G51CSA Handout One: Logic Equations

Steven R. Bagley

Introduction

In lecture four, we started to look at how we can take a truth table describing the input and output for a logic system and convert it into a series of logic equations, which could then be simplified, that could eventually be implemented as a circuit. Unfortunately, time ran out and it wasn't possible to complete the examples. These are now presented in full in this handout.

Each example will take the same form, first there will be a description of the problem. This is followed by a truth table for that problem. Then we convert that truth table into a series of logic equations for each output based on the truth table in long form. These equations are then simplified using the rules of Boolean algebera outlined in the lecture to reduce the number of gates needed. Finally, we look for common sub-equations between all the outputs and extract these out as separate equations to reduce the number of gates used in the final circuit (the output of the sub-equation can be split and sent to wherever that value is needed).

Please note, that we will be using the terms *true*, *high*, *1* and *logic one* to signify a true value, and *false, low*, *0,* and *logic zero* to represent a false value interchangably.

1. 2-to-4 decoder

1.1. Description

This logic circuit is simple, it takes in two inputs, I_1 and I_0 and converts the binary number encoded to one of four output singals: O_0, O_1, O_2 and O_3. These outputs are only high (at logic one) when the input contains the corresponding binary number, so O_0 is high when the input is `00` (0), O_1 is high when the input is `01` (1), and so on until O_3 is high when the input is `11` (3). For any other input, the output pin is logic zero.

Circuits similar to this are often used to decoded the address bus inside a computer with the output pins being used to enable different parts of the computer system.

1.2. Truth Table

I_1	I_0	O_0	O_1	O_2	O_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

1.3. Equations

The key to building up these equations is to first build up a logic equation for each combination of inputs in the truth table that is true when the inputs have that value (and only those values). This is usually done by **and**-ing together a combination of the input and inverted input signals. So for first row in the truth table above, the equation would be:

$$\overline{I_1} \cdot \overline{I_0}$$

Since this is only true when both I_0 and I_1 are zero. The equations for the rest of the entries are:

$$\overline{I_1} \cdot I_0$$

$$I_1 \cdot \overline{I_0}$$

$$I_1 \cdot I_0$$

With these equations, we can now to start to build up the equations for each output. To do this, we just take the equation for each row's input (as defined above) and **and** it with the output value for the output. We then **or** all of these equations together to produce a single equation for each output. Doing this for the first output, O_0 gives us:

$$O_0 = 1 \cdot \left(\overline{I_1} \cdot \overline{I_0} \right) + 0 \cdot \left(\overline{I_1} \cdot I_0 \right) + 0 \cdot \left(I_1 \cdot \overline{I_0} \right) + 0 \cdot \left(I_1 \cdot I_0 \right)$$

This works because when the input isn't met the equation for the input evaluates to false and so the value anded with it also evaluates to false. The ors in the equation then combine each of the terms. The equations for the other outputs can be built up in the same way, thus:

$$O_1 = 0 \cdot \left(\overline{I_1} \cdot \overline{I_0} \right) + 1 \cdot \left(\overline{I_1} \cdot I_0 \right) + 0 \cdot \left(I_1 \cdot \overline{I_0} \right) + 0 \cdot \left(I_1 \cdot I_0 \right)$$

$$O_2 = 0 \cdot \left(\overline{I_1} \cdot \overline{I_0} \right) + 0 \cdot \left(\overline{I_1} \cdot I_0 \right) + 1 \cdot \left(I_1 \cdot \overline{I_0} \right) + 0 \cdot \left(I_1 \cdot I_0 \right)$$

$$O_3 = 0 \cdot \left(\overline{I_1} \cdot \overline{I_0} \right) + 0 \cdot \left(\overline{I_1} \cdot I_0 \right) + 0 \cdot \left(I_1 \cdot \overline{I_0} \right) + 1 \cdot \left(I_1 \cdot I_0 \right)$$

Note how the equations for each output are very similar, the only change is the value anded with each of the input equations. It might worthwhile working through a couple of examples with real values assigned to the inputs to prove to yourself that the correct output values are reached.

1.4. Simplify

In the previous step, we developed an equation for each output signal from the circuit. These equations could be built using logic gates, but each output would make use of 15 gates. However, we can simplify these equations by applying some of the boolean alegbra rules that were discussed in lecture four. The first rule that can be applied to the above equations is:

$$A \cdot 0 = 0$$

That is anything anded with false is always false. Our equations have a lot of terms were we anded things with false (0) and so these can immediately replaced with false in the equation, thus:

$$O_0 = 1 \cdot \left(\overline{I_1} \cdot \overline{I_0} \right) + 0 + 0 + 0$$

$$O_1 = 0 + 1 \cdot \left(\overline{I_1} \cdot I_0 \right) + 0 + 0$$

$$O_2 = 0 + 0 + 1 \cdot \left(I_1 \cdot \overline{I_0} \right) + 0$$

$$O_3 = 0 + 0 + 0 + 1 \cdot \left(I_1 \cdot I_0 \right)$$

Another rule that can easily be applied to these equations is:

$$A \cdot 1 = A$$

In other words, a value anded with true (1) always gives the same value back. This allows us to replace all the input equation anded with true terms with just the input equation:

$$O_0 = \left(\overline{I_1} \cdot \overline{I_0} \right) + 0 + 0 + 0$$

$$O_1 = 0 + \left(\overline{I_1} \cdot I_0 \right) + 0 + 0$$

$$O_2 = 0 + 0 + \left(I_1 \cdot \overline{I_0} \right) + 0$$

$$O_3 = 0 + 0 + 0 + \left(I_1 \cdot I_0 \right)$$

We can also apply similar rules for the **or**s in the equations, namely these rule:

$$A+0 = A \qquad A+1 = 1$$

giving:

$$O_0 = \left(\overline{I_1} \cdot \overline{I_0} \right)$$

$$O_1 = \left(\overline{I_1} \cdot I_0 \right)$$

$$O_2 = \left(I_1 \cdot \overline{I_0} \right)$$

$$O_3 = \left(I_1 \cdot I_0 \right)$$

It is debatable with these set of equations whether you'd want to remove common sub-equations (such as $\overline{I_0}$ and $\overline{I_1}$), but this could be done.

1.5. Extension

You might like to work through the same process yourself and consider how you could develop a 3-to-8 decoder, this would have three inputs (I_2, I_1, I_0) and eight outputs (O_0 — O_7). With this variant, you may well want to extract some of the common sub-equations that might appear (such as $\overline{I_2} \cdot \overline{I_1}$).

2. 2-bit 3 to 1 Multiplexer

Another circuit often used in a computer is a multiplexer. This takes in a series of inputs (A_0, A_1, B_0, B_1, C_0, C_1) and switches which signal goes to the output ($O0$,O1) based on another input (S_0, S_1, S_2) such that when S_0 is high, the output is input A_0, A_1, when S_1 is high the output is from input *B*, while S_2 selects input *C*.

Note that we haven't described here what happens if, for example, both S_1 and S_2 are high. This may not be a problem, if you can guarentee the input won't let that happen (e.g if it was fed from the output of the 2-to-4 decoder described above).

In this example, the various stages described above are repeated, but they won't be elaborated upon in as much detail.

2.1. Truth Table

S_0	S_1	S_2	O_0	O_1
1	0	0	A_0	A_1
0	1	0	B_0	B_1
0	0	1	C_0	C_1

Note that we can simplify the writing of the truth table here by recognising that the value of O_0 and O_1 only depend on S_0 — S_2. We know that when S_0 is true, O_0 must contain the value in A_0 (whether it is 0 or 1) and so we can just pop that in the truth table — it saves us having to write out and consider 48 entries!

2.2. Equations

First, we work at equations that are true for each row based on all three inputs that matter (S_0 — S_2). These are:

$$S_0 \cdot \overline{S_1} \cdot \overline{S_2}$$
$$\overline{S_0} \cdot S_1 \cdot \overline{S_2}$$
$$\overline{S_0} \cdot \overline{S_1} \cdot S_2$$

From these, we can work out equations for O_0 and O_1 in the same manner as the previous example.

$$O_0 = A_0 \cdot \left(S_0 \cdot \overline{S_1} \cdot \overline{S_2} \right) + B_0 \cdot \left(\overline{S_0} \cdot S_1 \cdot \overline{S_2} \right) + C_0 \cdot \left(\overline{S_0} \cdot \overline{S_1} \cdot S_2 \right)$$
$$O_1 = A_1 \cdot \left(S_0 \cdot \overline{S_1} \cdot \overline{S_2} \right) + B_1 \cdot \left(\overline{S_0} \cdot S_1 \cdot \overline{S_2} \right) + C_1 \cdot \left(\overline{S_0} \cdot \overline{S_1} \cdot S_2 \right)$$
$$O_2 = A_2 \cdot \left(S_0 \cdot \overline{S_1} \cdot \overline{S_2} \right) + B_2 \cdot \left(\overline{S_0} \cdot S_1 \cdot \overline{S_2} \right) + C_2 \cdot \left(\overline{S_0} \cdot \overline{S_1} \cdot S_2 \right)$$

2.3. Simplify

With the equations given above there isn't much that needs to be done to simplify these rules. However, we can extract some common sub-equations from the equations like this:

$$X = S_0 \cdot \overline{S_1} \cdot \overline{S_2}$$
$$Y = \overline{S_0} \cdot S_1 \cdot \overline{S_2}$$
$$Z = \overline{S_0} \cdot \overline{S_1} \cdot S_2$$

giving:

$$O_0 = A_0 \cdot X + B_0 \cdot Y + C_0 \cdot Z$$
$$O_1 = A_1 \cdot X + B_1 \cdot Y + C_1 \cdot Z$$
$$O_2 = A_2 \cdot X + B_2 \cdot Y + C_2 \cdot Z$$

2.4. Extension

If we assume that the only values we will ever see on S_0 — S_2 are the ones presented in the truth table then we can simplify the equations further to:

$$O_0 = A_0 \cdot S_0 + B_0 \cdot S_1 + C_0 \cdot S_2$$
$$O_1 = A_1 \cdot S_0 + B_1 \cdot S_1 + C_1 \cdot S_2$$
$$O_2 = A_2 \cdot S_0 + B_2 \cdot S_1 + C_2 \cdot S_2$$

Consider why we can do this, and also what the result would be if an 'invalid' input was presented to the resulting circuit.