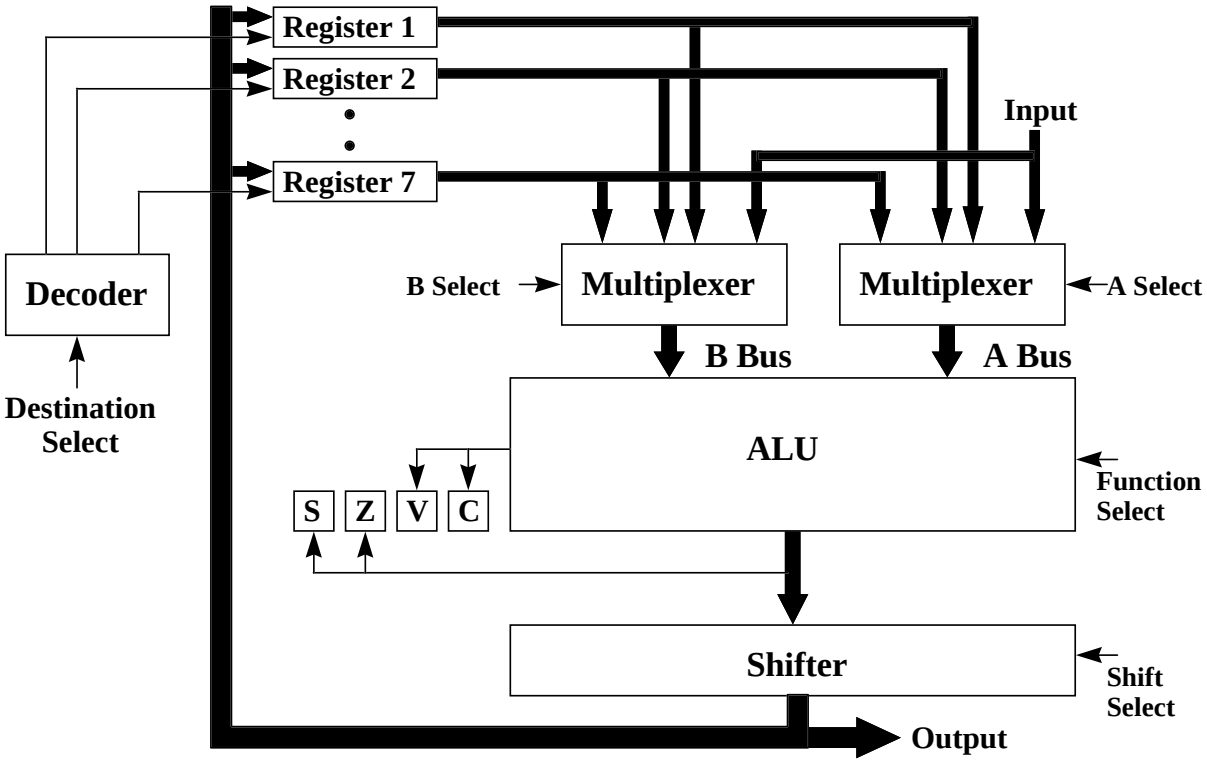
# A Simple Processor Architecture








# ALU

- Performs Arithmentic Functions
- Performs Logic Functions
- Function is Selected by Control
- Status Bits
  - C - Carry
  - V - Overflow
  - Z = 1 If Resultant Contains All Zeros
  - S - Sign Bit of the Result
- Decoder Selects Destination for the Resultant

# ALU

- Inputs
  - Operands
  - Input Carry
  - Operation Select
    - Add
    - Subtract
    - AND
    - OR
    - XOR
  - Mode (Arithmetic or Logic) Select
- Outputs
  - Resultant
  - Output Carry





# ALU Function Table

| Operation Select | | | | | |
|---|---|---|---|---|---|
| $S_2$ | $S_1$ | $S_0$ | $C_{in}$ | Operation | Function |
| 0 | 0 | 0 | 0 | F=A | Transfer A |
| 0 | 0 | 0 | 1 | F=A+1 | Increment A |
| 0 | 0 | 1 | 0 | F=A+B | Add A and B |
| 0 | 0 | 1 | 1 | F=A+B+1 | Add A and B With Carry |
| 0 | 1 | 0 | 0 | F=A+B’ | Add A and One’s Compement of B |
| 0 | 1 | 0 | 1 | F=A+B’+1 | Subtract B From A |
| 0 | 1 | 1 | 0 | F=A-1 | Decrement A |
| 0 | 1 | 1 | 1 | F=A | Transfer A |
| 1 | 0 | 0 | 0 | F=AB | AND |
| 1 | 0 | 1 | 0 | F=A+B | OR |
| 1 | 1 | 0 | 0 | F=A XOR B | Exclusive OR |
| 1 | 1 | 1 | 0 | F=A’ | Complement |

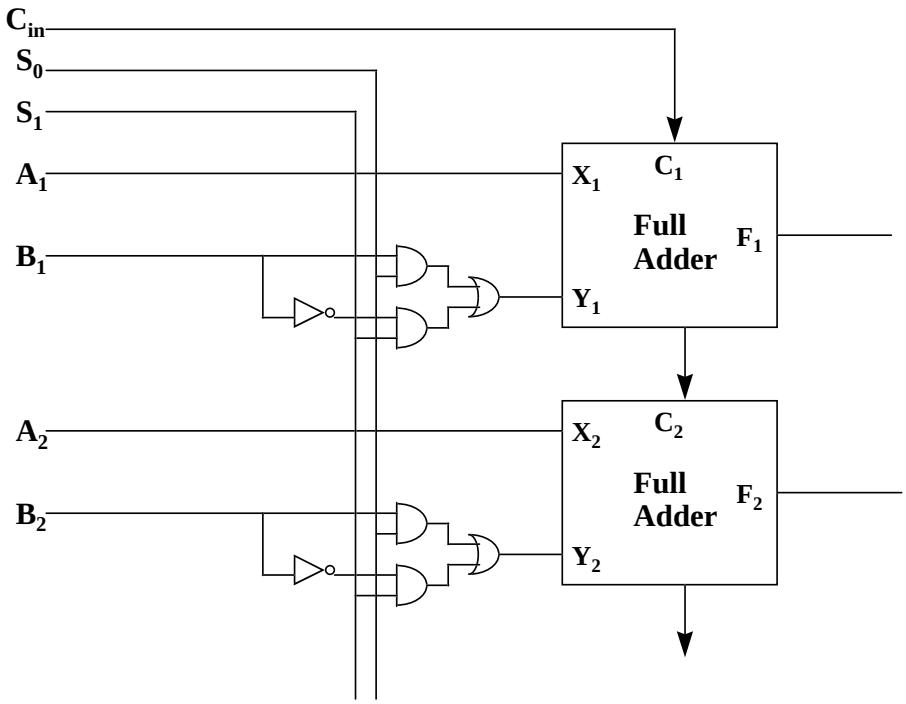
# ALU Components

- Arithmetic
  - Parallel Add
  - One Full Adder per Bit
  - Selection Logic
- Logic
  - Gates
  - Multiplexer

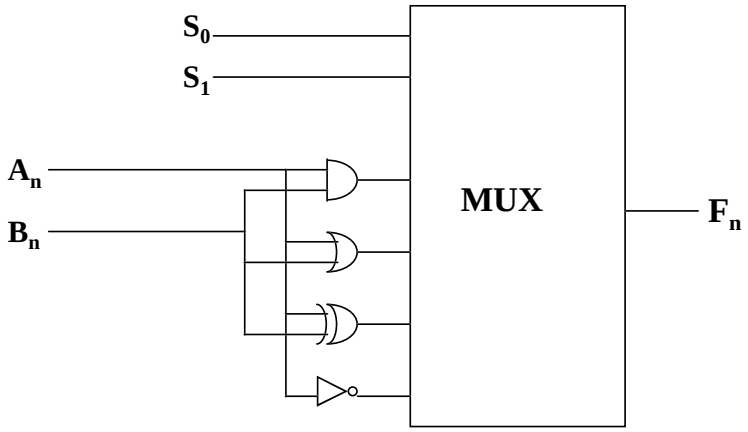




# Circuit for Arithmetic Component



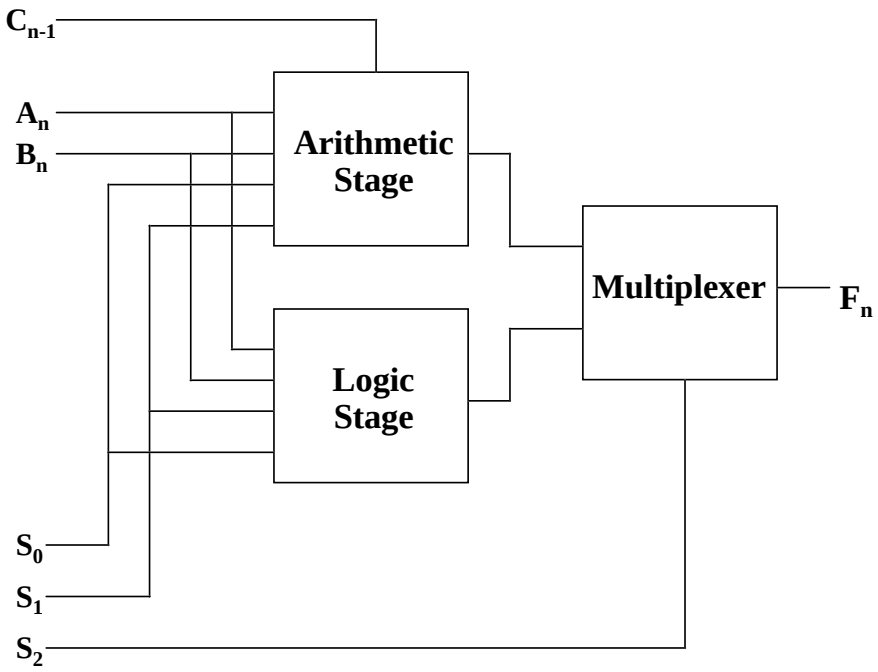
# Circuit for Logic Component








# Internal Structure of ALU



# Shifter

- General
  - Extension of Shift Register Circuit is Possible
  - This Requires Several Clock Pulses
  - This is Time Consuming
- Alternate Approach (Figure 7-18, p. 246 of Mano)
  - Use Multiplexers
  - Wire to Cause Shift Effect
  - Control Determines Nature of Shift
  - Thus, a Single Clock Cycle is Used





# Control Unit Requirements

- MUX A Selector
- MUX B Selector
- ALU Operation Selector
- Shift Selector
- Destination Selector

# Control Word

- Number and Organization of Bits Required to Control ALU
- Bit Requirements
  - A: A Bus Select (Seven Registers Plus Input): 3 bits
  - B: B Bus Select (Seven Registers Plus Input): 3 bits
  - D: Destination Select (Seven Registers): 3 bits
  - F: ALU Control (Four bits)
  - H: Shift Control (Three bits)
  - TOTAL = 16 bits
  - Thus, 16 Bits Can Be Used to Perform All Microoperations






# Control Word Encoding

| | Operation (F) | | | | | |
|---|---|---|---|---|---|---|
| Code | $C_{in}$ = 0 | $C_{in}$ = 1 | A | B | D | H |
| 0 0 0 | F=A | F=A+1 | Input | Input | None | No Shift |
| 0 0 1 | F=A+B | F=A+B+1 | R1 | R1 | R1 | SHL |
| 0 1 0 | F=A+B’ | F=A+B’+1 | R2 | R2 | R2 | SHR |
| 0 1 1 | F=A-1 | F=A | R3 | R3 | R3 | Bus=0 |
| 1 0 0 | F=AB | | R4 | R4 | R4 | |
| 1 0 1 | F=A+B | | R5 | R5 | R5 | ROL |
| 1 1 0 | F=A XOR B | | R6 | R6 | R6 | ROR |
| 1 1 1 | F=A’ | | R7 | R7 | R7 | |

# Microoperations and Microprograms

- **Example Microoperation**
  - **R1← R2 - R3**
  - **Symbolically: R2,R3,R1,F=A-B,No Shift**
  - **Control Word = 010 011 001 0101 000 = 4CA8 (H)**
- **Clearly, Many Microoperations Are Possible**
- **Control Memory**
  - **Location of Available Microoperations**
  - **Width of Control Memory = Control Word**
- **Microprograms Can be Written Using a Sequence of Microoperations**