Week 1
Introduction to Computing and the 8051 Microcontrollers
Chapters 0 and 1

Binary and Hexadecimal Systems

- Conversion to decimal:
  - 110.101 b = ?
  - 6A.C h = ?
- Conversion from decimal
  - for a whole number: divide by the radix and save the remainder as the significant digits
  - 10 = ? B
  - 10 = ? $_8$
- Converting from a decimal fraction
  - multiply the decimal fraction by the radix
  - save the whole number part of the result
  - repeat above until fractional part of step 2 is 0
  - 0.125 = ? b
  - 0.046875 = ? h

- 110.101 b = 6.625
- 6A.C h = 106.75

- 10 = 1010 b

- 10 = 12 $_8$

- 0.125 = 0.001 b
- 0.046875 = 0.0C h

Base 16 Number systems

Decimal	Binary	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

- Represent 100111110101b in hex

  Group them

  1001 1111 0101

  9 F 5 hex

- Convert 1714d to binary

  1714 = 16*107 +2

  107 = 16*6 + 11

  1714 = 6B2 h

  = 0110 1011 0010 b

Two's Complement

- If the number is positive make no changes
- If the number is negative, complement all bits and add by 1
  - -6 => $\overline{0000\ 0110}$ + 1 = 1111 1001 + 1 = FAh
- 8 bit signed numbers
  - 0 to 7Fh (+127) are positive numbers
  - 80h (-128)to FFh (-1) are negative numbers
- Conversion of signed binary numbers to their decimal equivalent
  - 1101 0001
    - $\overline{1101\ 0001}$ + 1 = 0010 1110 + 1 = 0010 1111 = 2Fh => -47
  - 1000 1111 0101 1101
    - 0111 0000 1010 0010 + 1 = 0111 0000 1010 0011 = 70C3h => -28835
- Two's complement arithmetic
  - +14 - 20
  - 0000 1110 + $\overline{0001\ 0100}$ + 1 = FAh
- *Overflow:* Whenever two signed numbers are added or subtracted the possibility exists that the result may be too large for the number of bits allocated Ex: +64 +96 using 8-bit signed numbers

Two's complement

Decimal	Binary	Hex
-128	1000 0000 b	80h
-127	1000 0001b	81h
-126	1000 0010b	82h
…		…
-2	1111 1110b	FEh
-1	1111 1111b	FFh
0	0000 0000b	00h
1	0000 0001b	01h
…		…
+127	0111 1111b	7Fh

Numbers in the range -2^n … 2^n –1
are represented by 8 bit signed arithmetic

ASCII

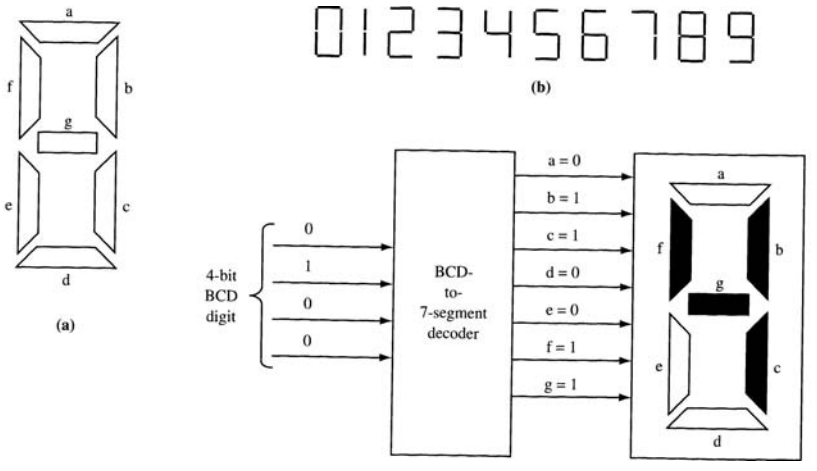
- The standard for text
- In this code each letter of the alphabet, punctuation mark, and decimal number is assigned a unique 7-bit code number
- With 7 bits, 128 unique symbols can be coded
- Often a zero is placed in the most significant bit position to make it an 8-bit code
  - e.g., Uppercase A 41h
- Digits 0 through 9 are represented by ASCII codes 30h to 39h

ASCII - more

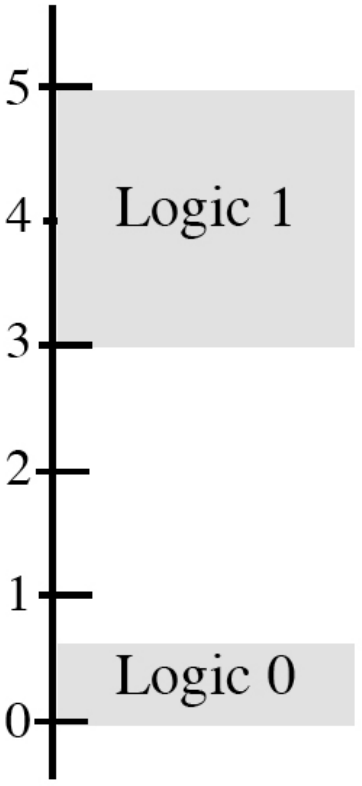
Hex	*Symbol*	*Hex*	*Symbol*
41	A	61	a
42	B	62	b
43	C	63	c
44	D	64	d
...	...	...	...
59	Y	79	y
5A	Z	7A	z

BCD

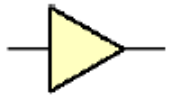
- BCD code provides a way for decimal numbers to be encoded in binary form that is easily converted back to decimal
  - 26 => 0010 0110 (BCD) => 11010 (unsigned binary)
  - 243 => 0010 0100 0011 (BCD) => 1111 0011 (unsigned binary)
- Used in seven segment displays



Digital Primer



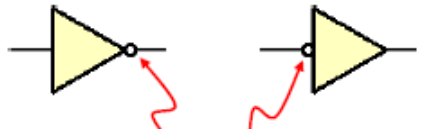
- Buffer
  - Identity S=A



Truth table

A	S
0	0
1	1

- Inverter
  - Negation S=$\overline{A}$

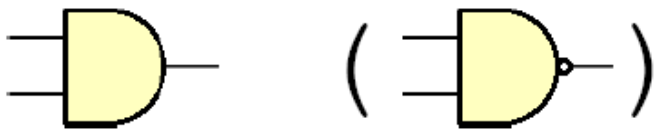


A	S
0	1
1	0

Inversion

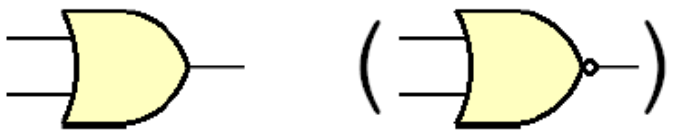
AND and OR Gates

- AND (and NAND)
  - Multiplication S=A·B
  - Zero forces output to zero



A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

- OR (and NOR)
  - Addition S=A+B
  - One forces output to one

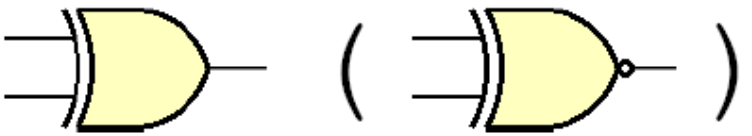


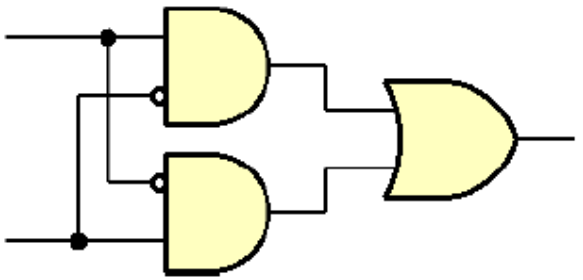
A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

XOR Gate

XOR (and XNOR)

- Modulo-2 addition $S = A \oplus B$
- "One or the other but not both"
- Inequality (XNOR expresses equality)



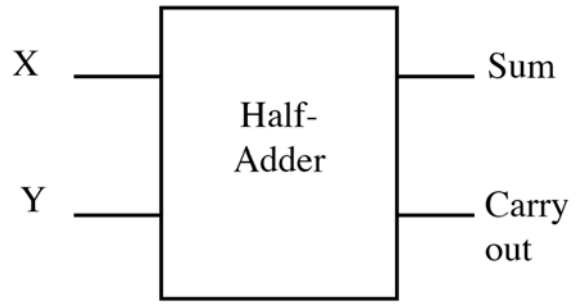


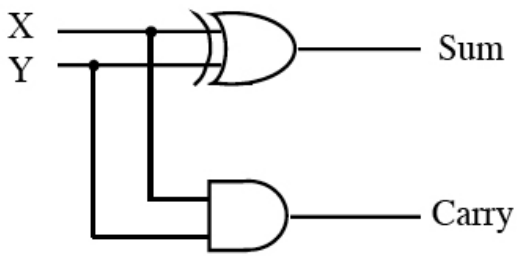
A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

$$S = A \oplus B = A\overline{B} + B\overline{A}$$

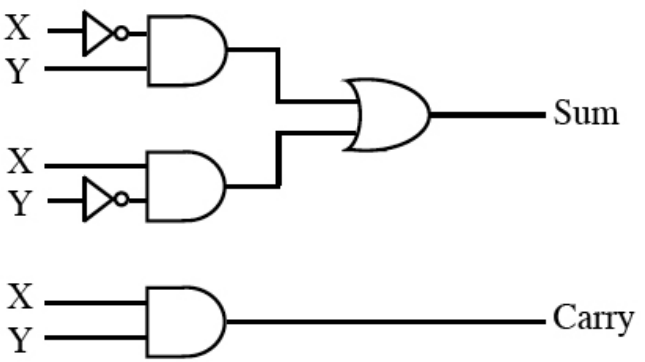
Logic Design using Gates

- Two implementations of a half-adder



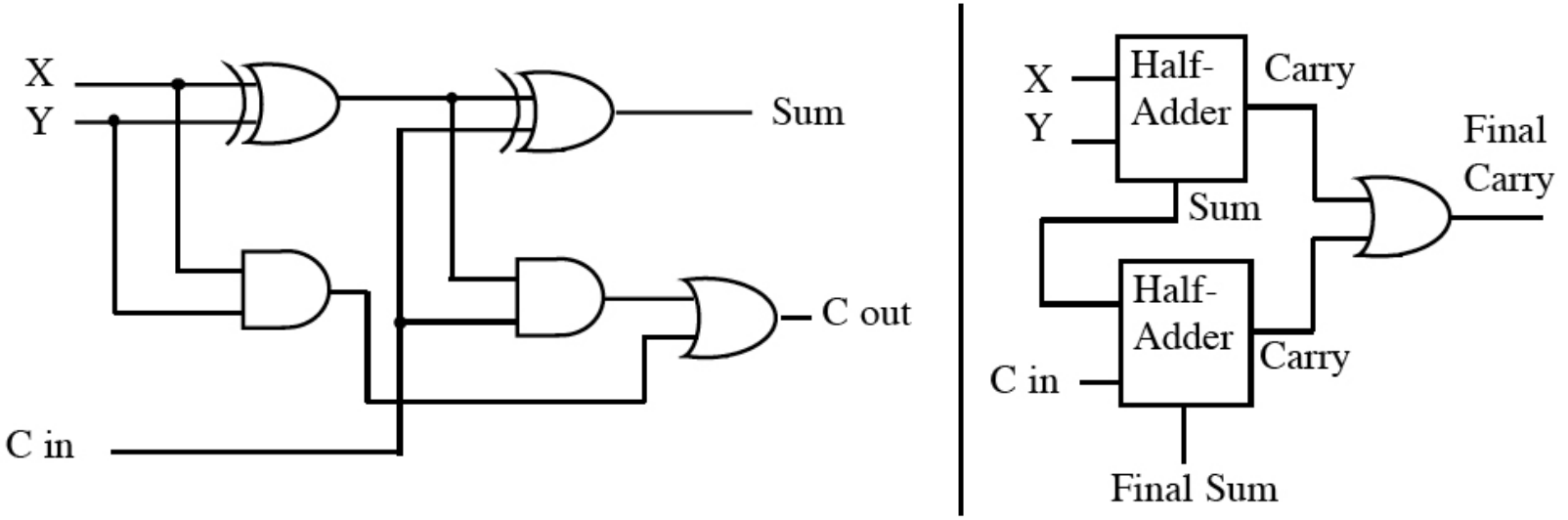


(a) Half-Adder Using XOR and AND

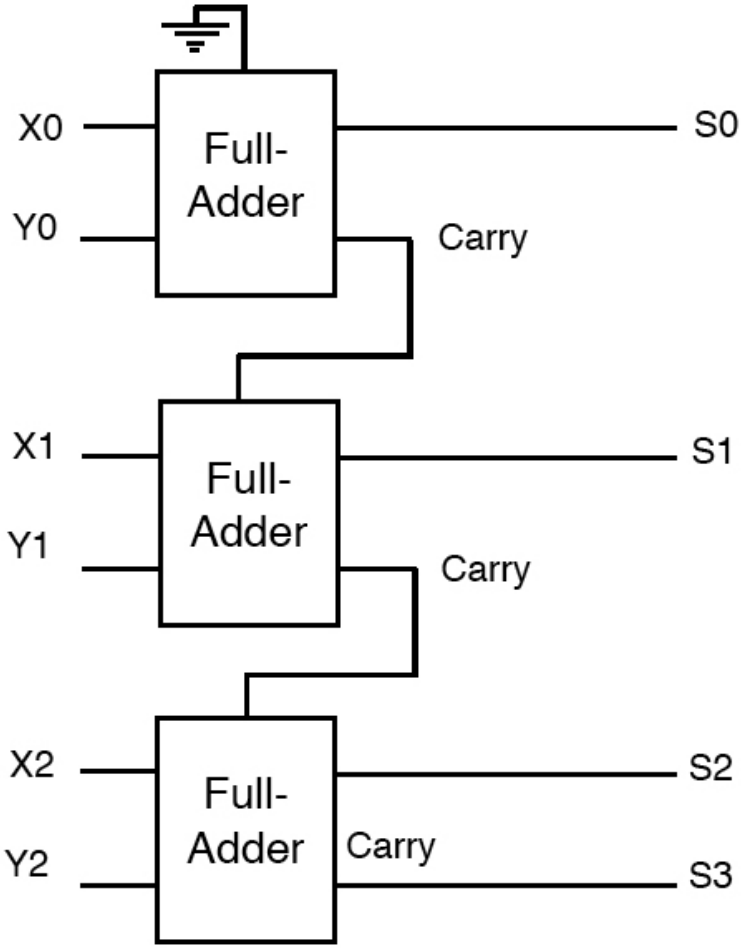


(a) Half-Adder Using AND, OR, Inverters

Full adder using half adders



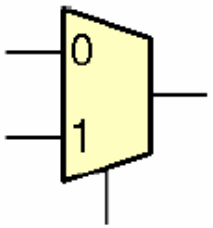
3-bit adder using 3 full-adders

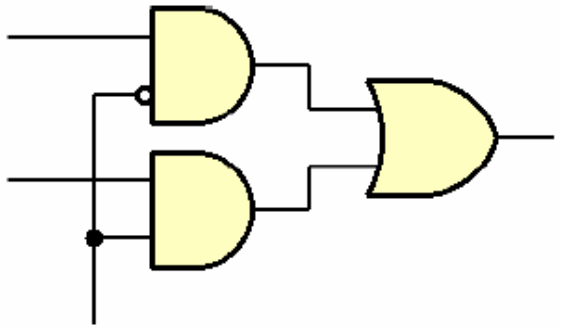


Multiplexer

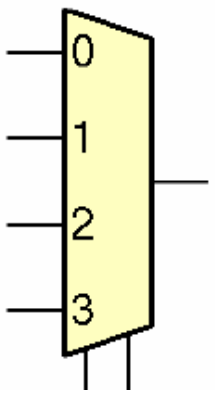
Multiplexer ("mux")

- If S=0 → Z=A else Z=B

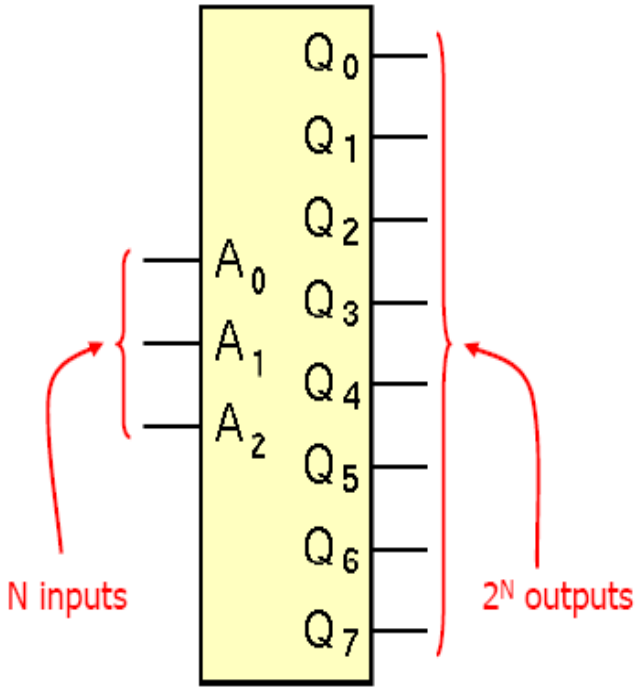




n-input Multiplexer



N to 2^N Decoder

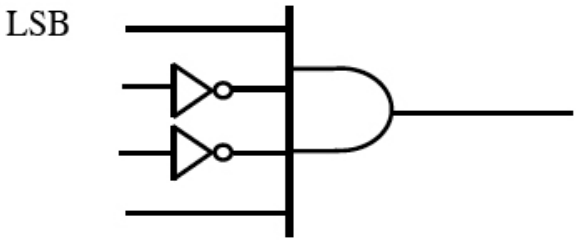


A_2	A_1	A_0	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

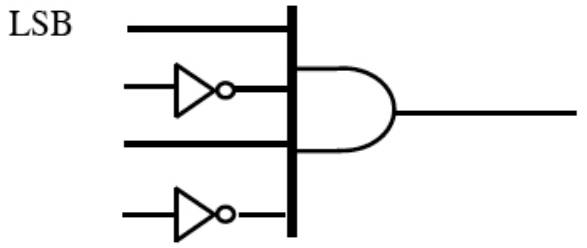
Binary representation of a number

"One hot"

Address decoders



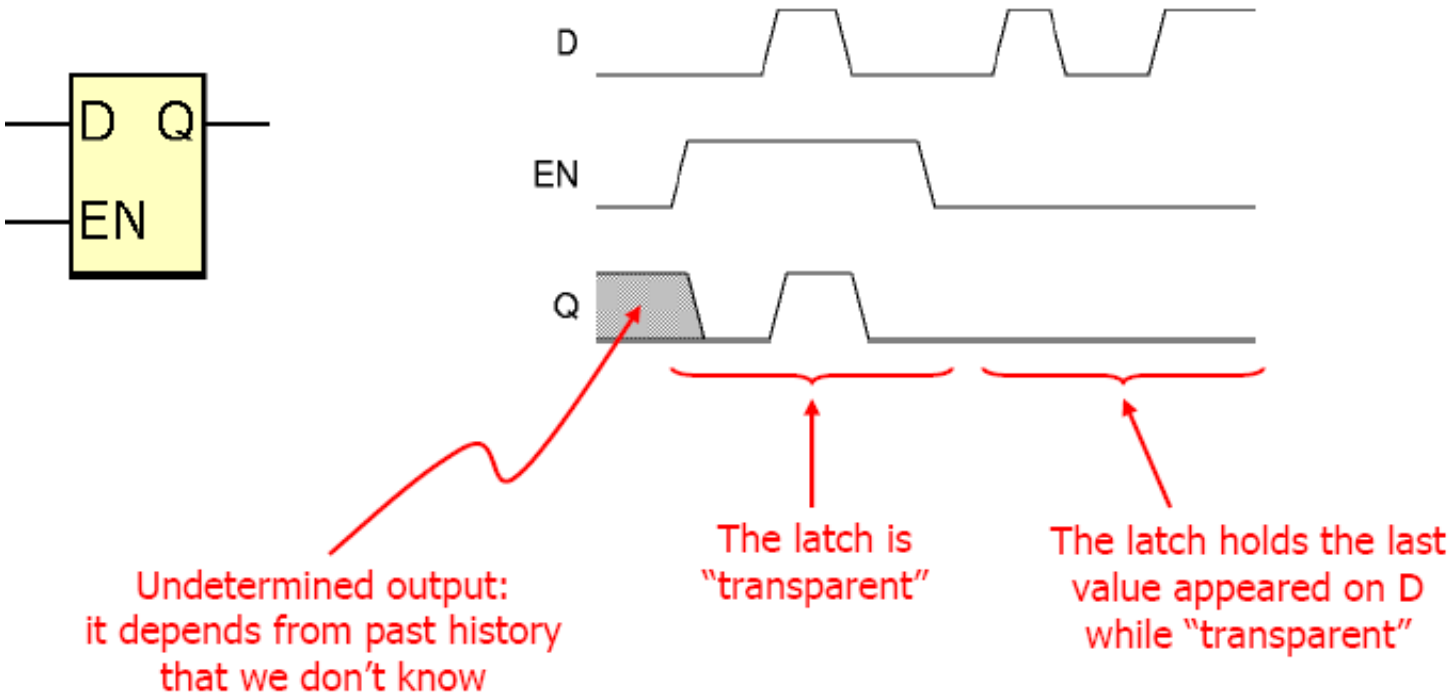
(a) Address decoder for 9 (binary 1001)
The output of the AND gate will be 1
if and only if the input is binary 1001.



(b) Address decoder for 5 (binary 0101)
The output of the AND gate will be 1
if and only if the input is binary 0101.

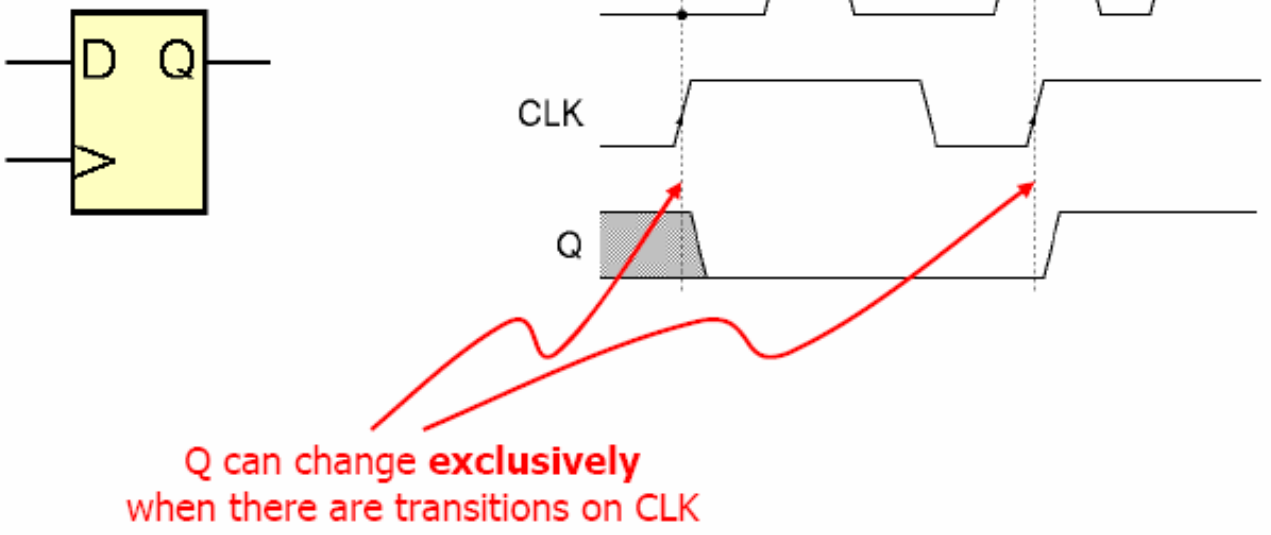
Latch

- The simplest memory element
- **Level-sensitive**: it memorizes the input data when there is a given level on the control input

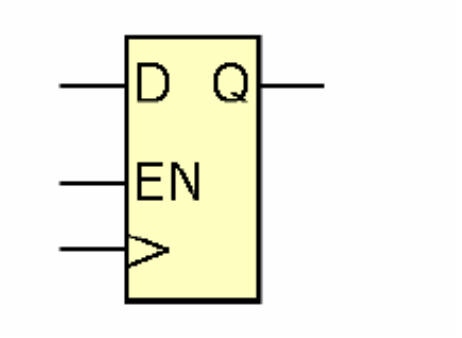


D type flip flop (DFF)

- The only Flip-Flop we use (forget SR, JK, etc.)
- The most used memory element
- **Edge-sensitive**: it memorizes the input data when there is a specific transition (e.g., 0 → 1) on the control input



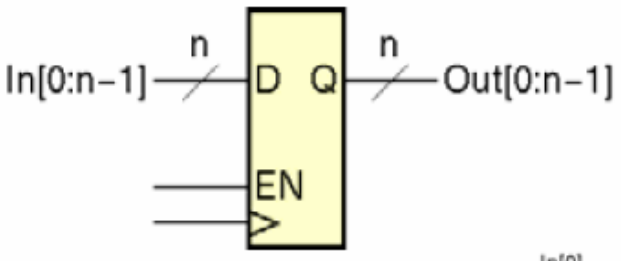
DFF with Enable

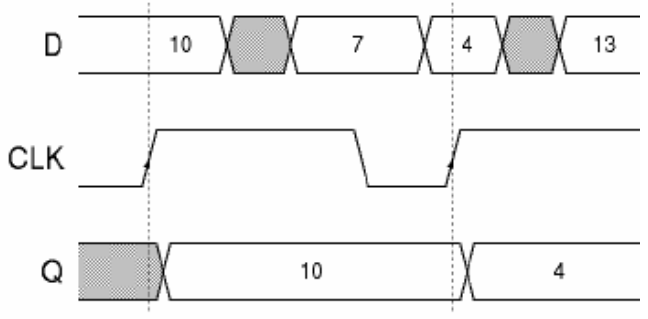


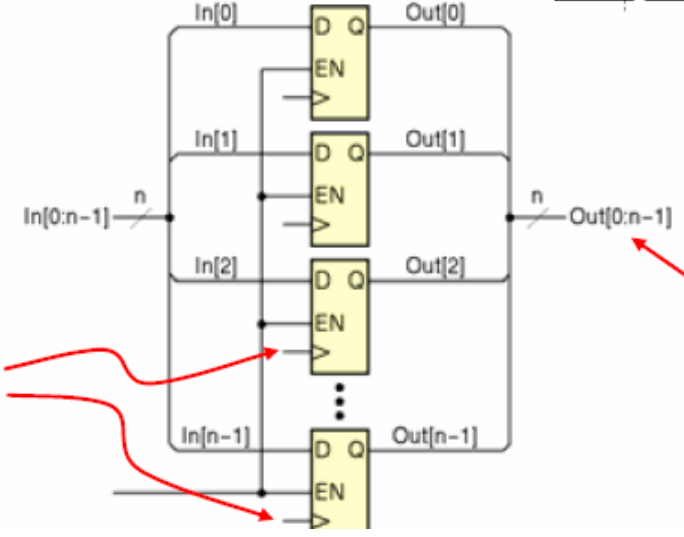
- If EN = 0 when there is an edge, the edge is ignored;
- If EN = 1, normal behaviour

Registers

- Just several DFF, usually with Enable, in parallel





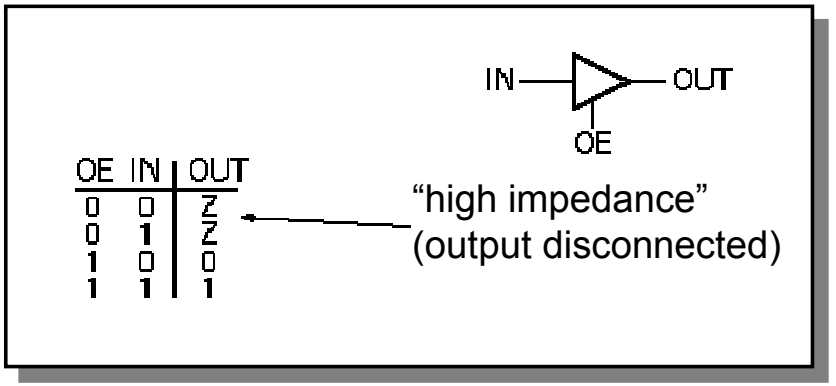


CLK inputs are all in parallel too, but we don't always indicate it...

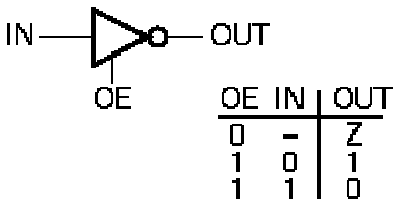
A **bus**, that is many signals handled together

Tri-state Buffers

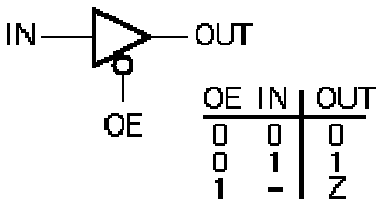
- Tri-state



- Variations

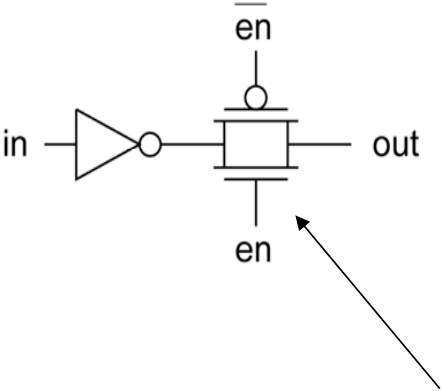


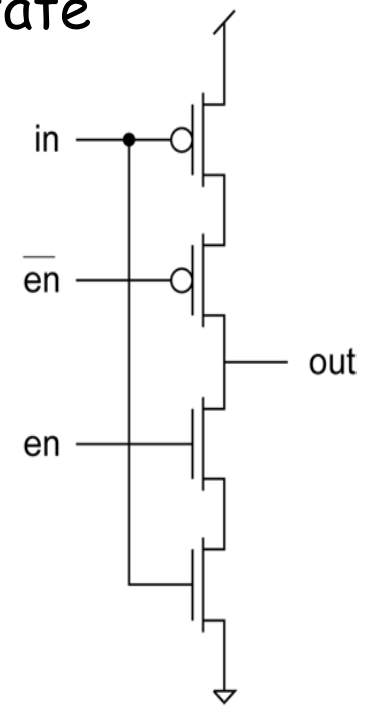
Inverting buffer



Inverted enable

- Transistor circuit for inverting tri-state buffer:



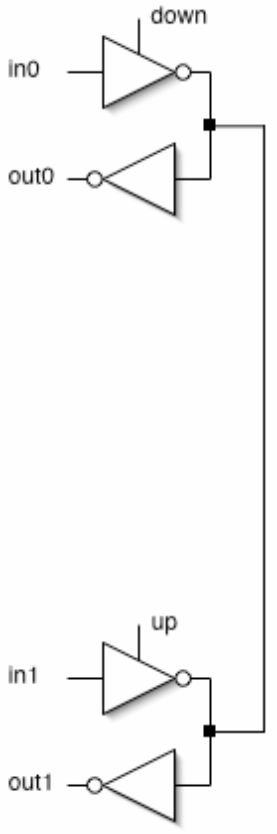


"transmission gate"

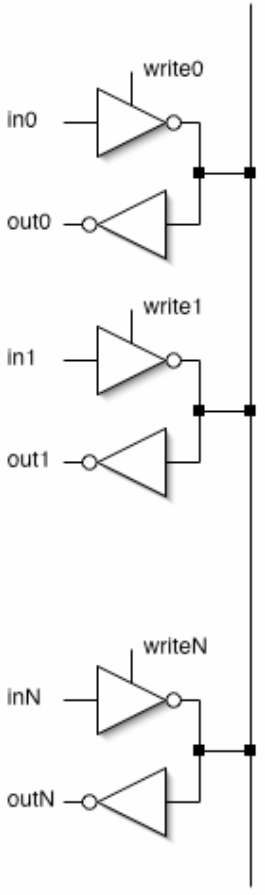
Tri-state Buffers

Tri-state buffers are used when multiple circuits all connect to a common bus. Only one circuit at a time is allowed to drive the bus. All others "disconnect".

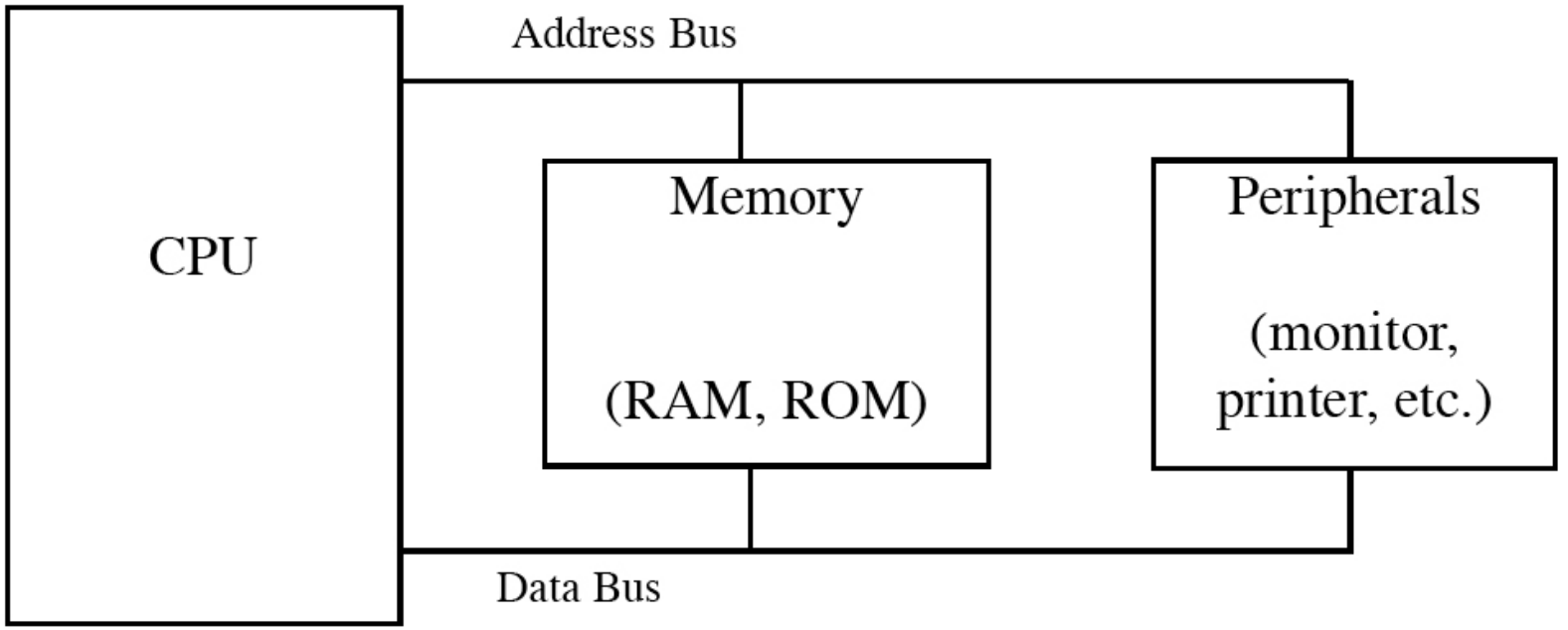
- Bidirectional connections:



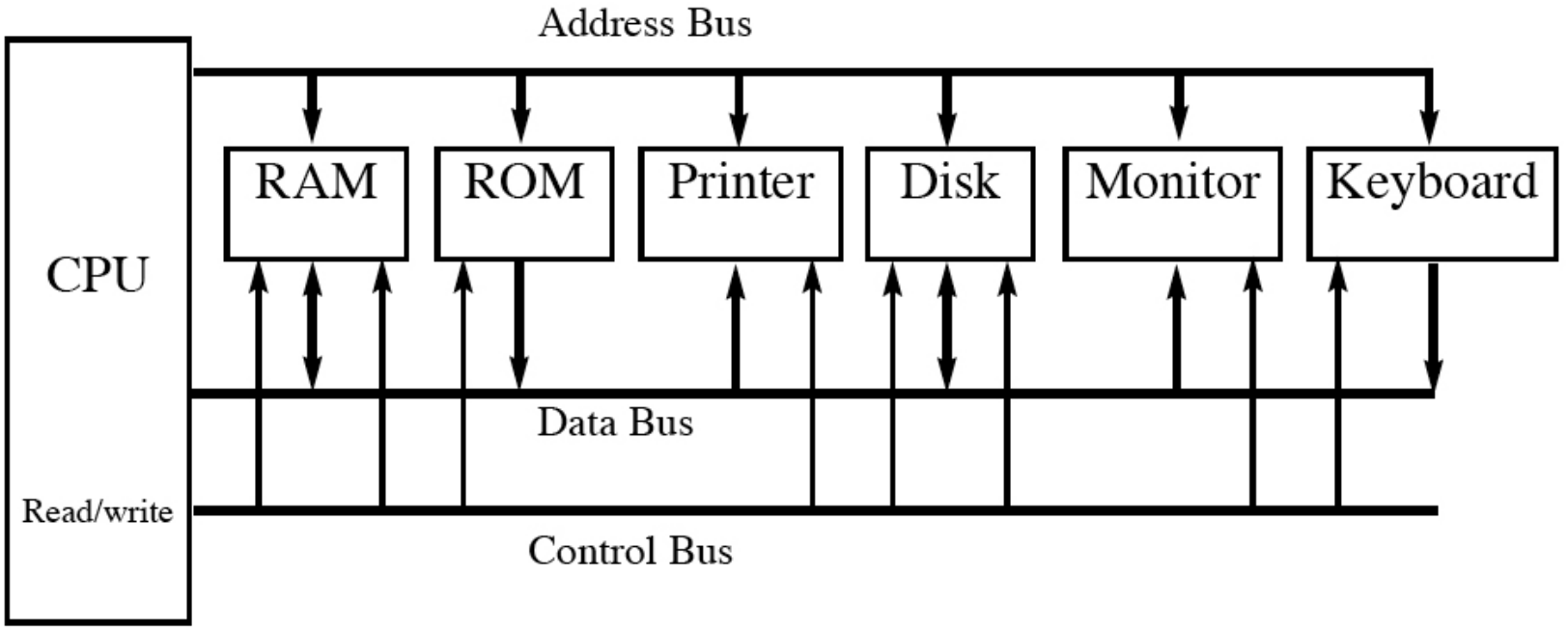
- Busses:



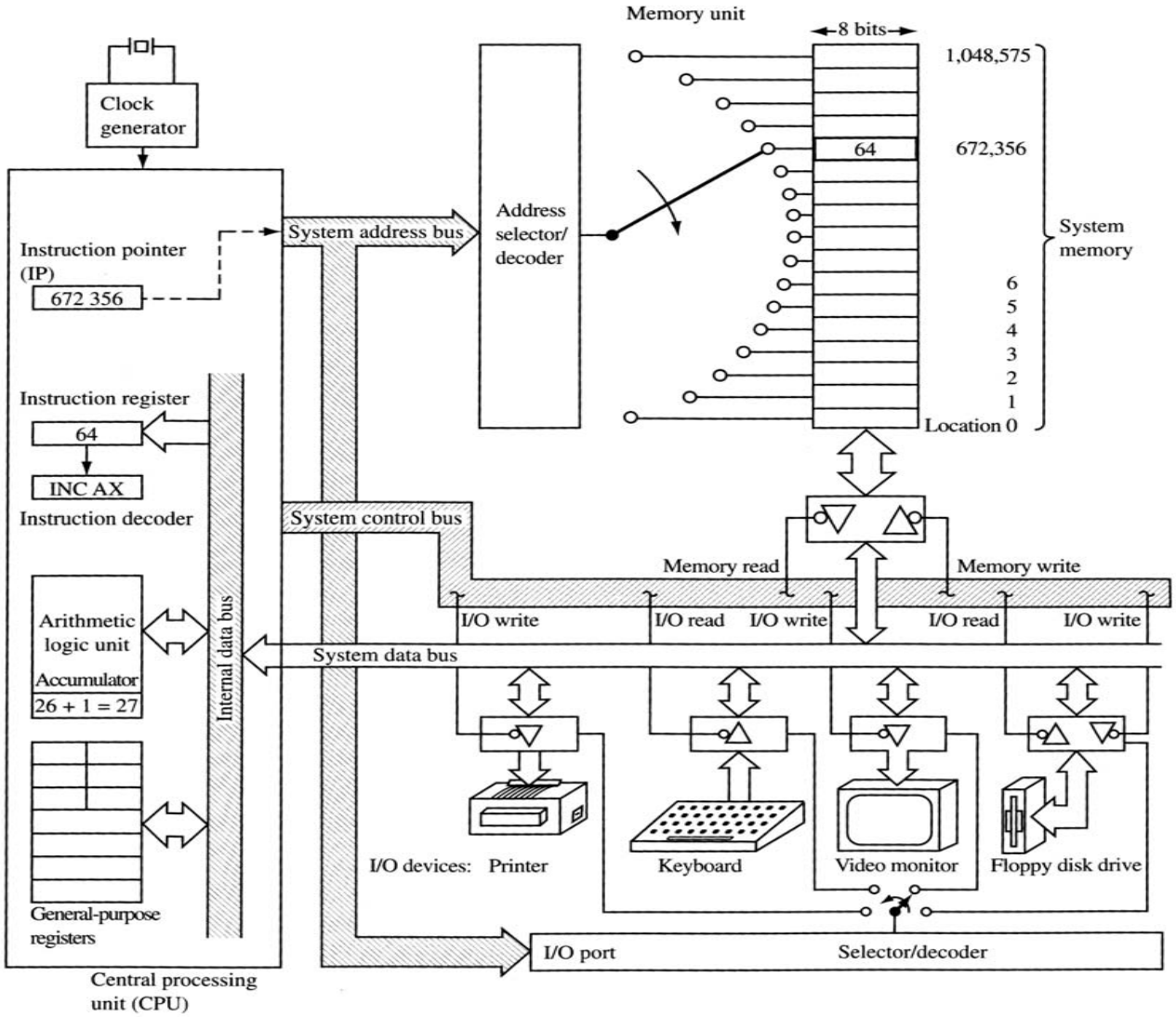
Inside the Computer



Inside the Computer - More



Stored Program concept



Stored Program Concept

- There are three major parts
  - The CPU (Central Processing Unit) which acts as the brain coordinating all activities within the computer
  - The memory unit where the program instructions and data are temporarily stored
  - The I/O (Input/Output) devices which allow the computer to input information for processing and then output the result
- Today the CPU circuitry has been reduced to *ICs* called the *microprocessor*, the entire computer with the three parts is called a *microcomputer*

Stored Program Concept - more

- Several registers (e.g., flip-flops wired in series with each other)
  - Some are general purpose, the accumulator for example is reserved for performing complex mathematical operations like multiply and divide, and all I/O data has to go thru the accumulator
- The basic timing of the computer is controlled by a square wave oscillator or a *clock* generator circuit.
  - Synchronization
  - Determines how fast the program can be fetched from memory and executed
- Memory Read or Fetch Cycle
  - IP: Instruction Pointer

Stored Program Concept

- Memory unit consists of a large number of storage locations each with its own address
  - RAM (Random Access Memory) and its volatility
    - Typically each memory location is 8 bits wide (byte accessible memory)
  - ROM (Read Only Memory)
- The memory unit's address selector/decoder circuit examines the binary number on the address line and selects the proper memory location to be accessed.

Stored Program Concept

- In this example, CPU is reading from memory, it activates its MEMORY READ control signal
  - This causes the selected data byte in memory to be placed onto the data lines and routed to the instruction register in the CPU
- Once in the CPU, the instruction is decoded and executed
  - In this example, instruction has the decimal code 64 which for a 8086 microprocessor is decoded to be INC AX
  - The ALU (Arithmetic Logic Unit) is instructed to add 1 to the contents of the AX
- The cycle repeats itself

Stored Program Concept

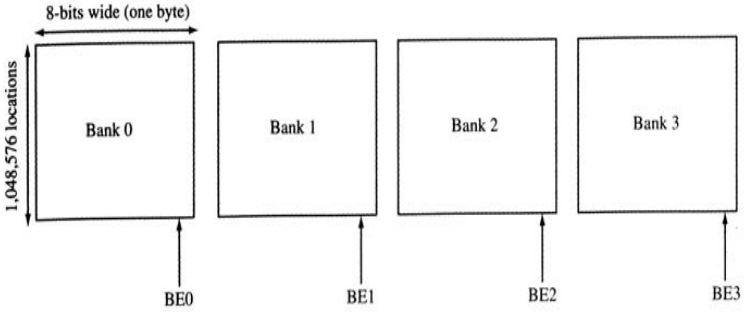
- Memory unit consists of a large number of storage locations each with its own address.
- RAM (Random Access Memory): also called read/write memory
  - used for temporary storage of programs
  - typically 8 bits wide
  - data is lost when the power is turned off (volatile)
- ROM (Read Only Memory)
  - The information in ROM is permanent
  - Non-volatile memory

Three Bus System Architecture

- A collection of electronic signals all dedicated to particular task is called a *bus*
  - *address bus*
  - *data bus*
  - *control bus*
- Data Bus
  - The width of the data bus determines how much data the processor can read or write in one memory or I/O cycle
  - 8-bit microprocessor has an 8-bit data bus
  - 80386SX 32-bit internal data bus, 16-bit external data bus
  - 80386 32-bit internal and external data buses
- How can a 64-bit (or 16 bit) microprocessor access an 8-bit memory?
  - The trick is to divide the memory into banks
  - 64-bit Pentium requires eight banks of memory with each bank set up to be one-byte wide
  - Bank enable signals are then output by the microprocessor to specify which bank to access

Address Bus

- Address Bus
  - The address bus is used to identify the memory location or I/O device (also called port) the processor intends to communicate with
  - 20 bits for the 8086 and 8088
  - 32 bits for the 80386/80486 and the Pentium
  - 36 bits for the Pentium II and III
  - The total number of memory locations addressable by a given CPU is always equal to 2^x where x is the number of address bits, regardless of the data bus.



Total amount
of memory is 4Mbytes

8086 has a 20-bit address bus and therefore addresses all combinations of addresses from all 0s to all 1s. This corresponds to 2 20 addresses or 1M (1 Meg) addresses or memory locations.

Pentium: 4Gbyte main memory

Control Bus

- How can we tell the address is a memory address or an I/O port address
  - Memory Read
  - Memory Write
  - I/O Read
  - I/O Write
- When Memory Read or I/O Read are active, data is *input* to the processor.
- When Memory Write or I/O Write are active, data is *output* from the processor.
- The control bus signals are defined from the processor's point of view.
- Control and address lines are output lines only but the data bus is bidirectional

Some Important Terminology

- Bit is a binary digit that can have the value 0 or 1
- A byte is defined as 8 bits
- A nibble is half a byte
- A word is two bytes
- A double word is four bytes
- A kilobyte is 2^10 bytes (1024 bytes), The abbreviation K is most often used
  - Example: A floppy disk holding 356Kbytes of data
- A megabyte or meg is 2^20 bytes, it is exactly 1,048,576 bytes
- A gigabyte is 2^30 bytes

Internal Working of Computers

- Assume that an imaginary CPU has registers called A,B,C, D.
- It has an 8-bit data bus and a 16-bit address bus.
- Therefore the CPU can access memory from addresses 0000h to FFFFh for a total of 2^16 locations
- The action to be performed by the CPU is to put a hexadecimal value 21 into register A, and add to register A values 42h and 12h.
- Assume that the code for the CPU to move a value to register A is 1011 0000b (B0h) and the code for adding a value to register A is 0000 0100b (04h)

Action	Code	Data
Move 21h to A	B0h	21h
Add 42h to A	04h	42h
Add 12h to A	04h	12h

Example Continued

Memory Address	Content of memory
1400h	B0h
1401h	21h
1402h	04h
1403h	42h
1404h	04h
1405h	12h
1406h	F4h (the code for halt)

- Assume program is stored at memory locations starting at 1400h

Internal Working Of Computers

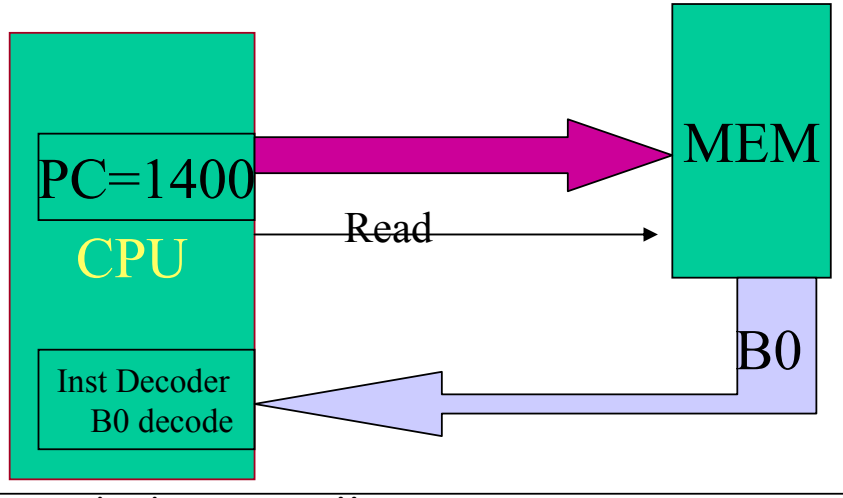
ACTION	Code	Data
Move value 21 into register A	B0H	21H
Add value 42H to register A	04H	42H
Add value 12H to register A	04H	12H

Memory Address	Contents of memory address
1400	(B0) the code for move to A
1401	(21) the value for A
1402	(04) the code for adding a value to A
1403	(42) the value to be added
1404	(04) the code for adding a value to A
1405	(12) the value to be added
1406	(F4) the code for halt

Internal Working Of Computers

1- the CPU program counter can have any value between 0000 → FFFF H. This one is set to start with 1400

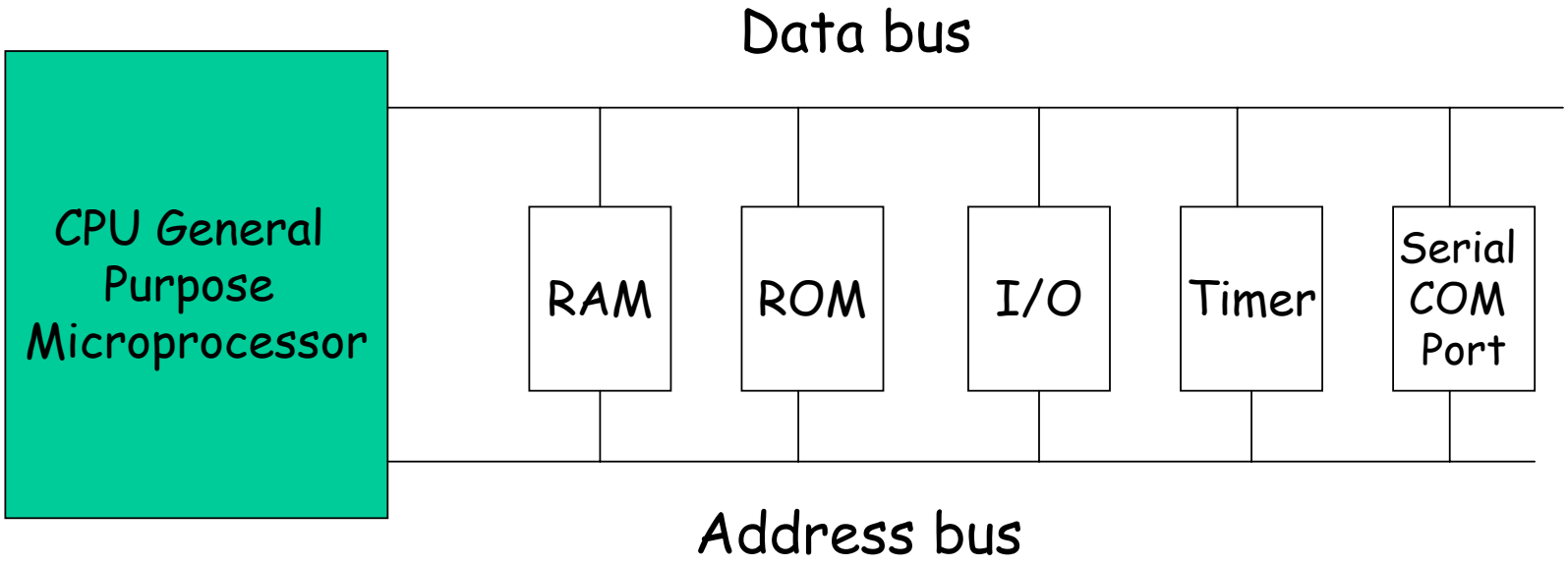
2- the CPU puts out 1400. The memory circuitry finds the location. Activates the read signal, indicating the memory location 1400. B0 is put on the bus and brought to the CPU



3- B0 is decoded internally it now knows it needs to fetch the next byte!. It brings 21h from 1401. The program counter automatically increments itself to the next location to fetch the next data/instruction.

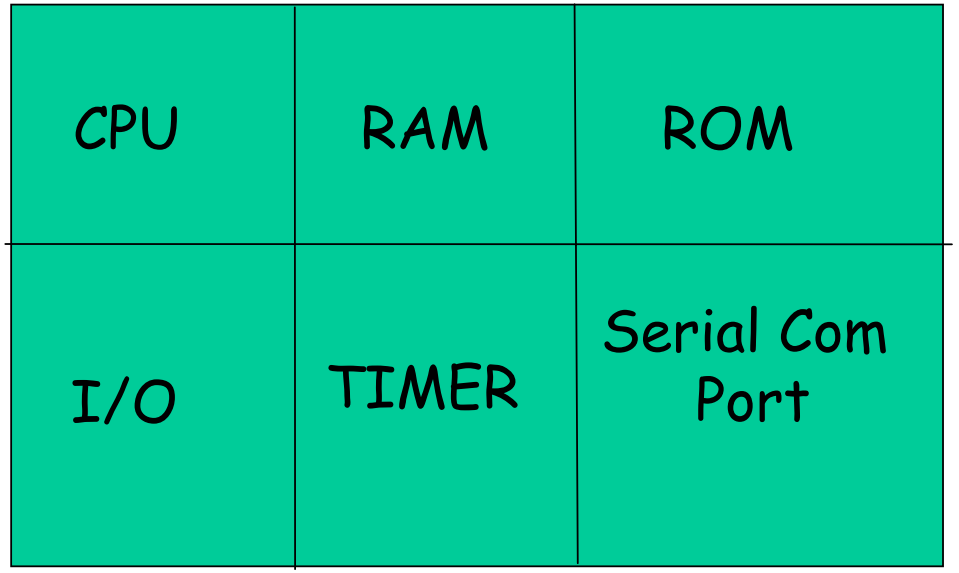
General Purpose Microprocessors

Microprocessors lead to versatile products



- These general microprocessors contain no RAM, ROM, or I/O ports on the chip itself
- Ex. Intel's x86 family (8088, 8086, 80386, 80386, 80486, Pentium)
- Motorola's 680x0 family (68000, 68010, 68020, etc)

Microcontrollers



- Examples
- Motorola's 6811, Intel's 8051, Zilog's Z8 and PIC 16X

• A microcontroller has a CPU in addition to a fixed amount of RAM, ROM, I/O ports on one single chip; this makes them ideal for applications in which cost and space are critical

• Example: a TV remote control does not need the computing power of a 486

Microprocessor vs microcontroller

Microprocessor

- CPU is stand-alone, RAM, ROM, I/O, timer are separate
- Designer can decide on the amount of ROM, RAM and I/O ports.
- General-purpose
- Expensive

Microcontroller

- CPU, RAM, ROM, I/O and timer are all on a single chip
- Fix amount of on-chip ROM, RAM, I/O ports
- Single-purpose
- Inexpensive
- For applications in which cost, power and space are critical

Embedded Systems

- Embedded system means the processor is embedded into that application.
- An embedded product uses a microprocessor or microcontroller to do one task only.
- In an embedded system, there is only one application software that is typically burned into ROM.
- Table 1-1, some embedded products using microcontrollers. Examples: printer, keyboard, video game player, door opener, copier, ABS, fax machine, camera, cellular phone, keyless entry, microwave...

Embedded Systems

- Although microcontrollers are the preferred choice for embedded systems, there are times that the microcontroller is inadequate for the task
- Intel, Motorola, AMD, Cyrix have also targeted the embedded market with their general purpose microprocessors
- For example, Power PC microprocessors (IBM Motorola joint venture) are used in PCs and routers/switches today
- Microcontrollers differ in terms of their RAM,ROM, I/O sizes and type.
  - ROM: One time-programmable, UV-ROM, flash memory

Embedded Systems - more

- Which is your choice for an embedded product?
- microcontroller
  - cost down
  - embedded processor or microcontroller
- microprocessor
  - In future, an entire computer on a chip
  - high-end embedded systems use microprocessors
  - Advantage: rapid software development, all (appliances) in one.

How to choose a microcontroller

1. meeting the computing needs of the task efficiently and cost effectively
   - speed, the amount of ROM and RAM, the number of I/O ports and timers, size, packaging, power consumption
   - easy to upgrade
   - cost per unit
2. availability of software development tools
   - assemblers, debuggers, C compilers, emulator, simulator, technical support
3. wide availability and reliable sources of the microcontrollers.

Intel 8051

- 1981, Intel MCS-51
- The 8051 became popular after Intel allowed other manufacturers to make and market a flavor of the 8051.
  - different speed, amount of on-chip ROM
  - code-compatible with the original 8051
  - form a 8051 family

8051 Features

Feature	Quantity	Notes
ROM	4K bytes	a fixed program
RAM	128 bytes	temporary data
Timers 0,1	2	Timer/counter
I/O pins	32	P0,P1,P2,P3
Serial port	1	TxD, RxD
Interrupt sources	6	

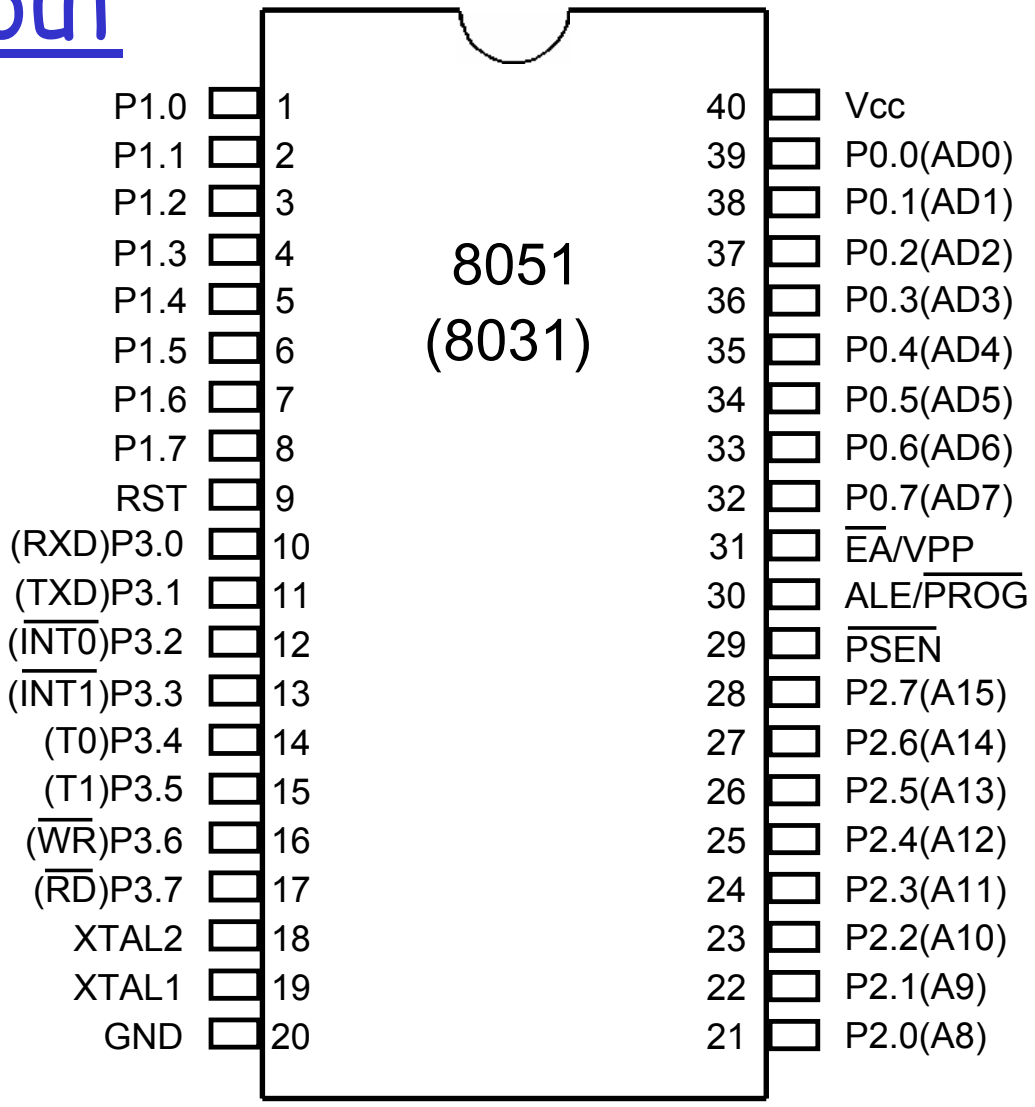
Comparison of 8051 Members

Table 1-4 Comparison of the 8051 Family Members

Feature	**8051**	**8052**	**8031**
ROM (program space in bytes)	4K	8K	0
RAM (bytes)	128	256	128
Timers	2	3	2
I/O pins	32	32	32
Serial port	1	1	1
Interrupt sources	6	8	6

- 8031 is referred as ROM-less 8051
- To use ROM you must add external ROM to it but you lose two ports

8051 Layout


8051
(8031)
P1.0 1
P1.1 2
P1.2 3
P1.3 4
P1.4 5
P1.5 6
P1.6 7
P1.7 8
RST 9
(RXD)P3.0 10
(TXD)P3.1 11
(INT0)P3.2 12
(INT1)P3.3 13
(T0)P3.4 14
(T1)P3.5 15
(WR)P3.6 16
(RD)P3.7 17
XTAL2 18
XTAL1 19
GND 20
40 Vcc
39 P0.0(AD0)
38 P0.1(AD1)
37 P0.2(AD2)
36 P0.3(AD3)
35 P0.4(AD4)
34 P0.5(AD5)
33 P0.6(AD6)
32 P0.7(AD7)
31 EA/VPP
30 ALE/PROG
29 PSEN
28 P2.7(A15)
27 P2.6(A14)
26 P2.5(A13)
25 P2.4(A12)
24 P2.3(A11)
23 P2.2(A10)
22 P2.1(A9)
21 P2.0(A8)

8051 Block Diagram

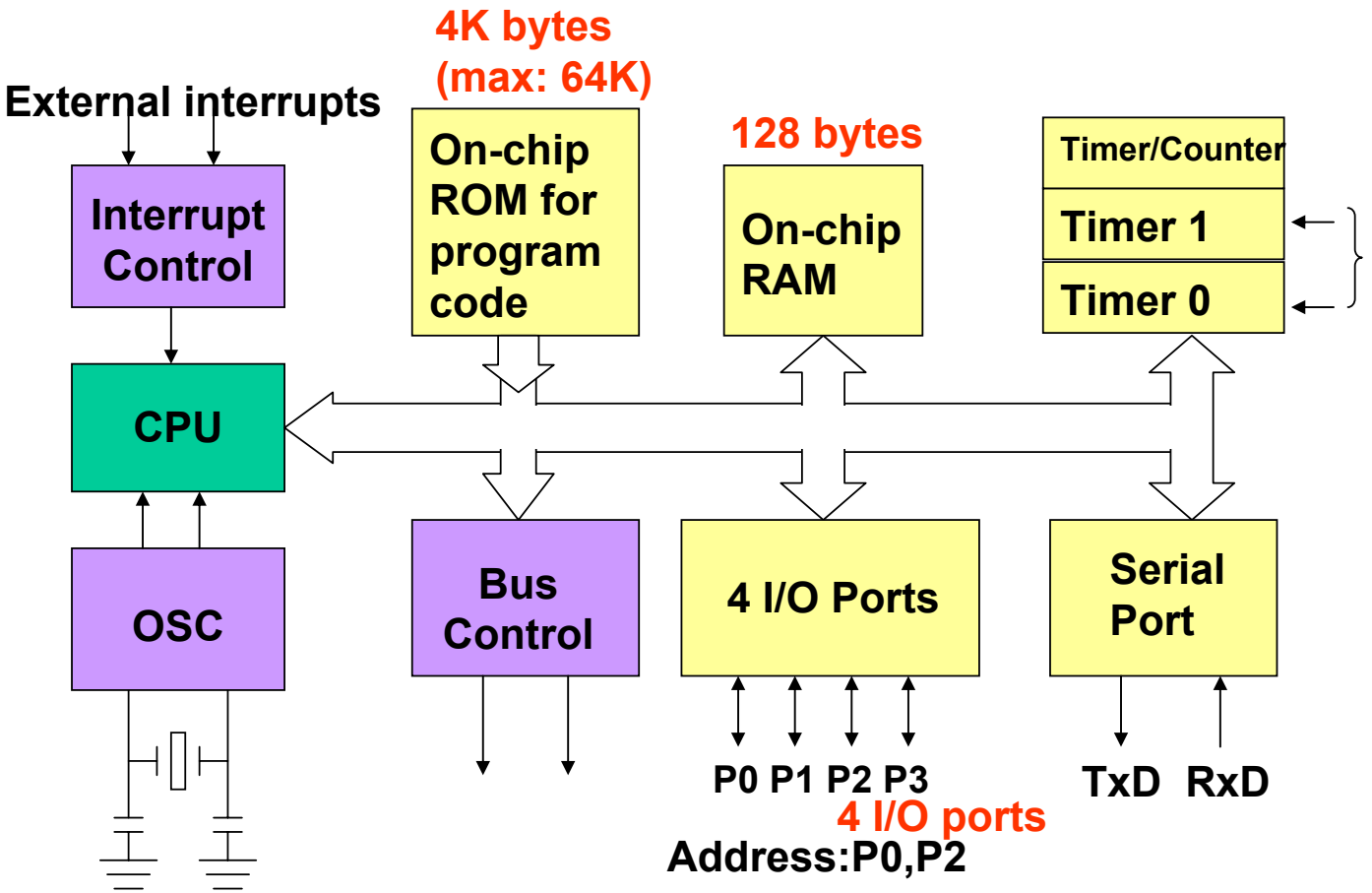


Figure 1-2. Inside the 8051 Microcontroller Block Diagram

Type	*Volatile?*	*Writeable?*	*Erase Size*	*Max Erase Cycles*	*Cost (per Byte)*	*Speed*
SRAM	Yes	Yes	Byte	Unlimited	Expensive	Fast
DRAM	Yes	Yes	Byte	Unlimited	Moderate	Moderate
Masked ROM	No	No	n/a	n/a	Inexpensive	Fast
PROM	No	Once, with a device programmer	n/a	n/a	Moderate	Fast
EPROM	No	Yes, with a device programmer	Entire Chip	Limited (consult datasheet)	Moderate	Fast
EEPROM	No	Yes	Byte	Limited (consult datasheet)	Expensive	Fast to read, slow to erase/write
Flash	No	Yes	Sector	Limited (consult datasheet)	Moderate	Fast to read, slow to erase/write
NVRAM	No	Yes	Byte	Unlimited	Expensive (SRAM + battery)	Fast

Different 8051 Products

- Distinguish by types of ROM:
- 8751 microcontroller — 4k bytes UV-EPROM, PROM burner, UV-EPROM eraser
- AT89C51 from Atmel Corporation — flash memory, PROM burner only
- DS89C4x0 from Dallas Semiconductor — flash memory, r/w from/to COM port
- DS5000 from Dallas Semiconductor — NV-RAM, r/w from/to PC serial port
- OTP (one-time-programmable) version of the 8051 — for large market
- 8051 family from Philips
- Note
  - Memory is the biggest difference between them; see Chapters 14 and Chapter 15.

DS89C420/430/…

Part Number	ROM	RAM	I/O pins	Timers	Interrupts	V_{CC}
DS89C420/30	16K (Flash)	256	32	3	6	5V
DS89C440	32K (Flash)	256	32	3	6	5V
DS89C450	64K (Flash)	256	32	3	6	5V
DS5000	8K (NVRAM)	128	32	2	6	5V
DS80C320	0 K	256	32	3	6	5V
DS87520	16K (UVROM)	256	32	3	6	5V

Source: www.maxim-ic.com/products/microcontrollers/8051_drop_in.cfm

In the lab experiments, we'll use the MDE8051 trainer which uses the Dallas (now part of Maxim) DS89C420/430.

Atmel's products

Part Number	ROM	RAM	I/O pins	Timer	Interrupt	V_{CC}	Packaging
AT89C51	4K	128	32	2	6	5V	40
AT89LV51	4K	128	32	2	6	3V	40
AT89C1051	1K	64	15	1	3	3V	20
AT89C2051	2K	128	15	2	6	3V	20
AT89C52	8K	128	32	3	8	5V	40
AT89LV52	8K	128	32	3	8	3V	40

Note: "C" in the part number indicates CMOS.

Part Number	Speed	Pins	Packaging	Use
AT89C51-12PC	12 MHz	40	DIP plastic	commercial
AT89C51-16PC	16 MHz	40	DIP plastic	commercial
AT89C51-20PC	20 MHz	40	DIP plastic	commercial