

ENEE244 (sec 0101-0104) Spring 2006

Midterm Examination II

Pages: 7 printed sides

Name: _____ *Answer key*
Student ID: _____

Time allotted: **50 minutes.**
Maximum score: **50 points**

University rules dictate strict penalties for any form of academic dishonesty. Looking sideways will be penalized. Look at only your own exam at all times.

There are 6 questions, some with subparts. Read them *carefully* to avoid throwing away points!! Write your answer in the space provided. **Closed book, closed notes, no calculators.**

Partial credit rule: Must show your intermediate steps clearly for partial credit!

1. For each subpart (i) to (v) below, circle **all** correct answers from among the four given - note that **more than one** answer may be correct!

(2 points * 5 = 10 points)

- (i) Regarding the minimum form of a combinational circuit,

(a) the minimum in sum-of-products always has the same number of literals as the minimum in product-of-sums.

(b) it may be the same size as the canonical form of the circuit.

(c) the minimum form in sum-of-products is always unique.

(d) it can only be smaller if some '1' minterms in the canonical form are converted to don't cares and the rest are retained as in the original function.

- (ii) A gated SR latch

(a) has only one combination of inputs that remembers the previous state.

(b) can be built using NAND gates alone.

(c) has no input combinations that result in unpredictable operation.

(d) changes state only on the negative edge of the clock pulse.

(iii) A master-slave JK flip-flop

- (a) stores inputs to the master only when the clock input is one.
- (b) changes output only on a particular edge of the clock pulse.
- (c) has no input combinations that result in unpredictable operation.
- (d) can have its operation completely specified by a truth table.

(iv) A D-latch provides the following functionality:

- (a) Set (make output=1).
- (b) Reset (make output=0).
- (c) Retain state.
- (d) Toggle (complement) state.

(v) A n-input decoder with enable input

- (a) can be used to implement any Boolean function of n inputs.
- (b) is identical to a demultiplexer of a certain size.
- (c) is a combinational circuit.
- (d) is a good fit for building the keyboard interface circuit for a cell phone to convert button-press events into binary numbers.

2. Minimize the 3-input function $\Sigma m(3,4,7) + dc(1,5,6)$ using a K-map.

(4 + 3 = 7 points).

(a) Express the answer in sum-of-products form.

Since this function is in canonical form, we can directly fill the K-map instead of going through the truth table.

K-map:

		$x \backslash yz$			
		00	01	11	10
0	0	X	1	0	
1	1	X	1	X	

$$F_{\min} = x + z.$$

(b) Re-compute the minimum form in product-of-sums form.

For product of sums we combine the 0's on the K-map.

		$x \backslash yz$			
		00	01	11	10
0	0	X	1	0	
1	1	X	1	X	

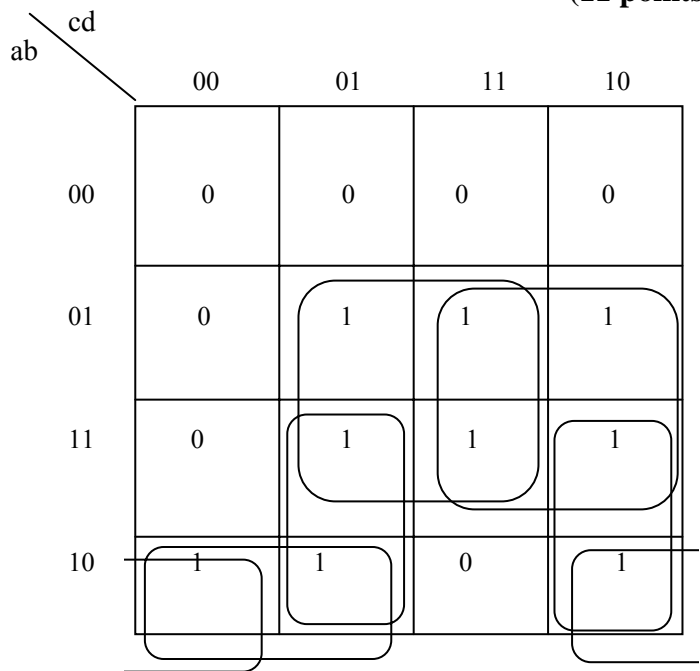
$$F_{\min}' = x'z'$$

$$(F_{\min}')' = (x'z')'$$

$$F_{\min} = x + z.$$

$x + z$ is unusual in that it is an expression in both product-of-sums and sum-of-products.

3. Draw the K-map for the 4-input function $F(a,b,c,d) = \sum m(5,6,7,8,9,10,13,14,15)$. List all the essential prime implicants, as well as the non-essential prime implicants. Write all the minimum sum-of-products forms of the function. **(11 points)**



Prime implicants:

m_5, m_7, m_{13}, m_{15} : Essential (induced by m_5) // term is bd

m_6, m_7, m_{14}, m_{15} : Essential (induced by m_6). // term is bc

m_8, m_9 : Non-essential (all minterms included in other PIs). // term is $ab'c'$

m_8, m_{10} : Non-essential (all minterms included in other PIs). // term is $ab'd'$

m_9, m_{13} : Non-essential (all minterms included in other PIs). // term is $ac'd$

m_{10}, m_{14} : Non-essential (all minterms included in other PIs). // term is acd'

Minimum functions contain all essential PIs and only those non-essential PIs needed to cover minterms not covered by essentials.

Minterms not covered by essential PIs: m_{12}, m_{13} and m_{14} .

These three minterms can be covered by three combinations of two non-essential PIs, resulting in three possible minimum functions.

(i) $bc + bd + ab'c' + ab'd'$

(ii) $bc + bd + ab'c' + acd'$

(iii) $bc + bd + ac'd + ab'd'$

All the above forms are equivalent and any one can be used to implement the function.

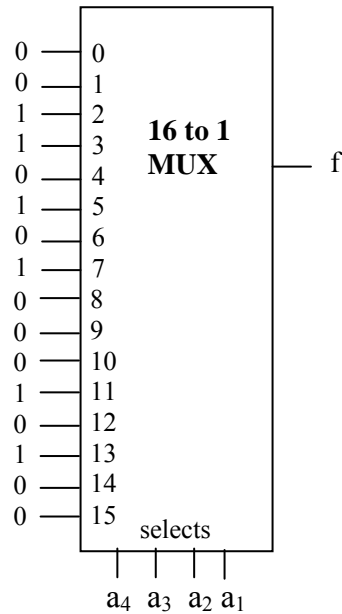
4. Design a circuit that takes a four-bit input $A = a_4 a_3 a_2 a_1$ which checks if A is a prime number. Use a **16-to-1 MUX** in your design. Recall that a number A is prime if the only numbers that divide it with no remainder are 1 and A itself. (0 and 1 are defined as not prime).

(6 points)

Among the four-bit number 0 to 15, the prime numbers are 2,3,5,7,11,13.

The prime number checker is therefore simply $\Sigma m(2,3,5,7,11,13)$.

This is implemented as with a MUX by connecting the included minterms to 1, rest to 0:

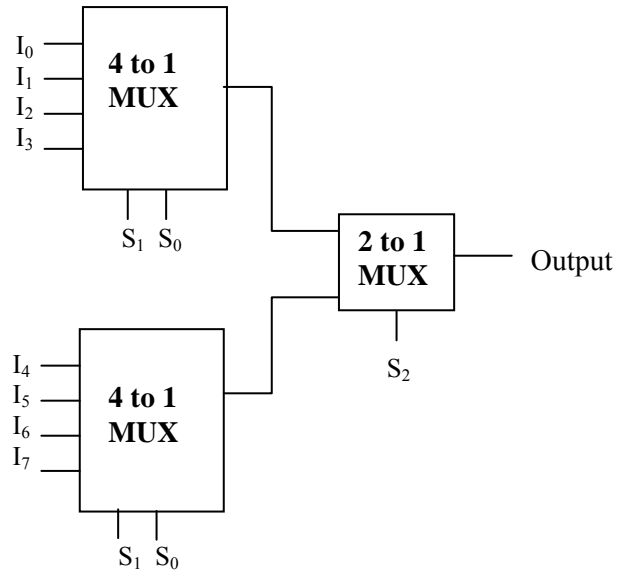


In this circuit, $f=1$ when A is prime; otherwise $f=0$.

5. Construct an 8-to-1 MUX using any number of 4-to-1 and 2-to-1 MUXs as building blocks. Use the smallest number of component MUXs that you can.

(8 points)

In the circuit below for the 8-to-1 MUX, $I_0 \dots I_7$ are the data inputs; S_2, S_1, S_0 are the selects.



There is another correct solution which uses four 2-to-1 MUXs at the first level selected by S_0 , and one 4-to-1 MUX at the second level, selected by S_2 and S_1 . This solution uses more MUXs than the above solution, but the MUXs are of smaller size, so it may be just as good. Both solutions get full credit.

6. We wish to design a **prefix detector** circuit that takes two 16-bit inputs $A=a_{15}\dots a_0$ and $B=b_{15}\dots b_0$. The output should be a 16-bit number $C=c_{15}\dots c_0$ which has a 1 at the first position from the right at which A and B are unequal. For example:

$$\begin{aligned} A &= 1101\ 1010\ 0101\ 0111 \\ B &= 0100\ 1110\ 0101\ 0111 \end{aligned}$$

Therefore:

$$C = 0000\ 0100\ 0000\ 0000$$

Here A and B are identical for bits 1-10 but not a bit 11; hence $c_{11} = 1$. All the other outputs are zero.

Design a prefix-detector (PD) cell such that 16 PD-cells linked together compose the entire circuit. Show how the PD cells should be linked. You do not need to formally minimize your outputs using K-maps, just use your intuition and, if necessary, Boolean simplification.

(8 points)

Let $A_i = a_i \dots a_0$. Similarly define B_i and C_i .

PD-cell inputs: a_i, b_i and e_i , where $e_i \equiv (A_i = B_i)$.

PD-cell outputs: c_i and e_{i+1} .

We express the outputs in terms of the inputs:

c_i is true if $a_i \neq b_i$ and e_i is true.

$\Rightarrow c_i = (a_i \oplus b_i) \cdot e_i$. //XOR is inequality checker.

e_{i+1} is true if $(a_i = b_i)$ and e_i .

$\Rightarrow e_{i+1} = (a_i \odot b_i) \cdot e_i$. //XNOR is equality checker.

The 16 PD-cells are linked such that the output e of stage i is linked to the input e of stage $i+1$. Moreover $e_1=1$ (vacuously true), and e_{17} is discarded.

