

# Architecture, Design and Implementation of Embedded Systems for Real-Time Applications

## CpE-450 Spring 05

### Class 9

Bruce McNair

[bmcnair@stevens.edu](mailto:bmcnair@stevens.edu)


# Loop timing with interrupts

```
main()
{
    start_timer(500);
    enable_interrupts();
    while(1)
    {
        perform_background_processing();
        halt_processor(WAKE_ON_INT);
    }
}
```

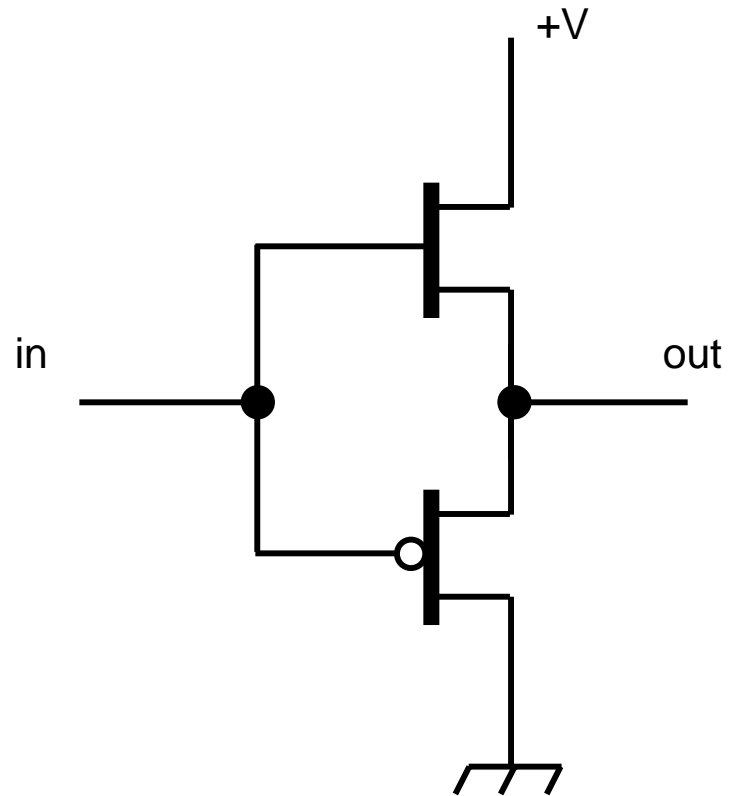
```
void IO_ISR(void)
{
    disable_interrupts();
    do_IO();
    enable_interrupts();
}
```

```
void blink_ISR (void)
{
    restart_timer(500);
    disable_interrupts();
    TOGGLE_LED();
    do_other_stuff();
    enable_interrupts();
}
```

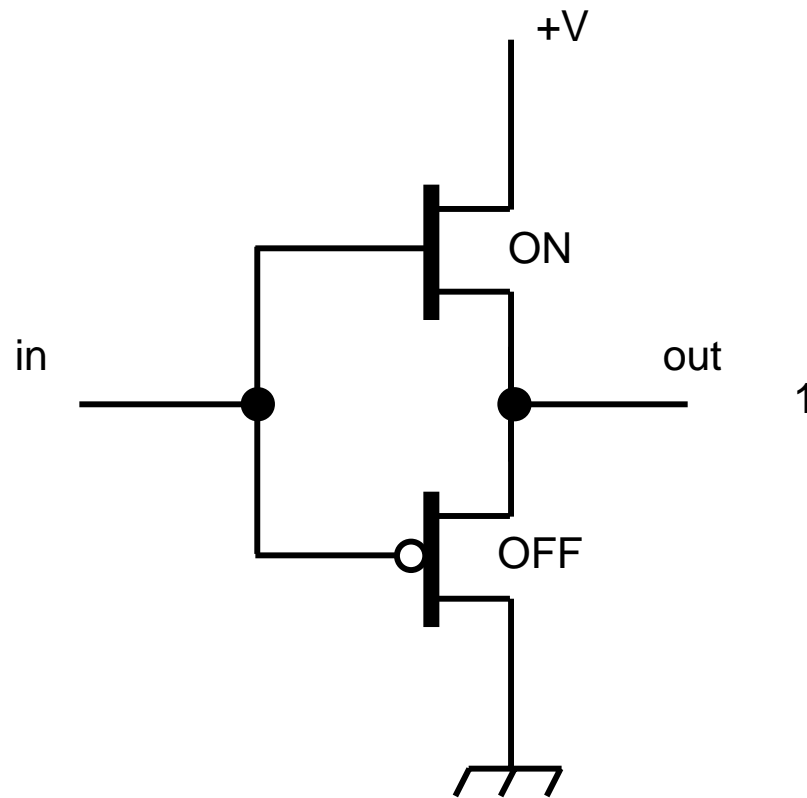
No unnecessary functions  
Minimal power consumption  
Wastes no real-time



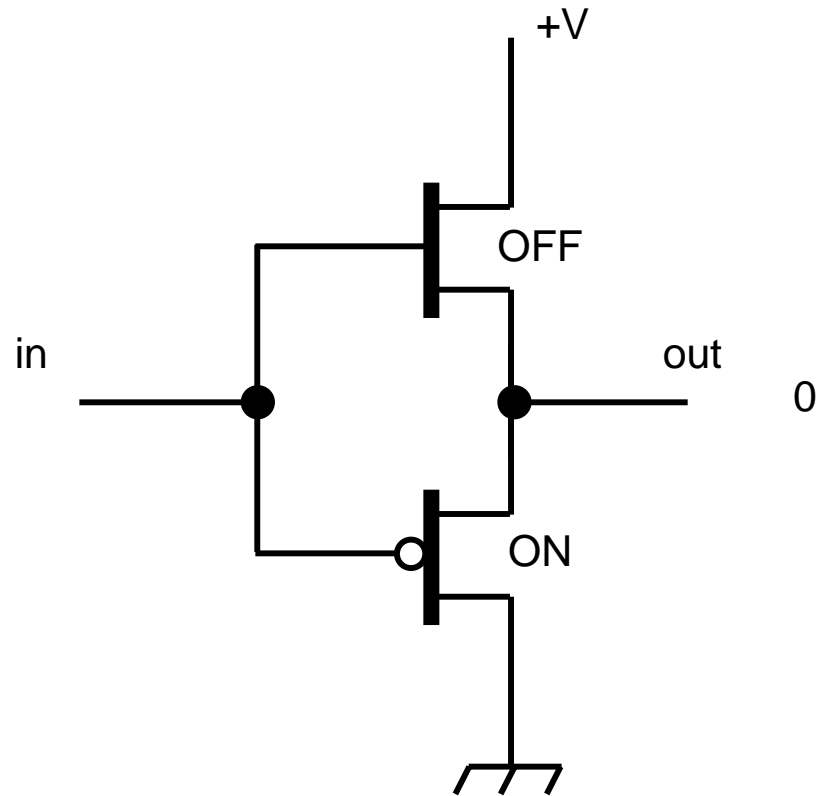
# Power Consumption in CMOS Processors



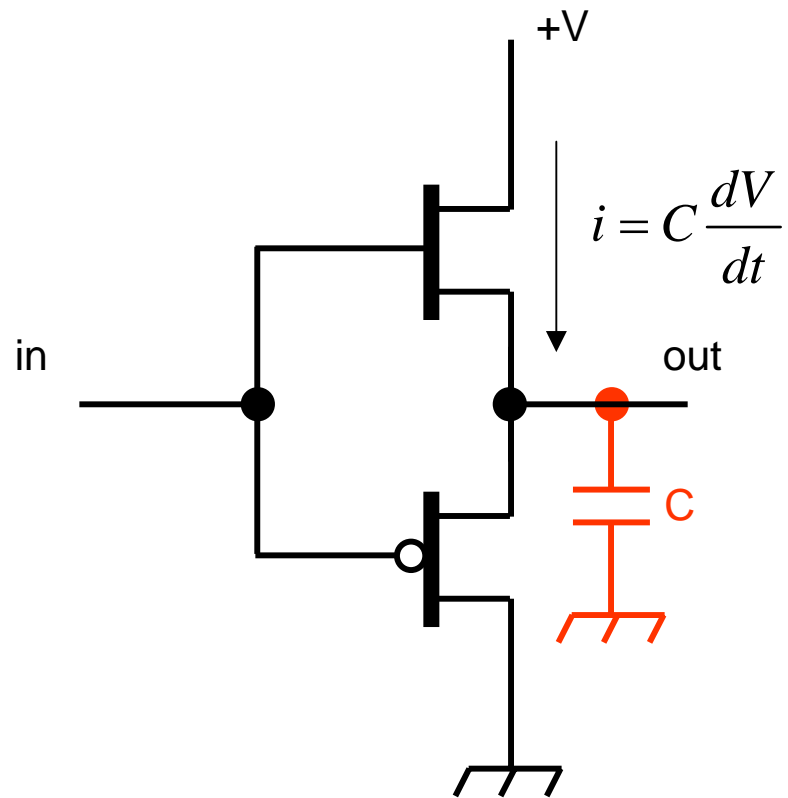
# Power Consumption in CMOS Processors



# Power Consumption in CMOS Processors

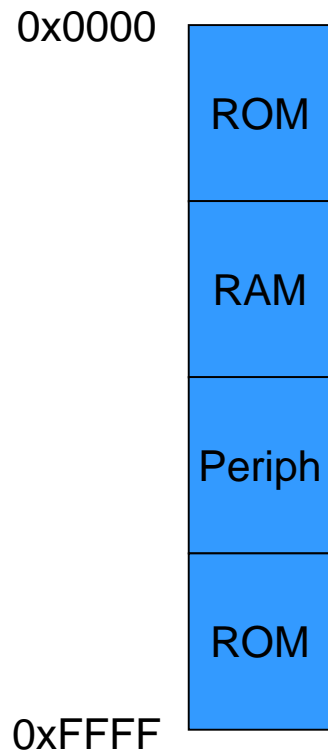


# Power Consumption in CMOS Processors



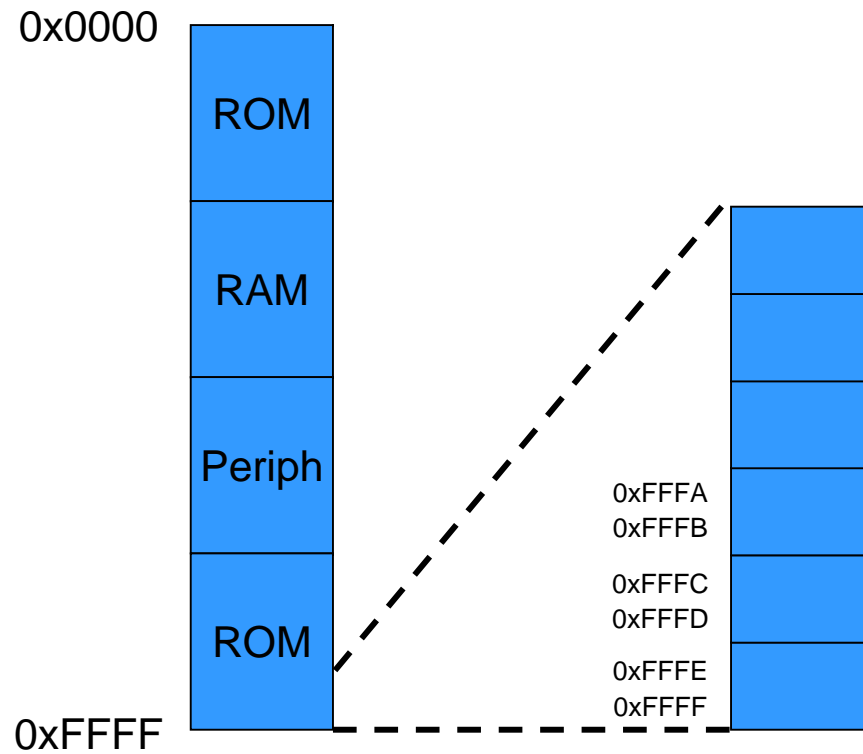
# Vectored Interrupts

- Example memory map:



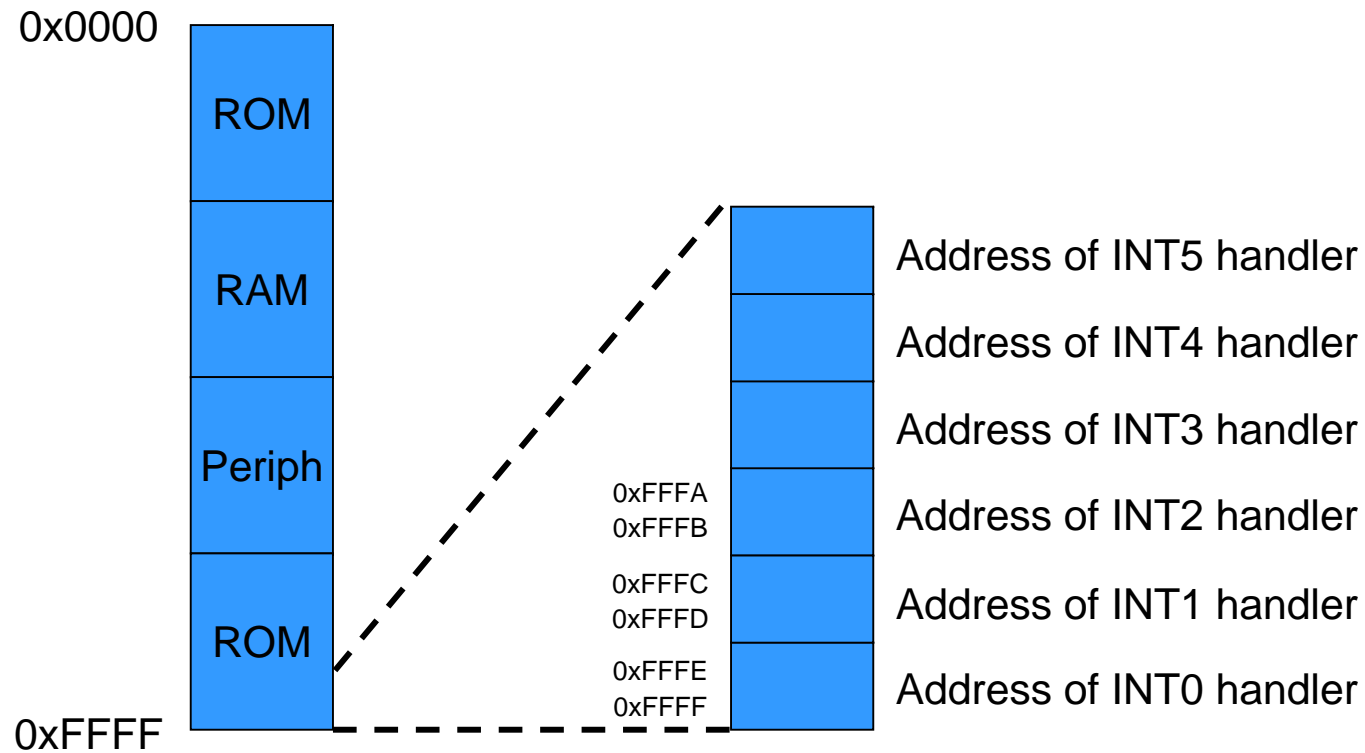
# Vectored Interrupts

- Example memory map:



# Vectored Interrupts

- Example memory map:



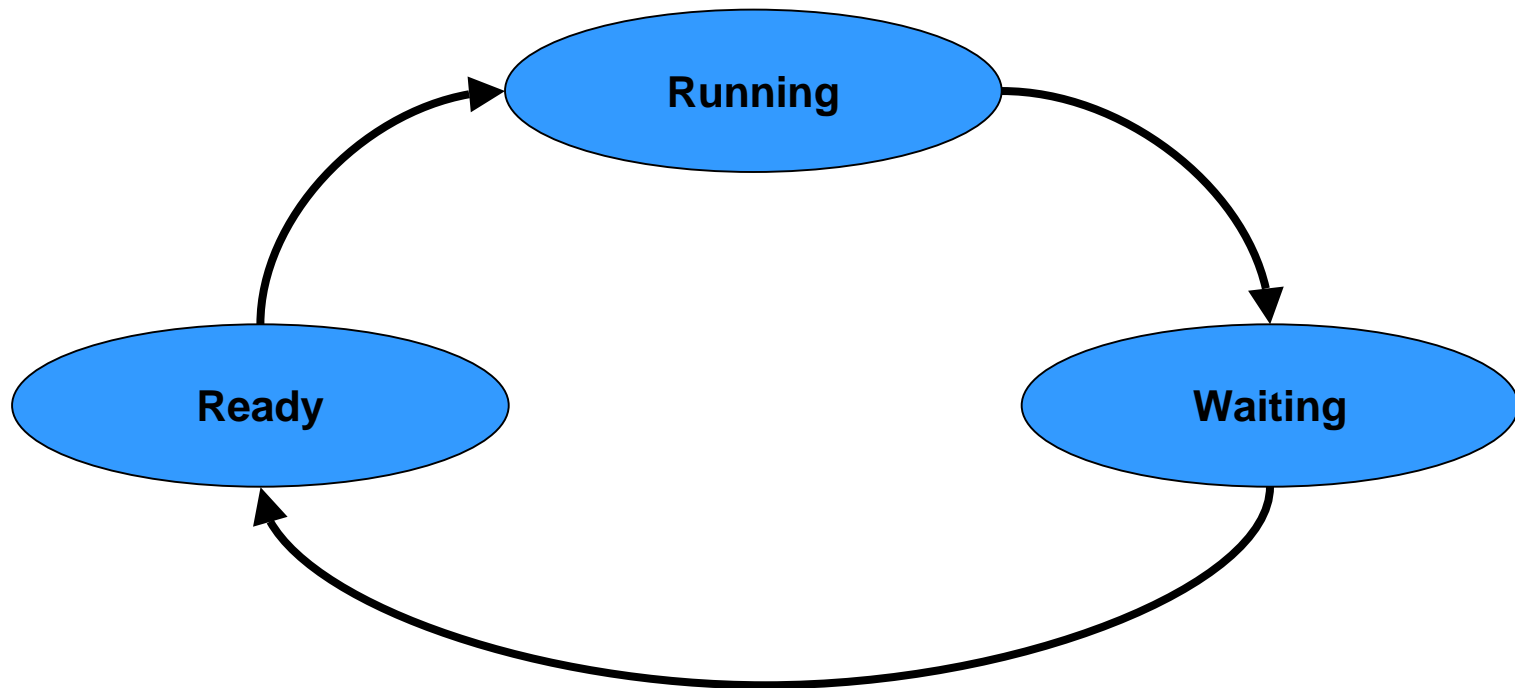
# Interrupt Processing

- Example pseudo-microcode

```
void microcomputer_microcode(void)
{
    while(1)
    {
        while(!interrupt())
        {
            fetch_instruction(addr);
            execute_instruction();
            addr++;
        }
        push(addr);
        addr := handle_interrupt(int_level)
    }
}

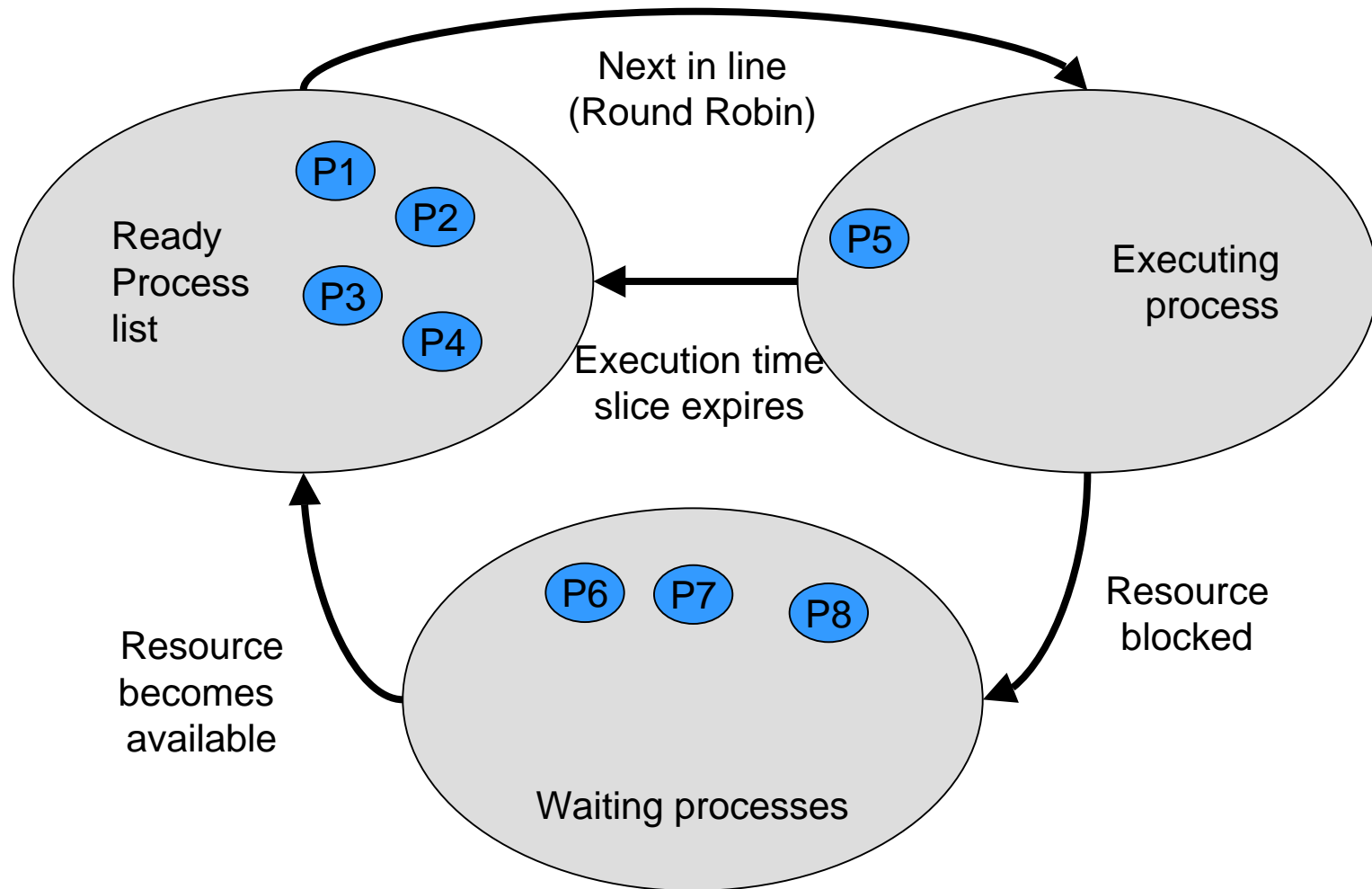
unsigned int *handle_interrupt(unsigned int int_level)
{
    unsigned int_vec;
i := get_interrupt_level();
    int_vec := 0xFFFF - 2*int_level - 1;
    return( (unsigned int *)int_vec );
}
```

# Process Scheduling



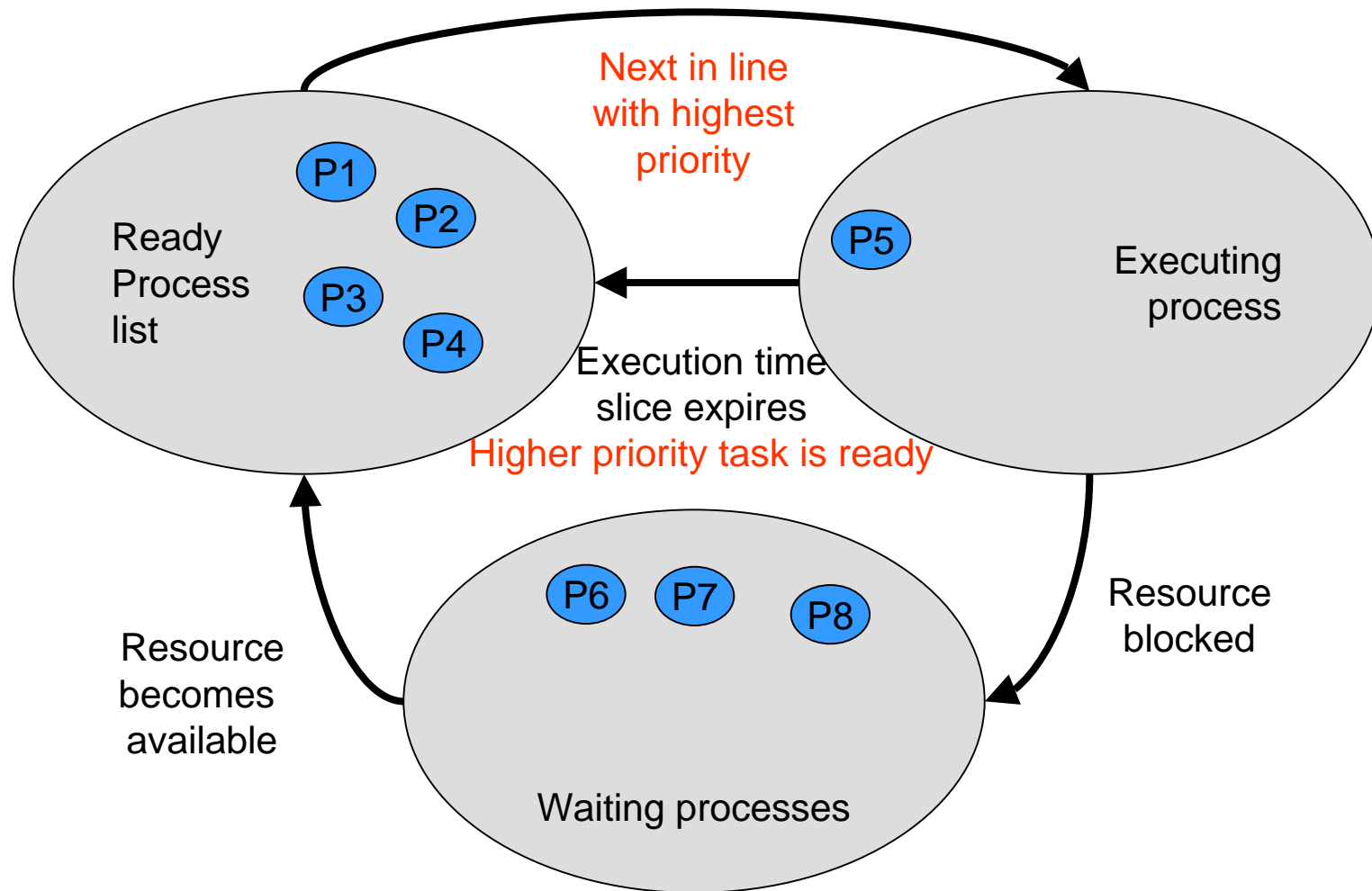
# Choosing a process to run

- Standard OS:

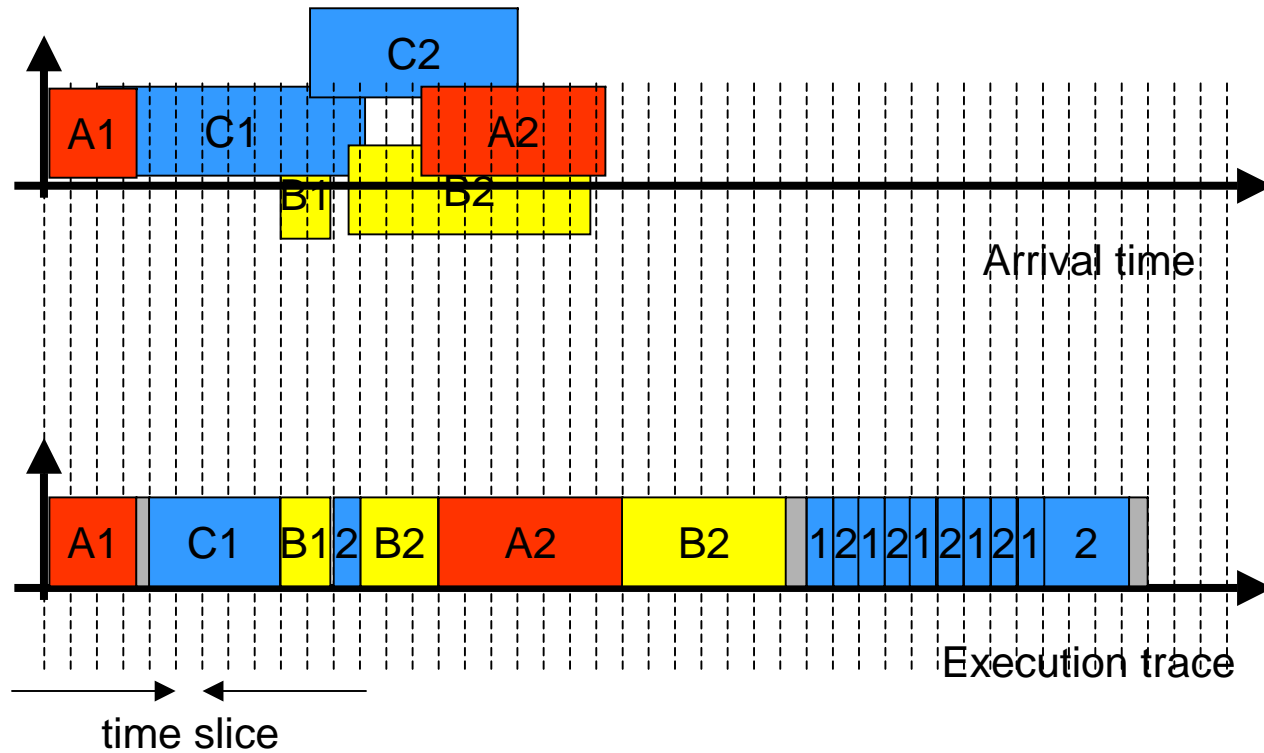
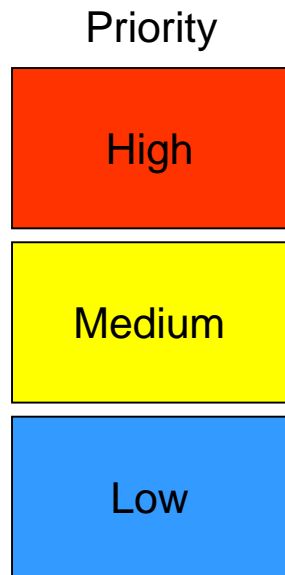


# Choosing a process to run

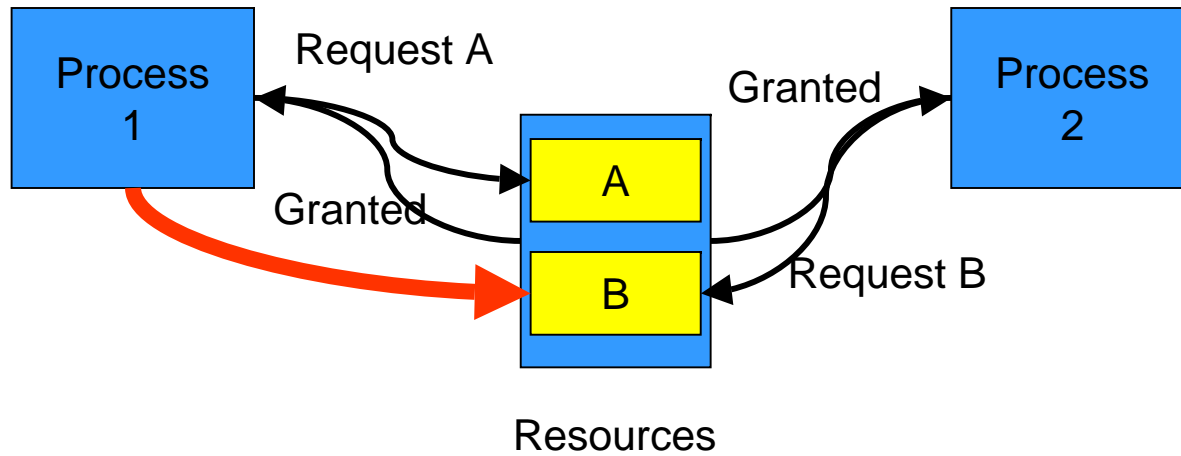
- Real-time OS:



# Task Priority Example

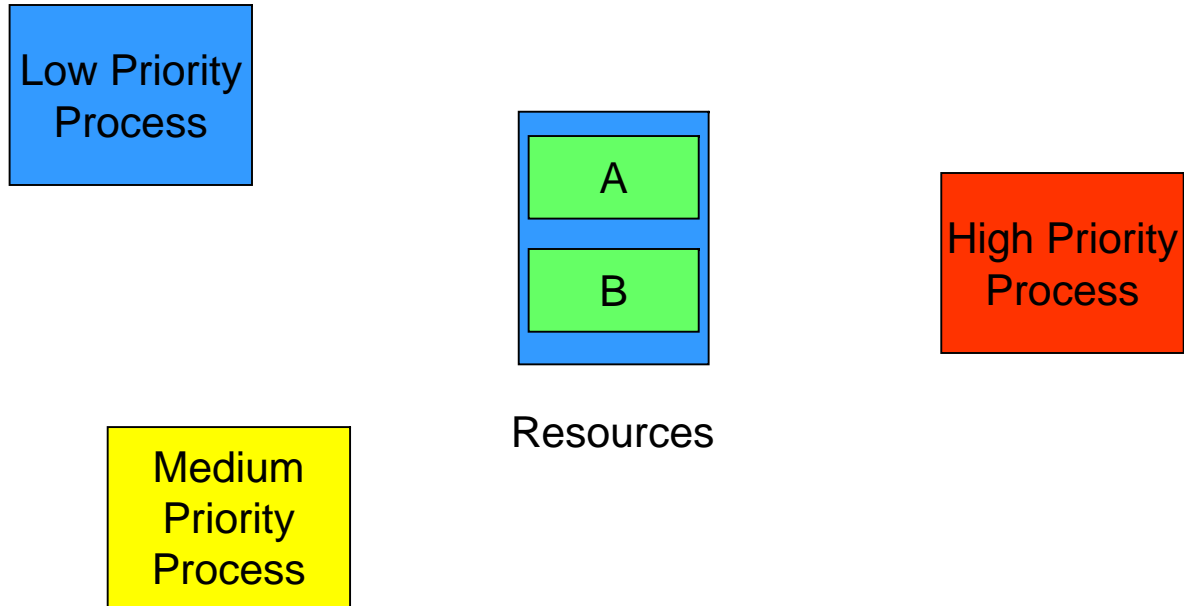


# Shared Resources and Process Contention

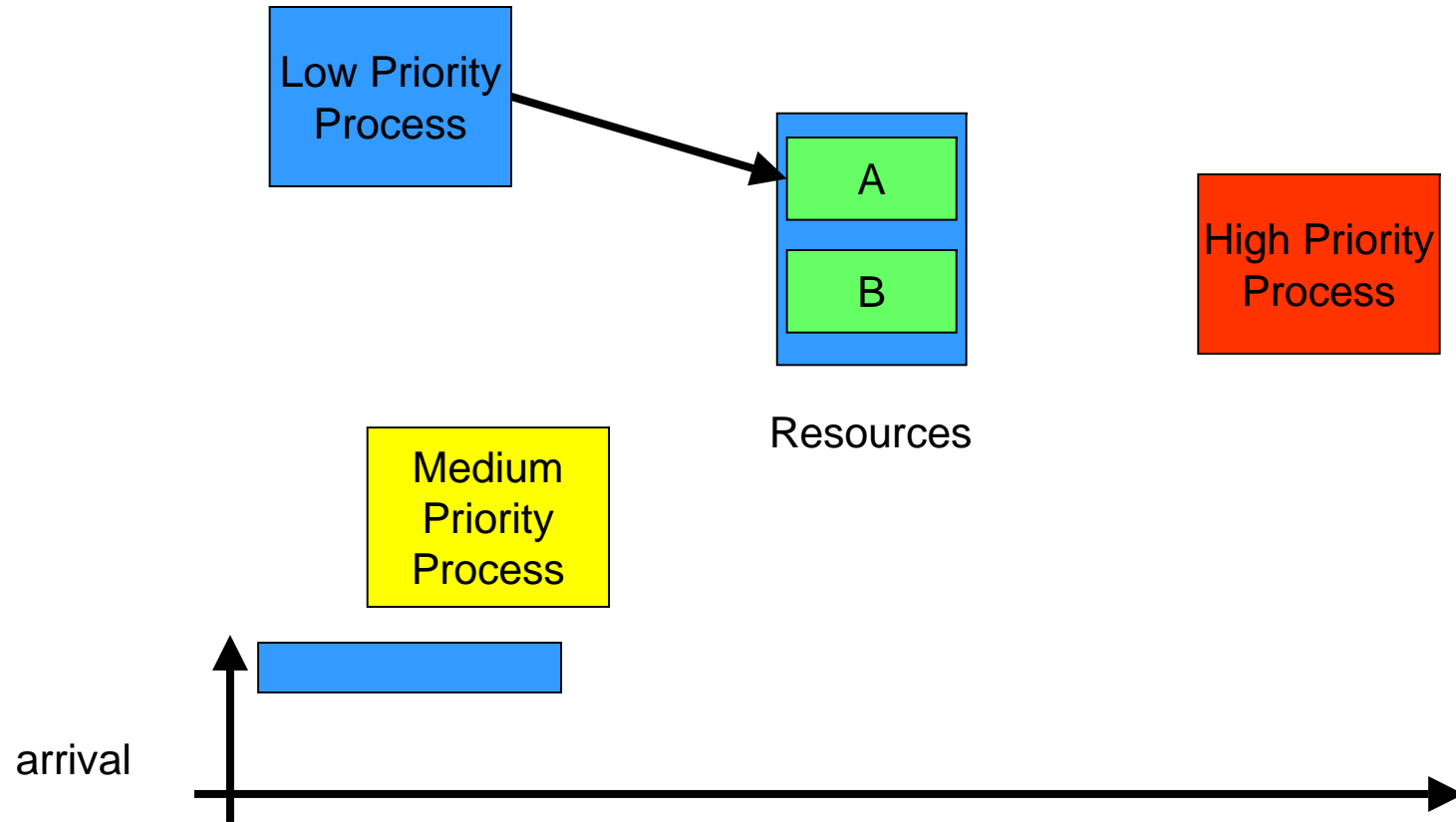


- Shared resources and contending processes create the potential for deadlock

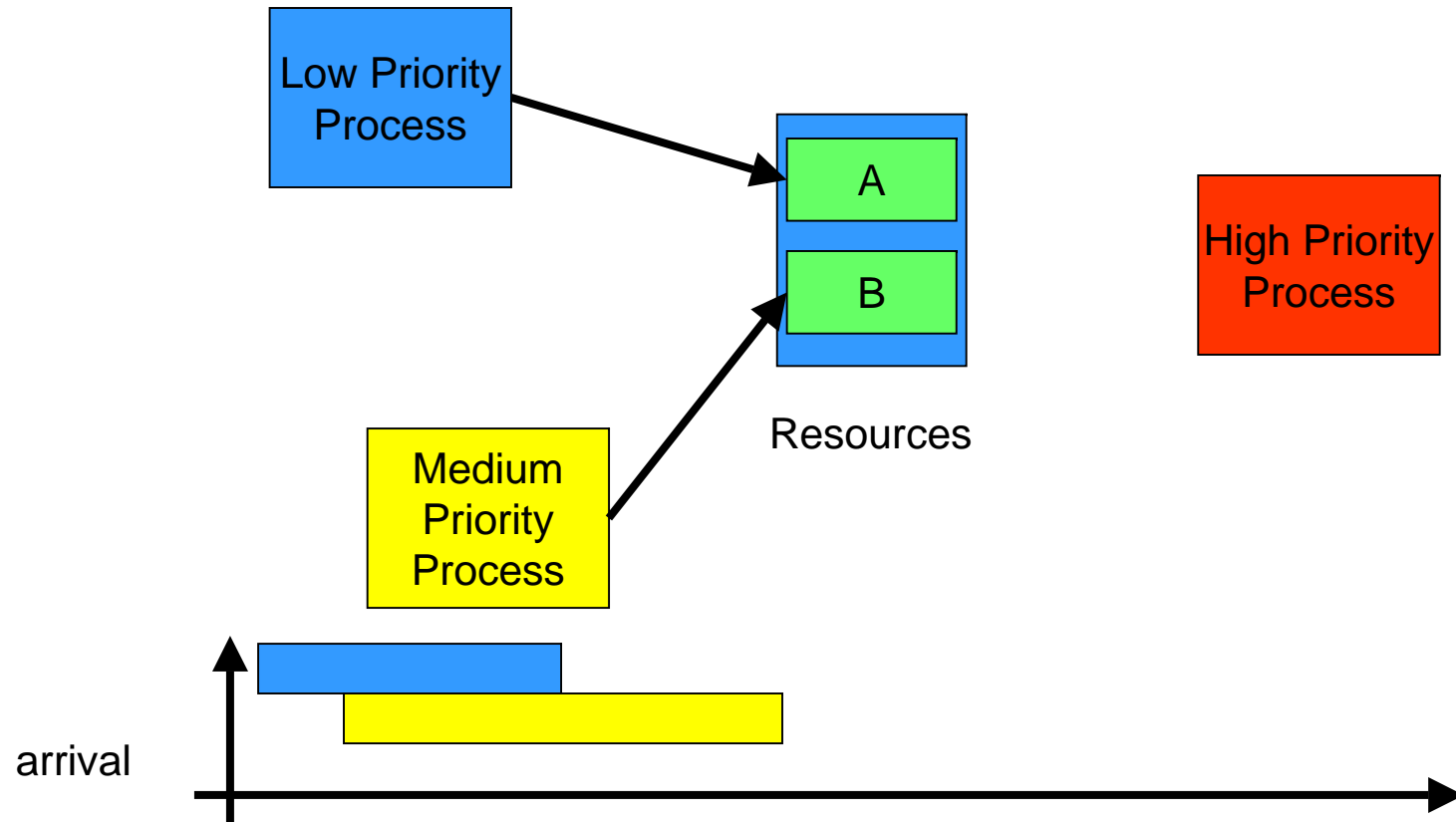
# Task Priority Inversion



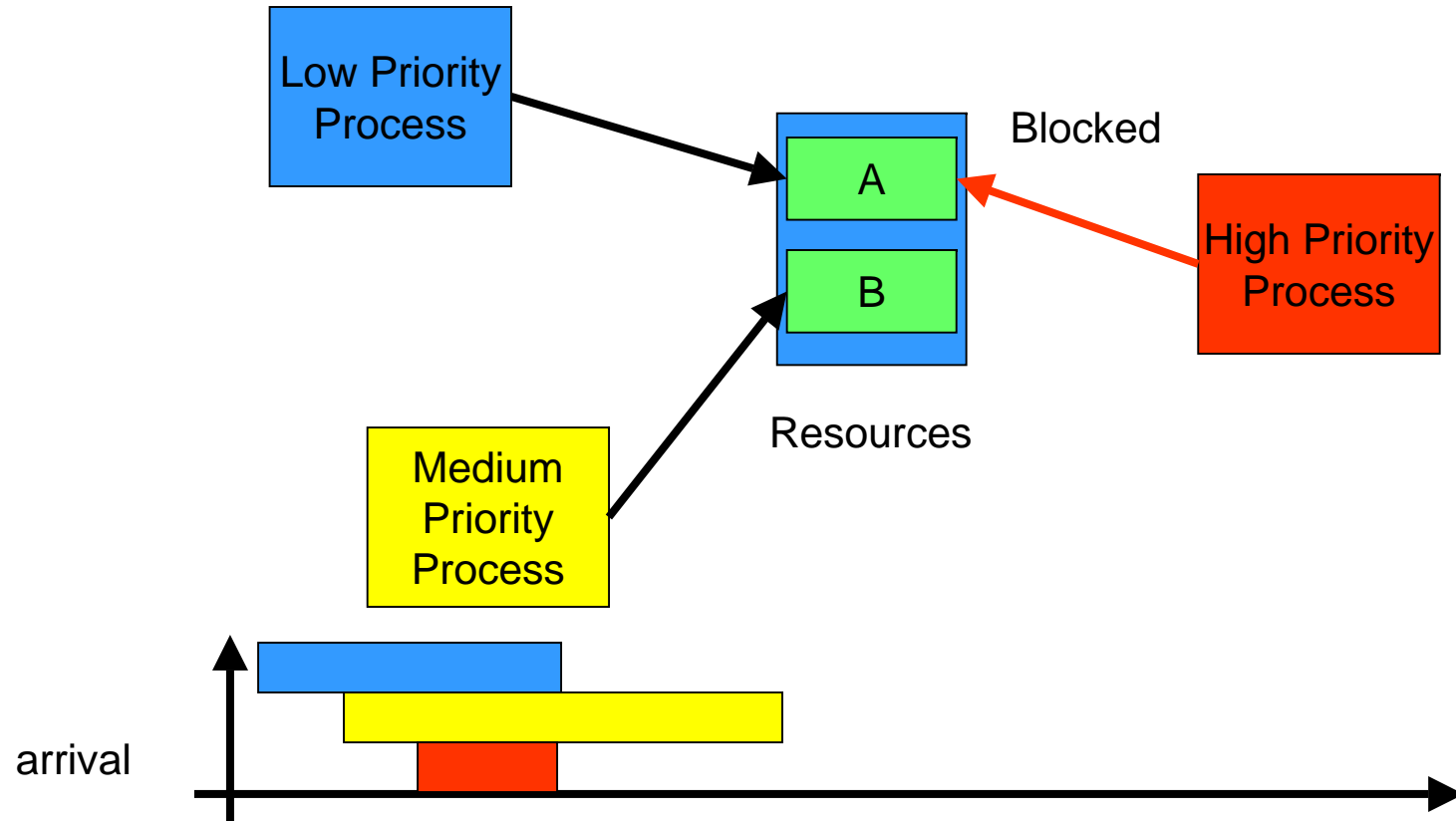
# Task Priority Inversion



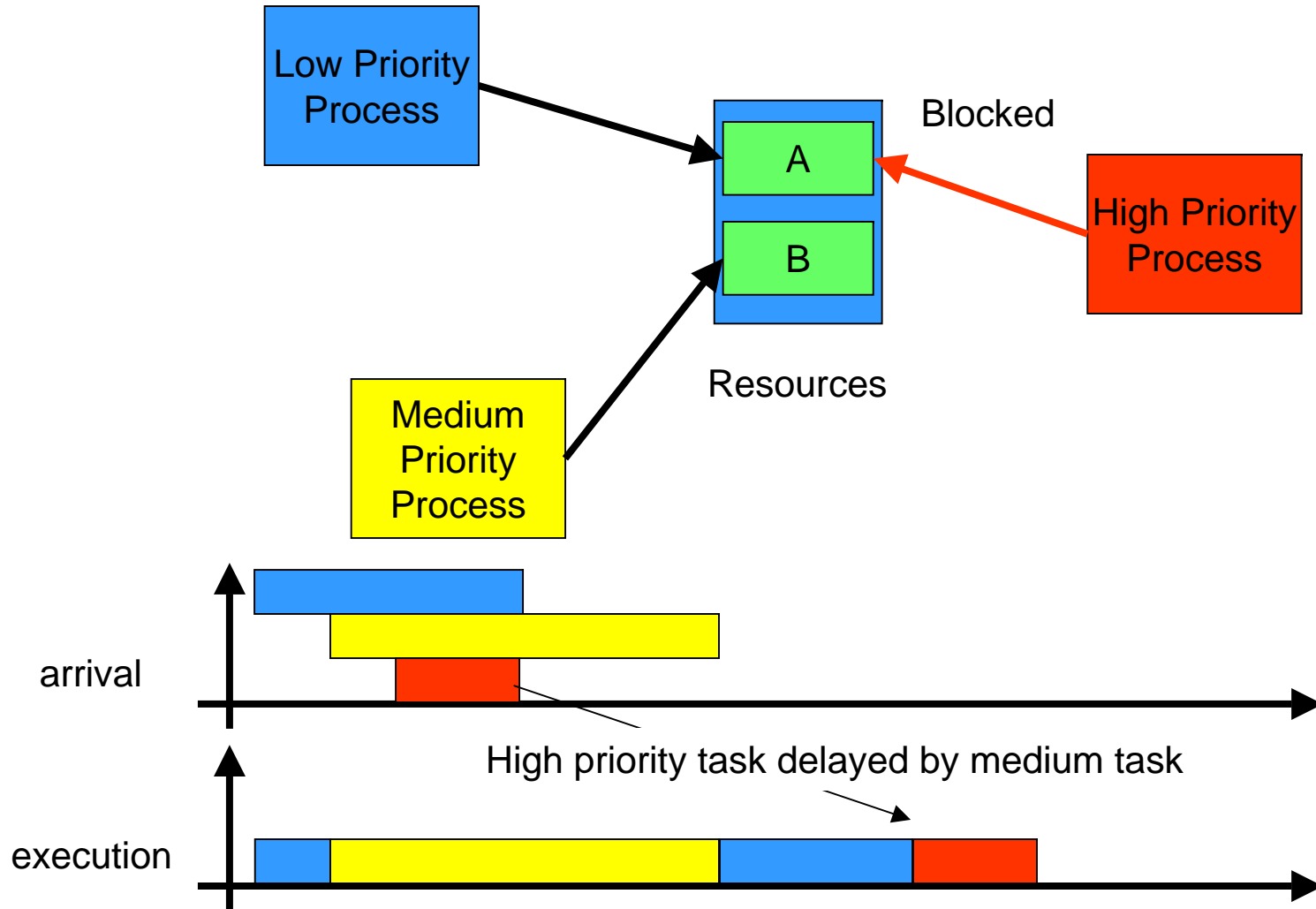
# Task Priority Inversion



# Task Priority Inversion



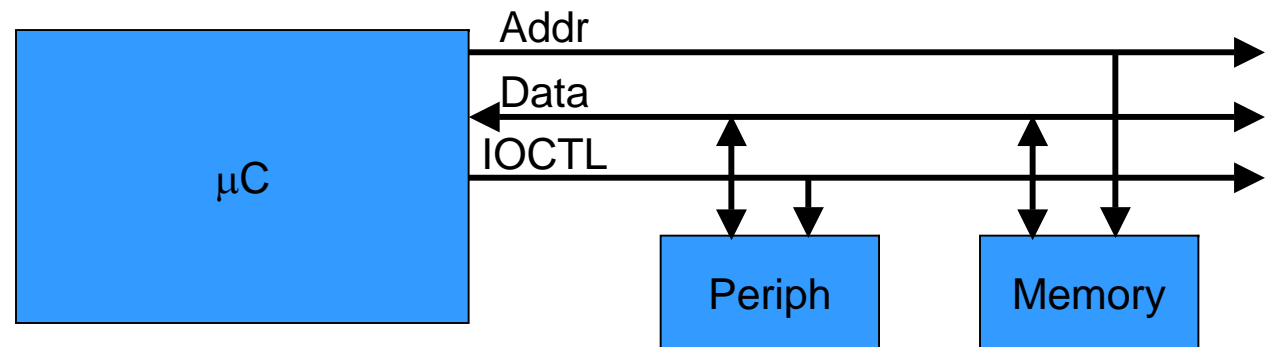
# Task Priority Inversion



# Memory Mapped I/O vs. I/O Ports

- I/O Ports

LDA ADDR :Get data at ADDR to A  
OUT PORT :Output data from A reg  
.  
.  
IN PORT :Load port to A  
STA ADDR :Store data



# Memory Mapped I/O vs. I/O Ports

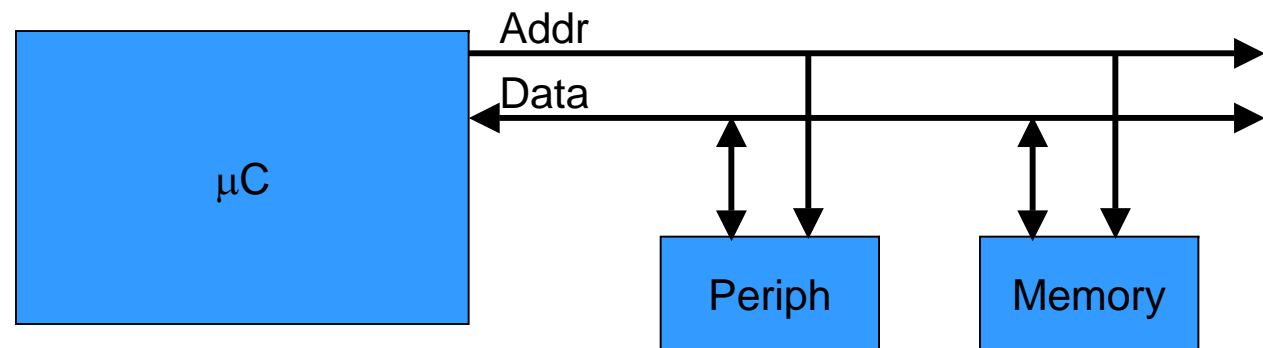
- Memory Mapped I/O

LDA ADDR :Load A from ADDR

•

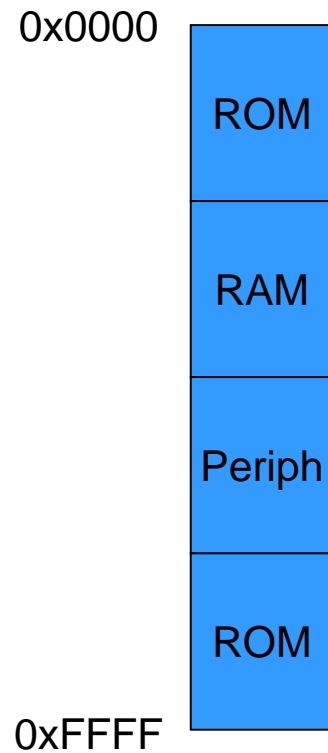
•

STA ADDR :Write A to ADDR



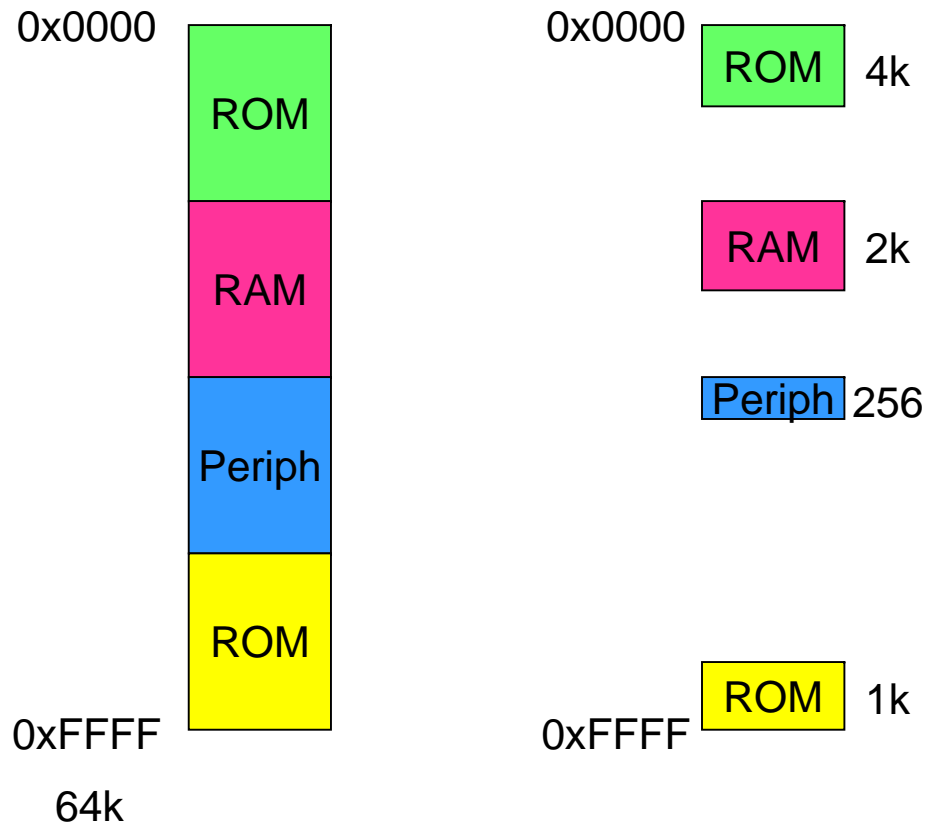
# Architectural Implications of Limited Hardware

- Example memory map:



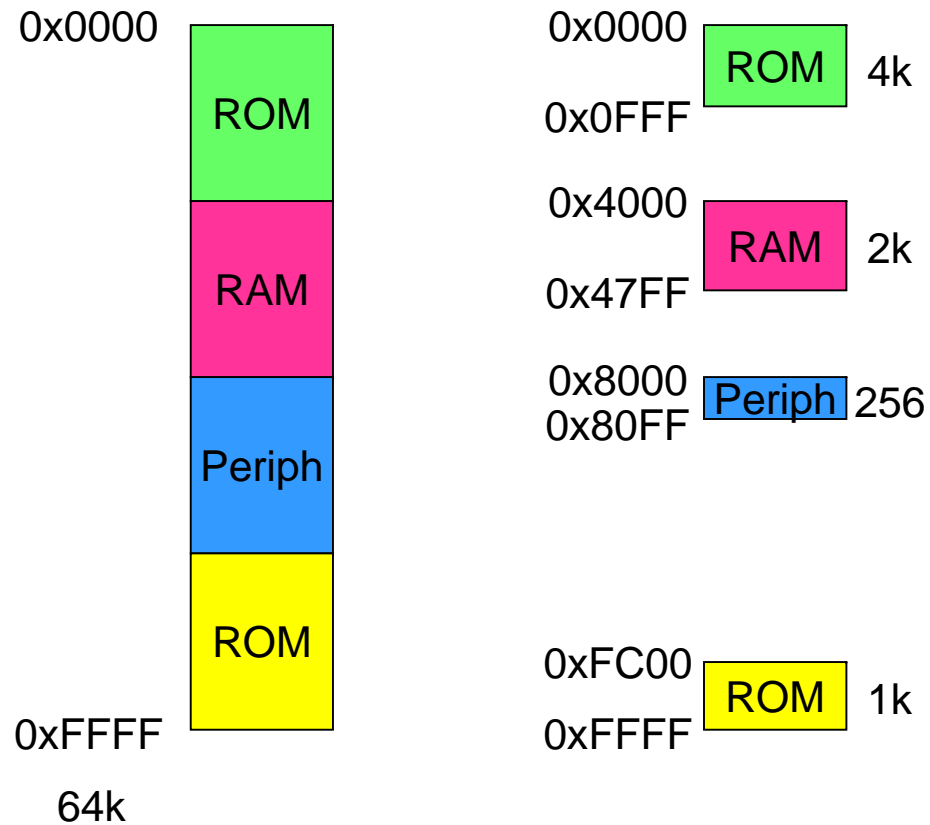
# Architectural Implications of Limited Hardware

- Example memory map:



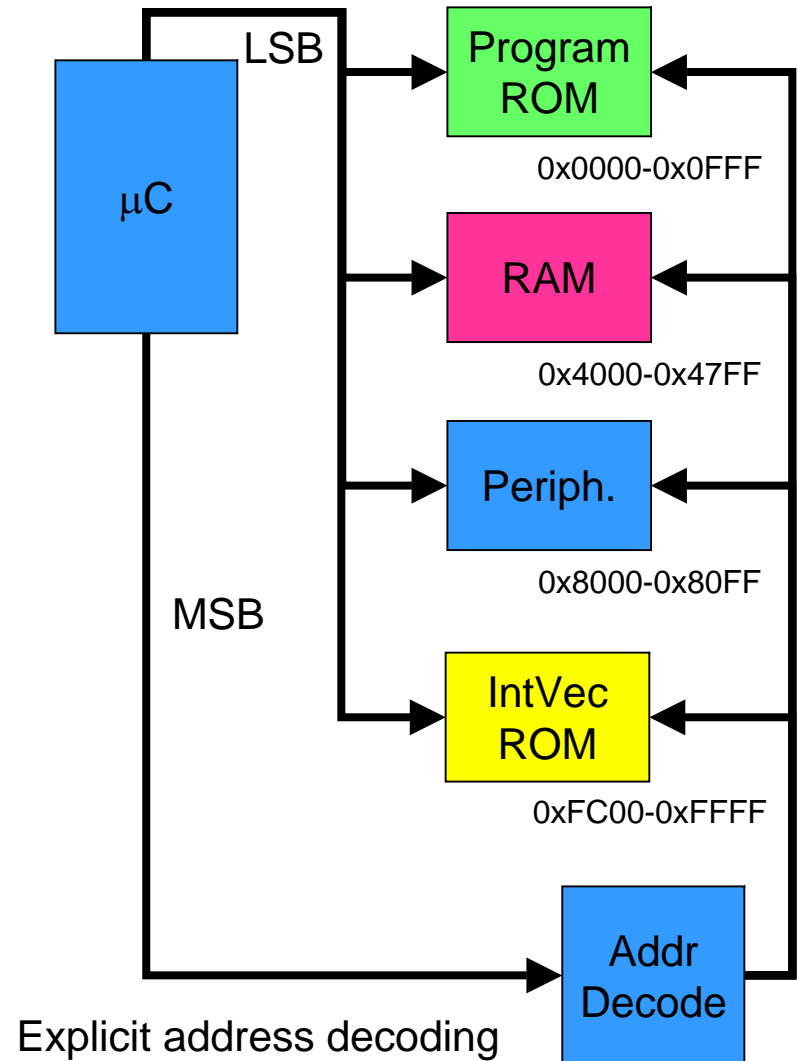
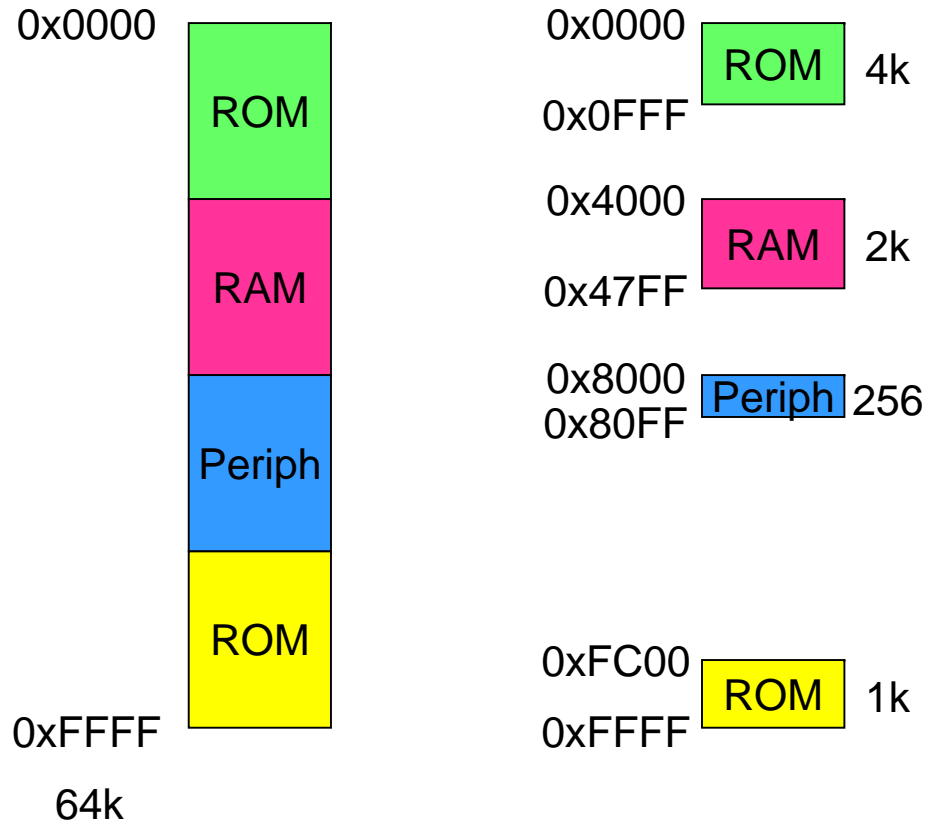
# Architectural Implications of Limited Hardware

- Example memory map:



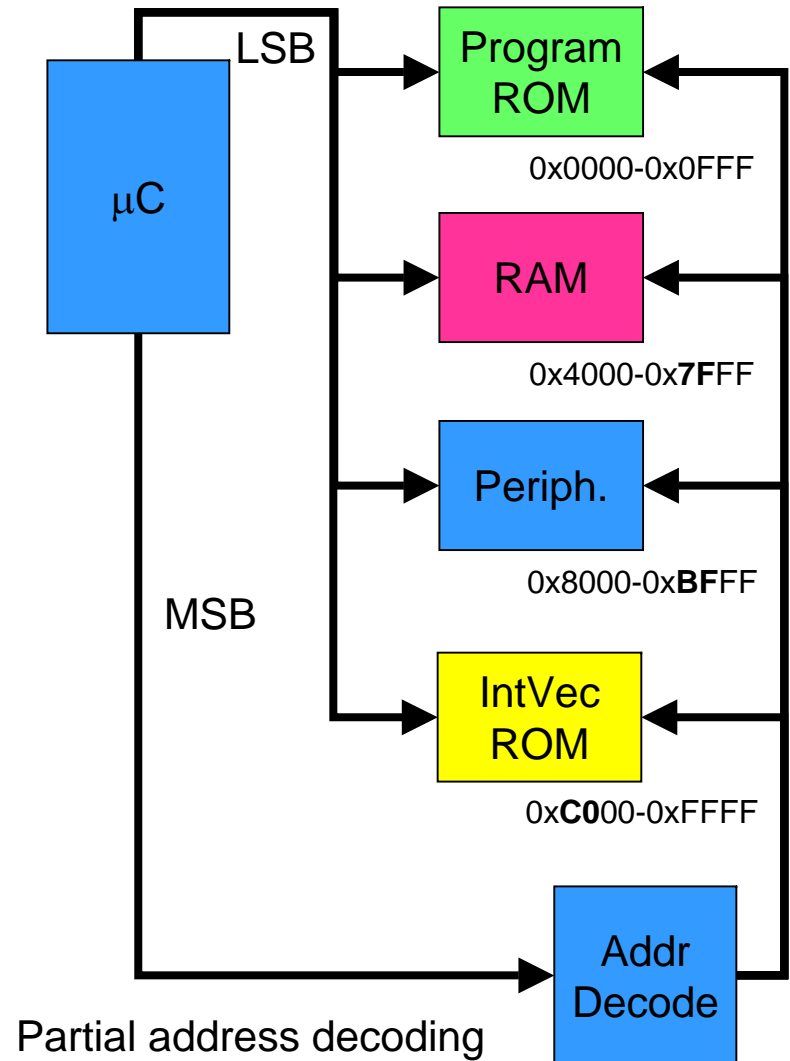
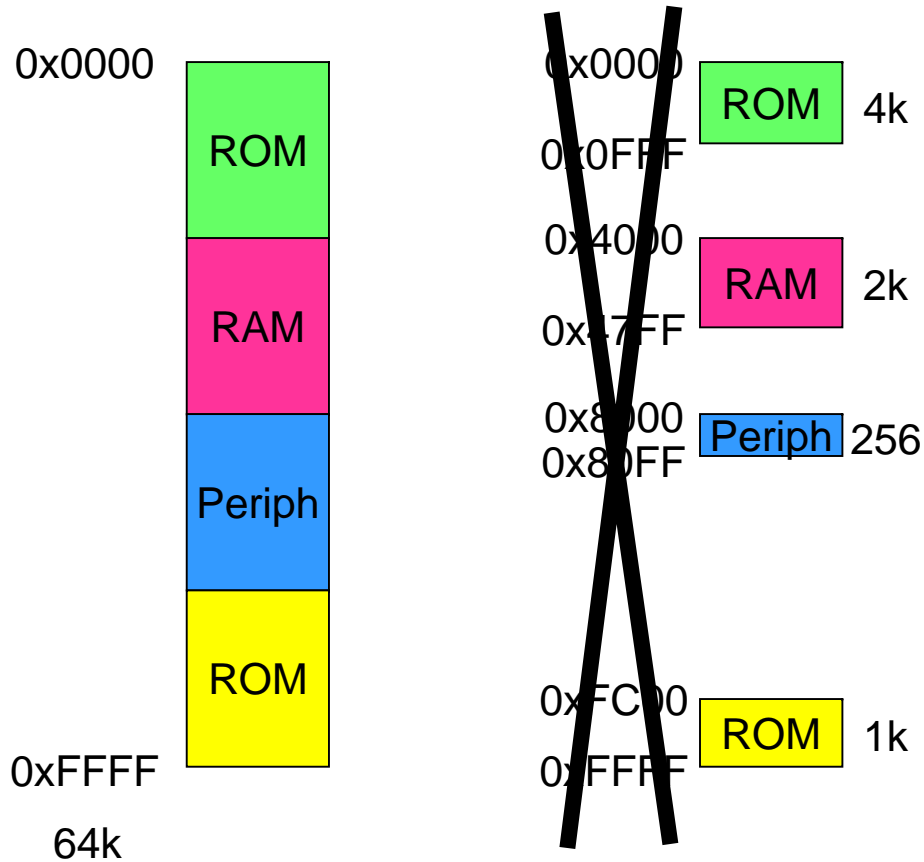
# Architectural Implications of Limited Hardware

- Example memory map:



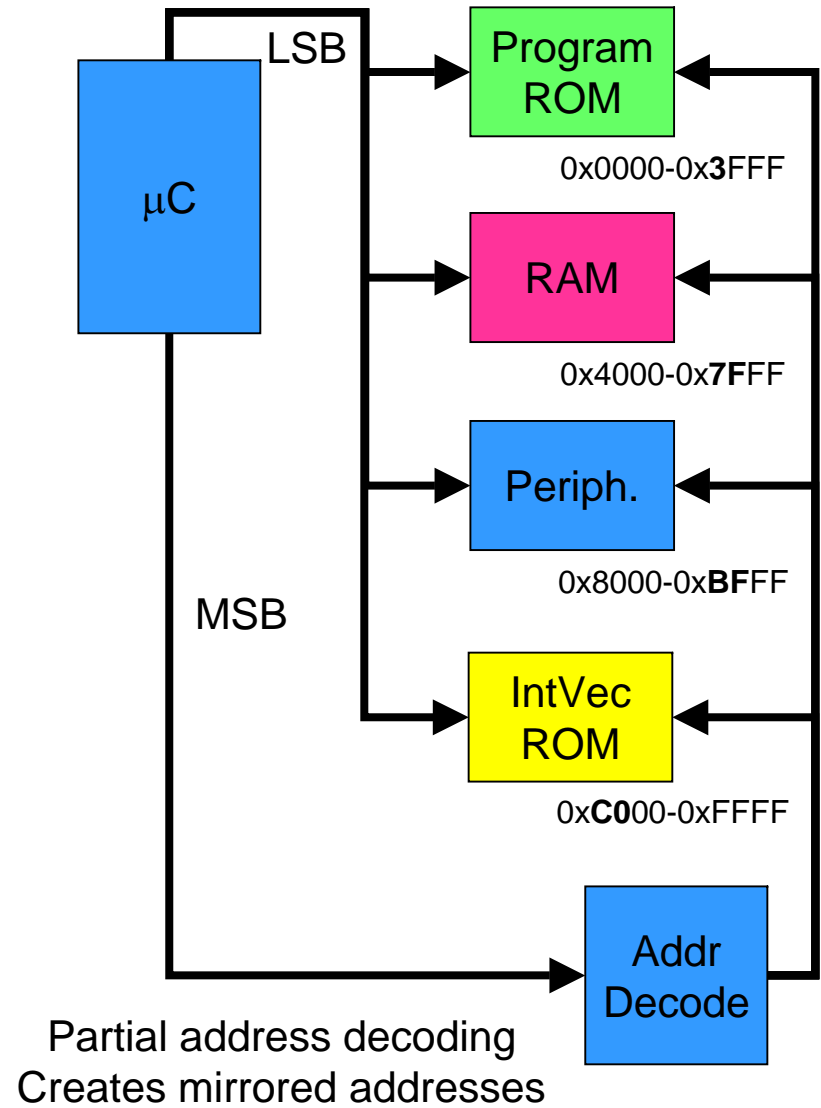
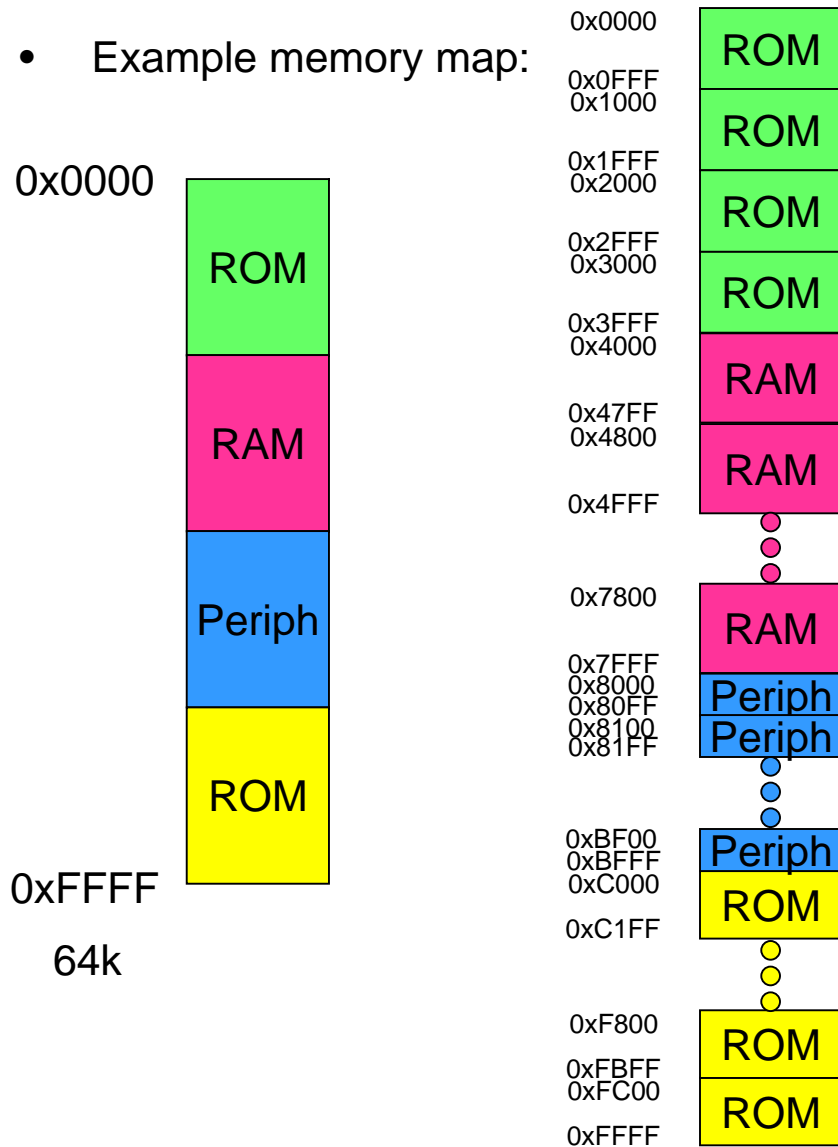
# Architectural Implications of Limited Hardware

- Example memory map:

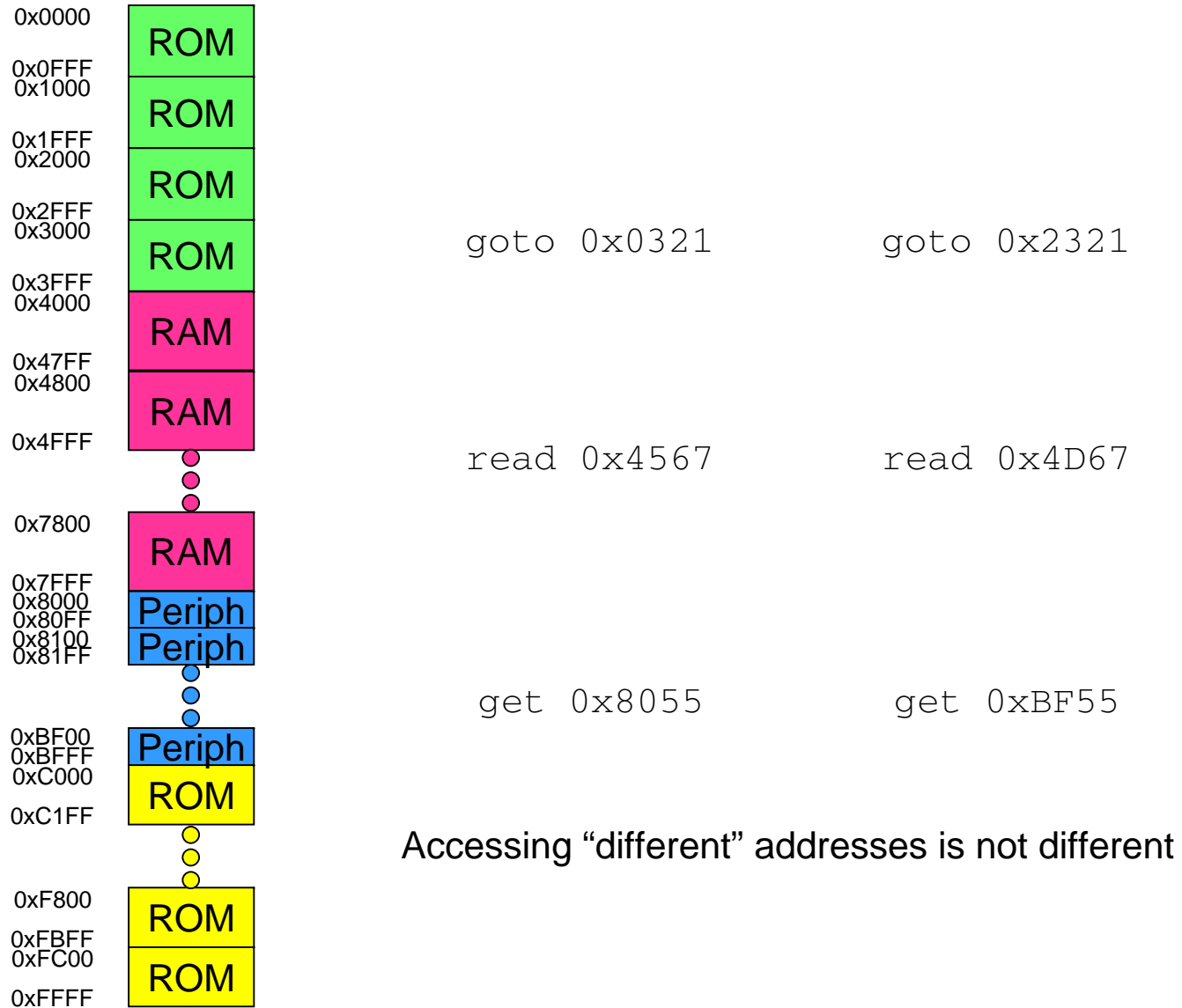


# Architectural Implications of Limited Hardware

- Example memory map:

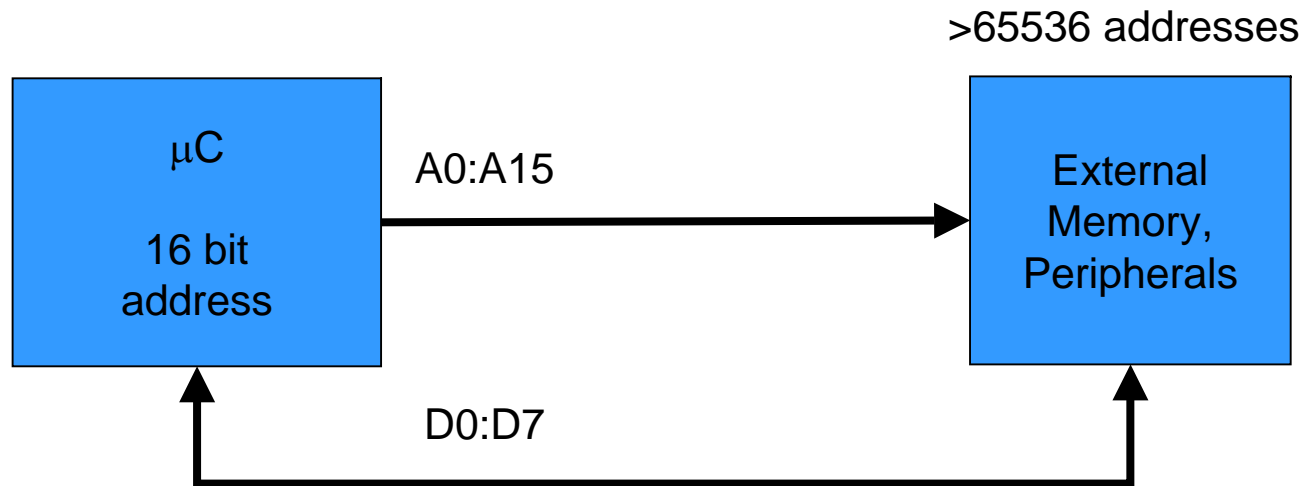


# Effects of Mirrored Memory Addresses



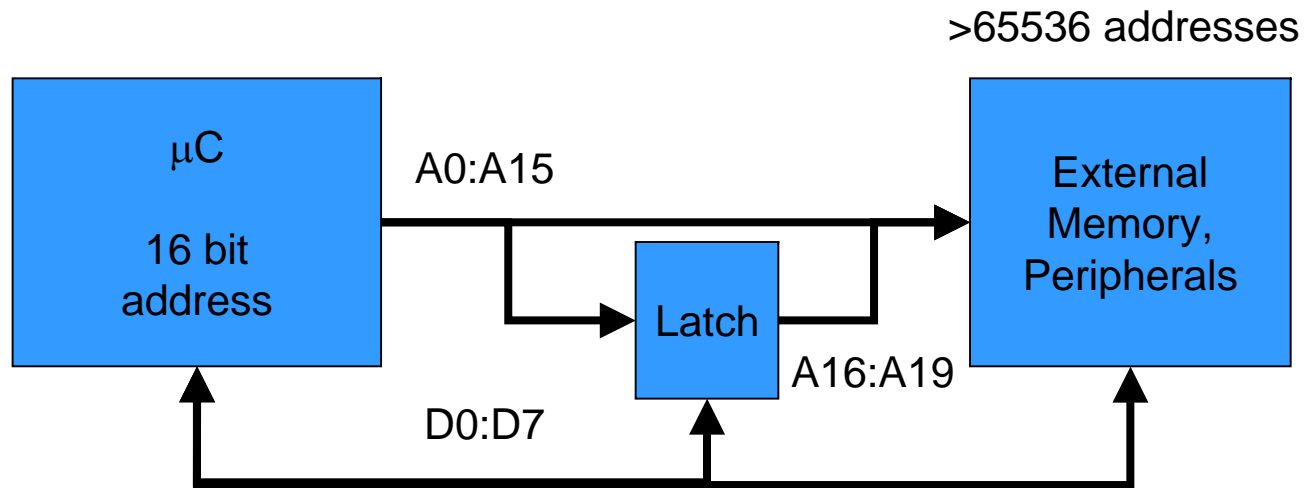
# Dealing with Insufficient Address Space

- Programs, RAM, and peripherals may require additional address space



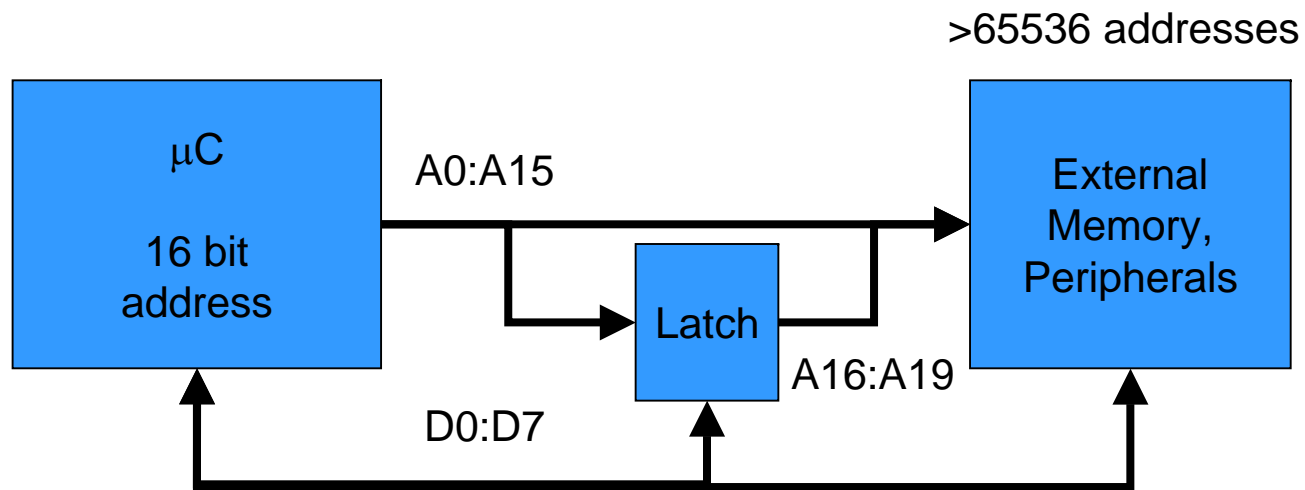
# Dealing with Insufficient Address Space

- Programs, RAM, and peripherals may require additional address space
- Use bank selection



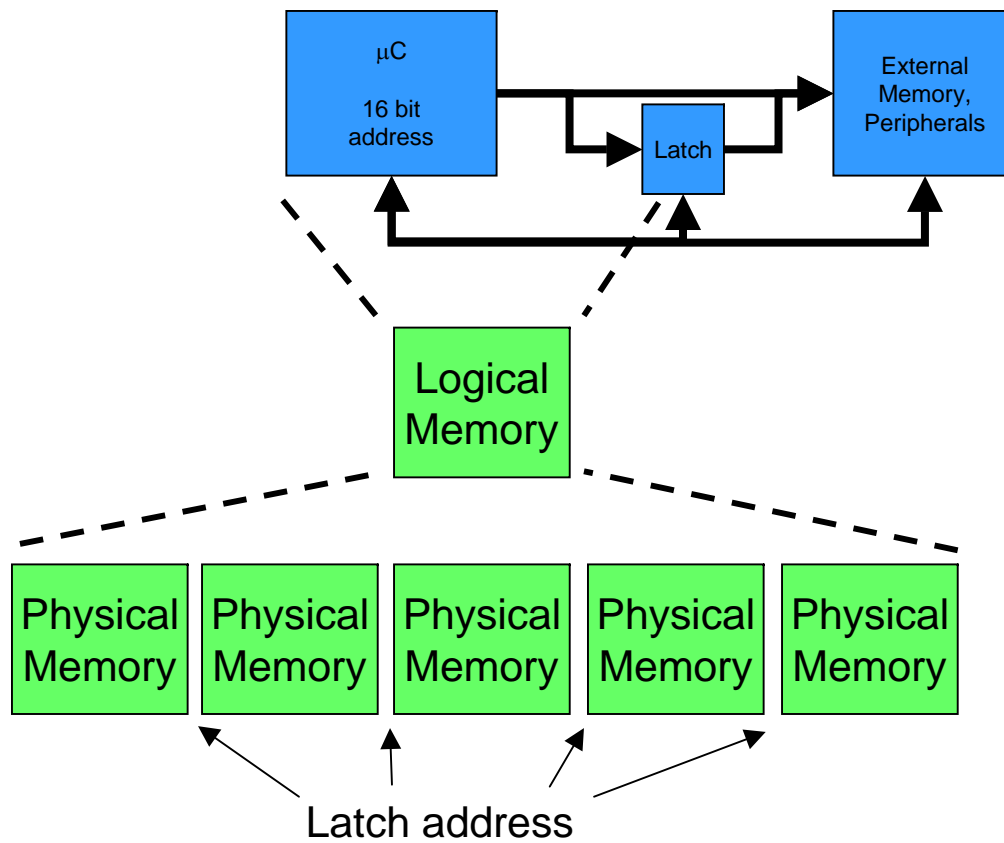
# Dealing with Insufficient Address Space

- Programs, RAM, and peripherals may require additional address space
- Use bank selection



Latch occupies 1 or more address locations  
Creates a larger physical address space

# Memory Paging



# Assignment 4

- The Motorola 68000 and its descendents use a linear address space while the Intel 8086 and its descendents have used a segmented address space, requiring a page register and an offset register (within a page). Research the hardware and software advantages of these approaches.