Name: Email: ID#

UNIVERSITY OF CALIFORNIA
Department of Electrical Engineering and Computer Sciences
EECS150 Fall 2001

Prof. Subramanian

Final Examination

1) You are to design a sequential circuit with two JK FFs A and B, and two inputs E and X. If E=0, the circuit remains in the same state, regardless of the value of X. When E=1 and X=1, the circuit clocks in the sequence 00→01→10→11→00… When E=1 and X=0, the circuit goes through the state transitions 00→11→10→01→00…

a) Complete the state transition table for the aforementioned circuit

Current State		Inputs		Next State		FF Inputs			
A	B	E	X	A	B	J_A	K_A	J_B	K_B
0	0	0	0	0	0	0	X	0	X
0	0	0	1	0	0	0	X	0	X
0	0	1	0	1	1	1	X	1	X
0	0	1	1	0	1	0	X	1	X
0	1	0	0	0	1	0	X	X	0
0	1	0	1	0	1	0	X	X	0
0	1	1	0	0	0	0	X	X	1
0	1	1	1	1	0	1	X	X	1
1	0	0	0	1	0	X	0	0	X
1	0	0	1	1	0	X	0	0	X
1	0	1	0	0	1	X	1	1	X
1	0	1	1	1	1	X	0	1	X
1	1	0	0	1	1	X	0	X	0
1	1	0	1	1	1	X	0	X	0
1	1	1	0	1	0	X	0	X	1
1	1	1	1	0	0	X	1	X	1

b) Determine the FF input functions. Simplify as appropriate.

i) $J_A = E(BX + B'X')$

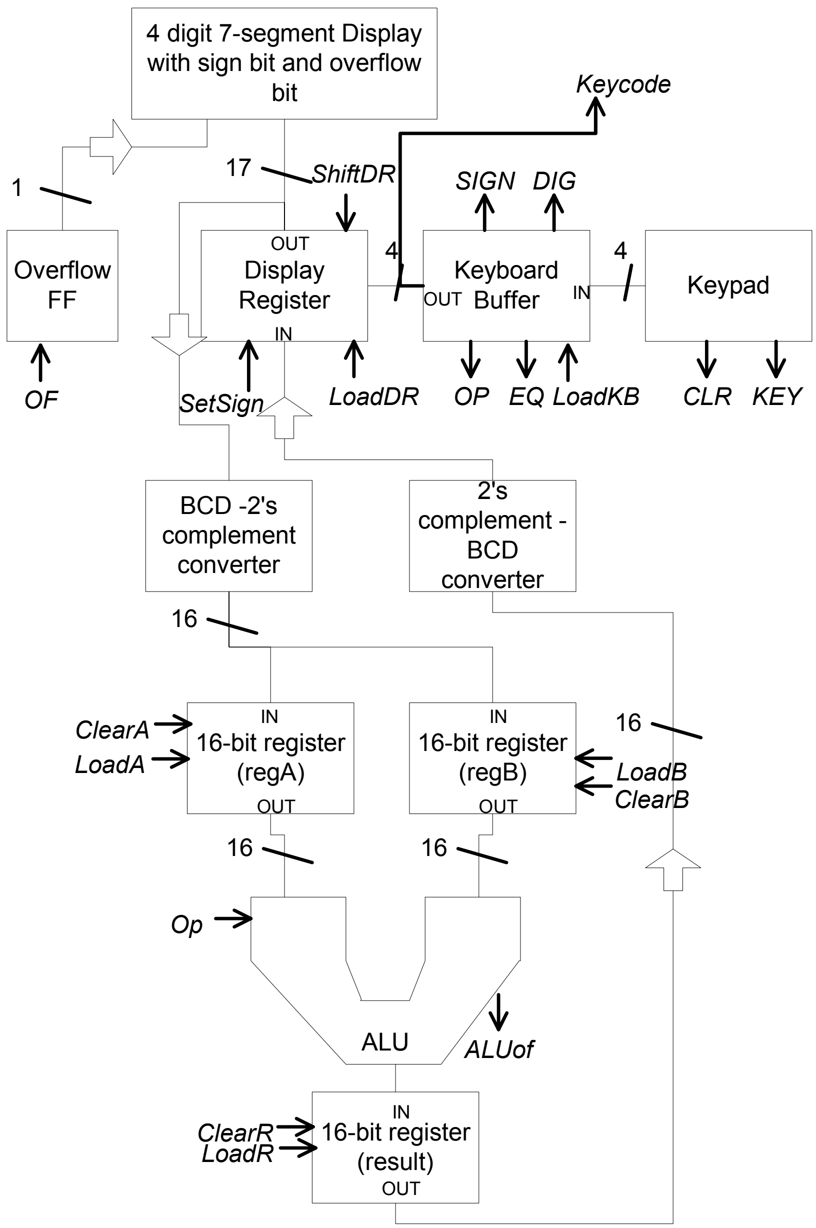
ii) $K_A = E(BX + B'X')$

iii) $J_B = E$

iv) $K_B = E$

Name: Email: ID#

2) In this problem, you are going to implement a basic calculator that performs simple integer arithmetic. The calculator datapath is shown below:



Keypad

- 0-9 in BCD
- + = 1010
- - = 1011
- × = 1100
- ÷ = 1101
- ± = 1110
- = = 1111
- CLR key produces direct output
- KEY = a key has been pressed

Display

- 4× BCD-to-7segment digits
- 1-bit sign (1 = negative)
- 1-bit overflow (1=overflow)

Keyboard Buffer

- LoadKB – instructs buffer to load value from keyboard
- EQ – keypress: =
- OP – keypress: +/-/×/÷
- SIGN - keypress: ±
- DIG – keypress: 0-9
- Keycode – keycode to controller

Overflow Buffer

- OF – sets overflow display

Registers A, B and R

- Clear – Resets register contents
- Load – Loads value

Display Register

17 bit register (4× BCD + 1 sign bit)

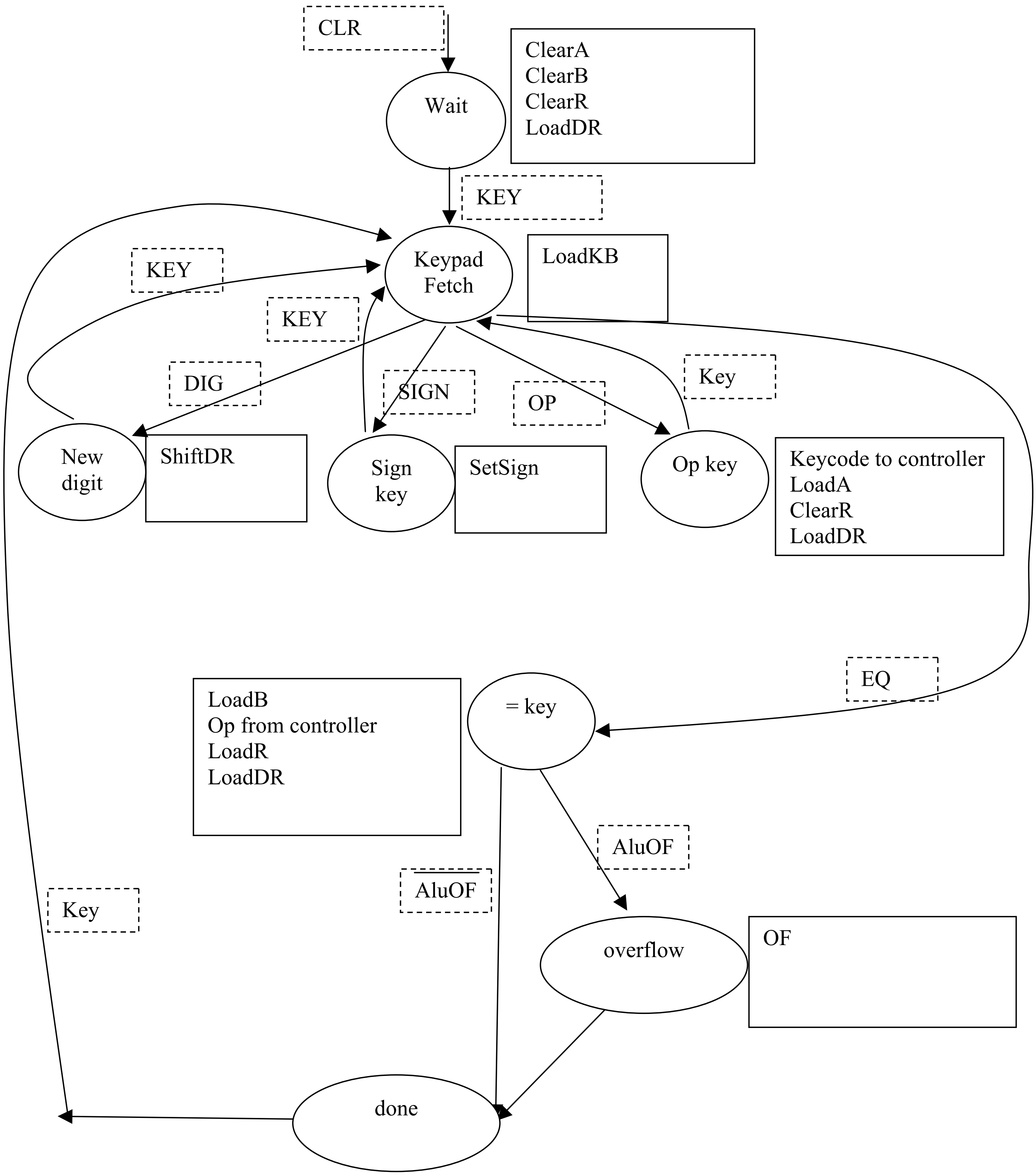
- ShiftSR – shifts LSB towards MSB. Sign bit is not shifted
- LoadSR – parellel load. Sign bit is also loaded.
- SetSign – Toggles sign bit

ALU

- Op – Opcode from controller

Name: Email: ID#

a) The basic controller FSM is provided below. Fill in the input conditions for the transition arcs in the dashed boxes. Also fill in the control signal outputs for each state in the solid boxes. Some of the values are entered for you.



Name: Email: ID#

b) What is the range of numbers (in decimal) that can be stored in:

i) The display register: -9999 to +9999

ii) RegA: -32768 to 32767

c) In the aforementioned FSM, what is the value visible on the display for the following key sequences:

i) CLR, 9, 3, 2, 3, 6: 3236

ii) CLR, 5, 3, +: 0

iii) CLR, 6, 3, 2, +, 1, 2, = 644

iv) CLR, 8, 2, +, +, 1, 2, = 12

d) Would you choose to implement the controller using a pure jump counter? Give reasons for your answer.

I would not use a pure jump counter, since several extra states would be required to implement the four-way branch at the keypad fetch step

e) Suppose you are implementing the controller using a vertical branch sequencer. List the contents of the ROM. Assume the states are Wait=000, Fetchkey=001, equalkey=010, overflow=011, done=100, digitkey=101, signkey=110, opkey=111. Assume that the inputs are coded as follows:

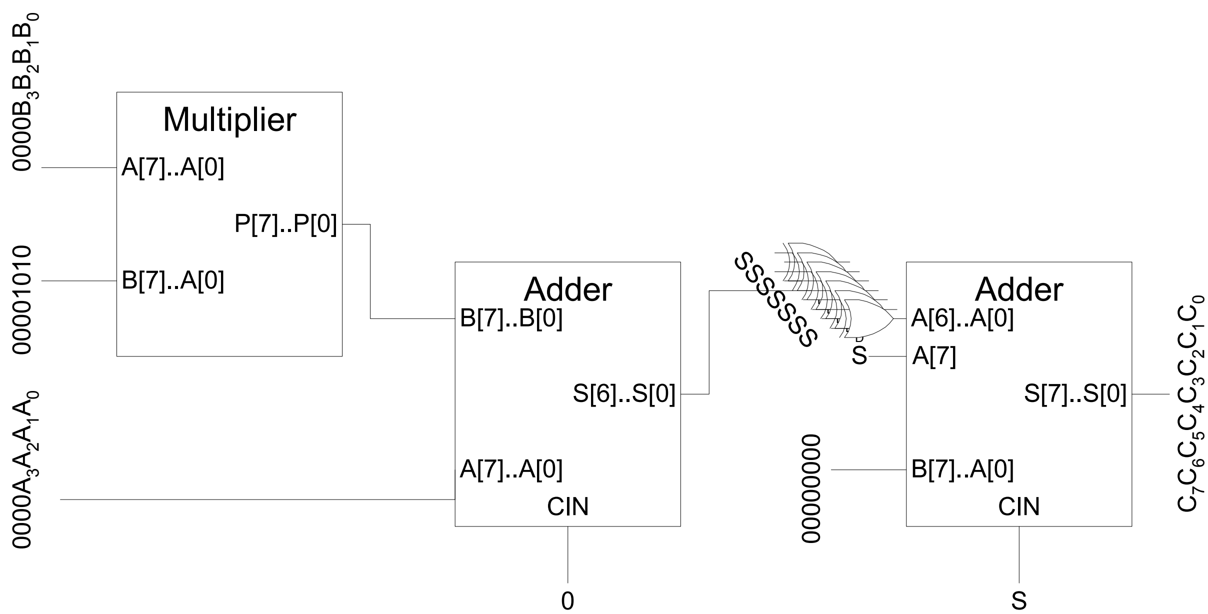
Input	α	β
DIG	0	0
OP	0	1
EQ	1	0
SIGN	1	1
KEY	X	1
KEY'	X	0
ALUOF	1	X
ALUOF'	0	X

State	Clear	Current State	α β	Next State	Outputs
	1	XXX	X X	000	ClearA, ClearB. ClearR, LoadDR
wait	0	000	X0	000	
	0		X1	001	LoadKB
Fetch	0	001	00	101	ShiftDR
	0		01	111	LoadA, ClearR, LoadDR, key→
	0		10	010	LoadB, Op←, LoadR, LoadDR
	0		11	110	SetSign
Equal	0	010	0X	100	
	0		1X	011	OF
Overf	0	011	XX	100	
Done	0	100	X0	100	
	0		X1	001	LoadKB
Dig	0	101	X0	101	
	0		X1	001	LoadKB
Sign	0	110	X0	110	
	0		X1	001	OF
Op	0	111	X0	111	
	0		X1	001	LoadKB

Name: Email: ID#

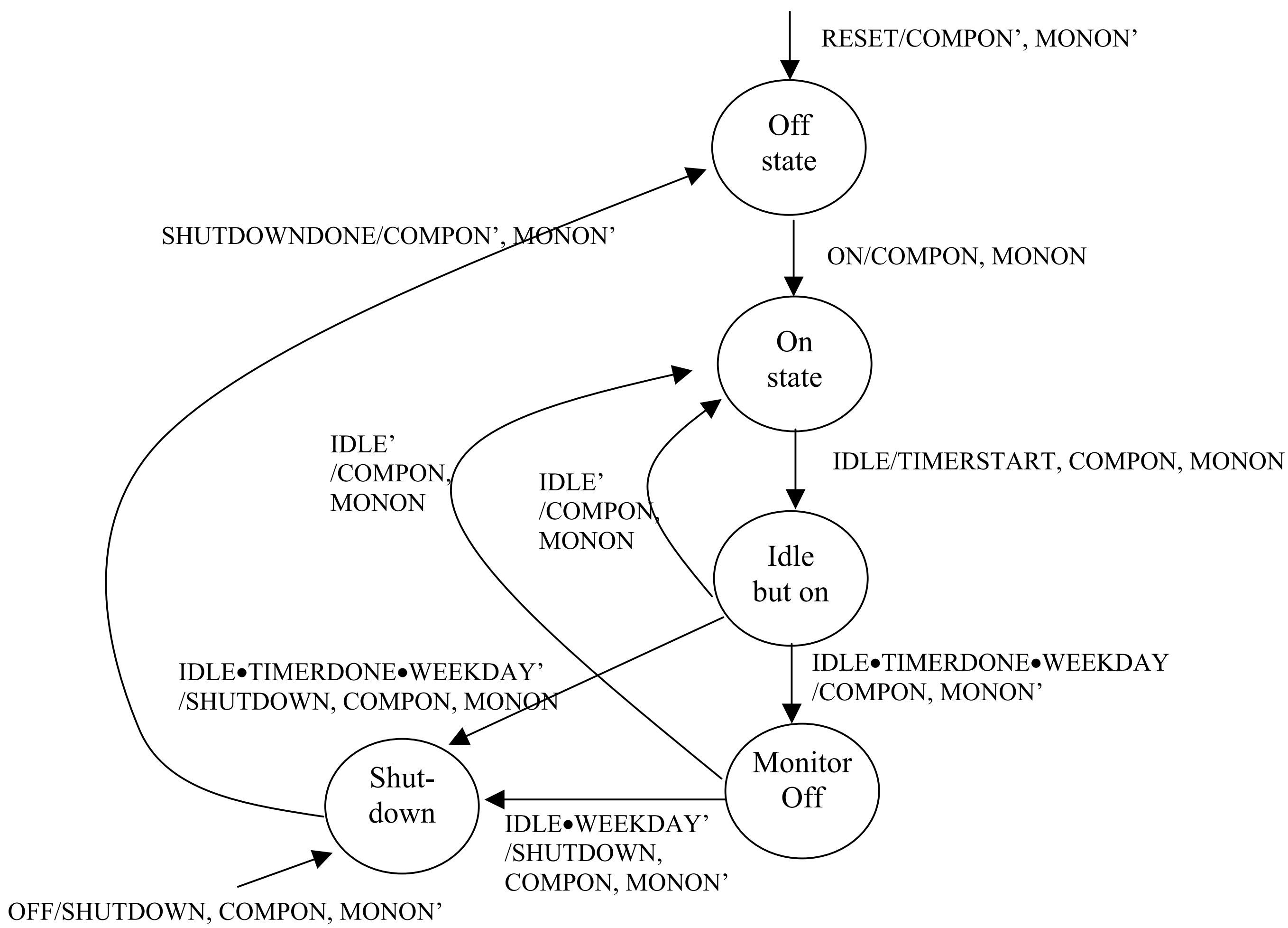
3) Design a 2 digit BCD to 2's complement converter, similar to the 4-digit system used in the calculator in Q1 (remember that there is a sign bit in front of the two BCD digits). You are given as many of the following blocks as you need:
   - 8-bit adders (***Inputs***: A[7..0], B[7..0], Cin ***Outputs***: S[7..0], Cout)
   - 8×8-bit multiplier (***Inputs***: A[7..0], B[7..0] ***Outputs***: P[15..0])
   - 2-input XOR gates
   - 2-input AND gates

The input word is: S B_3 B_2 B_1 B_0 A_3 A_2 A_1 A_0 (S is the sign, BA is the 2-digit BCD number). The output word should be $C_7 \ldots C_0$.



Name: Email: ID#

4) A substantial amount of power is wasted by leaving computers on (and unused) overnight. Design an advanced power supply for a computer that automatically turns it off when it is not needed as follows:
   - The computer is turned on by pressing the "ON" key on the power supply. This boots up the computer (COMPON output) and turns on the monitor (MONON output).
   - The computer is turned off by pressing the "OFF" key in the power supply. This gracefully shuts down the computer and monitor.
   - The power to the computer and monitor are disconnected by de-asserting the COMPON and MONON output respectively.
   - The computer shutdown proceeds in two phases – a SHUTDOWN signal causes the computer to save any open files. When this is complete, it responds with a SHUTDOWNDONE signal, at which point the power supply turns off the computer.
   - When the user logs off, the computer sends an IDLE signal. During office hours (WEEKDAY signal), the monitor is turned off after 20 minutes of inactivity. During evenings/nights/weekends, both the monitor and computer are turned off after this time.
   - If someone uses the computer while the monitor only is off, the monitor wakes up.
   - If the computer is completely off, it must be turned on my pressing the ON key
   - There is a 20-min timer started by TIMERSTART, ending with TIMERDONE.
   - A RESET key turns off the computer/monitor without waiting for the graceful shutdown.

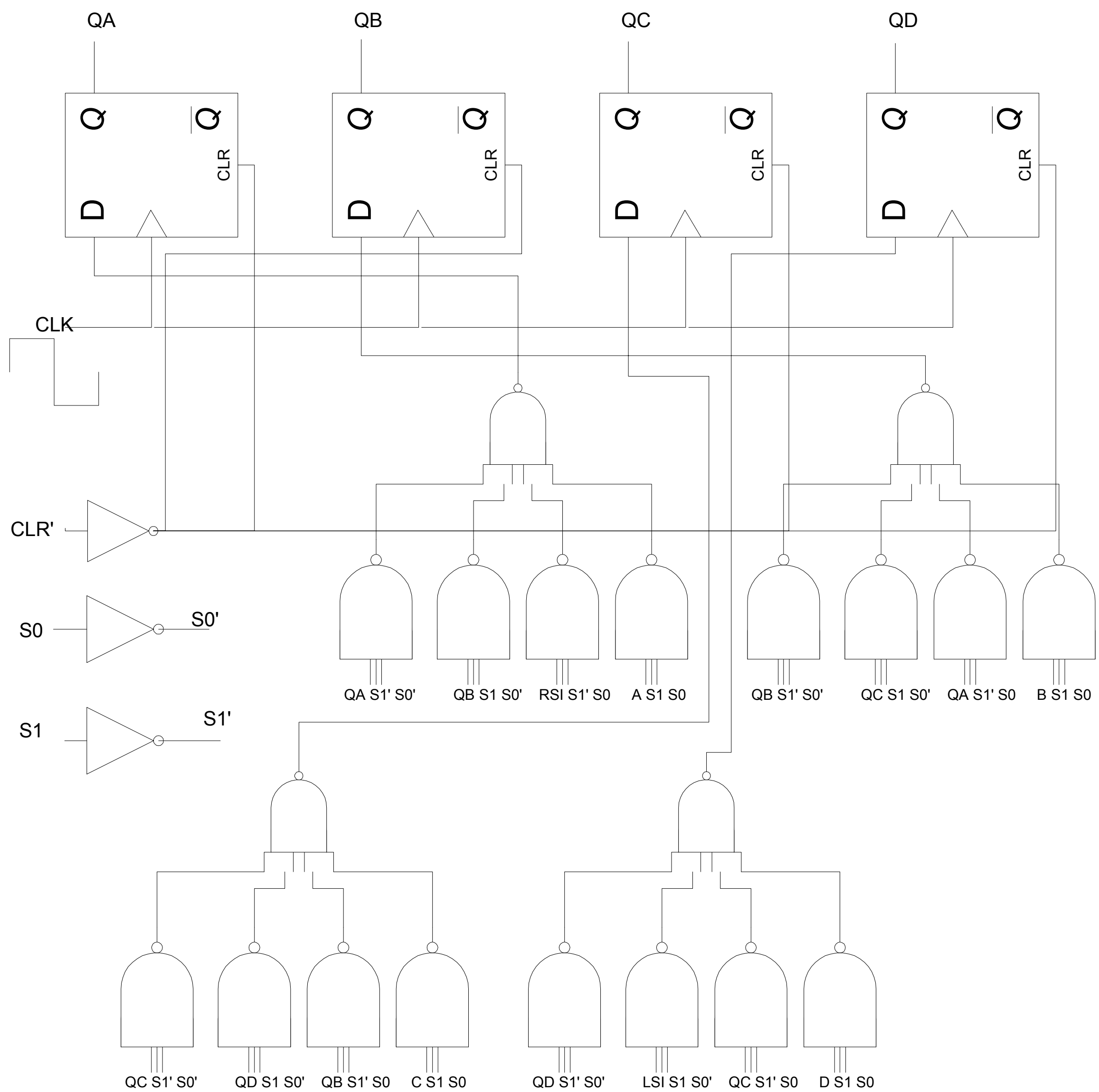


Name: Email: ID#

5) You are familiar with the 4-bit universal shift register, shown on page 333 of the text and on slide 33 of the sequential logic notes.

a) Design the internals of same, using D-FF's, NAND gates, and inverters as appropriate.

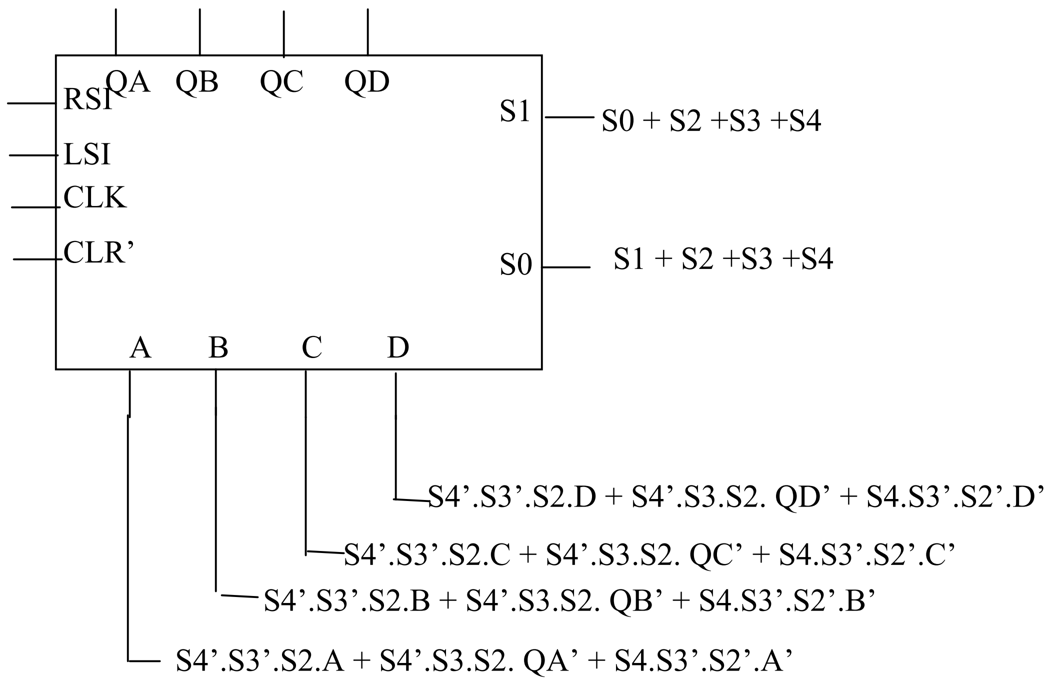
- D-FF pins are D (input), Q (output) Q' (output), CLK (clock) and R (reset).



Name: Email: ID#

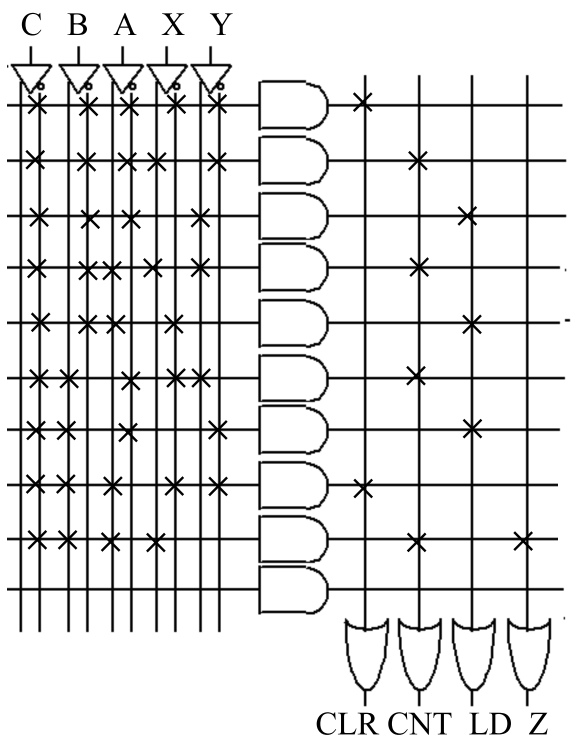
b) Using a standard universal shift register and external combinational logic, implement a one-hot “hyper-universal” shift register that offers the following functionality

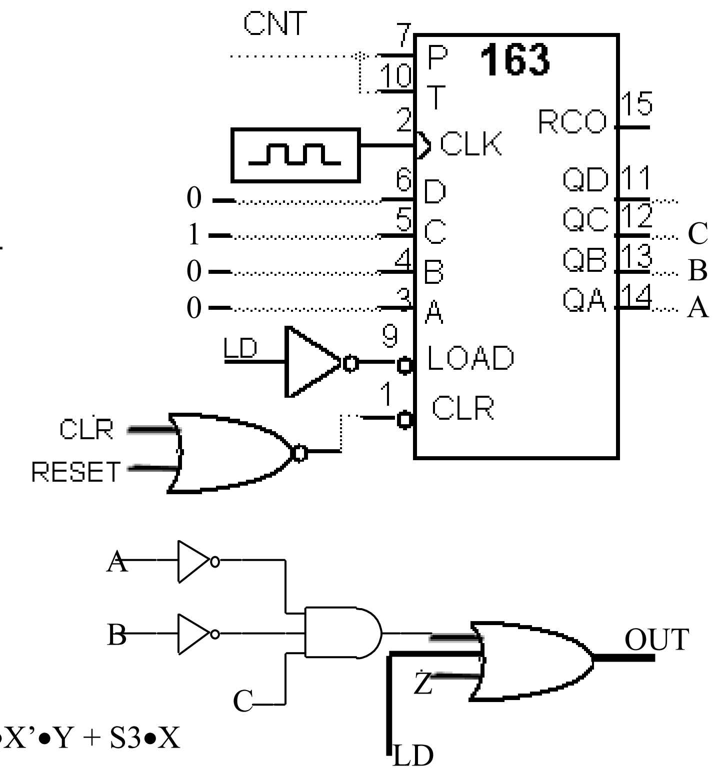
S4	S3	S2	S1	S0	Mode
0	0	0	0	0	Hold
0	0	0	0	1	Left Shift
0	0	0	1	0	Right Shift
0	0	1	0	0	Parallel Load data
0	1	0	0	0	Invert contents
1	0	0	0	0	Parallel Load inverted data



Name: Email: ID#

6) Given the following jump counter schematic, you are to derive the associated state diagram, including all transitions and outputs (X and Y are inputs, OUT is the output).





a) Determine the following functions:

i) CNT = S0•X•Y' + S1•X•Y+ S2•X'•Y + S3•X

ii) LD = S0•Y + S1•X' + S2•Y'

iii) CLR = S3•X'•Y' + S0•X'•Y'+ RESET

iv) OUT = S0•Y + S1•X' + S2•Y' + S3•X + S4

b) Draw the finite state diagram implemented by the above circuit

FORMAT: XY/OUT