

Bus-Design

Bus Design

CS8625 High Performance and Parallel Computing

Dr. Ken Hoganson

Class

Will

Start

Momentarily...

Bus construction alternatives

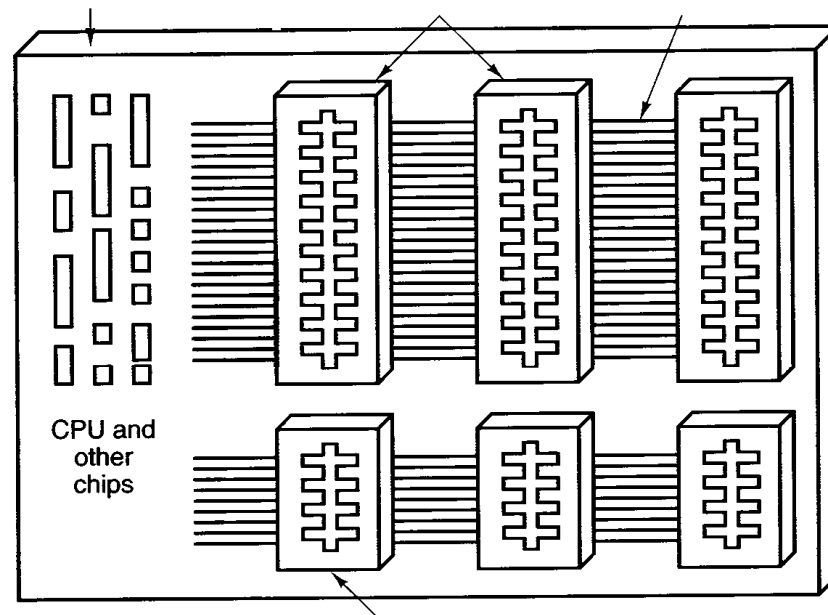
Bus control issues

Bus arbitration (deciding who gets access to the bus)

Bus is a set of parallel wires, that connect computer system components

We have seen that the address bus size determines the number of addressable bytes (or words)

Data bus size is an indication of computing power



How to connect multiple inputs to a single shared output (bus) line?

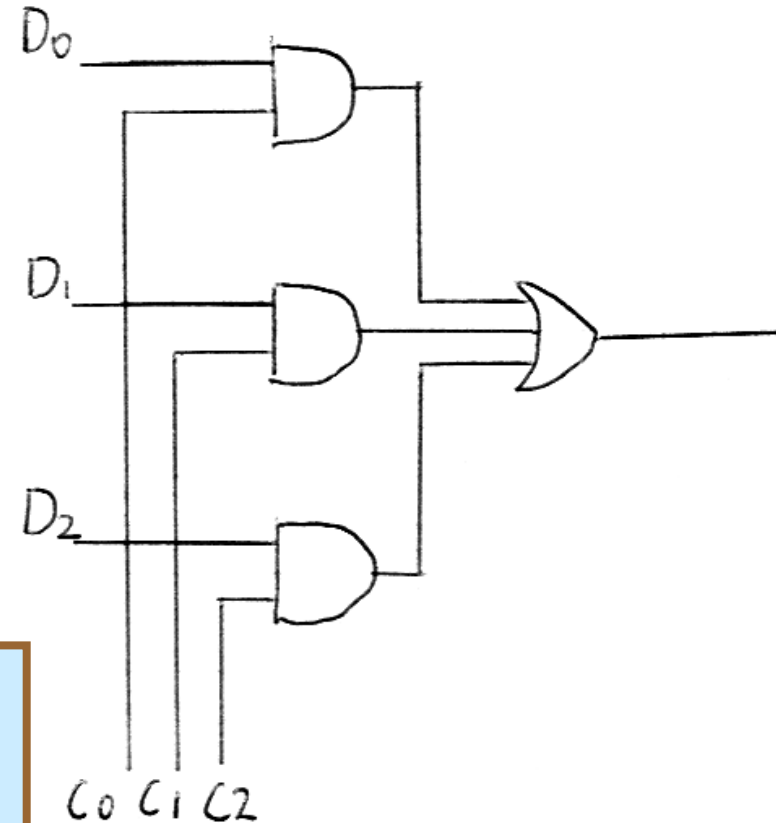
- AND/OR gate construction (MUX)
- Expensive: large transistor count (about 4trans per line)
- two gate delays
- (slow)

Hardware Intensive

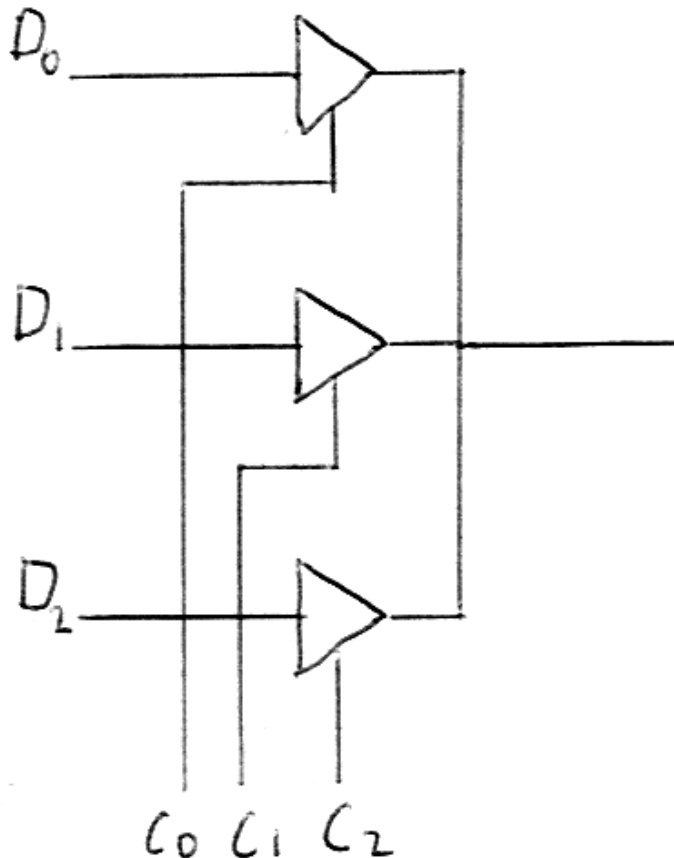
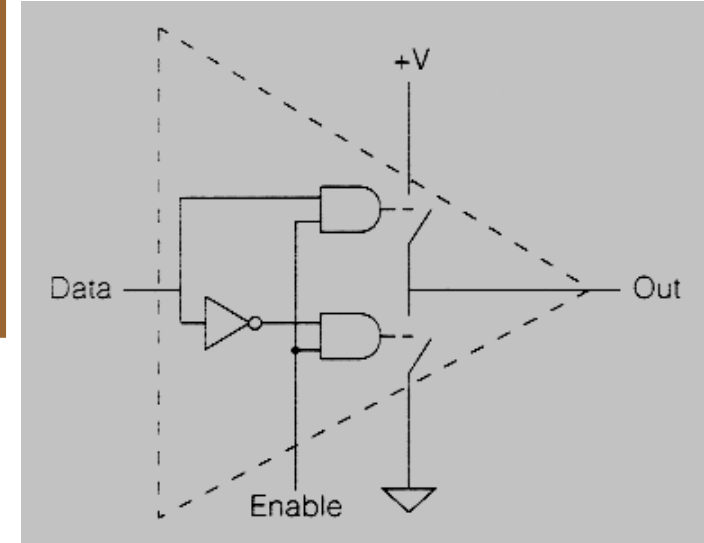
*n = # of bits in word

- n ANDS (3 trans each)
- n Enables/Controls
- n AND to OR lines
- 1 n-way OR (n+1 trans)

Total: $4n + 1$ transistors per input -
 $(4n+1) * \text{number of inputs to select from}$



- Tri-state internal construction: two ANDs + two transistors
- Faster than MUX construction
- Very expensive: (about $8n$ transistors)

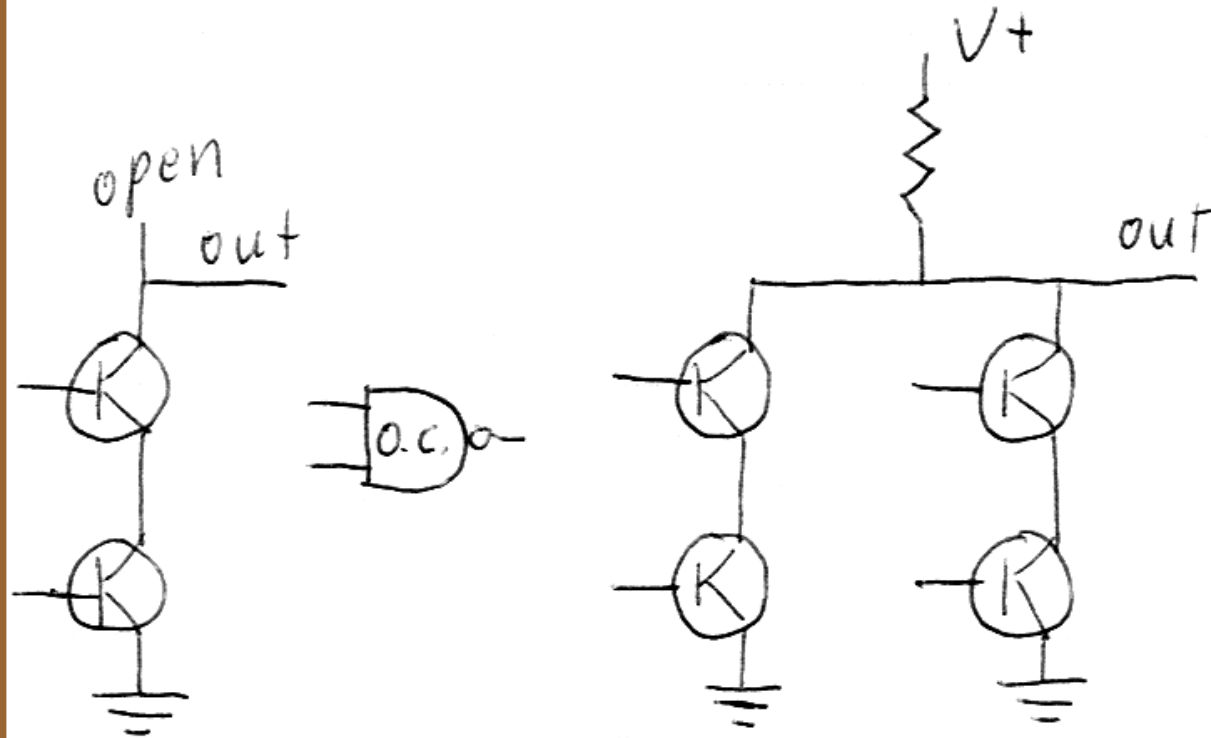


Hardware Intensive

* n = # of bits in word

- n tri-states
- Tri-state = 2 ANDS + 2 trans
- = $2 \times 3 + 2 = 8$ trans per tri-state
- $8n$ transistors

- Both **fast** and **inexpensive**
- Uses one "Open-Collector NAND" per input
- OC NANDs share a common resistor (eliminating a problem with parallel resistors – resistance drops)
- 2n transistors
- called "wired-or" because the connecting OR gate is eliminated and replaced with just a wired connection



How to control the transmission of data -
control the signals on the bus

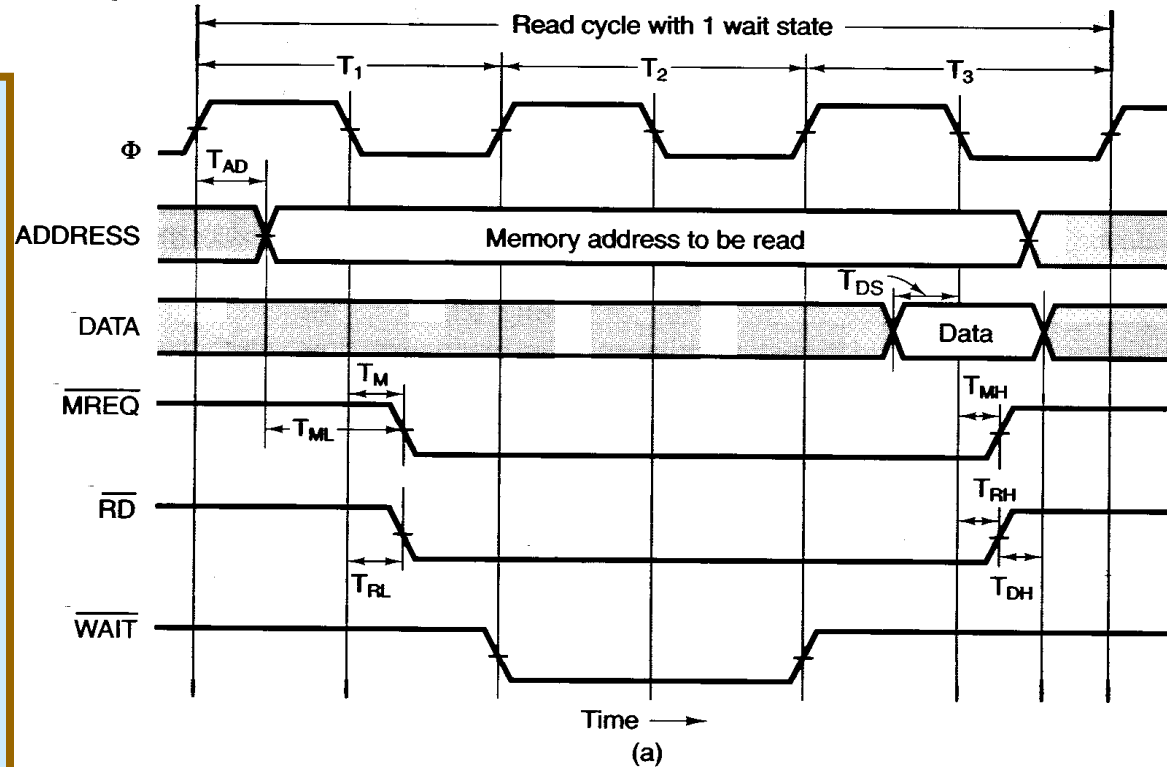
Similar to issues from Data Communications

- Signaling
- Timing
- Access control (shared access to media)

Will look at:

- Synchronous timing
- Asynchronous control
- Bus arbitration

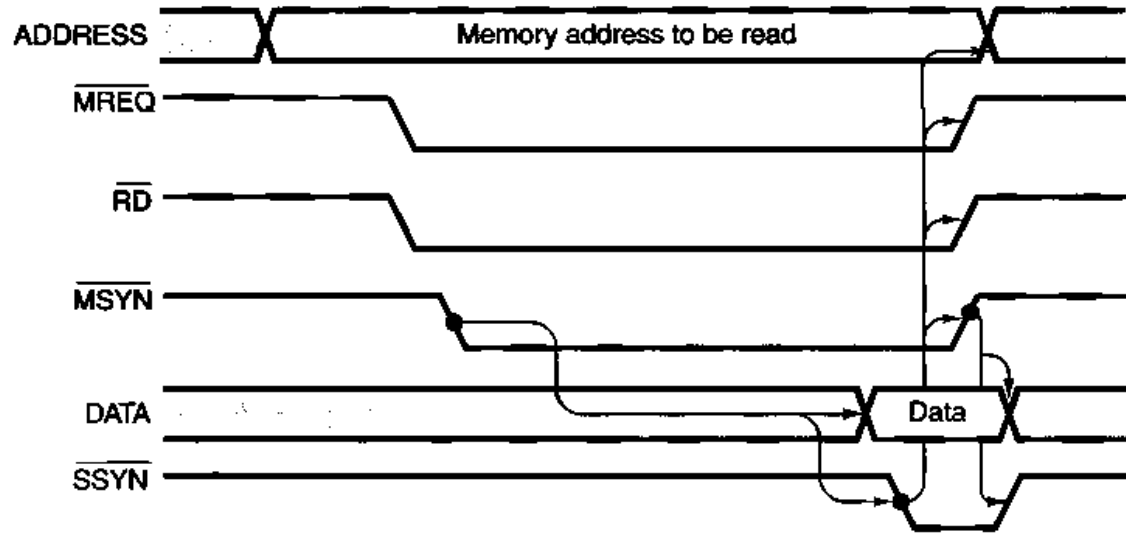
- Governed by a common clock
- Specified timings (a published protocol)
- Any manufacturer can design to bus specifications
- Events occur in relation to clock cycles
- Fixed performance - bound by the clock time



Symbol	Parameter	Min	Max	Unit
T_{AD}	Address output delay		11	nsec
T_{ML}	Address stable prior to \overline{MREQ}	6		nsec
T_M	\overline{MREQ} delay from falling edge of Φ in T_1		8	nsec
T_{RL}	\overline{RD} delay from falling edge of Φ in T_1		8	nsec
T_{DS}	Data setup time prior to falling edge of Φ	5		nsec
T_{MH}	\overline{MREQ} delay from falling edge of Φ in T_3		8	nsec
T_{RH}	\overline{RD} delay from falling edge of Φ in T_3		8	nsec
T_{DH}	Data hold time from negation of \overline{RD}	0		nsec

(b)

- NOT clock timed
- Uses a “hand-shaking” protocol
- Requires extra lines to for signaling control data
- can accommodate different speed devices



A set of signals that interlocks this way is called a **full handshake**. The essential part consists of four events:

1. $\overline{\text{MSYN}}$ is asserted.
2. $\overline{\text{SSYN}}$ is asserted in response to $\overline{\text{MSYN}}$.
3. $\overline{\text{MSYN}}$ is negated in response to $\overline{\text{SSYN}}$.
4. $\overline{\text{SSYN}}$ is negated in response to the negation of $\overline{\text{MSYN}}$.

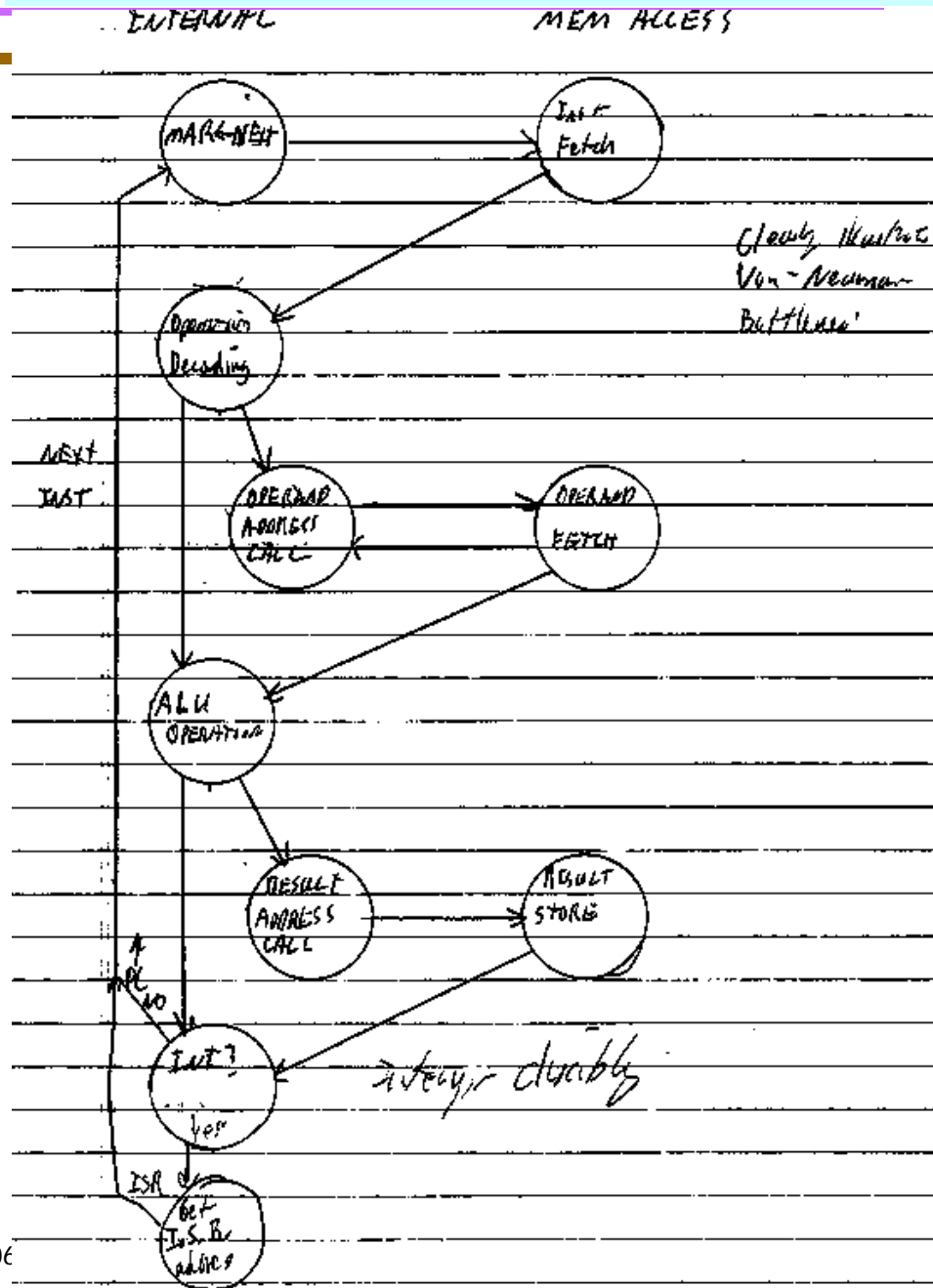
Interrupts

Interrupts are signals from system devices to the CPU requesting CPU attention.

Fetch/Decode/Execute/Interrupt Bubble

When CPU gets an interrupt, it

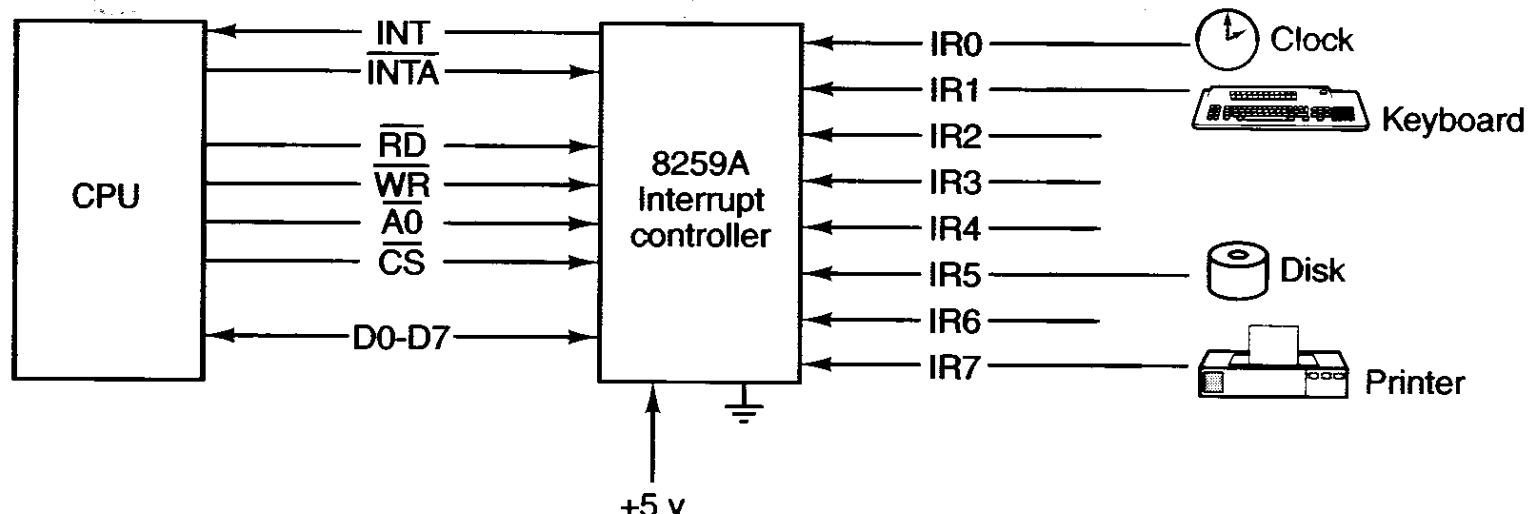
- saves current state
- gets interrupt needing service
- jumps to device driver code
- upon completion, restores state and continues



The 8259A can accommodate 8 devices

- When a device needs service, it generates an interrupt to the 8259A
- The 8259A sends an INT signal to the CPU
- When the CPU can service the interrupt, it sends a INTA
- The 8259A places the index of the ISR (for the requesting device) on the bus data lines
- The CPU uses the ISR index to lookup the address of the Interrupt Service Routine for that device
- CPU runs ISR, then returns to previous processing

8259As can be "Daisy-Chained" two levels deep (total of 64 devices)



Bus Arbitration

Issues:

- Where to locate the control logic - centralized or decentralized
- How to support priorities

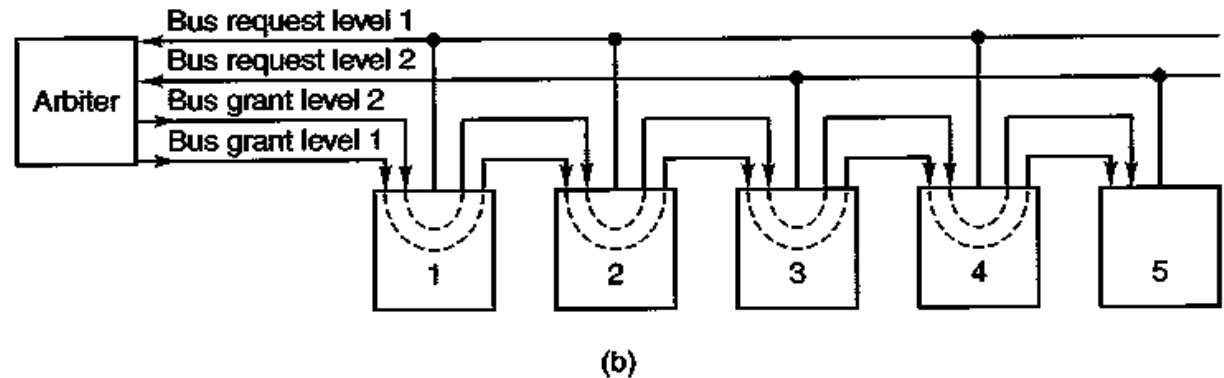
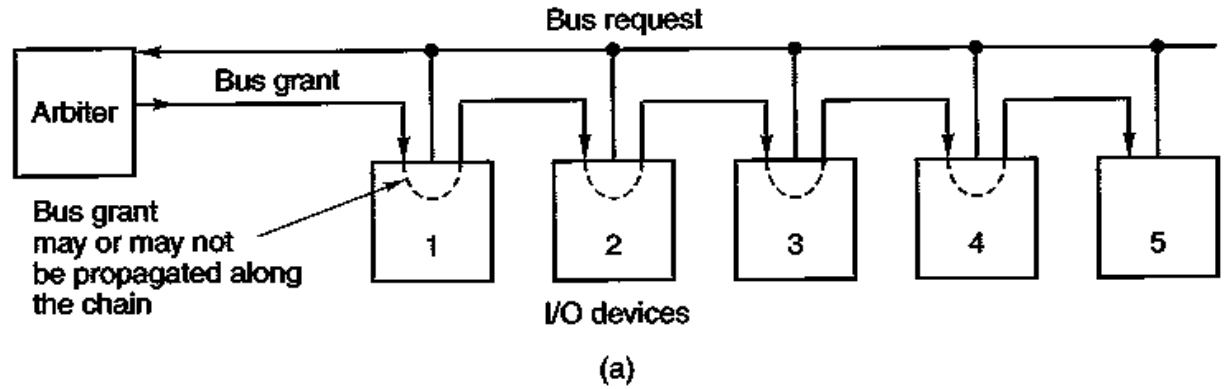


Figure 3-39. (a) A centralized one-level bus arbiter using daisy chaining. (b) The same arbiter, but with two levels.

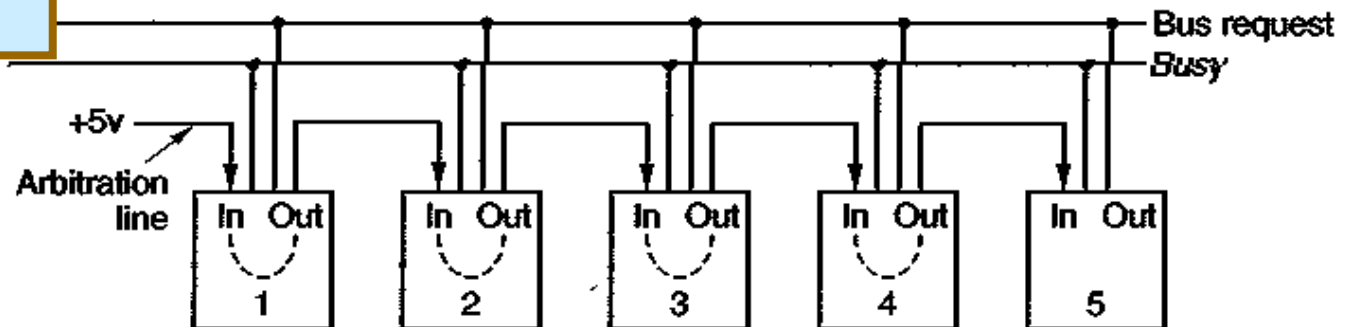


Figure 3-40. Decentralized bus arbitration.

There are many bus arbitration strategies, some overlap with Media Access Control strategies for networking and data comm

- Fixed Priority
- Arbitration Device
- Token Passing
- Centralized
- Etc.

Performance (speed) and fairness in granting access to the shared bus are design issues.

**End
Of
Today's
Lecture.**

This slide left intentionally blank.