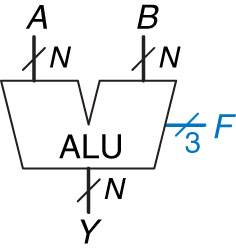
**ALU: Arithmetic Logic Unit**

* **Combine Math and Logic**
* **Heart of most computers**
* **Has 7 functions**

|  |  |  |  |
| --- | --- | --- | --- |
| Control  Signals | | |  |
| F2 | F1 | F0 | Function |
| 0 | 0 | 0 |  |
| 0 | 0 | 1 |  |
| 0 | 1 | 0 |  |
| 0 | 1 | 1 | Not used |
| 1 | 0 | 0 |  |
| 1 | 0 | 1 |  |
| 1 | 1 | 0 |  |
| 1 | 1 | 1 | SLT  (set less than) |

****

**Build an ALU with logic Functions:**

**What do we need?**

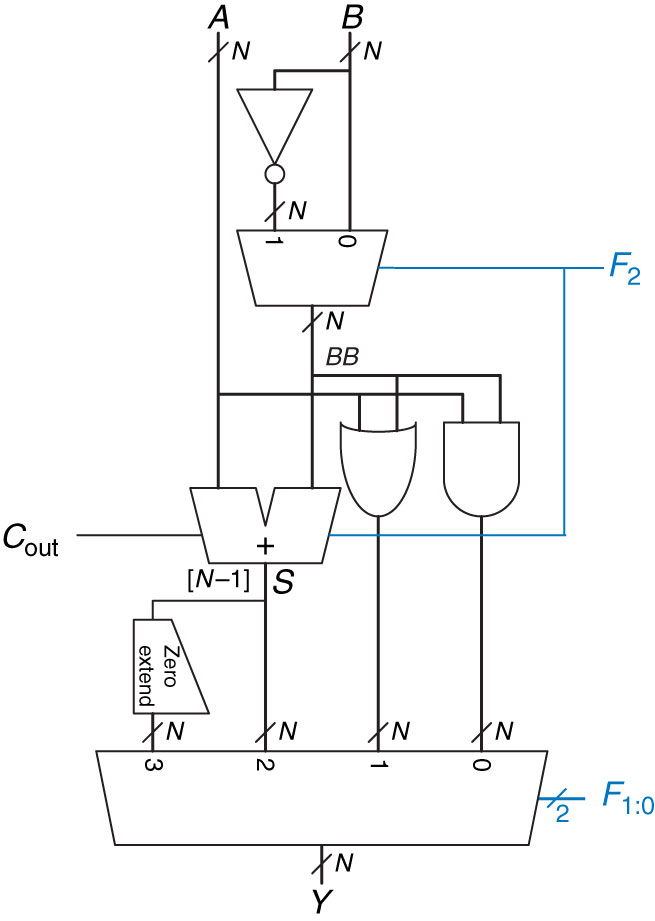
* **Adder**
* **Inverter**

**Magnitude Comparison:**

Compute and looking at the sign (Most Significant Bit) of the result. If the result is negative (i.e. sign bit is 1), then . Otherwise, A is greater than or equal to B.

Can’t connect a single wire to an N-Bit bus, so we zero extend (stick 0’s in front to achieve N-Bits)

* **And**
* **Or**
* **Mux**
* **Comparator**

****

5

4

7

Add for subtract

A

3

2

1

8

6

**How do we multiply in Binary?**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **1** | **0** | **1** |  |  | **5** |
|  | **X** | **0** | **1** | **1** |  | **X** | **3** |
|  |  | **1** | **0** | **1** |  |  |  |
|  | **1** | **0** | **1** | **0** |  |  |  |
| **0** | **0** | **0** | **0** | **0** |  |  |  |
|  | **1** | **1** | **1** | **1** |  | **🡺** | **15** |

**Shifter:**

* **Logical – fill empty spots with 0’s**
* **Arithmetic – when shifting right, fills empty spots with MSB (to preserve sign)**
* **ASL is same as LSL**

**Rotator:**

* **Rotates in circles**
* **# that falls off one end attaches to other end**

**Shamt:**

* **Shift Amount**

**Why is this useful?**

* **Helps multiply and divide!**

**Examples:**

**100110 🡺 Logical Shift Right 🡺 010011**

**32+…+4+2=38 16+…+2+1 = 19**

**Shift Right divide by 2**

**What if it was an Arithmetic Shift Right on a signed number?**

**100110 🡺 Arithmetic Shift Right 🡺 110011**

**2’s Comp 011010 001101**

**16+8+2= -26 8+4+1 = -13**

**ASR still divide by 2**

**What happened…Should be 76?**

We ran out of bits and lost a bit.

6-Bit System only goes up to 63!

**Now assume an 8-Bit System?**

And we get the value of 76

**100110 🡺 Logical/Arithmetic Shift Left (6-Bit System) 🡺 001100**

**32+…+4+2=38 8+4 = 12**

**00100110 🡺 Logical/Arithmetic Shift Left (8-Bit System)🡺 01001100**

**32+…+4+2=38 64+…+8+4 = 76**

**Shift Left Multiply by 2**

**Practice:**

**Use Only two 2:1 mux to create a 3:1 mux:**