4CS015

Lecture 4: Arithmetic Logic Units

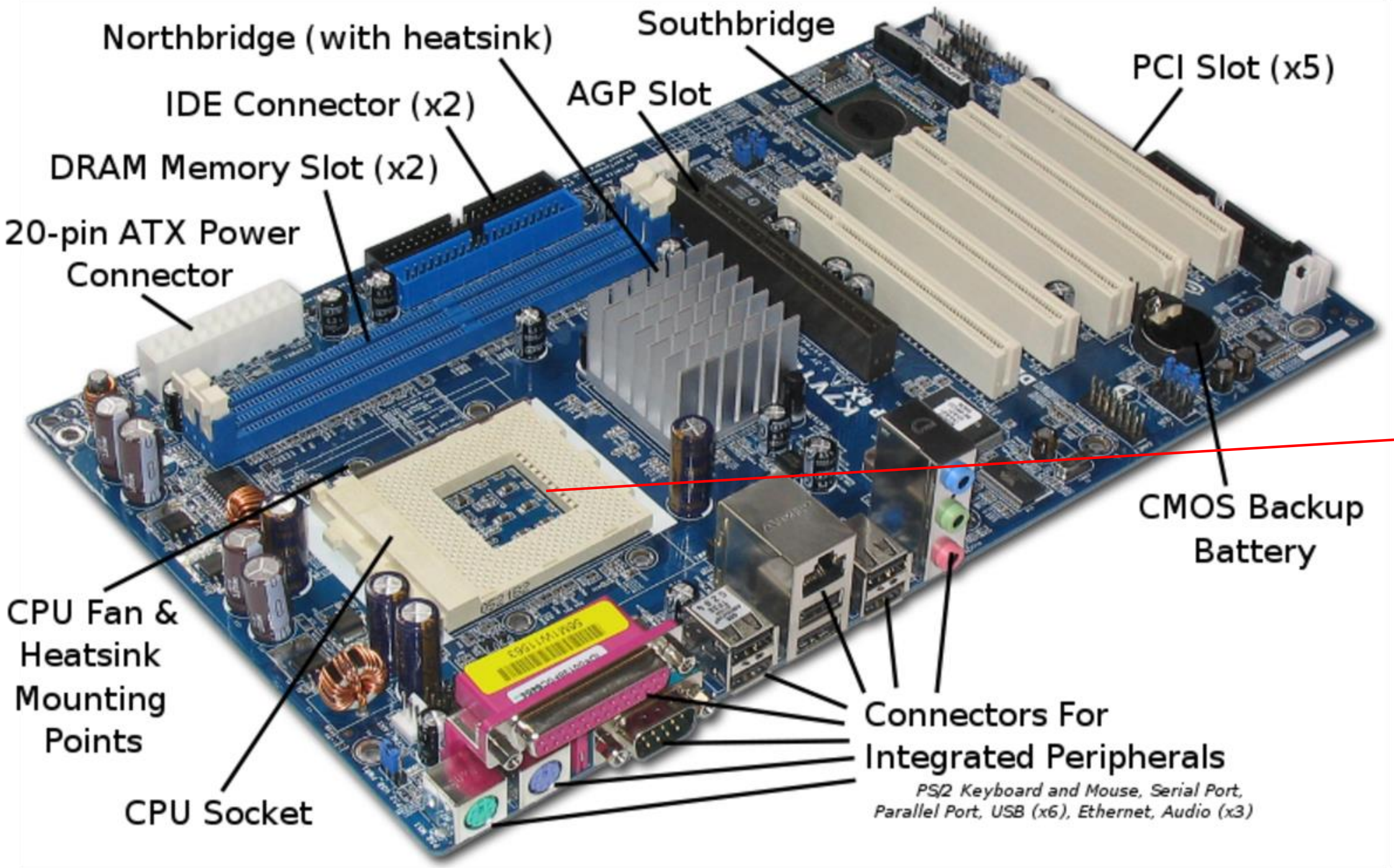
Prepared by: Uttam Acharya

1. Today

- What is an **ALU?**
  - **Function Unit**
  - **Instruction Decoder**
  - **Output Multiplexor**
- Turning our adder into a subtractor

2. What is ALU?

ALU



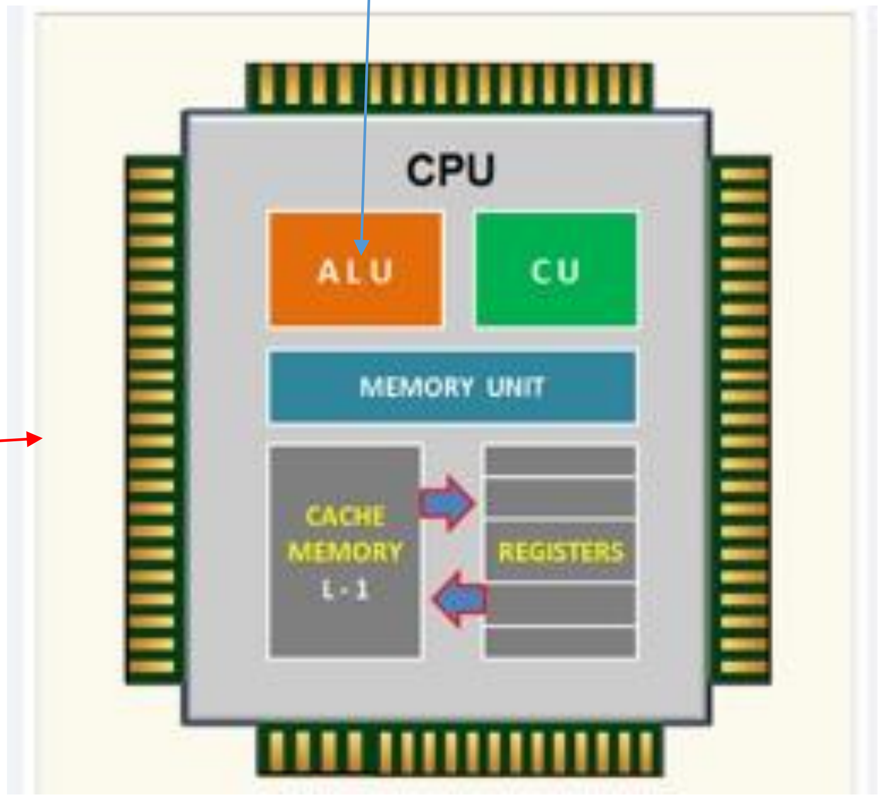
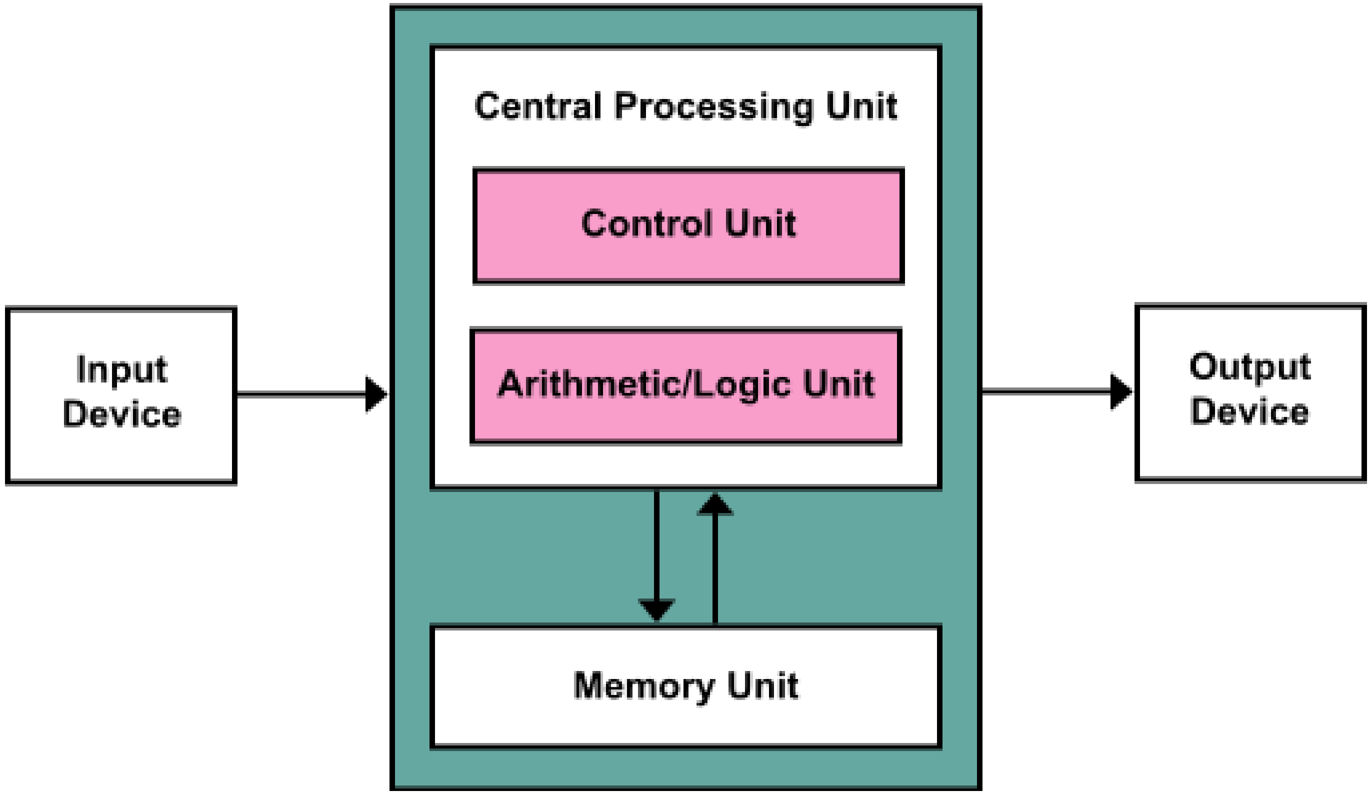


Fig: Mother Board

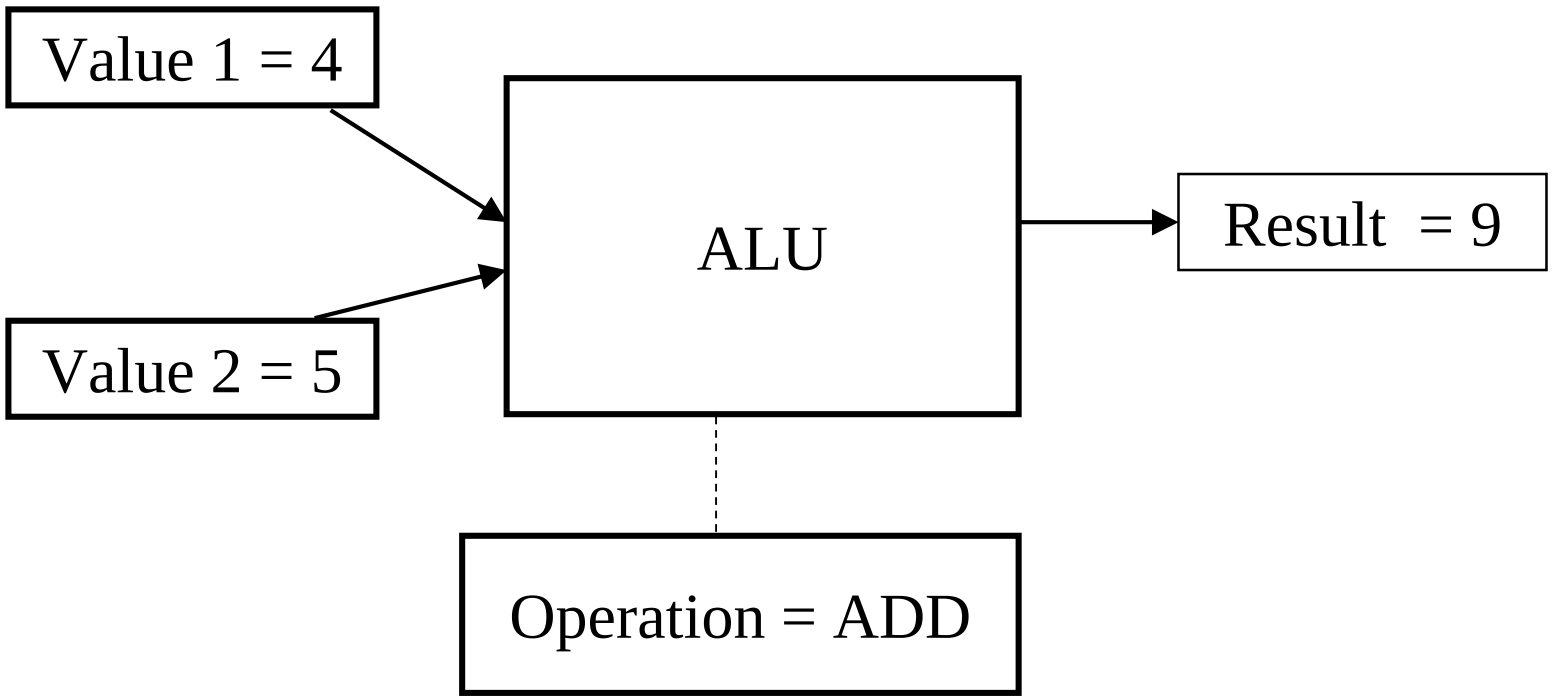
Fig: CPU

2. What is ALU?

- Performs set of arithmetic operations and set of logic operations
- Multi operation combinational logic circuit.
- It has a number of selection lines to select particular operation in the unit.

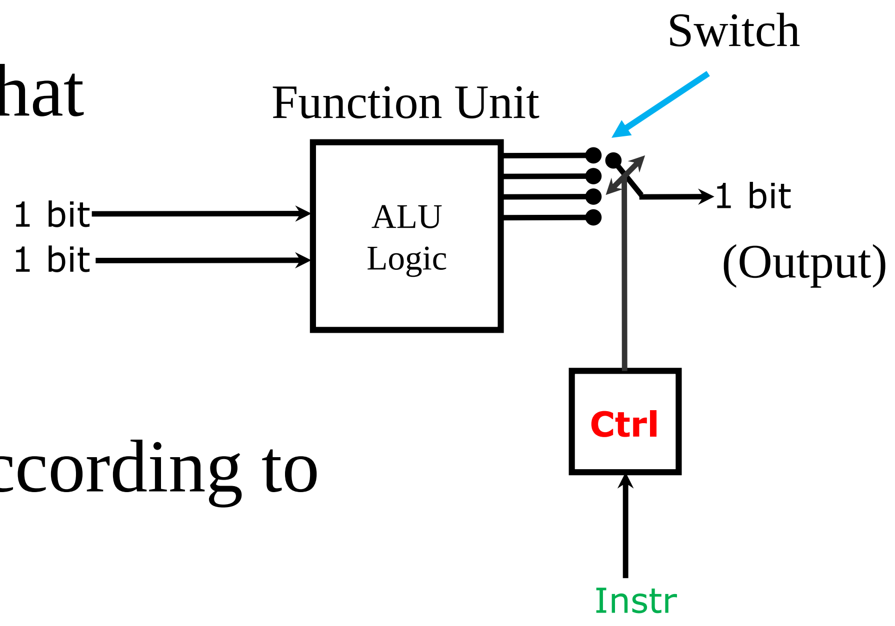


2.1 Arithmetic Logic Unit diagram



2.2 The ALU

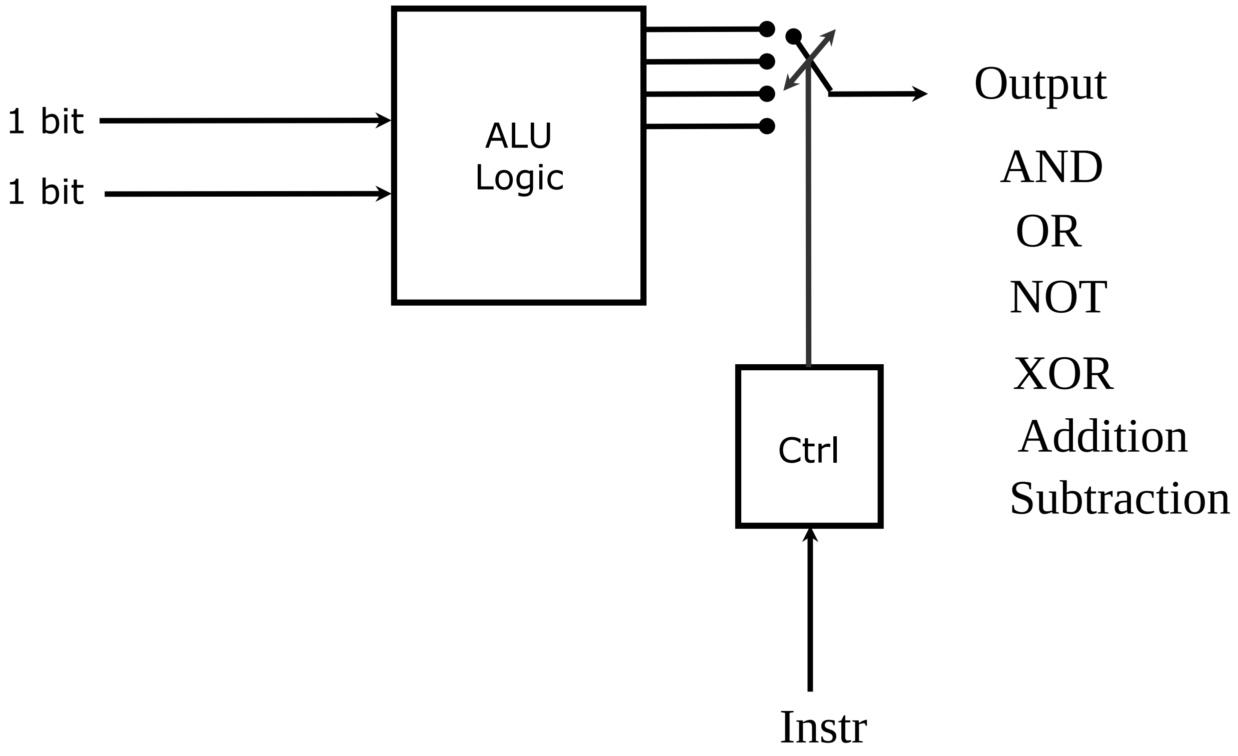
- A **function unit** containing the **logic blocks** that simultaneously carry out each **operation**



- A controller that selects the required output according to the instruction
- The instruction is a **binary ‘word’**
- A ‘**switch**’ that obeys the controller which connect the **output** to the **required function**

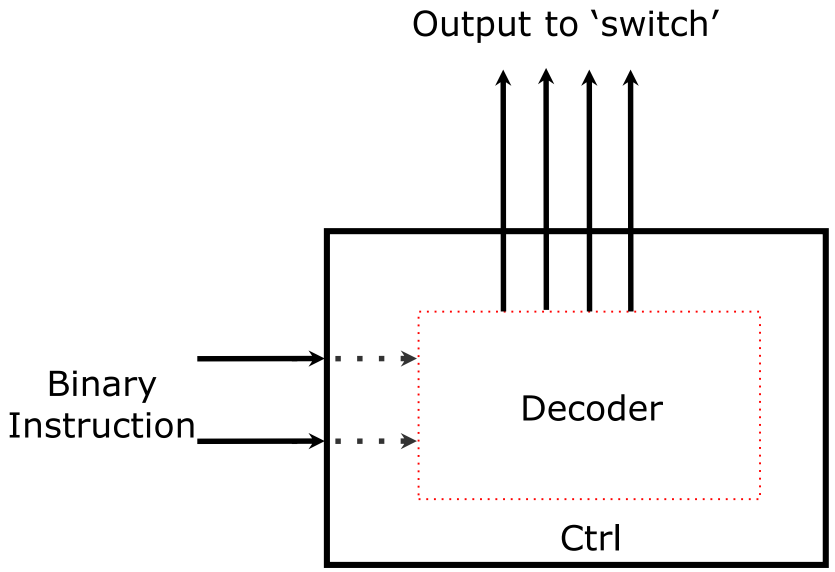
3.1 ALU Arithmetic and Logic Functions

- Output examples
  - AND
  - OR
  - NOT
  - XOR
  - Addition
- The controller "decodes" the instruction to select the required output



3.2 ALU Decoder & Output Selection

- **Controller** takes an instruction
- Logic is used so that only one of the logic operation outputs reaches the result output.



3.3. ALU Instruction Decoder

- The decoder is a combinatorial logic circuit(CLC)
- The circuit in which, at any time output is only depends upon inputs only is called CLC
- It converts binary information from 'n' i/p lines to a maximum of 2^n o/p lines
- Multiple output lines, **ctrl1 to clrl4**
- Only one of the output lines is 'on' for each instruction
- 'n' instruction lines = 2^n control lines. Hence size of decoder is n* 2^n

Instruction		Output			
Inst1	Inst2	ctrl1	ctrl2	ctrl3	ctrl4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

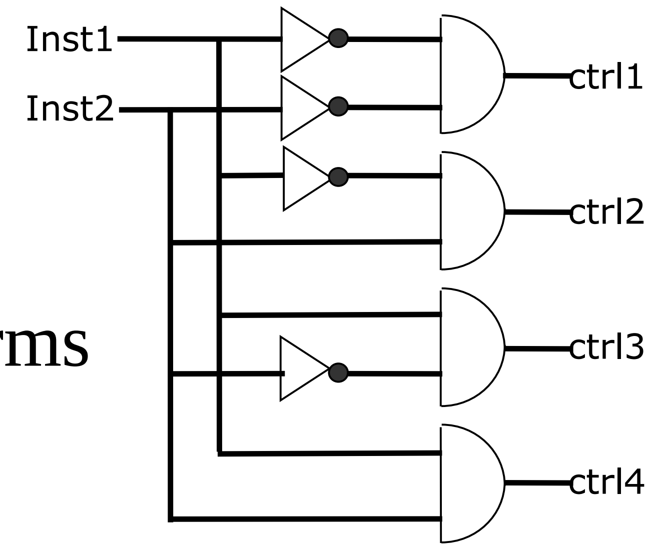
3.3 ALU Instruction Decoder (Contd.)

- From the truth table, we can design a logic circuit.
- Boolean expression for o/p is:

  ctrl1=Inst1’Inst2’, ctrl2=Inst1’Inst2,

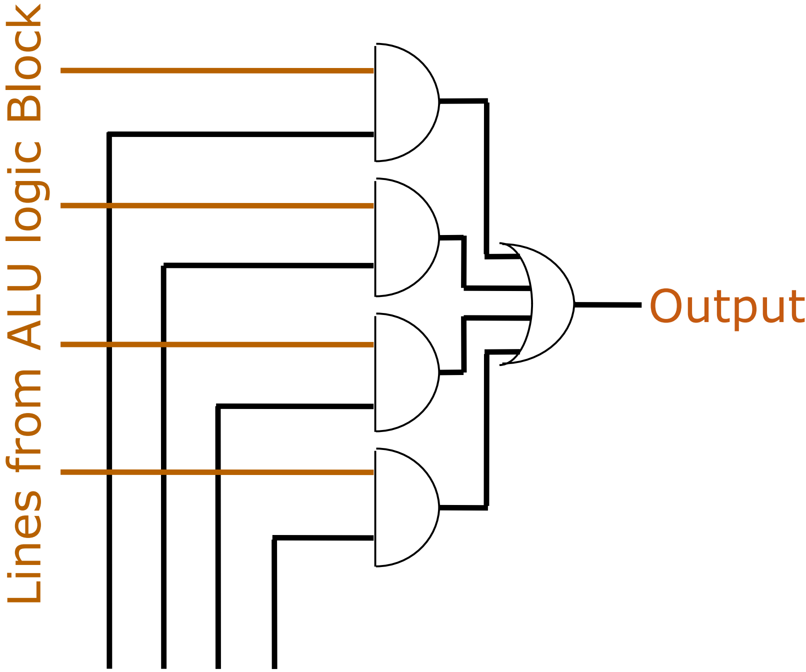
  ctrl3=Inst1Inst2’ and ctrl4=Inst1Inst2

- The arithmetic and logic function section performs all operations at the same time
- So we use the controller to select the output we want



4. ALU Output Multiplexor

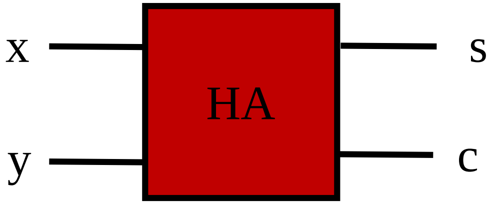
- If we have a collection of **AND** gates, connected to an **OR** gate, we have a logic circuit for the **multiplexer**
- Many to one
- Selects the outputs with the help of decoder outputs



Lines form Decoder output

5. Half adder (HA)

HA Symbol



- Two bit adder circuit
  - Number of inputs = 2 (x and y 'say')
  - Number of outputs= 2 (sum and carry)

  For example:

$$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +1 \\ \hline 10 \end{array}$$

carry

sum

Input		Output	
x	**y**	**Sum(s)**	**Carry(c)**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

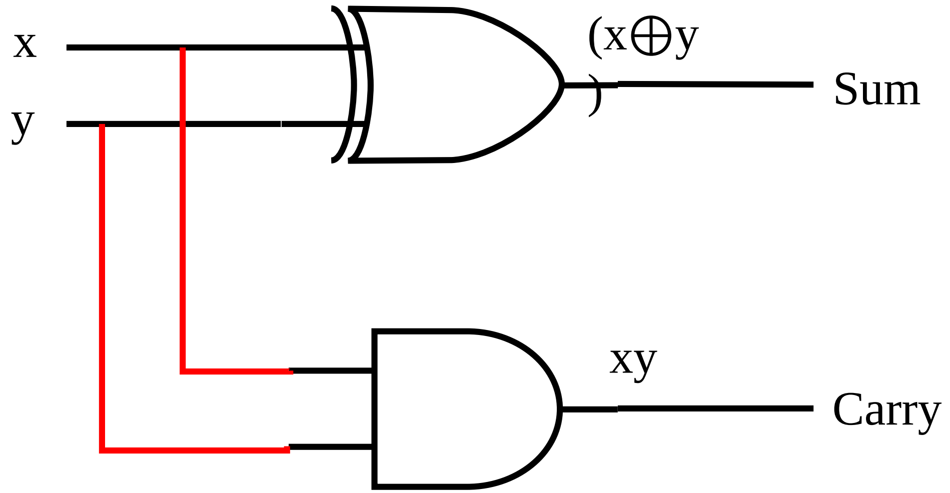
5. Half Adder (Contd…)

- Boolean expression for half adder output from truth table is as;

$$\text{Sum(s)} = x'y + xy'$$
$$= (x \oplus y)$$

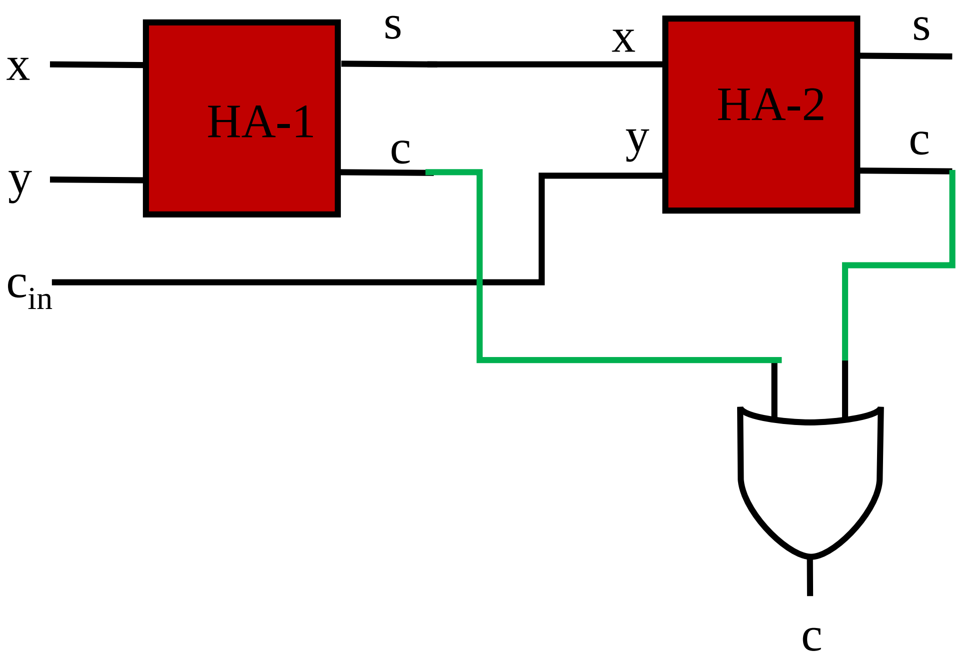
$$\text{Carry(c)} = xy$$

- Logic diagram

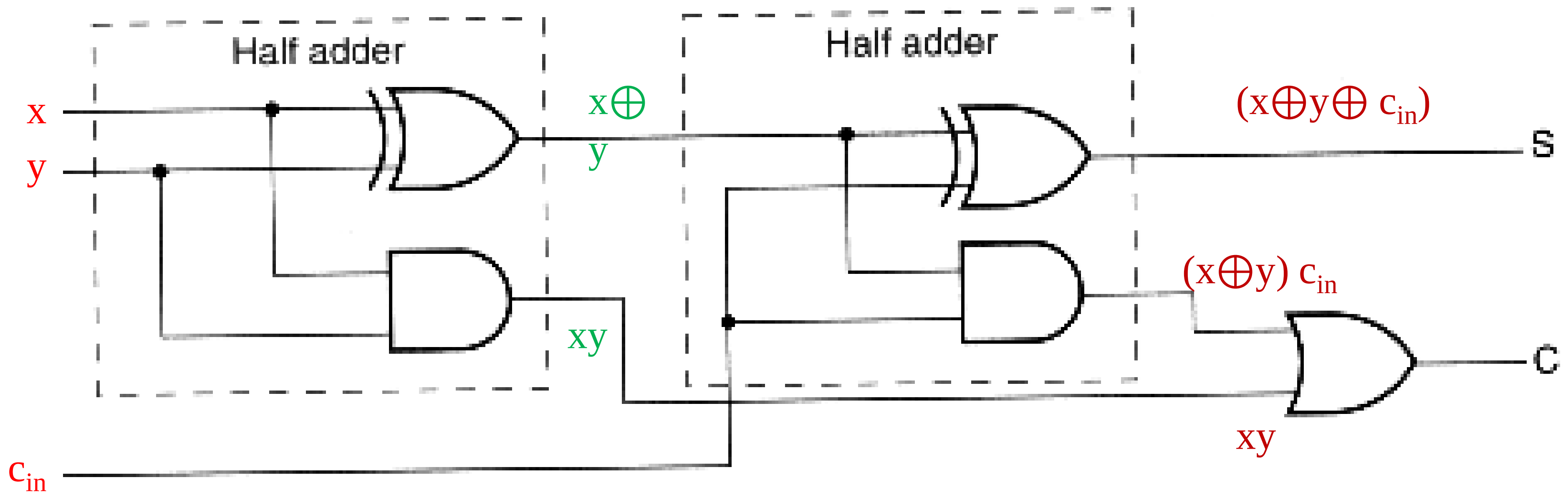


6. Full Adder (FA)

- Three bit adder circuit
  - Number of input= 3(x,y and c_{in})
  - Number of output =2(sum and carry)
- FA can be constructed using two HA and OR gate
- Block diagram

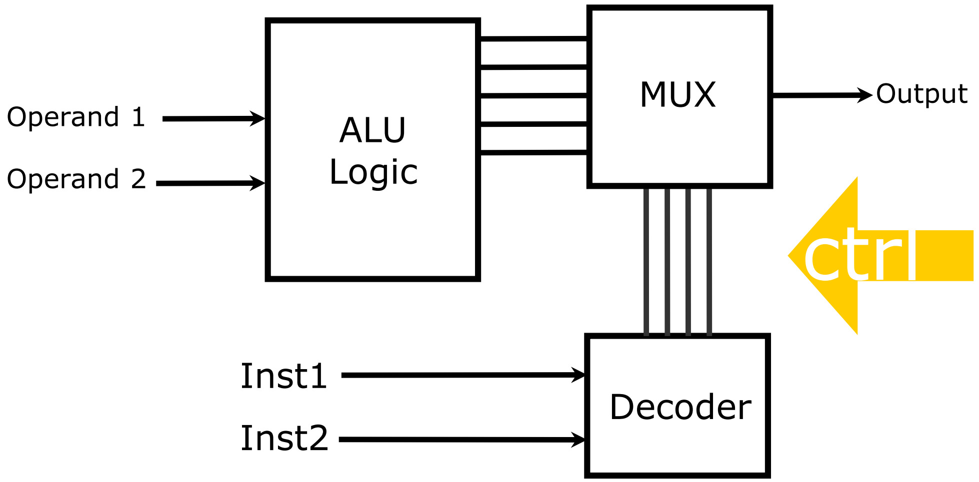


6.1 Logic circuit diagram of FA

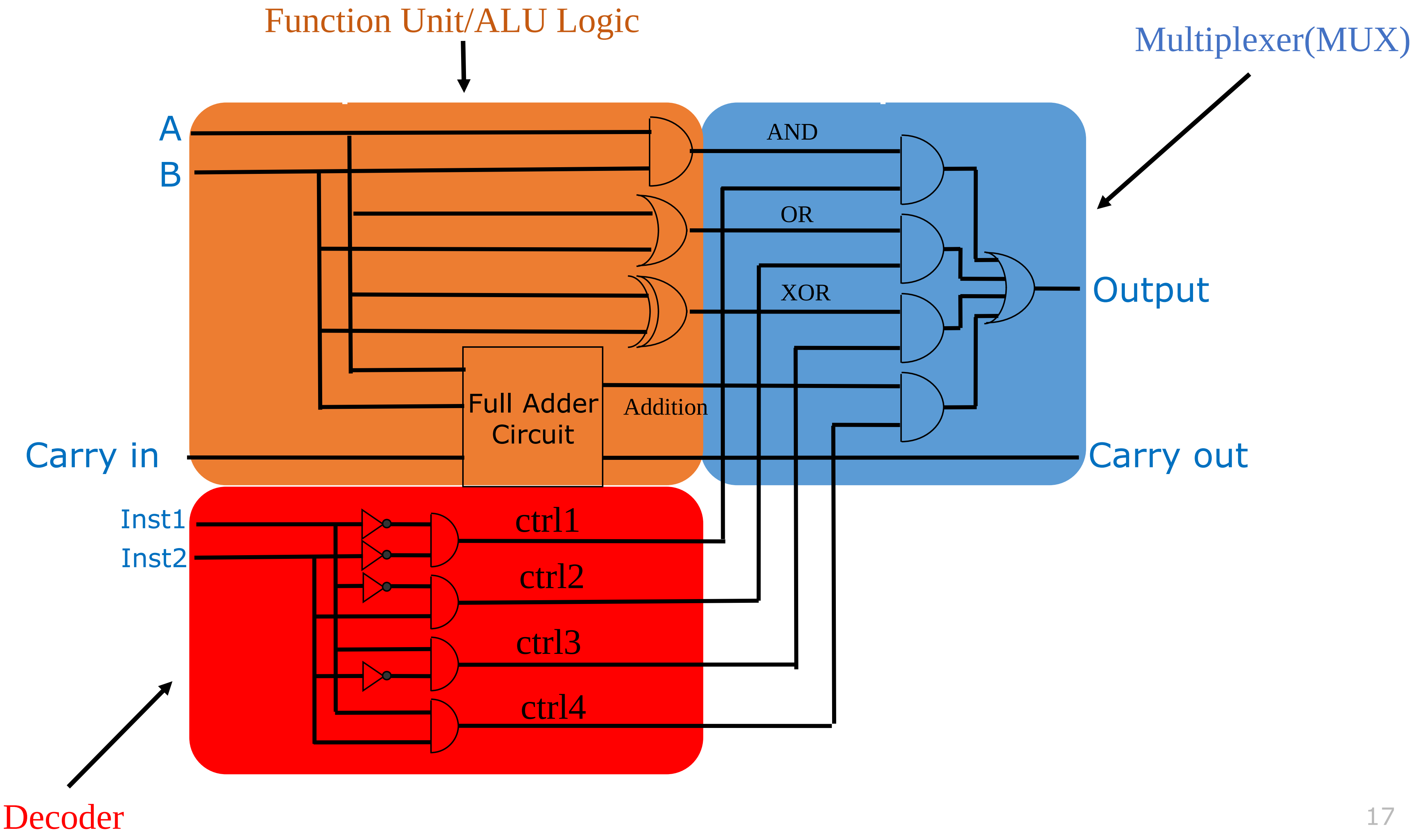


7. ALU Components

- The ALU logic which performs the set of logic and set of arithmetic operations
- The decoder which selects the output we want from the ALU logic
- The multiplexer which implements the choice made through the decoder

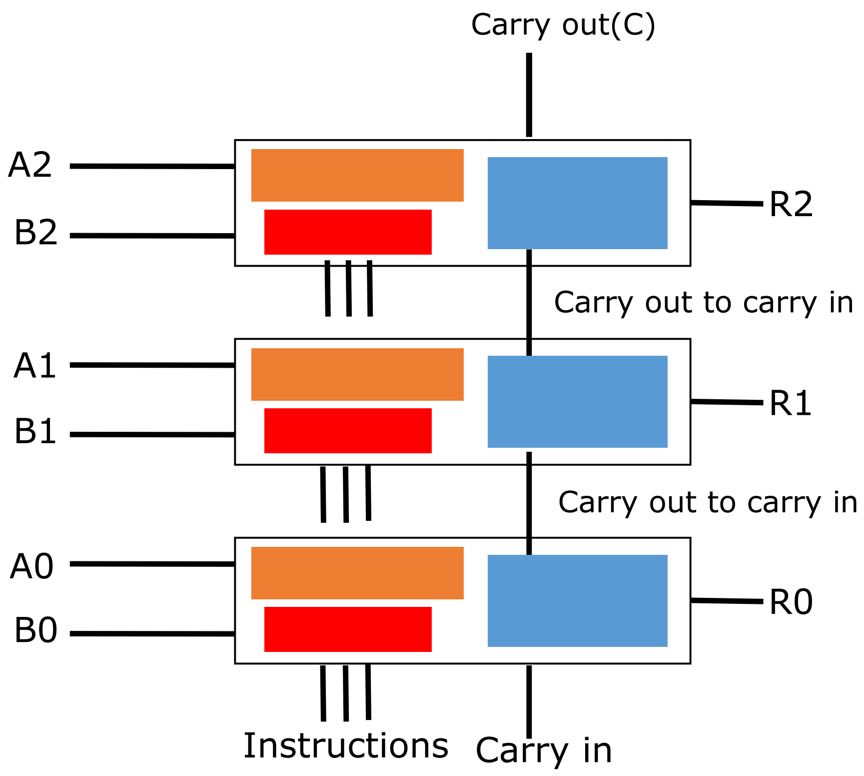


7. ALU Components: Contd.



8. Multi Bit ALU

- Connect multiple single bit ALUs together
- The ‘carry out’ of each ALU links to ‘carry in’ of next ALU
- Final result is CR2R1R0

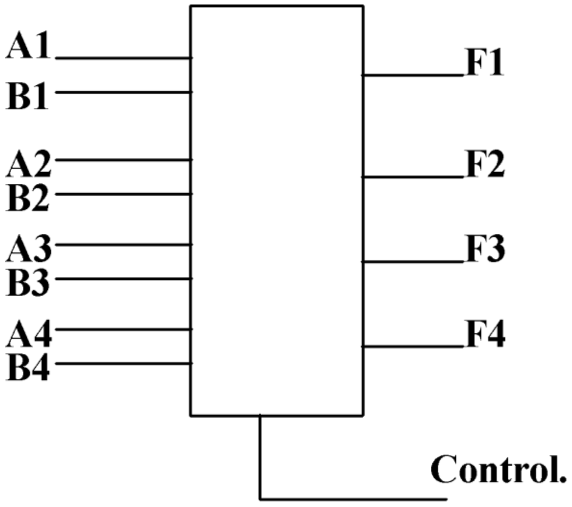


9. Subtraction

- Subtraction can be done by adding **A** to the 2’s complement form of **B**.
  - **S = A+(2’s Complement of B)**
    =A+(1’s complement of B+1)
    =A+(B’+1)
- The rule to convert **B** to **–B** in 2’s complement form is:
  - Invert each bit (using NOT gates).
  - Add 1.

10. Add / Subtract Circuit

- Both addition and subtraction circuits can be combined into a single circuit by using **Controlled Inversion**.
- A Control input determines whether the circuit adds or subtracts.



Adder / Subtractor

11. Controlled Inversion

- Consider the following truth table:
    - **Input Control** will be the **CONTROL.**
    - **Input B** is the **Data.**

Input Control	Input B	Output F
0	0	0
0	1	1
1	0	1
1	1	0

Output F=B

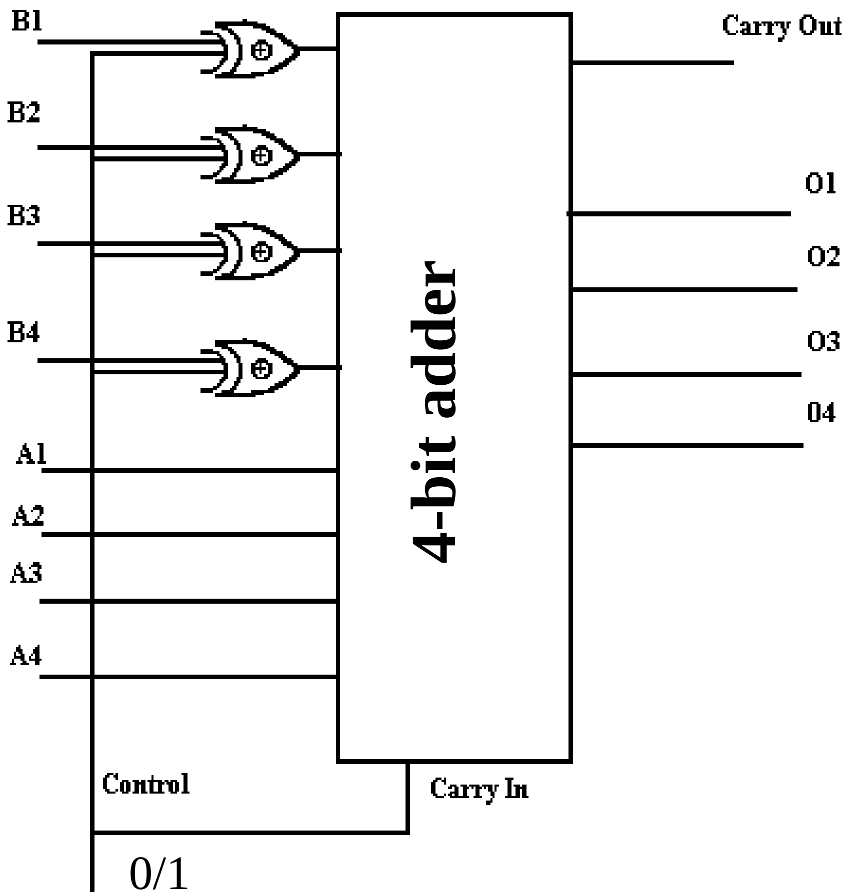
Output F=B′

11. Controlled Inversion (Contd.)

- When Control = 0
  - Then $\mathbf{F} = \mathbf{B}$
- When Control = 1
  - Then $\mathbf{F} = \overline{\mathbf{B}}$
- This is the truth table for XOR

Control	Input B	Output F
0	0	0
0	1	1
1	0	1
1	1	0

12. Circuit for Add / Subtract Unit



Input Control	Input B	Output F
0	0	0
0	1	1
1	0	1
1	1	0

Output F=B

Output F=B'

13. Control of the Add / Subtract Unit

- The Control determines the operation:
- **Control = 0 : Addition**
  - Carry-in is 0 .
  - Output is **A** plus **B** plus 0.

- **Control = 1 : Subtraction**
  - Carry-in is 1.
  - Output is **A** plus (inverse of **B** plus 1).
  - which is **A** plus **(-B)**.

14. Summary

- ALU
  - Functions, controlling and multiplexing.
  - Controlled inversion and subtraction.