Université d'Ottawa
Faculté de génie
École de science informatique
et de génie électrique

uOttawa
L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering
School of Electrical Engineering
and Computer Science

CEG2136: Computer Architecture I / CEG2536: Architecture des Ordinateurs I

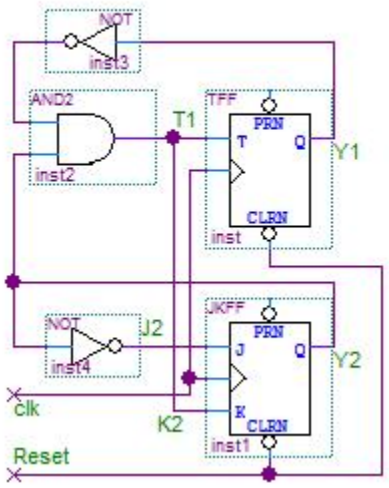
MIDTERM EXAMINATION

Professors: Voicu Groza, Aziz Abdesselam and Rami Abielmona

SOLUTIONS

Question 1 (20 points)

A sequential circuit is presented below:



a) Find the equations of the inputs J2, K2, and T1 of the two flip-flops

T1 = K2 = Y2 Y1'

J2 = Y2'

3

2

b) If the system's initial state is Y2 = Y1 = 0, derive the transition table and then draw the state diagram.

Use the FF's Characteristic Tables to derive the transition table of the sequential circuit :

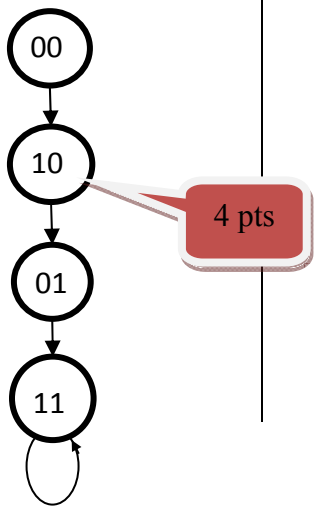
J	K	$Q(n+1)$
0	0	$Q(n)$
0	1	0
1	0	1
1	1	$Q(n)'$

T	$Q(n)$	$Q(n+1)$
0	0	0
0	1	1
1	0	1
1	1	0

transition table						
$Y2\ (n)$	$Y1(n)$	J_2	K_2	T1	$Y2(n+1)$	$Y1(n+1)$
0	0	1	0	0	1	0
0	1	1	0	0	1	1
1	0	0	1	1	0	1
1	1	0	0	0	1	1
1	1	1	1	1	4	2

11pts

state diagram



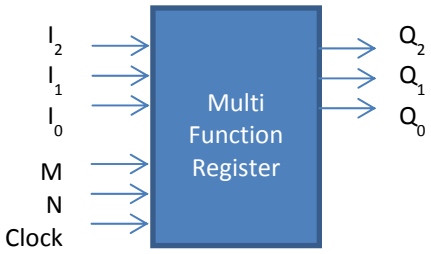
4 pts

Question 2 (20 points)

Design a 3-bit register whose function is described in the following table, where M and N are two control bits.

Using the proper digital components (encoders, decoders, multiplexers, etc.), logic gates, and D flip-flops, draw a detailed diagram of the logic circuit of the register.

Clock	M	N	Operation
↑	0	0	No change
↑	0	1	Increment by 1
↑	1	0	Decrement by 1
↑	1	1	Loading external inputs, say I_2 I_1 I_0



Use excitation table or equation

$Q(n)$	$Q(n+1)$	D
0	0	0
0	1	1
1	0	0
1	1	1

M N = 00 => no change

<=> $Q_i\ (n+1) = Q_i\ (n)$; i= {0,1,2}

=> $D_2 = Q_2\ (n)$; $D_1 = Q_0\ (n)$; $D_0 = Q_0\ (n)$

M N = 11 =>Load $I_2I_1I_0$ to $Q_2Q_1Q_0$:

<=> $Q_i\ (n+1) = I_i\ (n)$; i= {0,1,2}

$Q_i(n)$	I_i	$Q_i(n+1)$	D_i
0	0	0	0
0	1	1	1
1	0	0	0
1	1	1	1

=> $D_i = I_i\ (n)$; i= {0,1,2}

MN= 01 =>Increment by 1

Present State			Next State			D FF inputs		
$Q_2(n)$	$Q_1(n)$	$Q_0(n)$	$Q_2(n+1)$	$Q_1(n+1)$	$Q_0(n+1)$	D_2	D_1	D_0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	1	1	0	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0

D_2

Q_2 \ Q_1Q_0	00	01	11	10
0	0	0	1	0
1	**1**	**1**	0	1

$D_2 = \mathbf{Q_2\ Q_1'} + Q_2\ Q_0' + Q_2'\ Q_1\ Q_0$

D_1

Q_2 \ Q_1Q_0	00	01	11	10
0	0	1	0	1
1	0	1	0	1

$D_1 = Q_1'\ Q_0 + Q_1\ Q_0' = Q_1 \oplus Q_0$

D_0

Q_2 \ Q_1Q_0	00	01	11	10
0	1	0	0	1
1	1	0	0	1

$D_0 = Q_0'$

MN= 10 =>Decrement by 1

Present State			Next State			D FF inputs		
$Q_2(n)$	$Q_1(n)$	$Q_0(n)$	$Q_2(n+1)$	$Q_1(n+1)$	$Q_0(n+1)$	D_2	D_1	D_0
0	0	0	1	1	1	1	1	1
0	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	1
0	1	1	0	1	0	0	1	0
1	0	0	0	1	1	0	1	1
1	0	1	1	0	0	1	0	0
1	1	0	1	0	1	1	0	1
1	1	1	1	1	0	1	1	0

D_2

Q_2 \ $Q_1 Q_0$	00	01	11	10
0	1	0	0	0
1	0	1	1	1

D_1

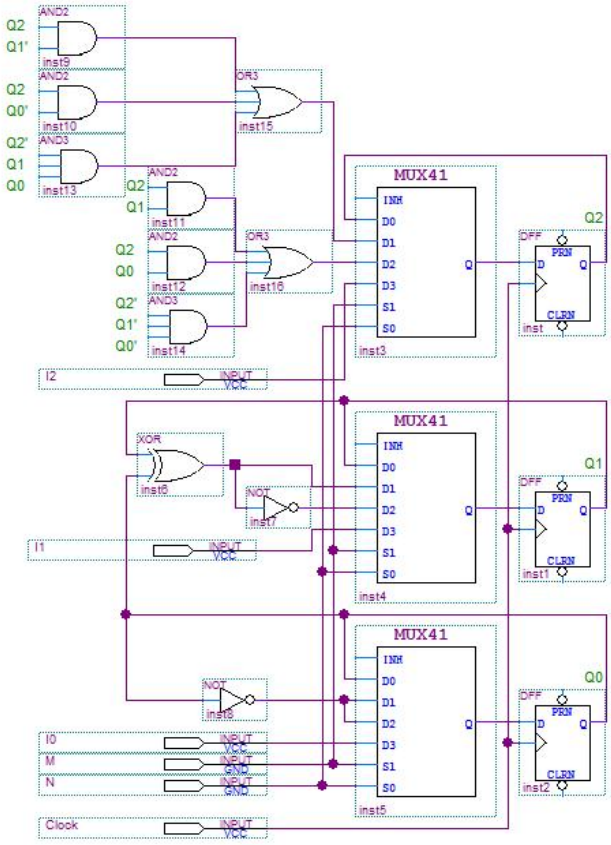
Q_2 \ $Q_1 Q_0$	00	01	11	10
0	1	0	1	0
1	1	0	1	0

D_0

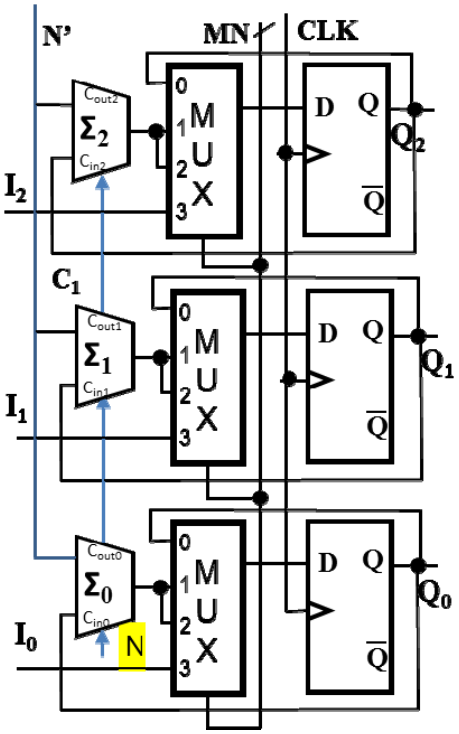
Q_2 \ $Q_1 Q_0$	00	01	11	10
0	1	0	0	1
1	1	0	0	1

$D_2 = \boldsymbol{Q_2\, Q_1} + Q_2\, Q_0 + Q_2\text{'}\, Q_1\text{'}\, Q_0\text{'}$ $D_1 = Q_1\text{'}\, Q_0\text{'} + Q_1\, Q_0 = (Q_1 \oplus Q_0)\text{'}$ $D_0 = Q_0\text{'}$

Logic Diagram of the multi-function register:



OR use adders for incrementation and decrementation:



Question 3 (25 points)

Two's complement representation is used for signed numbers in this question.

The 8-bit registers VR, XR, YR and ZR initially have the following values:

VR = 1011 1100; XR = 0100 0010;

YR = 1011 1001; ZR = 1110 1010

1. An 8-bit adder is employed to perform the following operations:

   YR = VR + XR

   ZR = VR – XR

   Determine the 8-bit values in each register after the execution of these micro-operations.

Do not take points if verification is missing here.

YR = VR + XR

Carry:	C_8	C_7	C_6	C_5	C_4	C_3	C_2	C_1	C_0	
	0	0	0	0	0	0	0	0		Verification in decimal:
VR		**1**	0	1	1	1	1	0	0	= -68
XR		**0**	1	0	0	0	0	1	0	= (64+2)= 66
YR		**1**	1	1	1	1	1	1	0	-2

4

ZR = VR - XR = VR + (- XR) = VR + 2's compl (XR)= 1011 1100+ 2's compl (0100 0010)

-XR=2's compl (XR)= 2's compl (0100 0010) = 1011 1110; => **VR = -68**

Carry:	C_8	C_7	C_6	C_5	C_4	C_3	C_2	C_1	C_0	
	1	0	1	1	1	1	0	0		Verification in decimal:
VR		**1**	0	1	1	1	1	0	0	= -68
-XR		**1**	0	1	1	1	1	1	0	= -66
ZR		**0**	1	1	1	1	0	1	0	BAD!? : 122

5

1

VR = 1011 1100; XR = 0100 0010; YR = 1111 1110; ZR = 0111 1010

2. Convert to decimal all four binary numbers stored in these registers after the execution of the micro-operations.

3

VR = 1011 1100 = -v => v=-VR=2's compl (VR)= 2's compl (1011 1100) = 0100 0100 = 64+4 = 68; => **VR = -68**
XR = 0100 0010 = 64 + 2 = **66**;
YR = 1111 1110 = -y => y=-YR=2's compl (YR)= 2's compl (1111 1110) = 0000 0010 = 2 = 71; => **YR = -2**
ZR = 0111 1010 = 64+32+16+10 = 106+16 = **122**

1

1

3

Represent in BCD the decimal number equivalent to the binary number stored in register XR.

XR = 66=0110 0110

2

3. Is there any overflow? Justify your answer. How would a computer detect overflows in these operations? Show your work in detail.

2

YR = VR + XR -> no OFL : operands have different signs or $C_8 = C_7 = 0$

+ correct result (-68+66=2)

3

ZR = VR - XR -> OFL : operands have same signs (1) but different from result or $C_8 \neq C_7$

+ wrong result (-68-66=+122?)

Question 4 (35 points)

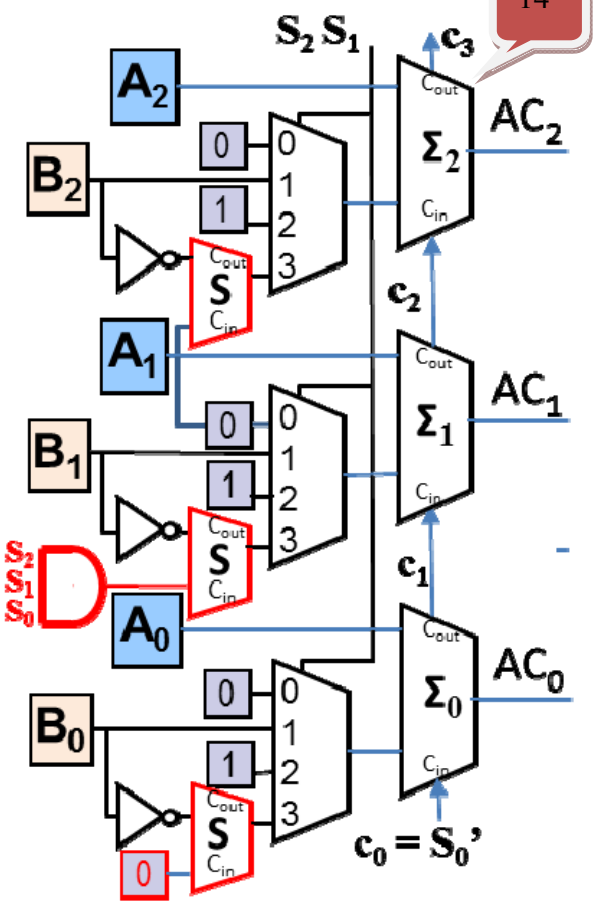
The function of a 3-bit ALU is described by the following table, where A = $A_2A_1A_0$ and B = $B_2B_1B_0$ are the ALU's two inputs and C = $C_2C_1C_0$ is its output.

Draw the logic circuit of this ALU using multiplexers, full adders and gates AND, OR, NOT, XOR, as necessary.

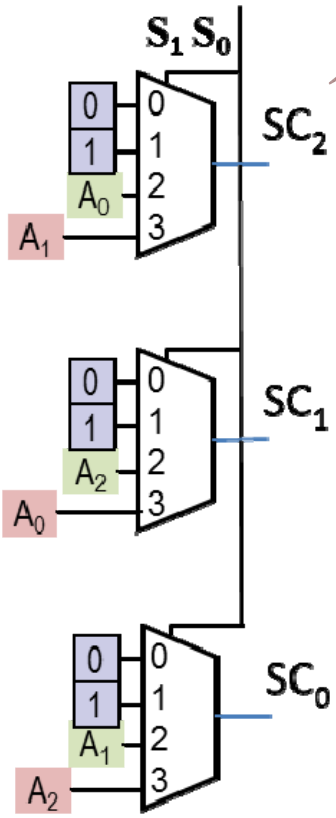
Points		S_3		S_2	S_1	S_0	Output	Description	C ←			
14	1.5	0	0	0	0	0	C←A+1	Increment A	$A_2A_1A_0$ +	000	+ 001	
	1.5	0	1	0	0	1	C←A	Transfer A	$A_2A_1A_0$ +	000		
	1.5	0	2	0	1	0	C←A+B+1	Add with carry	$A_2A_1A_0$ +	$B_2B_1B_0$	+ 001	
	1.5	0	3	0	1	1	C←A+B	Addition	$A_2A_1A_0$ +	$B_2B_1B_0$		
	2	0	4	1	0	0	C←A	Transfer A	$A_2A_1A_0$ +	111	+ 001	
	2	0	5	1	0	1	C←A-1	Decrement A = A + 2's compl (001)	$A_2A_1A_0$ +	111		
	2	0	6	1	1	0	C←A-B	Subtraction	$A_2A_1A_0$ +	B_2' B_1' B_0'	+ 001	+ 000
	2	0	7	1	1	1	C←A-B+1	Subtraction with borrow	$A_2A_1A_0$ +	B_2' B_1' B_0'		+ 010
8	2	1	0	0	0	0	C←A∧B'	Selective clear bits by "mask" B	A_i ∧	B_i'		i = 0,1,2
	2	1	1	0	0	1	C←A⊕B	Comparison	A_i ⊕	B_i		i = 0,1,2
	2	1	2	0	1	0	C←A∧B	A AND B	A_i ∧	B_i		i = 0,1,2
	2	1	3	0	1	1	C←A∨B	Selective set bits by "mask" B	A_i ∨	B_i'		i = 0,1,2
8	2	1	4	1	0	0	C←"0000"	Reset C	C_i ←	0		i = 0,1,2
	2	1	5	1	0	1	C←"1111"	Set C	C_i ←	1		i = 0,1,2
	2	1	6	1	1	0	C←cirA	Circular shift right	C_i ←	A_{i+1};	C_2 ← A_0	i = 0,1
	2	1	7	1	1	1	C←cilA	Circular shift left	C_i ←	A_{i-1};	C_0 ← A_2	i = 1,2

Final block diagram 5

Arithmetic Circuit — 14

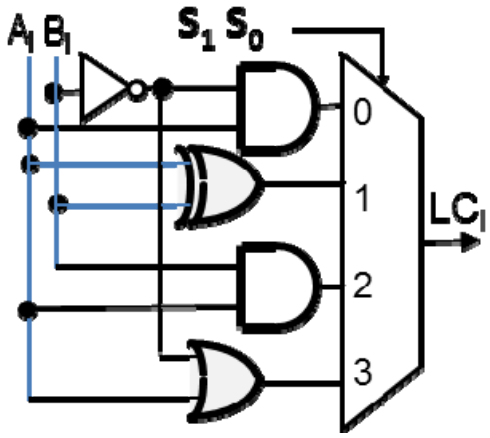


Shift Circuit — 8



Logic Circuit — 8

$i = 0, 1, 2$



ALU — 5