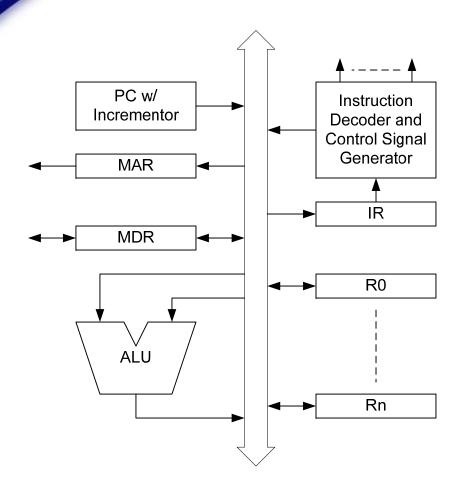


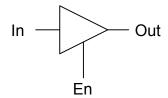


- Design, synthesize, and simulate a simple microprocessor in Verilog
- Combinational and sequential logic circuits
- Hardware and software
- Hardwired control
- Bus architecture

### **Overall Structure**



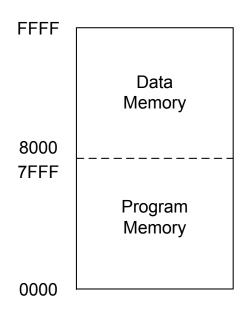
Tri-state buffers are needed for bus drivers





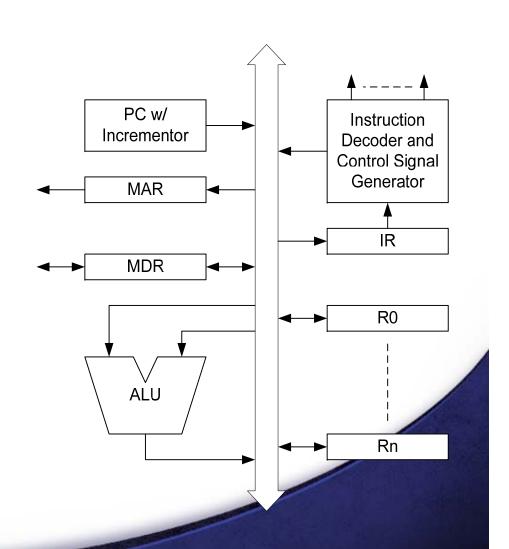
- Single thread
- 16-bit Data bus
- 16-bit Address bus
- Program memory and data memory are mapped into single address space
- Primary inputs: data (16-bit), clock, reset, MFC
- Primary outputs: data (16-bit), address (16-bit), R/W, EN
- Virtual memory module

# **Virtual Memory Map**



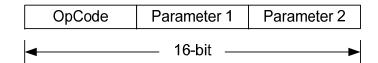
## **Components and Control Signals**

- IR: 16-bit, "In", "<u>Out</u>"
- GPR: 16-bit, "In", "Out"
- PC: 16-bit, "Out", "Inc"
- MAR: 16-bit, "In", "Out"
- MDR: 16-bit, "In-uP", "In-Mem", "Out-uP", "Out-Mem"
- ALU: "In1", "In2", "Out", Add, Sub, Not, And, Or, Xor, Xnor



### **Instruction Set**

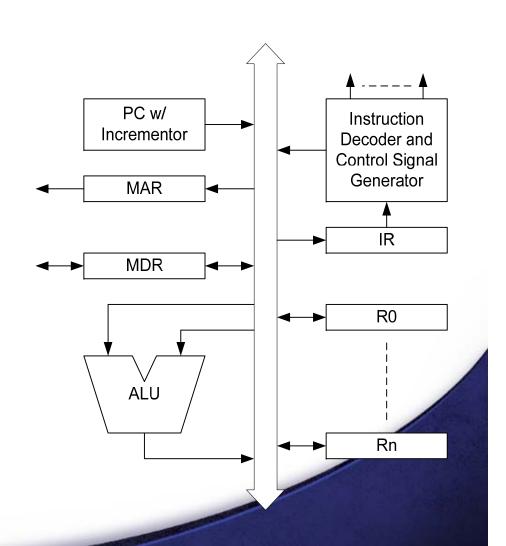
- Mov Ri, Rj
- Add Ri, Rj
- Addi Ri, #
- Sub Ri, Rj
- Subi Ri, #
- Not Ri
- And Ri, Rj
- Or Ri, Rj
- Xor Ri, Rj
- Xnor Ri, Rj
- Movi #, Ri
- Load (Ri), Rj
- Store Ri, (Rj)



- OpCode: 4 bits
- Parameter 1: 6 bits
- Parameter 2: 6 bits

## **Hardwired Control Example**

- Add R1, R2
- 1. PC Out, MAR In
- 2. MAR Out, R/W = 1, EN
- 3. Wait for MFC
- 4. MDR In-Mem
- 5. MDR Out-uP, IR In
- 6. R1 Out, ALU Input-Reg1 In, PC Inc
- 7. R2 Out, ALU Input-Reg2 In
- 8. Add, ALU Output-Reg In
- 9. ALU Output-Reg Out, R1 In



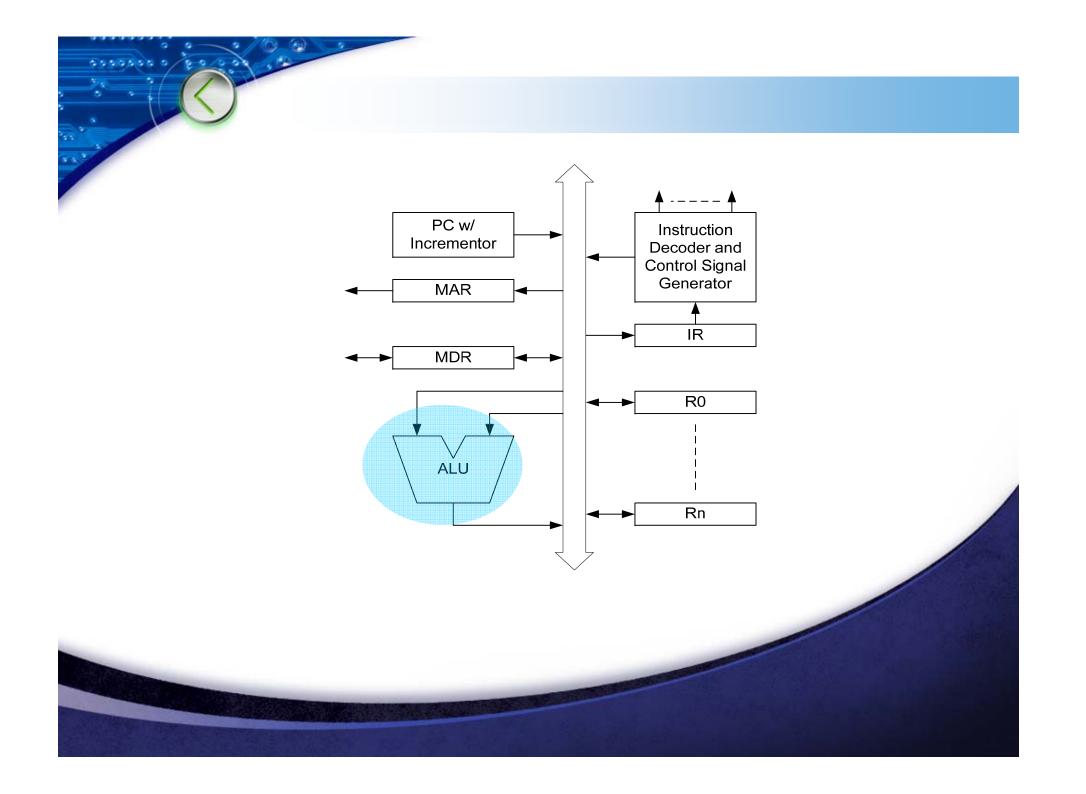


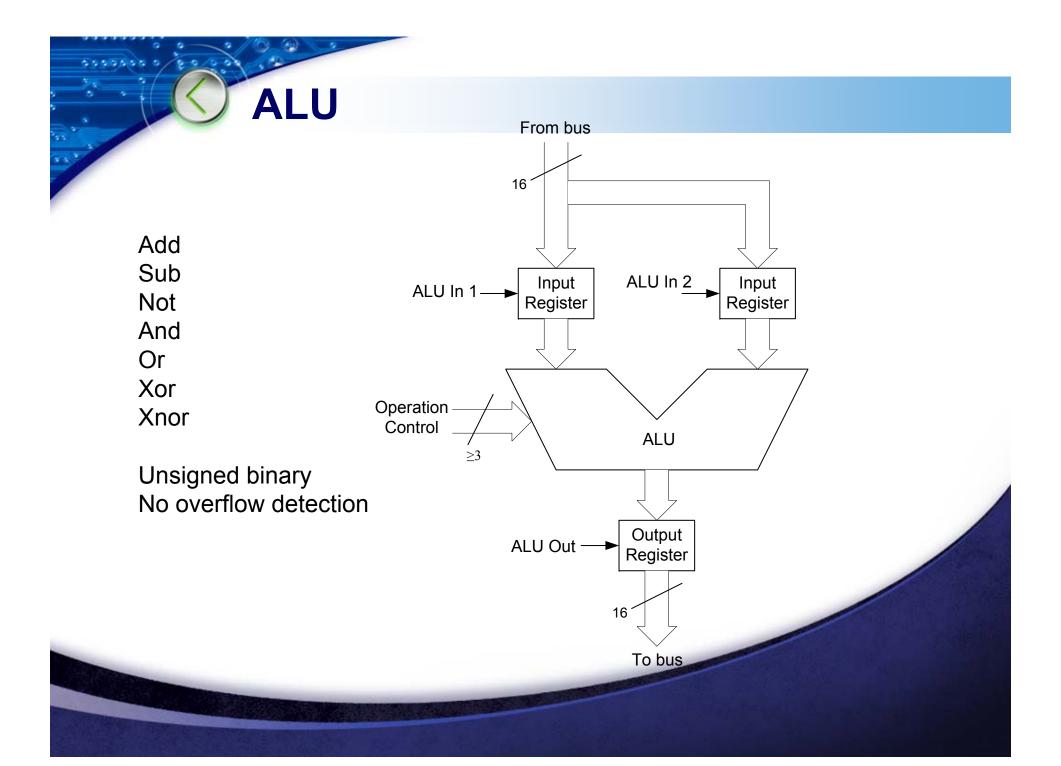
- Additional instructions/addressing modes
- Other hardware blocks
- # of registers
- # of clock cycles for each operation



# Divide-and-Conquer #1

**ALU** 





### Final Project Step #1: ALU

- Design an ALU following the specification
- Simulate in ModelSim to ensure the functionality using a testbench that covers everything
- Synthesize the designed ALU in Design Vision
- Simulate the synthesized netlist
- Write a simple project report (code, testbench, conclusion)
- Due date: Oct. 31

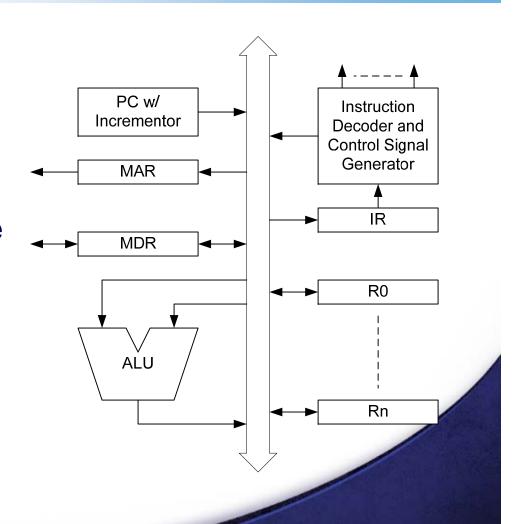


## Divide-and-Conquer #2



## Registers

- All registers are D-flip flops
- Edge-sensitive
- All registers driving the bus need to have tri-state buffers at their outputs
- MAR is uni-directional
- MDR is bi-directional
- Use one register to handle each direction





- 16-bit counter
- Need tri-state buffer at the output
- Inputs include reset and increment control signal

### Final Project Step #2: Registers and PC

- Design registers and PC following the specification
- Simulate in ModelSim to ensure the functionality using a testbench that covers everything
- Synthesize the design in Design Vision
- Simulate the synthesized netlist
- Write a simple project report (code, testbench, conclusion)
- Due date: next Monday (Nov. 7)