

## Module-15

### Arithmetic Logic Unit (ALU)

1. Introduction
2. Arithmetic and Logic units
  - 2.1 1-bit ALU design
  - 2.2 n-bit ALU design
  - 2.3 ALU design process
3. ALU ICs
4. Summary

#### Learning Objectives:

1. To understand the importance of ALU
2. To design a simple arithmetic as well as logical unit
3. To learn various ALU designs techniques
4. To study of commercially available ALU ICs

## 1. Introduction

An arithmetic logic unit (ALU) is a digital system that performs arithmetic and logical operations. ALU is a fundamental building block of a central processing unit (CPU) or a microprocessor of a computer that performs arithmetic (Addition, Subtraction...) and logical operations (AND, OR, NOT ...). The concept of ALU was proposed by mathematician John Von Neumann. ALU is basically a combinational circuit that performs arithmetic and logical operations on binary integers. Whereas many applications demand processing of floating point numbers. Such unit is called floating point unit (FPU), which operates on floating point numbers. Some of the ALUs are also used for processing graphics. Such ALUs require a special unit called as floating point unit (FPU). Sometimes the FPU is used as partner chip called as “coprocessor”. When a floating point operations needs to be performed, the CPU sends it off to the coprocessor. Coprocessor after performing the operation sends back the result to CPU. Presently, there are modern CPUs with an internal FPU.

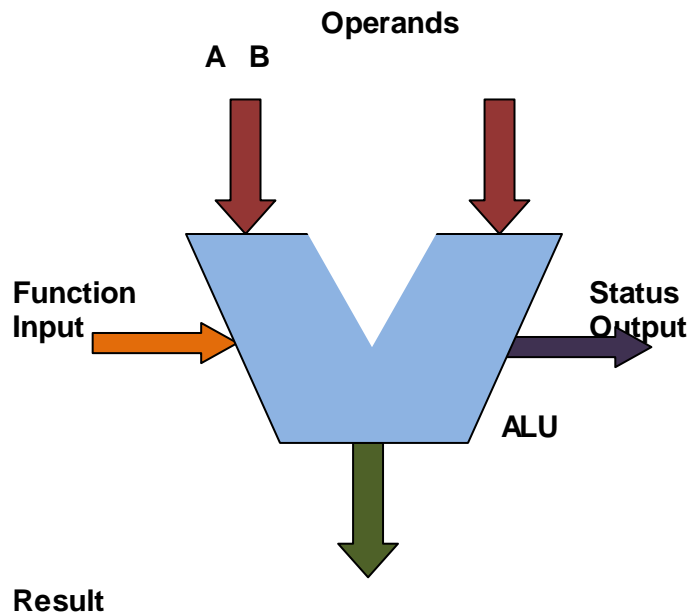
The operations of a microprocessor are performed by one or more ALUs. In a processor, ALU is divided into an AU and LU. The AU performs the arithmetic operations and the LU performs the logical operations. In this module, design of simple ALU will be described. Design of simple ALU begins with identifying the function implemented by AU as well as LU. Use of proper function selector of multiplexer helps the designer to implement simple ALU. Concept of 1-bit ALU can be further extended to 4-bit, 8-bit or n-bit ALU. Modern CPUs contain very powerful and complex ALUs. Different approaches to implement such ALUs will also be discussed. The module ends with the discussion of commercially available simple ALU's.

## 2. Arithmetic logic unit (ALU)

The arithmetic logic unit (ALU) is designed to be a part of computer CPU or a microprocessor for performing arithmetic and logic operations. In many cases, an ALU is supposed to handle either finite field arithmetic or with allowed range infinite field arithmetic. This indicates use of integer ALU. Present superscalar processors have a separate FPU (floating point unit) and integer ALU.

The CPU of the computer loads data from input registers to an ALU. The control unit of a processor tells ALU what of operation to perform on that data. After the operation control unit transfers result of ALU into output register.

Figure-1 indicates a typical logic symbol of an ALU. In this diagram, operands A and B are the two inputs of an ALU. F is another type of input indicating the arithmetic or logic function to be performed on the operands. Y indicates the result of the operation and S indicates status of the ALU result (overflow, negative, carry out...) after the operation.



**Figure -1: Logic symbol of ALU**

An ALU performs basic arithmetic and logical operations. The arithmetic operations usually implemented are addition, subtraction, multiplication, and division. Similarly, the logic operations are NOT, AND, OR etc. are normally implemented. Along with the result, a set of status flags are set which include Zero, Carry, Negative or overflow to indicate the status of the result. The arrow symbols indicates more number of bits are available at inputs as well as output. The ALU can handle integers of some data width , which decides the processing ability of the processor. For example 8 bit ALU handles 8-bit data at the inputs.

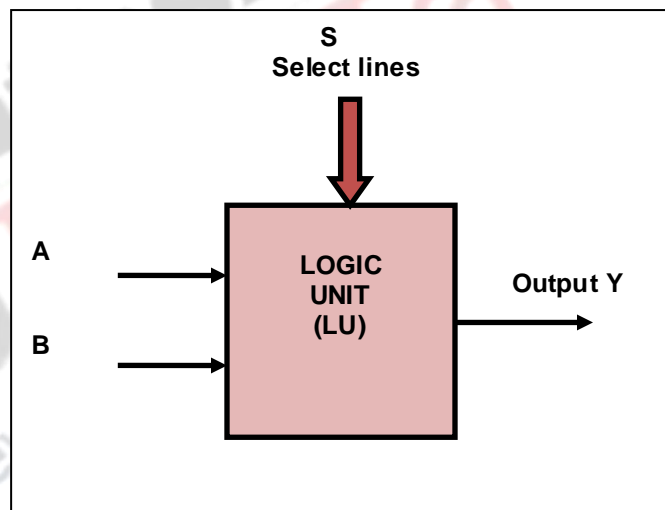
An ALU can be programmed to perform any series of complex arithmetic or logical calculations. The computational capacity of the ALU decides the power of the CPU of a computer.

## 2.1 1-bit ALU design

An ALU is the fundamental unit of any computing system. Understanding how an ALU is designed and how it works is essential to building any advanced logic circuits. Using this knowledge and experience, we can move on to designing more complex integrated circuits.

The goal is to design an n-bit ALU. Designing n-bit ALU is similar to the way we design n-bit adder. The design begins with building a 1-bit ALU version. This one bit ALU can be built from logic gates, adder and multiplexer. Such 1-bit ALU can be combined together to get n-bit ALU. Let us first build the logic unit first and then combine arithmetic block in it to make an ALU.

Let us assume that the unit can handle two 1 bit inputs to produce a required output based on the function selector line. Figure -2 indicates a simple block for 1-bit logic unit.



**Figure-2: A simple logic unit**

Actually there are relatively few basic arithmetic and logic operations needed.. Many operations can be implemented in terms of the basic ones. The basic building blocks of a simple logic unit are NOT gate, AND gate, OR gate, XOR gate and a multiplexer as shown in figure 3.

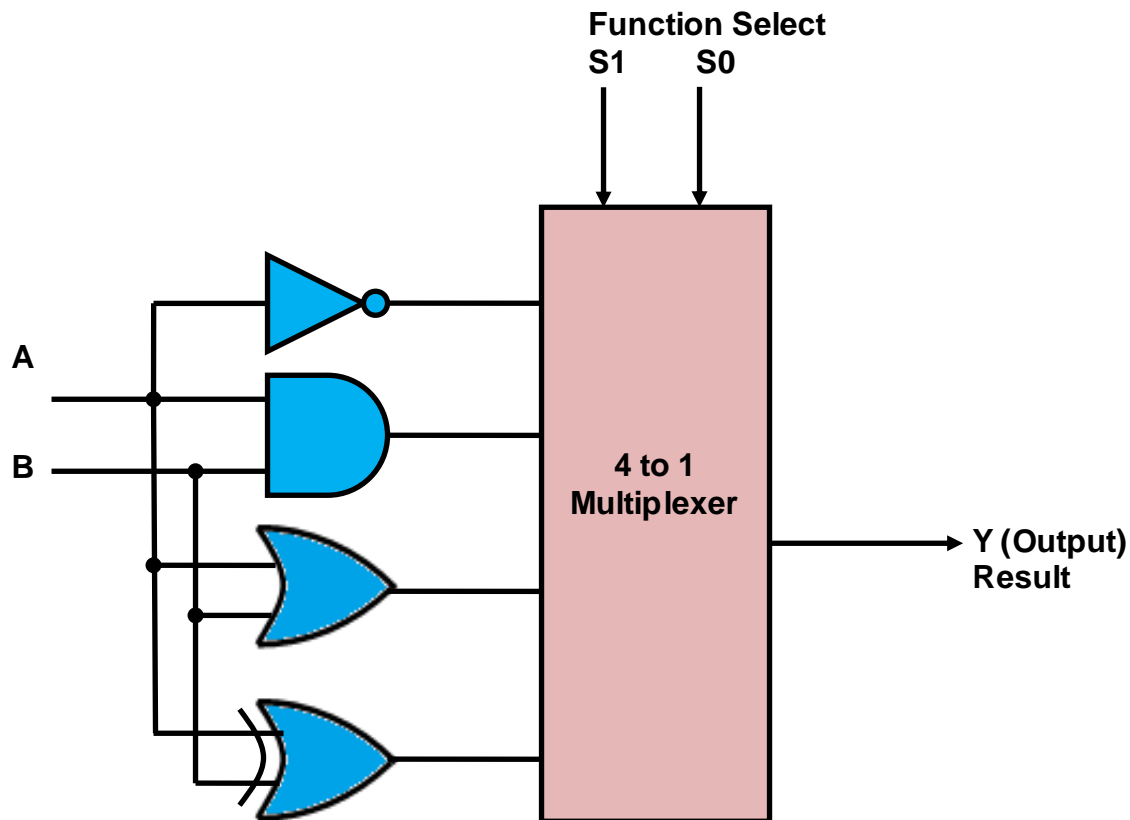


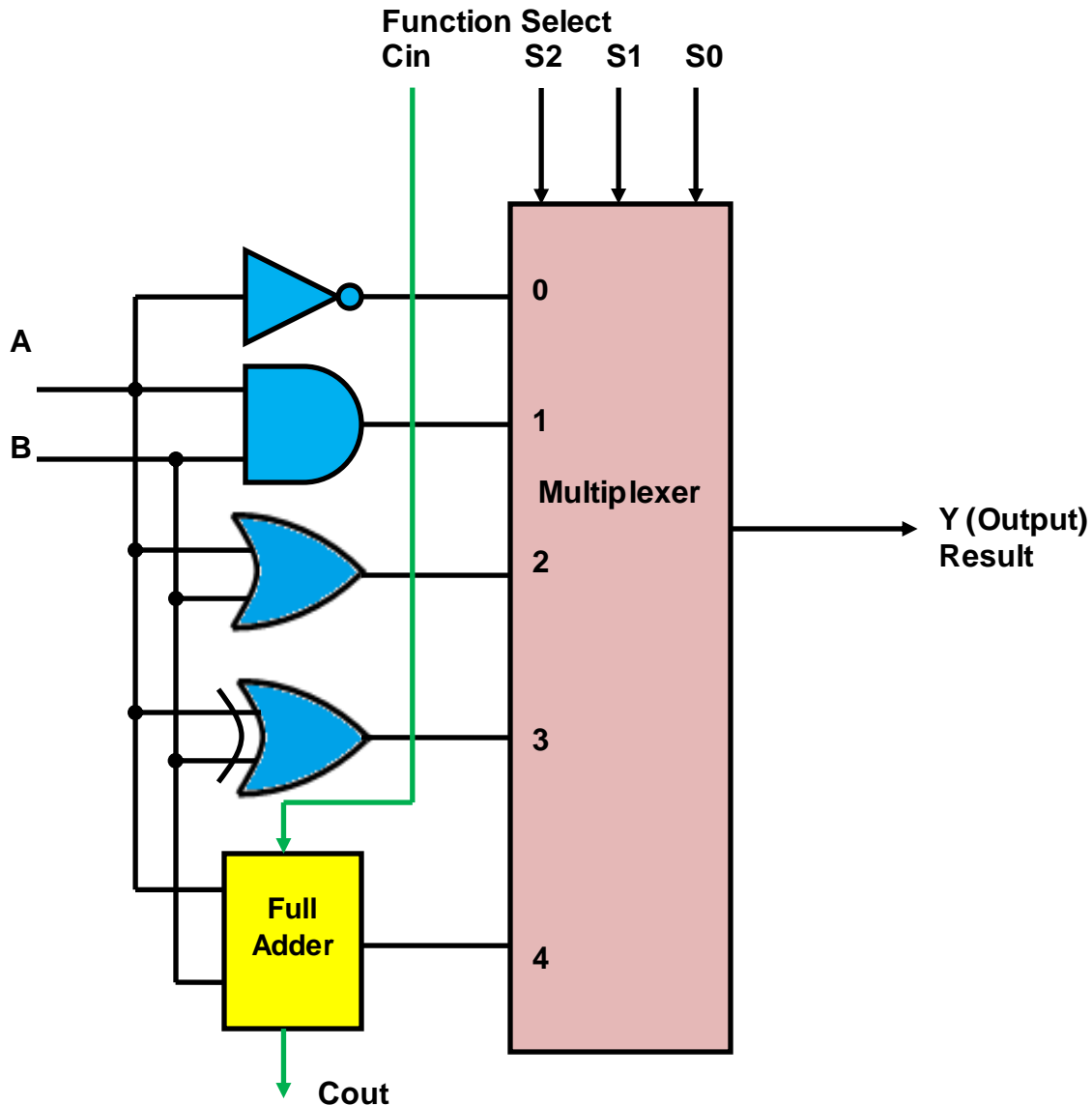
Figure -3: Logic diagram for 1-bit Logic Unit

The simple logic unit contains four basic logic gates to provide four basic logic operations like NOT (complement), AND, OR and XOR. To select one of four logic operations a selector is required. In this unit a simple 4 to 1 multiplexer is preferred for designing of logic unit. To select one of 4 inputs, there are two select inputs S1 and S0 as shown in figure 4. The function table assumes that both the data inputs A, B are present. The values of select lines decide the nature of operation to be performed on the operands.

S1	S0	Output	Operation
0	0	$Y=A'$	Complement
0	1	$Y=A \cdot B$	AND
1	0	$Y=A + B$	OR
1	1	$Y= A \oplus B$	XOR

Figure -4: Function table of Logic Unit

After designing the 1-bit logic unit, the next obvious step is to incorporate an adder block to logic unit. Addition of a full adder allows the unit to perform addition as well. Figure -5 shows indicates 1-bit logic unit with addition feature.



**Figure-5: 1-bit logic unit with addition**

To select one of four logic operations and addition, multiplexer with 5 inputs is required. In this unit a simple 8 to 1 multiplexer is preferred for designing of logic unit with adder. To select one of 5 inputs, there are three select inputs S2, S1 and S0 as shown in figure 5. The function table assumes that both the data inputs A, B are present. The full adders have three data inputs A, B and Ci with

sum and carry as outputs. The values of select lines decide the nature of operation to be performed on the operands. Function table as shown figure-6 indicates the correlation between the values of select inputs and the nature of operation i.e. arithmetic or logical operation.

S2	S1	S0	Output	Operation
0	0	0	$Y=A'$	Complement
0	0	1	$Y=A \cdot B$	AND
0	1	0	$Y=A + B$	OR
0	1	1	$Y= A \oplus B$	XOR
1	X	X	$Y= \text{Sum}(A,B)$ & C0	Addition

Figure-6: Function table of logic unit with addition

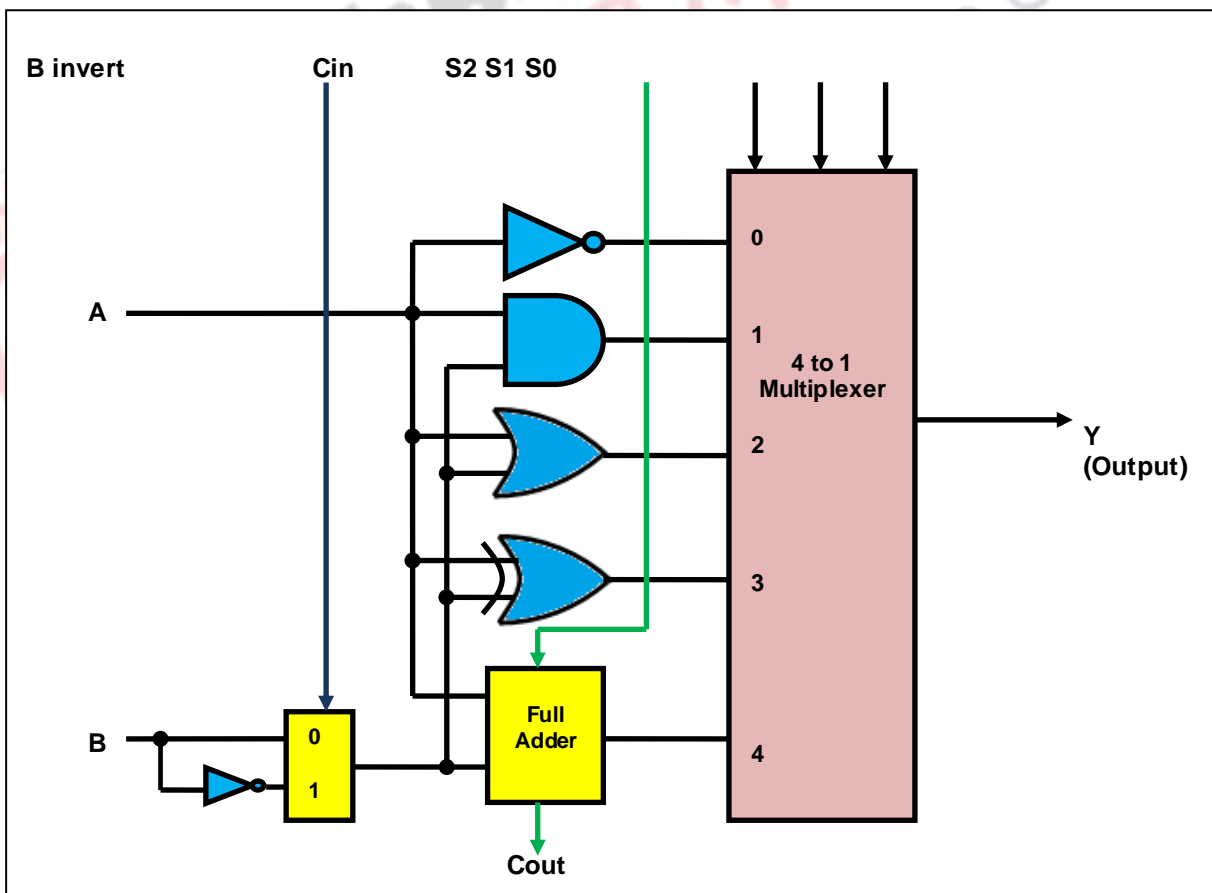


Figure-7: 1-bit Arithmetic and logic unit

So far we have discussed how to incorporate adder in an ALU. For performing subtraction, it is necessary to perform  $A-B$ . This subtraction can also be represented as an addition of  $A$  and  $B$ 's complement. By adding 1 through the carry input, it is possible to implement 2's complement addition as shown in figure 7. In this diagram, addition of one 2 to 1 multiplexer allows us to select either  $B$  input without or with complement. Binvert input value decides whether to perform addition or subtraction. It should be noted that the value of Binvert should be 0, to perform logical operation. Figure-8 indicates the function table for 1-bit ALU.

S2	S1	S0	Binvert	Cin	Output	Operation
0	0	0	0	0	$Y=A'$	Complement
0	0	1	0	0	$Y=A \cdot B$	AND
0	1	0	0	0	$Y=A + B$	OR
0	1	1	0	0	$Y= A \oplus B$	XOR
1	X	X	0	0	$Y= A+B$	Addition
1	X	X	0	1	$Y= A+B+Cin$	Add with carry
1	X	X	1	0	$Y= A -B$	Subtraction (1's compl. Addition)
1	X	X	1	1	$Y= A -B+cin$	2's complement addition

**Figure-8: Function table of 1-bit ALU**

The adder with multiplexer can help us to combine adder and subtractor circuit together in a single circuit. When binvert=0, the multiplexer will send  $B$  directly to the multiplexer which enables the addition. If Binvert=1 then inverted input will be taken and transferred to the adder which performs subtraction.

From the design of 1-bit ALU and function table, it is clear that it is possible to perform addition as well as subtraction through an ALU. Subtraction is implemented by setting initial carry as 1 which is normally 0 for addition. In short, there are 2 main changes in the system. First each  $B$  bit is inverted and initial carry is set to 1.



## 2.2n-bit ALU design

Once the complete 1-bit ALU is implemented using various combination block, we can extend the design to build n-bit ALU.

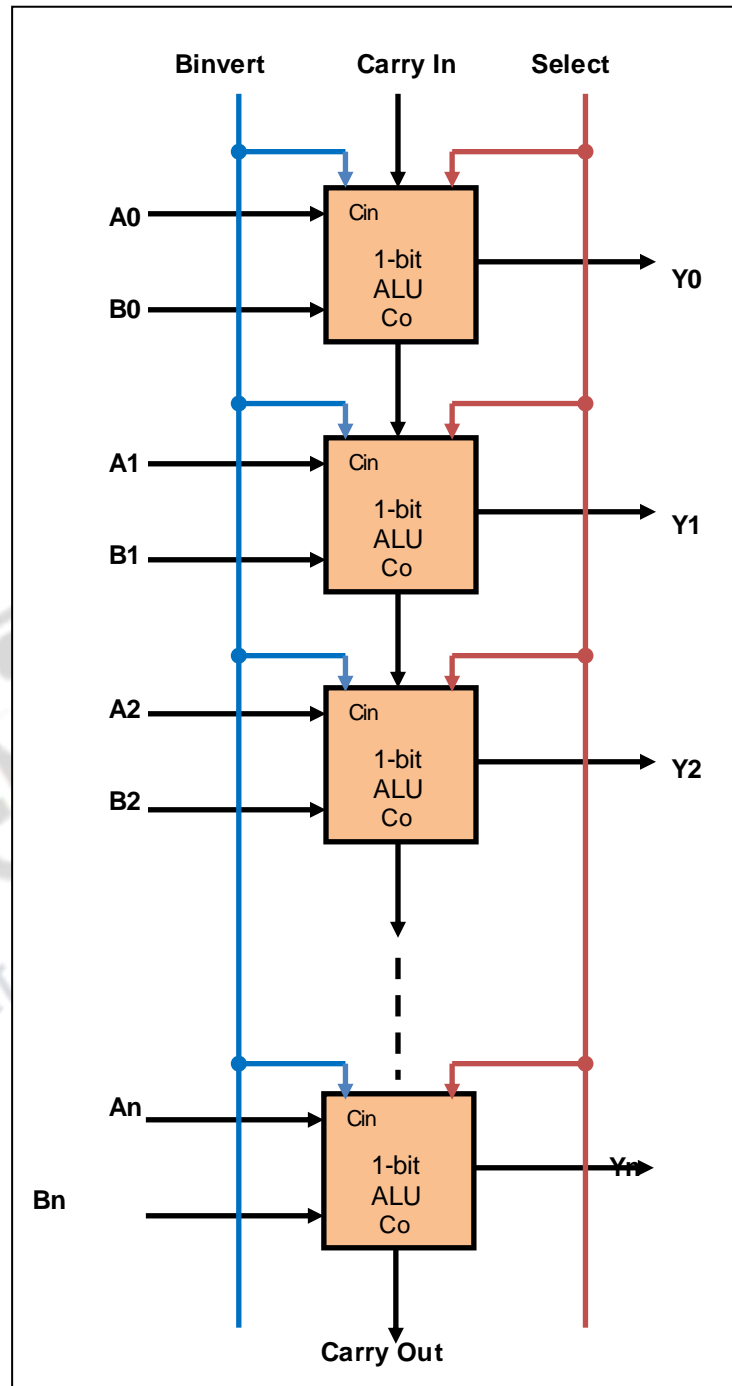


Figure -9: Implementation of n-bit ALU

Figure -9 indicate cascading of 1-bit ALUs to construct n-bit ALU. One can thus extend the concept to 4-bit, 8-bit, 16-bit or 32-bit ALU. The input carry is applied to the first 1-bit ALU and the final carry is obtained from the last 1-bit ALU. Select lines and the B invert lines are connected in common. Select lines can decide on the nature of operation like arithmetic or logic operation. The binary value given to select lines along with Binvert and carry in will decide final function output as n-bit arithmetic or logic output. The function table for the n-bit ALU can also be prepared on the similar lines of 1-bit ALU.

### 2.3 ALU design process

The ALU design process begins with the specifications of ALU.

1. Identify Arithmetic and Logical functions needs to be implemented.
2. Start with word problem or the function table.
3. Use divide and conquer approach i.e. Design arithmetic and logic unit separately.
4. For arithmetic unit map the function table to an adder & build the input logic.
5. For logic unit use basic logic gates and multiplexer for implementation.
6. Integrate arithmetic and logic unit.

#### Example: Design a 4 bit ALU.

Let us start with the functional specifications of the ALU. Figure -10 indicates some of the different possible arithmetic operations. Following table indicates the select inputs and the desired functional output.

S2	S1	S0	Output	Arithmetic operation
0	0	0	X	Transfer
0	0	1	X+1	Increment
0	1	0	X+Y	Add
0	1	1	X+Y+1	Add with carry
1	0	0	X+Y'	1's compl. Subtraction
1	0	1	X+Y'+1	2's compl. subtraction
1	1	0	X-1	Decrement
1	1	1	X	Transfer

**Figure-10: Functional specifications of the ALU**

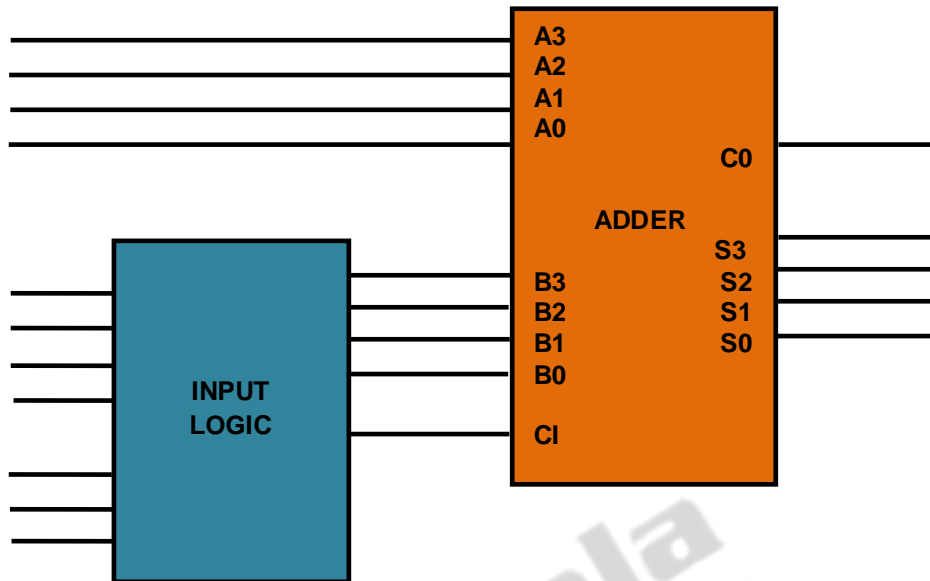
Let us now map the function table to an adder. Figure 11 indicates what the adder's inputs should be for each of our eight desired arithmetic operations.

S2	S1	B INPUT	Function Select		Expected arithmetic function	Required Adder inputs		
S2	S1	S0				A	B	Ci
0	0	0000						
0	1	Y						
1	0	Y'						
1	1	1111						
0	0	0	X		Transfer	X	0000	0
0	0	1	X+1		Increment	X	0000	1
0	1	0	X+Y		Add	X	Y	0
0	1	1	X+Y+1		Add with carry	X	Y	1
1	0	0	X+Y'		1's compl. Subtraction	X	Y'	0
1	0	1	X+Y'+1		2's compl. subtraction	X	Y'	1
1	1	0	X-1		Decrement	X	1111	0
1	1	1	X		Transfer	X	1111	1

**Figure-11: Logic and arithmetic functions**

In this table, the adder input  $C_i$  is always the same as function select code bit  $S_0$ . The input  $A$  is always connected to  $X$ . The value of input  $B$  depended on function select inputs  $S_2$  and  $S_1$ . The desired output of ALU depends on all these inputs.

As a next step let us build the input logic required for the  $B$ -input of an adder. Figure 12 indicates the desired inputs required for the input logic and the logic diagram of an ALU. Here, what is required is to compute input  $B$  given the arithmetic unit input  $Y$  and the functional select code.



**Figure -12: Input logic for Adder block**

The logic unit supports different logical functions on two multi-input X and Y producing the output  $G_i$ . The logic function table is shown in figure 13, indicates that there are four possible functions selected by logic select lines.

S2	S1	Output
0	0	$G_i = X_i \cdot Y_i$
0	1	$G_i = X_i + Y_i$
1	0	$G_i = X_i \oplus Y_i$
1	1	$G_i = X_i$

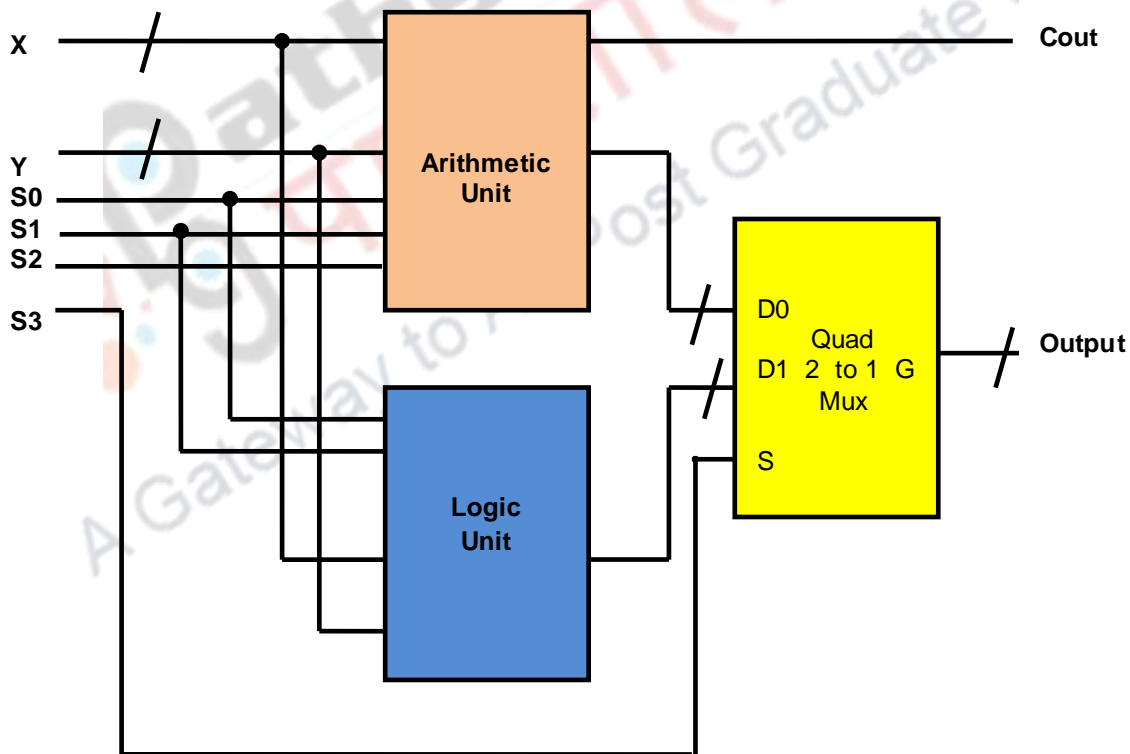
**Figure-13: Function table of Logic Unit**

This table can be implemented using multiplexers and few basic gates. It should be noted that for n-bit logic unit we need a four to one multiplexers.

Next step is to prepare combined the function tables of both arithmetic and logic functions together as shown in figure 14. Let us incorporate  $S_3=0$  for all arithmetic functions and logic functions when  $S_3=1$ . There are already three variables in the arithmetic function table.

Function select lines				Operation
S3	S2	S1	S0	
0	0	0	0	$G=X$
0	0	0	1	$G=X+1$
0	0	1	0	$G=X+Y$
0	0	1	1	$G=X+Y+1$
0	1	0	0	$G=X+Y'$
0	1	0	1	$G=X+Y'+1$
0	1	1	0	$G=X-1$
0	1	1	1	$G=X$
1	X	0	0	$G=X \text{ and } Y$
1	X	0	1	$G= X \text{ or } Y$
1	X	1	0	$G= X \text{ xor } Y$
1	X	1	1	$G=A'$

**Figure-14: Complete ALU function Select table**



**Figure -15 A Complete ALU block diagram**

The complete ALU diagram as shown in figure 15 integrates both AU and LU. AU performs arithmetic functions and LU performs logic functions on 4-bit  $X_i$  and  $Y_i$  decided by select inputs.  $S_3$  select input is connected to select line of quad 2 to 1 multiplexer which either

selects AU or LU. For 4 bit operands, four 2 to 1 multiplexer is used. AU can also generate a carry output.

### 3. ALU ICs

Arithmetic and logic unit is very popular device which is capable of performing different arithmetic and logical functions. Basically, ALU is the main part of central processing unit (CPU) of the computer. Let us study commercially available IC 74181. It is one of the popular and most widely used ALU. Figure 16 indicates the logic symbol of IC74181.

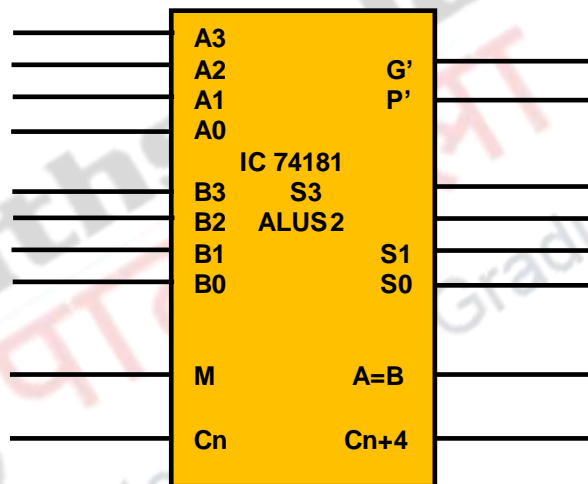


Figure 16: Functional diagram of IC 74181

IC 74181 is basically a 4 bit ALU which can perform all the possible 16 logic operations on two operands and a variety of arithmetic operations.

Features of IC74181:

- Arithmetic operating modes- Addition, subtraction, shift operand A one position, magnitude comparison and twelve other arithmetic operations
- Logic function modes – XOR, AND, OR, NOR, NAND, comparator, and ten other logic operations.
- Full Look-Ahead logic for high speed operations on Long word

#### Functional Description

The IC 74181 is an arithmetic logic unit (ALU). This ALU is capable of performing 16 binary arithmetic operations on two 4-bit words as shown in Table 17. These operations are selected by the four function select lines (S3,S2, S1,S0) and include addition, subtraction, decrement, and straight transfer.

Mode Select Inputs				Active LOW inputs and outputs		Active HIGH inputs and outputs	
S3	S2	S1	S0	Logic (M=H)	Arithmetic (M=L,Cn=L)	Logic (M=H)	Arithmetic (M=L, Cn=H)
0	0	0	0	$A'$	A minus 1	$A'$	A plus 1
0	0	0	1	$(AB)'$	AB minus 1	$(A+B)'$	(A+B) plus 1
0	0	1	0	$A'+B$	$AB'$ minus 1	$A'B$	$(A+B)'$ plus 1
0	0	1	1	1	Minus 1	0	Zero
0	1	0	0	$(A+B)'$	A plus $(A+B)'$	$(AB)'$	A plus $AB'$ plus 1
0	1	0	1	$B'$	AB plus $(A+B)'$	$B'$	$(A+B)$ plus $AB'$ plus 1
0	1	1	0	$(A\oplus B)'$	A minus B minus 1	$A\oplus B$	A minus B
0	1	1	1	$A+B'$	$A+B'$	$AB'$	$AB'$
1	0	0	0	$A'B$	A plus $(A+B)$	$A'+B$	A plus $AB$ plus 1
1	0	0	1	$A\oplus B$	A plus B	$(A\oplus B)'$	A plus B plus 1
1	0	1	0	B	$AB'$ plus $(A+B)$	B	$(A+B)'$ plus $AB$ plus 1
1	0	1	1	$A+B$	$A+B$	$AB$	$AB$
1	1	0	0	0	A plus $A^*$	1	A plus A plus 1
1	1	0	1	$AB'$	$AB$ plus A	$A+B'$	$(A+B)$ plus A plus 1
1	1	1	0	$AB$	$AB'$ plus A	$A+B$	$(A+B)'$ plus A plus 1
1	1	1	1	A	A	A	A




**Figure: 17: Arithmetic and logic operations in IC 74181**

The arithmetic and logic function can be selected by selecting mode, M. If M=1 then logic operations on the operands. Similarly, when M=0, the ALU performs the arithmetic operations. The operations on active low operands and active high operands are shown in figure 17. In addition

there are two additional lines carry in and carry out provided for cascading ‘n’ numbers of ALU. The devices provides full internal carry look ahead and provides for either ripple carry between devices  $C_{n+4}$  output or for carry look ahead between multiples ALU using signal carry propagate (P) and carry generate (G).

#### 4. Summary

- The arithmetic logic unit (ALU) is designed to be a part of computer CPU or a microprocessor for performing arithmetic and logic operations.
- ALU has two operands A and B inputs, some function inputs indicating the arithmetic or logic function to be performed on the operands. Along with the final outputs there are some outputs to indicate status of the ALU result (overflow, negative, carry out....) after the operation.
- One bit ALU can be built from logic gates, adder and multiplexer. Such 1-bit ALU can be combined together to get n-bit ALU. Let us first build the logic unit first and then combine arithmetic block in it to make an ALU.

		
<b>Development Team</b>		
<b>Principal Investigator:</b>	<b>Prof. A. D. Shaligram</b> Savitribai Phule Pune University, Pune	
<b>Paper Coordinator:</b>	<b>Dr. N. M. Kulkarni</b> Fergusson College, Pune	
<b>Content Writer:</b>	<b>Dr. N. M. Kulkarni</b> Fergusson College, Pune	
<b>Content Reviewer:</b>	<b>Prof. D. B. Gaikwad</b> Modern College, Pune	