

**CS1050**

**COMPUTER ORGANIZATION & DIGITAL  
DESIGN**

*Eng. DR. Chathura de Silva*

# Computer Organization & Computer Architecture

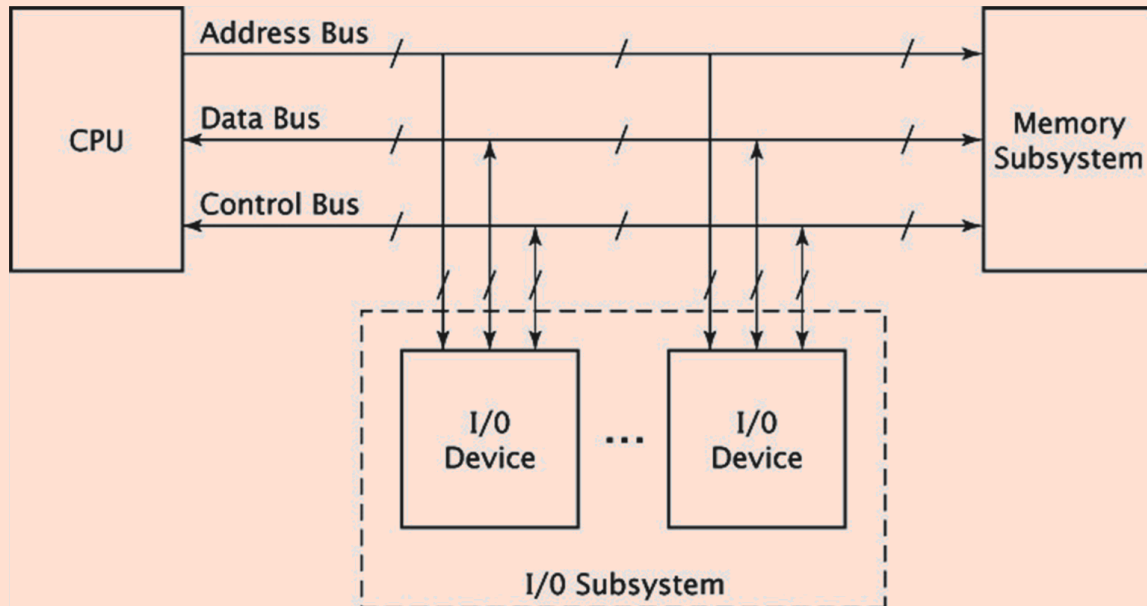
## Computer Organization

- *What matters for the hardware engineers*
- *Mostly about the physical aspects of computer systems.*
  - *circuit design, control signals, memory types.*
  - *Hardware modules, interfacing*
- *What components are required to build a computer and how they are inter-connected?*
- *How does a computer work?*

## Computer Architecture

- *What matters for the software developers*
- *Logical aspects of system as seen by the programmer.*
  - *Instruction sets, instruction formats, data types, addressing modes.*
  - *Instruction timing*
- *How software can perform best in the computer*
- *How do I design a computer system?*

# Recall from CS1033 .....



- *The CPU understands and executes instructions, one at a time*
- *Instructions are stored in the Memory*
- *Called the Von-Neumann architecture*

- *The CPU performs the Instruction cycle*
  - *Fetch sub cycle*
  - *Decode sub cycle*
  - *Operand fetch sub cycle*
  - *Execute sub cycle*
- *CPU communicates with other devices through the system bus*
- *IO devices interface external world with the CPU*
- *All operations of the CPU (and hence other devices) are sync'd based on the system clock*
  - *Often these activities are expressed in the form of a “timing diagram”*

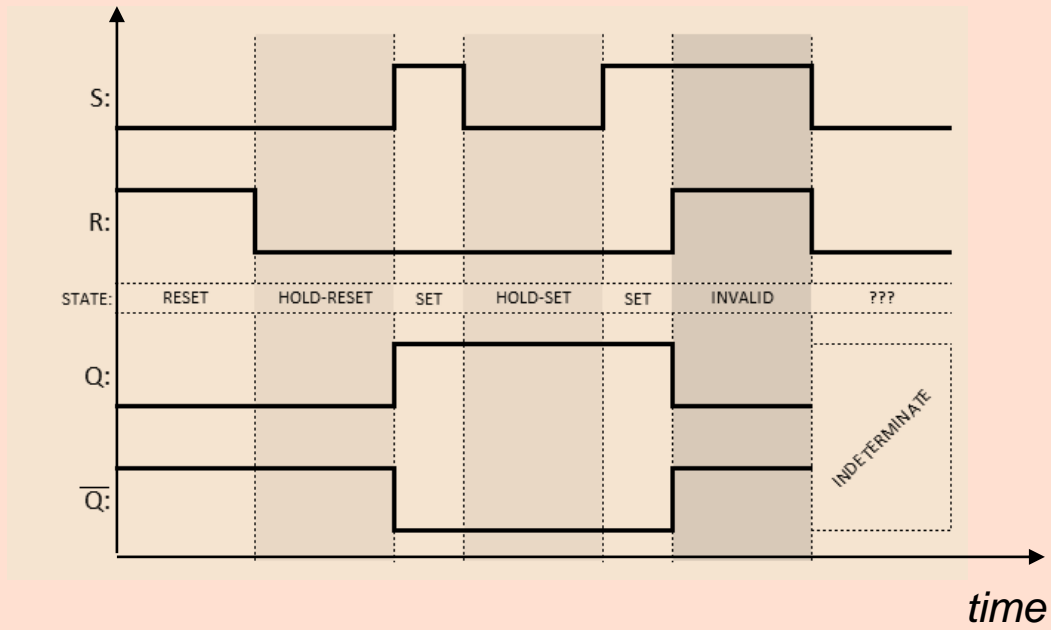
# How CPU Works: in a nutshell



# Timing diagrams

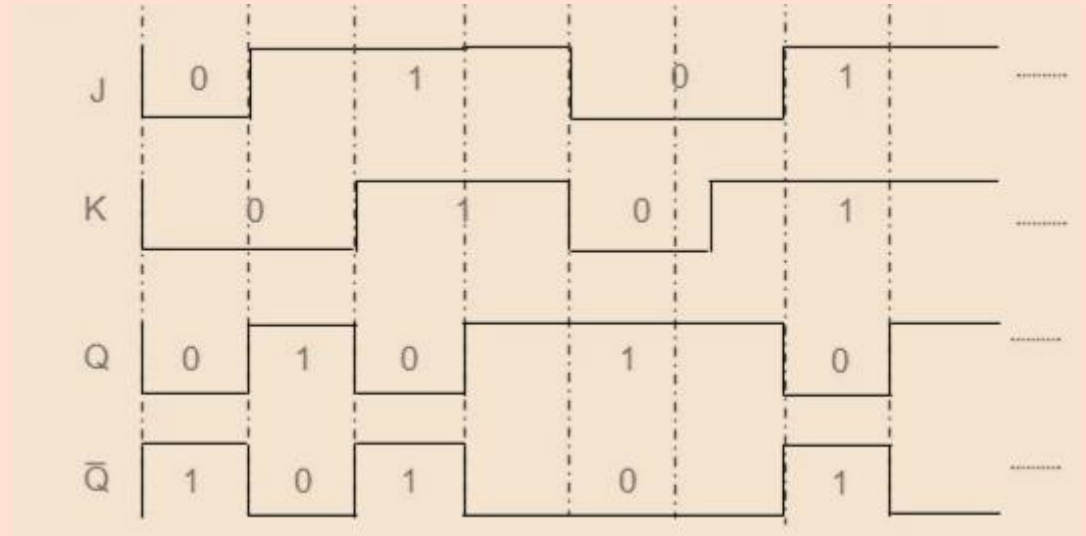
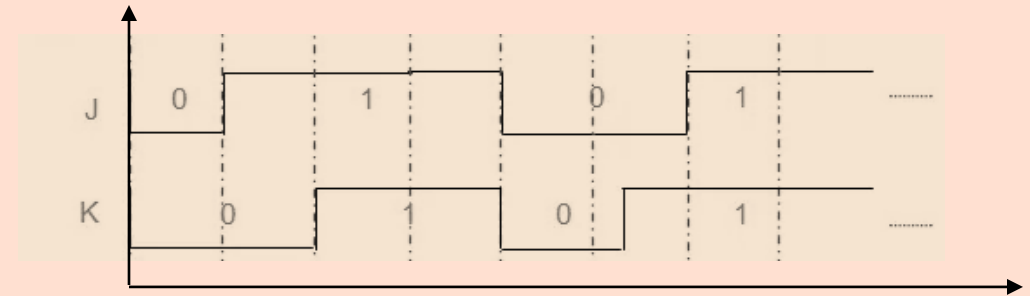
- A timing diagram is a graphical representation of how different logic signals change relatively to each other in time.*

Example: A SR flipflop



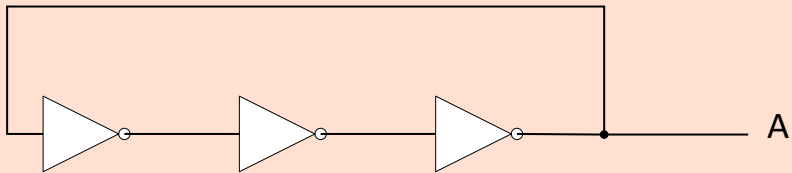
## Exercise

- Complete the timing diagram of the JK flipflop*



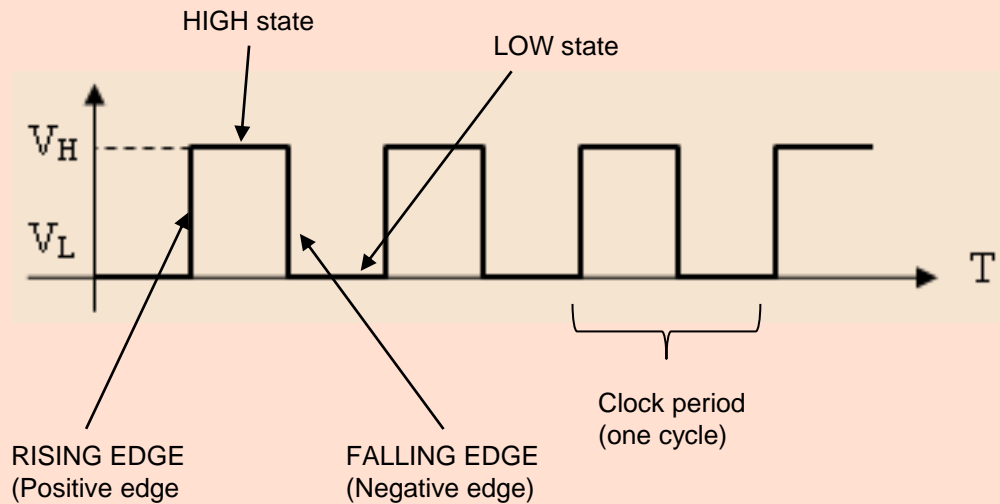
# Timing diagrams: Clock signals

- *What would be the timing diagram at point “A”*



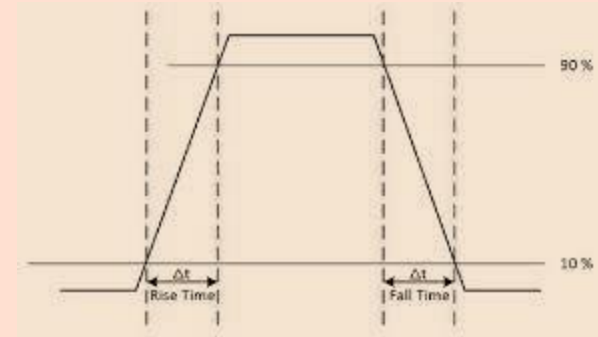
# Clock signals

- A clock signal is a periodic waveform that is used to synchronize state changes of other signals
  - Activities are sync'd to different parts (states) of the clock signal

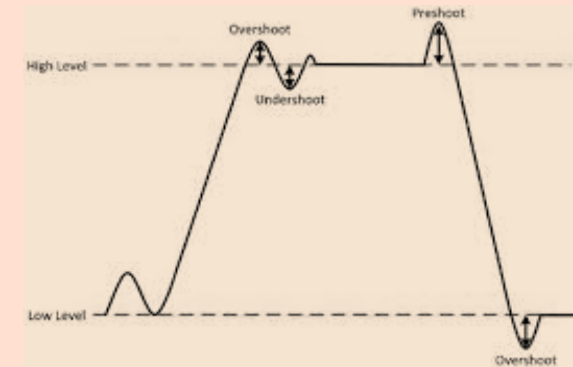


- **Rise-time and fall-time**

- In practical situations, electrical properties like capacitance, prevents signals from changing their state instantaneously

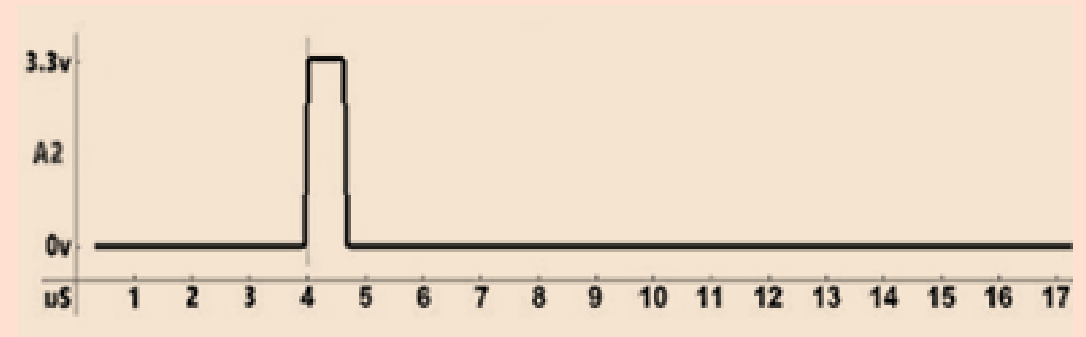
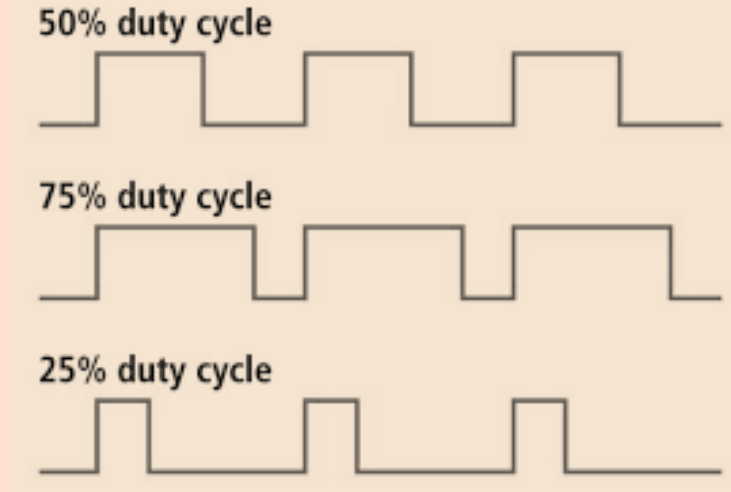


Affects the maximum clock rate



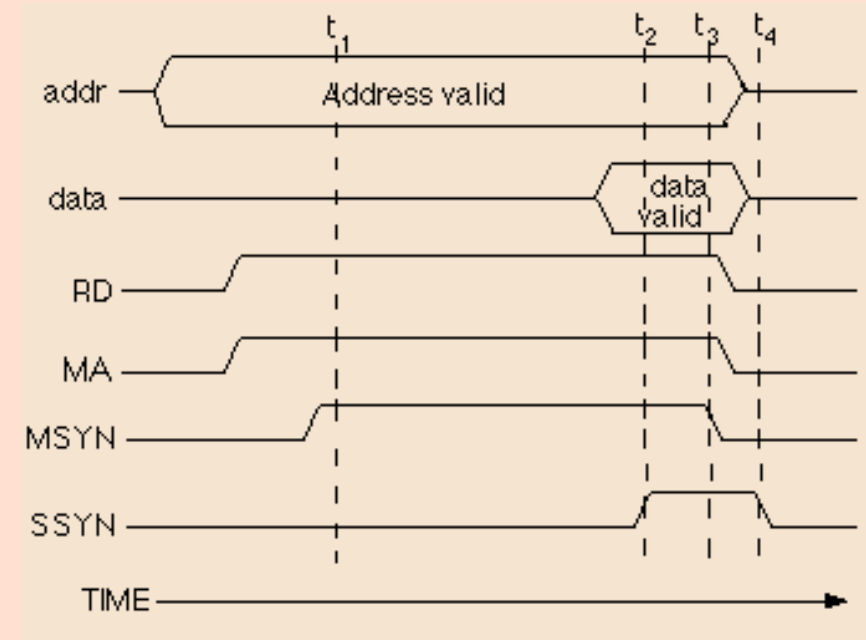
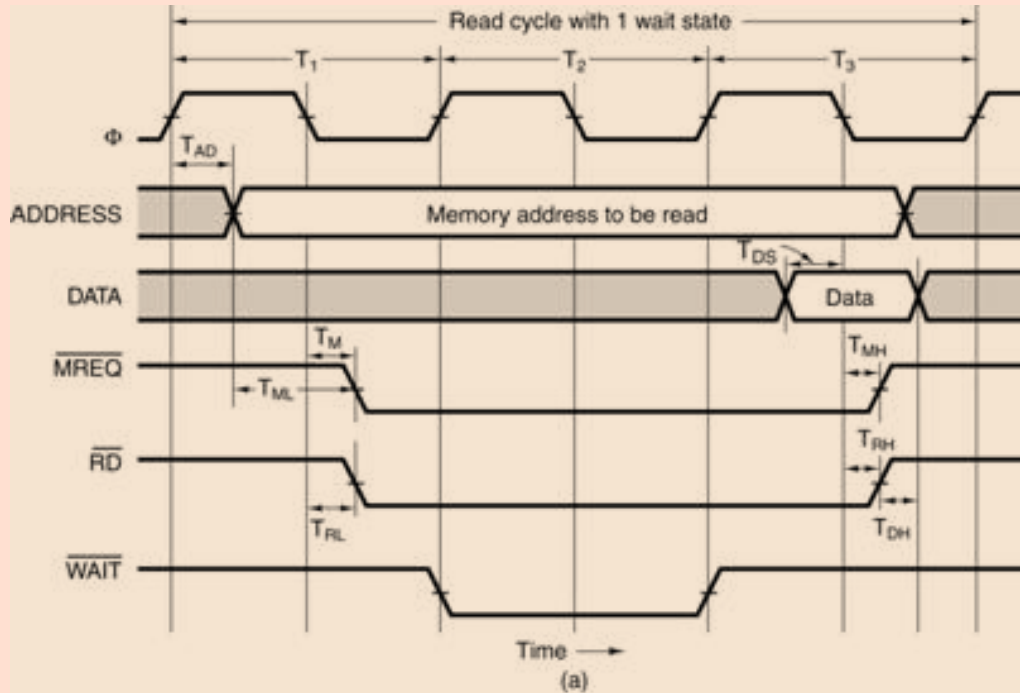
# Clock, pulse signals and the Duty Cycle

- *A clock signal changes periodically between the High-state and the LOW state*
  - *The ratio between time spent in HIGH state compared to the period is called the Duty Cycle*
  - *A 50% duty cycle represents a symmetrical clock signal*
- *A pulse usually remains in one state (idle state), moves briefly to the other state and return's back to the idle state*
  - *Active-high pulse: signal briefly moves to high-state from idle low-state*
  - *Active-low pulse: signal briefly moves to low-state from idle high-state*





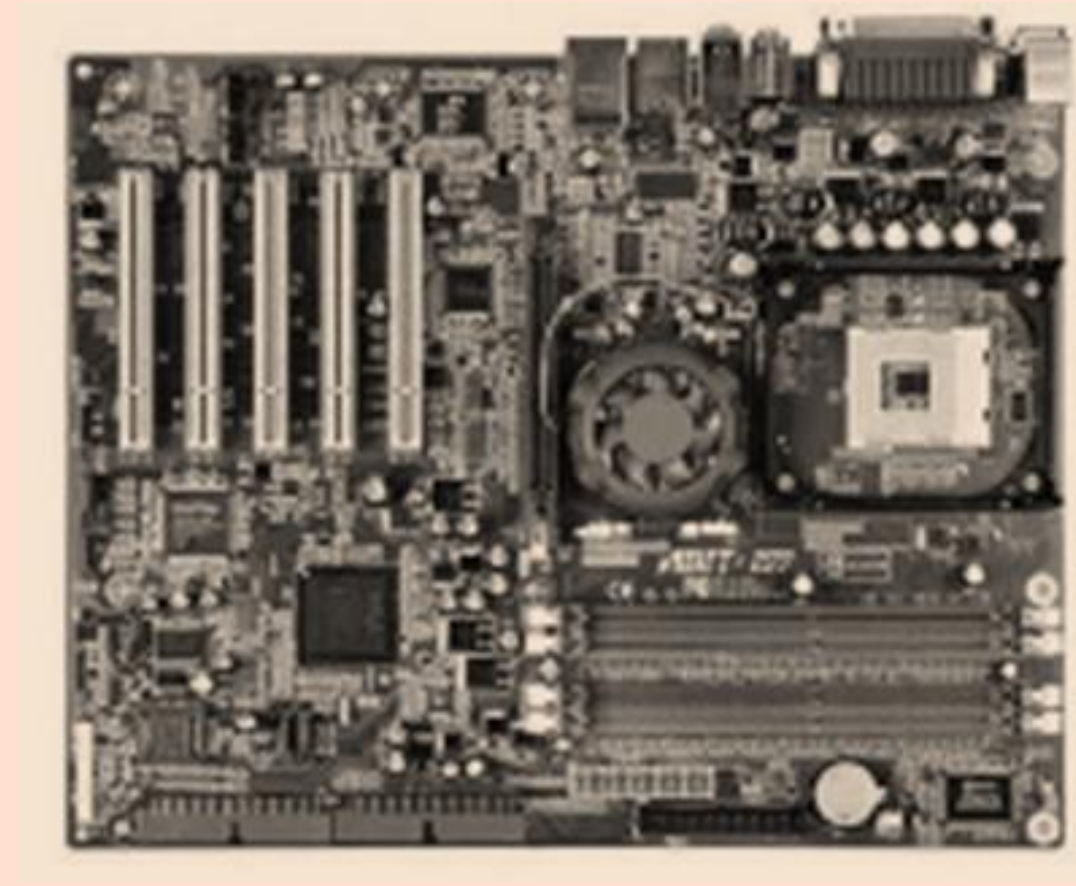
# Reading timing diagrams



- A CPU timing diagram specifies how different signals state change against each other during a task
- Generally, these changes are sync'd to the clock signal

# The system bus

- *The CPU often need to send and receive various types of values (data / instructions) with memory and IO devices*
- *The system bus is a common path-way that interconnects these devices with the CPU*
  - *Contains a few signals transferring data and control information*
- *All most all CPU operations depends on the bus. Hence it is considered a vital component in the computer*
- *Generally, the bus operates under the control of the CPU.*
  - *Hence the CPU is referred to as the bus-master of the system*
  - *Other devices that use the bus are called bus-slaves*
- *A typical computer system may have multiple busses with different characteristics*



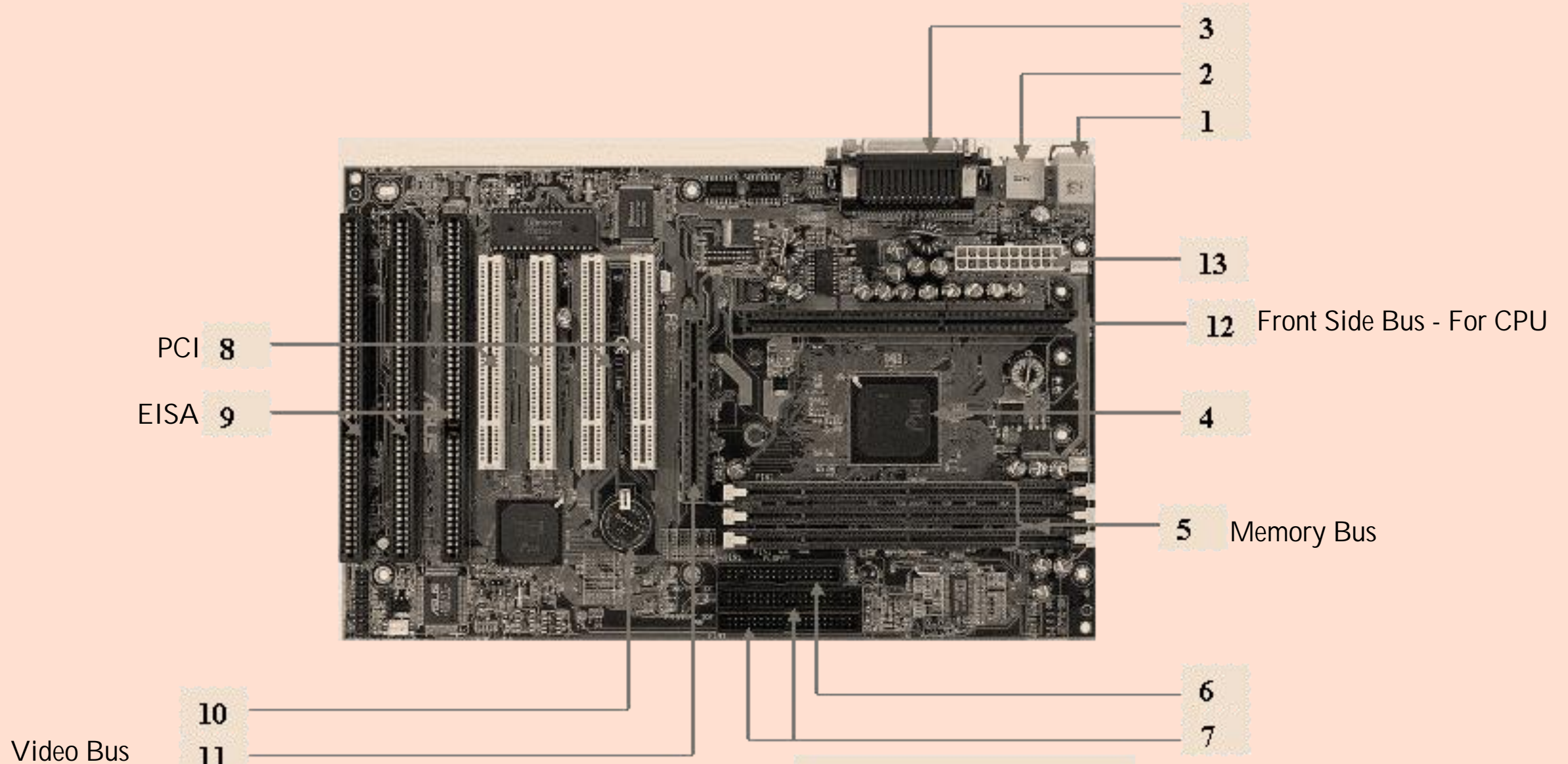
# Bus characteristics

- *The bus needs to send and receive*
  - **Addresses** that uniquely identify the device at the other end
  - **Data** to be transferred
  - **Control signals** that ensure correct data transfer
- *Address is usually unidirectional (from master to slaves). Other signals are bidirectional*
- *Some busses send these values serially using a few communication lines. Others may use several parallel lines for each category of signals*
- *Early computers used a single bus to connect with all components. Modern computers use multiple busses, based on characteristics of each device type*

# Bus characteristics

- *The bus is characterized by*
  - *Electrical properties*
    - *Voltages, currents of communication lines*
  - *Mechanical properties*
    - *Dimensions and properties of connectors*
  - *Address space*
    - *How many devices can be connected, addresses*
  - *Data width (data space)*
    - *How many bits of data transferred at a time*
  - *Clock frequency*
    - *How many transfers per second*
  - *Protocols*
    - *Timing of different signals to ensure correct bus operations*
- *Common types of busses*
  - *Industrial System Bus (ISA)*
    - *(16-bit address, 8-bit data, 8Mhz)*
  - *Extended Industrial Bus (EISA)*
    - *(16-bit address, 16-bit data, 8Mhz)*
  - *Peripheral Component Interconnect (PCI)*
    - *(32-bit address, 32-bit data, 66/133Mhz)*
  - *Universal Serial Bus (USB)*
  - *Control Area Network (CAN) bus*
  - *FireWire bus*
  - *I2C (Inter IC communication) bus*

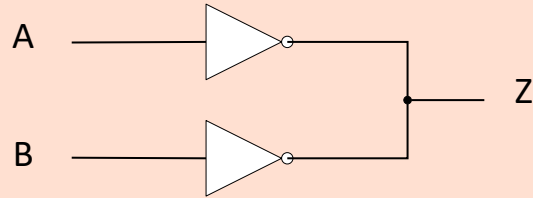
# Multiple busses on a motherboard





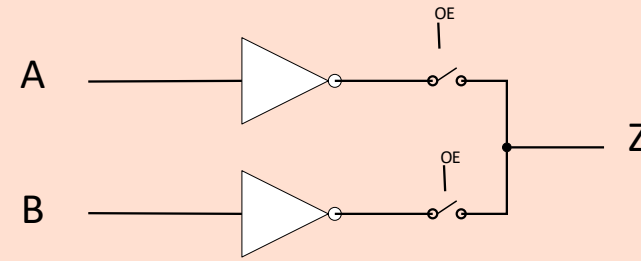
# Busses require logic with tri-state outputs

- Write down the truth table for the (A,B) inputs and the Z output*

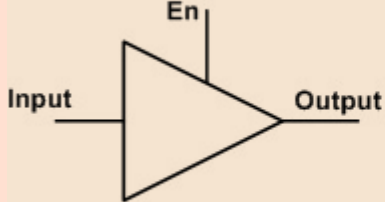


A	B	Z
0	0	
1	1	
1	0	
0	1	

- Gates with tri-state outputs*



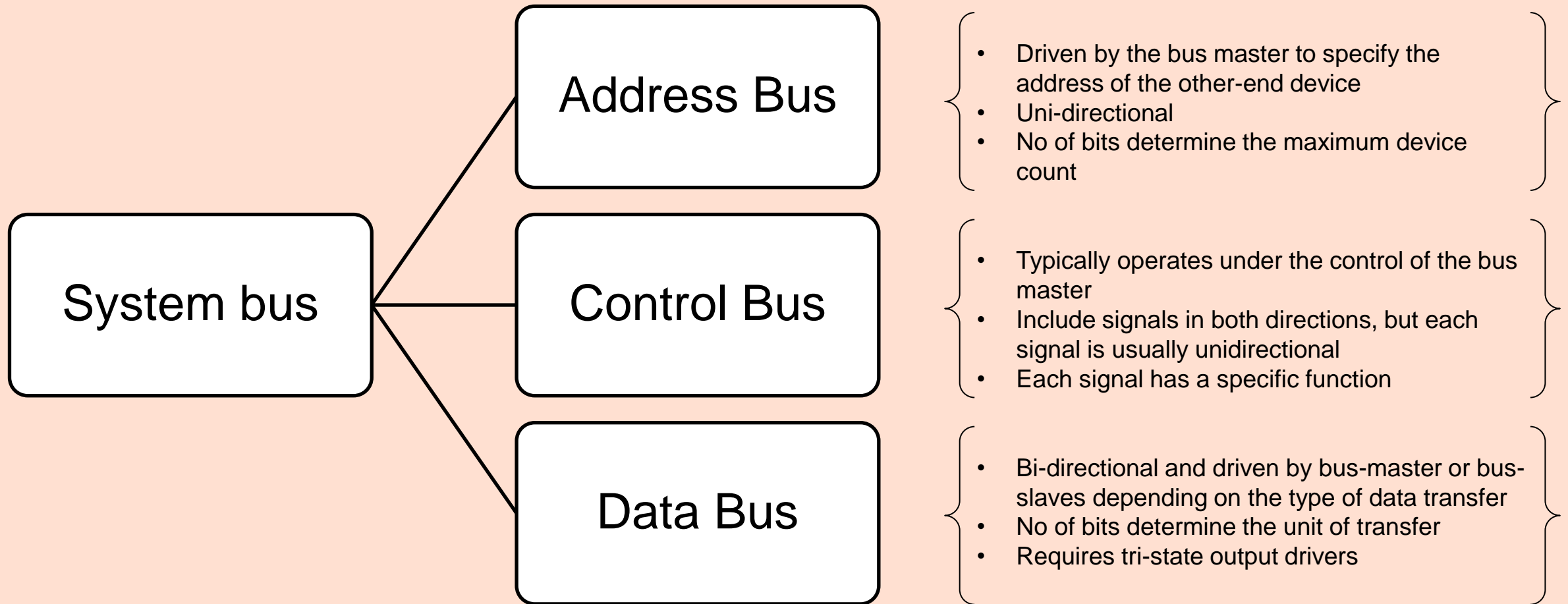
Symbol



Truth Table

En	Input	Output
0	X	Hi-Z
1	0	0
1	1	1

# 3-Bus architecture



# Z80 Memory Read/Write

- *Z80 uses 3-bus architecture*
  - *Address bus: 16-bit*
  - *Data bus: 8-bits*

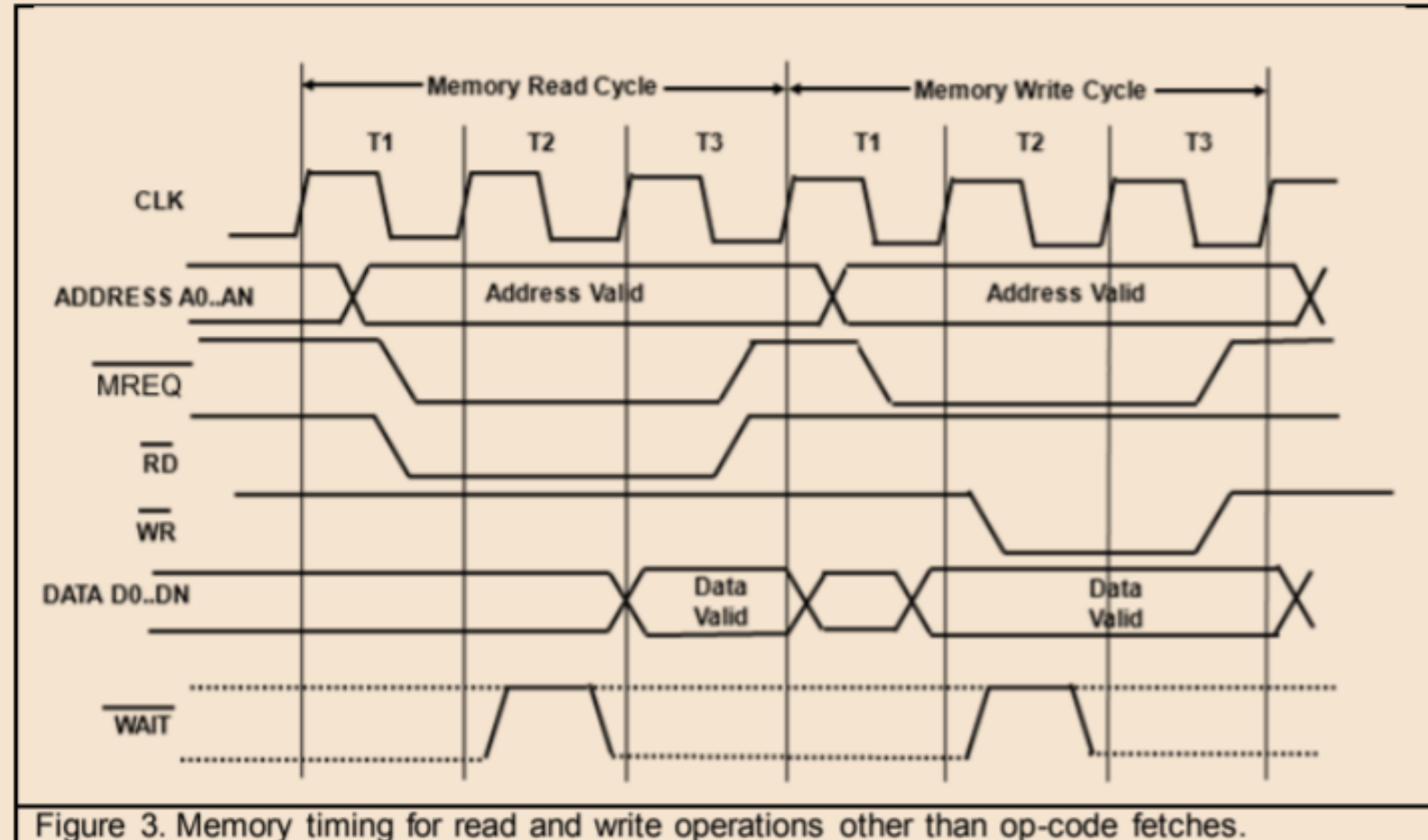
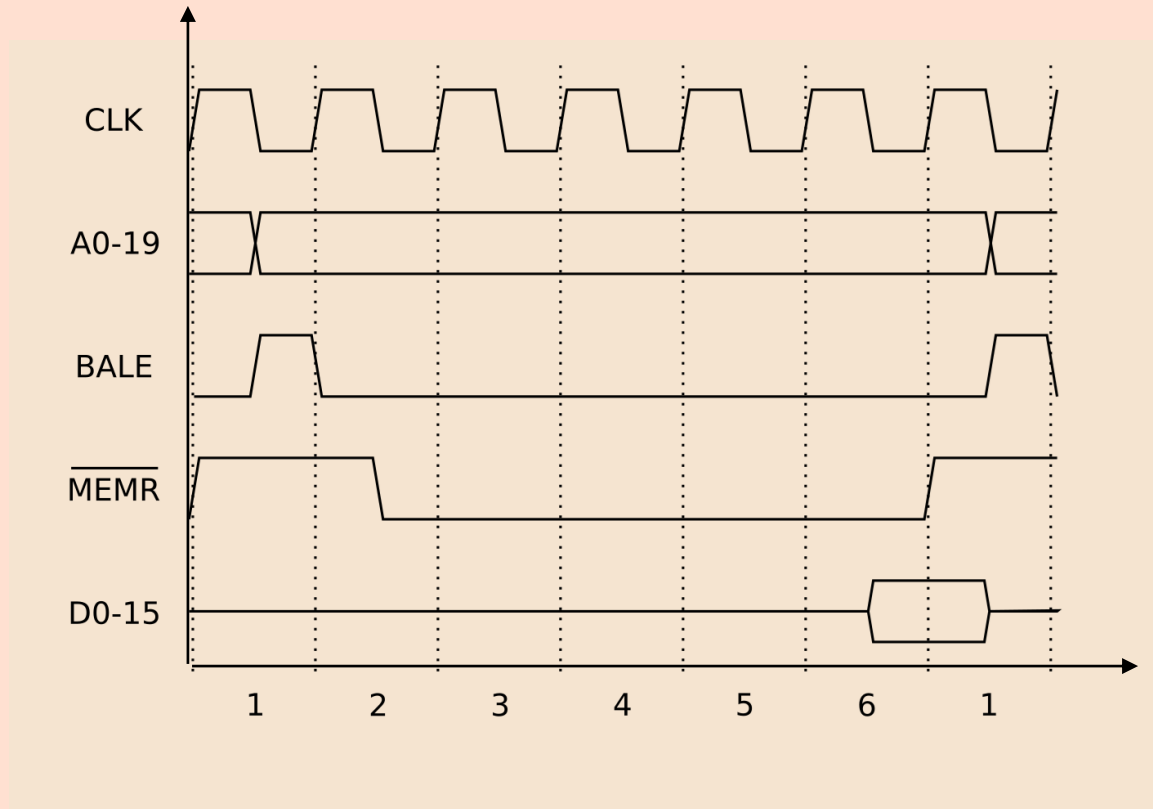


Figure 3. Memory timing for read and write operations other than op-code fetches.

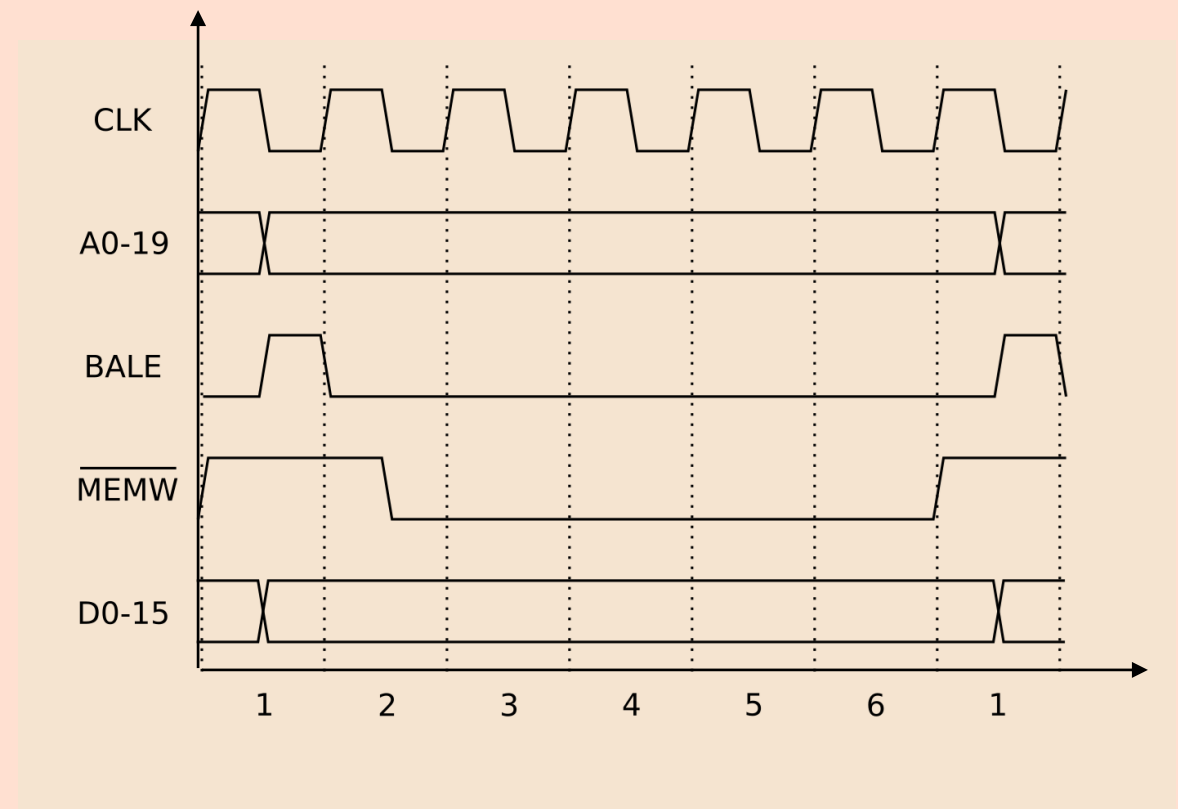


# ISA BUS Read/Write cycles

- *Memory read*

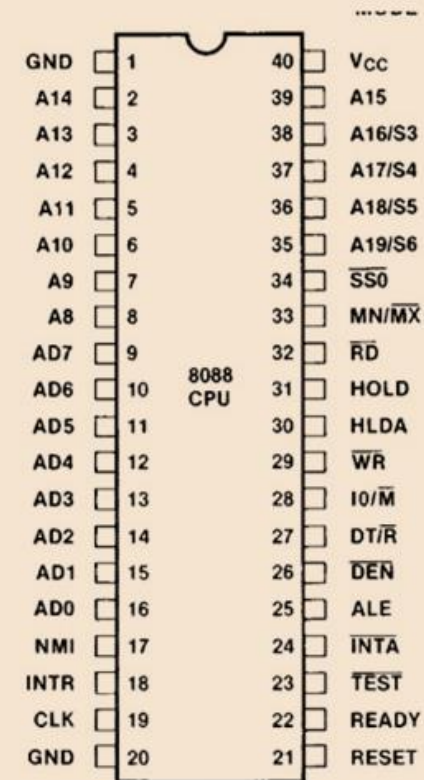
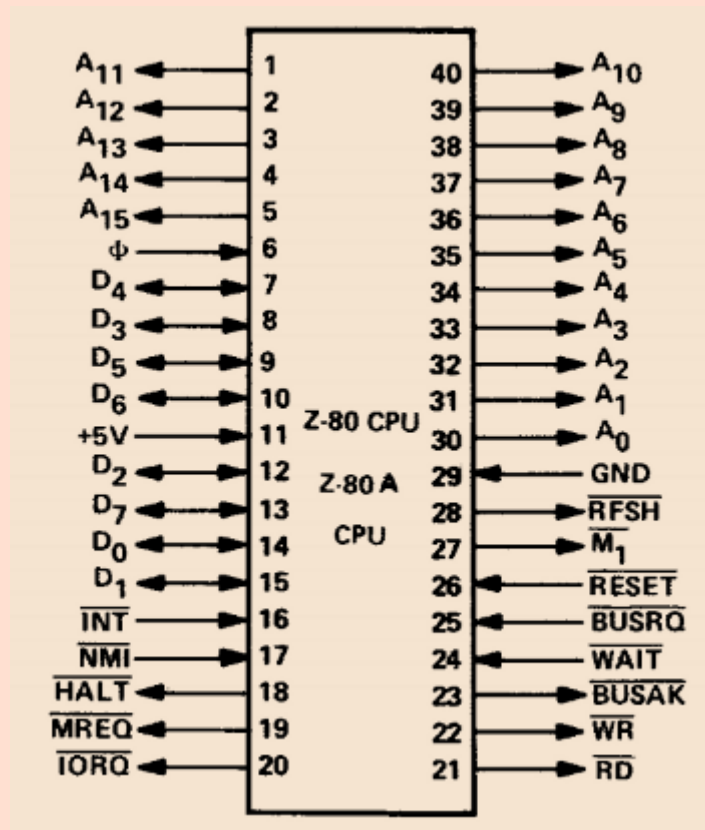


- *Memory write*



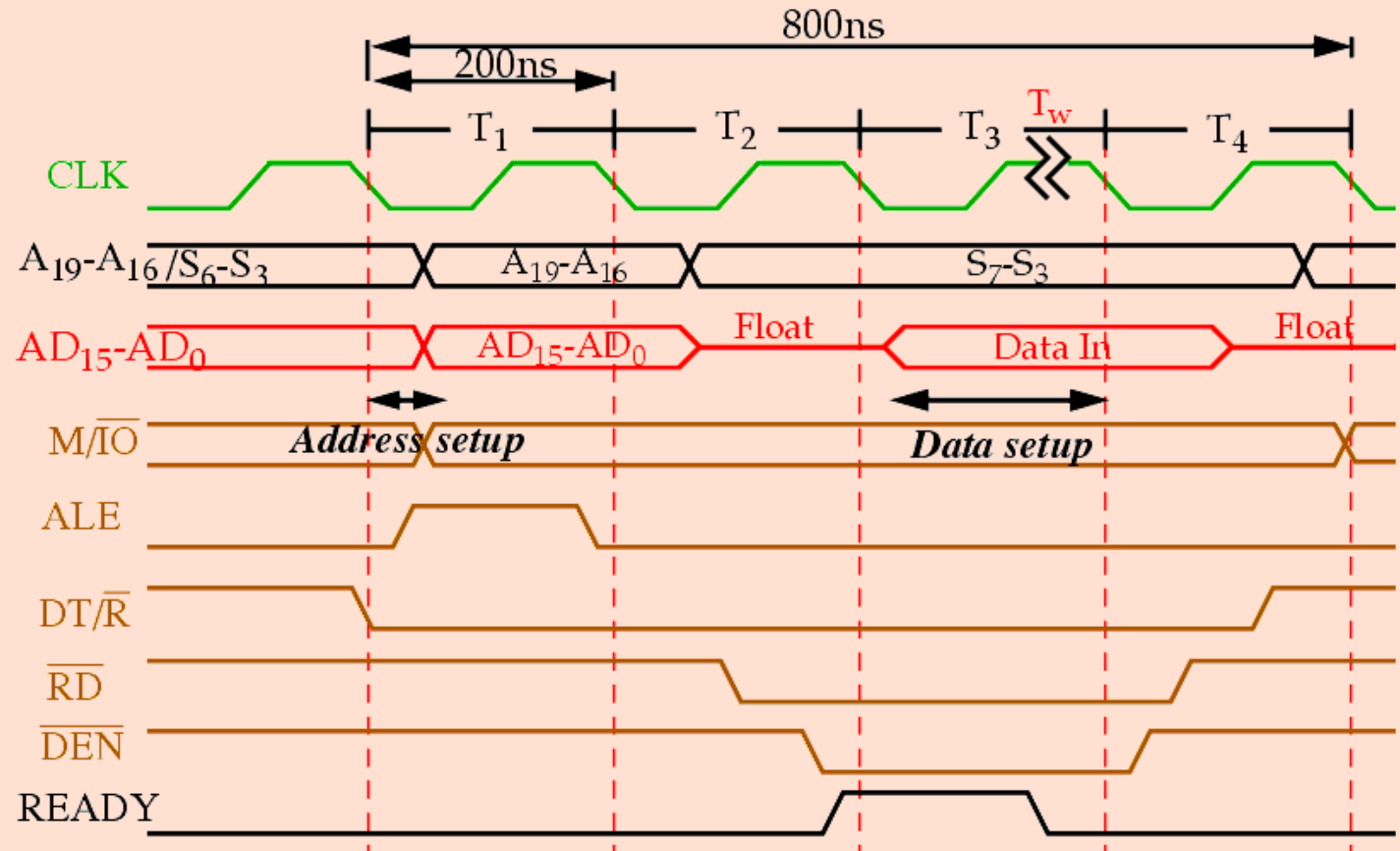
# Multiplexed busses

- Z80 has separate pins for Address, Data and Control busses*
- Intel 8088 uses part of its address bus to send and receive data*



# 8088 bus demultiplexing

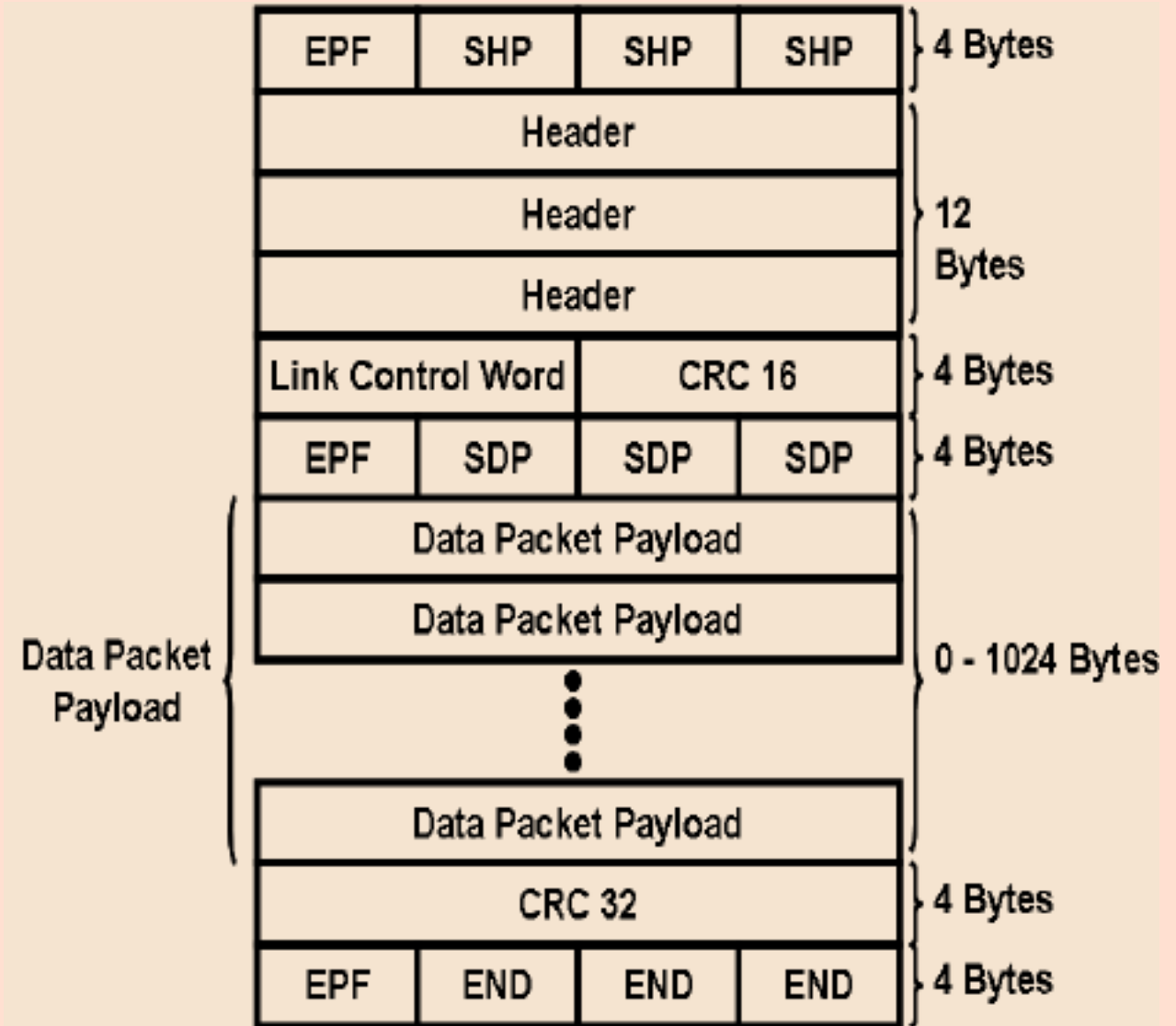
- Data bus is multiplexed onto part of the address bus
- A separate signal ALE (Address Latch Enable) differentiate between address and data within the multiplexed portion
- External device is expected to demultiplex by latching the address



Bus Timing for a Read Operation

# Serial busses (USB/CAN/Firewire/I2C)

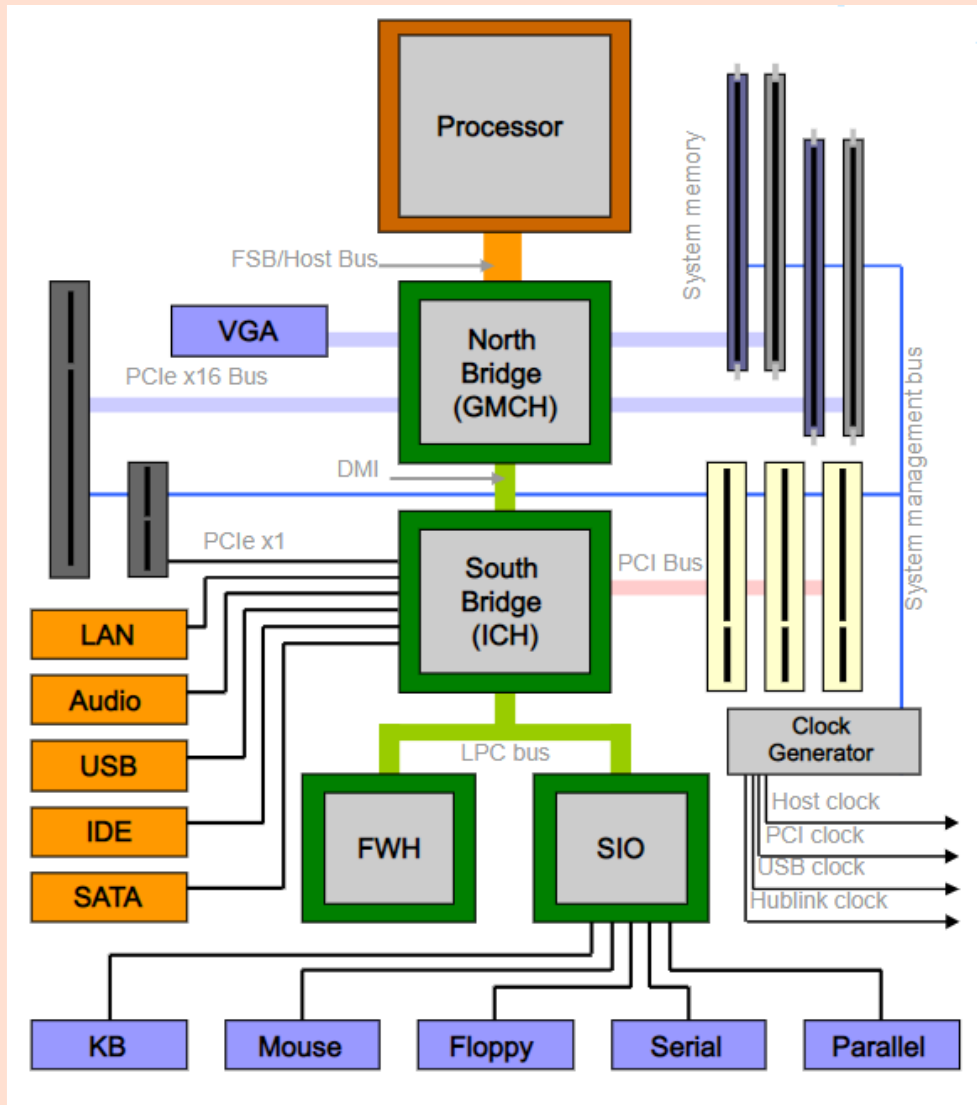
- *Transfer all information (Addresses, Data and control) using the same set of lines*
- *Hence data is transferred as a “frame” in a burst mode*
  - *A frame is a data structure that includes, address fields, data fields and control fields*
- *Transmitting and receiving devices serialize/deserialize these frames into appropriate structures before they are used*
- *The device will not consume/transmit the package unless the address is correct*



# Bus limitations

- *The system bus is a shared resource. Only one pair of devices can use it at any given time*
- *All devices must communicate at the same speed, using the same protocol*
- *Bus must always be under control of one bus-master*
- *Parallel busses require a larger number of lines (multiplexing reduces throughput)*
- *Serial busses use less lines but add complexity and have lower capacity*

# Multi-bus systems

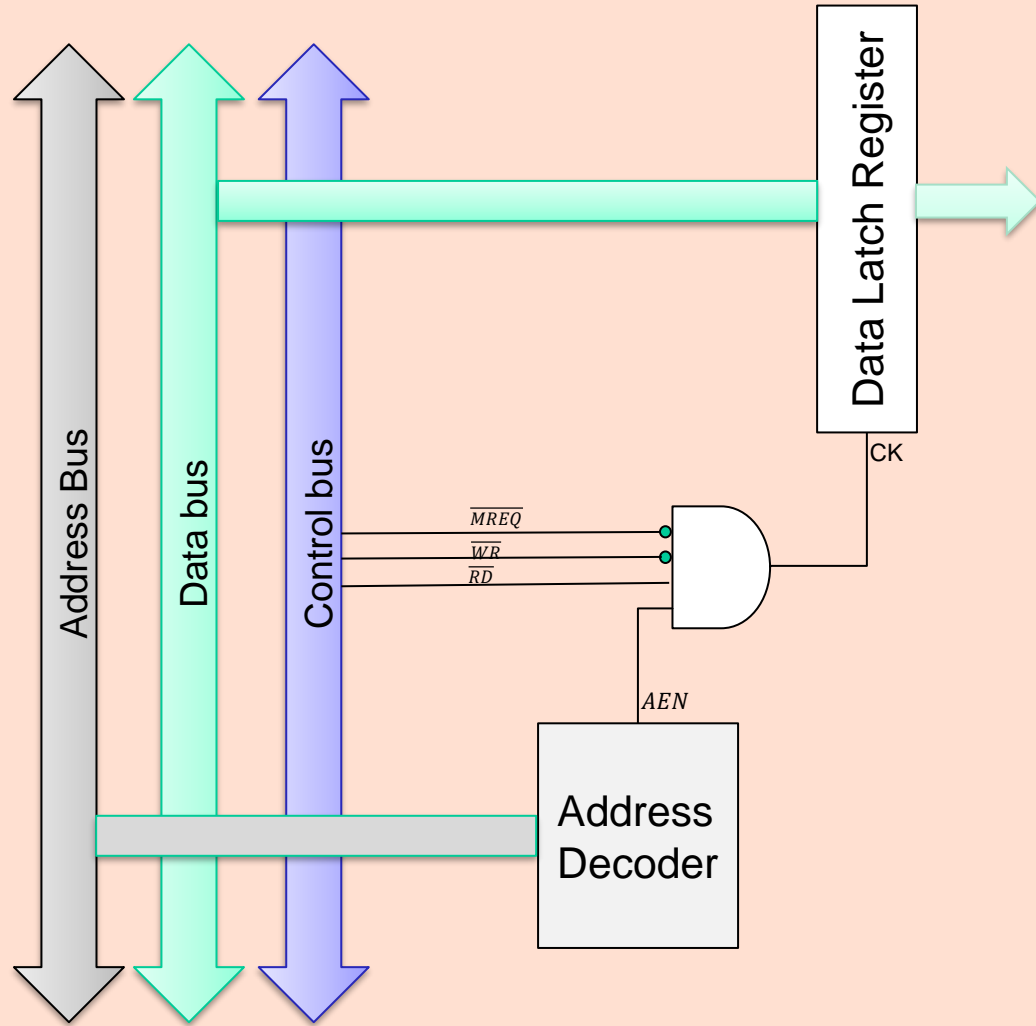


- *Modern computers need interconnections with different types of devices with different characteristics*
- *A single bus is not efficient to interconnect all of them*
- *Hence these systems use multiple busses, each designed to meet requirements of a category of devices.*
- *Special devices called Bridges inter-link these busses*
  - *Matches multiple characteristics across busses*
  - *Uses buffering to match speeds*

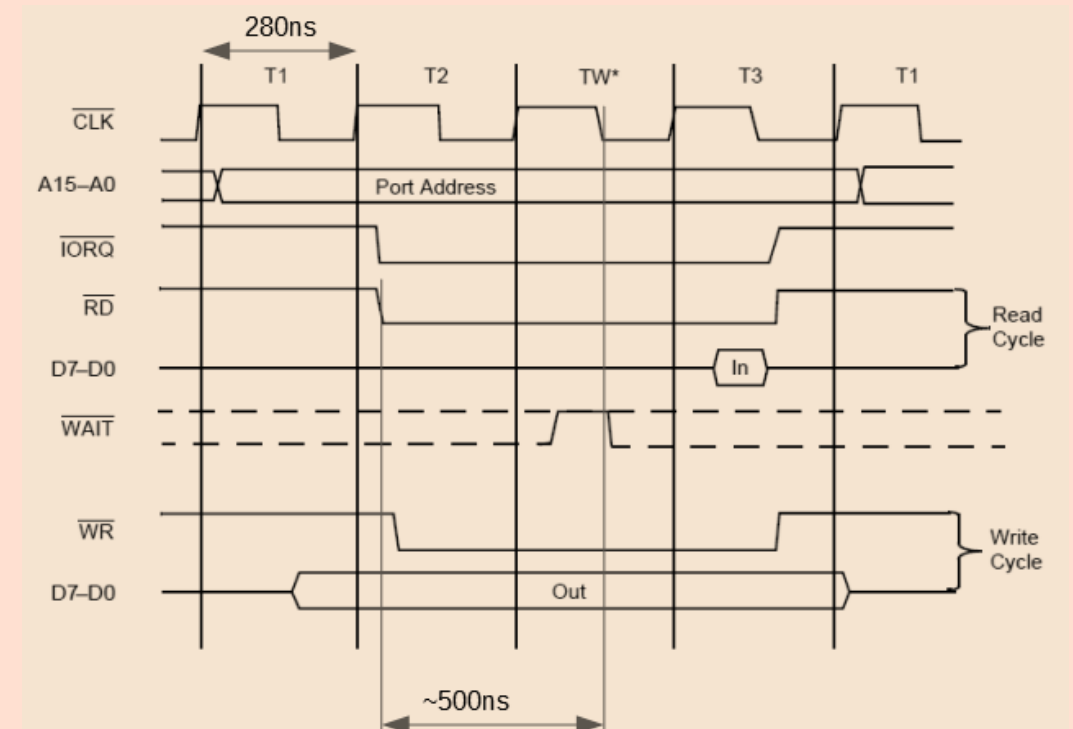
# ISA bus interface



# Bus interfacing: Receiving data

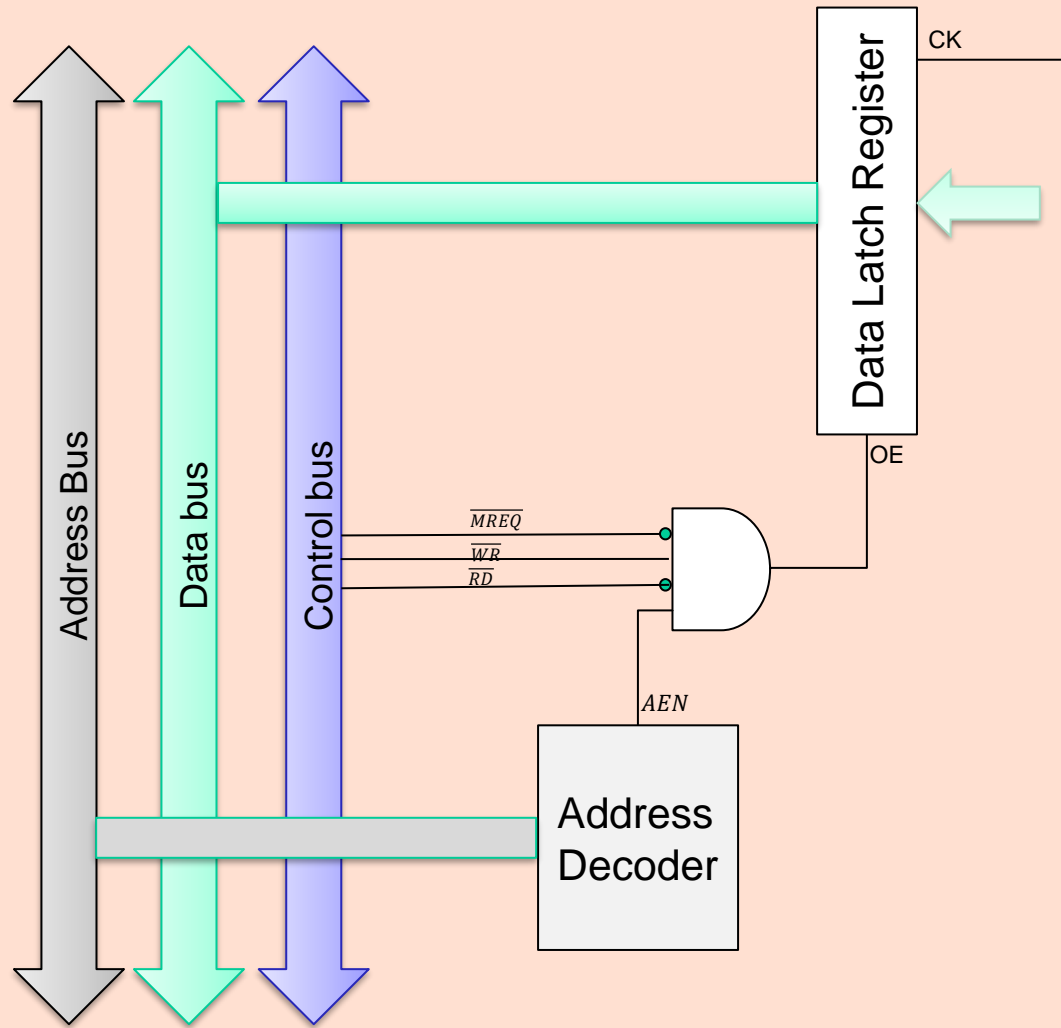


- *Z80 Write Cycle*

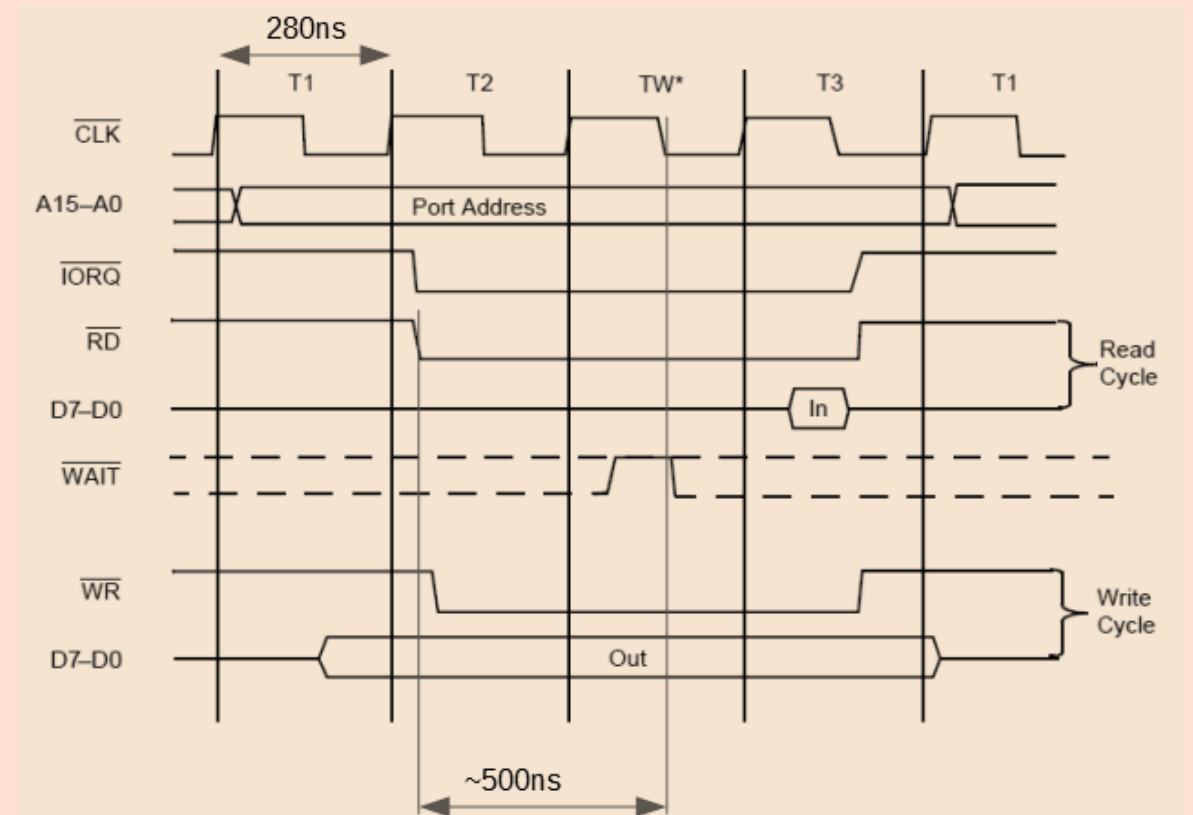




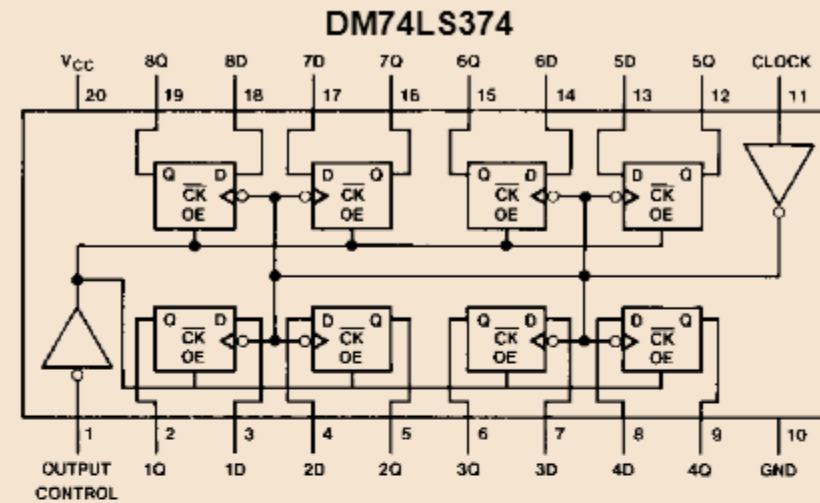
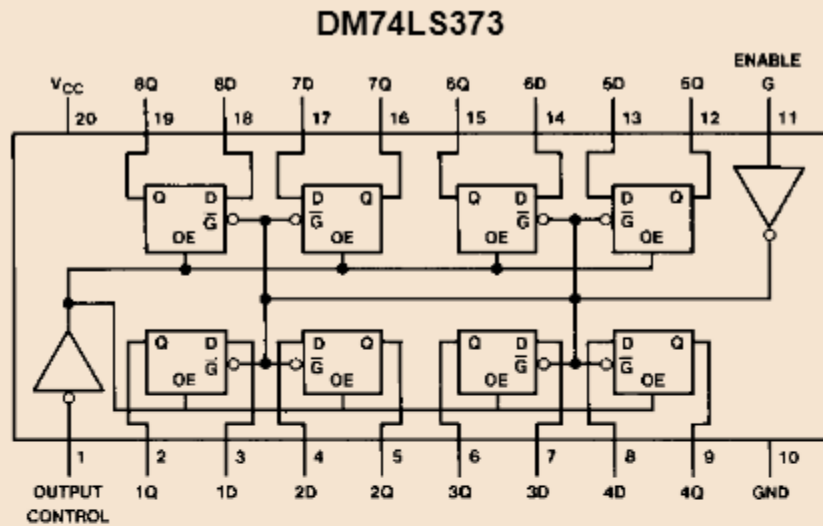
# Bus interfacing: Sending data



- Z80 Read Cycle*



# LSI/MSI Components



## Function Tables

**DM74LS373**

Output Control	Enable G	D	Output
L	H	H	H
L	H	L	L
L	L	X	$Q_0$
H	X	X	Z

**DM74LS374**

Output Control	Clock	D	Output
L	$\uparrow$	H	H
L	$\uparrow$	L	L
L	L	X	$Q_0$
H	X	X	Z

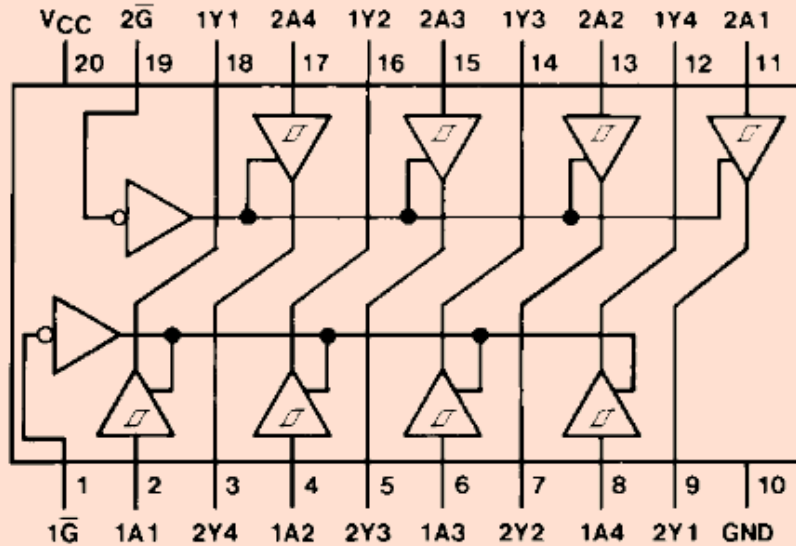
H = HIGH Level (Steady State)    L = LOW Level (Steady State)    X = Don't Care    Z = High Impedance State

$\uparrow$  = Transition from LOW-to-HIGH level

$Q_0$  = The level of the output before steady-state input conditions were established.

# LSI/MSI components

Connection Diagram

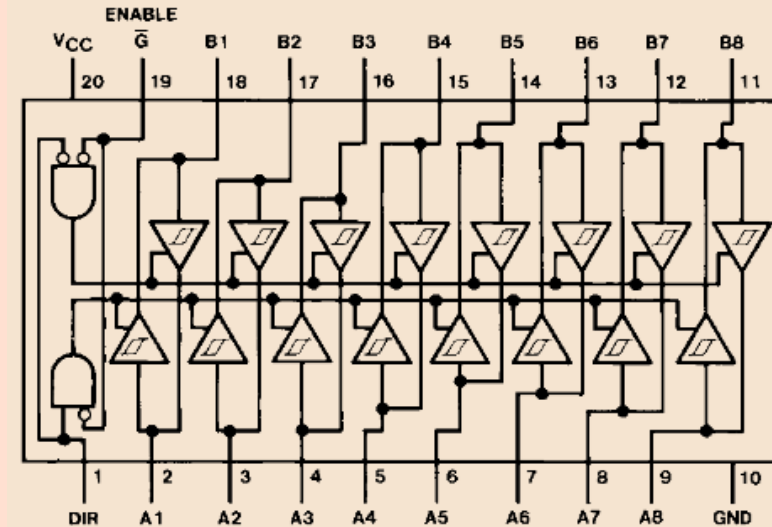


Function Table

Inputs		Output
$\overline{G}$	A	Y
L	L	L
L	H	H
H	X	Z

L = LOW Logic Level  
H = HIGH Logic Level  
X = Either LOW or HIGH Logic Level  
Z = High Impedance

Connection Diagram

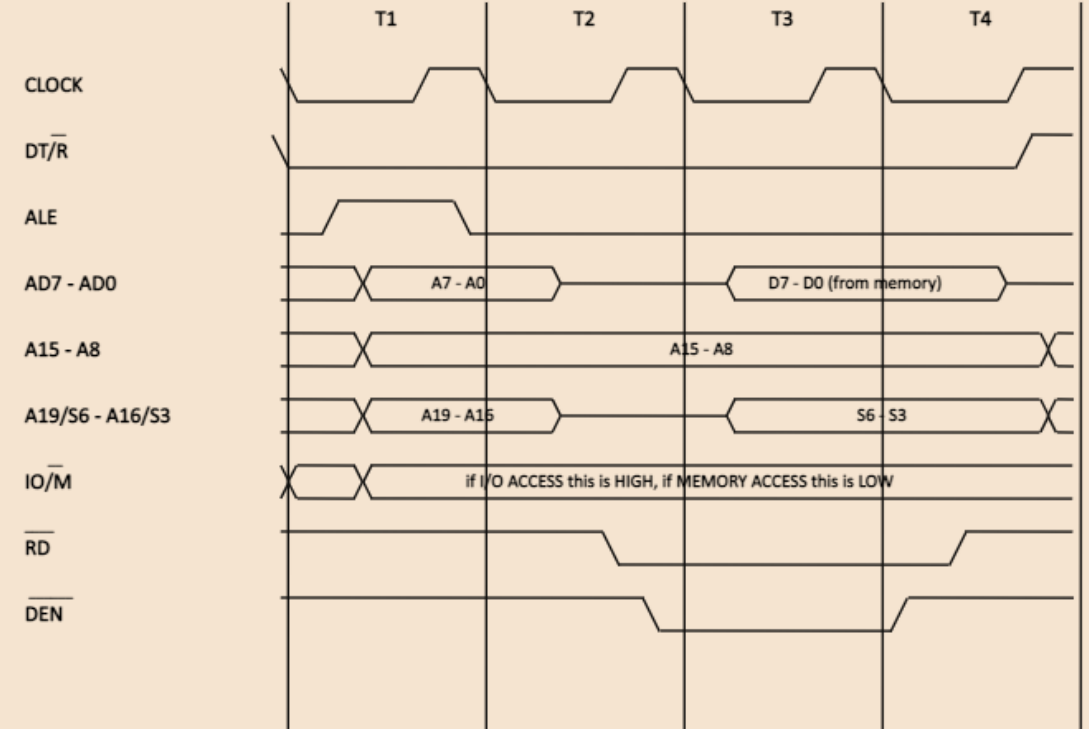
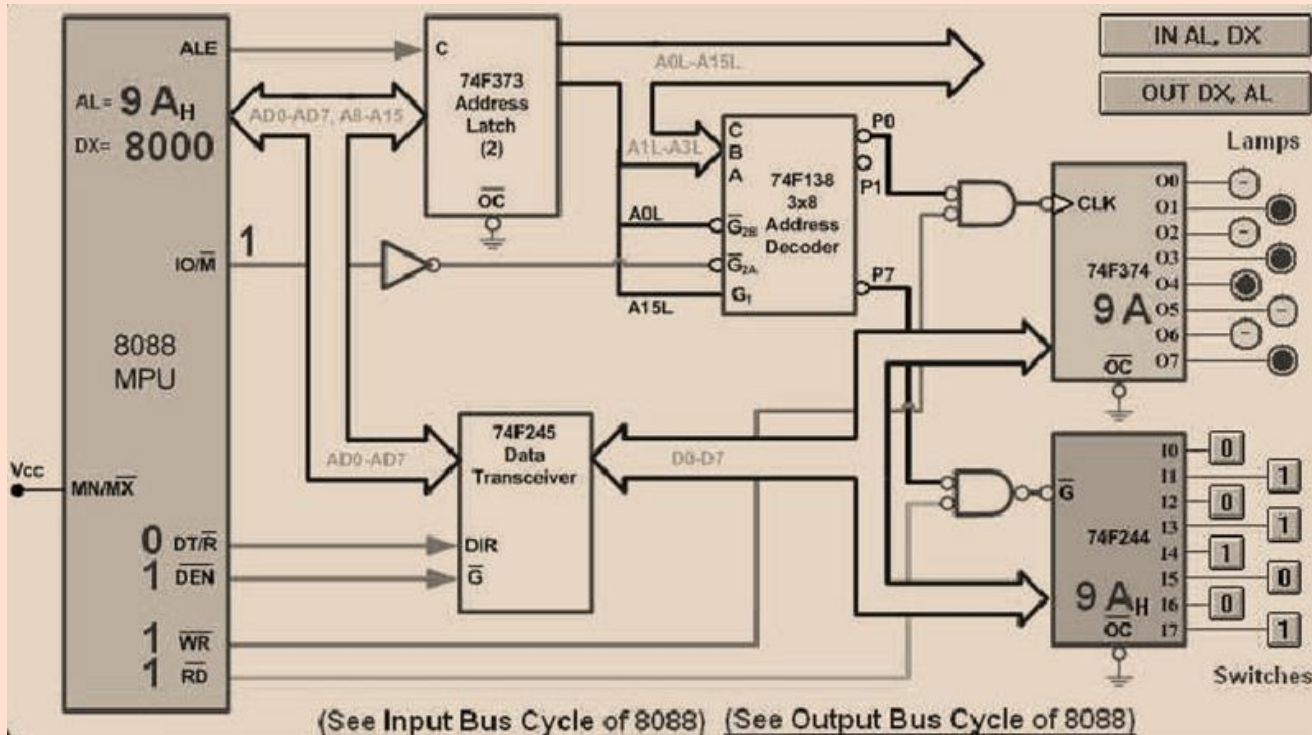


Function Table

Enable $\overline{G}$	Direction Control DIR	Operation
L	L	B Data to A Bus
L	H	A Data to B Bus
H	X	Isolation

H = HIGH Level  
L = LOW Level  
X = Irrelevant

# Multiplexed buses: 8088 processor



# Address decoder

- *Full address decoder*

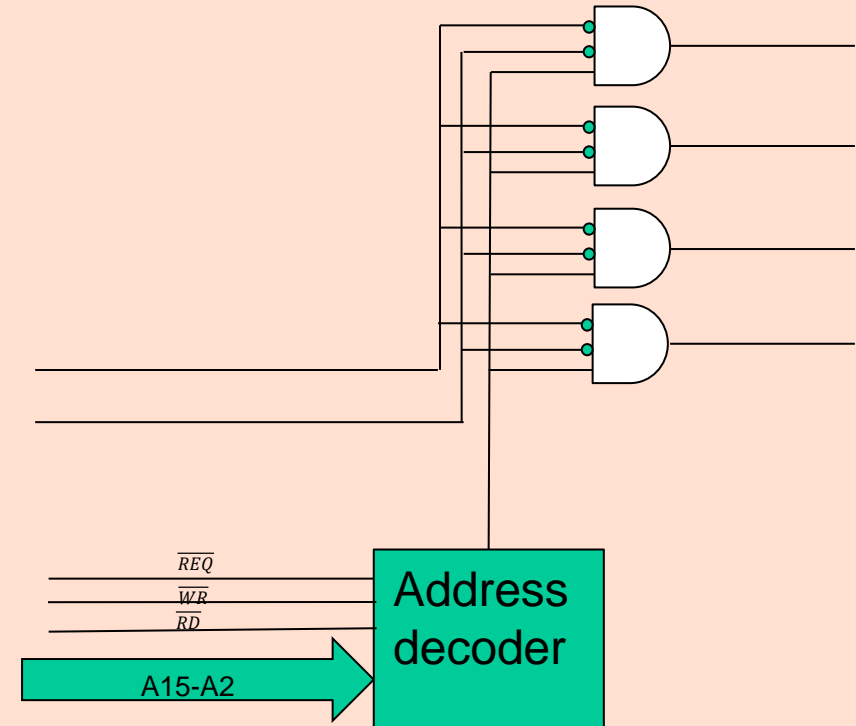
- *Decodes all address bits and map to a unique address in the full space*
- *Typically, in the form of a 2-level MSOP or MPOS format*

- *Partial address decoder*

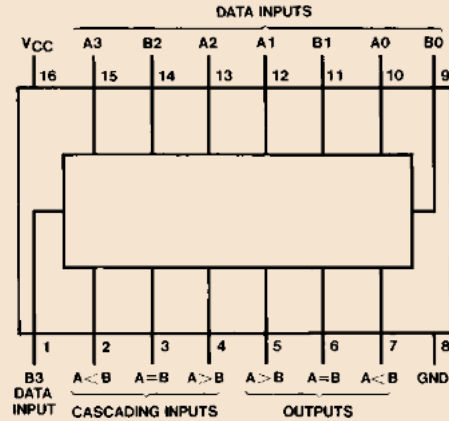
- *Decodes only a few address lines and therefore maps multiple addresses to the same port*
- *Used when only a few ports are needed*

- *Hierarchical address decoding*

- *Typically used when the interface require multiple ports mapped onto an address range*

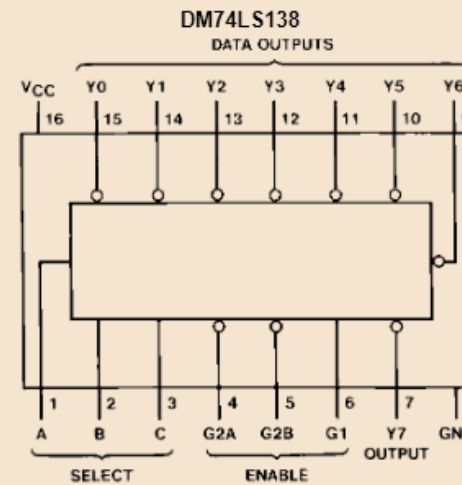


# LSI/MSI components: Decoders & Comparators



Comparing Inputs				Cascading Inputs			Outputs		
A3, B3	A2, B2	A1, B1	A0, B0	A > B	A < B	A = B	A > B	A < B	A = B
A3 > B3	X	X	X	X	X	X	H	L	L
A3 < B3	X	X	X	X	X	X	L	H	L
A3 = B3	A2 > B2	X	X	X	X	X	H	L	L
A3 = B3	A2 < B2	X	X	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 > B1	X	X	X	X	H	L	L
A3 = B3	A2 = B2	A1 < B1	X	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 > B0	X	X	X	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 < B0	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	L	L	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	H	L	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	X	X	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	H	L	L	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	L	H	H	L

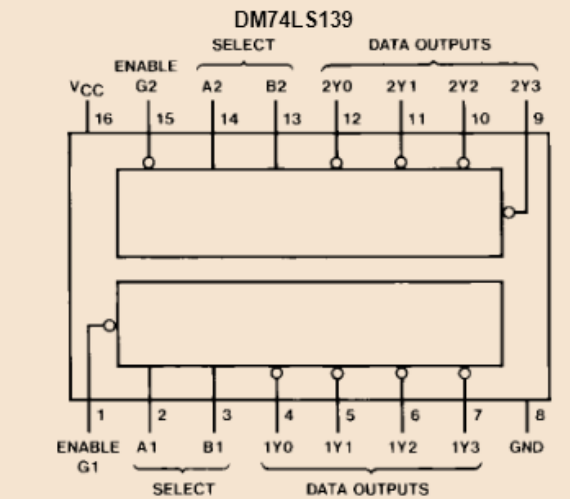
H = HIGH Level, L = LOW Level, X = Don't Care



Function Tables

DM74LS138

Inputs			Outputs									
Enable		Select										
G1	G2 (Note 1)	C B A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7		
X	H	X X X	H	H	H	H	H	H	H	H		
L	X	X X X	H	H	H	H	H	H	H	H		
H	L	L L L	L	H	H	H	H	H	H	H		
H	L	L L H	H	L	H	H	H	H	H	H		
H	L	L H L	H	H	L	H	H	H	H	H		
H	L	L H H	H	H	H	L	H	H	H	H		
H	L	H L L	H	H	H	H	L	H	H	H		
H	L	H L H	H	H	H	H	H	L	H	H		
H	L	H H L	H	H	H	H	H	H	L	H		
H	L	H H H	H	H	H	H	H	H	H	L		



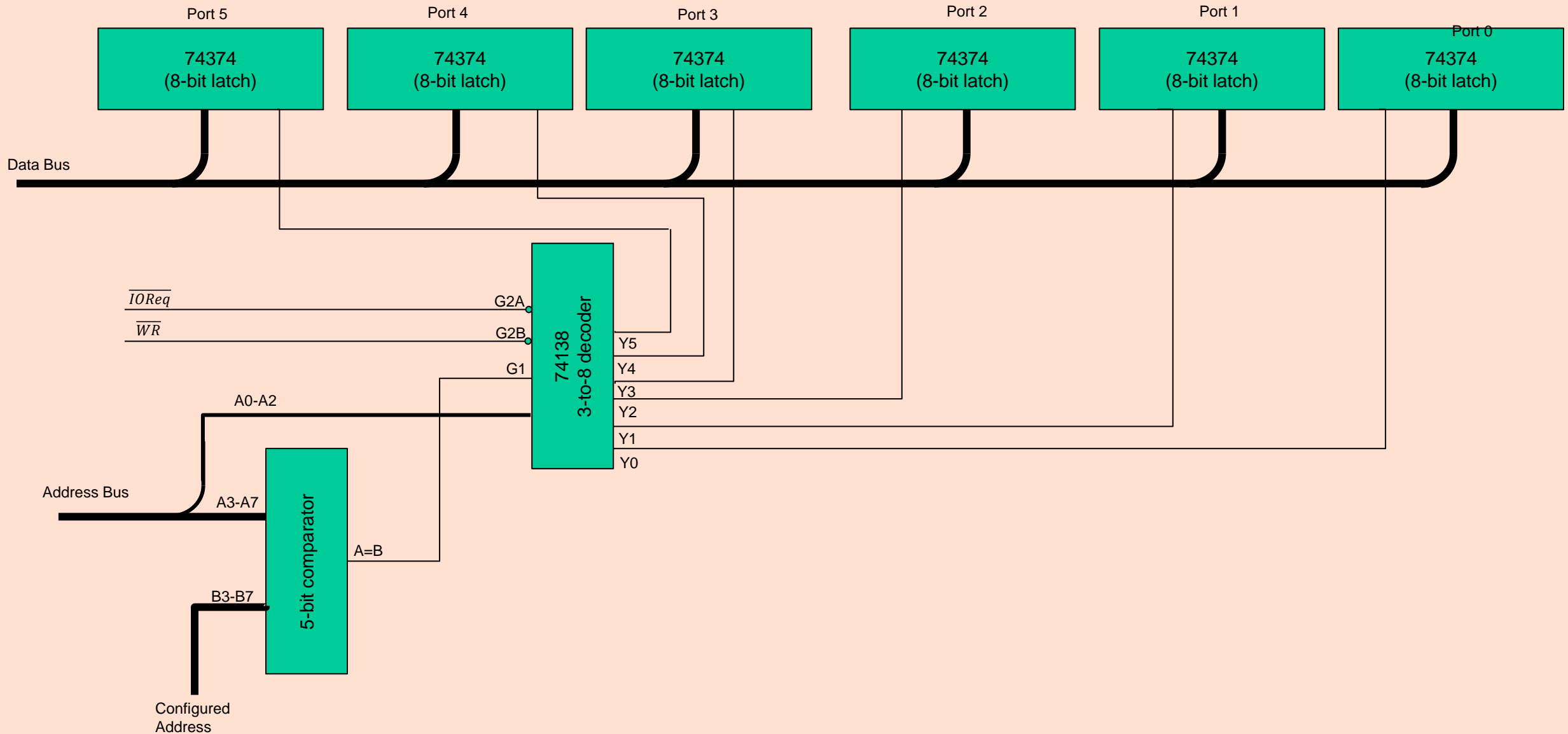
DM74LS139

Inputs			Outputs			
Enable	Select					
G	B	A	Y0	Y1	Y2	Y3
H	X	X	H	H	H	H
L	L	L	L	H	H	H
L	L	H	H	L	H	H
L	H	L	H	H	L	H
L	H	H	H	H	H	L

H = HIGH Level  
L = LOW Level  
X = Don't Care

Note 1: G2 = G2A + G2B

# Design with LSI/MSI components



# Use of integrated devices (Intel 8255)

CS <sup>b</sup>	A1	A0	Sel
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	CRW

Bi directional  
Data Bus  
D7-D0

