Name: ________________________ Roll number: ____________

CS230 DLDCA Mid-Sem, Tue 17 Sep 2024, 13.30-15.30pm, Max. Marks: 30

General instructions

- Write only in the space provided. Answer briefly but crisply (not lengthily or loosely).
- You are allowed to refer to your own hand-written notes only.
- Write neatly and clearly. Up to **+2 HP** for neat handwriting, neat/crisp answers.
- Answers generally have to be (briefly) explained. State any necessary assumptions.

1. **[1 x 2 = 2 marks] Short answer questions:**

   a) Mention any two advantages of dynamic linking.

   Smaller exe file (better use of memory)
   Libraries can be updated for bugfix or performance improvement, without changing existing programs

   b) In a MIPS32 program, a beq instruction is at location 80,000 (decimal value of byte address). In its machine code, the immediate value is –200 (decimal value again). What is the branch target address (which is executed when branch is taken)?

   Imm value is taken as word offset from PC+4
   So branch target is 80000 + 4 + (4*-200) = 79204 (decimal value)

2. **[2 marks]** Prove that f(x,y,z) = xy+yz+zx is self-dual using algebraic manipulations (i.e., Boolean Algebra rules and theorems) only. Show all your steps.

   See at the end

3. **[1 x 4 = 4 marks]** MIPS32 program P.exe is generated from assembly files p1.s & p2.s, it has no other external library. MIPS32 program Q.exe is generated from 2 assembly files q1.s and q2.s as well as a statically linked external library lib1.o. There is no dynamically linked library. Answer the following.

   a) In P.exe, suppose registers $at ($1) and $t0 ($8) are exchanged in all instructions in which they appear, in the machine code. Would P.exe continue to work? Justify.

      MIPS32 machine hardware has no special meaning for $at vs $t0. So P.exe will continue to work. (This has nothing to do with caller-callee saving conventions).

   b) Answer the same as above with justification, for Q.exe.

      Same answer and reason as for P.exe

   c) In P.exe, suppose registers $s0 ($16) and $ra ($31) are exchanged in all instructions in which they appear, in the machine code. Would P.exe continue to work? Justify.

      MIPS32 machine hardware does have a special meaning for $ra. The jal instruction saves the return address PC+4 onto $ra. So swapping will NOT work. (This also has nothing to do with caller-callee saving conventions).

   d) Answer the same as above with justification, for Q.exe.

      Same answer and reason as for P.exe

4. **[2 marks] Design decision using computer performance:** Suppose that the MIPS32 designers are considering the inclusion of an instruction called add3, which adds 3 registers instead of two added by the current add instruction. Using simulation on a benchmark program, they found that when add3 is used, 5% of the executed instructions use add3. The designers also determined that, with inclusion of add3, the clock cycle length of a single cycle implementation increases from 500ps to 550ps. (1 pico-second = 10^{-12} seconds). Is the inclusion of add3 beneficial?

   With add3, execution time T1 = N x 1 x 550 ps (CPI=1, N=executed instructions)
   Without add3, each add3 will be replaced with two add instructions, so number of executed instructions will be Nx1.05
   So, without add3, execution time T2 = Nx1.05 x 1 x 500
   Clearly, T1 is larger than T2, so inclusion of add3 is NOT beneficial

Name: ________________________ Roll number: ____________

5. **[3+2=5 marks]** A new flip-flip, called MN flip-flop, is constructed from a JK flip-flop as follows:

   $J = M; K = N \oplus M$

   a) Derive the characteristic table and excitation table for this new flip-flop

      See at the end

   b) Construct a D flip-flop from this new flip-flop

      See at the end

6. **[1+2+2=5 marks]** Construct a counter for the following sequence using T flip-flops: 000 → 100 → 111 → 010 → 011

   a) Draw the state-table.

   See at the end

   b) Construct the present-state next-state table with T-flip-flop as the memory element.

   See at the end

   c) Derive the inputs of the T flip-flops.
   See at the end

7. **[2+4+2+2=10 marks]** Consider the single cycle implementation for the ISA subset {add, sub, and, or, lw, sw, beq} (as in the videos). We are now going to add support for a new instruction (this is not really a MIPS instruction though) called sw0_inc. This instruction is the same as sw except for two differences: it always uses offset as 0, and it has the side effect of incrementing the value of the base register by a given 16-bit immediate operand.

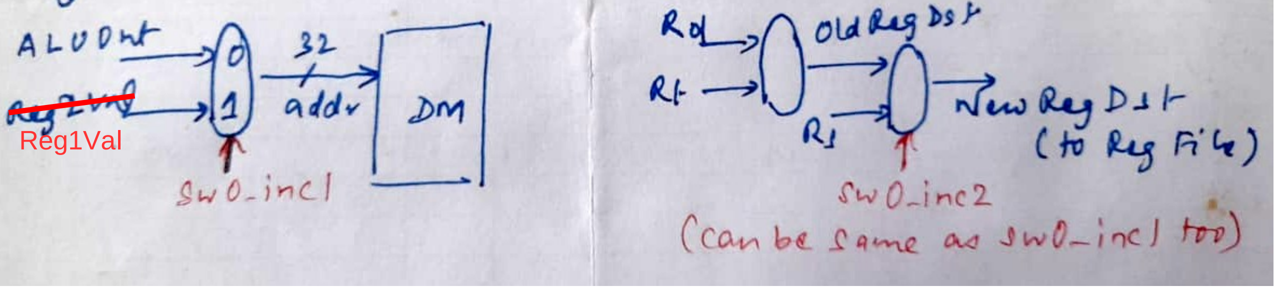
   a) What instruction format can sw0_inc use? Draw it, along with brief justification.

      It can use exactly the same format as sw (I format)
      Rs is base register, Rt is register to be saved to memory
      Note that now Rs will have to be written back while executing the instruction

Opcode 6	Rs 5	Rt 5	Immediate 16

   b) Draw *only* the datapath changes from the original datapath for the ISA subset. Your changes should be *minimal*, adding only muxes as necessary: *avoid* additional ALU/adder or changes to the register file or memory components (it is of course ok to change the possible *inputs* to these components).

Reg reading - same
ALU operation - same - base reg + Imm (SgnExt)
Mem addr - take from Reg1Val [NEW]
Mem value to write - same (Reg2Val)
Reg to write - Rs [NEW]
Reg value to write - ALUOut (already in datapath)



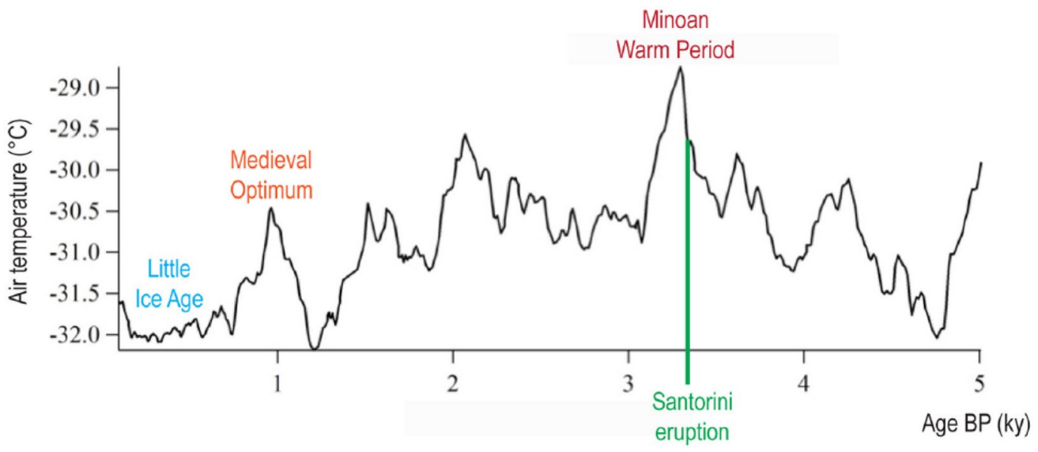
c) Draw *only* the control signal changes from the original ISA subset, i.e. show additional row(s)/column(s) in the truth table.

New row in truth table for sw0_inc instruction:
RegWr=1, RegDst=x, Mem2Reg=0 (choose ALUOut)
sw0_inc1 = sw0_inc2 = 1
ALUSrc=1 (SgnExtImm16), ALUOp = add
MemRd=0,MemWr=1, Branch=0
New cols in truth table for sw0_inc1/sw0_inc1
sw0_inc1=0 for lw and sw, sw0_inc1=1 for sw0_inc instruction, sw0_inc1=x for other instructions
sw0_inc2=0 for all reg-reg instructions, and for lw; sw0_inc2=1 for sw0_inc instruction, sw0_inc2=x for sw, beq

d) Can a lw0_inc instruction be similarly implemented without any change to the main components? Why or why not?

No. Implementing lw0_inc will require two write ports for the register file component.

8. **[Optional, up to 10HP]** The following graph is the estimated Greenland temperature, from the GISP2 (Greenland Ice Sheet Project 2) data, from a 2018 publication. The x-axis is in units of kilo-years Before Present (BP), where BP=0 is taken as 1950AD. As additional data (not in graph), the global average warming since 1950 is about 0.75 °C.



a) How much warmer/colder was Greenland during the Medieval Warm Period (1000AD) compared to today? _(1-0.75) = 0.25 °C warmer than today approximately___________

b) How much warmer/colder was Greenland during the Roman Warm Period (0 AD) compared to today? _(2-0.75) = 1.25 °C warmer than today approximately___________

c) How much warmer/colder was Greenland during the Minoan Warm Period (1300 BC) compared to today? _(2.75-0.75) = 2 °C warmer than today approximately___________

d) Connect the theme of the various house-point questions so far with a point made in the first lecture – Think *independently* using data/numbers, to avoid brainwashing

1.

$f_2(x, y, z)$

$:= (x+y)(y+z)(z+x)$

$:= (y+x)(y+z)(z+x)$

$= (y+xz)(x+z)$

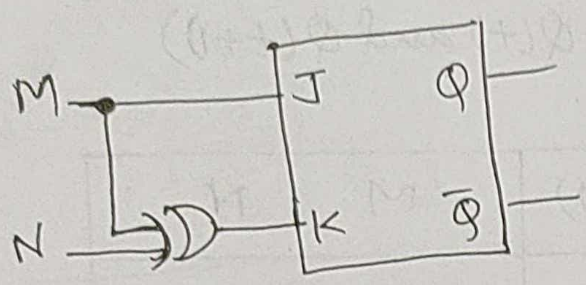
$= xy + xz + xz + yz$

$= xy + yz + zx$

$= f(x, y, z)$

Q.2

$\longrightarrow$ MN flip-flop : $J = M$; $K = N \oplus M$



State-transition table

M	N	J	K	Q(t+1) [from JKFF]
0	0	0	0	Q(t)
0	1	0	1	0
1	0	1	1	Q'(t)
1	1	1	0	1

Excitation table

Q(t)	Q(t+1)	M	N
0	0	0	X
0	1	1	X
1	0	0 —OR— 1	1 —OR— 0
1	1	0 —OR— 1	0 —OR— 1

With the help of excitation table, we can model D flip-flop's state transition table in the form of M and N. (by observing $Q(t)$ and $Q(t+1)$)

D	$Q(t)$	$Q(t+1)$	M N
0	0	0	0 X
0	1	0	0 1 —OR— 1 0
1	0	1	1 X
1	1	1	0 0 —OR— 1 1

Using K-map, we find 4 solutions for each pair of MN combinations

$\boxed{1}$

D	$Q(t)$	M	N
0	1	0	1
1	1	0	0

for M:

D \ $Q(t)$	0	1
0	0	0
1	1	0

$$\boxed{M = D \cdot \overline{Q(t)}}$$

for N:

D \ $Q(t)$	0	1
0	X	1
1	X	0

$$\boxed{N = \overline{D}}$$

$\boxed{2}$

D	$Q(t)$	M	N
0	1	0	1
1	1	1	1

for M:

D \ $Q(t)$	0	1
0	0	0
1	1	1

$$\boxed{M = D}$$

for N:

D \ $Q(t)$	0	1
0	X	1
1	X	1

$$\boxed{N = 1}$$

3

D	Q(t)	M	N
0	1	1	0
1	1	0	0

for M:

D \ Q(t)	0	1
0	0	1
1	1	0

for N:

D \ Q(t)	0	1
0	0	0
1	1	0

$$\boxed{M = D\overline{Q(t)} + \bar{D}Q(t)}$$

$$\boxed{M = D \oplus Q(t)}$$

$$\boxed{N = D\overline{Q(t)}}$$

4

D	Q(t)	M	N
0	1	1	0
1	1	1	1

for M:

D \ Q(t)	0	1
0	0	1
1	1	1

for N:

D \ Q(t)	0	1
0	0	0
1	1	1

$$\boxed{M = D + Q(t)}$$

$$\boxed{N = D}$$

Q.3]

$\longrightarrow$ The sequence is: $000 \rightarrow 010 \rightarrow 111 \rightarrow 100 \rightarrow 011$

The state table is as follows:

S_0	000
S_1	010
S_2	111
S_3	100
S_4	011

The next-state map is as follows.

Lets say x is an input variable.

$Q(t)$	$Q(t+1)$	
	$X=0$	$X=1$
S_0	S_0	S_1
S_1	S_1	S_2
S_2	S_2	S_3
S_3	S_3	S_4
S_4	S_4	S_0

$$\begin{tikzcd}
S_0 \arrow[loop above, "X=0"] \arrow[r, "X=1"] & S_1 \arrow[loop above, "X=0"] \arrow[r, "X=1"] & S_2 \arrow[loop above, "X=0"] \arrow[r, "X=1"] & S_3 \arrow[loop above, "X=0"] \arrow[r, "X=1"] & S_4 \arrow[loop above, "X=0"]
\end{tikzcd}$$

Now, the truth table of T flip flop is:

T	Q(t)	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

To derive the inputs of T flip-flops, we look at the next-state map and truth table of T flipflop.

Q(t)			Q(t+1)						T F/F inputs					
			x = 0			x = 1			x = 0			x = 1		
y_3	y_2	y_1	Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	T_3	T_2	T_1	T_3	T_2	T_1
0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
0	1	0	0	1	0	1	1	1	0	0	0	1	0	1
1	1	1	1	1	1	1	0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	1	1	0	0	0	1	1	1
0	1	1	0	1	1	0	0	0	0	0	0	0	1	1

Using K-map, we find:

for T_3 :

$y_3 y_2$ \ $y_1 x$	00	01	11	10
00	0	0	X	X
01	0	1	0	0
11	X	X	0	0
10	0	1	X	X

$$T_3 = y_2 \bar{y}_1 x + y_3 \bar{y}_2 x$$

$$\boxed{T_3 = x(y_2 \bar{y}_1 + y_3 \bar{y}_2)}$$

for T_2:

$y_3y_2 \backslash y_1x$	00	01	11	10
00	0	1	X	X
01	0	0	1	0
11	X	X	1	0
10	0	1	X	X

$$\boxed{T_2 = y_1 x + \bar{y}_2 x}$$

for T_1:

$y_3y_2 \backslash y_1x$	00	01	11	10
00	0	0	X	X
01	0	1	1	0
11	X	X	1	0
10	0	1	X	X

$$\boxed{T_1 = y_2 x + y_3 x}$$