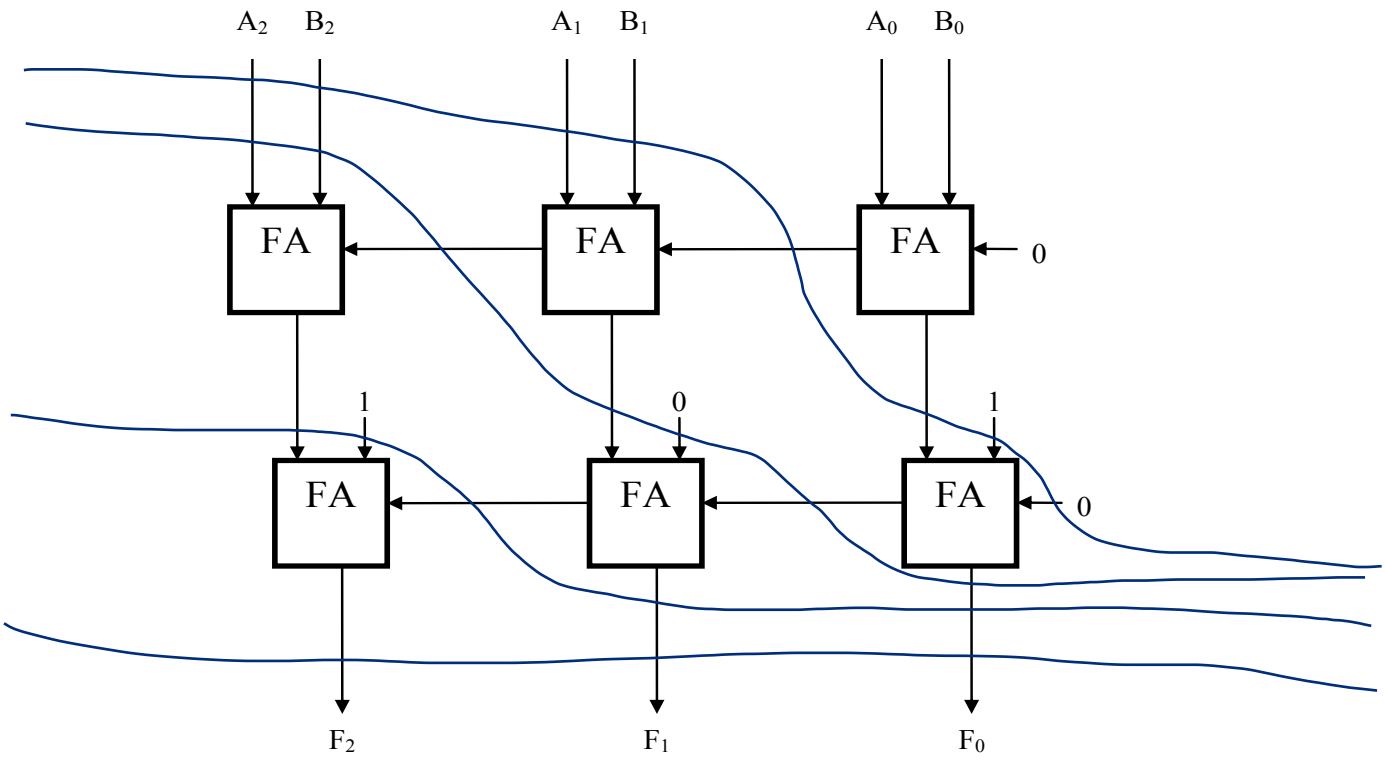
List set(s) of states that can be merged or write NONE: S, E

(B) (5 Points) Please fill in as many entries as possible in the following truth table for the FSM. The *light* output is a function of the current state and should be 1 when the light is "on" and 0 when it's "off."

S1	S0	B	S1'	S0'	light
0	0	0	0	0	1
0	0	1	1	0	1
0	1	0	0	0	1
0	1	1	1	0	1
1	0	0	1	1	0
1	0	1	1	1	0
1	1	0	0	1	0
1	1	1	1	1	0

Problem 3. Pipelining (9 points)

The following circuit uses six full adder modules (as you've seen in lecture and lab) arranged in a combinational circuit that computes a 3-bit value **F=A+B+5** for 3-bit inputs **A** and **B**:



The full adders have a t_{PD} of 6ns.

(A) (1 Point) Give the latency and throughput of the combinational circuit.

Latency: 24 **ns; Throughput:** 1/24

(B) (4 Points) Indicate, on the above diagram, appropriate locations to place ideal (zero-delay) registers to pipeline the circuit for *maximum throughput* using a *minimum number of registers*. Be sure to include a register on each output.

(mark circuit above)

(C) (2 Points) Give the latency and throughput of your pipelined circuit.

Latency: 24 **ns; Throughput:** 1/6

(D) (2 Points) Consider generalizing the circuit to accommodate A and B values with N bits. Give the asymptotic hardware cost of the **pipelined** circuit as a function of N.

Asymptotic hardware cost: Θ(N^2 **)**

Problem 4. Stacks & Procedures (11 points)

The following C program computes the log base 2 of its argument. The assembly code for the procedure is shown on the right, along with a stack trace showing the execution of ilog2(10). The execution has been halted just as it's about to execute the instruction labeled "rtn:"

```
/* compute log base 2 of arg */
int ilog2(unsigned x) {
    unsigned y;
    if (x == 0) return 0;
    else {
        /* shift x right by 1 bit */
        y = x >> 1;
        return ilog2(y) + 1;
   }
}
```

```
ilog2: PUSH(LP)
       PUSH(BP)
       MOVE(SP,BP)
       ALLOCATE(1)
       PUSH(R1)

       LD(BP,-12,R0)
       BEQ(R0,rtn,R31)

       LD(BP,-12,R1)
       SHRC(R1,1,R1)
       ST(R1,0,BP)

       LD(BP,0,R1)
       PUSH(R1)
       BR(ilog2,LP)
       DEALLOCATE(1)
       ADDC(R0,1,R0)

rtn:   POP(R1)
xxx:   DEALLOCATE(1)
       MOVE(BP,SP)
       POP(BP)
       POP(LP)
       JMP(LP)
```

(A) (4 Points) What are the values in R0, SP, BP and LP at the time execution was halted? Please express the values in hex or write "CAN'T TELL".

Value in R0: 0x 1 **in SP: 0x** 24C

Value in BP: 0x 244 **in LP: 0x** 1A8

(B) (5 Points) Please fill in the values for the five blank locations in the stack trace shown on the right. Please express the values in hex.

Fill in values (in hex!) for 5 blank locations

Values are in hex!

	5
	1A8
	208
	2
	5
	2
	1A8
	21C
	1
	2
	1
	1A8
	230
BP→	0
	1
	0

(C) (1 Point) In the assembly language code for ilog2 there is the instruction "LD(BP,-12,R0)". If this instruction were rewritten as "LD(SP,NNN,R0)" what is correct value to use for NNN?

Correct value for NNN: -20

(D) (1 Point) In the assembly language code for ilog2, what is the address of the memory location labeled "xxx:"? Please express the value in hex.

Address of location labeled "xxx:": 0x 1B4

END OF QUIZ 2!