EDA322: Digital Design
Final Exam - Spring 2014 study period 3

Date: Wednesday, March 12, 2014

Time: **14:00-18:00**

Examiner: Ioannis Sourdis

Department: Computer Science and Engineering

Inquiries: Ioannis Sourdis (extension 1744); will visit the room at **15:30** and at **15:30**

Results and grading review: See me in my office on **April 2nd at 10:00-12:00**.

Exam Duration: 4 hours

Grading scale: 100 points in total

Chalmers:
0: 0%-49%, 3: 50%-64%, 4: 65%-84%, 5: 85%-100%
GU:
Fail (U): 0%-49%, Pass (G): 50%-79%, Pass with Distinction (VG): 80%-100%

Available references: Blank paper and a calculator are allowed. No textbooks or lecture notes, etc. allowed.

General: Submit your solutions, in English, on blank paper sheets. Write legibly; feel free to use figures to get your point across.

The order of answering the questions does not matter (start with the easiest ones).

Please start the solutions for each problem on a new sheet. Please number the sheets so that the solutions are in numerical order.

Note that it is possible to receive partial credit for an answer even if it is not 100% correct.

Your personal identity code is required on each submitted sheet!

Good luck!

Question 1: (10 points)

a) Derive the canonical form of Sum of Products (SoP) for the following function F= ad'+ab+a'c'd (5 points)
b) Then, derive the canonical form of the Product of Sums (PoS) for the function F. (3 points)
c) What is a prime implicant and what is an essential prime implicant? (2points)

Answer:

Can be solved in 3 different ways:

1. Making the truth table of the function and then finding the minterms (the sum of which is the SoP canonical form of F) and the maxterms (the product of which is the PoS of F). (similar to Lecture 2 slide 12)
2. Through Shannon Theorem (similar to Lecture 2 slide 16)
3. Expanding each product (similar to Lecture 2 slide 17)
4. After step 2. or 3. The PoS is derived (similar to Lecture 2 slide 12)

(Above mentioned lecture slides can be found at the end of this document)

a)
F= ad'+ab+a'c'd =
ab'c'd' + ab'cd' + abc'd' + abcd' ~~+ abc'd'~~ + abc'd ~~+ abcd'~~ + abcd +a'b'c'd +a'bc'd =
Σ(1,5,8,10,12,13,14,15)

b) Π(0,2,3,4,6,7,9,11)

c) Answer in Lecture 2, slide 30
(Above mentioned lecture slides can be found at the end of this document)

Question 2: (10 points)

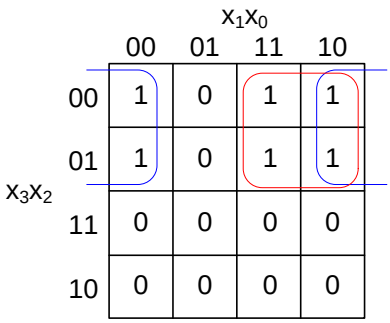
a) Minimize the circuit implementing the following two functions using Karnaugh (multiple functions minimization) F1(x3,x2,x1,x0)=Σ(0,2,3,4,6,7) and F2(x3,x2,x1,x0)=Σ(0,2,4,6,11,15). Draw the circuit. Which parts of the circuit are shared between the two functions? (3 points)
b) Minimize the cost of function:
F(x4,x3,x2,x1,x0) = Σ(0,1,2,3,17,21,25,29) + D(7,8,9,10,11,15,23,31)
using Quine-McCluskey. Measure the cost of the minimized function by counting the total number of 2-input gates of the circuit (e.g. a*b + c*d, has cost of 3). (7 points)

Answer:

a)

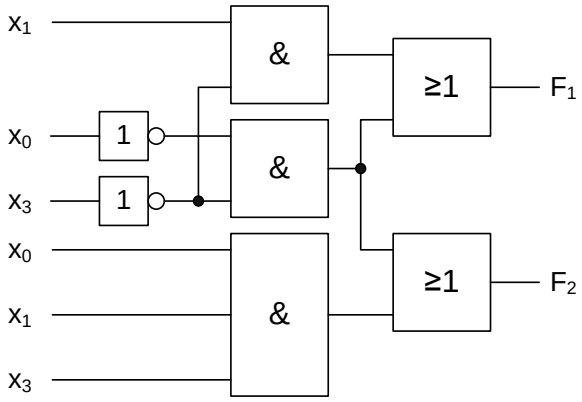
$$F_1(x_3, x_2, x_1, x_0) = \sum(0,2,3,4,6,7)$$

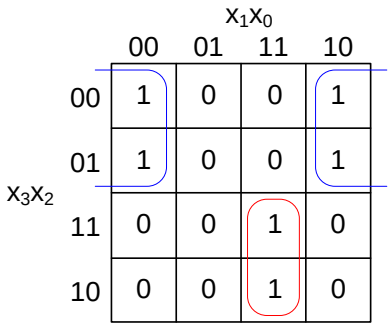
$$F_1(x_3, x_2, x_1, x_0) = \overline{x_3} \cdot \overline{x_0} + \overline{x_3} \cdot x_1$$



$F_1(x_3, x_2, x_1, x_0) = \sum(0,2,4,6,11,15)$

$F_1(x_3, x_2, x_1, x_0) = \overline{x_3} \cdot \overline{x_0} + x_3 \cdot x_1 \cdot x_0$





b)

Truth table for the minterms Σ(0,1,2,3,17,21,25,29):

i	x4	x3	x2	x1	x0	y
0	0	0	0	0	0	1
1	0	0	0	0	1	1
2	0	0	0	1	0	1
3	0	0	0	1	1	1
4	0	0	1	0	0	0
5	0	0	1	0	1	0
6	0	0	1	1	0	0
7	0	0	1	1	1	0
8	0	1	0	0	0	0
9	0	1	0	0	1	0
10	0	1	0	1	0	0
11	0	1	0	1	1	0
12	0	1	1	0	0	0
13	0	1	1	0	1	0
14	0	1	1	1	0	0
15	0	1	1	1	1	0
16	1	0	0	0	0	0
17	1	0	0	0	1	1
18	1	0	0	1	0	0
19	1	0	0	1	1	0
20	1	0	1	0	0	0
21	1	0	1	0	1	1
22	1	0	1	1	0	0
23	1	0	1	1	1	0
24	1	1	0	0	0	0
25	1	1	0	0	1	1
26	1	1	0	1	0	0
27	1	1	0	1	1	0
28	1	1	1	0	0	0
29	1	1	1	0	1	1
30	1	1	1	1	0	0
31	1	1	1	1	1	0

Truth table for the don't care terms D(7,8,9,10,11,15,23,31):

i	x4	x3	x2	x1	x0	h*
0	0	0	0	0	0	0
1	0	0	0	0	1	0
2	0	0	0	1	0	0
3	0	0	0	1	1	0
4	0	0	1	0	0	0
5	0	0	1	0	1	0
6	0	0	1	1	0	0
7	0	0	1	1	1	1
8	0	1	0	0	0	1
9	0	1	0	0	1	1
10	0	1	0	1	0	1
11	0	1	0	1	1	1
12	0	1	1	0	0	0
13	0	1	1	0	1	0
14	0	1	1	1	0	0
15	0	1	1	1	1	1
16	1	0	0	0	0	0
17	1	0	0	0	1	0
18	1	0	0	1	0	0
19	1	0	0	1	1	0
20	1	0	1	0	0	0
21	1	0	1	0	1	0
22	1	0	1	1	0	0
23	1	0	1	1	1	1
24	1	1	0	0	0	0
25	1	1	0	0	1	0
26	1	1	0	1	0	0
27	1	1	0	1	1	0
28	1	1	1	0	0	0
29	1	1	1	0	1	0
30	1	1	1	1	0	0
31	1	1	1	1	1	1

The minterms and don't care terms are put together:

Step 1:

Indexgruppe	Index
0	0 *
1	1 * 2 * 8 *
2	3 * 9 * 10 * 17 *
3	7 * 11 * 21 * 25 *
4	15 * 23 * 29 *
5	31 *

step2:

Indexgruppe	Index
0/1	1,0 (1) * 2,0 (2) * 8,0 (8) *
1/2	3,1 (2) * 9,1 (8) * 17,1 (16) * 3,2 (1) * 10,2 (8) * 9,8 (1) * 10,8 (2) *
2/3	7,3 (4) * 11,3 (8) * 11,9 (2) * 25,9 (16) * 11,10 (1) * 21,17 (4) * 25,17 (8) *
3/4	15,7 (8) * 23,7 (16) * 15,11 (4) * 23,21 (2) * 29,21 (8) * 29,25 (4) *
4/5	31,15 (16) * 31,23 (8) * 31,29 (2) *

step3:

Indexgruppe	Index
0/1/2	3,2,1,0 (1,2) * 9,8,1,0 (1,8) * 3,1,2,0 (2,1) * 10,8,2,0 (2,8) * 9,1,8,0 (8,1) * 10,2,8,0 (8,2) *
1/2/3	11,9,3,1 (2,8) * 11,3,9,1 (8,2) * 25,17,9,1 (8,16) P1 25,9,17,1 (16,8) 11,10,3,2 (1,8) * 11,3,10,2 (8,1) * 11,10,9,8 (1,2) * 11,9,10,8 (2,1) *
2/3/4	15,11,7,3 (4,8) P2 15,7,11,3 (8,4) 29,25,21,17 (4,8) P3 29,21,25,17 (8,4)
3/4/5	31,23,15,7 (8,16) P4 31,15,23,7 (16,8) 31,29,23,21 (2,8) P5 31,23,29,21 (8,2)

step 4:

Indexgruppe	Index
0/1/2/3	11,10,9,8,3,2,1,0 (1,2,8) P6 11,10,3,2,9,8,1,0 (1,8,2) 11,9,10,8,3,1,2,0 (2,1,8) 11,9,3,1,10,8,2,0 (2,8,1) 11,3,10,2,9,1,8,0 (8,1,2) 11,3,9,1,10,2,8,0 (8,2,1)
1/2/3/4	
2/3/4/5	

P1= x2'x1'x0
P2= x4'x1x0
P3= x4x1'x0
P4= x2x1x0
P5= x4x2x0
P6= x4'x2'

P/M1	0	1	2	3	17	21	25	29
P1		*			*		*	
P2				*				
P3					*	*	*	*
P4								
P5						*		*
P6	*	*	*	*				

F = P3 + P6 = x4x1'x0 + x4'x2'

Question 3: (10 points)

a) Make the block diagram and describe a 16-bit carry-select adder using blocks of 4-bits. Show the internals of a block. (7 points)
b) What is the area cost of the carry select adder compared to a ripple carry? (1 point)
c) What is the delay of a (i) ripple carry, (ii) a carry select, and (iii) a carry look-ahead adder, with respect to the number of their operant bits (N)? (2 points)

Answer:

a) Solution in Lecture 2 slides 78, showing the internals of a block as in slide 77.

(Above mentioned lecture slides can be found at the end of this document)

b) A 16-bit ripple carry adder would have 16 FAs. A carry select adder of 16 bits and blocks of 4 bits would have 7 blocks of 4 FAs, that is 28 FAs in total, plus 3* 5 Multiplexers (2-to-1). So the carry-select needs 1.75 times more area than the ripple-carry plus the area of the multiplexers.
c) i. O(N), ii. O($\sqrt{N}$), iii. O(logN).

Question 4: (10 points)

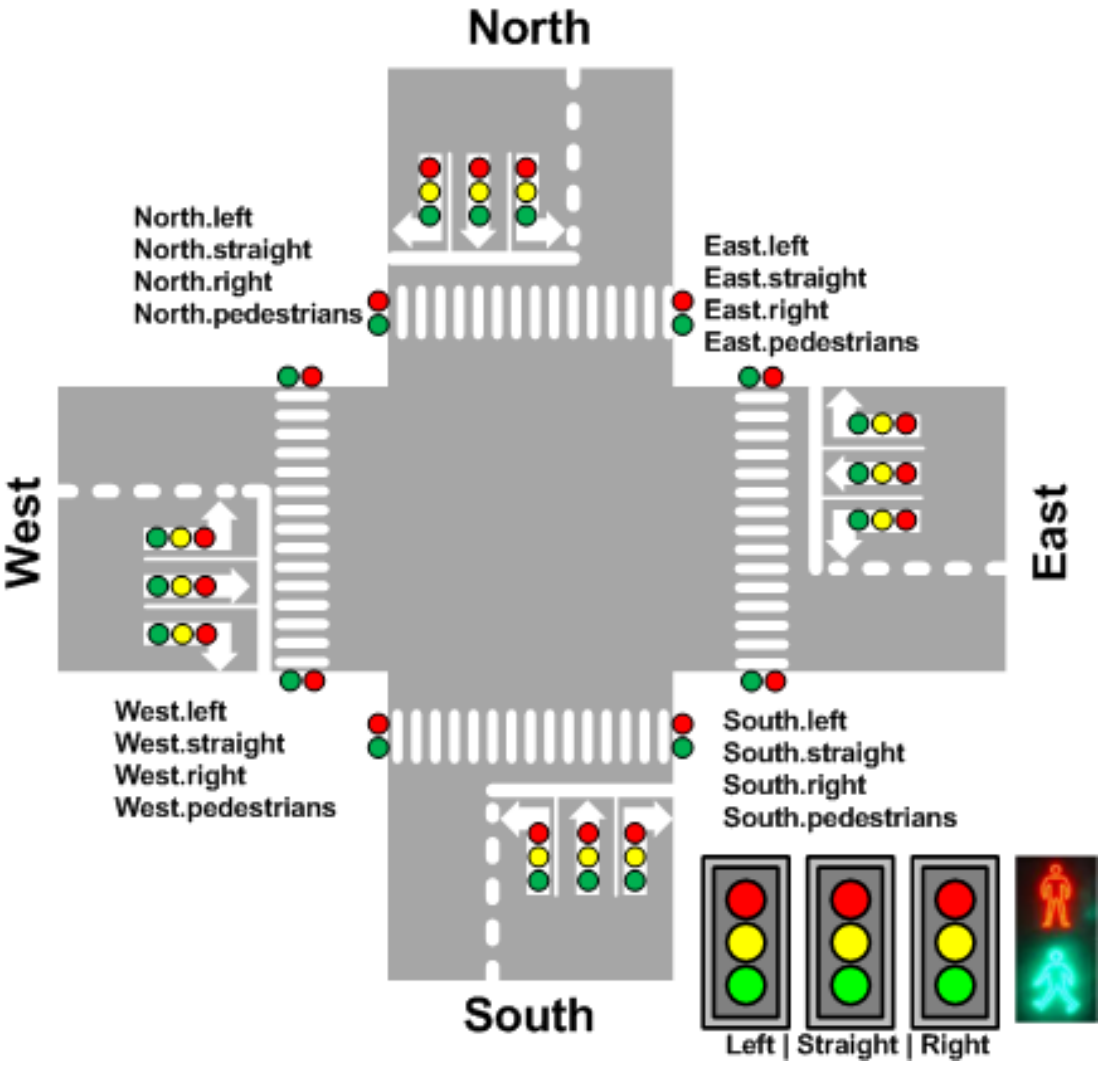
Design a synchronous Moore FSM (Finite State Machine) to control the traffic lights on the following crossroad.

Requirements:

- Make sure that each car has the opportunity to go to the direction it desires at least in one FSM state and that it does not crash with other cars!
- The pedestrians have the opportunity to cross the road at least in one FSM state, too.
- Each traffic light (for cars or for pedestrians) can be either red or green, represented in binary as '0' and '1' respectively (for simplicity do not consider the yellow color for the car traffic lights)

Optimizations:

- Try to maximize the number of directions/car-lanes "served" in parallel



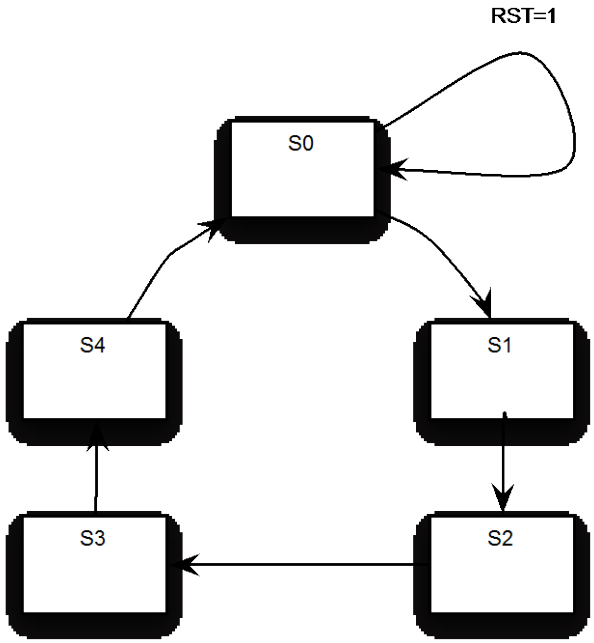
Inputs and Outputs of the FSM:

Name	IN/OUT	Number of bits	Description
Clock	IN	1	Clock of the FSM
Reset	IN	1	Asynchronous Reset
Cars.West.Left	OUT	1 (R=0, G=1)	Traffic lights for the cars and the pedestrians placed at the West, north, East or South part of the crossroad, indicating whether cars have permission to turn left, go straight, or turn right and whether pedestrians have permission to cross the road. Possible values: "red" and "green" indicating "stop" and "pass", respectively, encoded as "0" and "1", respectively.
Cars.West.Staight	OUT	1 (R=0, G=1)	
Cars.West.Right	OUT	1 (R=0, G=1)	
Cars.North.Left	OUT	1 (R=0, G=1)	
Cars.North.Staight	OUT	1 (R=0, G=1)	
Cars.North.Right	OUT	1 (R=0, G=1)	
Cars.East.Left	OUT	1 (R=0, G=1)	
Cars.East.Staight	OUT	1 (R=0, G=1)	
Cars.East.Right	OUT	1 (R=0, G=1)	
Cars.South.Left	OUT	1 (R=0, G=1)	
Cars.South.Staight	OUT	1 (R=0, G=1)	
Cars.South.Right	OUT	1 (R=0, G=1)	
Pedestr.West	OUT	1 (R=0, G=1)	
Pedestr.North	OUT	1 (R=0, G=1)	
Pedestr.East	OUT	1 (R=0, G=1)	
Pedestr.South	OUT	1 (R=0, G=1)	

a) Draw the state diagram of your FSM. (8 points)
b) What state encoding would you choose for your FSM, binary or one-hot? What would be the effect of your encoding state in terms of area and delay of your FSM? (2 points)

Answer:
(there can be more than one solutions)

a)



S0 outputs that take the value one (the rest of the outputs are at 0):

Pedestr.West
Pedestr.North
Pedestr.East
Pedestr.South

S1 outputs that take the value one (the rest of the outputs are at 0):

Cars.West.Left
Cars.West.Staight
Cars.West.Right

Cars.North.Right

S2 outputs that take the value one (the rest of the outputs are at 0):

Cars.East.Left
Cars.East.Staight
Cars.East.Right

Cars.South.Right

S3 outputs that take the value one (the rest of the outputs are at 0):

Cars.North.Left
Cars.North.Staight
Cars.North.Right

Cars.East.Right

S4 outputs that take the value one (the rest of the outputs are at 0):

Cars.South.Left
Cars.South.Staight
Cars.South.Right

Cars.West.Right

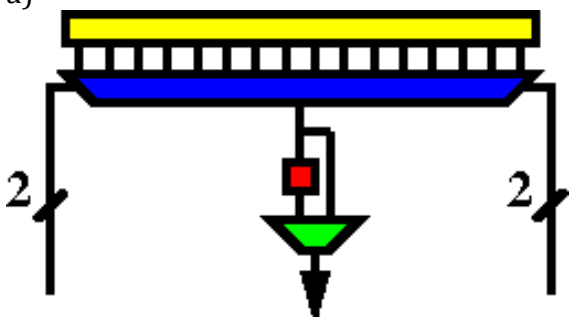
b) Binary encoding would require fewer storage bits than onehot and therefore would probably be more area efficient. On the other hand, onehot would probably create simpler logic and therefore be faster than binary.

Question 5: (10 points)

a) Draw a programmable FPGA logic cell, which has 4 inputs and 1 output. Consider that the output has the option to be either registered or not registered. (3 points)
b) Explain how this FPGA logic cell will be configured to implement a NOR gate without a flip-flop in the output and how it will be configured to implement an NAND gate with a registered output. (3 points)
c) How are the inputs and outputs of the logic cell connected to the reconfigurable wires of an FPGA? Draw the connections (4 points)

Answer:

a)



The yellow box is 16 bits of memory (or flip-flops), which together with the multiplexer implement a 4-input lookup table. The 4 bits inputs go to the select of the multiplexer. The output of the LUT goes to a flip-flip, and 2-to-1 multiplexer selects either the registered output or the one directly coming from the LUT.

b) for a 4-input NOR gate without registered output the LUT would have in the position "0000" the value "1" and in the rest of the 15 positions the value "0", in addition the select of the 2-to-1 multiplexer would choose the LUT output and not the registered one.
For a 4-input NAND gate with registered output the LUT would have in the position "1111" the value "0" and in the rest of the 15 positions the value "1", in addition the select of the 2-to-1 multiplexer would choose the the registered output coming from the flip-flop and not the LUT output.

c) Lecture 12, slide 30

(Above mentioned lecture slides can be found at the end of this document)

Question 6: (10 points)

a) Draw the block diagram of a memory with 1024 entries/rows and 32-bit elements (1k*32-bits), which is composed of other memory blocks, that have 512 entries and 8 bit elements (512*8bits). (5 points)
b) How many 512*8-bit memory elements do you need? (1 point)
c) How many address bits are needed for the 1k*32-bits? (1point)
d) Given a 1k*32-bits memory, how can one modify it to use it as byte-addressable? Draw the block diagram of the modified memory. How many bits of address are needed for the byte-addressable version of this memory? (3 points)

Answer:

a) Similar figure to slide 37 in lecture 11 (only with 4 columns of memory blocks instead of 2 columns and with 2 rows of blocks instead of 4). Each block is a 512 x 8 bits, which needs 9 bits to be addressed (A8-A0) the decoder needs 1 more additional bit (A10)
b) 8
c) 10
d) Similar to Lecture 11 slide 39 (rightmost figure). It is sufficient to describe how reading a byte is performed. It is of course also fine to give a more complete answer explaining that in order to write a byte you need to modify the 1k* 32 bits to 4k*8bits by changing the structure of the memory in (a) as follows: having one 512*8bits block per row, and in total 8 rows.

(Above mentioned lecture slides can be found at the end of this document)

Question 7: (10 points)

a) Which equation gives the Average (dynamic) Power of a hardware design? (4 points)
b) What are the (3) parameters that a Hardware designer can change (not including the technology parameters) when designing a hardware module in order to reduce the average power consumption, and how can this be done? (6 points)

Answer:
Answer in Lecture 16 slide 49

(Above mentioned lecture slides can be found at the end of this document)

Things a hardware designer can do to reduce power:

1. parameter "n": reduce the area of the design
2. parameter "α": reduce the activity of the submodules of a design, e.g. with clock gating (close the clock for submodules not used)
3. parameter "f": reduce the frequency of a design (in case it is not needed to be that fast, i.e., lower clock frequency gives good enough performance)

Question 8: (10 points)

a) Explain what is clock skew and how we can avoid it. (5 points)
b) Explain when a flip-flop may enter a metastable state and how we can avoid it. (5 points)

Answer:

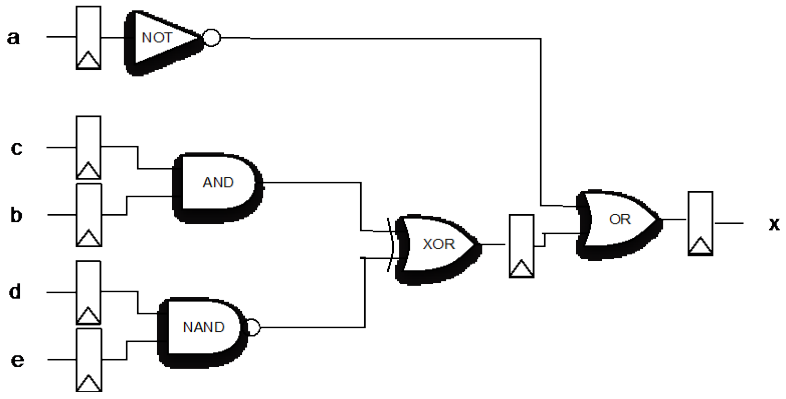
a) Answer in Lecture 16 slide 23. Clock skew can be reduced by making sure that all flip-flops and memories receive the clock with the same delay. To do that clock trees are designed where all the end leafs (connected to the flip-flops and memories) have the same distance from the root. These clock trees are H-trees.
b) Answer in Lecture 16 slides 31, 32

(Above mentioned lecture slides can be found at the end of this document)

Question 9: (10 points)

Calculate the maximum delay of the circuit (critical path delay), considering the following:

- a NOT gate has a delay of 1 ns
- an AND gate has a delay of 3 ns
- a NAND gate has a delay of 2 ns
- a XOR gate has a delay of 7 ns
- an OR gate has a delay of 3 ns
- Propagation time (clock to output) for a flip-flop 1ns
- Setup time for a flip-flop 2ns
- Wires have zero delay
- All inputs (a, b, c, d, e) have zero delay



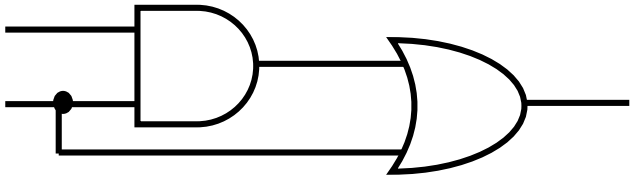
Answer:
Lets define the intermediate flip-flop Y, so the paths are:

a->x, delay = Tprop+Tsetup+Tnot+ Tor= 7ns
c-> Y, delay = Tprop+Tsetup+Tand+ Txor= 13ns
b->Y, delay = Tprop+Tsetup+Tand+ Txor= 13ns
d->Y, delay = Tprop+Tsetup+Tand+ Txor= 12ns
e->Y, delay = Tprop+Tsetup+Tand+ Txor= 12ns
Y->x, delay = Tprop+Tsetup + Tor= 6ns

So the two critical paths are c-> Y and d-> Y with delay 13 ns.

Question 10: (10 points)

a) The cost of a chip is 10 SEK when its yield is 60%. What will be its cost if you increased the yield to 90%. (3 points)
b) What is the total number of single stuck-at faults, counting both stuck-at-0 and stuck-at-1, in the following circuit? (4 points)
c) Which faults are left after equivalence fault collapsing? What is the collapse ratio? (3 points)



Answer:

Same like the example in lecture 15 slide 30->34.
For a) the numbers change so the answer is 6+2/3 SEK (6.666 SEK)

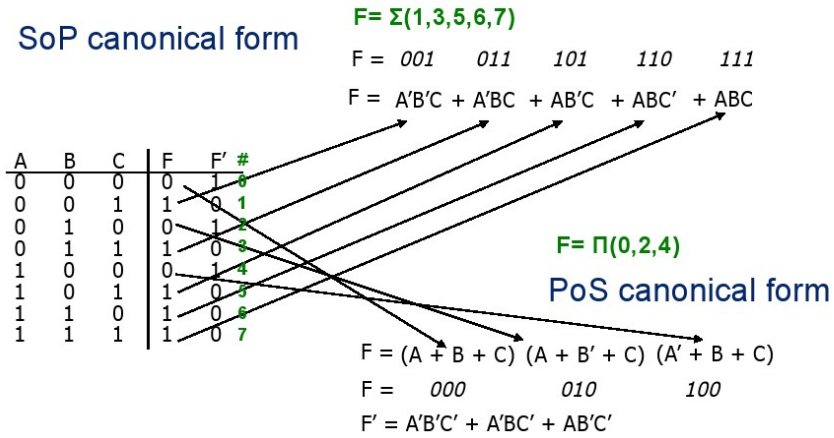
(Above mentioned lecture slides can be found at the end of this document)

END of the EXAM

Lecture Slides

Question 1:

Canonical Forms



canonical form ≠ minimal form

Conversion between canonical forms

Relation between maxterms and minterms of a function

$$F(x,y,z) = \Sigma(1,3,5,6,7) = \Pi(0,2,4)$$

… and for the inverted function:

$$F'(x,y,z) = \Pi(1,3,5,6,7) = \Sigma(0,2,4)$$

Shannon's expansion theorem

- ***Shannon's expansion theorem***

  (a). $f(x_1, x_2, \ldots, x_n) = x_1 * f(1, x_2, \ldots, x_n) + x_1' * f(0, x_2, \ldots, x_n)$

  (b). $f(x_1, x_2, \ldots, x_n) = [x_1 + f(0, x_2, \ldots, x_n)]\ [x_1' + f(1, x_2, \ldots, x_n)]$

Example:

$$F(x_1, x_2, x_3) = x_1x_2 + x_1x_3 + x_2x_3$$
$$= x_1'F(0, x_2, x_3) + x_1F(1, x_2, x_3)$$
$$= x_1' (0x_2 + 0x_3 + x_2x_3) + x_1(1x_2 + 1x_3 + x_2x_3)$$
$$= x_1' (x_2x_3) + x_1(x_2 + x_3 + x_2x_3)$$
$$= x_1' (x_2x_3) + x_1(x_2 + x_3(1 + x_2))$$
$$= x_1' (x_2x_3) + x_1(x_2 + x_3)$$

Derivation of Canonical Forms

- Derive canonical PoS or SoP using Boolean algebra.
- ***Shannon's expansion theorem***

  (a). $f(x_1, x_2, \ldots, x_n) = x_1 * f(1, x_2, \ldots, x_n) + x_1' * f(0, x_2, \ldots, x_n)$

  (b). $f(x_1, x_2, \ldots, x_n) = [x_1 + f(0, x_2, \ldots, x_n)]\ [x_1' + f(1, x_2, \ldots, x_n)]$

Convert a function to SoP

1. Apply Shannon Theorem until you get sum of minterms
2. Get the function to a SoP form, as follows:
   - if a product of literals is a minterm, keep it
   - for every variable x_i that doesn't exist in a product we add $(x_i + x_i')$, e.g.:
     - $x_1 * x_2 = x_1 * x_2 * (x_3 + x_3') = x_1 * x_2 * x_3 + x_1 * x_2 * x_3'$
   - Continue and delete redundant products

Derivation of Canonical Forms

- Derive canonical PoS or SoP using Boolean algebra.
- ***Shannon's expansion theorem***

  (a). $f(x_1, x_2, \ldots, x_n) = x_1 * f(1, x_2, \ldots, x_n) + x_1' * f(0, x_2, \ldots, x_n)$

  (b). $f(x_1, x_2, \ldots, x_n) = [x_1 + f(0, x_2, \ldots, x_n)]\ [x_1' + f(1, x_2, \ldots, x_n)]$

- ***Example***: $f(A,B,C) = AB + AC' + A'C$
  - $f(A,B,C) = AB + AC' + A'C = A\,f(1,B,C) + A'\,f(0,B,C)$
    $= A(1 \cdot B + 1 \cdot C' + 1' \cdot C) + A'(0 \cdot B + 0 \cdot C' + 0' \cdot C) = A(B + C') + A'C$
  - $f(A,B,C) = A(B + C') + A'C = B[A(1+C') + A'C] + B'[A(0 + C') + A'C]$
    $= B[A + A'C] + B'[AC' + A'C] = AB + A'BC + AB'C' + A'B'C$
  - $f(A,B,C) = AB + A'BC + AB'C' + A'B'C$
    $= C[AB + A'B \cdot 1 + AB' \cdot 1' + A'B' \cdot 1] + C'[AB + A'B \cdot 0 + AB' \cdot 0' + A'B' \cdot 0]$
    $= ABC + A'BC + A'B'C + ABC' + AB'C'$

Derivation of Canonical Forms

- Derive canonical PoS or SoP using Boolean algebra.
- ***Shannon's expansion theorem***

  (a). $f(x_1, x_2, \ldots, x_n) = x_1 * f(1, x_2, \ldots, x_n) + x_1' * f(0, x_2, \ldots, x_n)$

  (b). $f(x_1, x_2, \ldots, x_n) = [x_1 + f(0, x_2, \ldots, x_n)]\ [x_1' + f(1, x_2, \ldots, x_n)]$

- ***Example***: $f(A,B,C) = AB + AC' + A'C$

  $f(A,B,C) = AB + AC' + A'C =$

  $= AB(C+C') + AC'(B+B') + A'C(B+B')$

  $= ABC + ABC' + ABC' + AB'C' + A'BC + A'B'C$

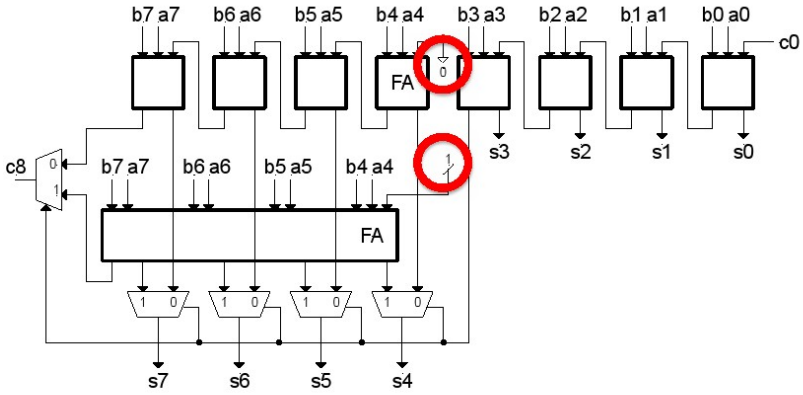
  $= ABC + ABC' + AB'C' + A'BC + A'B'C$

Definition of terms

- A function *F* **covers** a function *g*, if it takes the value '1' when function *g* does.
- **Implicant**: a product of literals of a function *f* for which *f* gets the value '1'.
- **Prime implicant:** an implicant that cannot be covered by a more general (more reduced - meaning with fewer literals) implicant.
- **Essential prime implicant:** a prime implicant of function *f* that includes a minterm, not included by any other prime implicant of the function.

Question 3:

Carry Select Adder

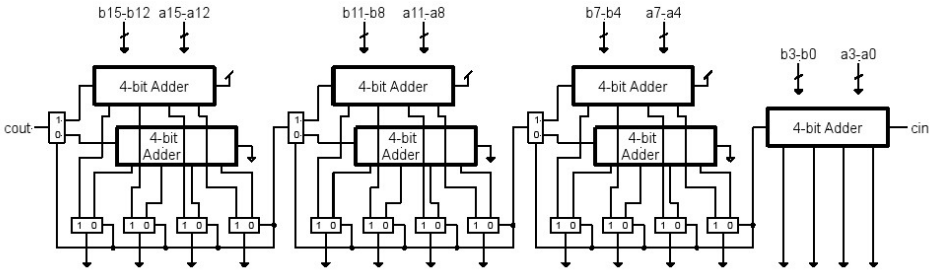


$$T = T_{ripple\_adder} / 2 + T_{MUX}$$

$$COST = 1.5 * COST_{ripple\_adder} + (n+1) * COST_{MUX}$$

Carry Select Adder

- Extending Carry-select to multiple blocks

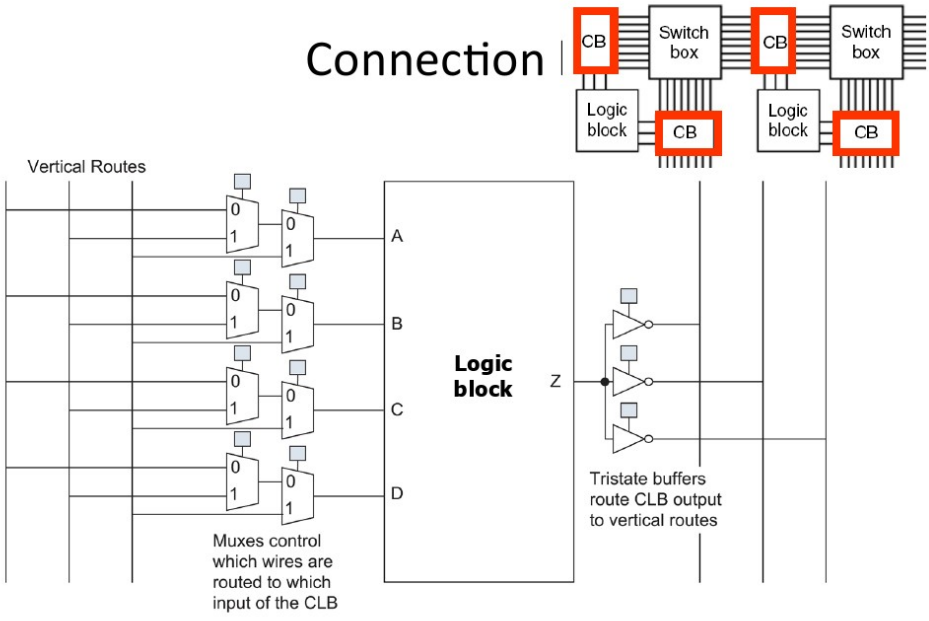


- What is the optimal # of blocks and # of bits/block?
  - If # blocks too large delay dominated by total mux delay
  - If # blocks too small delay dominated by adder delay

$\sqrt{N}$ stages of $\sqrt{N}$ bits

T α sqrt(N),
Cost ≈2*ripple + muxes

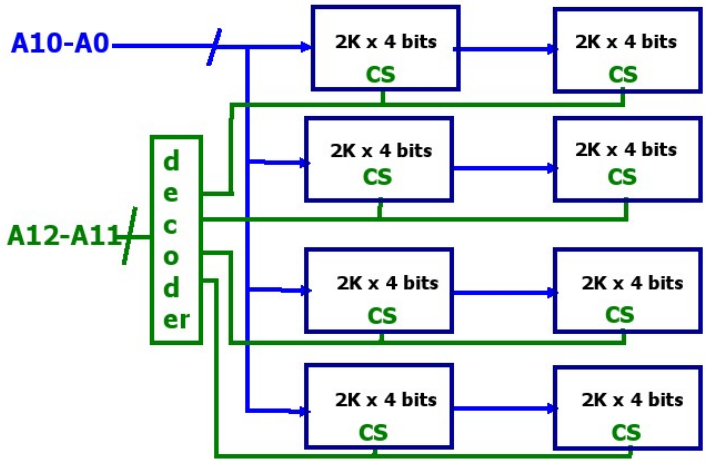
Question 5:



Question 6:

Using Memory Building Blocks

– Building an 8K byte memory using chips that are 2K by 4 bits.

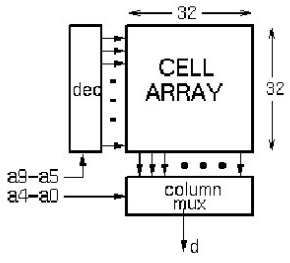


- **CS = chip select: when set, it enables the addressing, reading and writing of that chip.**

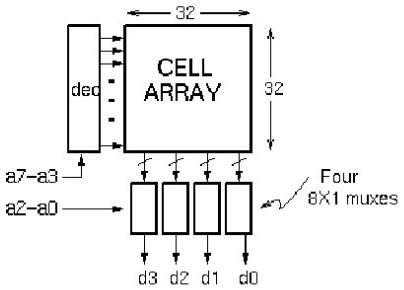
This is an 8KB byte addressable memory

Column MUX in ROMs and RAMs:

- Controls physical aspect ratio
- In DRAM, allows reuse of chip address pins



1K X 1 ROM



256 X 4 ROM

Question 7:

Controlling Energy Consumption

- Largest contributing component to CMOS power consumption is switching (dynamic) power:

Lower frequency less power, but longer execution time so, maybe more energy (?)

$$P_{avg} = n \cdot \alpha_{avg} \cdot f \cdot c_{avg} V_{dd}^{\ 2}$$

- What control do you have over each factor?
- How does each effect the total Energy? (think about f)

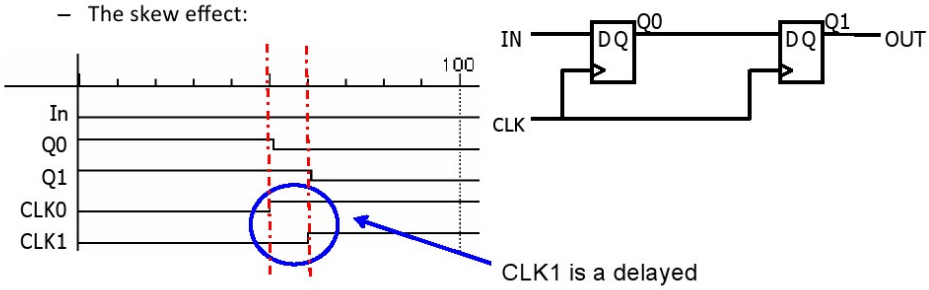
Less activity of the submodules of A design results in lower power

Smaller area
Fewer transistors+wires
Lower power

Question 8:

Clock Skew

- The Problem
  - correct behavior requires that the next state of all memory elements are determined by all the memory elements thus at the same time
  - This is difficult in high performance systems since the time it takes for the clock to arrive, are of the same magnitude as the delay through the logic
  - The skew effect:

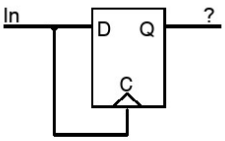


CLK1 is a delayed version of CKL0

Initial state: IN = 0, Q0 = 1, Q1 = 1
Due to skew next state is : Q0 = 0, Q1 = 0,
instead of Q0 = 0, Q1 = 1

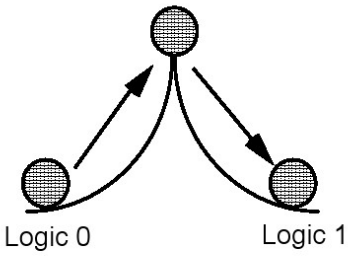
Metastability and Asynchronous Inputs

Synchronizer Failure

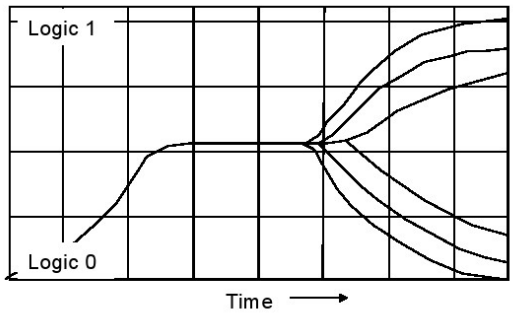


When FF input changes close to clock edge, the FF may enter the *metastable* state: neither a logic 0 nor a logic 1

It may stay in this state an indefinate amount of time, although this is not likely in real circuits



Small, but non-zero probability that the FF output will get stuck in an in-between state

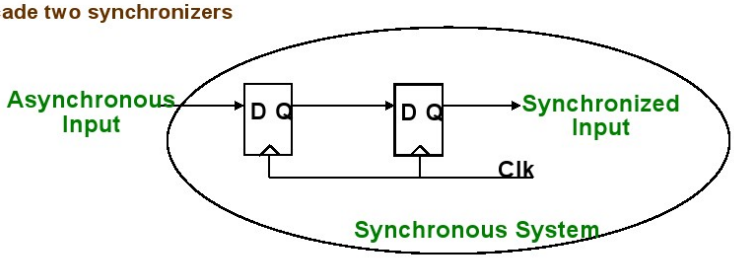


Oscilloscope Traces Demonstrating Synchronizer Failure and Eventual Decay to Steady State

Metastability and Asynchronous Inputs

Solutions to Synchronizer Failure

- **the probability of failure can never be reduced to 0, but it can be reduced**
- **slow down the system clock**
  this gives the synchronizer more time to decay into a steady state
  synchronizer failure becomes a big problem for very high speed systems
- **use fastest possible logic in the synchronizer**
  this makes for a very sharp "peak" upon which to balance
  S or AS TTL D-FFs are recommended
- **cascade two synchronizers**

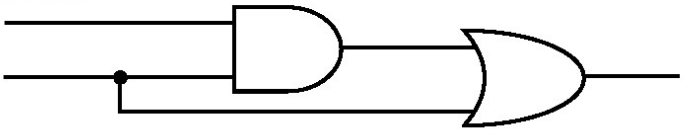


Question 10:

Quiz 13-1

http://m.socrative.com/student/#joinRoom
room number: 713113

- Q1: What is the yield of a wafer that has 3 out of 10 chips defective:
  a) 30%
  b) 70%
  c) 130%
- The cost of a chip is $1.00 when its yield is 50%. What will be its cost if you increased the yield to 80%.
- What is the total number of single stuck-at faults, counting both stuck-at-0 and stuck-at-1, in the following circuit?
- Which faults are left after equivalence fault collapsing? What is the collapse ratio?



Answers to the Exercise

- The cost of a chip is US$1.00 when its yield is 50%. What will be its cost if you increased the yield to 80%.

Assume a wafer has *n* chips, then

$$\text{Chip cost} = \frac{\text{wafer cost}}{0.5 \times n} = \$1.00$$

$$\text{Wafer cost} = 0.5n \times \$1.00 = 50n \text{ cents}$$

For yield = 0.8, chip cost = wafer cost/(0.8*n*) = 50*n*/(0.8*n*) = 62.5 cents

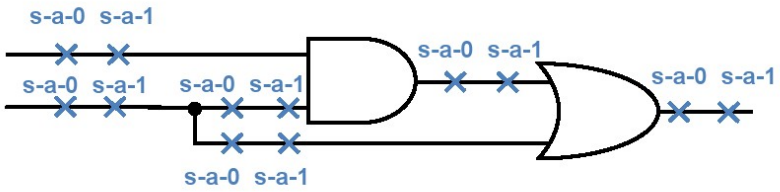
Answers Continued

- What is the total number of single stuck-at faults, counting both stuck-at-0 and stuck-at-1, in the following circuit?

Counting two faults on each line,

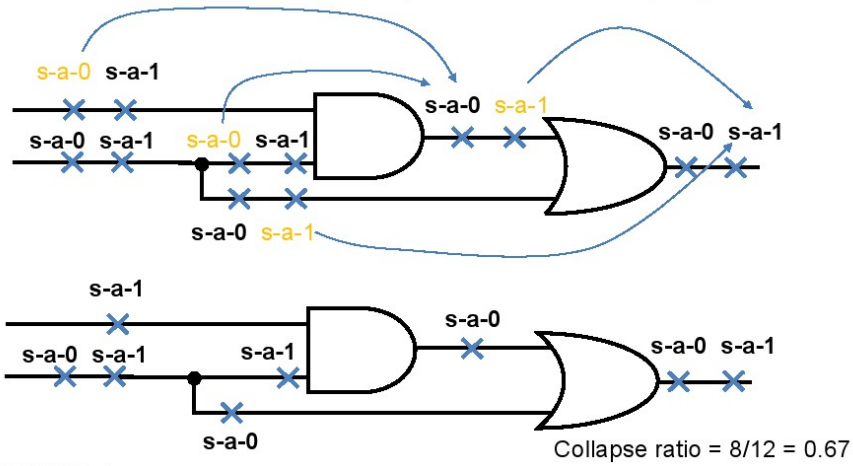
Total number of faults = 2 × (#PI + #gates + #fanout branches)

= 2 × (2 + 2 + 2) = 12



Answers Continued

- How many faults are left after equivalence fault collapsing?



Collapse ratio = 8/12 = 0.67