18-240: Structure and Design of Digital Systems

Electrical & Computer ENGINEERING

HW2 Solutions [9 problems, 64 points]

Covers lectures L4 – L5 *Due: 18 September 2023*

Drill Problems [28 points]

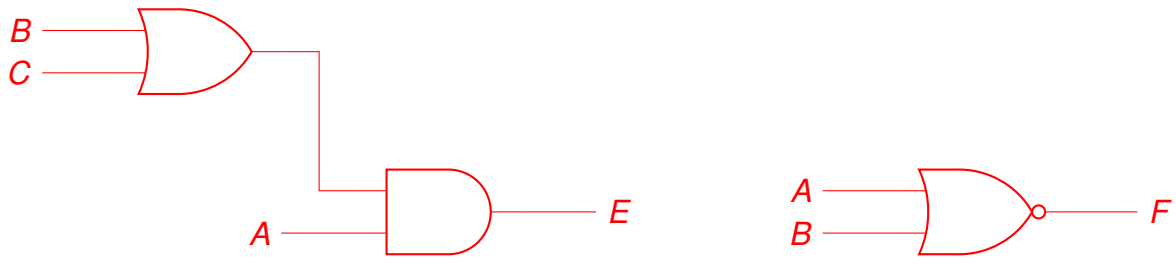
1. [12 points, Lecture 5] Write a SystemVerilog module and testbench for each of the following components.

   See **`hw2library.sv`** on Canvas. As you can guess from that filename, you will be adding more stuff to your **`library.sv`** file over future homework assignments.

2. [6 points, Lecture 4] Sketch a schematic of the circuit described by the following SystemVerilog code. Do not simplify:

```
module hw2prob2
  (output logic e, f,
   input  logic a, b, c);

  assign e = a & (b | c);
  assign f = ~ (a | b);
```



3. [4 points, Lecture 4] Using *procedural* SystemVerilog, write a module that computes a four-input XOR function. The module has a 4-bit input **`a`**, and output **`y`**.

   `hw2prob3.sv` has been posted to Canvas.

4. [6 points, Lecture 4] Using *procedural* SystemVerilog, write a module called **`Minority`**. The module's output is a one only if more than half of its inputs are zero.

   `hw2prob4.sv` has been posted to Canvas.

Non-Drill Problems [36 points]

5. [8 points, Lecture 5] Use an 8:1 multiplexer module to implement $F(A,B,C,D) = A'B'CD + A'B + ABD + A(B'CD' + C'D)$.

   hw2prob5.sv is posted on Canvas.

6. [12 points, Lecture 5] Let's test Shannon's Expansion Theorem. Here is a 6-input function:

   hw2prob6.sv is posted on Canvas.

7. [4 points, Lecture 4] Write a SystemVerilog module that receives as input a year number, **year**, and produces an output, **leap4**, that is one if the year is evenly divisible by 4 (i.e., the remainder is 0), and zero otherwise. Assume that valid years are zero through 2040, inclusive.

   hw2prob7.sv has been posted to Canvas.

8. [4 points, Lecture 4] But, wait a minute! If you know about leap years, you probably know that the "divide by four" definition is not quite correct. A year is a leap year if:

   (a) The year is evenly divisible by four

   (b) except in the case where it is evenly divisible by 100 and thus is not a leap year,

   (c) except in the case where it is evenly divisible by 400 and thus is a leap year.

   Your answer to the last problem only followed rule #1. Implement a better version, using the following header, to correctly determine if a year is, in fact, a leap year. You may use your module from the last problem to implement this newer version, but if you do, please include a copy in your file submission.

   hw2prob8.sv has been posted to Canvas.

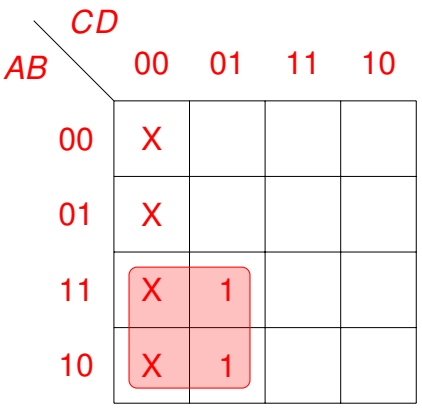
9. [8 points, Lecture 4] After all this talk about not dividing in the previous problems, let's go ahead and build a very simple divider.

Develop a minimized Boolean implementation of a 2-bit divider. The system has 2-bit inputs **AB** (dividend) and **CD** (divisor), and generates 2-bit outputs, **EF** (quotient), and **GH** (remainder).

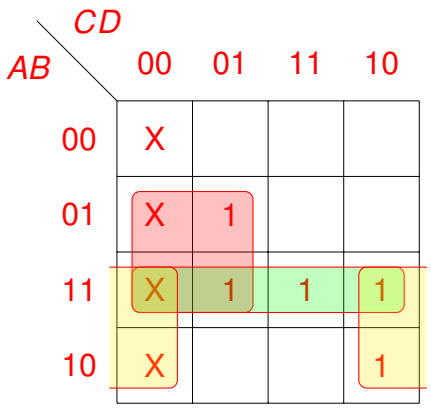
(a) Draw the truth table for **E**, **F**, **G**, and **H**.

AB	CD	EF	GH
00	00	xx	xx
00	01	00	00
00	10	00	00
00	11	00	00
01	00	xx	xx
01	01	01	00
01	10	00	01
01	11	00	01
10	00	xx	xx
10	01	10	00
10	10	01	00
10	11	00	10
11	00	xx	xx
11	01	11	00
11	10	01	01
11	11	01	00

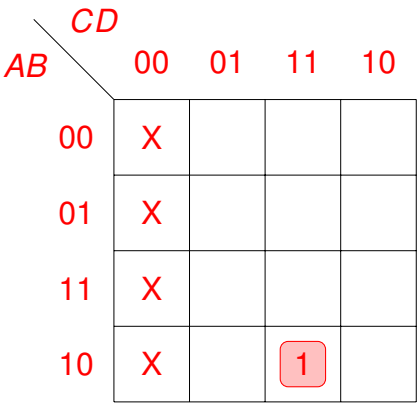
(b) Minimize the four output functions.

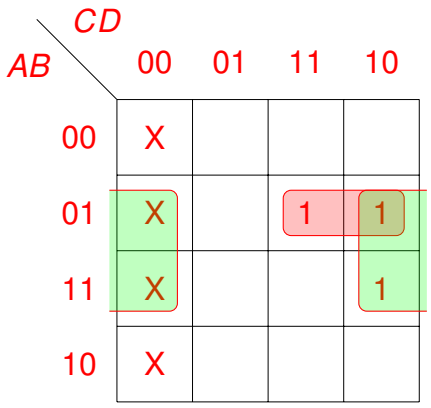


$E = A\overline{C}$



$F = B\overline{C} + AB + A\overline{D}$





$G = A\overline{B}CD$

$H = B\overline{D} + \overline{A}BC$

(c) and (d) The code is posted to Canvas. Note the testbench (you did write a testbench, right?) which instantiates both modules and compares the outputs of each.