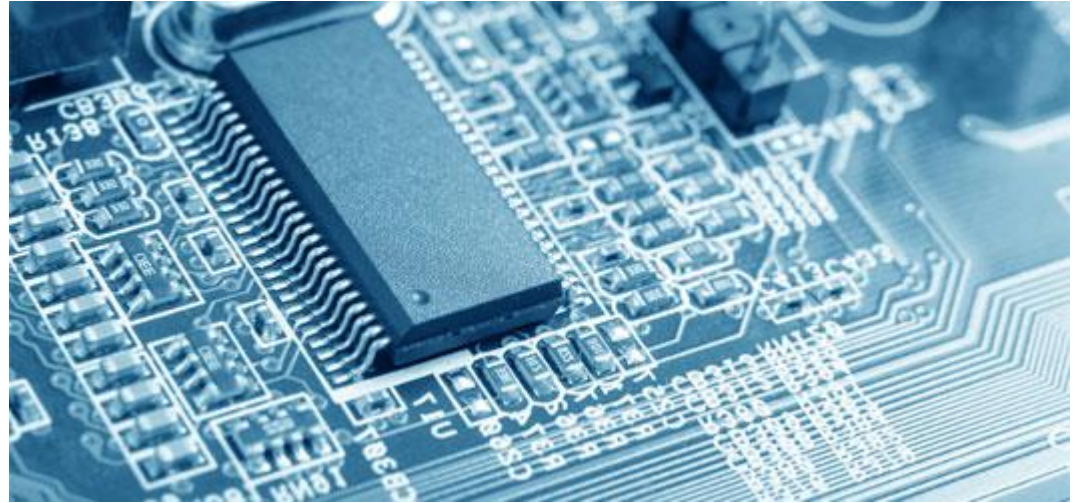




NAZARBAYEV  
UNIVERSITY  
SCHOOL OF SCIENCE AND TECHNOLOGY



# **ROBT305 - Embedded Systems**

**Lectures 14-15 – Input/Output,  
Communications, Analog/Digital Conversion**

**10-17 November, 2015**

# Course Logistics

---

## **Reading Assignment:**

**Chapter 2** of the Real-Time Systems Design and Analysis textbook

**Quiz #4** is on Thursday 19 November

**Project Assignment 1** is on Moodle due to 17 November (class demonstration)

---

# Topics

---

## Topics to Cover Today

I/O interfaces:

Asynchronous Data Transfer

Serial Communication: USB, UART

Interrupt Driven I/O

Direct Memory Access

---

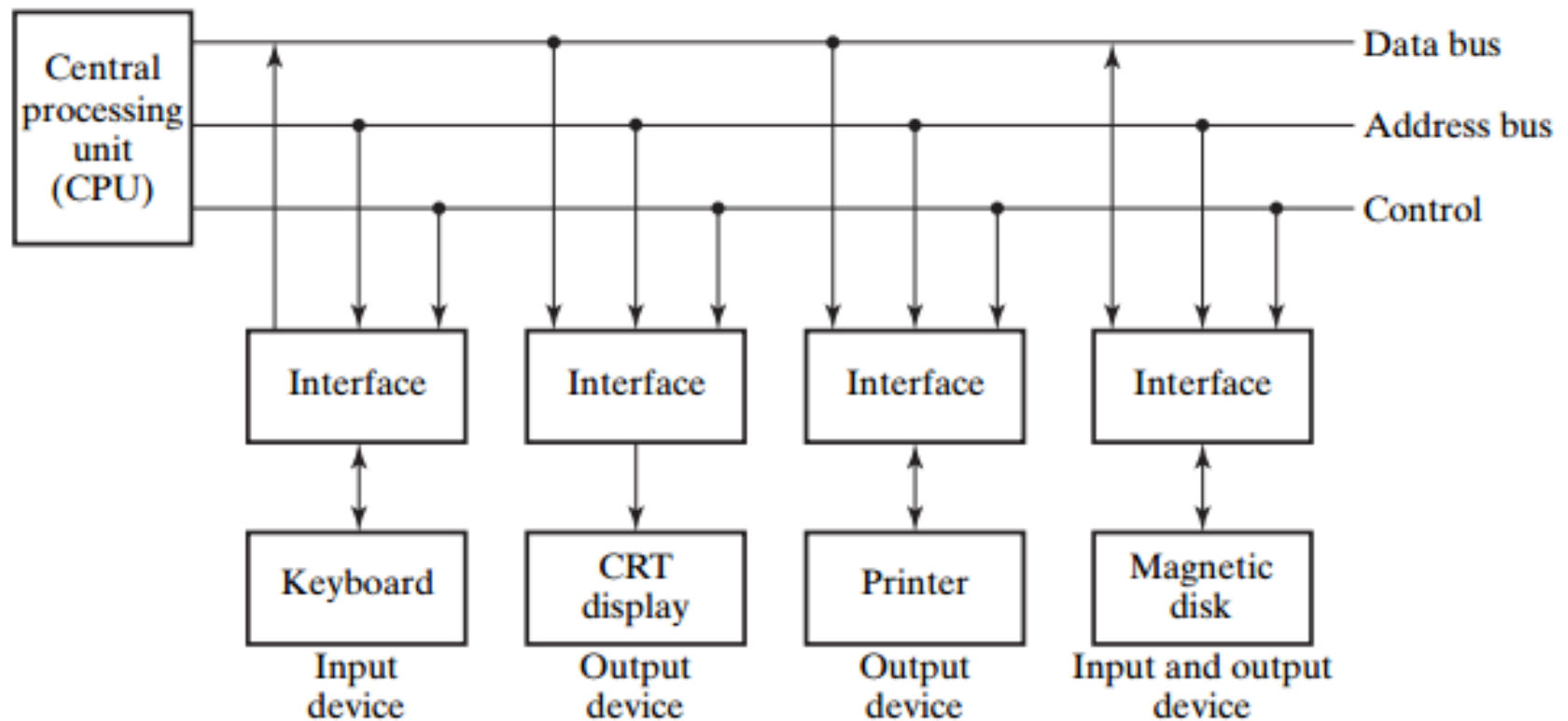
# I/O Interfaces

---

- ▶ Peripherals are often electromechanical devices whose manner of operation is different from that of CPU and memory. Therefore, a signal conversion may be required
- ▶ The data-transfer rate of peripherals is usually different from the clock rate of CPU. Hence, a synchronization mechanism may be needed.
- ▶ The overlapping modes of peripherals differ from each other, and each must be controlled in a way that does not disturb the operation of other peripherals connected to CPU
- ▶ Computer system include interface units between CPU and the peripherals.

# I/O Bus and Interface Unit

- ▶ A typical communication structure between CPU and several peripherals

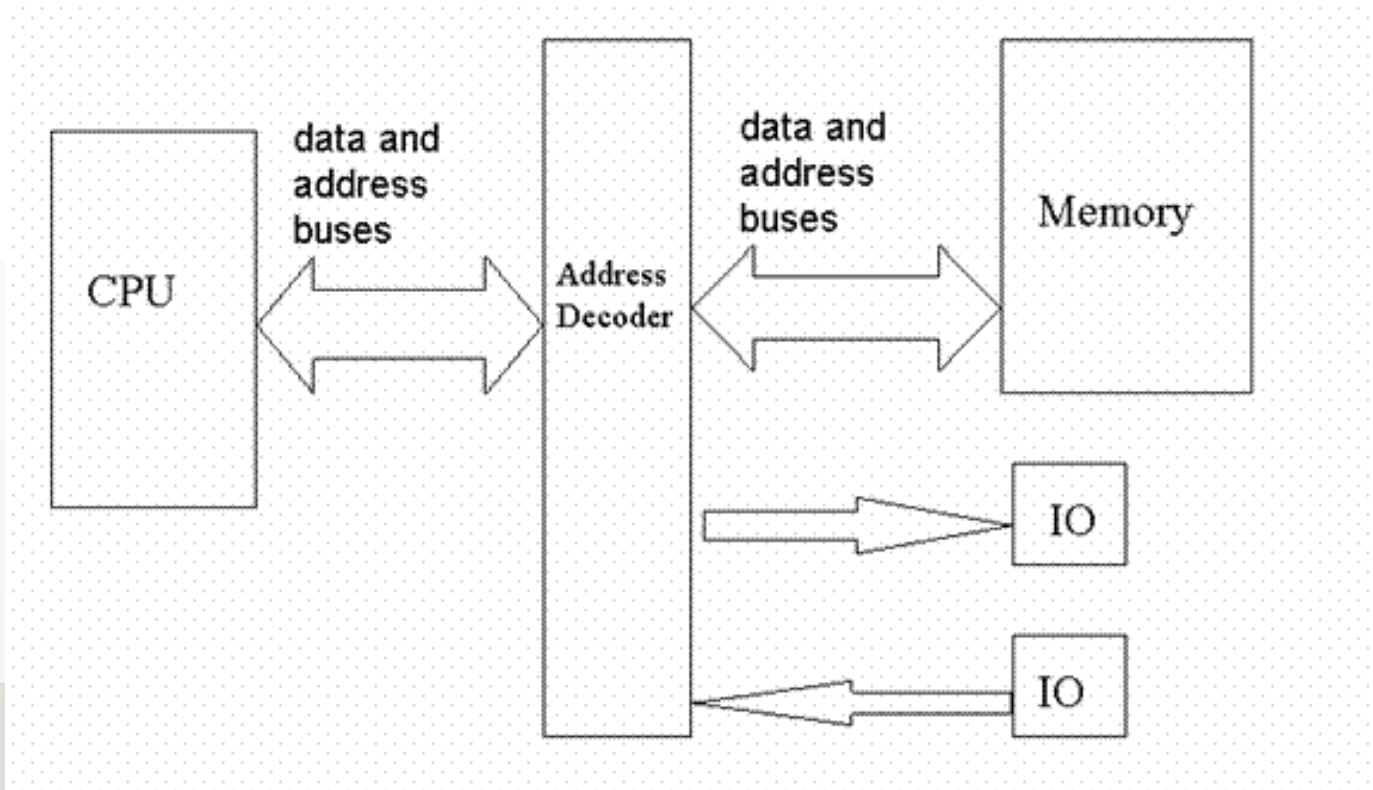


# Memory-Mapped I/O

---

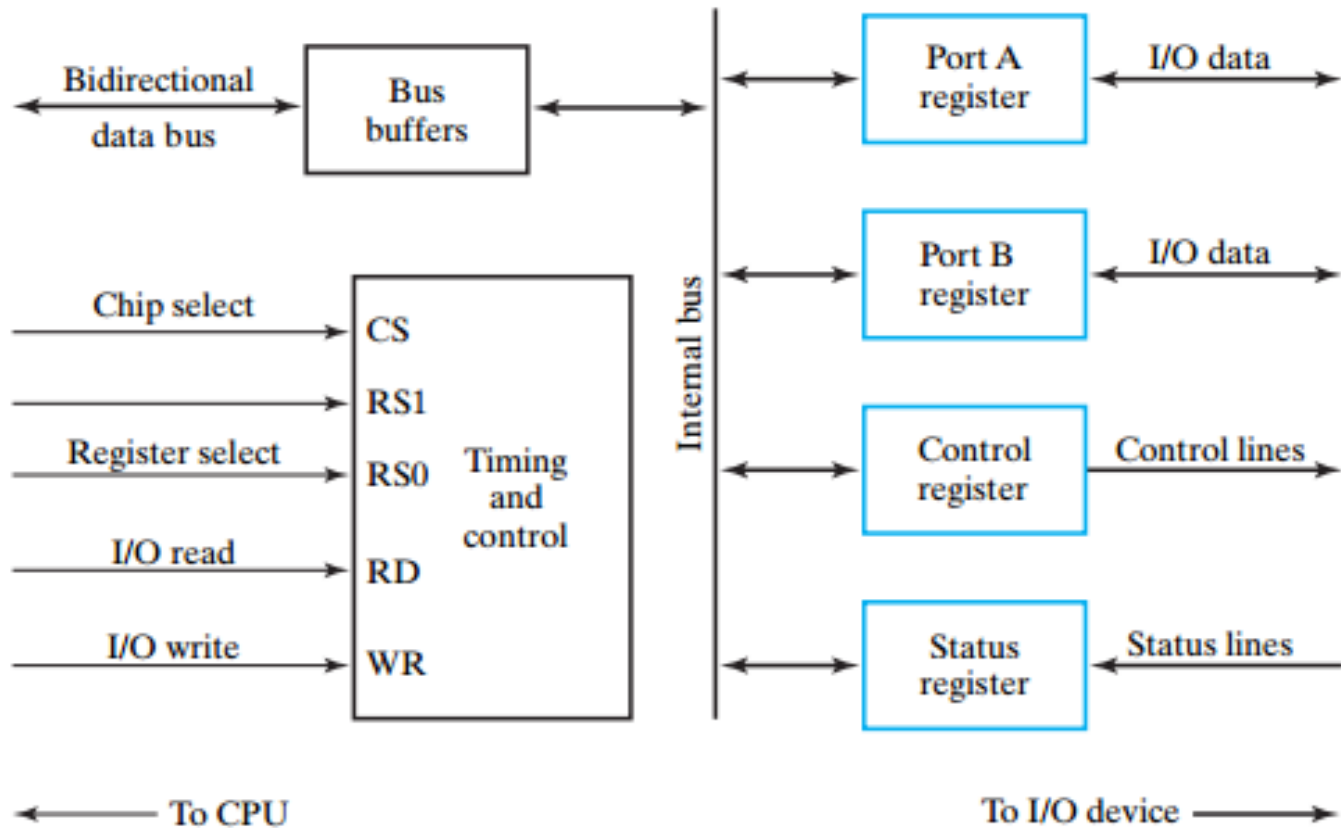
- ▶ In addition to communicating with I/O devices, CPU must communicate with memory unit through an address and data bus
- ▶ Memory-mapped I/O provides a data transfer mechanism that is convenient because it does not require the use of special CPU I/O instructions.
- ▶ In memory-mapped I/O certain designated locations of memory appear as virtual input/output ports.
- ▶ Could be advantageous when implementing efficient device drivers

# Memory-Mapped I/O



Input from an appropriate memory-mapped location involves executing a LOAD instruction on a pseudomemory location connected to an input device. Output uses a STORE instruction.

# Example I/O Interface



CS	RS1	RS0	Register selected
0	X	X	None: data bus in high-impedance state
1	0	0	Port A register
1	0	1	Port B register
1	1	0	Control register
1	1	1	Status register

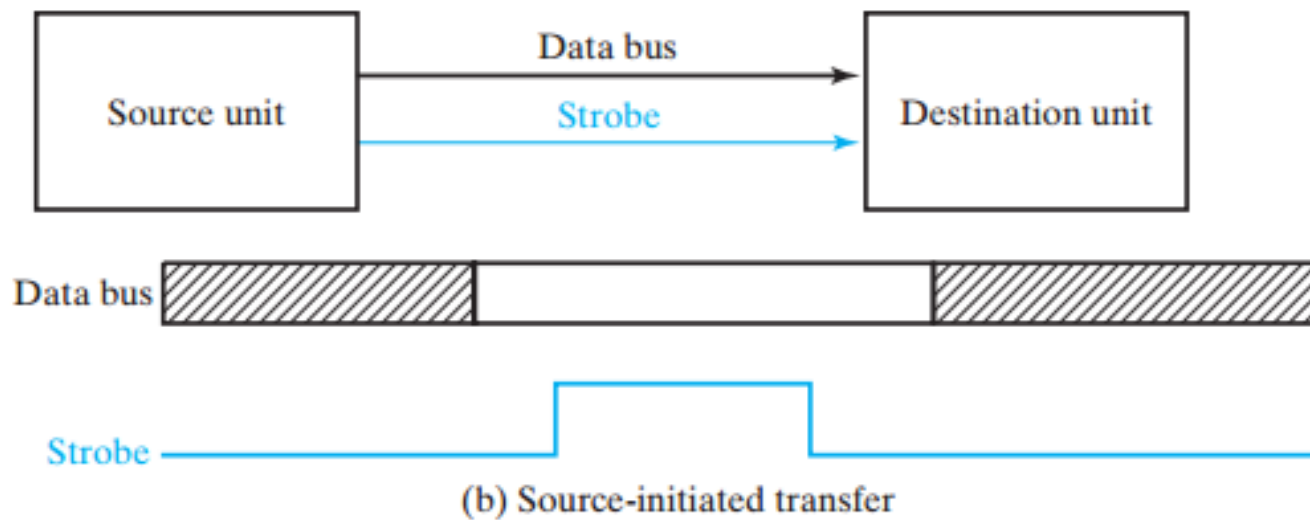
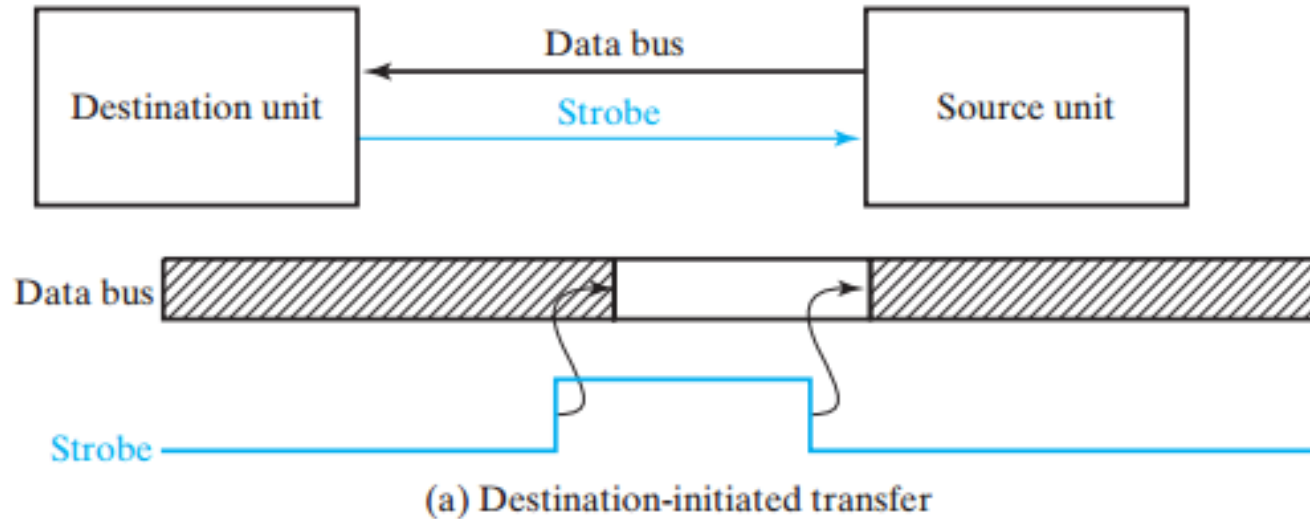


# Asynchronous Data Transfer

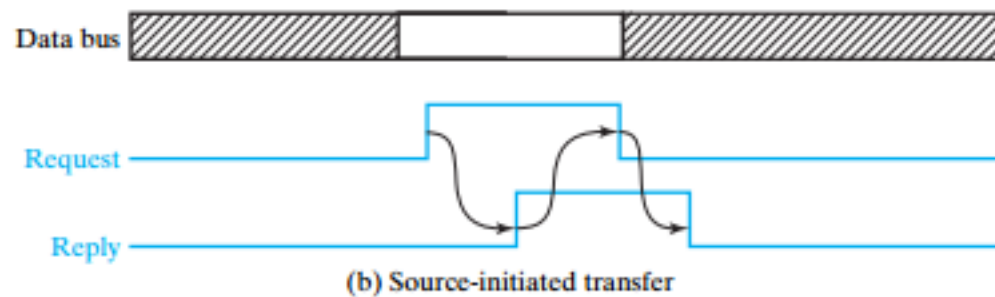
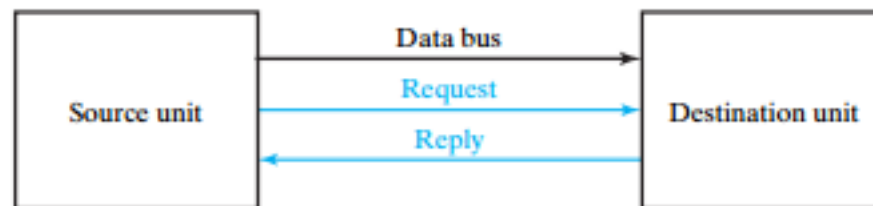
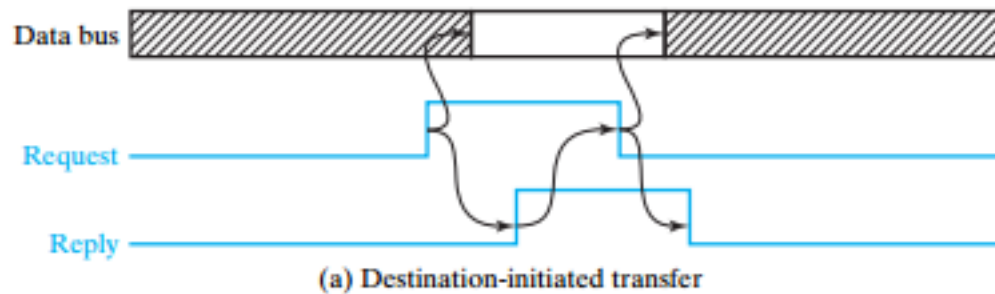
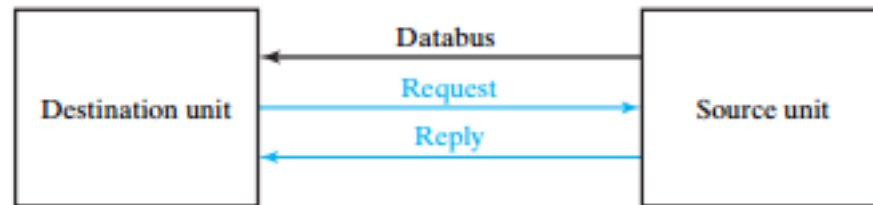
---

- CPU, interface and I/O devices are likely to have different clocks that are not synchronized - **asynchronous** units
- Asynchronous data transfer requires that control signals be transmitted between the units to indicate transmission time
- Two methods for implementing this timing:
  - Strobing
  - Handshaking

# Strobing



# Handshaking



# Serial Communication

---

- ▶ Transfer can be parallel or serial.
- ▶ In serial data communication, each bit in the message is sent in sequence, one at a time.
- ▶ Requires the use of one or two signal lines.
- ▶ Can be asynchronous or synchronous communication
- ▶ In asynchronous transmission binary information is sent only when it is available, and the line remains idle when there is no transmitted data. Example – USB, UART
- ▶ Synchronous transmission transmits bits continuously to keep the clock frequencies in both units synchronized

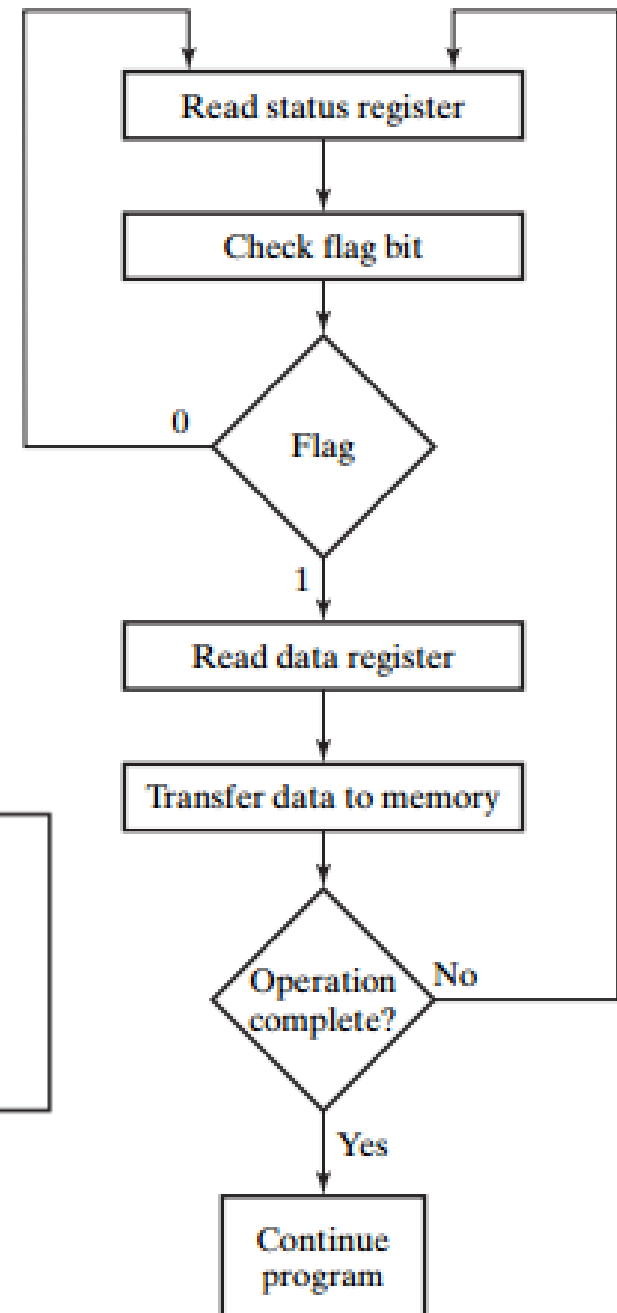
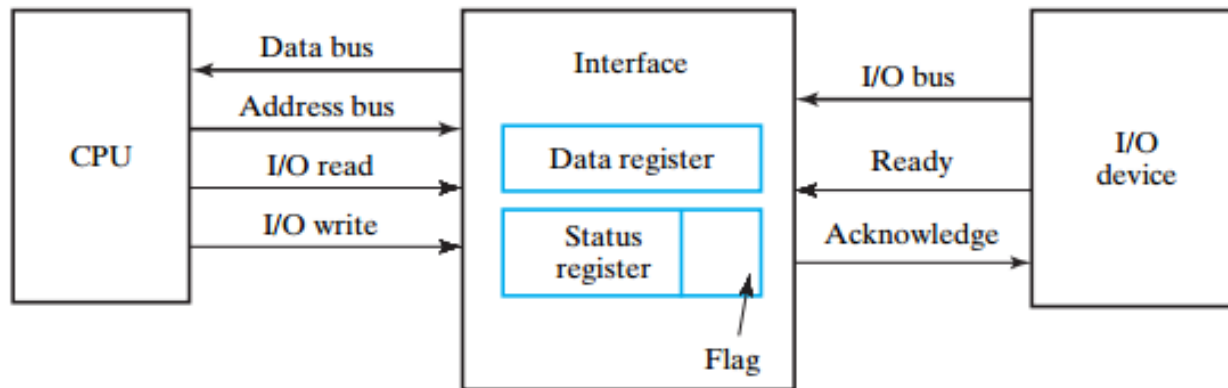
# Peripheral Interfacing

---

- ▶ Peripheral, sensor and actuator interfacing developing much slower than memory and processor architectures
  - ▶ Fundamental practices for I/O handling are the same:
    - ▶ Polled I/O (program-controlled transfer)
    - ▶ Interrupt-driven I/O
    - ▶ Direct memory access
  - ▶ **In polled I/O system** the status of I/O device is checked periodically, or regularly.
  - ▶ Advantage - simplicity
  - ▶ Disadvantage – CPU loading with unnecessary status requests
-

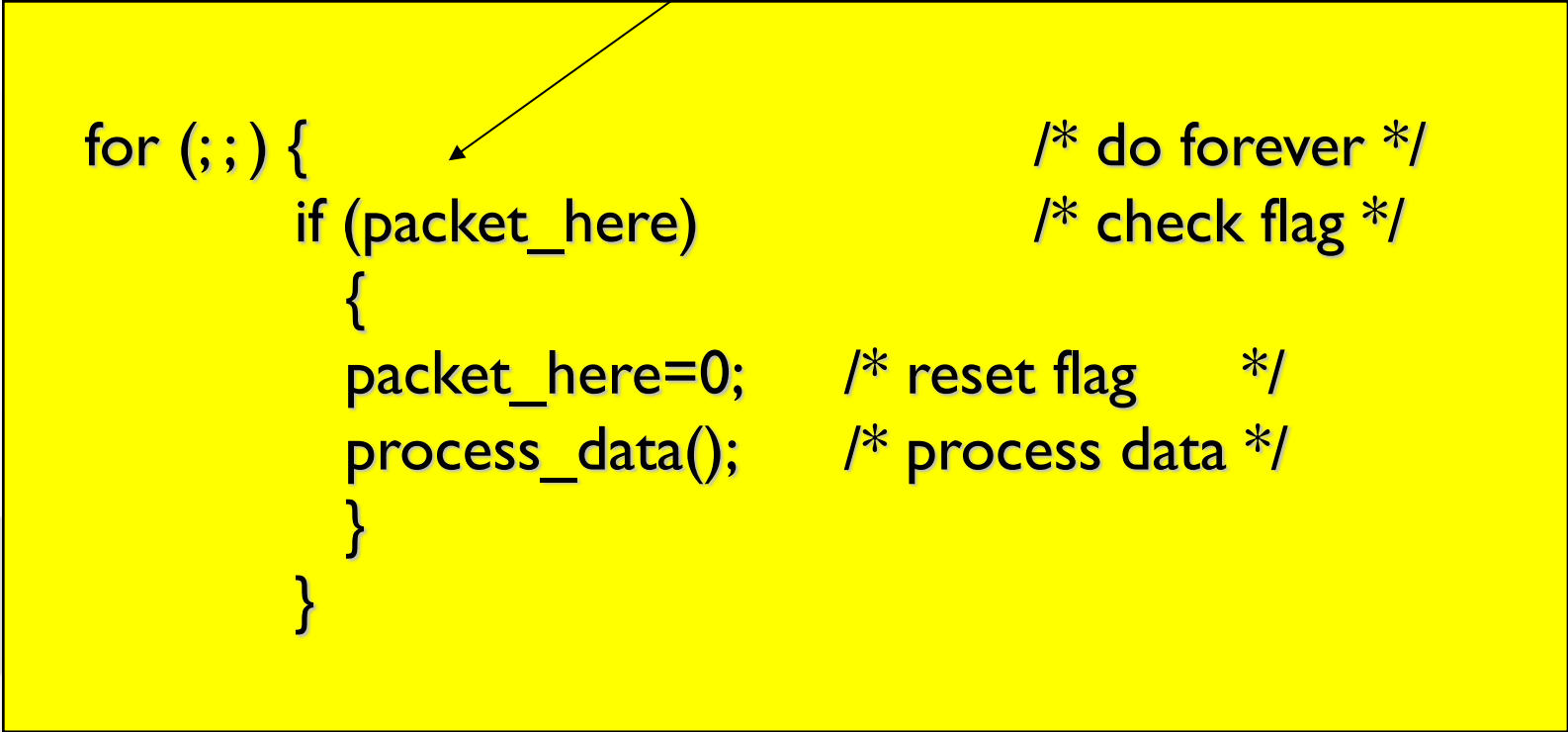
# Polled loop systems

- ▶ CPU check the flag to define whether there is new byte in the interface data register or not.
- ▶ Assume the input device transmits data at 100 bytes/s or 1 byte every 10 msec.
- ▶ If CPU checks the flag in 100 nsec than CPU will check the flag 100000 times between each transfer



# Polled loop systems

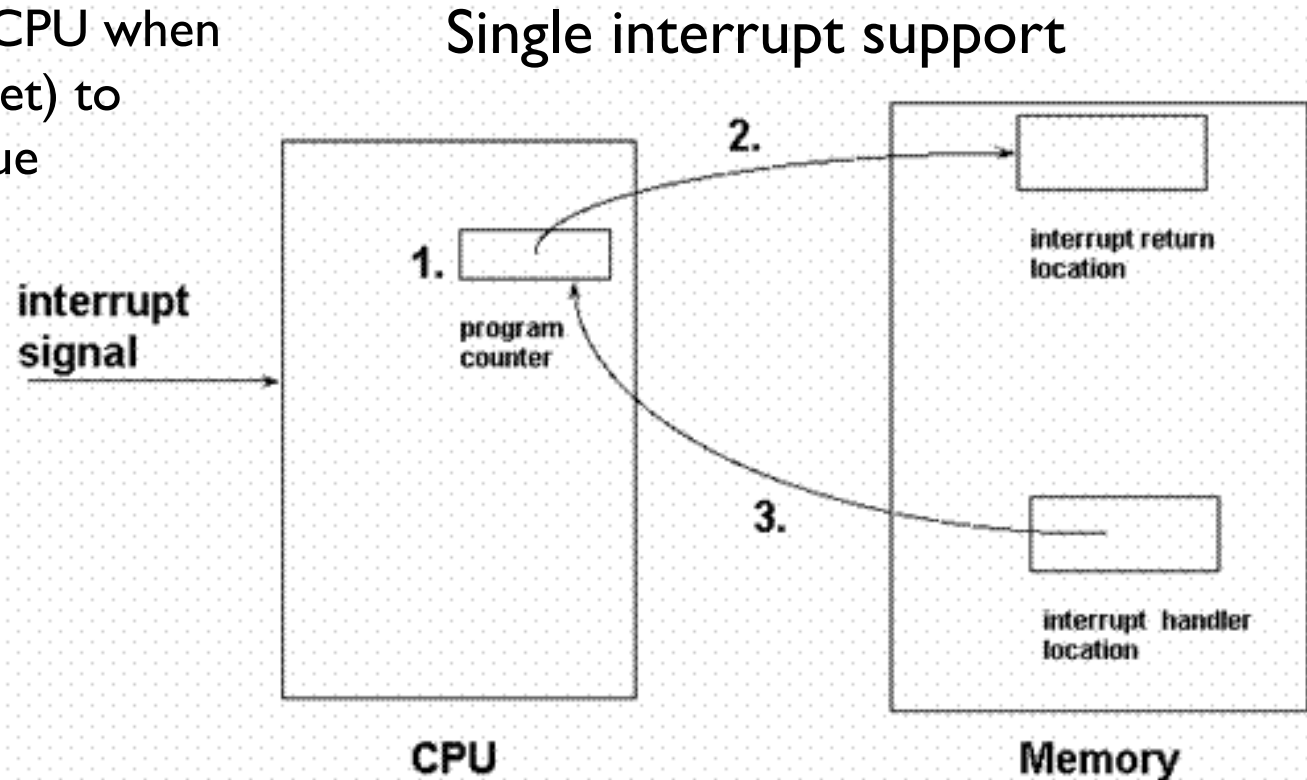
Hardware causes event via memory mapped I/O, DMA or a pin input



```
for (;;) {                                /* do forever */
    if (packet_here)                      /* check flag */
    {
        packet_here=0;                  /* reset flag */
        process_data();                 /* process data */
    }
}
```

# Interrupt Driven Input/Output

- Interface informs CPU when it is ready (flag is set) to transfer data – issue interrupt request



Step 1: finish the currently executing macroinstruction.

Step 2: save the contents of the program counter to the interrupt return location.

Step 3: load the address held in the interrupt handler location into the program counter. Resume the fetch and execute sequence.

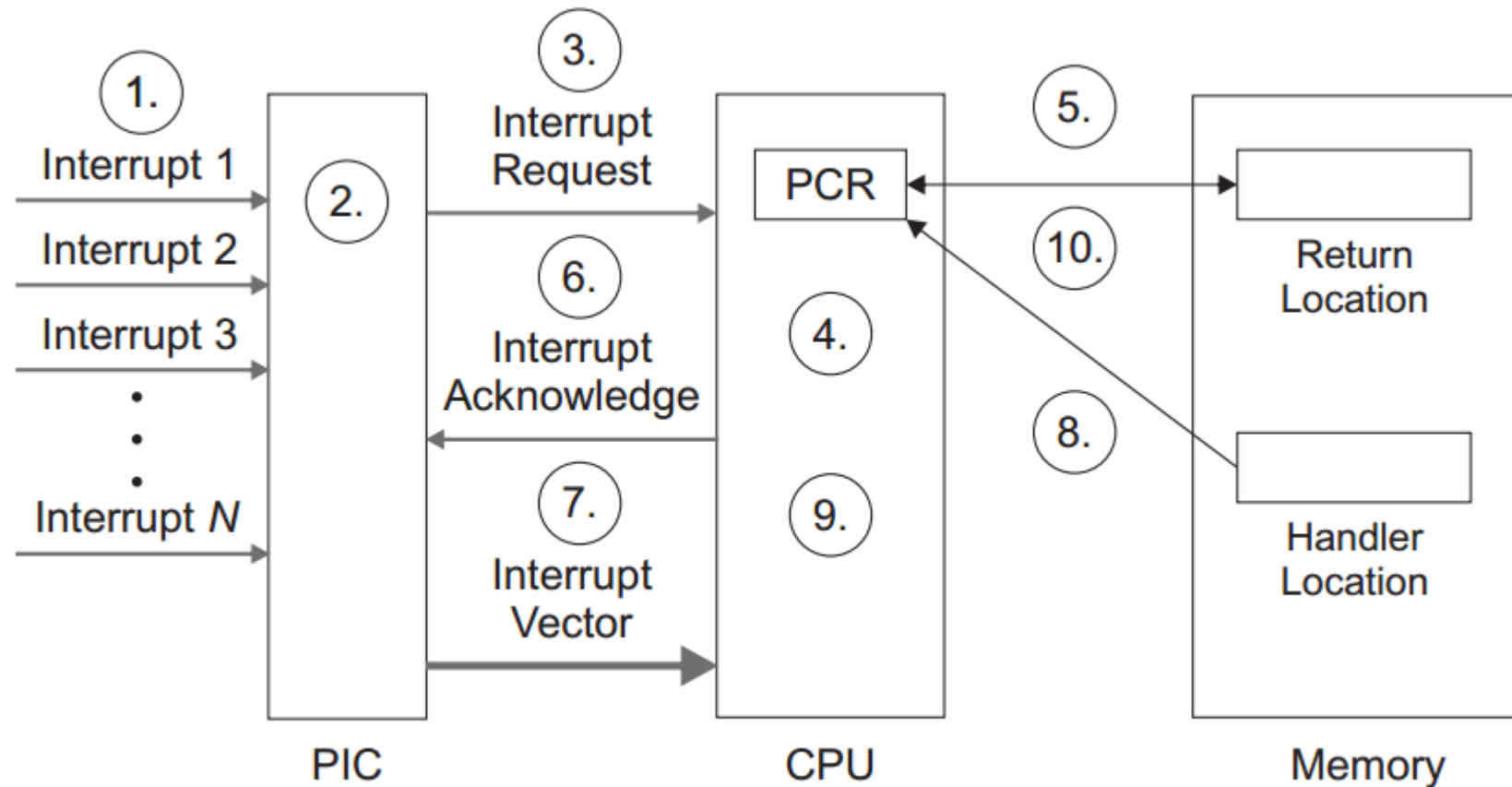


# Priority Interrupt Controller (PIC)

---

1. The PIC receives several simultaneous interrupt requests
2. The PIC processes first the request with highest priority
3. The CPU receives an interrupt request from the PIC
4. The CPU completes the currently executing instruction
5. The CPU stores the content of the PC to memory
6. The CPU acknowledges the interrupt to the PIC
7. The PIC sends the interrupt vector to the CPU
8. The CPU loads the corresponding interrupt- handler address to the PC
9. The CPU executes the interrupt routine
10. The CPU reloads the original program counter content (PCR) from memory

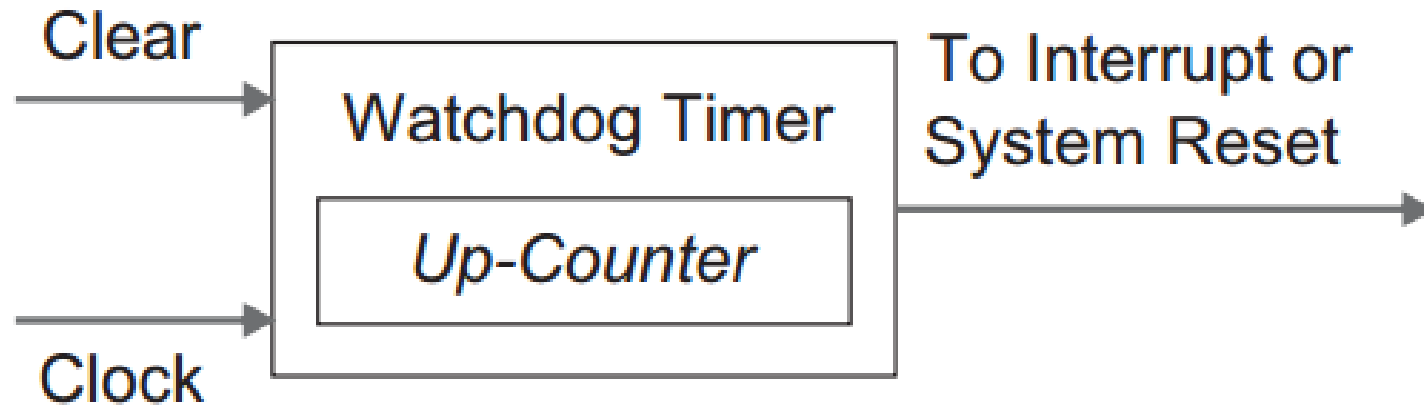
# Priority Interrupt Controller (PIC)



# Watchdog Timer

Is used in embedded systems to ensure that:

- certain devices are services at regular intervals;
- software tasks execute according to their prescribed rate;
- CPU functions normally



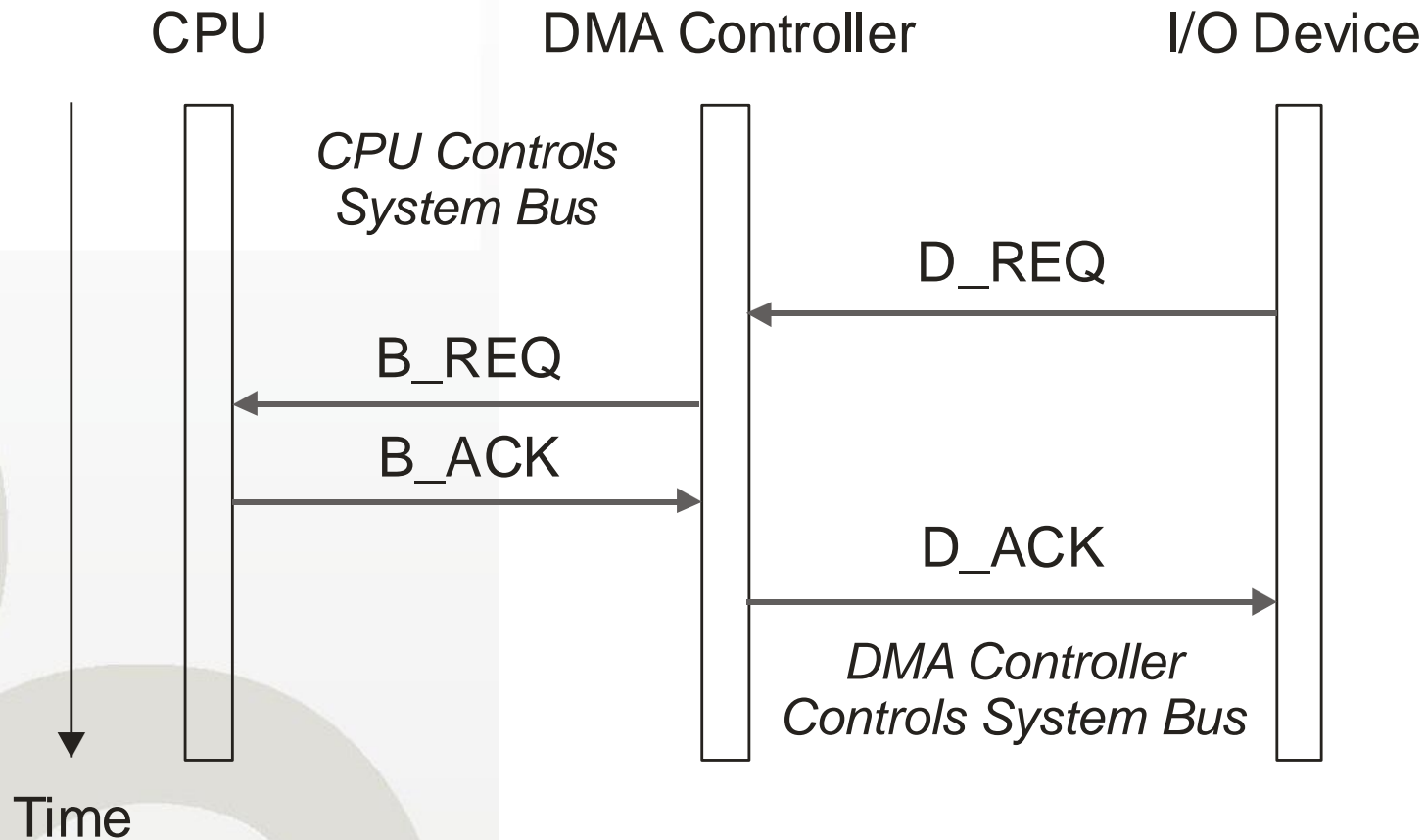
A watchdog timer. Software issues a reset signal via memory-mapped or programmed I/O to reset the timer before it can overflow, issuing a watchdog timer interrupt.

# Direct Memory Access (DMA)

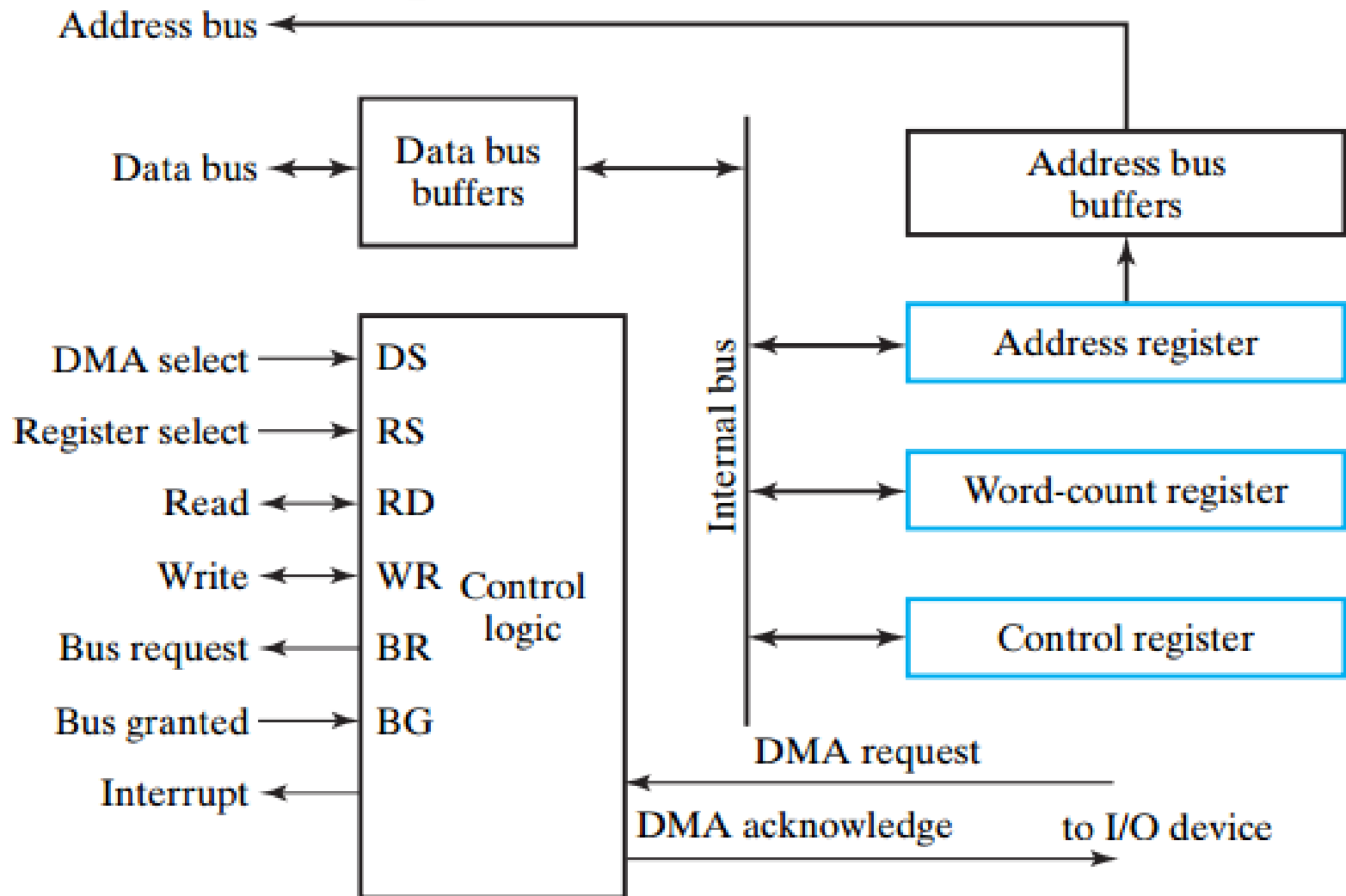
---

- ▶ Access to the computer's memory is given to other devices in the system without CPU intervention.
- ▶ Information is deposited directly into main memory by the external device.
- ▶ DMA controller is required unless the DMA circuitry is integrated into the CPU.
- ▶ Because CPU participation is not required, data transfer is fast.

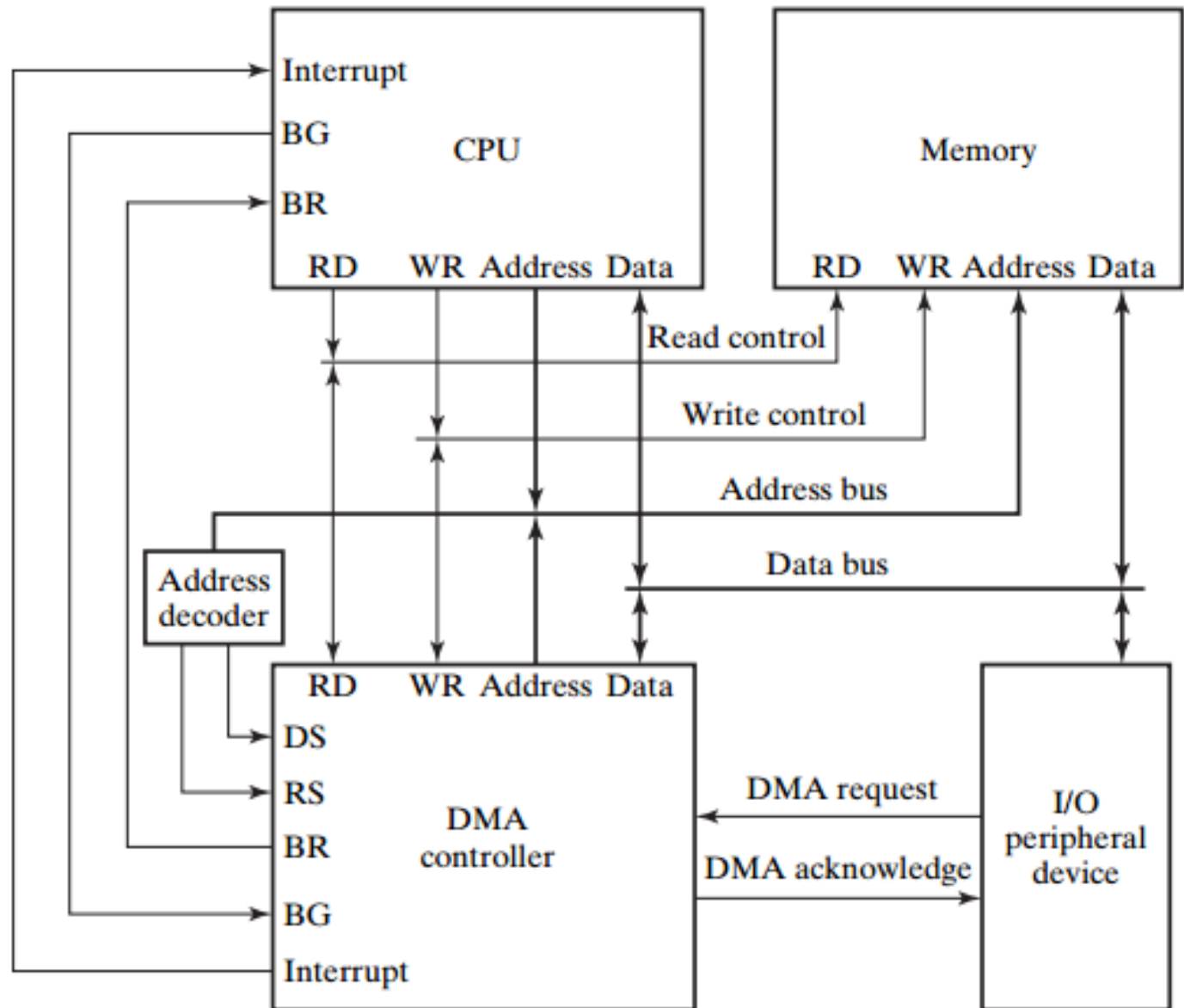
# DMA Handshaking



# DMA Controller



# DMA Transfer



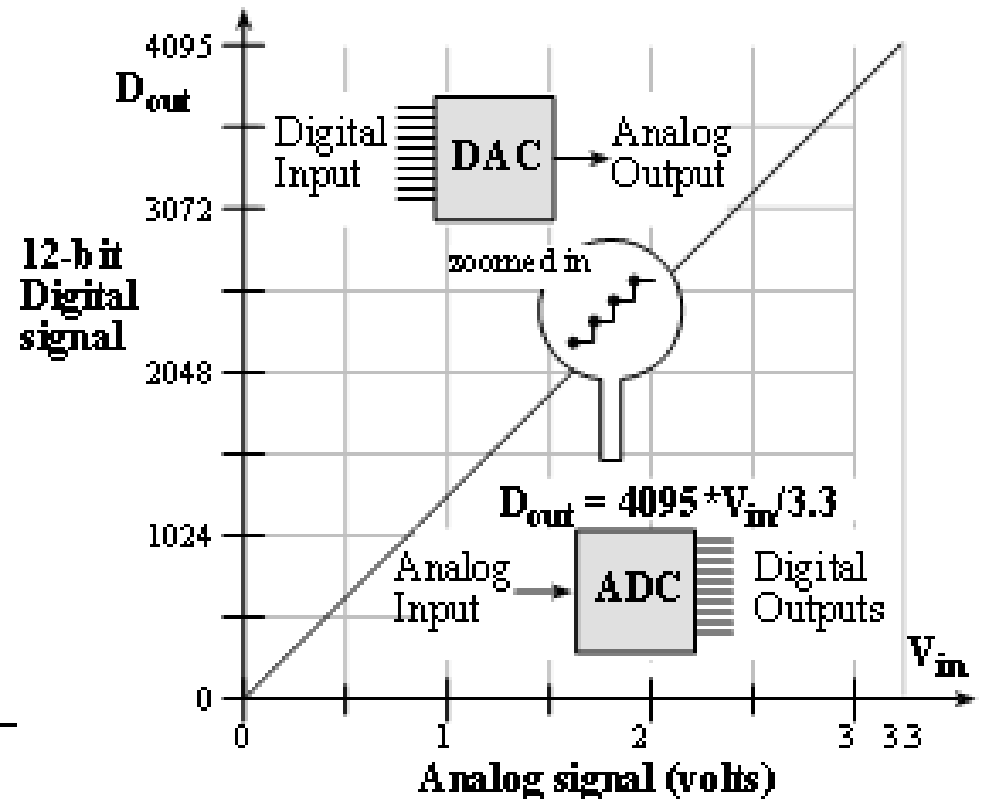
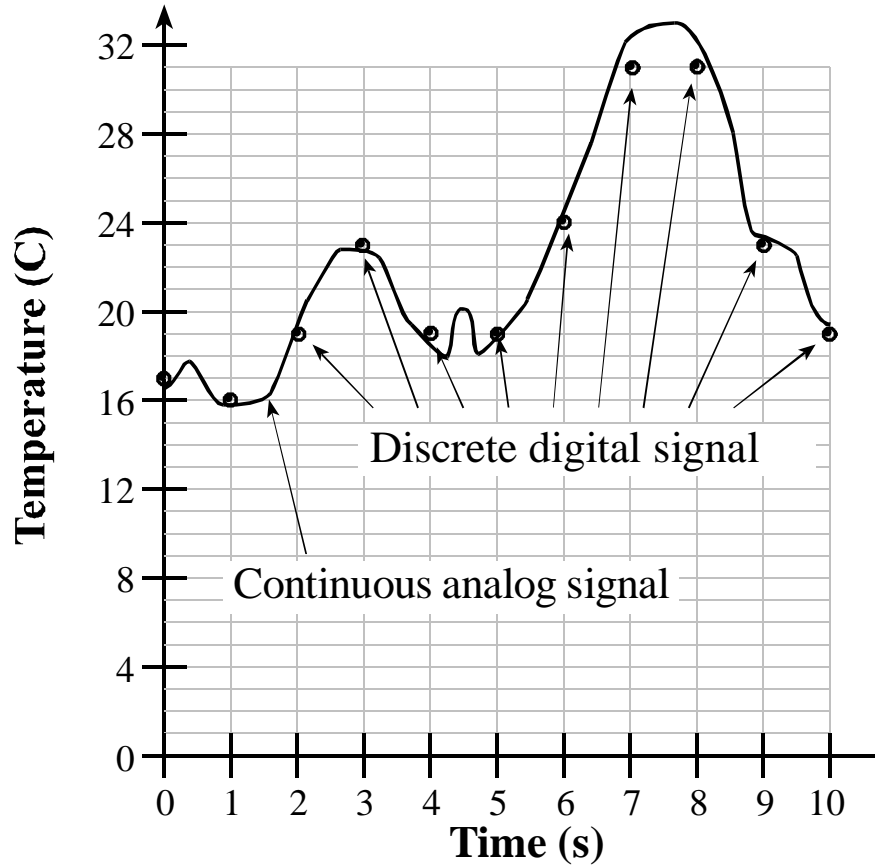
# A/D Converters

---

- ▶ Analog-to-digital conversion converts continuous (analog) signals from various transducers and devices into discrete (digital) ones.
- ▶ Similar circuitry can be used to convert temperature, sound, pressure, and other inputs from transducers using a variety of sampling schemes to perform the conversion.
- ▶ The output of A/D circuitry is a discrete version of the time-varying signal being monitored.
- ▶ The discrete version of the continuous value can be treated as a scaled number.
- ▶ The key factor in the service of A/D circuitry for time varying signals is the sampling rate (the Nyquist rate).



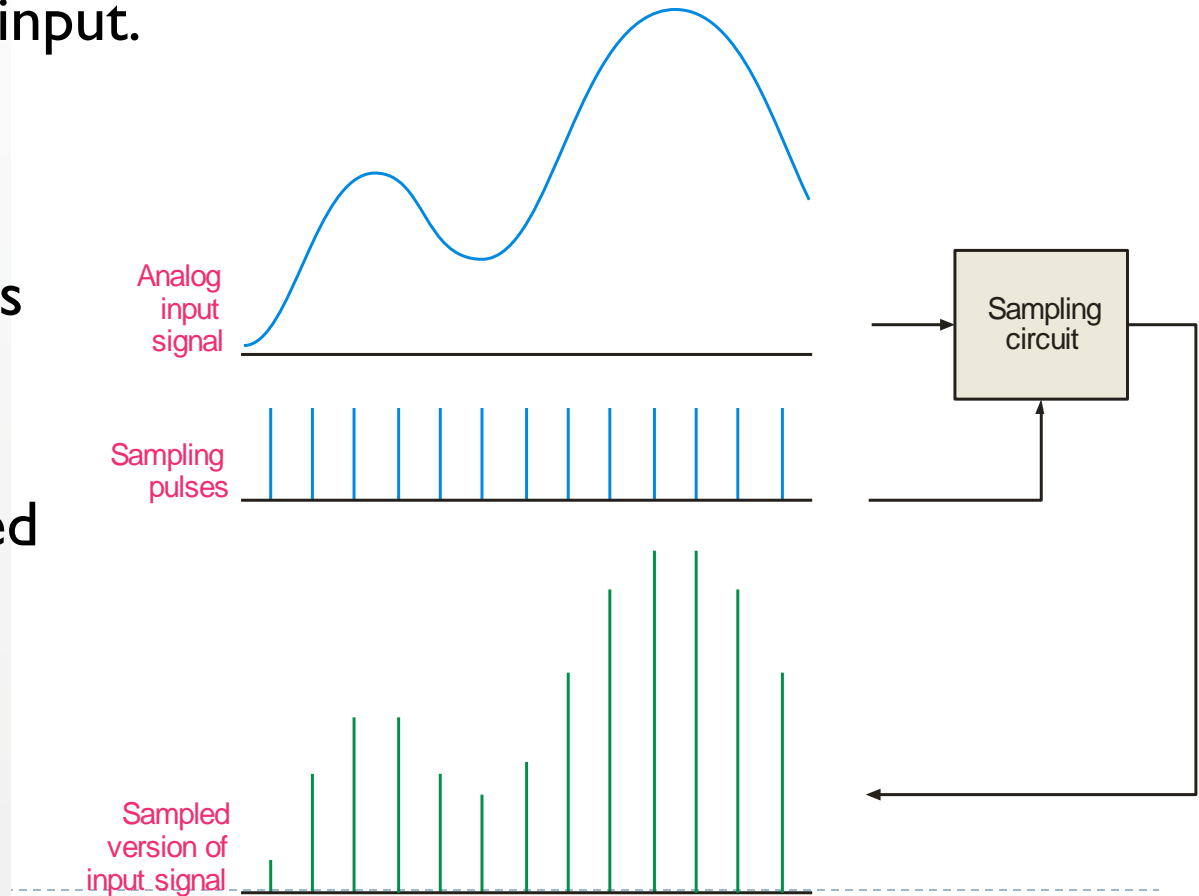
# Analog to Digital Converter (ADC)



# Signal Sampling

Most input signals to an electronic system start out as analog signals. For processing, the signal is normally converted to a digital signal by sampling the input.

Before sampling, the analog input must be filtered with a low-pass anti-aliasing filter. The filter eliminates frequencies that exceed a certain limit that is determined by the sampling rate.

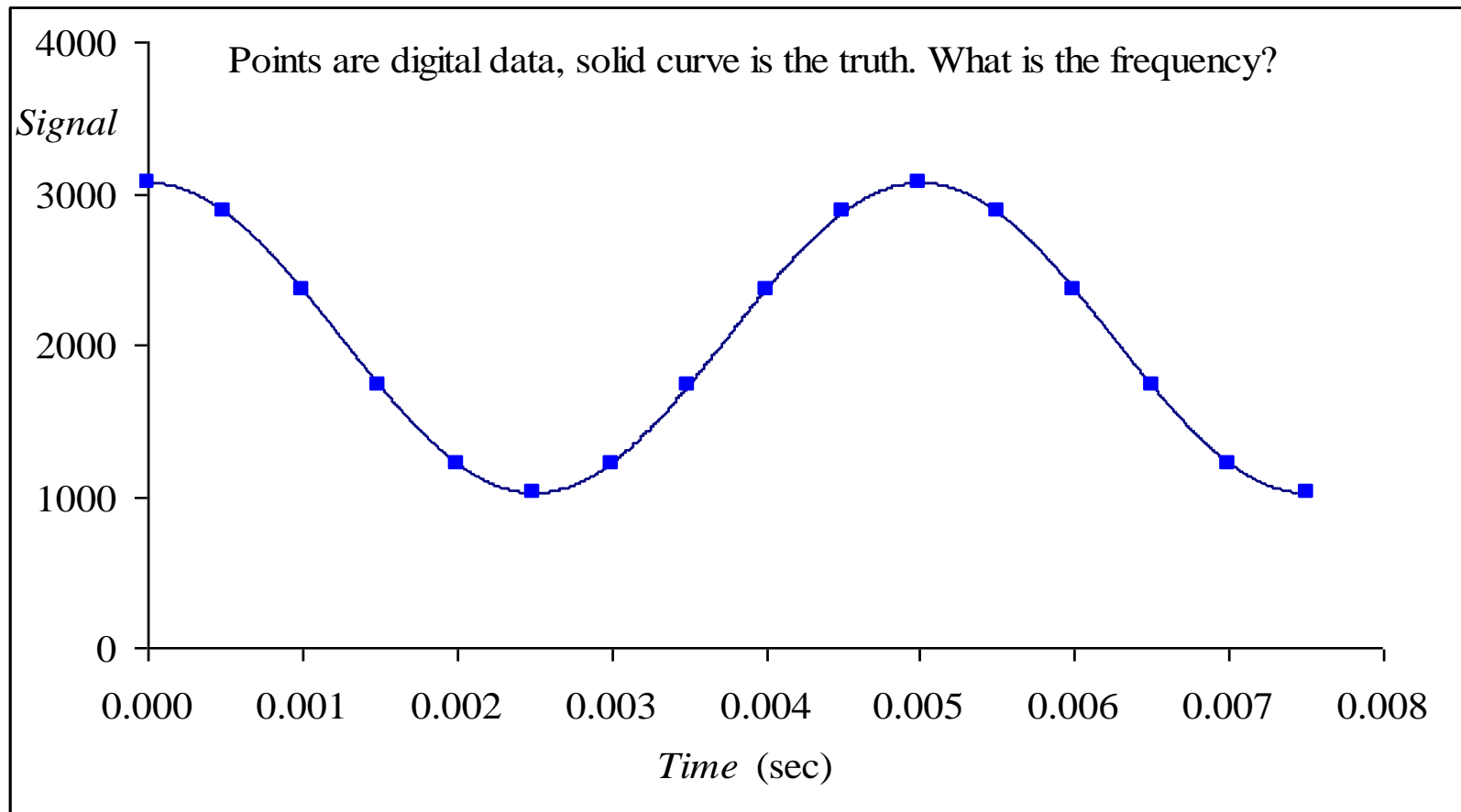


# Nyquist Theorem

- ▶ A *bandlimited* analog signal that has been sampled can be perfectly reconstructed from an infinite sequence of samples if the sampling rate  $f_s$  exceeds  $2f_{max}$  samples per second, where  $f_{max}$  is the highest frequency in the original signal.
- ▶ If the analog signal does contain frequency components larger than  $(1/2)f_s$ , then there will be an **aliasing** error.
- ▶ Aliasing is when the digital signal appears to have a different frequency than the original analog signal.
- ▶ **Valvano Postulate:** If  $f_{max}$  is the largest frequency component of the analog signal, then you must sample more than ten times  $f_{max}$  in order for the reconstructed digital samples to **look like** the original signal when plotted on a voltage versus time graph.

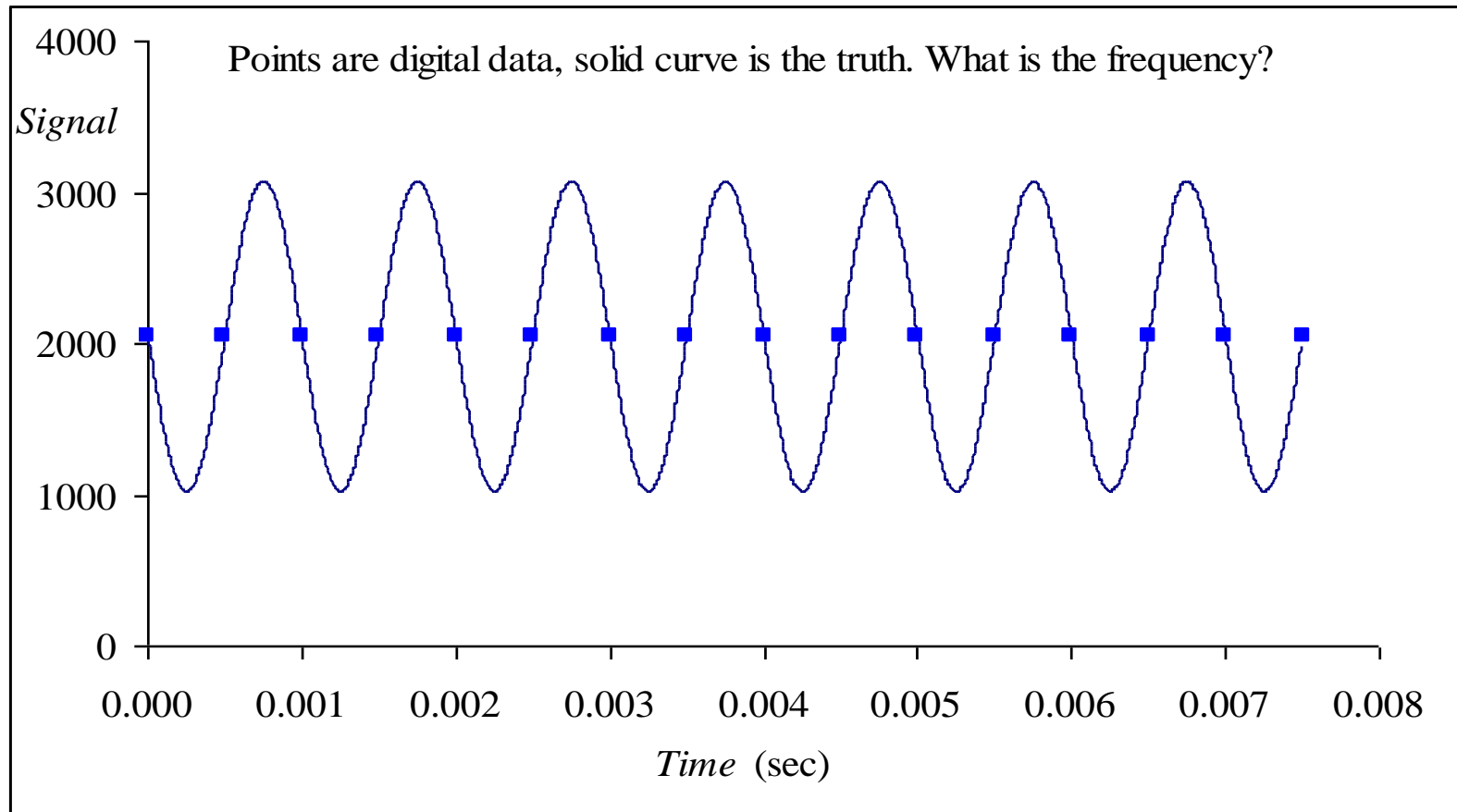
# Sampling (option 1)

- ▶ 200Hz signal sampled at 2000Hz



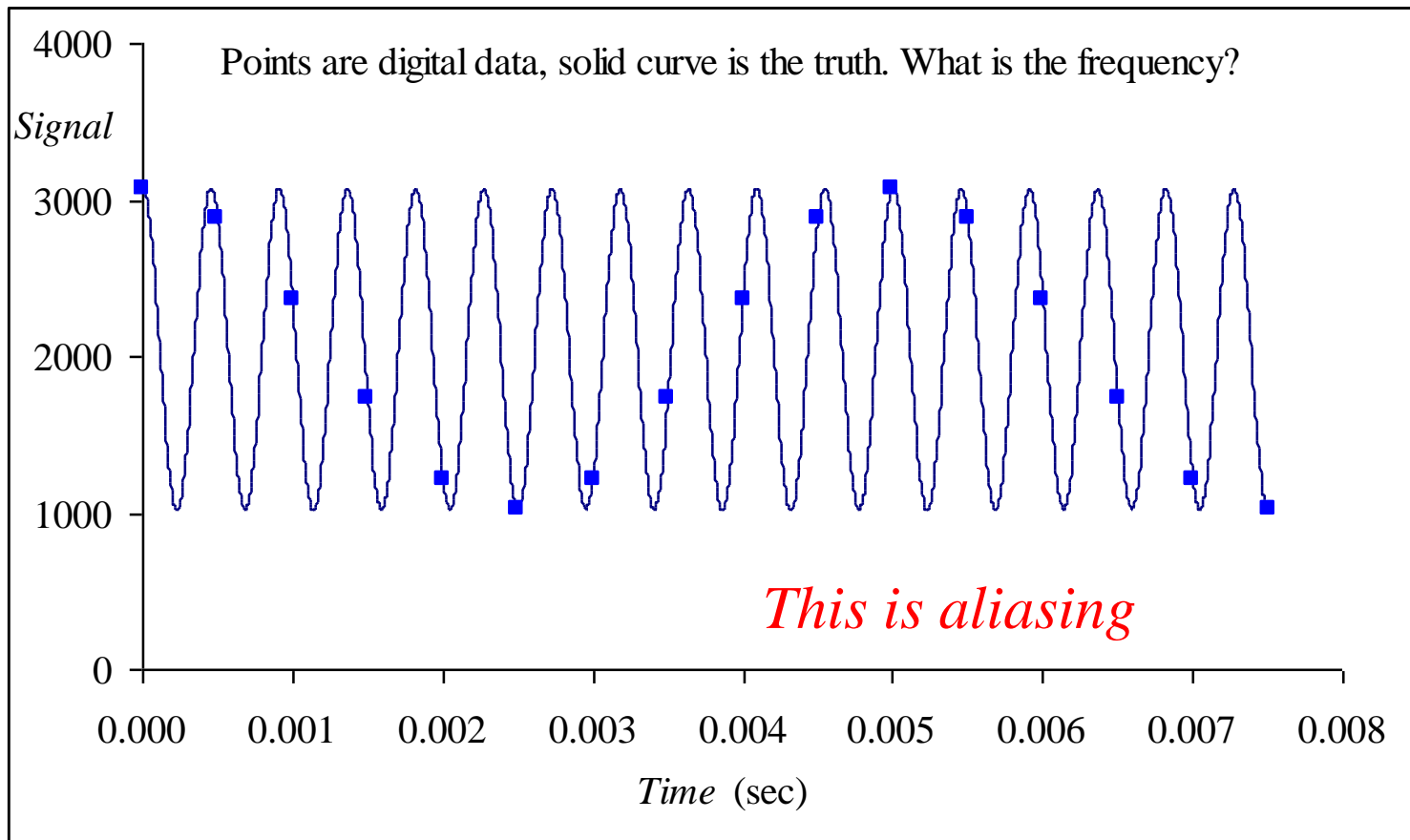
# Sampling (option 1)

- ▶ 1000Hz signal sampled at 2000Hz



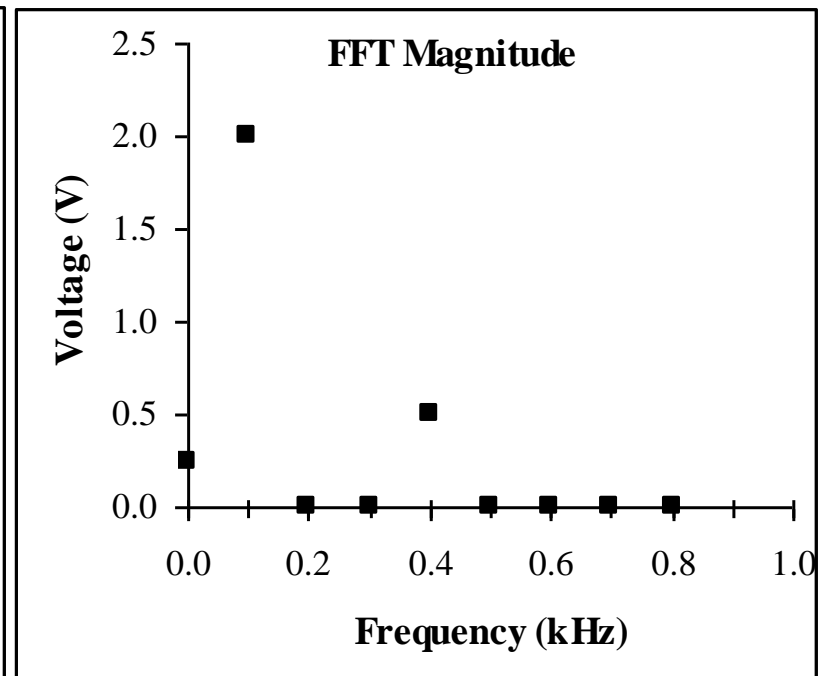
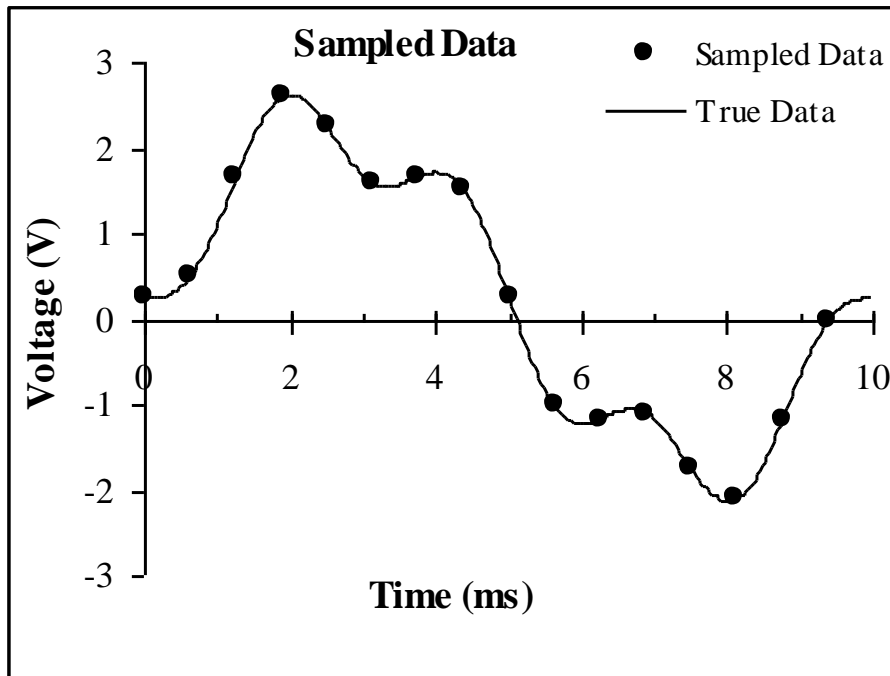
# Sampling (option 1)

- ▶ 2200Hz signal sampled at 2000Hz



# Sampling (option 2)

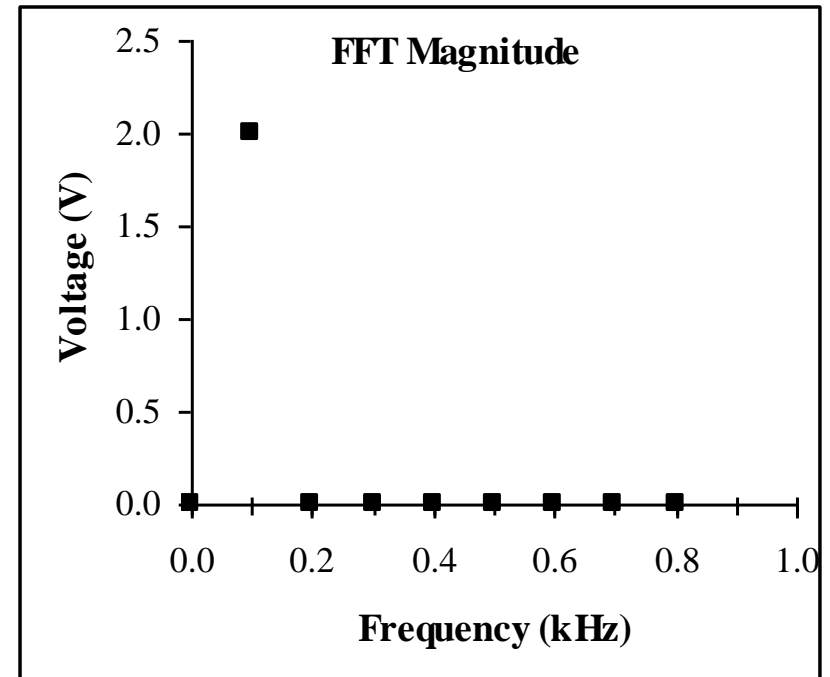
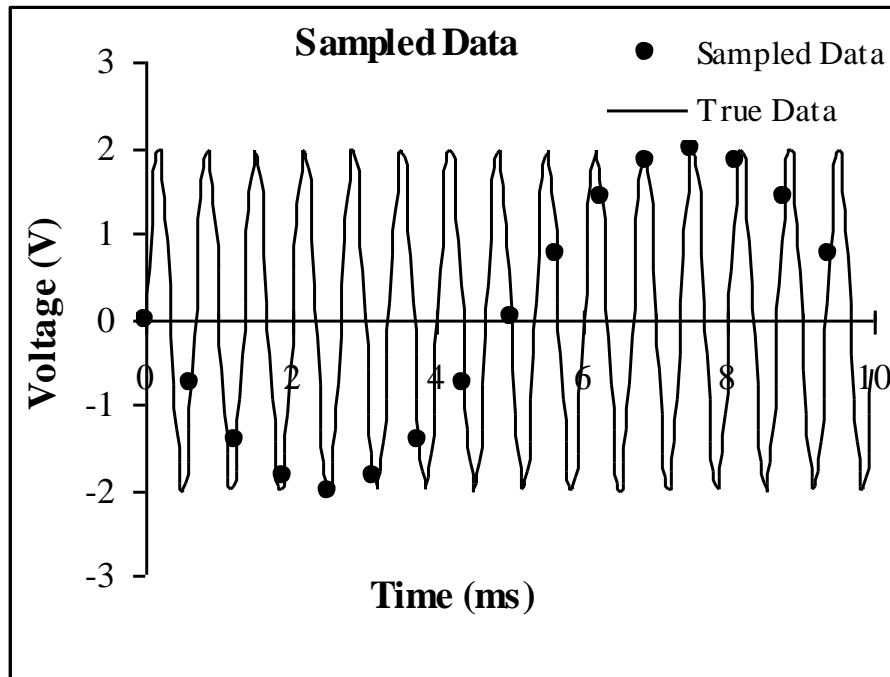
- ❑ A signal with DC, 100Hz and 400Hz sampled at 1600Hz



# Sampling (option 2)

□ 1500Hz signal sampled at 1600Hz

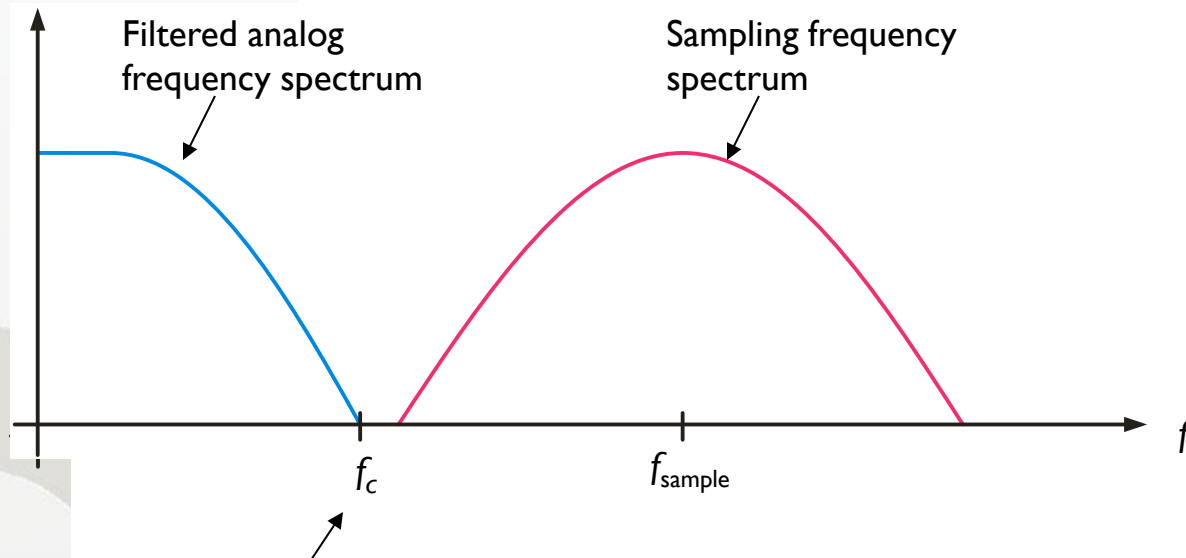
*This is aliasing*





# Anti-Aliasing Filter

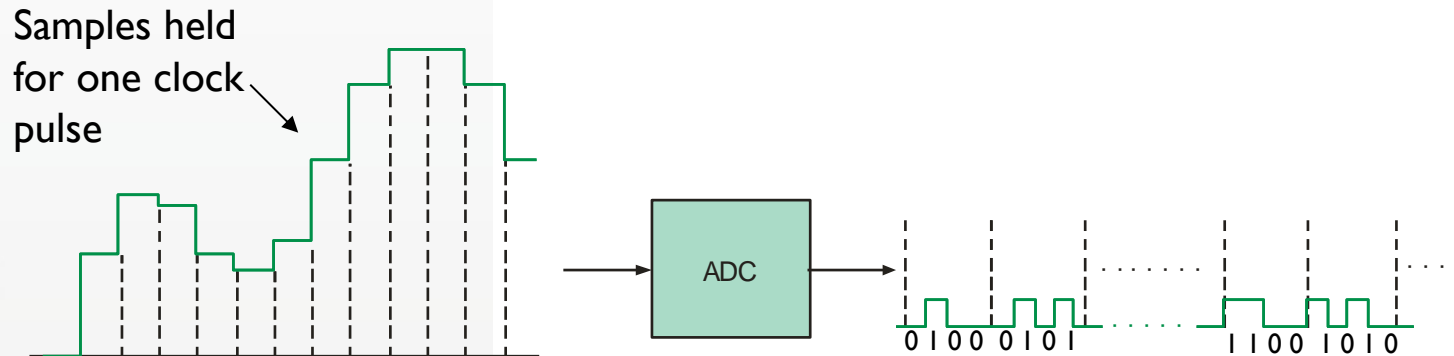
The anti-aliasing filter is a low-pass filter that limits high frequencies in the input signal to only those that meet the requirements of the sampling theorem.



The filter's cutoff frequency,  $f_c$ , should be less than  $\frac{1}{2} f_{\text{sample}}$ .

# A/D Conversion

Following the anti-aliasing filter, is the sample-and-hold circuit and the analog-to-digital converter. At this point, the original analog signal has been converted to a digital signal.



Many ICs can perform both functions on a single chip and include two or more channels.

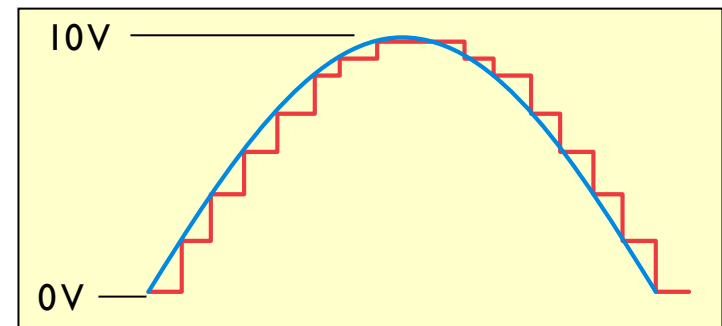
# A/D Conversion

---

To process naturally occurring analog quantities with a digital system, the analog signal is converted to digital form after the anti-aliasing filter.

The first step in converting a signal to digital form is to use a sample-and-hold circuit. This circuit samples the input signal at a rate determined by a clock signal and holds the level on a capacitor until the next clock pulse.

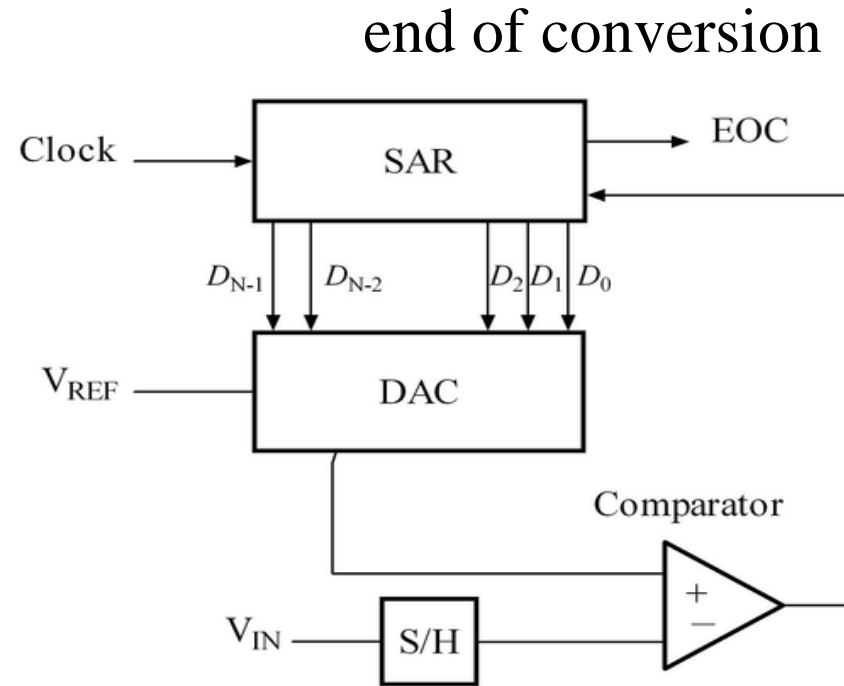
A positive half-wave from 0-10V is shown in blue. The sample-and-hold circuit produces the staircase representation shown in red.



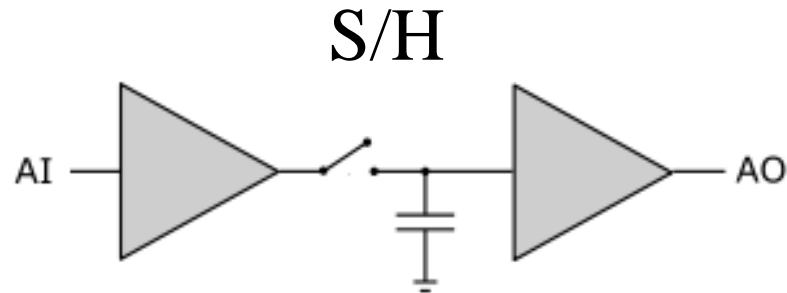
# Analog-to-Digital Converter (ADC)

## ▶ Successive approximation ADC

- ▶  $V_{IN}$  is approximated as a static value in a *sample and hold (S/H)* circuit
- ▶ the *successive approximation register (SAR)* is a counter that increments each *clock* as long as it is enabled by the *comparator*
- ▶ the output of the SAR is fed to a DAC that generates a voltage for comparison with  $V_{IN}$
- ▶ when the output of the DAC =  $V_{IN}$  the value of SAR is the digital representation of  $V_{IN}$



# Sample-And-Hold Circuit



- ❑ **Analog Input (AI)** is sampled when the switch is closed and its value is *held* on the capacitor where it becomes the **Analog Output (AO)**

# A/D Conversion

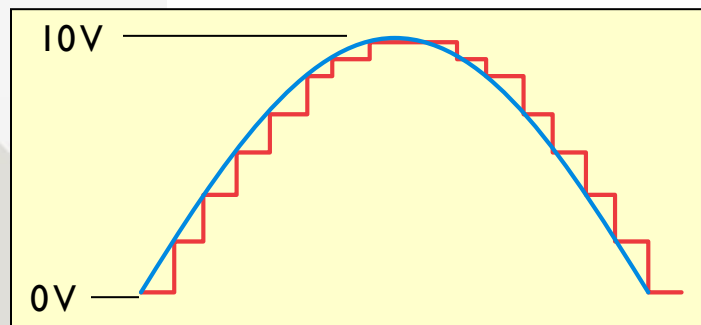
The second step is to **quantize** these staircase levels to binary coded form using an analog-to-digital converter (ADC). The digital values can then be processed by a digital signal processor or computer.

## Example

What is the maximum unsigned binary value for the waveform?

## Solution

$10\text{V} = 1010_2\text{V}$ . The table lists the quantized binary values for all of the steps.



Peak = 10V

0.0000
10.0001
100.0001
101.1110
111.0111
1000.1011
1001.1001
1010.0000
1010.0000
1001.1001
1000.1011
111.0111
101.1110
100.0001
10.0001
0.0000

# ADC on TM4C123

---

- ▶ Sampling Range/Resolution
    - ▶ 3.3V internal reference voltage
    - ▶ 0x000 at 0 V input
    - ▶ 0xFFF at 3.3 V
    - ▶ resolution = range/precision  
= 3.3V/**4096** alternatives < 1mV
-

# D/A Converters

---

- ▶ Digital-to-analog conversion performs the inverse function of A/D circuitry.
- ▶ Converts a discrete quantity to a continuous one.
- ▶ D/A devices are used to allow the computer to output analog voltages based on the digital version stored internally.
- ▶ Communication with D/A circuitry uses one of the three input/output methods discussed.

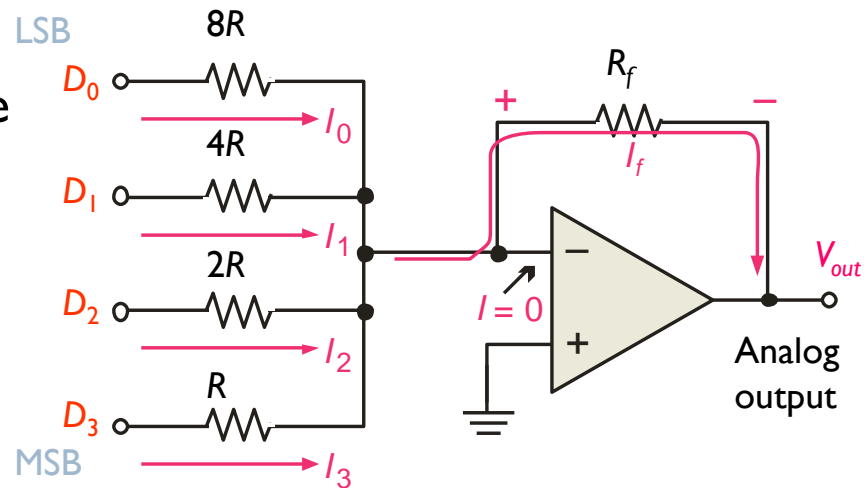


# D/A Converters

## Binary-weighted-input DAC:

The binary-weighted-input DAC is a basic DAC in which the input current in each resistor is proportional to the column weight in the binary numbering system. It requires very accurate resistors and identical HIGH level voltages for accuracy.

The MSB is represented by the largest current, so it has the smallest resistor. To simplify analysis, assume all current goes through  $R_f$  and none into the op-amp.

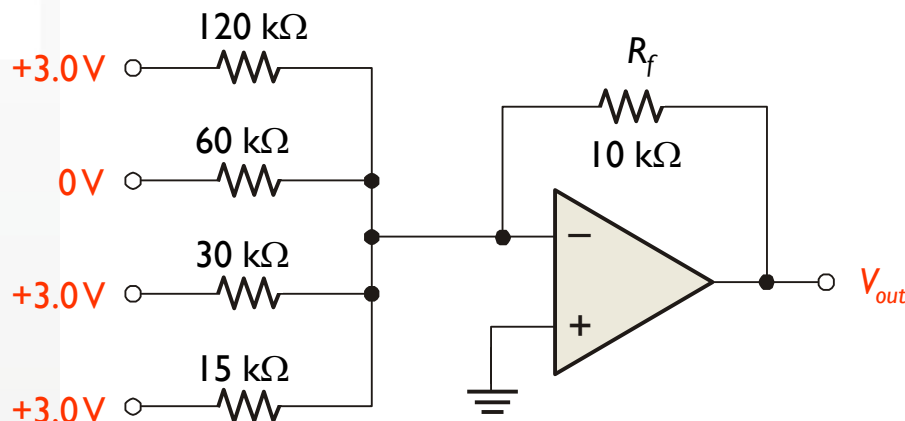


# D/A Converters

## Binary-weighted-input DAC:

**Example**

A certain binary-weighted-input DAC has a binary input of 1101. If a HIGH = +3.0 V and a LOW = 0 V, what is  $V_{out}$ ?



**Solution**

$$I_{out} = -(I_0 + I_1 + I_2 + I_3)$$

$$= -\left(\frac{3.0 \text{ V}}{120 \text{ k}\Omega} + 0 \text{ V} + \frac{3.0 \text{ V}}{30 \text{ k}\Omega} + \frac{3.0 \text{ V}}{15 \text{ k}\Omega}\right) = -0.325 \text{ mA}$$

$$V_{out} = I_{out} R_f = (-0.325 \text{ mA})(10 \text{ k}\Omega) = -3.25 \text{ V}$$

# D/A Converters

## R-2R ladder:

The  $R$ - $2R$  ladder requires only two values of resistors. By calculating a Thevenin equivalent circuit for each input, you can show that the output is proportional to the binary weight of inputs that are HIGH.

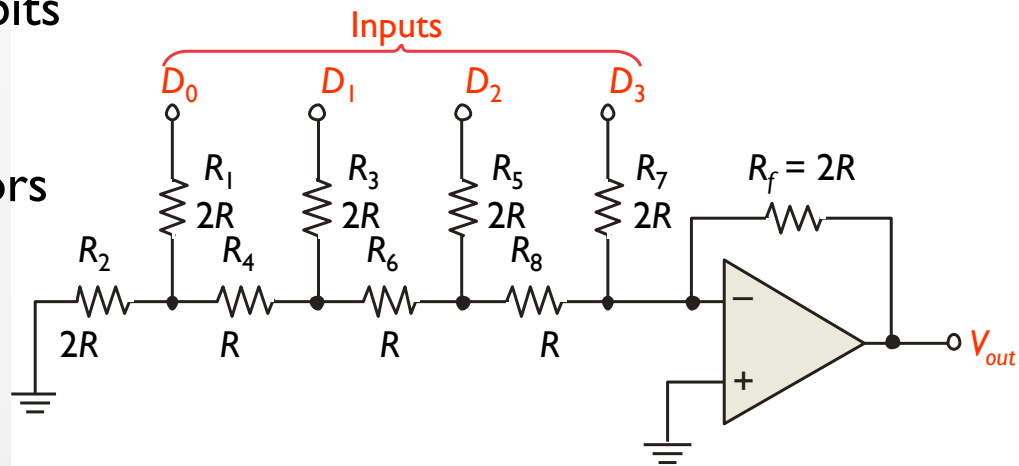
Each input that is HIGH contributes to the output: 
$$V_{out} = -\frac{V_S}{2^{n-i}}$$

where  $V_S$  = input HIGH level voltage

$n$  = number of bits

$i$  = bit number

For accuracy, the resistors must be precise ratios, which is easily done in integrated circuits.

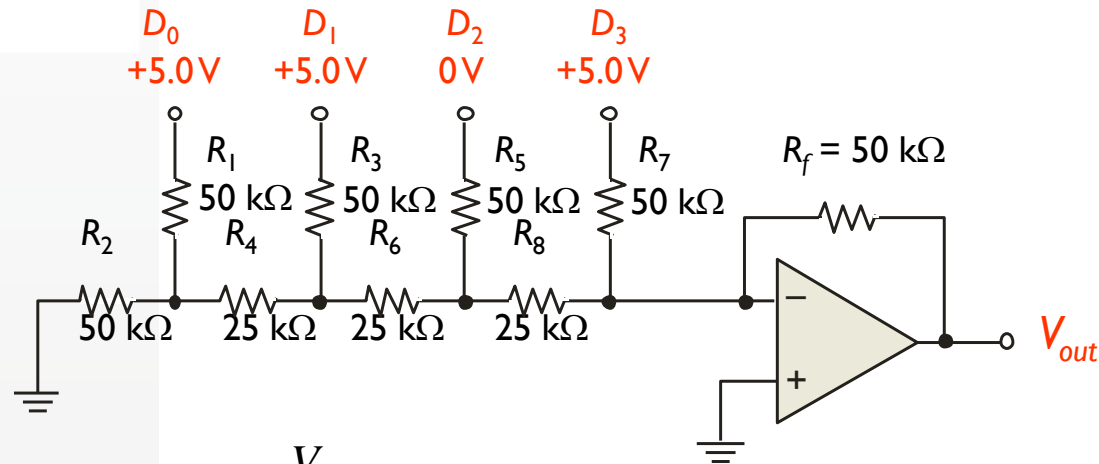


# D/A Converters

## R-2R ladder:

### Example

An R-2R ladder has a binary input of 1011. If a HIGH = +5.0 V and a LOW = 0 V, what is  $V_{out}$ ?



### Solution

Apply  $V_{out} = -\frac{V_S}{2^{n-i}}$  to all inputs that are HIGH, then sum the results.

$$V_{out}(D_0) = -\frac{5\text{ V}}{2^{4-0}} = -0.3125\text{ V} \quad V_{out}(D_1) = -\frac{5\text{ V}}{2^{4-1}} = -0.625\text{ V}$$

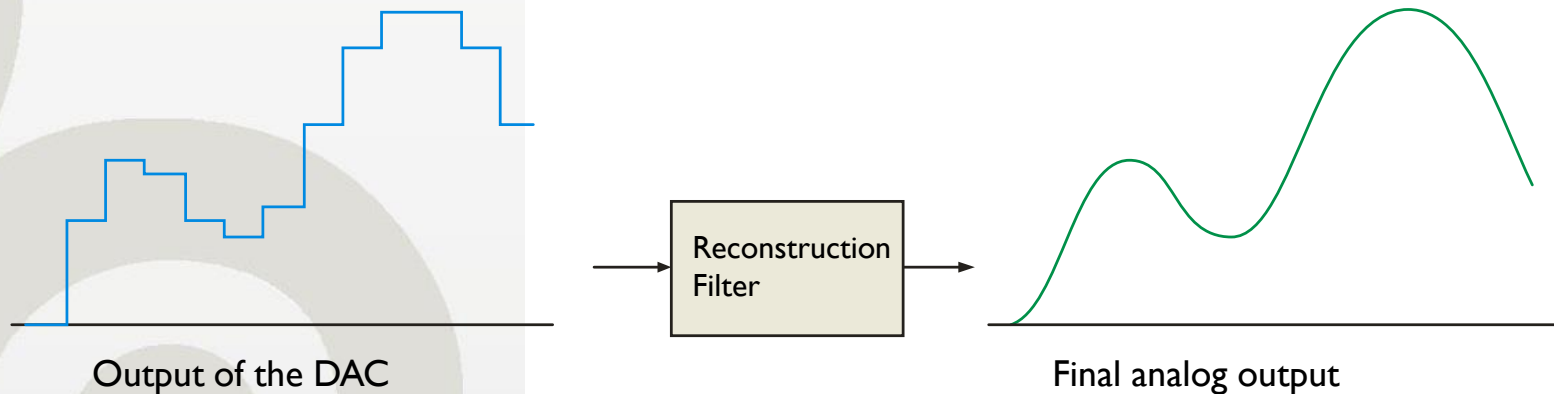
$$V_{out}(D_3) = -\frac{5\text{ V}}{2^{4-3}} = -2.5\text{ V} \quad \text{Applying superposition, } V_{out} = -3.43\text{ V}$$

# D/A Converters

---

## Reconstruction Filter:

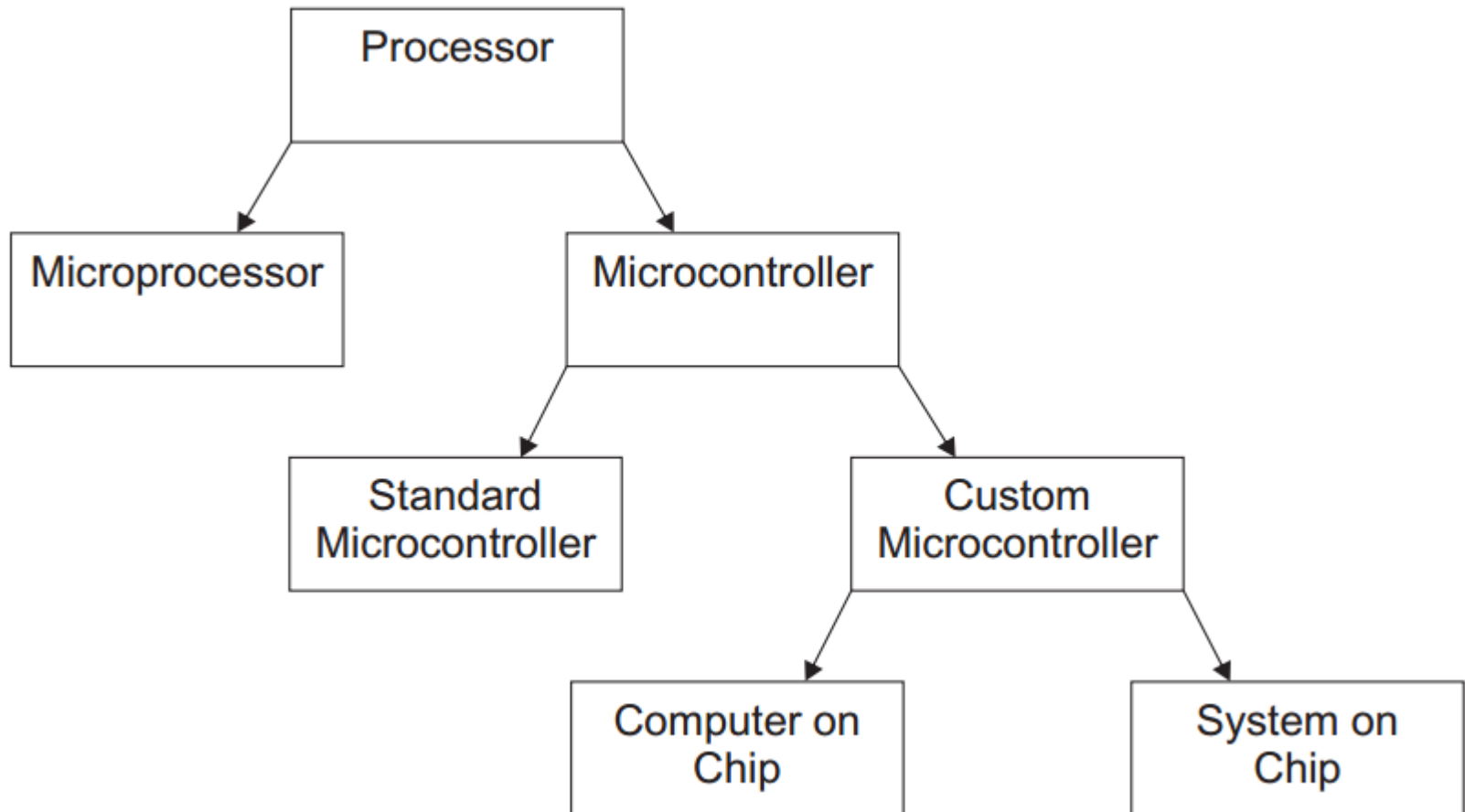
After converting a digital signal to analog, it is passed through a low-pass “reconstruction filter” to smooth the stair steps in the output. The cutoff frequency of the reconstruction filter is often set to the same limit as the anti-aliasing filter, to block higher harmonics due to the digitizing process.



# Processor technology

---

Principal evolution paths of processor technologies



# Standard Microcontroller

---

Modern microcontroller could contain the following:

- EEPROM or Flash
- SRAM
- ADC with a multiplexer
- DMA controller
- Parallel inputs and outputs
- Serial interface
- Timers and counters
- PWM
- Watchdog timers

# Custom Microcontrollers

---

- ▶ Applications specific integrated circuit – a special purpose integrated circuit designed for one application only.
- ▶ In essence, these devices are systems on a chip that can include a microprocessor, memory, I/O devices and other specialized circuitry.
- ▶ ASICs are used in many embedded applications including image processing, avionics systems, medical systems.
- ▶ Real-time design issues are the same for them as they are for most other systems.



# FPGAs

---

- ▶ Field programmable gate array (FPGA) – allows construction of a system on a chip with an integrated processor, memory, and I/O.
- ▶ It is reprogrammable, even while embedded in the system.
- ▶ A reconfigurable architecture allows for the programmed interconnection and functionality of a conventional processor
- ▶ Algorithms and functionality are moved from residing in the software side into the hardware side.
- ▶ Widely used in embedded, mission-critical systems where fault-tolerance and adaptive functionality is essential.

# Any Questions?

---

