**ALU:**

|  |  |  |  |
| --- | --- | --- | --- |
| Control  Signals | | |  |
| F2 | F1 | F0 | Function |
| 0 | 0 | 0 |  |
| 0 | 0 | 1 |  |
| 0 | 1 | 0 |  |
| 0 | 1 | 1 |  |
| 1 | 0 | 0 |  |
| 1 | 0 | 1 |  |
| 1 | 1 | 0 |  |
| 1 | 1 | 1 |  |

**Build an ALU with logic Functions:**

**What do we need?**

* **Adder**
* **Inverter**
* **And**
* **Or**
* **Mux**
* **Comparator**

**How do we multiply in Binary?**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **1** | **0** | **1** |  |  |  |
|  | **X** | **0** | **1** | **1** |  | **X** |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  | **🡺** |  |

**Shifter:**

* **\_\_\_\_\_\_\_\_\_\_\_\_ – fill empty spots with 0’s**
* **\_\_\_\_\_\_\_\_\_\_\_\_– when shifting right, fills empty spots with \_\_\_\_\_ (to \_\_\_\_\_\_\_\_ \_\_\_\_\_\_)**
* **\_\_\_\_\_ is same as \_\_\_\_\_**

**Rotator:**

* **\_\_\_\_\_\_\_\_\_\_\_\_ in \_\_\_\_\_\_\_\_\_\_\_**
* **\_\_\_\_\_\_\_\_\_\_\_\_that falls off one end attaches to \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_**

**Shamt:**

**Why is this useful?**

**Examples:**

**100110 🡺 Logical Shift Right 🡺**

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

**100110 🡺 Arithmetic Shift Right 🡺**

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

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

**Practice:**

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