COMPUTER ORGANIZATION AND ARCHITECTURE

Lecture 6

Digital Logic

Dr Victoria Mukami

INTRODUCTION

This lecture is an introduction to the fundamental building blocks of any logic circuit. We will review the formation of integrated circuits, what is used to develop them, and their purpose or role. We will then do an introduction to truth tables and logic gates. Finally, we will review various digital logic circuits and will be able to solve the circuits to understand the type of output each would produce.

Learning objectives

By the end of this topic, you should be able to:

1. Understand the role of logic gates in an integrated circuit
2. Understand how a logic gate works
3. Solve various digital logic circuits with their truth table.

OVERVIEW

We learnt during the second lecture, specifically during the third generation of computers that integrated circuits were invented and they consisted of hundreds of transistors. These circuits were used to aid in the processing capability of a computer. Within the fourth generation, these circuits were able to work hand in hand with the microprocessor to make computers faster. These integrated circuits were in the form of a chipset. There are fundamental building blocks of this chipsets known as logic gates [2].

The computer on the other hand uses binary numbers for all its needs. Any data the computer handles is normally changed to binary. Binary is a form of Boolean instruction which accepts one or the other input. This is like an on/off state. For a processor to be able to handle such calculations or instructions, it relies on logic gates. Logic gates are electronic components that allow or that manipulate input into specific output. Throughout this lecture we will review the role of logic gates and will review a logic circuit that can add numbers. Before we look at logic gates, we need to understand truth tables.

TRUTH TABLES

Truth tables help us to comprehend the actions of a logic gate. A truth table is setup in a way that it can show the inputs of a logic gate, what happens to the input and the resulting output. For a gate, the inputs are mainly one or two inputs depending on the number of inputs in the gate and the resulting output is displayed on the right. If I can take you back to lecture 5 where we worked on the octal and hexadecimal table. One area that came up during creation of the table was that the number of digits was determined by the base. The same is true for truth tables. To create a truth table will depend on the number of inputs. Remember the inputs are either a 0 or a 1. For instance, a gate with one input would have 2^1 different combinations. The inputs would be demarked by an appropriate letter. In the case of Table 1, the input is a letter A. The output would be dependent on the gate.

Table 1: One input truth table

A	Output
1	
0	

For a gate with 2 inputs, the number of combinations would be determined by 2^2. Assuming a gate had inputs A and B then the number of combinations is similar to Table 2. Arrangement of the inputs should be like the creation of the octal table. Where the topmost is the smallest number. In Table 2, letter B would have alternated 0 and 1's while A would have two 0's then two 1's. Again, the output would be determined by the gate.

Table 2: Two inputs truth table

A	B	Output
0	0	
0	1	
1	0	
1	1	

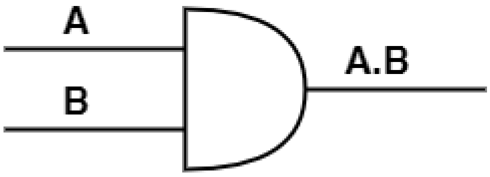
The formula for calculating the number of combinations will be determined by the number of input (x) as a power of 2. Therefore 3 inputs = 2^3 and 5 inputs would be 2^5.

LOGIC GATES

As defined on the overview, logic gates are the fundamental blocks of any circuit. There are 7 logic gates, however, we will only review 5 of the logic gates together with their truth tables.

AND Gate

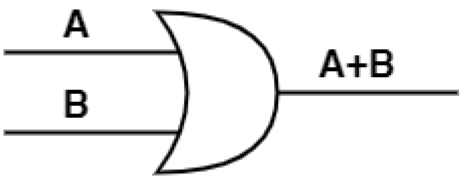
The AND gate gives an output of a high or 1 when ALL its inputs are high or 1. The output of the AND gate, assuming the inputs are A and B are shown as A.B. Sometimes the output would be shown as AB.



A	B	AB
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

OR Gate

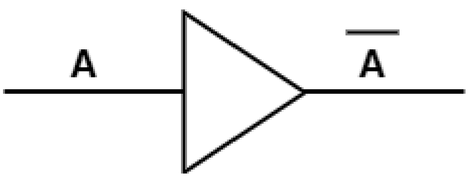
The OR gate gives an output of a high or 1 when ANY of its inputs are high or 1. The output of the OR gate, assuming the inputs are A and B are shown as A+B.



A	B	A + B
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

NOT Gate

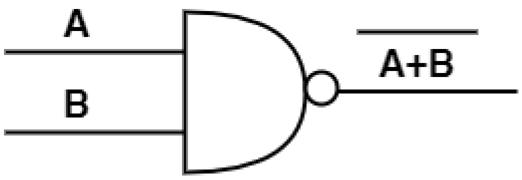
The NOT gate is known as the inverter. It inverts any input to the reverse. For instance, a 0 input would give a 1 output. The output of the OR gate, assuming the input is an A would be shown as $\overline{A}$ or as A' and is read as NOT A.



A	$\overline{A}$
0	**1**
1	**0**

NAND Gate

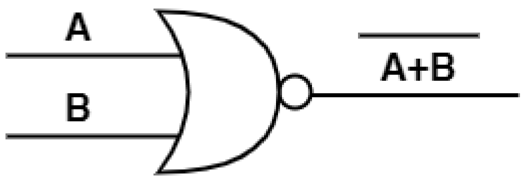
The NAND gate or NOT AND gate is a reverse of the AND gate. Inputs would first go through the AND part then get inverted. The NAND gate gives an output of a high or 1 when ANY of its inputs are low or 0. The output of the NAND gate, assuming the inputs are A and B are shown as $\overline{A.B}$. Sometimes the output would be shown as $\overline{AB}$.



A	B	$\overline{\mathbf{A.B}}$
0	0	**1**
0	1	**1**
1	0	**1**
1	1	**0**

NOR Gate

The NOR gate or NOT OR gate is a reverse of the OR gate. Inputs would first go through the OR part then get inverted. The NOR gate gives an output of a low or 0 when ANY of its inputs are high or 1. The output of the NOR gate, assuming the inputs are A and B are shown as $\overline{A+B}$.

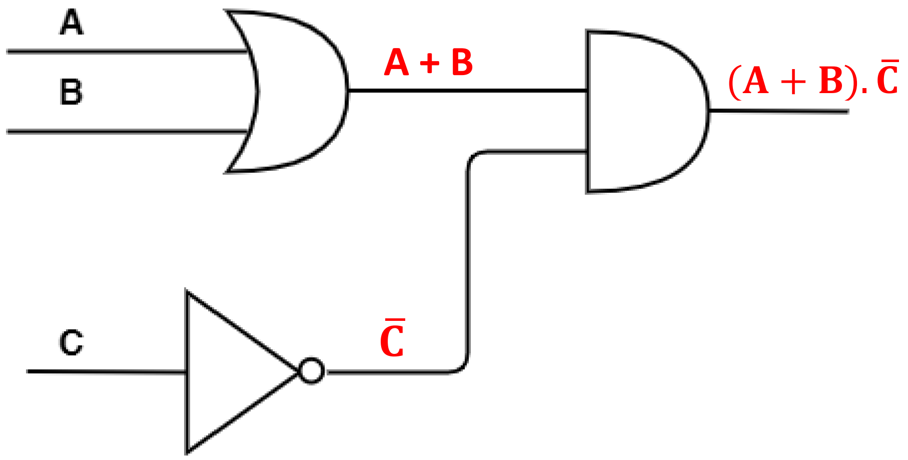


A	B	$\overline{\mathbf{A+B}}$
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**0**

LOGIC CIRCUIT

Based on the Logic Gates, Logic circuits are created whose aim is to perform various functions such as addition. A logic circuit is presented below.

Example 1



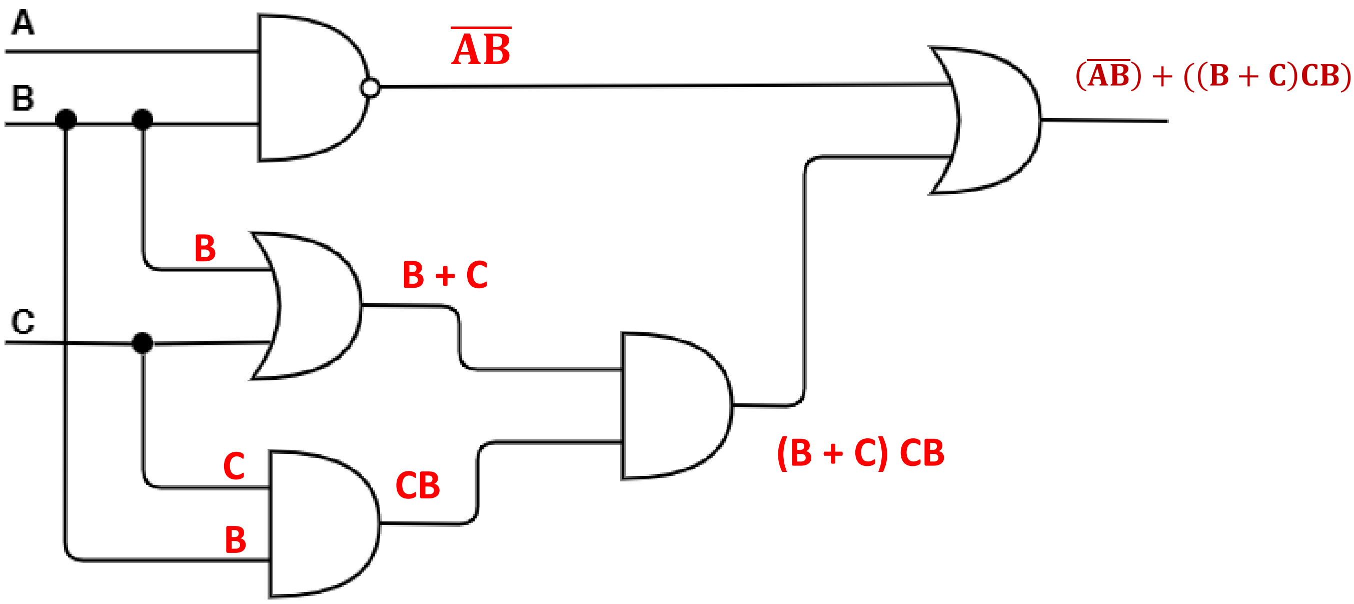
Before working on the truth table, the first thing would be to fill in the outputs and inputs of each gate. We have three gates, OR, NOT and AND. There are additionally three inputs. The truth table would have three inputs therefore the combinations would be 2^3 which is 8.

A	B	C	$A + B$	$\bar{C}$	$(A + B).\bar{C}$
0	0	0	0	1	0
0	0	1	0	0	0
0	1	0	1	1	1
0	1	1	1	0	0
1	0	0	1	1	1
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	0	0

Now remember the rules of the different tables and use those to fill in the outputs. Therefore, giving our final truth table output.

Example 2

Let us look at the following circuit:

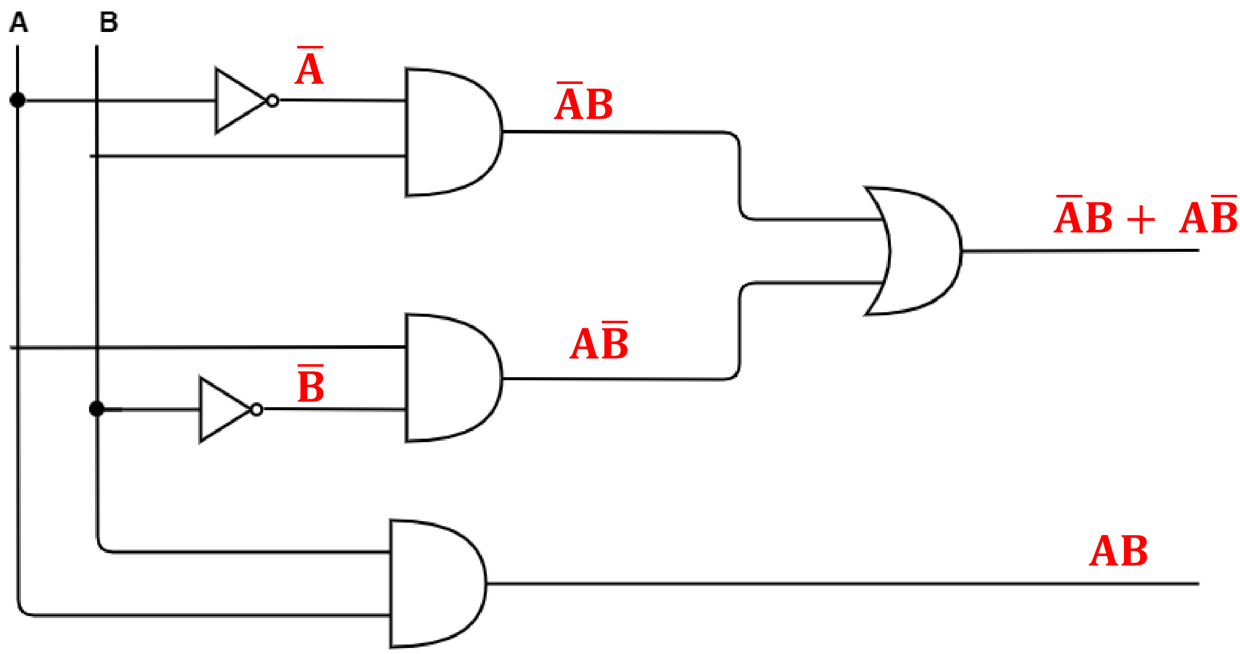


Now let us see how the truth table would look like.

A	B	C	AB	$\overline{AB}$	B+C	CB	(B+C) CB	$(\overline{AB}) + ((B + C)CB)$
0	0	0	0	1	0	0	0	1
0	0	1	0	1	1	0	0	1
0	1	0	0	1	1	0	0	1
0	1	1	0	1	1	1	1	1
1	0	0	0	1	0	0	0	1
1	0	1	0	1	1	0	0	1
1	1	0	0	1	1	0	0	1
1	1	1	1	0	1	1	1	0

Example 3

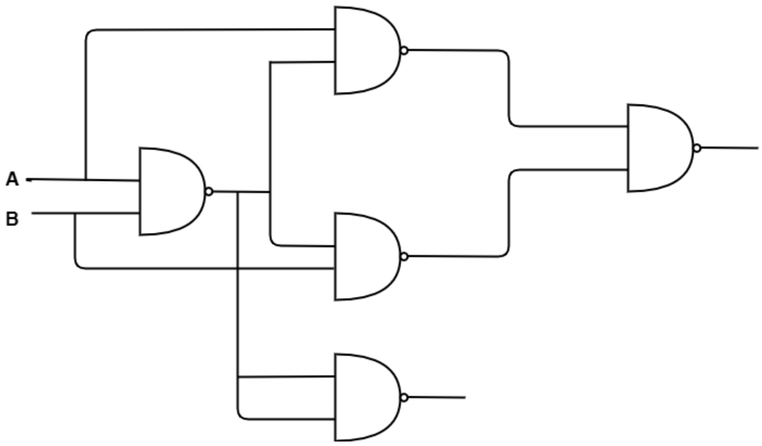
The following is a half adder example. A half adder is a circuit that can add two binary numbers. The half adder shows the result as well as the carry of the number. Generally, the half adder is created using the EX-OR gate, however since we do not work with the gate, it is easy to represent the half adder using AND, NOT and OR gates. NAND and NOR gates could also be used to represent the half adder.



Based on the truth table we can see that when you add the A and B column the sum corresponds to the binary addition rules, as well as the carry.

A	B	$\bar{A}$	$\bar{B}$	$\bar{A}B$	$A\bar{B}$	$\bar{A}B + A\bar{B}$ Sum	AB Carry
0	0	**1**	**1**	**0**	**0**	**0**	**0**
0	1	**1**	**0**	**1**	**0**	**1**	**0**
1	0	**0**	**1**	**0**	**1**	**1**	**0**
1	1	**0**	**0**	**0**	**0**	**0**	**1**

This same half adder circuit could be implemented using NAND gates as show.



SUMMARY

During this lecture, we have done a review of the logic gates. We first started with a review of truth table which show us how logic gates work. Next, we reviewed the basic logic gates and the related truth tables. Finally, we reviewed various logic circuits and their final equations. This then led to the final circuits.

DISCUSSION TOPIC

Since logic gates are the foundation blocks of any circuit, integrated circuit or chipset, what is the future like for any circuits when it comes to logic gates? Are computers still actively using logic gates? Are there special precautions when designing circuits with logic gates?

REFERENCES

[1] G. Shelly and M. Vermaat, Discovering Computers — Fundamentals: Your Interactive Guide to the Digital World. Boston, MA: Course Technology, 2012.

[2] W. Stallings, Computer Organization and Architecture Designing for Performance. Hoboken, NJ: Pearson Education, Inc, 2016.

[3] A. Evans, K. Martin and A. Poatsy, Technology in Action. New York, NY: Pearson, 2020

[4] D. Morley and C. Parker, Understanding Computers: Today and Tomorrow. Boston, MA: Course Technology, 2017.