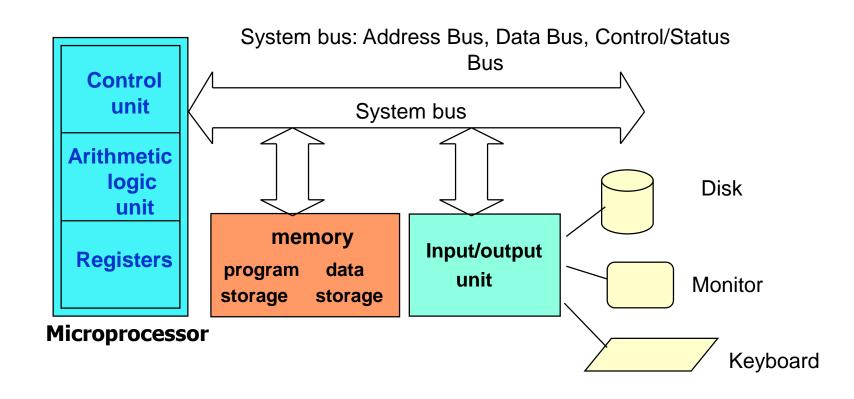
Microprocessor Systems

Dr. Gökhan İnce

Topics

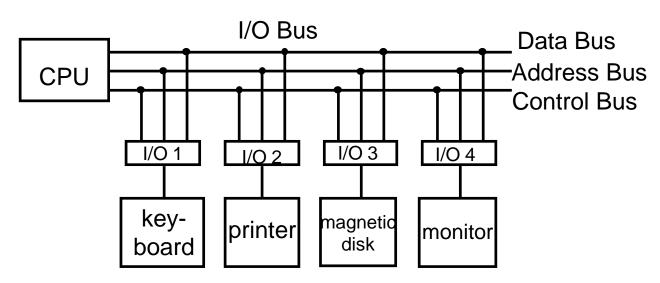
- I/O Interface
- I/O Transfer Synchronization
- Parallel Communication

Computer Organization



I/O Interface

- Peripheral devices (also called I/O devices) are pieces of equipment that exchange data with a CPU
 - Examples of I/O devices include switches, push-buttons, light-emitting diodes (LED), monitors, printers, modems, keyboards, and disk drives.
- Interface chips are needed to resolve the differences between CPU and I/O devices
 - Provides a method for transferring information between internal storage (such as memory and CPU registers) and external I/O devices



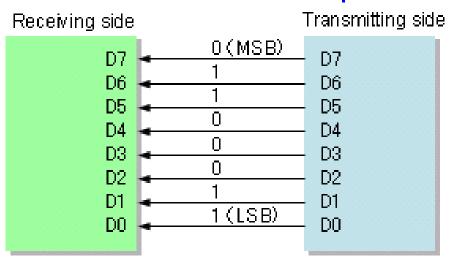


- Resolves the differences between the computer and peripheral devices
 - Peripherals are electromechanical and electromagnetic devices, CPU and memory are electronic devices
 - Data codes and formats in peripherals are different
- A synchronization mechanism is needed.
 - Data transfer rate of peripherals are usually slower
- Data transfer between an I/O device and the CPU can be proceeded bit-by-bit (serial) or in multiple bits (parallel).

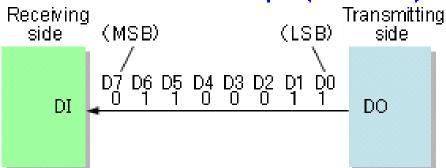
I/O Interfacing

The I/O communication can be parallel or serial

Parallel interface example



Serial interface example (MSB first)



I/O Addresses

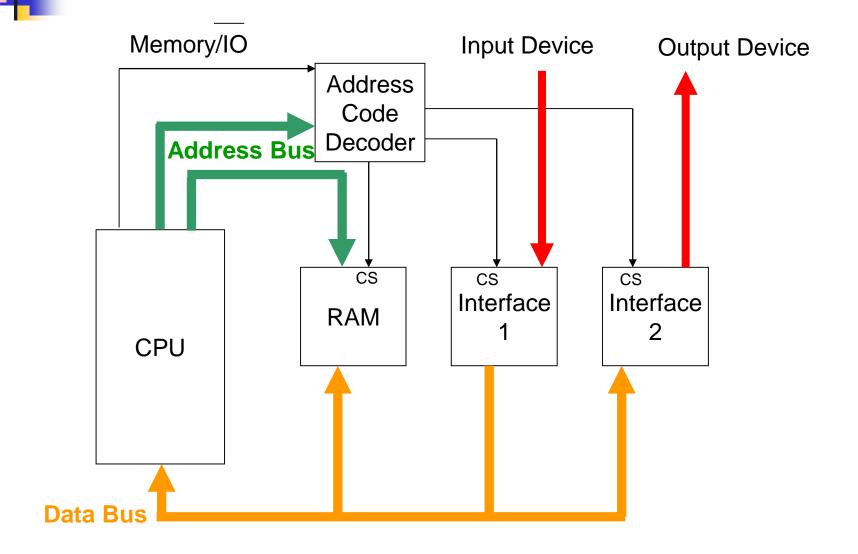
Isolated I/O Map

- Separate I/O read/write control lines in addition to memory read/write control lines
- Separate (isolated) memory and I/O address spaces
- Distinct input and output instructions

Memory Mapped I/O

- A single set of read/write control lines (no distinction between memory and I/O transfer)
- Memory and I/O addresses share the common address space
 - reduces memory address range available
- No specific input or output instructions
 - The same memory reference instructions can be used for I/O transfers
- Considerable flexibility in handling I/O operations

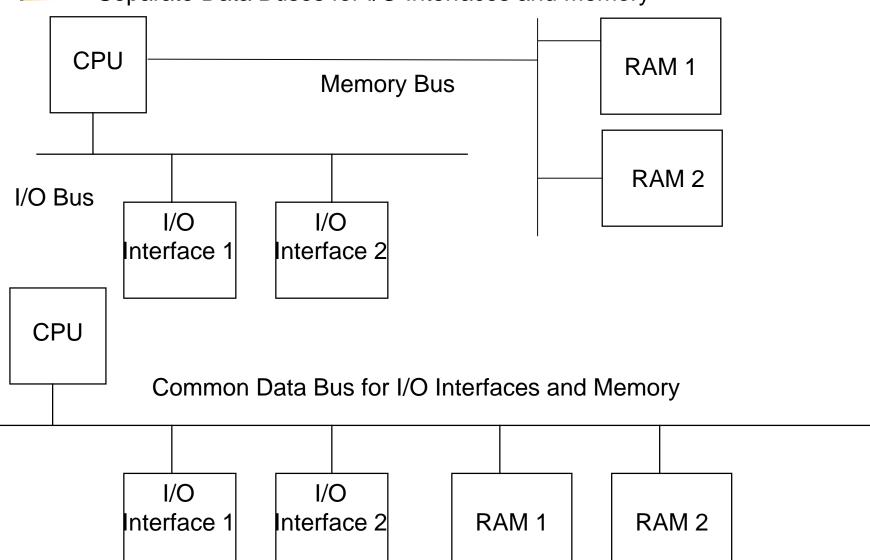
Isolated I/O Map-based Interfacing





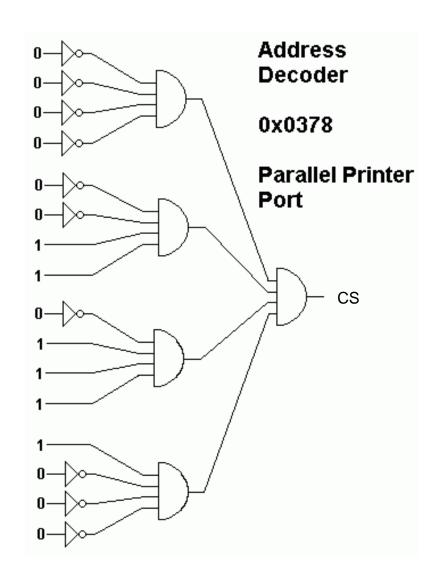
Connection of I/O Interfaces

Separate Data Buses for I/O Interfaces and Memory



9

Address Decoder for Memory-mapped I/O interface



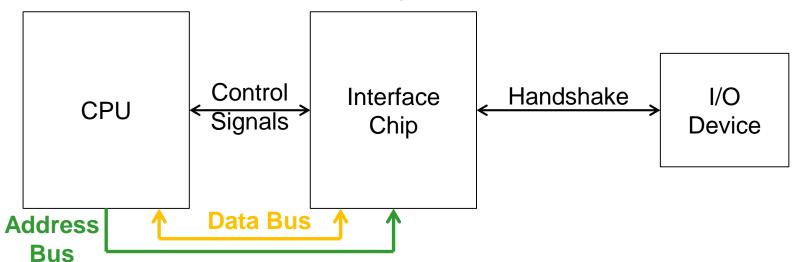
Topics

- I/O Interface
- I/O Transfer Synchronization
- Parallel Communication



I/O Transfer Synchronization

- The roles of the interface chip
 - Synchronizing data transfer between CPU and interface chip
 - Programmed I/O
 - Interrupt driven I/O
 - Direct Memory Access (DMA)
 - Synchronizing data transfer between interface chip and I/O device
 - Transmitter initiated handshaking
 - Receiver initiated handshaking



CPU and I/O Interfaces

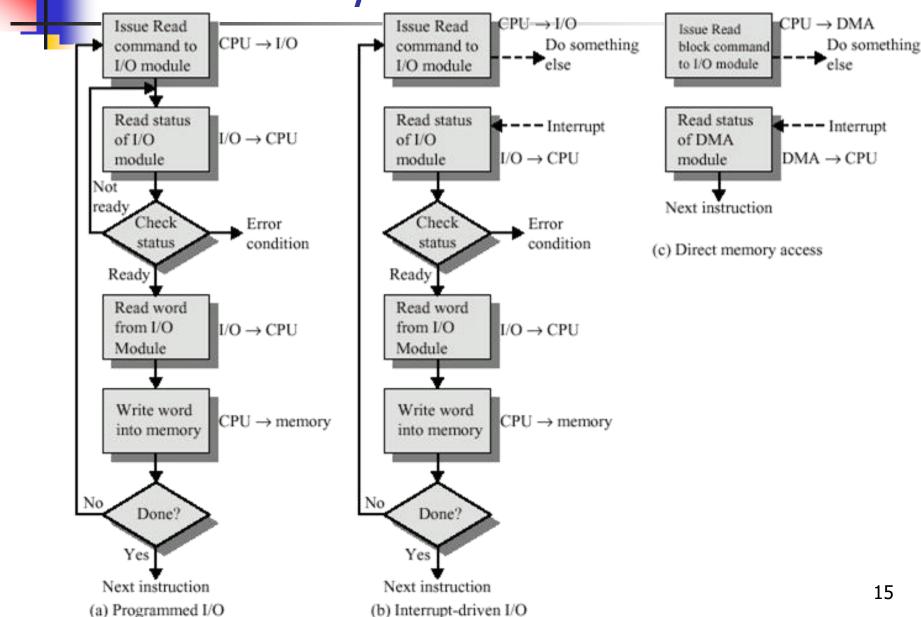
- Programmed I/O
 - Sensing status
 - Read/write commands
 - Transferring data
 - CPU waits for I/O module to complete operation
 - Wastes CPU time
- Interrupt Driven I/O
 - Overcomes CPU waiting
 - No repeated CPU checking of device
 - I/O module interrupts when ready
- Direct Memory Access (DMA)
 - Additional Module (hardware) on bus
 - DMA controller takes over from CPU for I/O operations



Synchronizing the CPU and the Interface Chip

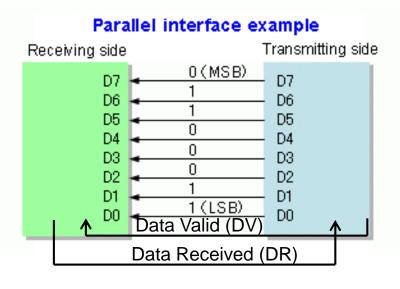
- The polling method:
 - for input -- the microprocessor checks a status bit of the interface chip to find out if the interface has received new data from the input device.
 - for output -- the microprocessor checks a status bit of the interface chip to find out if it can send new data to the interface chip.
- The interrupt-driven method:
 - for input -- the interface chip interrupts the microprocessor whenever it has received new data from the input device.
 - <u>for output</u> -- **the interface chip** interrupts the microprocessor whenever it **can accept** new data from the microprocessor.

CPU and I/O Interfaces

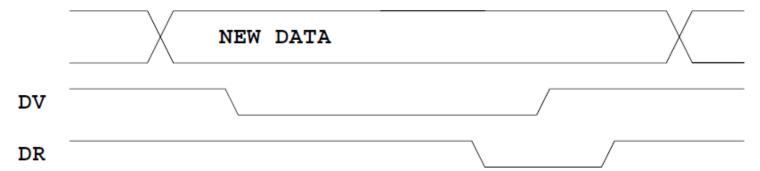


Synchronization of interface chip and peripheral

- Transmitter initiated 2-wire handshaking
 - In parallel communication transmitter and receiver uses handshaking protocol
 - In short distance serial communication, handshaking can also be used.



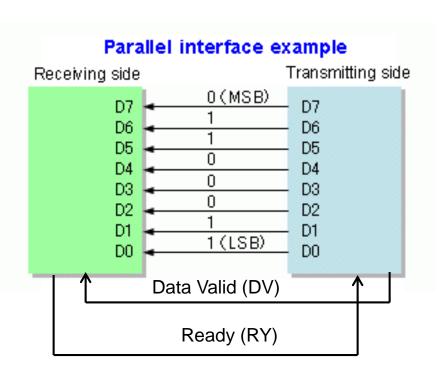
- Transmitter puts the data on the data lines and brings DV low to indicate new data is available
- When the receiver sees the new data is available, it reads the data, then brings DR low to say that it has read the data
- When the sending computer sees DR go low, it brings DV high
- When the receiving computer sees DV go high, it brings DR high
- Both sides are now ready for the next data transfer





Synchronization of interface chip and peripheral

Receiver initiated 2-wire handshaking



- Receiver brings Ready (RY) to low to indicate it is ready to receive new data
- Transmitter places data on data lines and lowers Data Valid (DV)
- Receiver sees DV and accepts data, then brings RY high
- Transmitter sees RY high and raises DV to high
- Both computers are now ready for the next data transfer

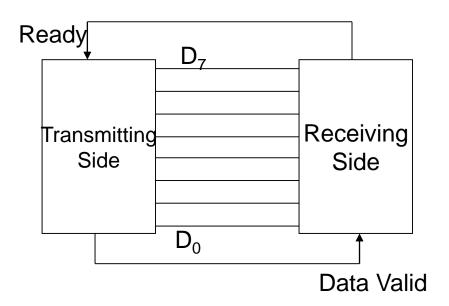
Topics

- I/O Interface
- I/O Transfer Synchronization
- Parallel Communication



Parallel Communication

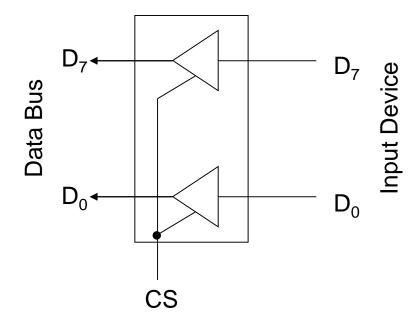
- Each bit is transferred along its own line and bits are transferred at the same time
- In addition to eight parallel data lines, other lines are used to read status information and send control signals.





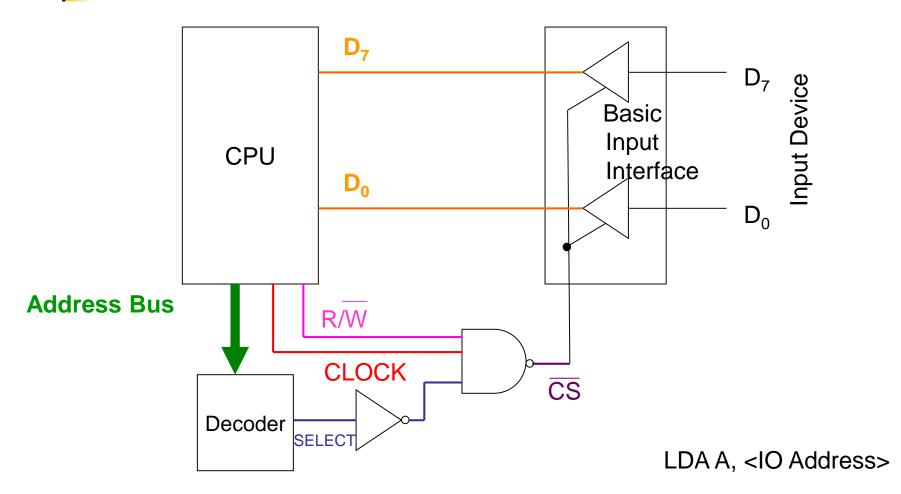
Basic Parallel Input Interface

- Tri-state buffers are used. (74LS244)
- Output pins connected to CPU Data Bus, Input pins connected to the Input device.





Basic Parallel Input Interface – CPU Connections



Basic Parallel Input Interface

- CPU Control -

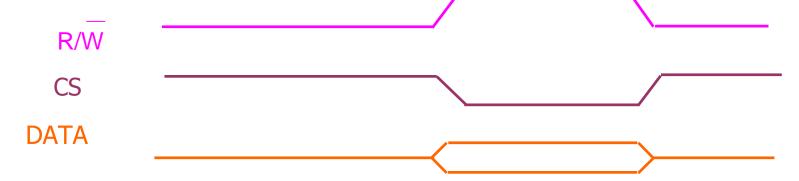
1) Address placed on address bus at the falling edge of clock

ADDRESS ADDRESS

2) Address decoded to form Select

SELECT _____

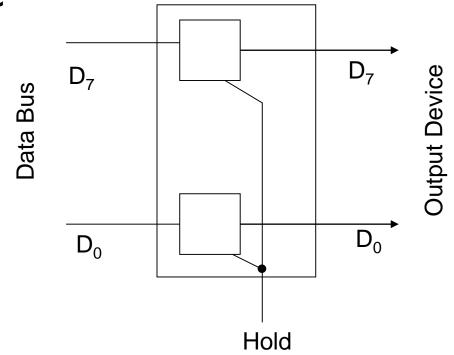
3) NAND {CLK, SELECT, R/W} to form CS



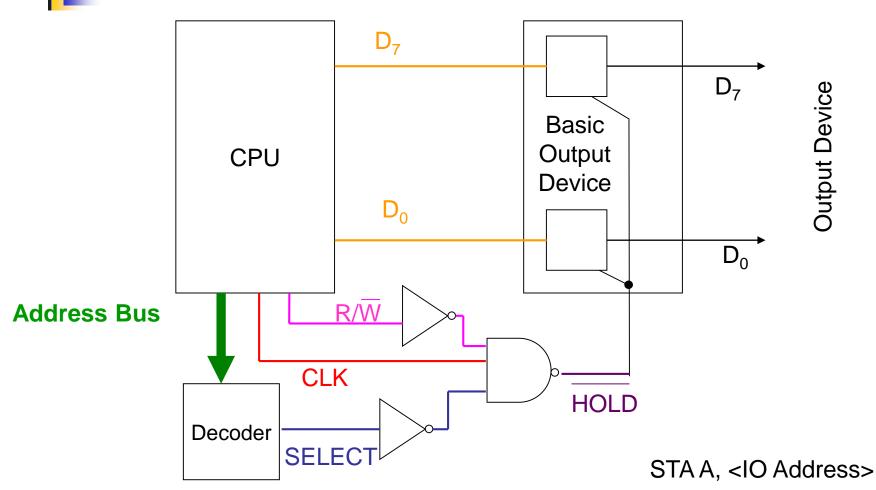


Basic Parallel Output Interface

- D Latches can be used (74LS374 Octal D Latch)
- Inputs of D latches are connected to the Data Bus. The outputs connected to the output device



Basic Parallel Output Interface – CPU Connections



Basic Parallel Output Interface – CPU Control

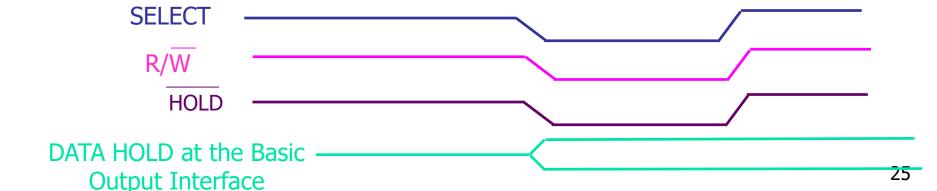
1) Address placed on address bus at the falling edge of clock



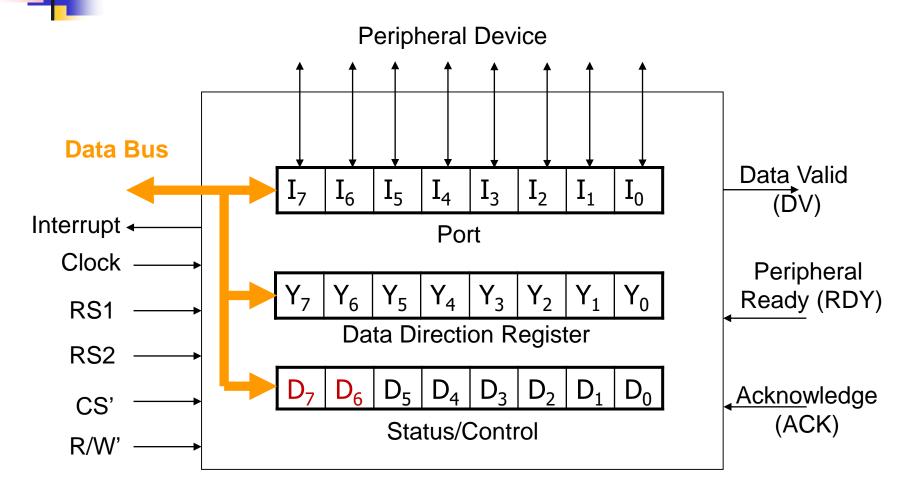
2) Data is placed on the data bus

DATA

- 3) Address decoded to form Select
- 4) NAND {CLK, SELECT, R/W} to form HOLD



- Can be programmed to handle input or output data
- PIA contains
 - Port(s): Connects the PIA to peripheral devices. Each pin of the port can be conditioned as transmit (TX) or receive (RX)
 - Data Direction Register: Conditions the port pins as TX or RX
 - 1 : Pin is Transmitter
 - 0 : Pin is Receiver
 - Control / Status Register:
 - Status bits: Indicates the status of the handshaking bits
 - Control bits: Used to control handshaking signals and interrupt conditions



RS: Register Select

CS: Control Signal

- No consensus on the condition of raising the flags $(1 \rightarrow 0)$ or $(1 \rightarrow 1)$
- READY (RDY) Input
 - Indicates whether the peripheral device is ready
 - D7 Status Bit indicates RDY is received
 - Active High
 - D1 and D0 control bits determine whether an interrupt will be generated when RDY is received
 - D7 is cleared when Status/Control register is read

D_1D_0	Ready (RDY) Input	Interrupt Output
0 0	$1 \rightarrow 0 => D_7 = 1$	High, No interrupt
0 1	$0 \rightarrow 1 => D_7 = 1$	High, No interrupt
1 0	$1 \rightarrow 0 => D_7 = 1$	Hi->Low->Hi, Interrupt
1 1	$0 \rightarrow 1 => D_7 = 1$	Hi->Low->Hi, Interrupt

28

ACKNOWLEDGE (ACK) Input

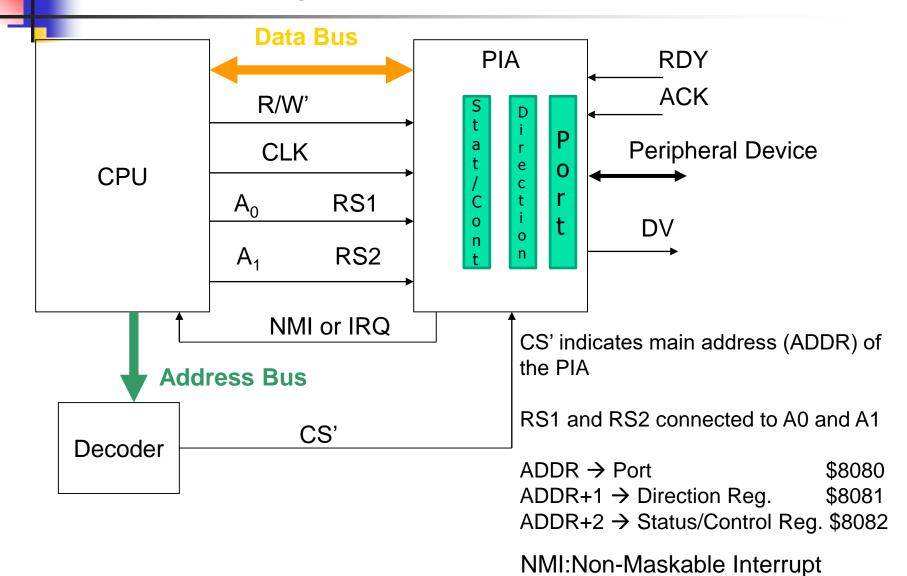
- Indicates whether the peripheral device received the data
- D6 Status Bit indicates ACK is received
 - Active High
- D3 and D2 control bits determine whether an interrupt will be generated when ACK is received
- D6 is cleared when Status/Control register is read

D_3D_2	Acknowledge (ACK) Input	Interrupt Output
0 0	$1 \rightarrow 0 => D_6 = 1$	High, No interrupt
0 1	$0 \to 1 => D_6 = 1$	High, No interrupt
1 0	$1 \rightarrow 0 => D_6 = 1$	Hi->Low->Hi, Interrupt
1 1	$0 \to 1 => D_6 = 1$	Hi->Low->Hi, Interrupt

- No consensus on the handshake conditions
- DATA VALID (DV) Output
 - Indicates valid data is at the Port
 - D5 and D4 control bits determine the DV conditions

D_5	D ₄	DATA VALID (DV)
0	0	DV is reset (low)
0	1	DV is set (high)
1	0	1 after the data is loaded on the port
1	1	0 after the data is loaded on the port

EDU-PIA / CPU Connections





EDU-PIA / CPU Connections

- Example I: The EDU-PIA is connected to an 8-bit microprocessor with 16-line address bus. The main address of the PIA is \$A0A0. The first four pins of the PIA are connected to four switches. The last four bits of the PIA are connected to LEDs.
 - Design a system that will illuminate the LEDs depending on the switch positions. The LEDs will illuminate after the user arranges the switches and presses a button.



Simple Output Device - LED

LED w resistance

LED is ON for an output of zero

Most LEDs drop a voltage of 1.7 to 2.5 volts and

need about 10mA

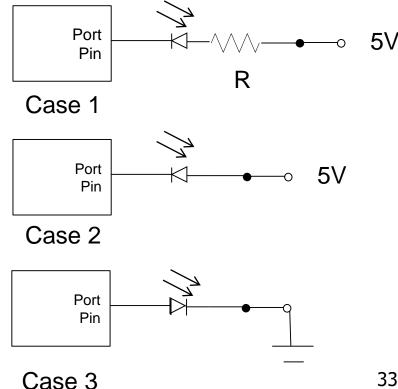
Current is (5-2)/R

LED wo resistance

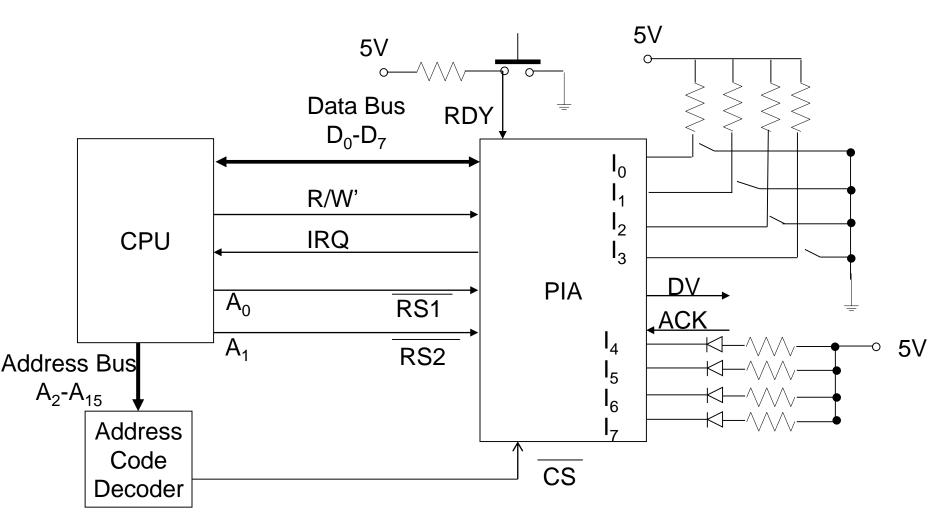
- Too much current
- Failure of Port or LED

LED wo resistance

- Not enough drive (1mA)
- LED too dim



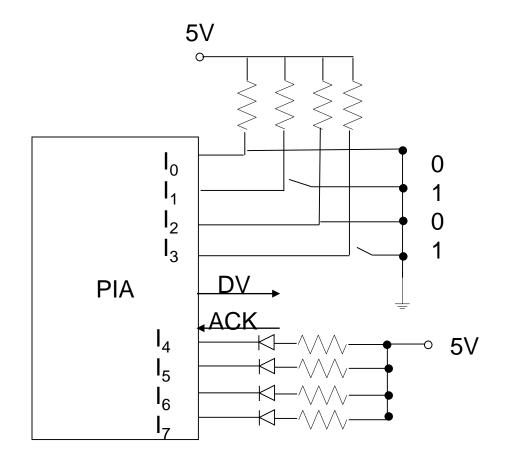
Exa





PORT	EQ	\$A0A0
DIRECT	EQ	\$A0A1
STATCON	EQ	\$A0A2

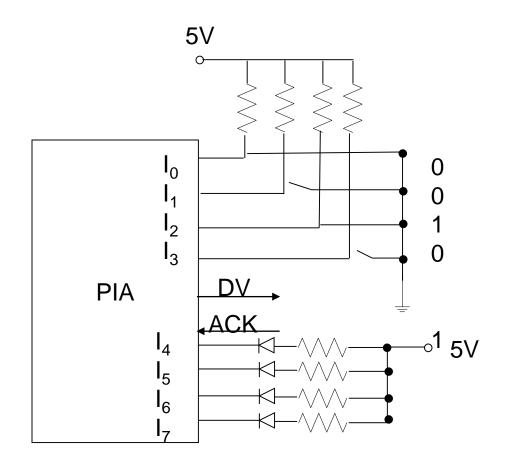
START	LDA	A,\$F0
	STA	A, DIRECT
	LDA	A,\$00
	STA	A, STATCON
REW	LDA	A, <statcon></statcon>
	TST	A,\$80
	BEQ	REW
	T.DA	A. <port></port>





PORT	EQ	\$A0A0
DIRECT	EQ	\$A0A1
STATCON	EQ	\$A0A2

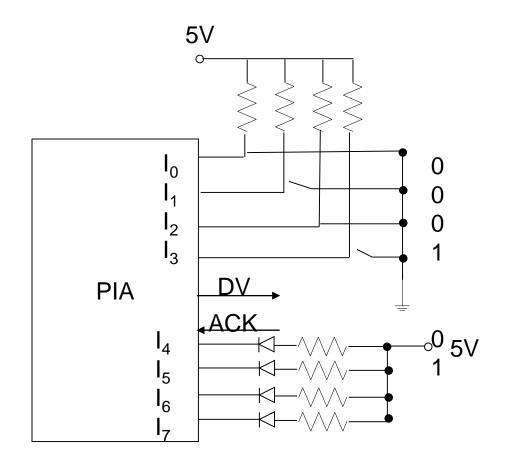
START	LDA	A,\$F0
	STA	A, DIRECT
	LDA	A,\$00
	STA	A, STATCON
REW	LDA	A, <statcon></statcon>
	TST	A,\$80
	BEQ	REW
	LDA	A, <port></port>
	SHL	A





PORT	EQ	\$A0A0
DIRECT	EQ	\$A0A1
STATCON	EQ	\$A0A2

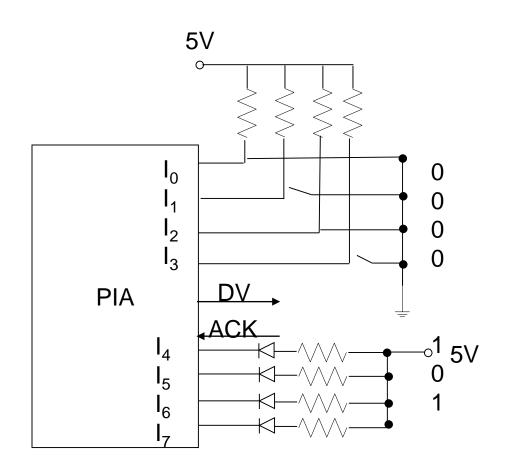
START	LDA	A,\$F0
	STA	A, DIRECT
	LDA	A,\$00
	STA	A, STATCON
REW	LDA	A, <statcon></statcon>
	TST	A,\$80
	BEQ	REW
	LDA	A, <port></port>
	SHL	A
	SHL	A





PORT	EQ	\$A0A0
DIRECT	EQ	\$A0A1
STATCON	EQ	\$A0A2

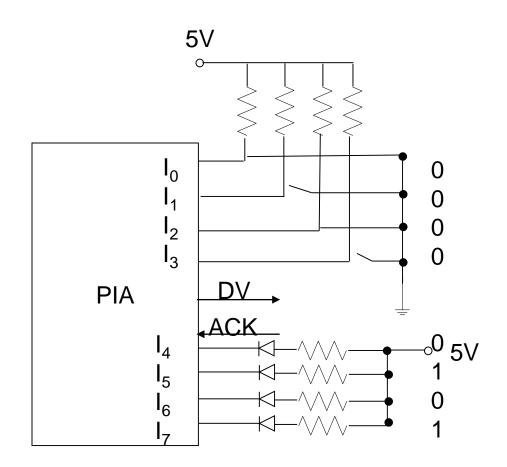
START	LDA	A,\$F0
	STA	A, DIRECT
	LDA	A,\$00
	STA	A, STATCON
REW	LDA	A, <statcon></statcon>
	TST	A,\$80
	BEQ	REW
	LDA	A, <port></port>
	SHL	A
	SHL	A
	SHL	A





PORT	EQ	\$A0A0
DIRECT	EQ	\$A0A1
STATCON	EQ	\$A0A2

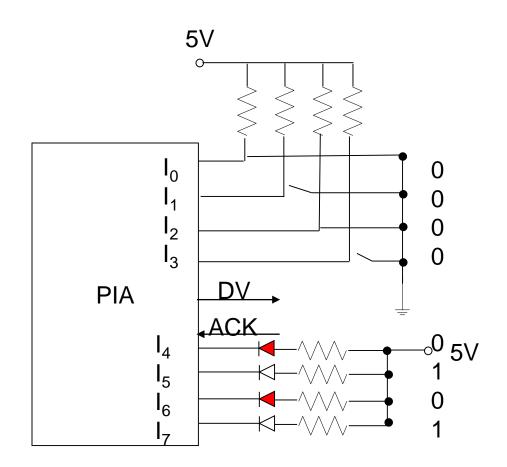
START	LDA	A,\$F0
	STA	A, DIRECT
	LDA	A,\$00
	STA	A, STATCON
REW	LDA	A, <statcon></statcon>
	TST	A,\$80
	BEQ	REW
	LDA	A, <port></port>
	SHL	A





PORT	EQ	\$A0A0
DIRECT	EQ	\$A0A1
STATCON	EQ	\$A0A2

START	LDA	A,\$F0
	STA	A, DIRECT
	LDA	A,\$00
	STA	A, STATCON
REW	LDA	A, <statcon></statcon>
	TST	A,\$80
	BEQ	REW
	LDA	A, <port></port>
	SHL	A
	STA	A, <port></port>
	BR	REW





Parallel vs. Serial Connection

- Before the development of high-speed serial technologies, the choice of parallel links over serial links was driven by these factors:
 - Speed
 - Cable length
 - Complexity

The decreasing cost of integrated circuits, combined with greater consumer demand for speed and cable length, has led to parallel communication links becoming deprecated in favor of serial links; for example, IEEE 1284 printer ports vs. USB, Parallel ATA vs. Serial ATA, and SCSI vs. FireWire.