CE2003 Digital Systems Design, Sem 2 2018/19

Solver: Ng Joshua Jeremiah

E-Mail Address: NGJO0009@e.ntu.edu.sg

Question 1: Multiple Choice Questions

Section A

1. **D**

   (The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand.)

2. **B**

   ($stop suspends the simulation, $finish makes the simulator exit and passes control back to the host OS.)

3. **D**

   (There are only three valid magnitudes for use in time_unit arguments in Verilog. 1, 10, 100)

4. **E**

   (Timescale 10ns:1ns. Step 1: 10.6 * 10ns = 106ns. Step 2: Round it to the nearest 1ns = 106ns. It will run the statement after 106ns.)

5. **C**

   (#5 + #15 = #20 = 20 time units)

6. **A**

   (Negative numbers can be specified by putting a minus sign before the size for a constant number)

7. **B**

   ("=" is blocking assignment and "<=" is nonblocking assignment.)

8. **B**

   (*NOT* combinational logic ⇒ latch is inserted)

9. **D**

   (Concatenate ‘{a,b}’ has to be used instead of brackets (a,b) for extending bits)

10. **D**

    (eg. Always #1 clk=~clk)

11. **B**

    (if a=1’d1, 2’b11=1’d3; 1’d1 != 1’d3, hence it’s false. Y=false=1’b0.)

12. E

(An always block acts as an infinite loop. Instantiating a module will result in stack overflow.)

13. A

(A=0; B[0]=0; C[1]=0; Hence, concatenating then will result in 3'b000.)

14. E

(3 integer bits and 5 frac bits can represent -2^3 to (2^3-2^-5), B and D fits in this range)

15. A

(worst case error is half of the smallest representable bit y, hence 2^-(y+1).)

Section B

16. D

(required int bits: Smallest int no = -2.2*0.8 - 1.2 = -2.96 (3 bits). Hence, frac bits for highest accuracy = 16-3 = 13 bits.)

17. A

(Just draw the table and it'll be obvious. Remember to read from **bottom to top**.)

18. D

(Note that branch takes 1 cycle to execute hence Add $3, $2 will run every loop, and that BGT runs before Add $3, $2 reaches the ALU stage.)

19. C

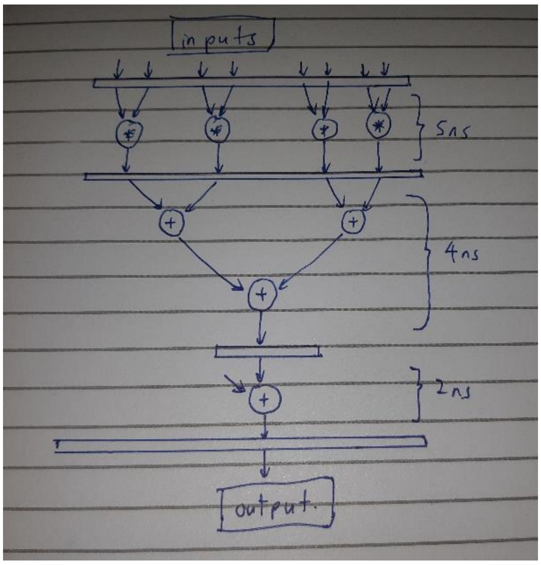
(Draw out the state transition diagrams, you can make use of the blank state or same letters in other states for double transition. Eg for PQ=01, State D -> State C -> State A.)

Question 2

(a) Information given: Tadd= 2ns, Tmul = 5ns, TcQ = 0.4ns and Ts = 0.1ns

i. Critical Path Delay = TcQ + Tmul + 3(Tadd) + Ts = 11.5ns
Max Freq = 1/11.5ns = 87.0MHz
Throughput = 1 sample every 11.5ns
Latency = 11.5ns

ii.

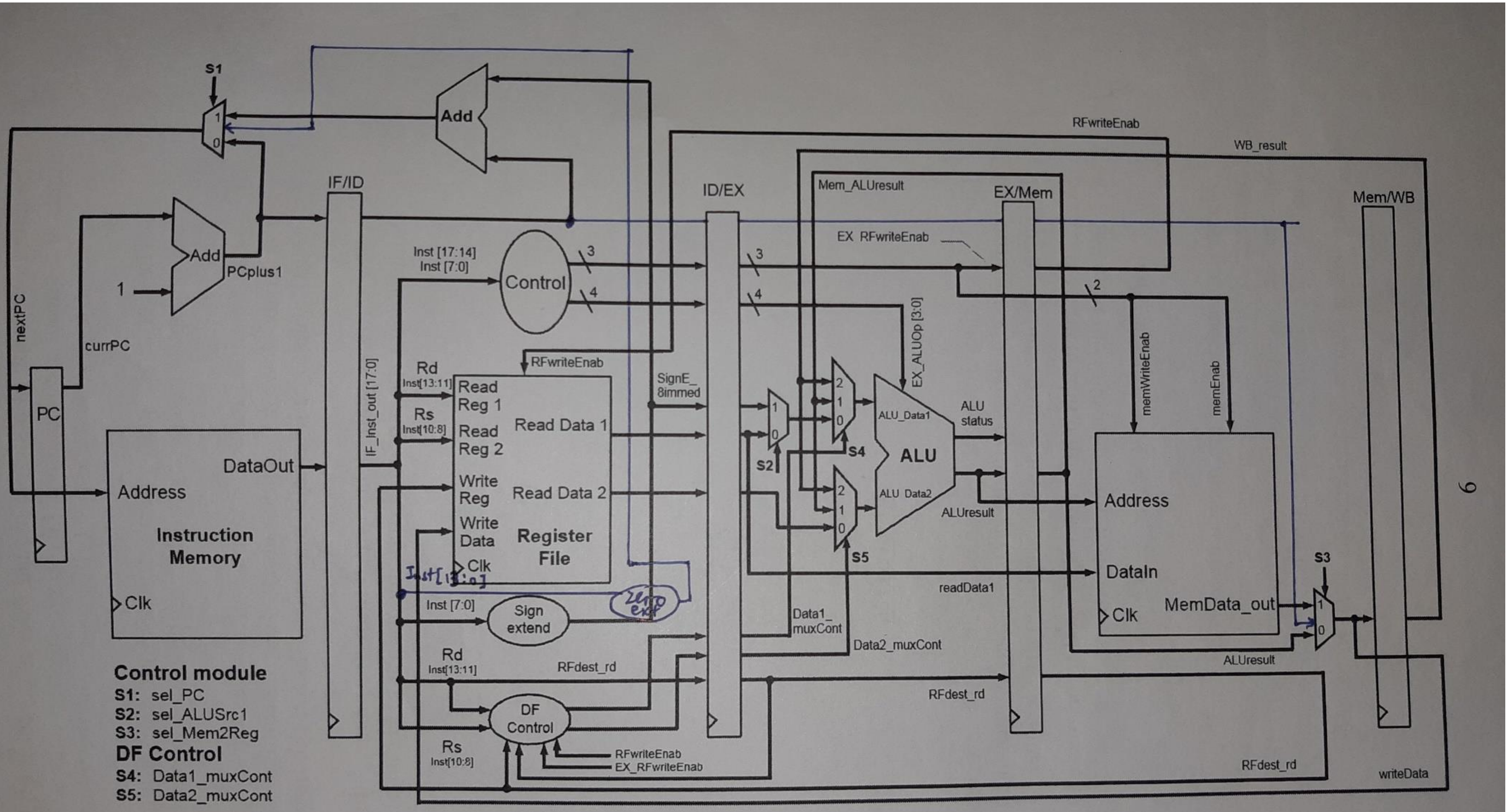


iii. Critical Path Delay = TcQ + Tmul + Ts = 5.5ns
Max Freq = 1/5.5ns = 181.8MHz
Throughput = 1 sample evert 5.5ns
Latency = 3*5.5=16.5ns

(b)

1) In the ID stage: inst[13:0] needs to be zero extended and fed to the S1 mux. The control module now needs to generate a 2-bit S1 signal.

2) PCplus1 needs to be fed through the pipeline registers to the S3 mux. The control module would need to generate a 2-bit S3 signal.



Control module

S1: sel_PC

S2: sel_ALUSrc1

S3: sel_Mem2Reg

DF Control

S4: Data1_muxCont

S5: Data2_muxCont

Question 3

```
(a) Always @*
            ns=s;
            Case({X,Y})
                    00: if (s == C)
                            ns = A;
                    01: if (s == B)
                            ns = C;
                    11: if ((s==A) || (s==B))
                            ns = C;
                    10: if (s==A)
                            ns=B;
            endcase
    assign Z = (s==B) ? 1 : 0;
    end
```

Is this synchronous part necessary? The above one is for state transitions, below is for reset and putting ns into s.

```
Always @ (posedge clk) begin
        If (rst)
                s<=A;
        else
                s<=ns;
end
```

(b) If B=01 and C=10, there will be dynamic hazard when changing between those states. Hence we have to create a transition state of D=11.

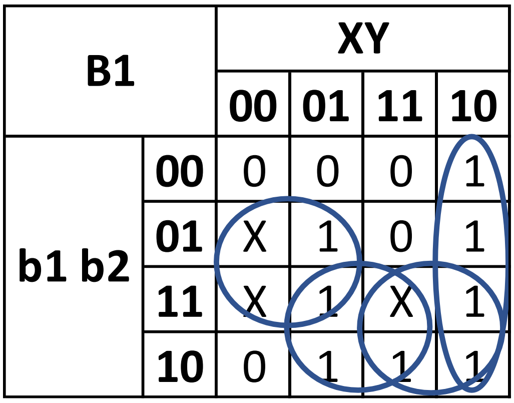
The resulting state transition table will look like this:

Present State	Input XY =				Output
	00	01	11	10	(Z)
A (00)	A	A	B	C	0
B (01)	--	D	B	D	1
D (11)	--	C	--	C	1
C (10)	A	C	C	C	0

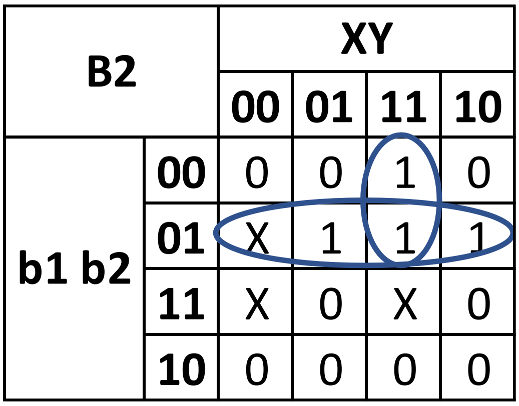
The excitation table will hence look like:

Present State	Input XY =				Output
	00	01	11	10	(Z)
A (00)	00	00	01	10	0
B (01)	--	11	01	11	1
D (11)	--	10	--	10	1
C (10)	00	10	10	10	0

Kmaps for 1st bit (B1) and 2nd bit (B2) are as follows (ensure that there are no static hazards):

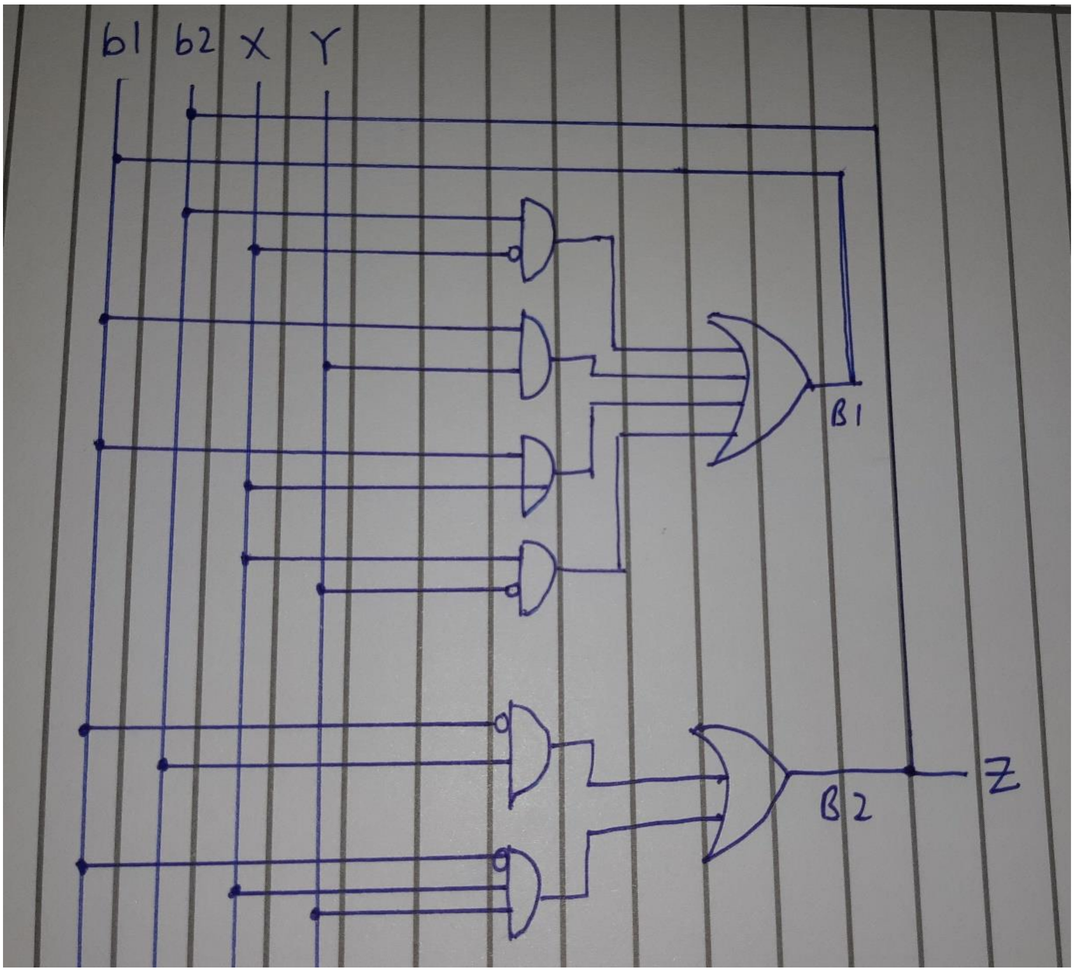


B1		XY			
		00	01	11	10
b1 b2	00	0	0	0	1
	01	X	1	0	1
	11	X	1	X	1
	10	0	1	1	1



B2		XY			
		00	01	11	10
b1 b2	00	0	0	1	0
	01	X	1	1	1
	11	X	0	X	0
	10	0	0	0	0

Note that the output Z is the same as the 2nd bit b2.
Write out equations:
B1 = b2X’ + b1Y + b1X + XY’
B2 = b1’b2 + b1’XY



This is all done without an answer key so do clarify via email if you find any errors.

All the best for your exams 😊