CS226 Practice Problem Set 3 (Spring 2016)

Date posted: Feb 22, 2016 **Expected Solving Time: 1 hour**

- *If you need to make any assumptions, state them clearly.*
- ***These are ungraded practice questions. You are strongly encouraged to solve these on your own to ensure you understand the material being taught in class.***
- ***Mutual discussion is allowed, but copying is not. Please read the guidelines on the course webpage if you don't understand the distinction between the two.***

1. Consider the Boolean function $F(x_1, x_2, x_3, x_4, x_5) = x_5 \oplus ((x_2 \oplus x_3).((x_1 \oplus x_3') + x_4))$, where $\oplus$ represents XOR (i.e. $0 \oplus 0 = 0, 1 \oplus 0 = 1, 0 \oplus 1 = 1$ and $1 \oplus 1 = 0$).

   (a) Construct an ROBDD (without complemented edges) for F using the variable order $x_5 < x_4 < x_2 < x_3 < x_1$.

   (b) Now show how the part of the ROBDD involving nodes labeled x_2 and x_3 would change if were to swap the order of x_2 and x_3. Recall our discussion in class about swapping the order of adjacent variables in an ROBDD.

   (c) Now suppose we want to introduce complement edges (i.e. edges with inversion bubbles) such that (i) no solid edge (or 1-edge) has an inversion bubble, *and* (ii) there is no leaf labeled 0. Show how the ROBDD from sub-question (a) above would change if we replaced every incoming edge to the 0 leaf with an edge with an inversion bubble to the 1 leaf, and then used the transformations shown in class to move inversion bubbles out of solid edges (or 1-edges).

   (d) Now simplify the ROBDD with complement edges obtained in the previous sub-question by merging nodes that represent the same functionality. How many nodes are reduced compared to the ROBDD (without complement edges) obtained in sub-question (a)?

2. Consider Binary Decision Diagrams (BDDs) without complemented edges. Suppose we relax the requirement that every path from the root to a leaf must have the same ordering of variables. However, we do require that no two distinct nodes must represent the same function (i.e. we have Reduced BDDs, and not Reduced Orderd BDDs). Fig. 1 shows an example of such a BDD. Note that the variable orders $a < b < c$ is used along one path from the root to a leaf, while $a < c < b$ is used along another path.

   Consider the ROBDD shown in Fig. 2. Give a BDD with the above relaxation (i.e. variables may be ordered differently along different paths) that represents the same function but has strictly fewer nodes.

3. In this question, we will consider Reduced Ordered Binary Decision Diagrams (ROBDDs) without complemented edges. Let $\{x_1, x_2, x_3, x_4\}$ be a set of Boolean variables, and let $x_1 < x_2 < x_3 < x_4$ be the ordering of variables when constructing ROBDDs. For every Boolean formula formula φ on $\{x_1, x_2, x_3, x_4\}$ (or a subset of it), let $N(\varphi)$ denote the number of nodes in the ROBDD of the formula φ using the above variable ordering.

   Give an example of a two-variable function $\psi(x, y)$ such that $N(\psi(x_1, x_4) \wedge \psi(x_2, x_3)) > N(\psi(x_1, x_4)) + N(\psi(x_2, x_3))$. Indicate clearly the ROBDDs $\psi(x_1, x_4)$ and $\psi(x_1, x_4) \wedge \psi(x_2, x_3)$.

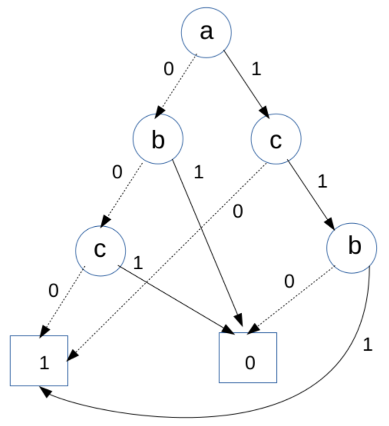

Figure 1: Binary Decision Diagram (not ordered though!)

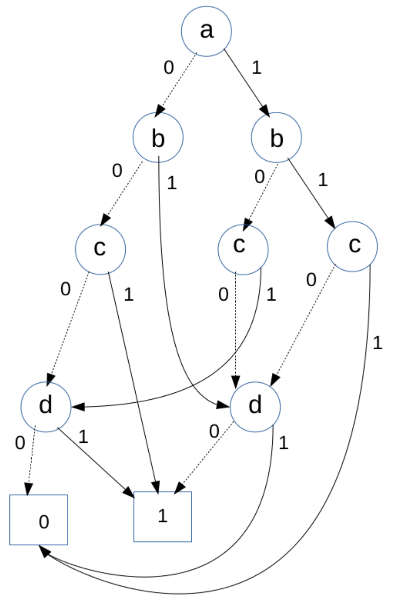

Figure 2: Reduced Ordered Binary Decision Diagram

4. This question shows how the same datapath can be used to implement two different algorithms, by using it with two different controllers. The datapath is as shown in Fig. 3, and has the following components: five registers named U, V, W, P, Q of appropriate bit-width, six multiplexors with control signals $M_w, M_u, M_v.M_a, M_b, M_q$, an adder/subtractor block that takes two inputs A and B and computes either $A+B$ or $A-B$ depending on the value of $doAdd$ (adds if $doAdd = 1$, subtracts otherwise). The adder/subtractor also always gives a one-bit output A_lt_B that is set to 1 iff $A < B$ (treated as signed integers). The datapath has an input $Reset$ that is used to reset all the registers, i.e when $Reset = 1$, the values stored in each of U, V, W, P, Q becomes 0. It is also assumed that all registers in the datapath are clocked by the same clock signal as the controller. There are two inputs X and Y to the datapath, and the output of register Q is assumed to be the result.

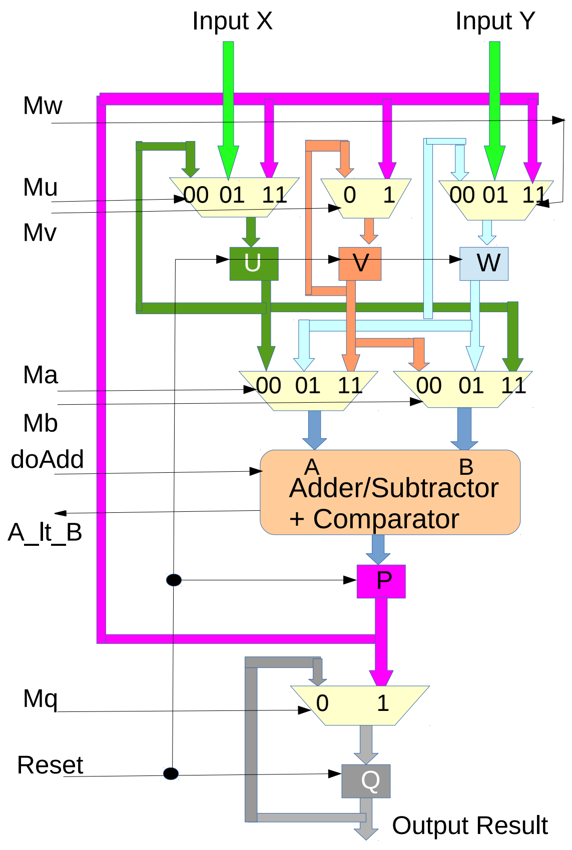

Figure 3: Common datapath

The two algorithms we wish to implement are given below:

```
Algorithm 1:                Algorithm 2:

 read X and Y;               read X and Y;
 Z = 0;                      do
 while (X < Y) do              X = 2X;
    Z = X + Y;                 Z = X-Y;
    X = X - Y;                 Y = 2Z;
    Y = 2Y;                  while (Z < X);

 Result = 2Z;                Result = Y;
```

Design two controllers (with appropriate state variables) that produces the right sequence of values on the various control inputs of the datapath such that the above two algorithms are implemented with the datapath shown in Fig. 3. You may choose to store the values of X, Y and Z in any of the registers in the datapath. You may also use the registers to store intermediate values, if required.

For each controller design, you must give the state transition table, clearly listing the values of next state and all control signals that are input to the datapath, for each combination of present state and inputs from the datapath. Try to reduce the number of states needed in your controller in each case.