Digital Systems L2 – Computer Systems

Fang Yuan

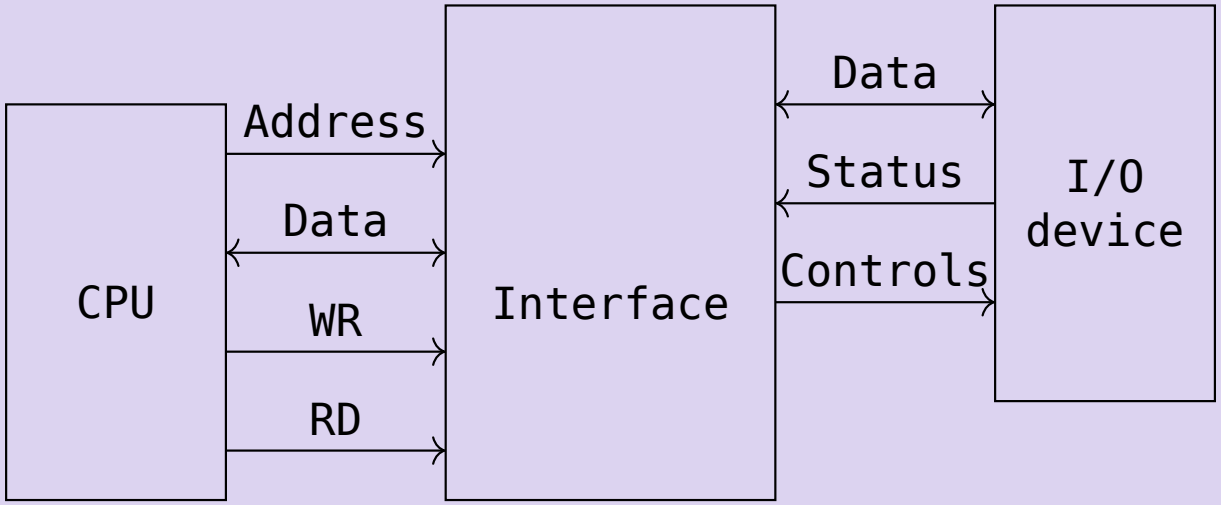
School of Electronics Science and Engineering
Nanjing University
Nanjing 210046

2022

Outline

Interfaces

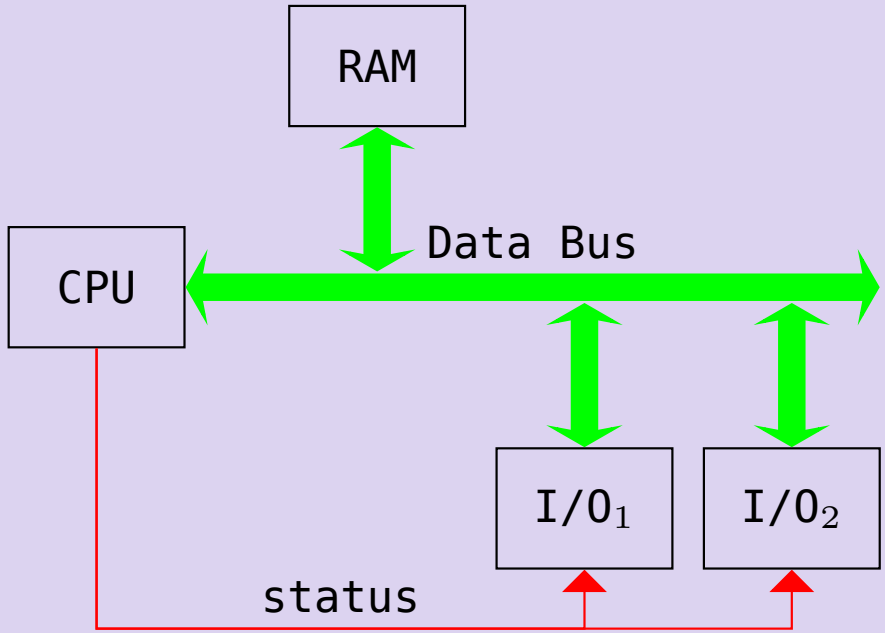
Interface between CPU and Devices



1. Data, byte(parallel) or bit (serial)
2. Addresses, identifying devices
3. Control signals: read/write, clock, etc. CPU $\rightarrow$ Device.
4. Status, informing CPU if data is available.

Polling

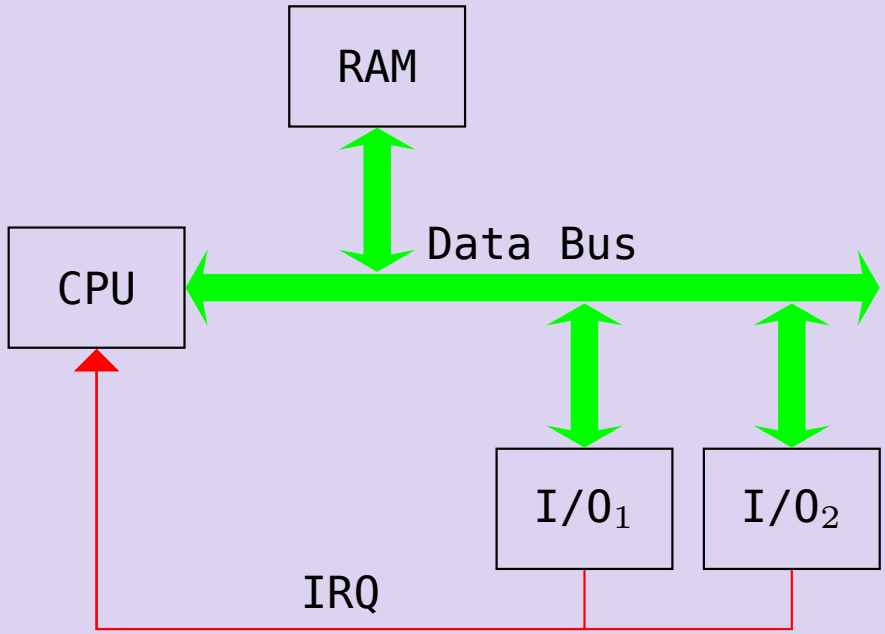
Polling



CPU reads status during waiting loop, until the device is ready for reading or writing.

Interrupt Mode

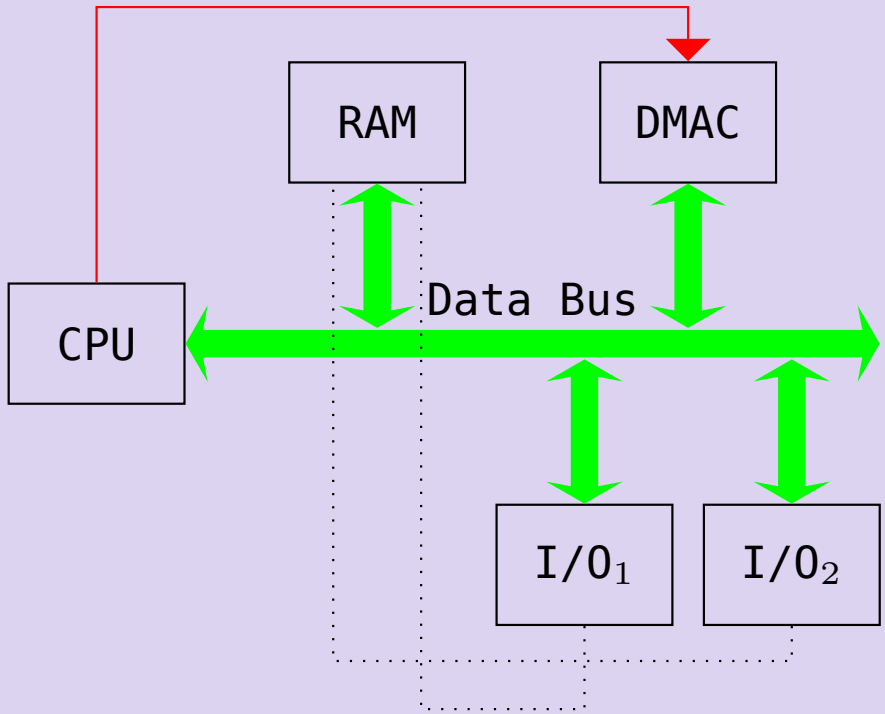
Interrupt



Device informs CPU when it is ready. CPU doesn't need to wait the status.

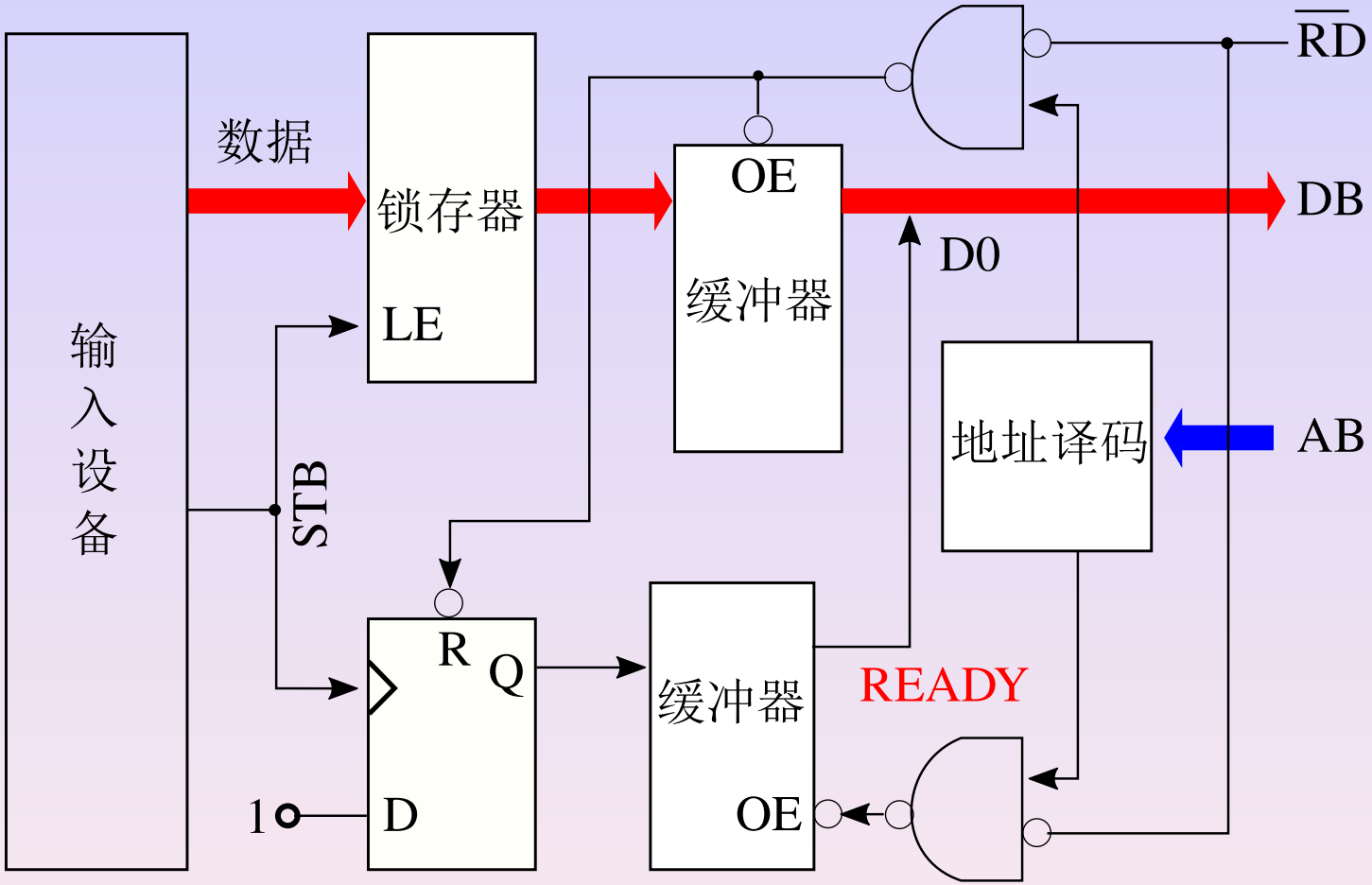
Transferring by DMA

DMA



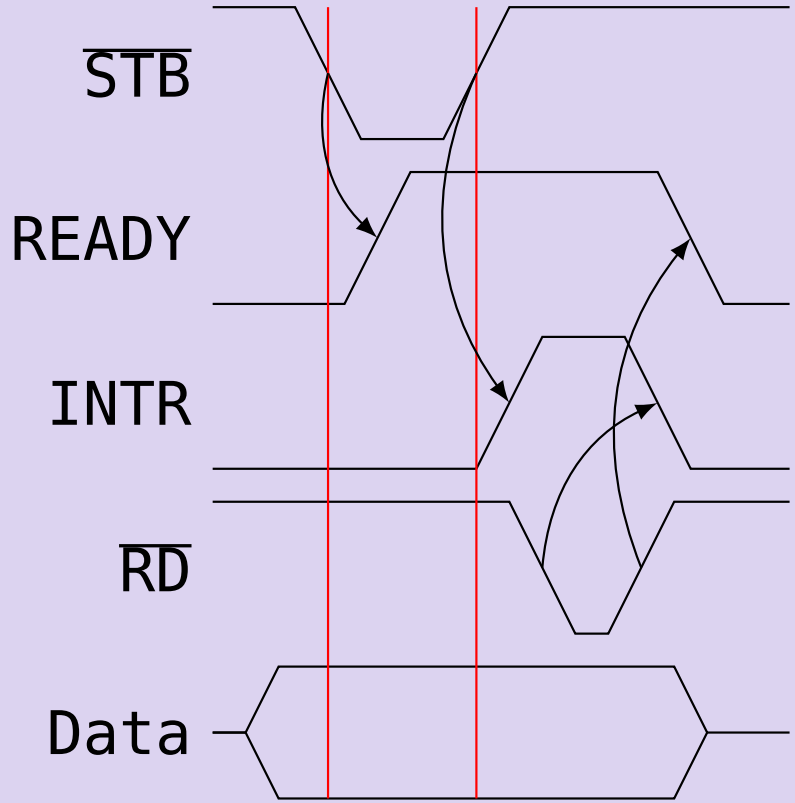
During transferring, bus is controlled by DMAC.

Polling Input

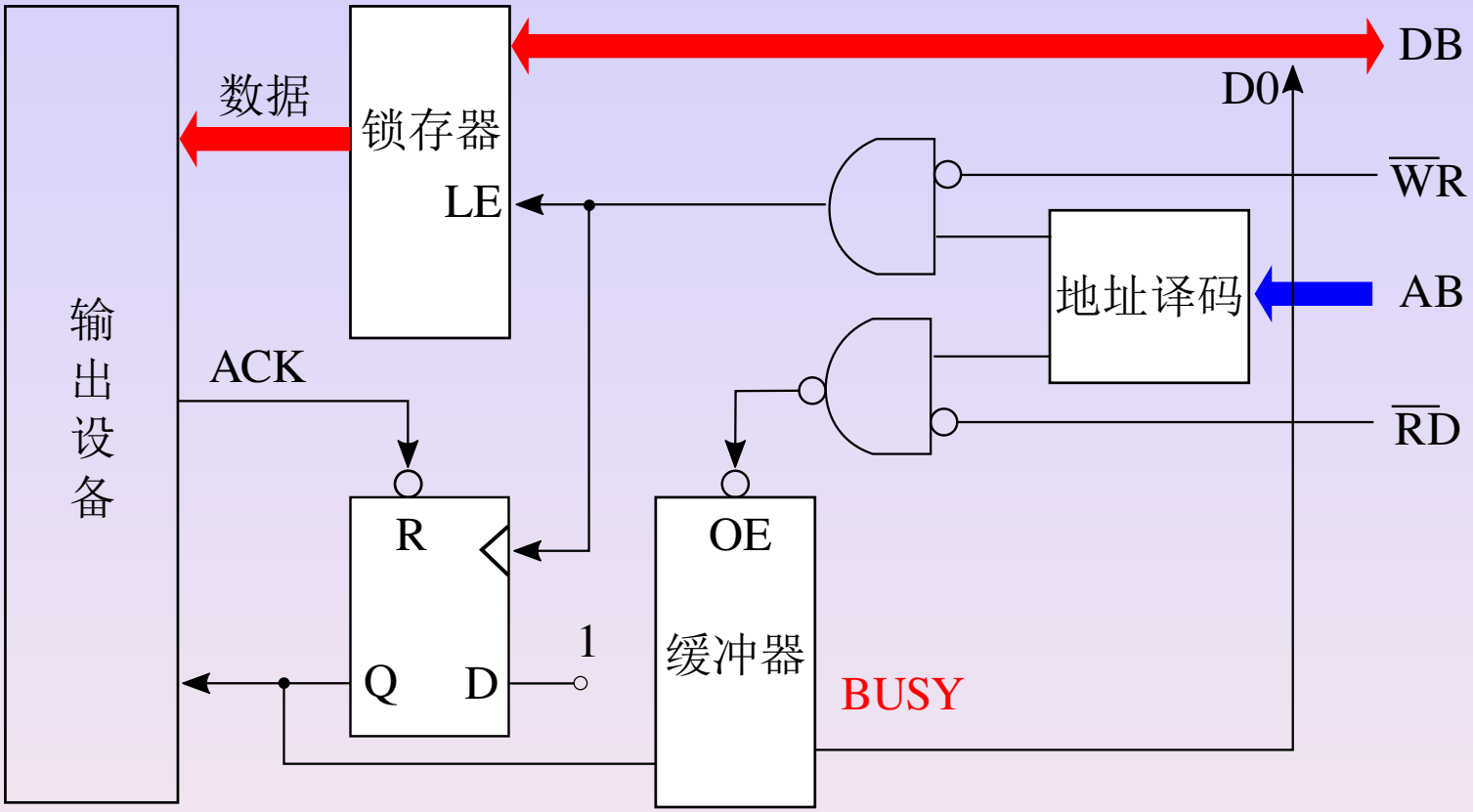


Polling Input

Strobe Input

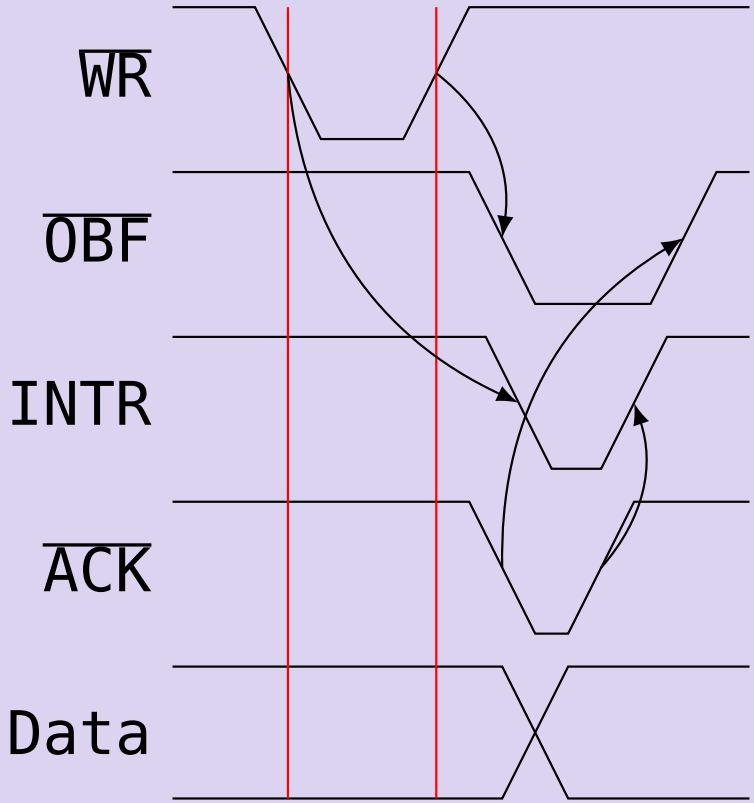


Polling Output



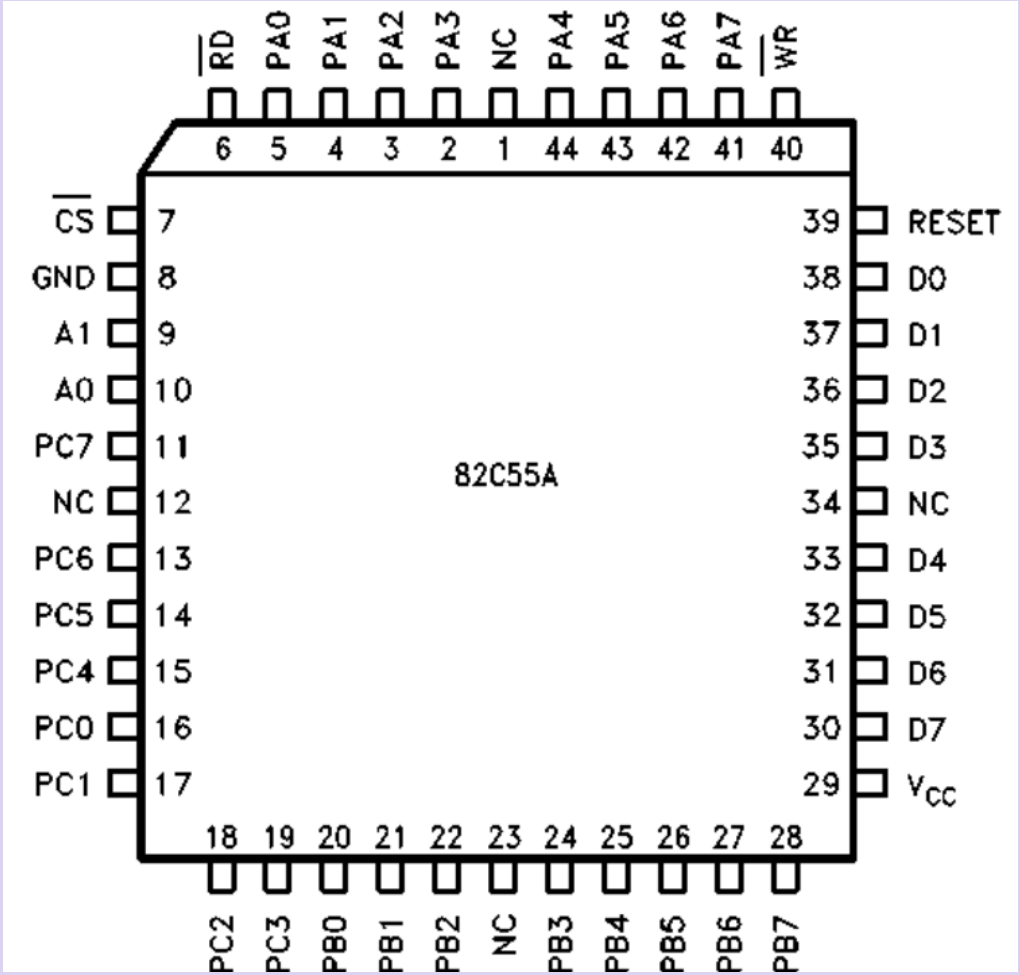
Polling Output

Strobe Output

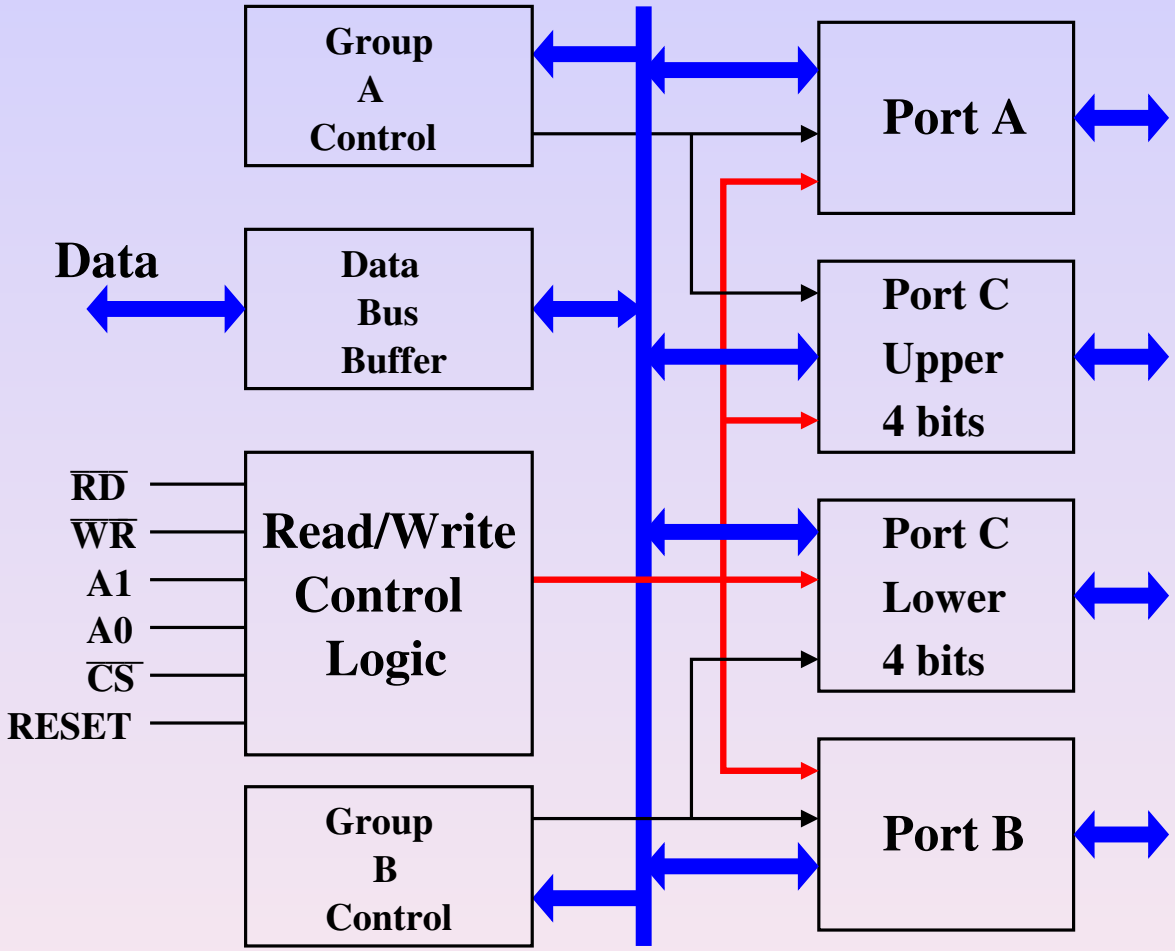


Programmable Parallel Interface

P8255



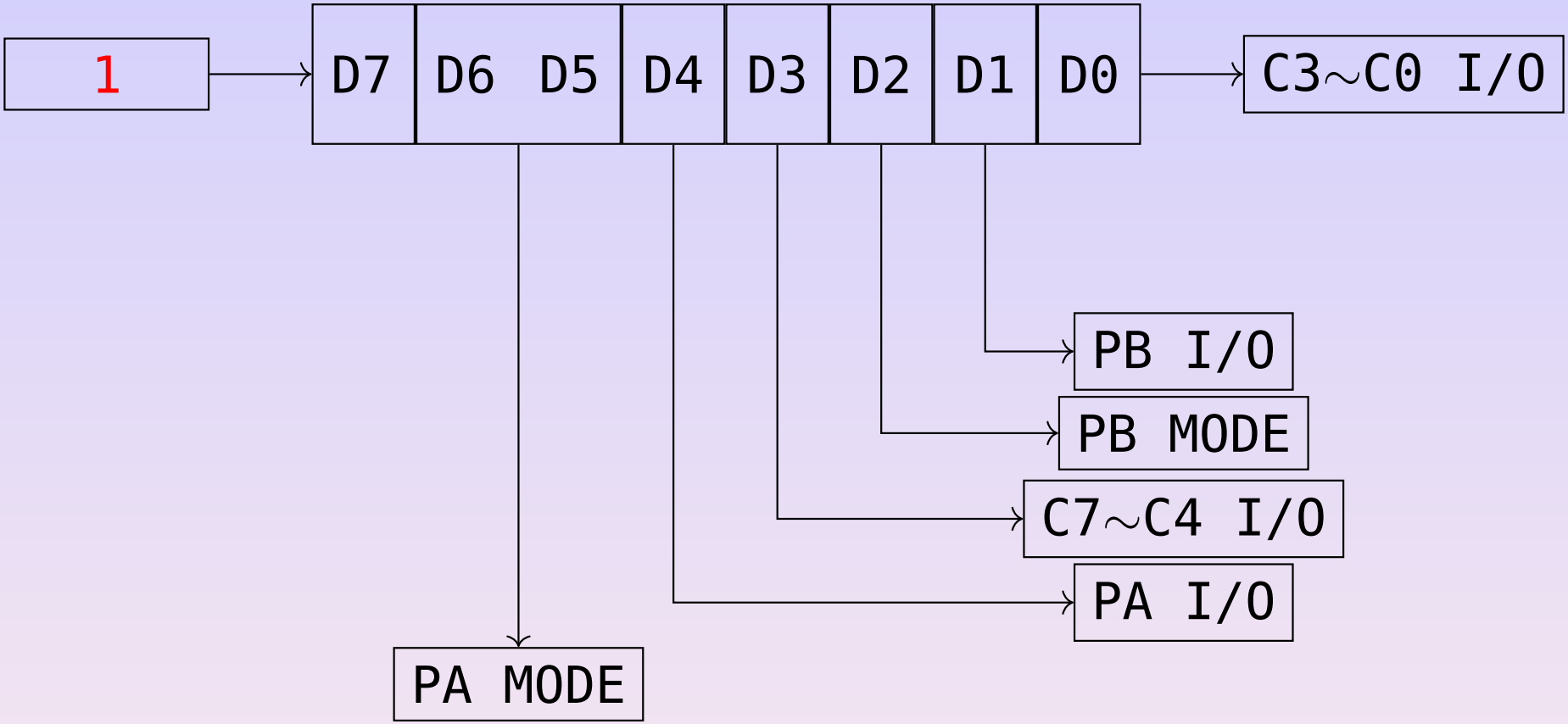
P8255


Group
A
Control
Port A
Data
Data
Bus
Buffer
Port C
Upper
4 bits
RD
WR
A1
A0
CS
RESET
Read/Write
Control
Logic
Port C
Lower
4 bits
Group
B
Control
Port B

P8255 Programming

A1	A0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Operation
0	0	0	1	0	Data bus $\Leftarrow$Port A
0	1	0	1	0	Data bus $\Leftarrow$Port B
1	0	0	1	0	Data bus $\Leftarrow$Port C
0	0	1	0	0	Data bus $\Rightarrow$Port A
0	1	1	0	0	Data bus $\Rightarrow$Port B
1	0	1	0	0	Data bus $\Rightarrow$Port C
1	1	1	0	0	Data bus $\Rightarrow$Control
–	–	–	–	1	Data bus $\Rightarrow$tri-state
1	1	0	1	0	Data bus $\Rightarrow$tri-state
–	–	1	1	0	Disable

P8255 Programming



Port A, B and C with only one command word.
Mode 0, 1, 2 for Port A, mode 0, 1 for port B.

Interrupts

Concept

An interrupt is an input signal to CPU indicating an event to be processed. CPU may break its work to process this request, and then back to the breakpoint.

Interrupts are commonly used by hardware to indicate electronic or physical state changes that require attention.

Types of Interrupts

Interrupts may be internal (CPU) or external (devices).

- internal:
  - Exception/Trap
  - Software (interrupt instructions, used as global function call)
- external, from devices. (Also called hardware interrupts)
  - Maskable interrupts – CPU may ignore them.
  - Non-maskable interrupts

Processes of an Interrupt

A typical interrupt process runs as follow:

1. Interrupt request.
2. Preparations.
3. Interrupt Service Routine (programmer's job).
4. Interrupt return.

IRQ

- For maskable interrupt, check permissions, mask bits, privilege, etc., mark Interrupt Flag, mask other maskable interrupts, then send INTA(INTerrupt Acknowledge).
- For Non-maskable interrupt, mask other interrupts, send INTA.

Processes of an Interrupt

A typical interrupt process runs as follow:

1. Interrupt request.
2. Preparations.
3. Interrupt Service Routine (programmer's job).
4. Interrupt return.

What CPU automatically does

- Disable other interrupts.
- Save breakpoint (onto stack) for later use (return).
- Jump to ISR.

Processes of an Interrupt

A typical interrupt process runs as follow:

1. Interrupt request.
2. Preparations.
3. Interrupt Service Routine (programmer's job).
4. Interrupt return.

IRQ

- saves processor's state (for safety return).
- enables other interrupts (optional, for nested interrupts).
- real task.
- retrieves state.

Processes of an Interrupt

A typical interrupt process runs as follow:

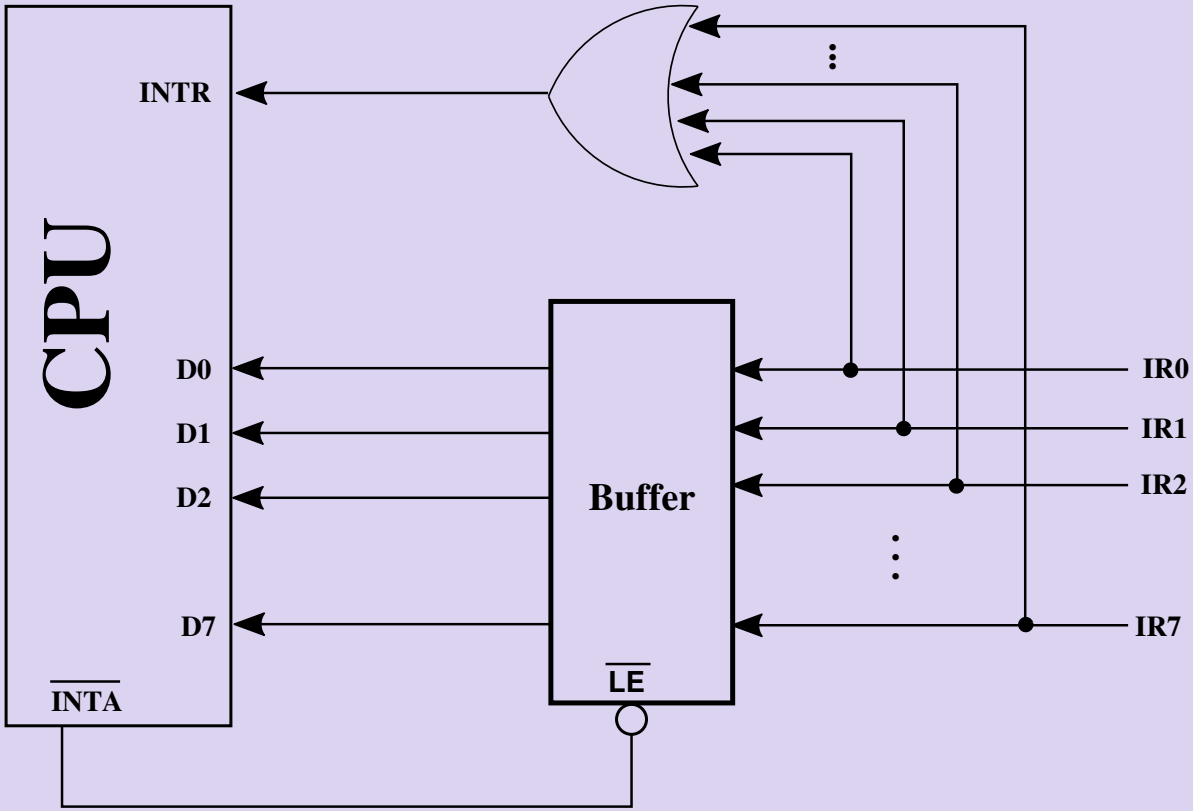
1. Interrupt request.
2. Preparations.
3. Interrupt Service Routine (programmer's job).
4. Interrupt return.

Different from normal return

Return from interrupts or from subroutines uses different instruction.

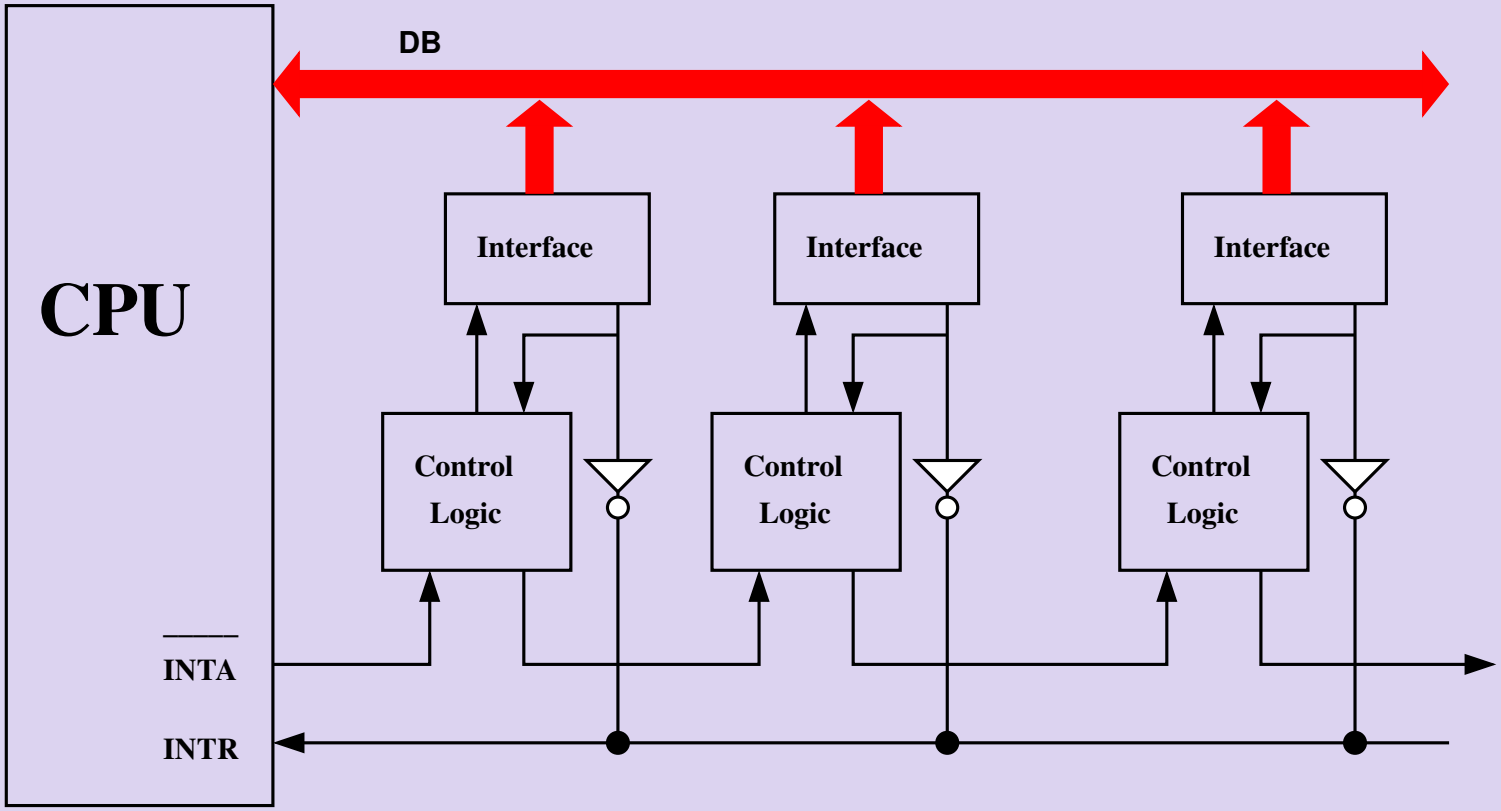
Polling-Interrupt and Hardware Interrupt

Software Polling-Interrupt



Polling-Interrupt and Hardware Interrupt

Hardware interrupt

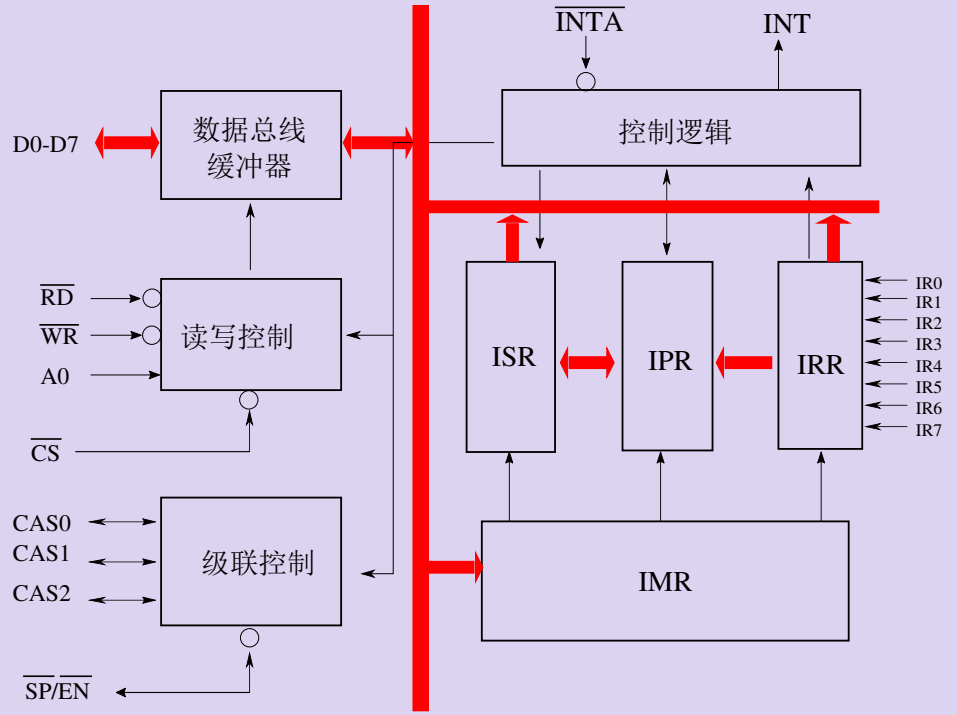


A typical PIC in PC

Features:

- 8 priority levels(64 levels when cascaded)
- Programmable priority and masking
- Programmable interrupt vector

8259



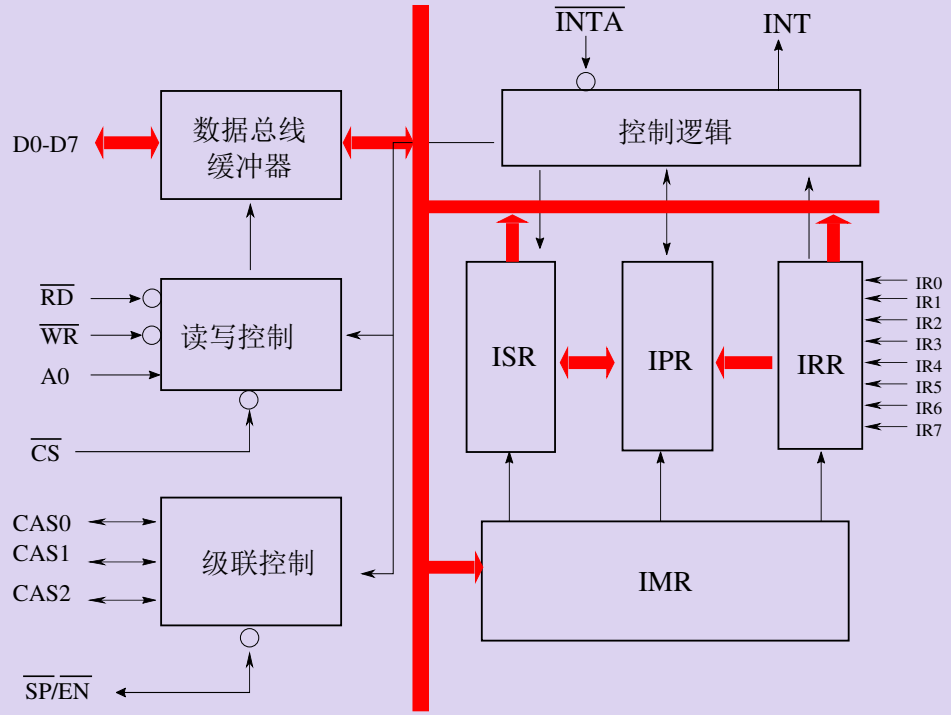
A typical PIC in PC

Interrupts

Interrupt request and acknowledge

- $IR_0 \sim IR_7$, interrupt request from devices
- INT,interrupt request to CPU or master
- $\overline{INTA}$,interrupt acknowledge

8259

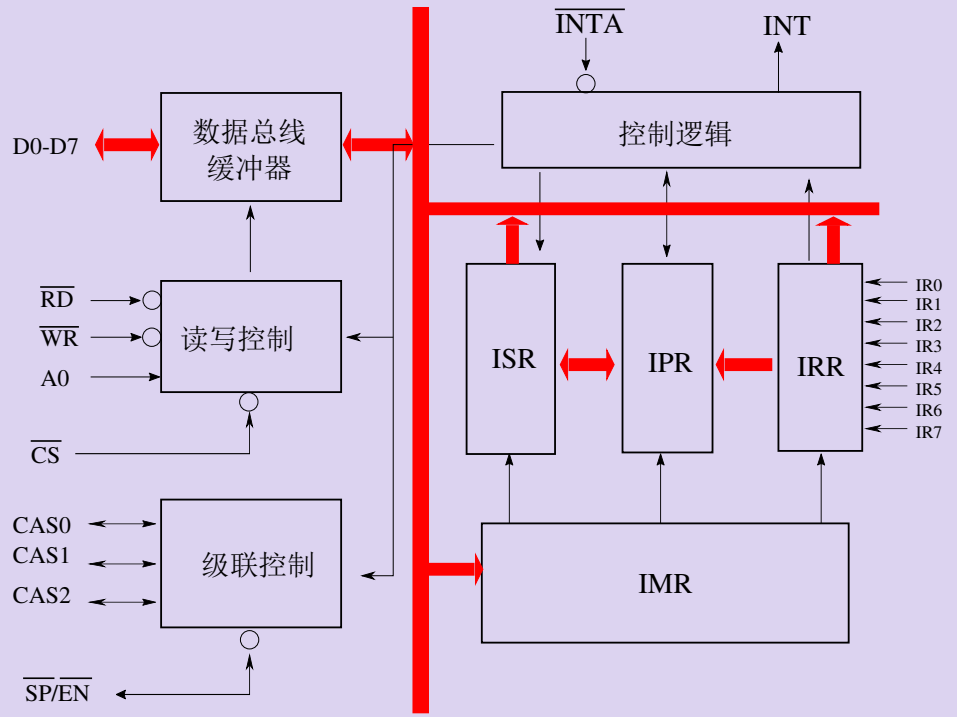


A typical PIC in PC

Cascade

Cascade $CAS_2 \sim CAS_0$,output(master) or input(slave)

8259

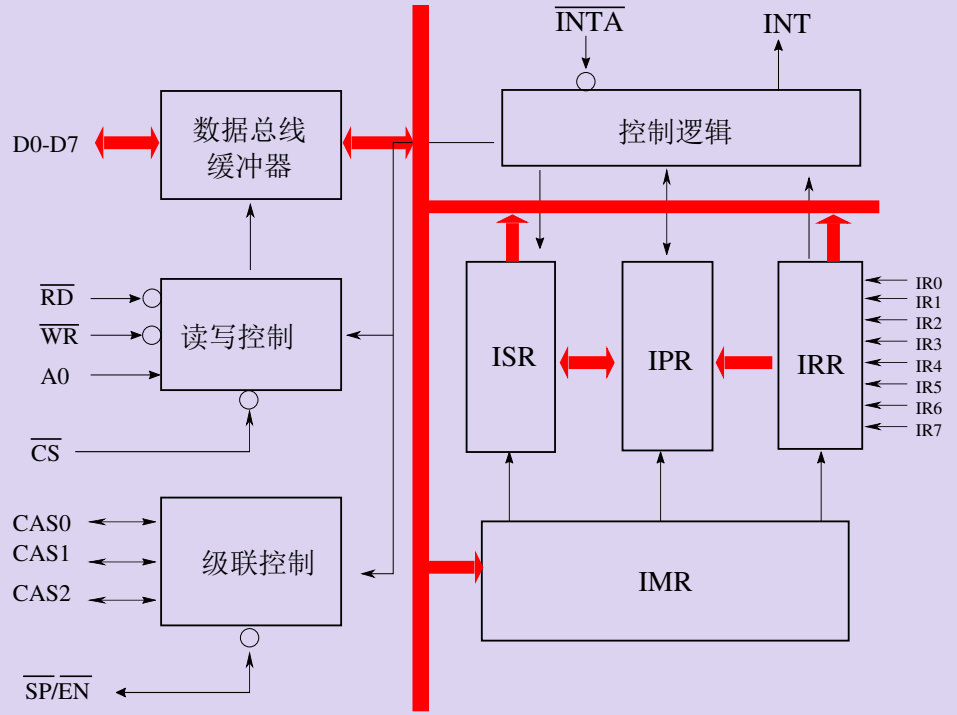


A typical PIC in PC

Control Logic

Controls, Read/write, etc., connect to CPU.

8259



Programming 8259A

1. ICW1, A0=0, D4=1.

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	A7	A6	A5	1	TM	ADI	SGL	IC4

2. ICW2,

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	T7	T6	T5	T4	T3	X	X	X

3. ICW3, A0=1, needed when cascade(SGL=0)

	S7	S6	S5	S4	S3	S2	S1	S0
Master								
Slave	×	×	×	×	×	D2	D1	D0

4. ICW4, needed by X86 systems(IC4=1)

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	SFNM	BUF	$M/\overline{S}$	AEOI	μPM

OCW1 to OCW3 can be accessed in any order after PIC initialized.

- OCW1, Interrupt mask

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	M7	M6	M5	M4	M3	M2	M1	M0

  Interrupt IR_i is masked when M_i is set to 1

- OCW2, priority. A0=0, D4 D3=00.

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	R	SL	EOI	0	0	L2	L1	L0

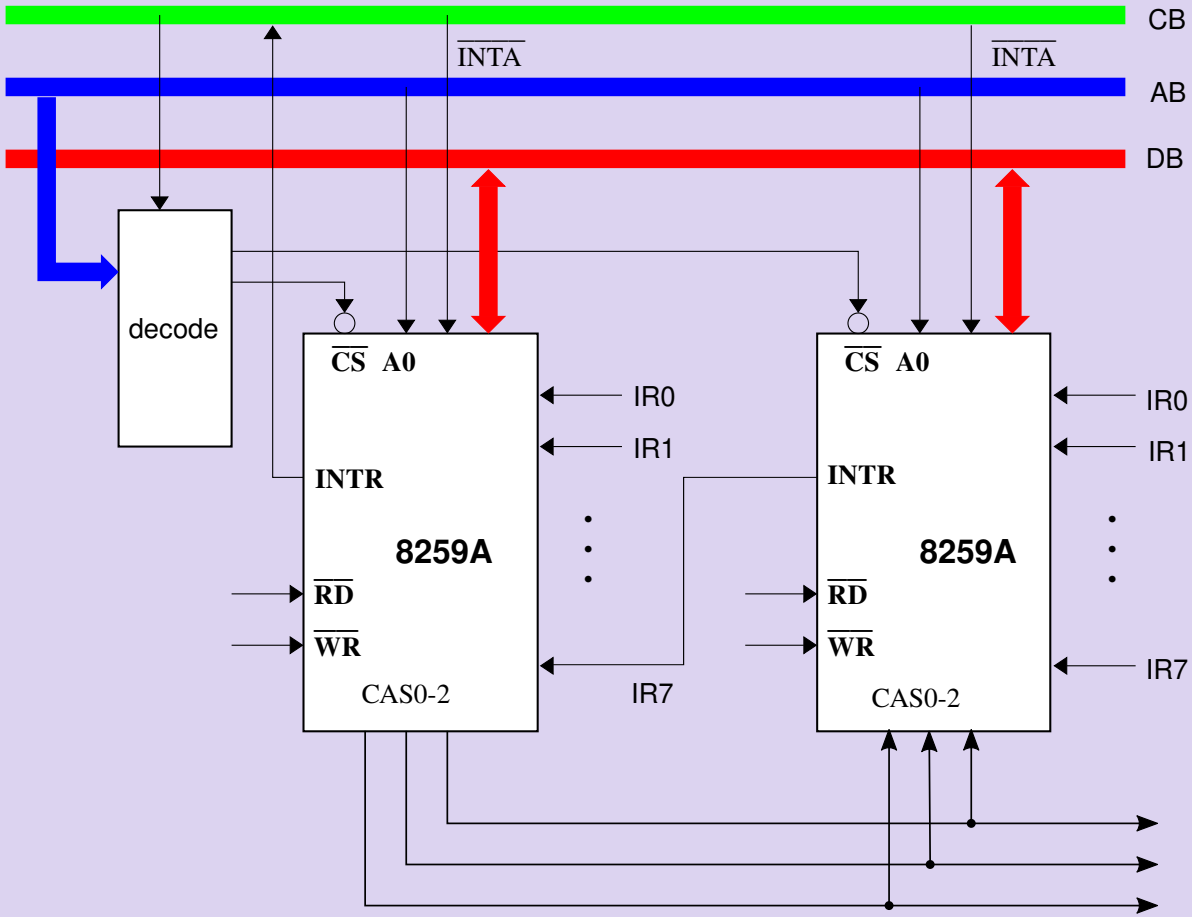
- OCW3, Polling mode. A0=0, D4 D3=01.

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	0	ESMM	SMM	0	1	P	RR	RIS

Cascade 8259A

A Cascaded structure can manager more devices.

Cascaded PIC



Interrupt programming

1. Disable interrupt
2. Initializing interrupt controller(may be initialized during booting)
3. Setting interrupt vector table
4. Enable interrupt

Interrupt service routine and interrupt setting are programmed parallel. OCWs of PIC can be accessed at any time during program. Some applications need to backup modified interrupt vector table entries and retrieve them later.

Direct Memory Access

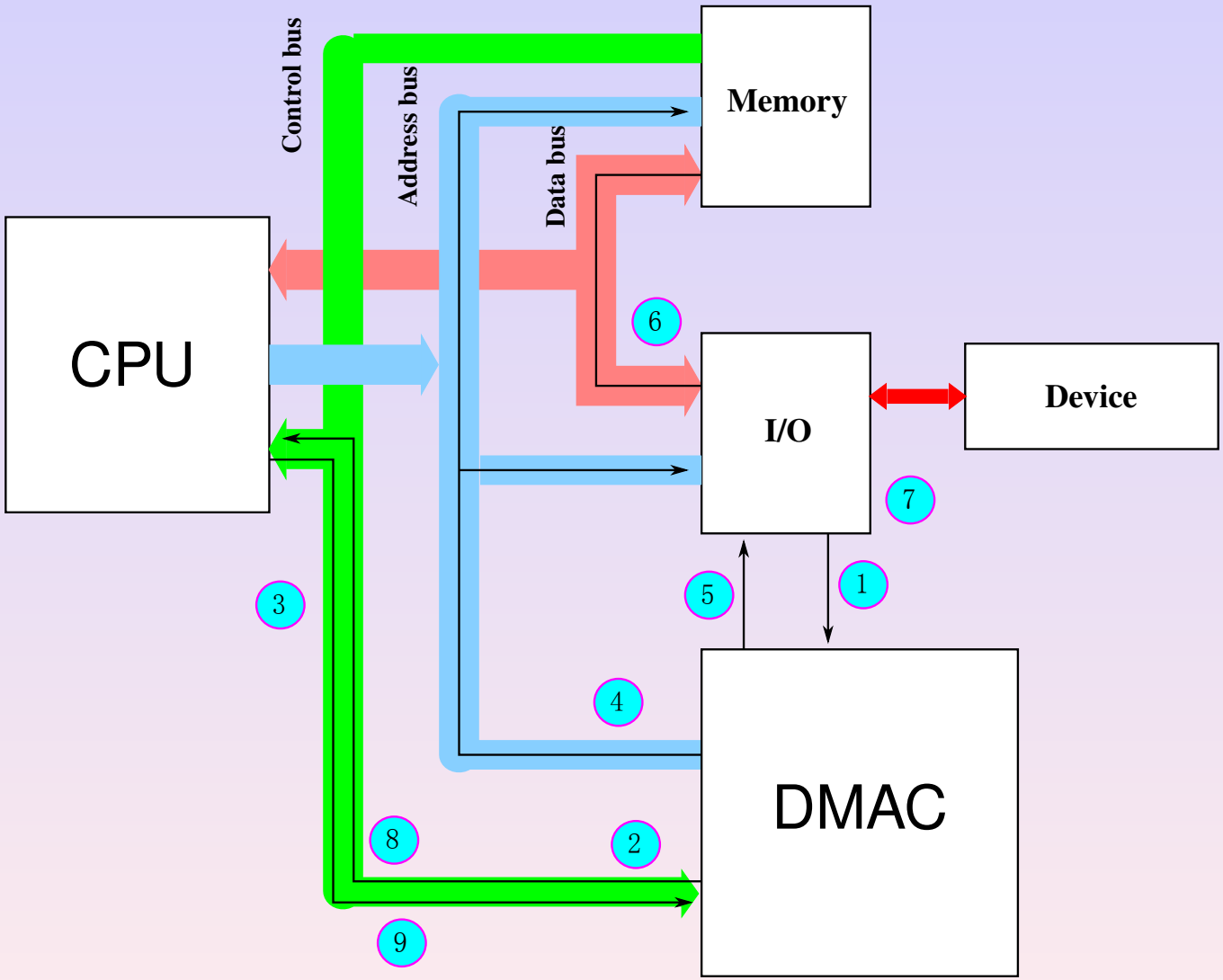
- During data transferring, CPU read from memory/device, write to memory/device without any processing.
- Direct memory access (DMA) is a feature of computer systems that allows certain hardware subsystems to access memory, independent of the CPU.
- DMA controller works under the direction of CPU. DMA is suitable for large, fast transferring.

Functions of DMAC

- DMA request (or bus request)
- Bus mastering
- Transferring (read/write, modifying address, counter)
- Signaling to CPU (interrupt)

A DMA event requests bus controlling from CPU.
DMA interrupts inform CPU postprocessing.

DMA Process


Control bus
Address bus
Data bus
Memory
CPU
I/O
Device
DMAC
1
2
3
4
5
6
7
8
9

Serial Ports

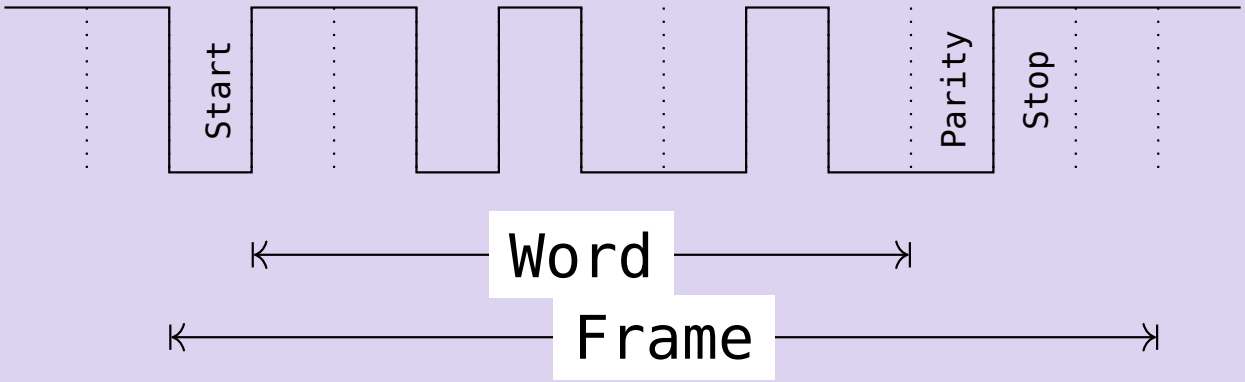
Serial communication must take followings into account:

- When bit begins
- How many bits in a word
- Error check reliability (often in a long distance application)
- Protocols.

Serial transferring are widely used in modern computer systems: RS-232, SPI, USB, I2C, I2S(sound system), SATA, Ethernet, 1-wire, PCIe, and more.

UART

Universal Asynchronous Receive and Transmit

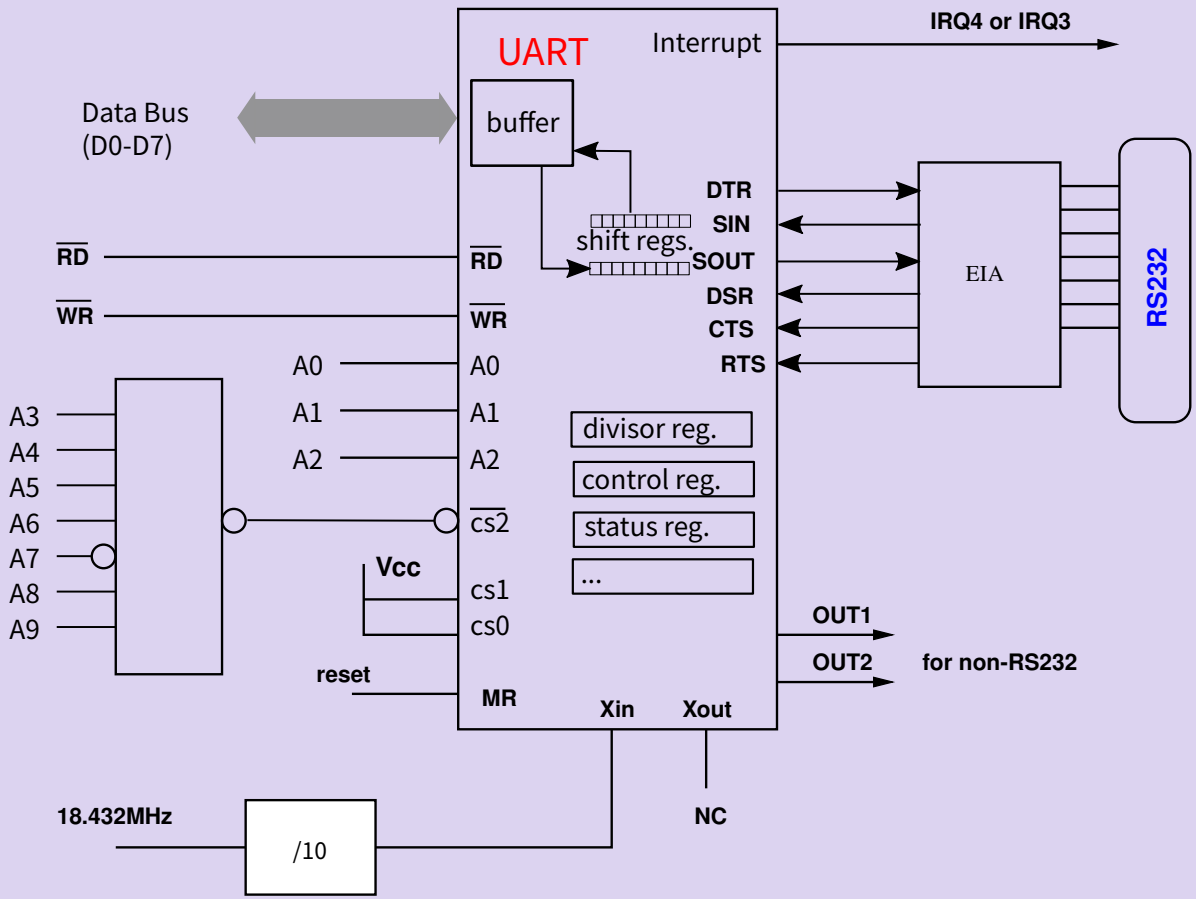


1. Send: CPU writes date into TX-register.
2. Receive:
   Data send into RX-register
   - Parity Error
   - Framing Error
   - Overrun Error

RS-232

RS-232, Recommended Standard 232 refers to a standard for serial transmission.

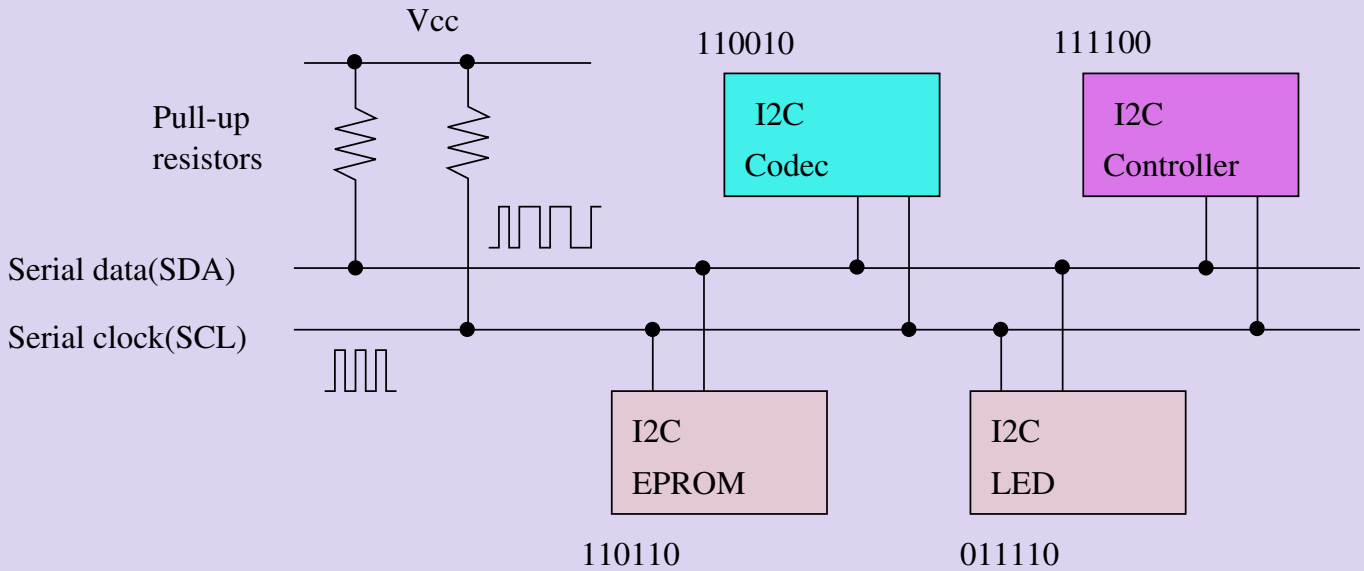
RS-232 in early PC



I2C

IIC (or I^2C, Inter-Integrated Circuit) is a synchronous, multi-master, multi-slave, packet switched, single-ended, serial (half-duplex) computer bus.

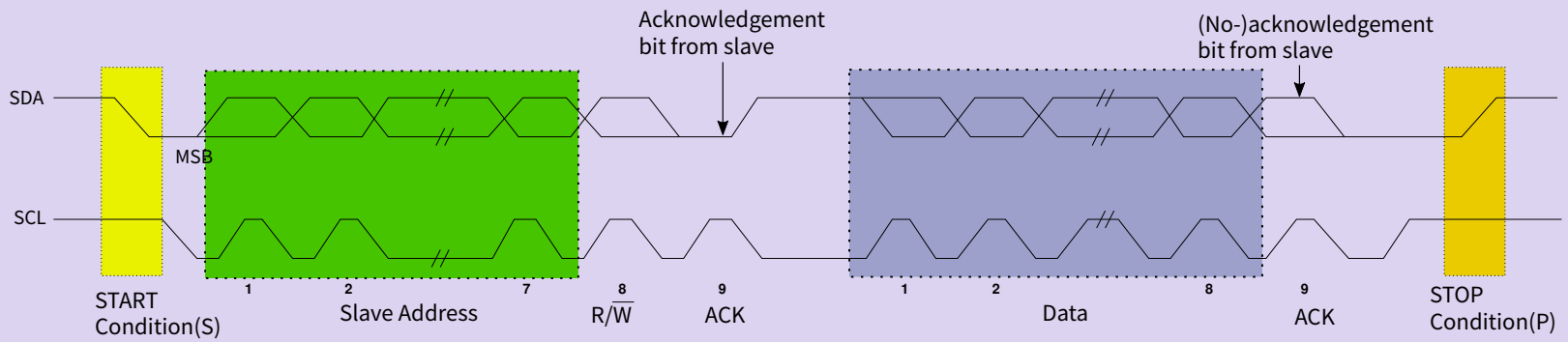
I2C



I2C Timing

I2C is widely used for attaching lower-speed (0.1–5Mbit/s) peripheral ICs to processors and microcontrollers in short-distance, intra-board communication.

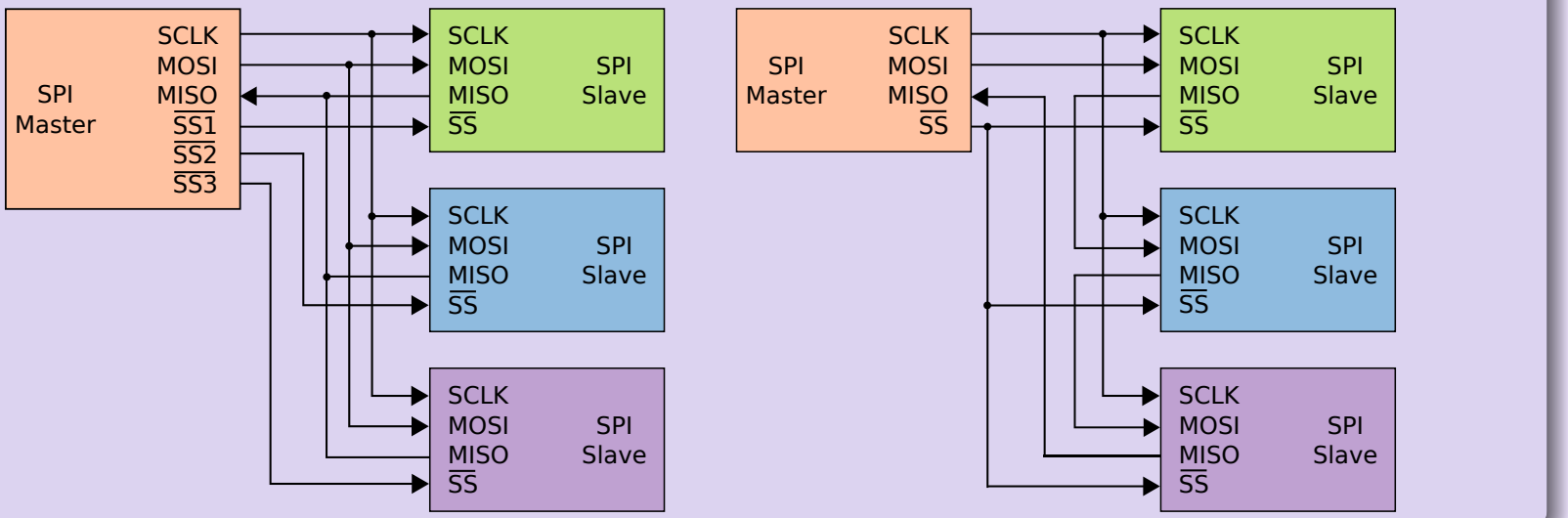
I2C Timing



SPI

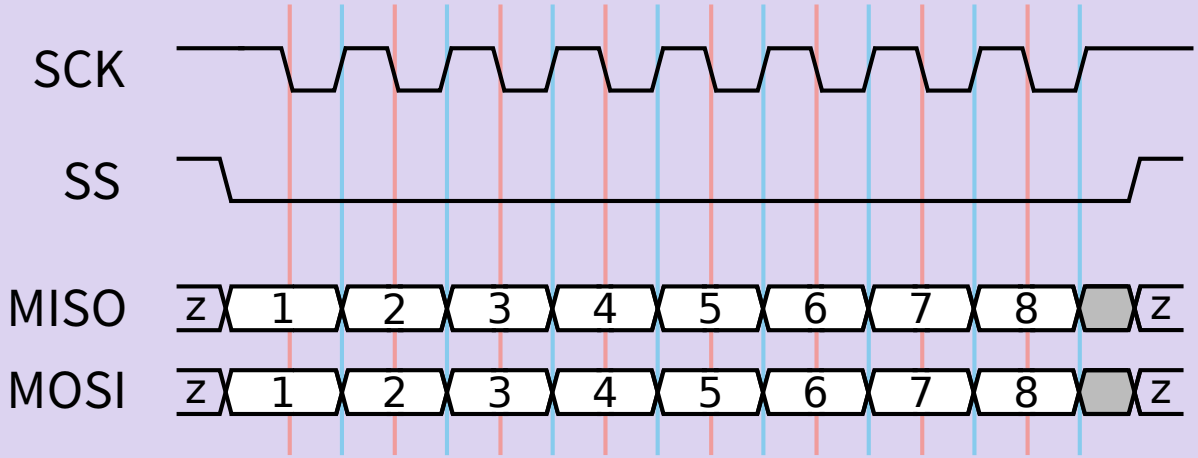
The Serial Peripheral Interface (SPI) is a synchronous serial communication interface specification (developed by Motorola) used for short-distance communication.
SPI devices communicate in full duplex mode using a master-slave architecture.

SPI



SPI Timing

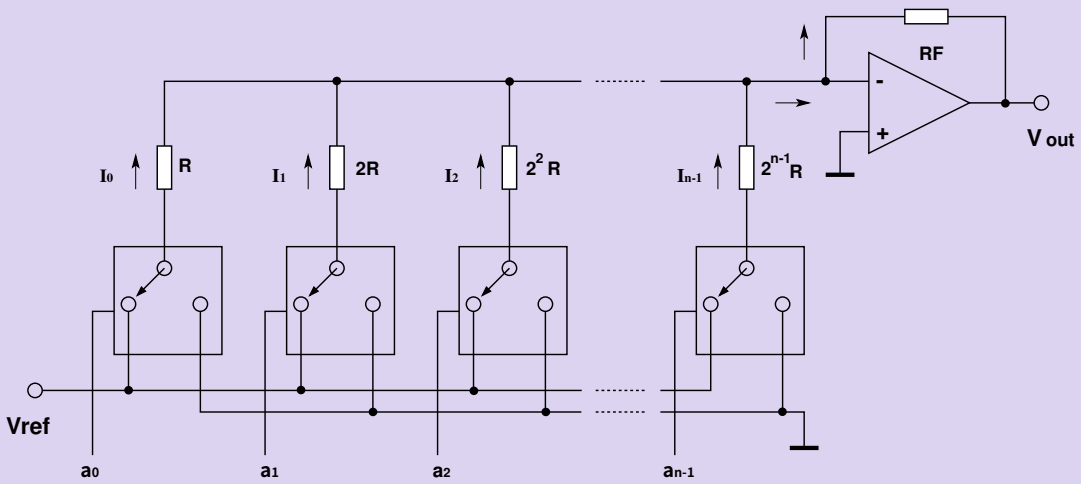
SPI Timing



DAC

A DAC converts a finite-precision number (fixed-point binary) into a physical quantity (voltage, etc.).

Binary-weighted DAC

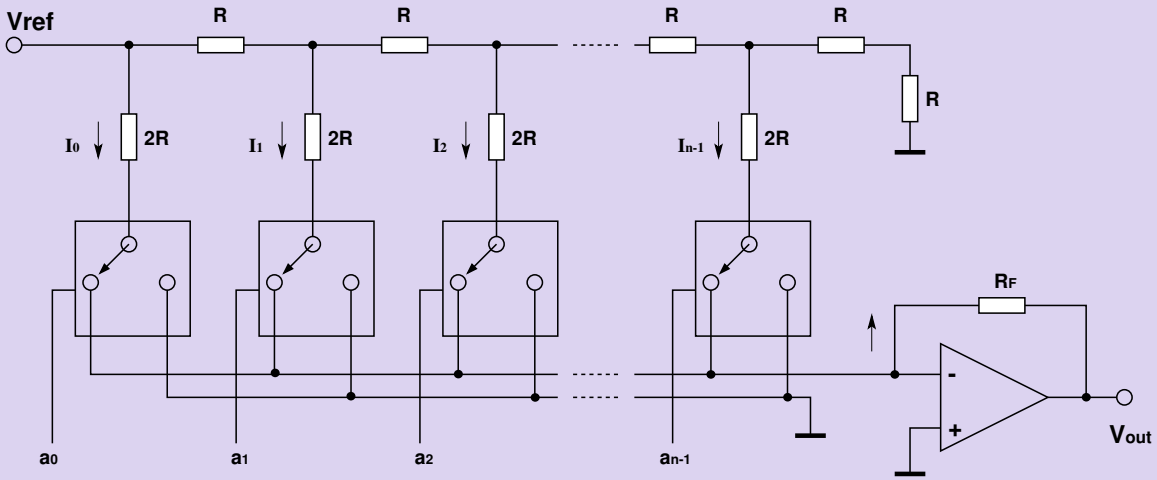


$$V_{out} = -V_{ref} \frac{R_F}{R} \sum_{i=0}^{n-1} a_i 2^{-i}$$

DAC

A DAC converts a finite-precision number (fixed-point binary) into a physical quantity (voltage, etc.).

R-2R ladder DAC



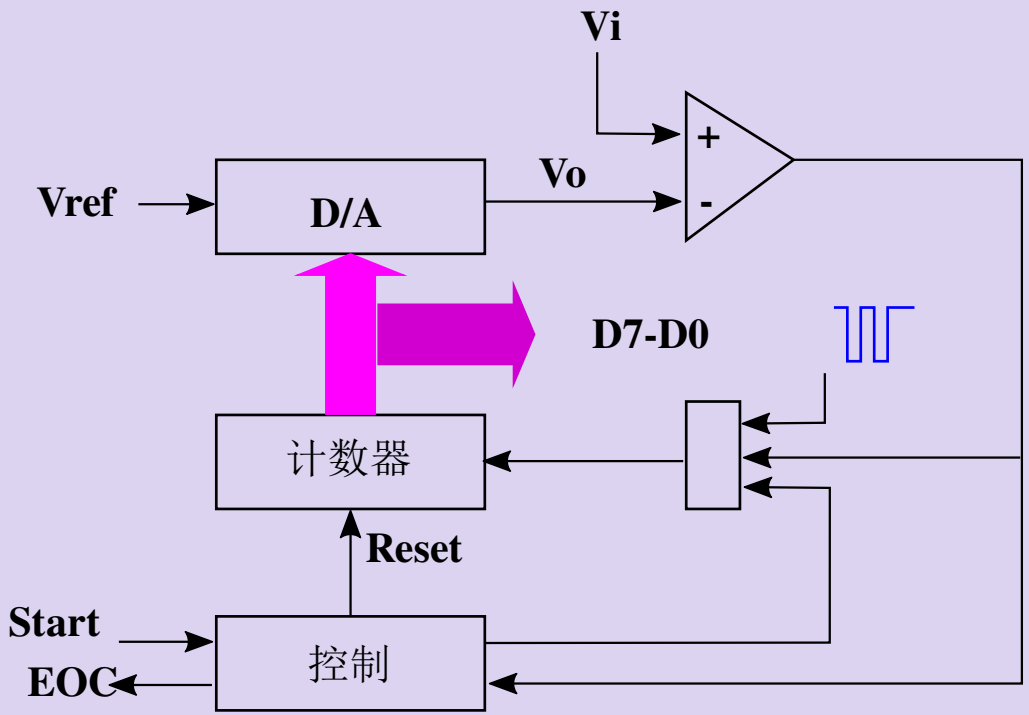
$$V_{out} = -V_{ref}\frac{R_F}{R}\sum_{i=0}^{n-1} a_i 2^{-i}$$

ADC

An ADC converts a continuous-time and continuous-amplitude analog signal to a discrete-time and discrete-amplitude digital signal.
The performance of an ADC is primarily characterized by its bandwidth (sampling rate) and signal-to-noise ratio (SNR) (resolution).

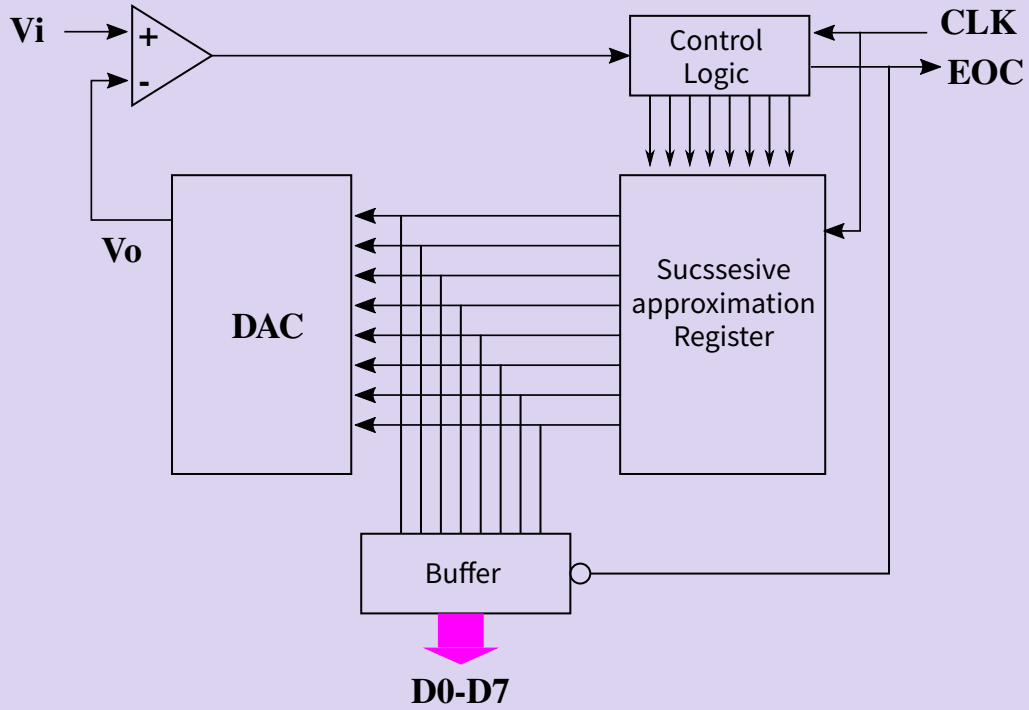
ADC

Counter type ADC



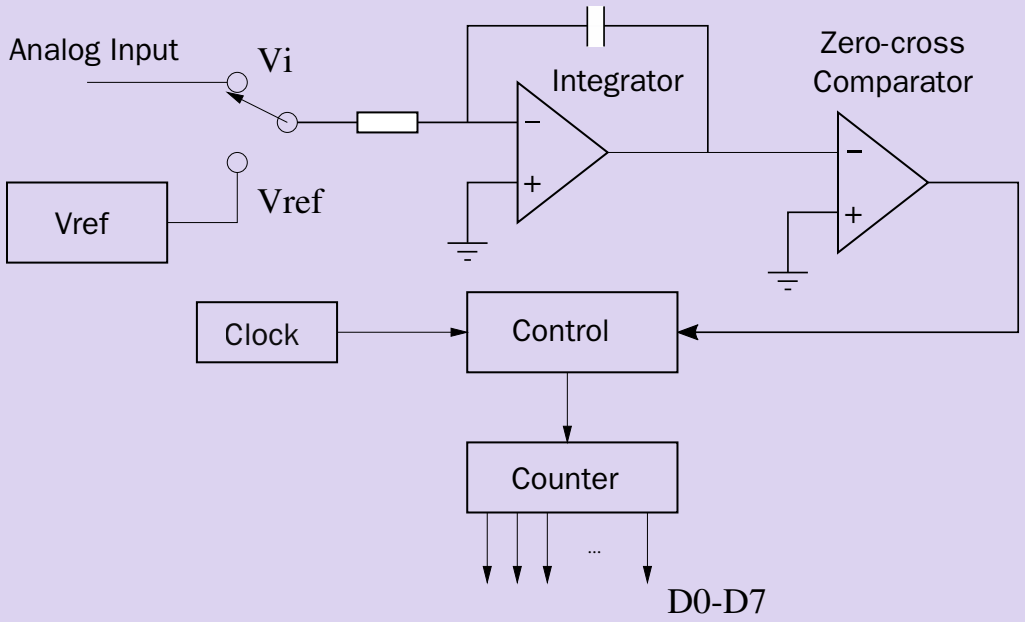
ADC

Sucssesive-Approximation ADC



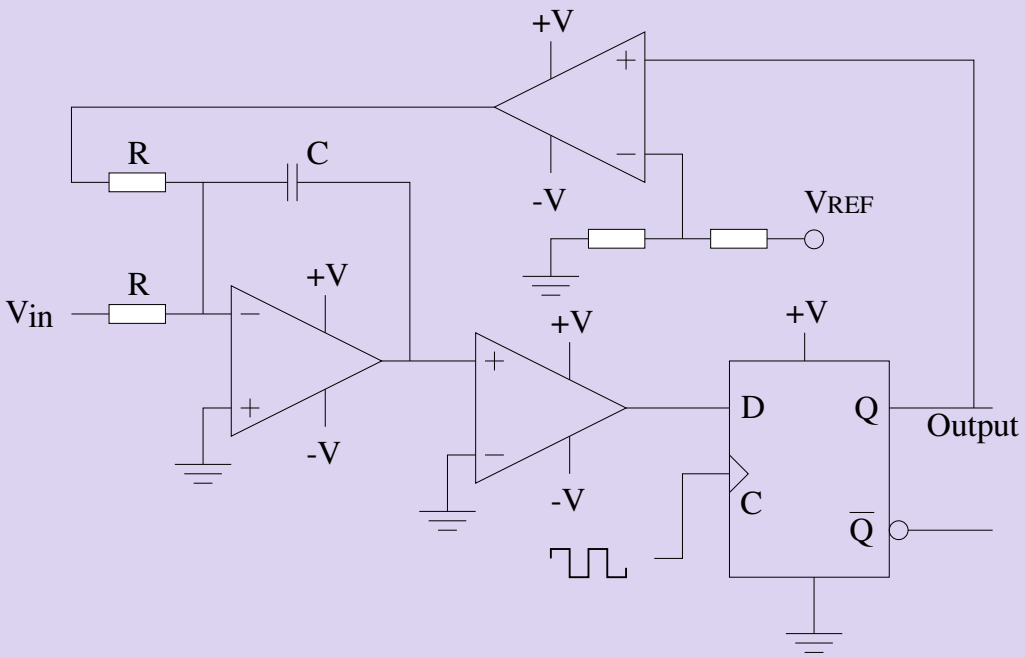
ADC

Dual Slope ADC



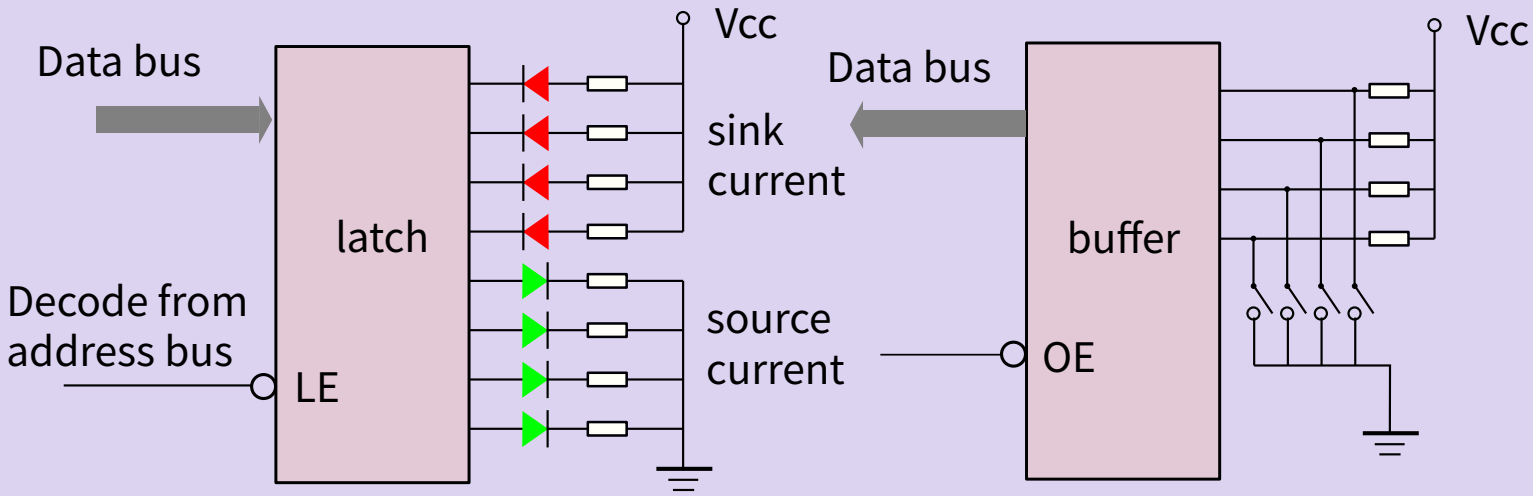
ADC

Sigma-Delta ADC



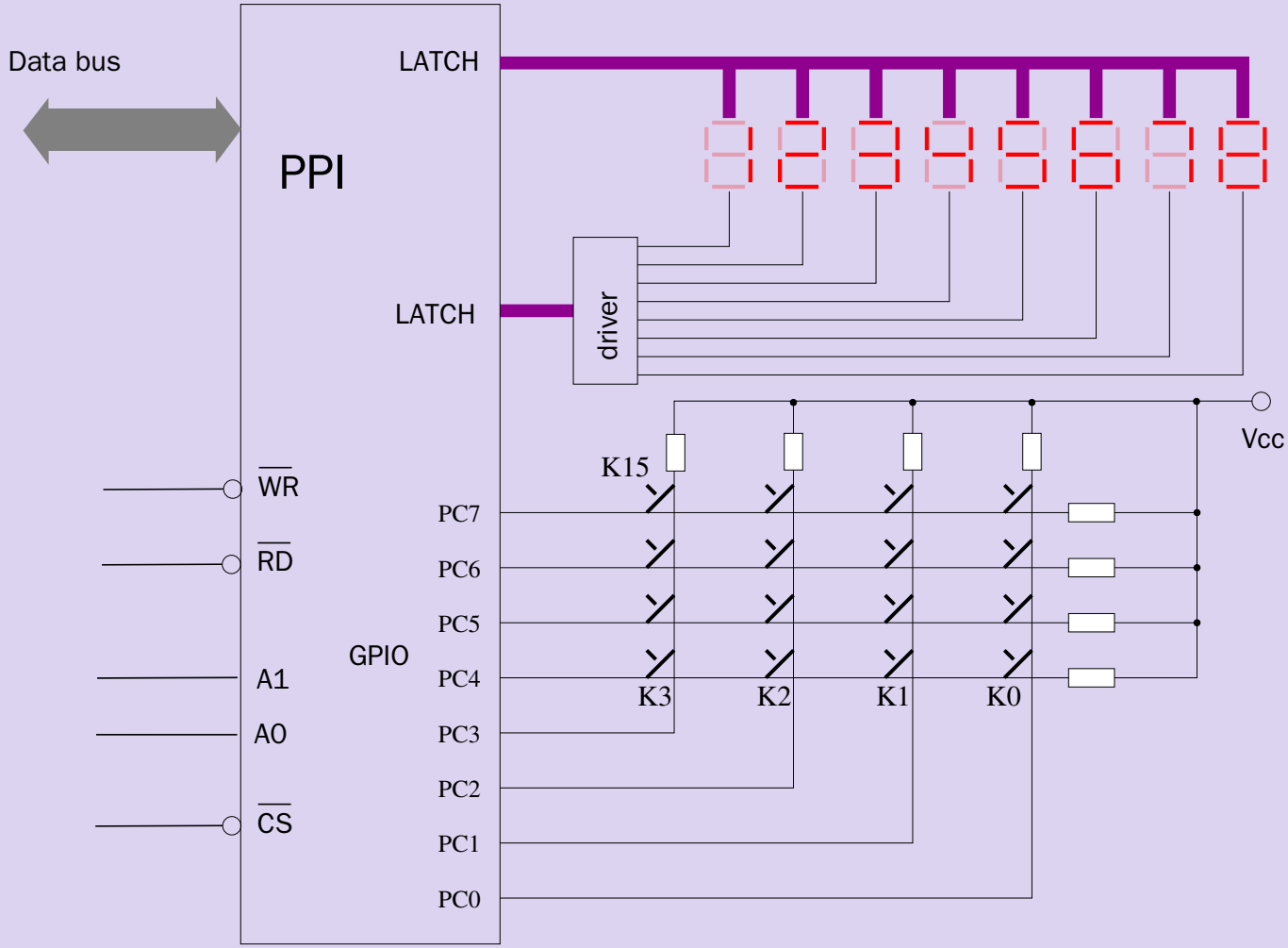
Simple IO Devices

Latch and buffer



Simple IO Devices

Devices via PPI



Summary

- Data transferring in may ways:
  - Direct
  - Polling, interfaces
  - Interrupt, Concepts about interrupt
  - Procedure of DMA
- Programmable interfaces (Programmable Interrupt Controller, Programmable Parallel Interface.)
- Transferring via Serial Port
- Ananlgue and Digital Converters.