Name: _____ **Student #:** _____

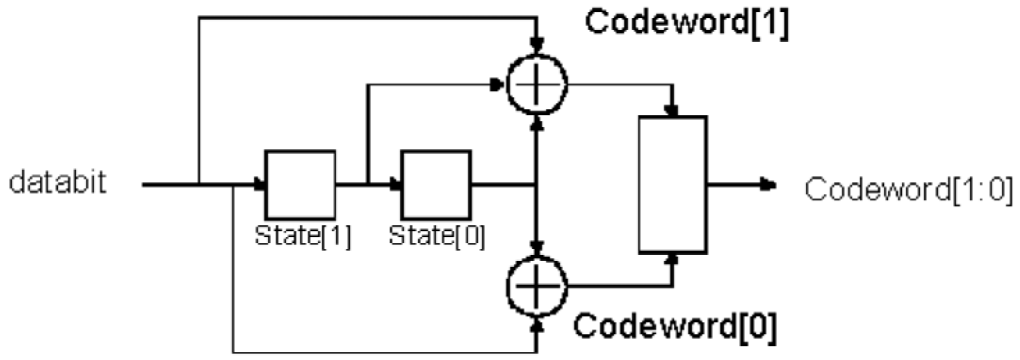
ID: A

CSE 2021 - Quiz 3 - Fall 2011

Problem

1. Convolutional encoding is commonly used on cell phones to ensure data is received with few errors using the low transmit power typically available on a cell phone. A rate 1/2 convolutional encoder is to be included in the core of a cell phone processor and requires a Verilog module to be written.

   The module will use a finite state machine approach and will accept two inputs - databit and state[1:0] and produce two outputs codeword[1:0] and nextstate[1:0]. The encoding process is diagrammed below, which shows the databit and state bits. The process outputs 2 bits for every 1 input bit. The nextstate[1:0] is defined by [databit : state 1]. In other words, the bits are shifted to the right to continue the process. The codeword bits are formed by XOR'ing the indicated bits together.



   (a) What type of finite state machine is this and why?

   (b) Fill out the following table for nextstate and codeword outputs

databit	state[1:0]	nextstate[1:0]	codeword[1:0]
0	00		
0	01		
0	10		
0	11		
1	00		
1	01		
1	10		
1	11		

   (c) Write the conv module based on the above description.

Name: _____

```
module conv(state, databit, nextstate, codeword);









endmodule
```

(d) A test bench Verilog module has been written (see below). What is the output from this (show on next page)?

```
module testbench;

reg [1:0] state;
reg [3:0] test_data;
reg test_bit;
wire [1:0] codeword, nextstate;
integer i;

conv uut_conv(state, test_bit, nextstate, codeword);

initial
begin

 test_data = 4'b1001;
 state = 2'b00;

 for (i = 0; i < 4; i = i+1)
  begin
   #200
   test_bit = test_data[i];
   #1 $display($time, " databit = %b, state = %b, nextstate = %b codeword
= %b",test_bit, state, nextstate, codeword);

   state = nextstate;
   end
  $finish;
  end
endmodule
```

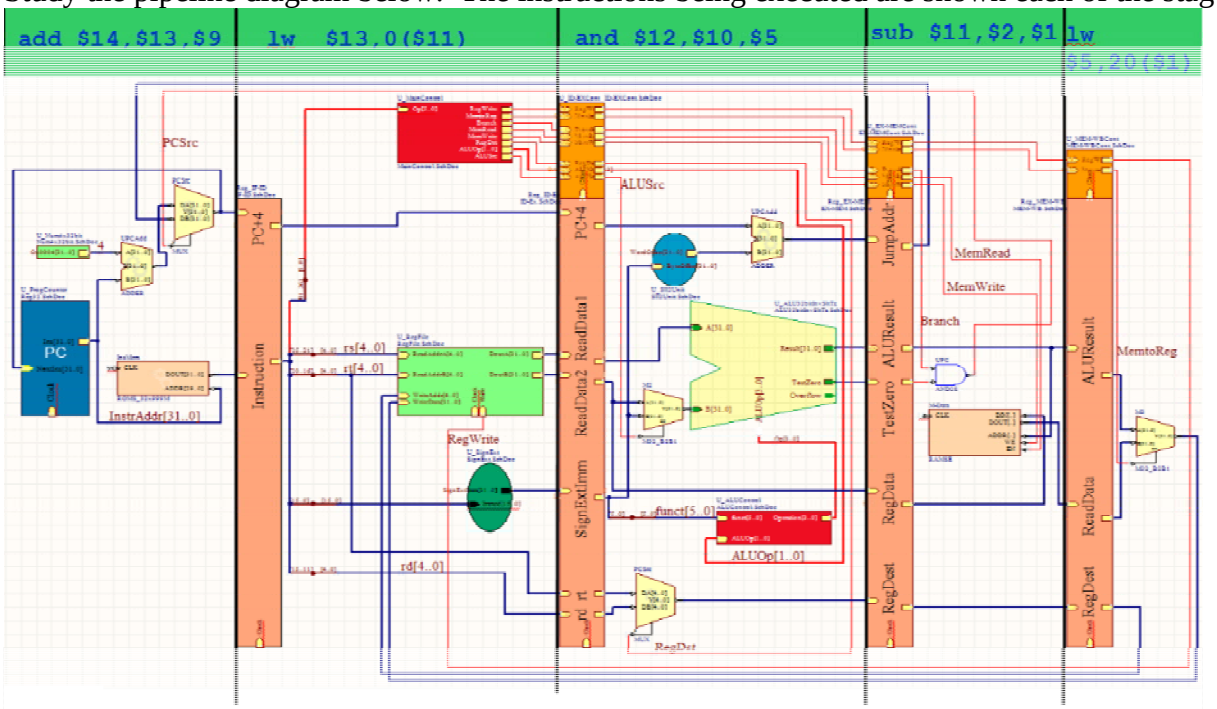
Name: _____

Output:

.

Name: _____

2. Study the pipeline diagram below. The instructions being executed are shown each of the stages.



(a) Provide the contents of each of the interstage registers. Wherever possible provide a numerical value. If this is not known specify a M[addr] or R[addr] for either a memory or register file location respectively.

IF/ID		ID/EX	
"PC+4"		WB(1)	
Instruction		MEM(1)	
		EX(1)	
		"PC+4"	
		ReadData1	
		ReadData2	
		SignExtImm	
		rt	
		rd	

EX/MEM		MEM/WB	
MEM(1)		WB(1)	
WB(1)		ALUResult	
JumpAddr		ReadData	
ALUResult		RegDest	
TestZero			
RegData			
RegDest			

(1) Format for control bits: **EX**: ALUSrc, ALUOp[1:0], RegDst, **Mem**: Branch, MemRead, MemWrite, **WB**: MemtoReg, RegWrite

Name: _____

(b) Identify any data hazards by filling out the following table (an example is shown for illustration only):

Hazards				
Instruction 1	Instruction 2	Ins 1 REG.item	Ins 2 REG.item	Stall or Forward?
add $3, $2, $1	*sub $5, $4, $3*	*EX/MEM.Rd*	*ID/EX.Rt*	*Forward*

Reference Table - 1

Instruction opcode	ALUOp	Instruction operation	Function code	Desired ALU action	ALU control input
LW	00	load word	XXXXXX	add	0010
SW	00	store word	XXXXXX	add	0010
Branch equal	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
R-type	10	subtract	100010	subtract	0110
R-type	10	AND	100100	AND	0000
R-type	10	OR	100101	OR	0001
R-type	10	set on less than	101010	set on less than	0111

Reference Table - 2

Instruction	Execution/address calculation stage control lines				Memory access stage control lines			Write-back stage control lines	
	RegDst	ALUOp1	ALUOp0	ALUSrc	Branch	Mem-Read	Mem-Write	Reg-Write	Memto-Reg
R-format	1	1	0	0	0	0	0	1	0
lw	0	0	0	1	0	1	0	1	1
sw	X	0	0	1	0	0	1	0	X
beq	X	0	1	0	1	0	0	0	X

CSE 2021 - Quiz 3 - Fall 2011
Answer Section

PROBLEM

1. ANS:
(a) 2 marks Meely machine - next state depends on both the input and the current state.

(b)8 marks

databit	state[1:0]	nextstate[1:0]	codeword[1:0]
0	00	00	00
0	01	00	11
0	10	01	10
0	11	01	01
1	00	10	11
1	01	10	00
1	10	11	01
1	11	11	10

(c)5 marks

```
module conv(state, databit, nextstate, codeword);
     input [1:0] state;
     input databit;
     output[1:0] nextstate, codeword;


     assign nextstate[1] = databit;
     assign nextstate[0] = state[1];

     assign codeword[1] = databit^state[1]^state[0];
     assign codeword[0] = databit^state[0];

endmodule
```

(d)5 marks

```
C:\Verilog>
C:\Verilog>iverilog testb-conv.v conv.v

C:\Verilog>vvp a.out
                 201 databit = 1, state = 00, nextstate = 10 codeword = 11
                 402 databit = 0, state = 10, nextstate = 01 codeword = 10
                 603 databit = 0, state = 01, nextstate = 00 codeword = 11
                 804 databit = 1, state = 00, nextstate = 10 codeword = 11
```

PTS: 1

2. ANS:
(a)

IF/ID 4 marks		**ID/EX 9 marks**		**EX/MEM 7 marks**		**MEM/WB 4 marks**	
"PC+4"	PC-4	WB(1)	1 1	WB(1)	0 1	WB(1)	1 1
Instruction *	M[PC-4] 0x8d6d0000	MEM(1)	0 1 0	MEM(1)	0 0 0	ALUResult	20+R[$1]
		EX(1)	1 00 1	JumpAddr	PC-12	ReadData	M[20+R[$1]]
		"PC+4"	PC-8	ALUResult	R[$2]-R[$1]	RegDest	00101
		ReadData1	R[$11]	TestZero	R[$2]=R[$1] ? 1:0		
		ReadData2	R[$13]	RegData	R[$1]		
		SignExtImm	0x00006024	RegDest	01011		
		rt	01101				
		rd	00000				

* - 1 mark for M[..] answer, 2 marks for 0x.... answer

(b)6 marks

Hazards				
Instruction 1	Instruction 2	Ins 1 REG.item	Ins 2 REG.item	Stall or Forward?
add $3, $2, $1	*sub $5, $4, $3*	*EX/MEM.Rd*	*ID/EX.Rt*	*Forward*
lw $5, 20($1)	and $12, $10, $5	MEM/WB.Rd	ID/EX.Rt	Forward
lw$13, 0($11)	add $14, $13, $9	ID/EX.Rt	IF/ID.Rt	Stall

PTS: 1