CSEE W4823x **Homework 1** Handout 8
Prof. Steven Nowick September 15, 2016

This homework is due **at the beginning of class** on Tuesday, September 27.
Note: A correct answer without adequate explanation or derivation will have points deducted. To get full credit, (a) write legibly or type, and (b) show all work: *label each step, show your intermediate results and derivations, and include clear verbal explanations of your steps.*

1. (10 points) **2:4 Decoder as a Finite Basis.** In this problem, you will demonstrate that a 2:4 decoder is a 'finite basis': *any logic circuit* can be built *only using 2:4 decoders.*

   Review: In class lecture, it was shown that one can implement any Boolean function using 3 simple gates: {AND2, OR2, INV}, where AND2 means 'two-input AND', OR2 means 'two-input OR', and INV is an inverter. This set of gates forms a "finite basis". Simpler finite bases were also identified: {AND2, INV}, {OR2, INV}, {NAND2}, and {NOR2}.

   What to Do: You are to demonstrate that a 2:4 decoder can be used as a finite basis by showing that, for one of the above finite basis sets (you pick one), every one of its gates can be implemented by using a 2:4 decoder.

   Rules: You are allowed to tie any of the decoder inputs together if needed. Also, because this decoder — unlike class examples — has *multiple outputs*, and we want to use this component to build a single-output gate, you are free to pick one of the four outputs as your new gate output; the others are ignored and not tied to the final output. However, of course, you *cannot* add any extra gates: you must use only 2:4 decoders.

   Show all work, and add text to justify your solution; inadequate explanation will result in point deductions.

2. (15 points) **Quine-McCluskey Method: a Word Problem.** In this problem, you are to explore the power and generality of the QM method to capture and solve different real-world optimization problems.

   Set-up: John and Abby plan to hold a party, and are deciding on what type of food to serve. They are considering 5 possible main dishes: pasta, steak, pad thai, burritos, and sushi. They are willing to serve more than one main dish, but want to serve the fewest possible number of different main dishes while still satisfying every guest. They are inviting 4 guests, so (including the hosts) there will be 6 people at the party. They have asked about food preferences.

   - John eats pasta, steak and burritos;
   - Abby eats pasta, pad thai and sushi;
   - Srini eats pasta and pad thai;
   - Igor eats pasta, burritos and sushi;
   - Francoise eats pasta, pad thai, burritos and sushi;
   - Xi eats steak.

   In this problem, you are to first set up this covering problem as a prime implicant table – clearly identifying what are the "covering objects" (analogous to prime implicants) and what are the "objects to be covered" (analogous to ON-set minterms). After defining this table, you are to iterate through the QM method, to produce a final solution.

   Note: clearly label each iteration number, and each substep (essentials, row dominance, column dominance) performed within each step. Also, be sure to show all intermediate results.

Answer the following:

(a) *QM Method, Step #2.* Formulate the above problem using a a prime implicant table. Show the final matrix, clearly labelling columns and rows. (Be sure to use the same orientation of rows and columns as in the handout.)

Also, answer briefly: what are the columns (i.e. *covering objects*)? and what are the rows (i.e. *objects to be covered*)?

(b) *QM Method, Step #3.* Using the Quine-McCluskey Method, reduce and solve your constraint matrix from part (a) above. For Step #3 of the method, follow the steps in the Quine-McCluskey Handout: **do the steps in precisely the same order as in the handout**. Iterate through as many loops as necessary to produce an empty table. (You should not need Step #4 for this problem.)

Show and label all intermediate results: show all work for each substep, clearly indicating essentials, row and col dominance at each step, in the same order as on the QM handout.

(c) *Short Questions.* In this food selection problem: what is an essential? what is the meaning of row dominance? what is the meaning of column dominance? (1-2 sentences each question)

3. (5 points) **Quine-McCluskey Method: Step #4.** For this problem, refer to an already-reduced constraint matrix, given below. This 4x4 matrix is cyclic. Apply Petrick's method to solve the matrix, as follows: (a) write the constraint equation for the matrix (i.e., product of sums equation); (b) multiply out and solve the equation, finding a minimum-cost cover, using Boolean simplification. (Refer to the Quine-McCluskey Handout, Example #2.) *Show all work.*

	p_1	p_2	p_3	p_4
m_1	1	1	0	0
m_2	0	1	1	1
m_3	1	0	0	1
m_4	1	0	1	0

4. (20 points) **Simple Word Problem: Mealy Machine Specification.** A finite state machine has one input (X) and two outputs (Z_1 and Z_2). An output $Z_1 = 1$ occurs each time the input sequence 010 is just observed, provided the sequence 100 has never been seen. An output $Z_2 = 1$ occurs each time the input 100 is just observed. Note that once $Z_2 = 1$, $Z_1 = 1$ can never occur.

Note: For full credit, your state diagram should have close to the minimum number of states. A small increase over minimum is fine, you do *not* need to perform a state minimization procedure. However, unduly large state diagrams will receive a small penalty.

What to Do: Write out the Mealy symbolic state diagram for this specification.

5. (24 points) **Complex Word Problem: Moore FSM Specification for DNA Sequencing.** For this problem, you will write a state diagram specification for a *pattern detector for DNA sequencing*.

Setup/Assumptions. The problem of detecting patterns or "sequencing" is critical in modern genomics. A DNA sequence consist of patterns of 4 distinct symbols or *bases:* A, C, G, and T. These symbols are ordered in DNA into large and complex sequences, which from our viewpoint can simply be regarded as a linear "string" which mixes the 4 symbols in some order.

For this problem, you are to assume that the 4 symbols have already been translated into a binary 2-bit code, as follows: $A = 00$, $C = 01$, $G = 10$, and $T = 11$. Also assume that the FSM receives 1 symbol per clock cycle, i.e. applied on 2 input wires, $x1$ $x0$. The detector has a single output, z.

In this problem, you will design a pattern detector to detect the presence of one particular *subsequence* embedded in the DNA input sequence. Whenever the complete subsequence is identified, the FSM will assert its z output to 1 for one clock cycle.

Note: For purposes of this assignment, your designs should allow the possibility not only of an isolated instance of the subsequence occurring in the DNA, but also two or more subsequences back-to-back as well as mutually-overlapping subsequences.

Problem Formulation: DNA Sequence Detector for the "Initiator Motif". The pattern $YYAN(T/A)YY$ is a *DNA transcription promoter* (also called "Inr").

For readability, the above pattern uses a shorthand notation: the variable Y is an abbreviation for "C or T" (just one of these two symbols, i.e. "C/T"), and the variable N is an abbreviation for "any symbol" (i.e. "just one of A, C, G or T"). The notation "(T/A)" means "just one of T or A". **Note that there are no actual Y or N input symbols! The inputs include only the 4 base symbols (A, C, G, T): Y and N are just used to simplify the notation of the pattern.** Hence, your state diagram must handle all strings consisting only of the alphabet of 4 base symbols (A, C, G, T). Therefore, the Initiator Motif contains 7 base symbols that meet the above pattern.

Some background (not critical to solving this problem): DNA encodes RNA that encodes proteins. The two respective processes of decoding are called transcription and translation. The Initiator Motif above demarcates where the transcription process should start for many genes. See also the Wiki URL: http://en.wikipedia.org/wiki/Initiator_element, and more general web sites such as http://en.wikipedia.org/wiki/Dna.

Problem Summary: In this problem, you are to write a single *Moore specification,* i.e. symbolic state diagram, that detects this subsequence. *Do not implement this specification, just write the state diagram!*

Your specification reads a new symbol on each clock cycle as its input, encoded digitally by *2 input wires* ($x1, x0$), and *1 output wire* (z). Input symbols can consist only of bases A, C, G and T, with 1 input symbol received per clock cycle. Each time a complete DNA transcription promoter subsequence is detected, the output is asserted high (i.e. 1) for one clock cycle, after the last symbol of the subsequence. At all other times, the output is deasserted low (i.e. 0). See "Setup/Assumptions" above for further details.

What to Do: Draw the corresponding *Moore* symbolic state diagram.

Note: Your specification should be neatly drawn, with all states clearly marked. *All 4 possible input transitions in each state must be explicitly labelled: do not omit any input combinations.* For good documentation, *a simple text label should indicate the purpose of each state,* either placed next to the state or in a readable separate key (i.e. table).

For full credit, your state diagram should have close to minimal number of states. You do not need to use state minimization, simply be careful to avoid large numbers of extra states (a few extra states are fine, though).

6. (26 points) **Complete Sequential Design Method: FSMs.**
In this problem, you are to *2 complete FSM designs*, following the method presented in class (see also B/V ch. 8.1 and 8.3, as well as ch. 8.5, and solved problems in ch. 8.13).

   The FSM is a variant of the *pattern detector* presented in B/V ch. 8.1: for this problem, you are to design an FSM which has 1 input *In* and 1 output *Out*. The FSM asserts its output, Out, to 1 precisely if the *most recent 2 inputs received are different,* i.e. either 01 or 10. Each implementation should use D flipflops.

   (a) Design a *Moore Machine* implementation.

   (b) Design a *Mealy Machine* implementation.

   For both parts (a) and (b), *show and label all steps:* (i) write the symbolic state diagram (Moore or Mealy); (ii) write the equivalent symbolic state table; (iii) assign state codes (i.e., state assignment); (iv) write the state-assigned table (also called the "binary state transition table", i.e. including the assigned state codes); (v) assuming D-flipflops, write the K-maps for next-state and outputs; (vi) write logic equations for each next-state and output; (vii) draw the final optimized 2-level circuit using D-flipflops.