MICROCONTROLLER & DIGITAL SIGNAL PROCESSING (AEC022)

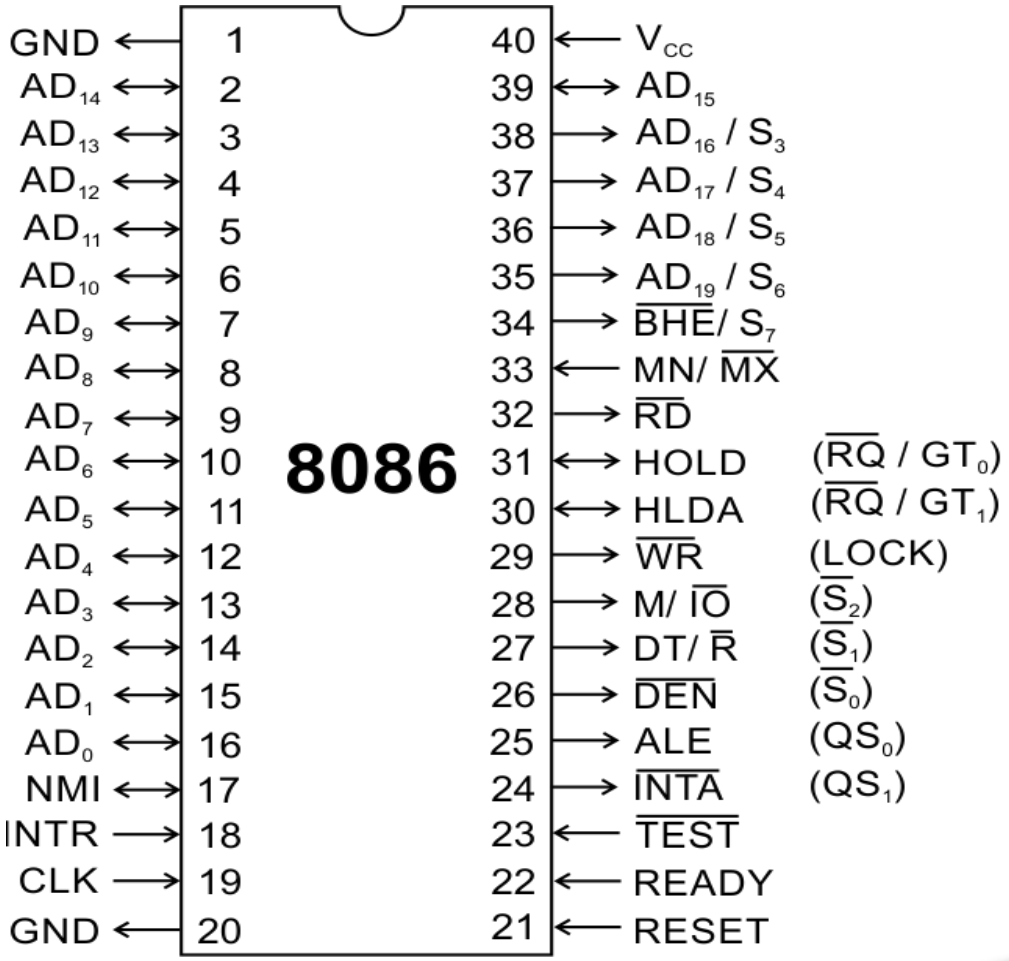
VI SEMESTER

ELECTRICAL AND ELECTRONICS ENGINEERING

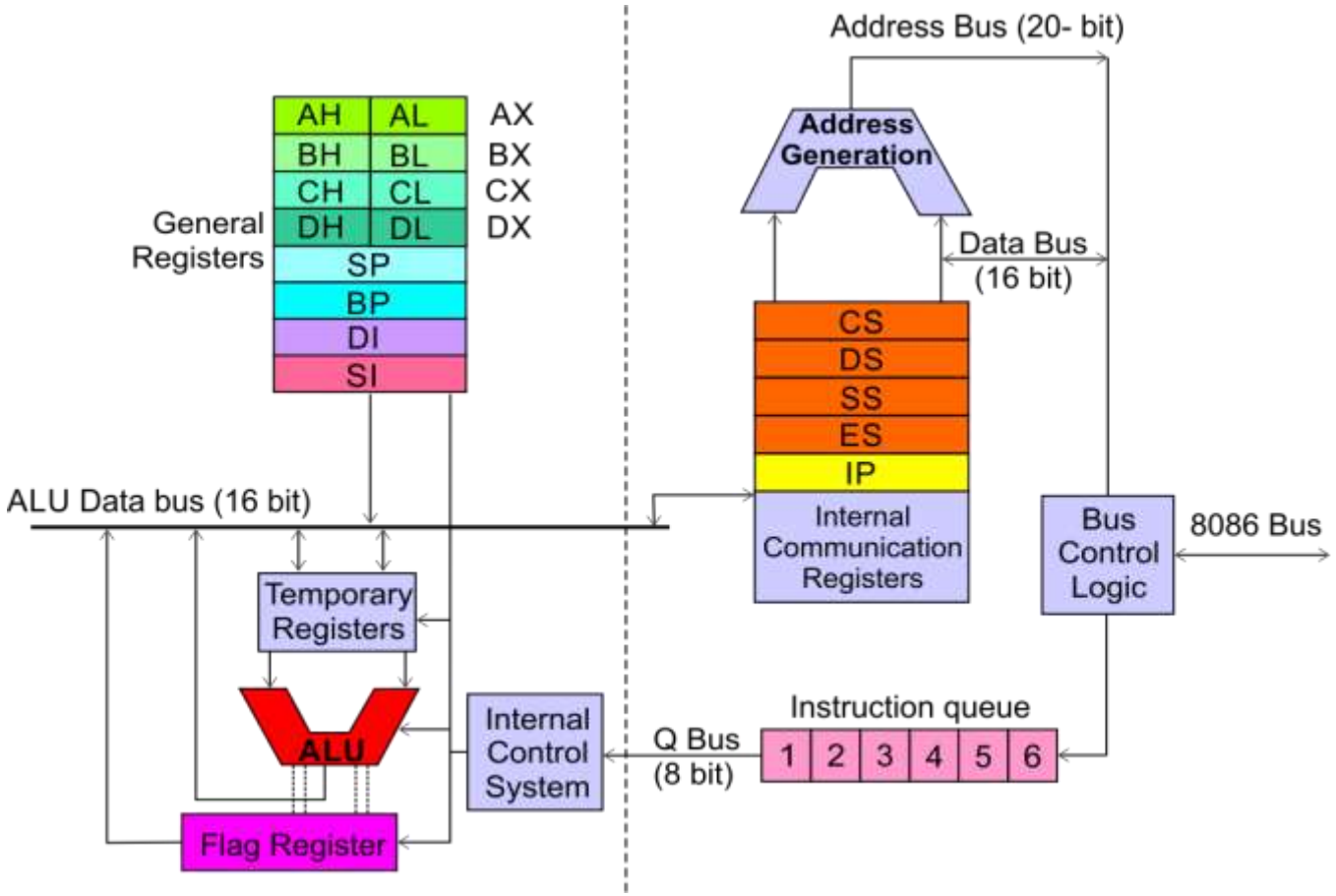
PREPARED BY
J.SRAVANA

Unit-I
MICROPROCESSORS AND MICROCONTROLLER

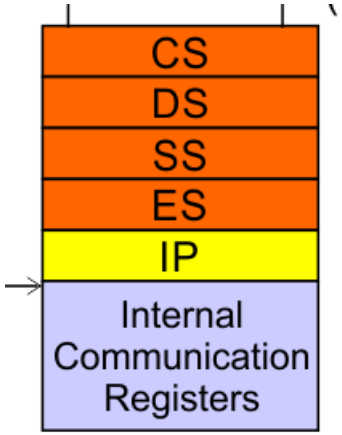
PIN DIAGRAM OF 8086

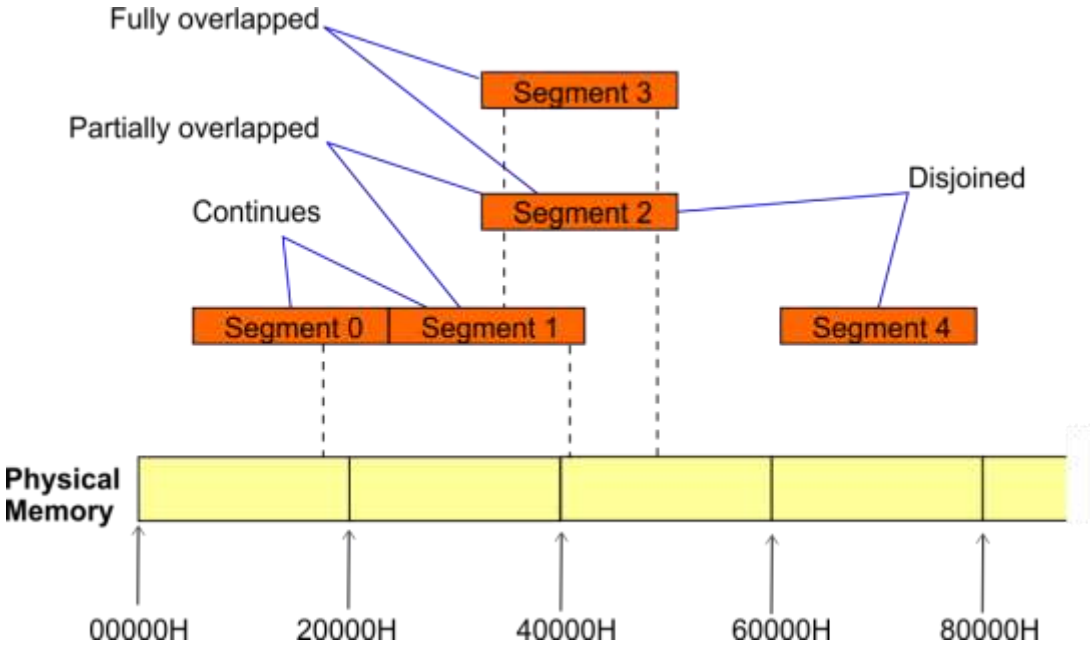


ARCHITECTURE



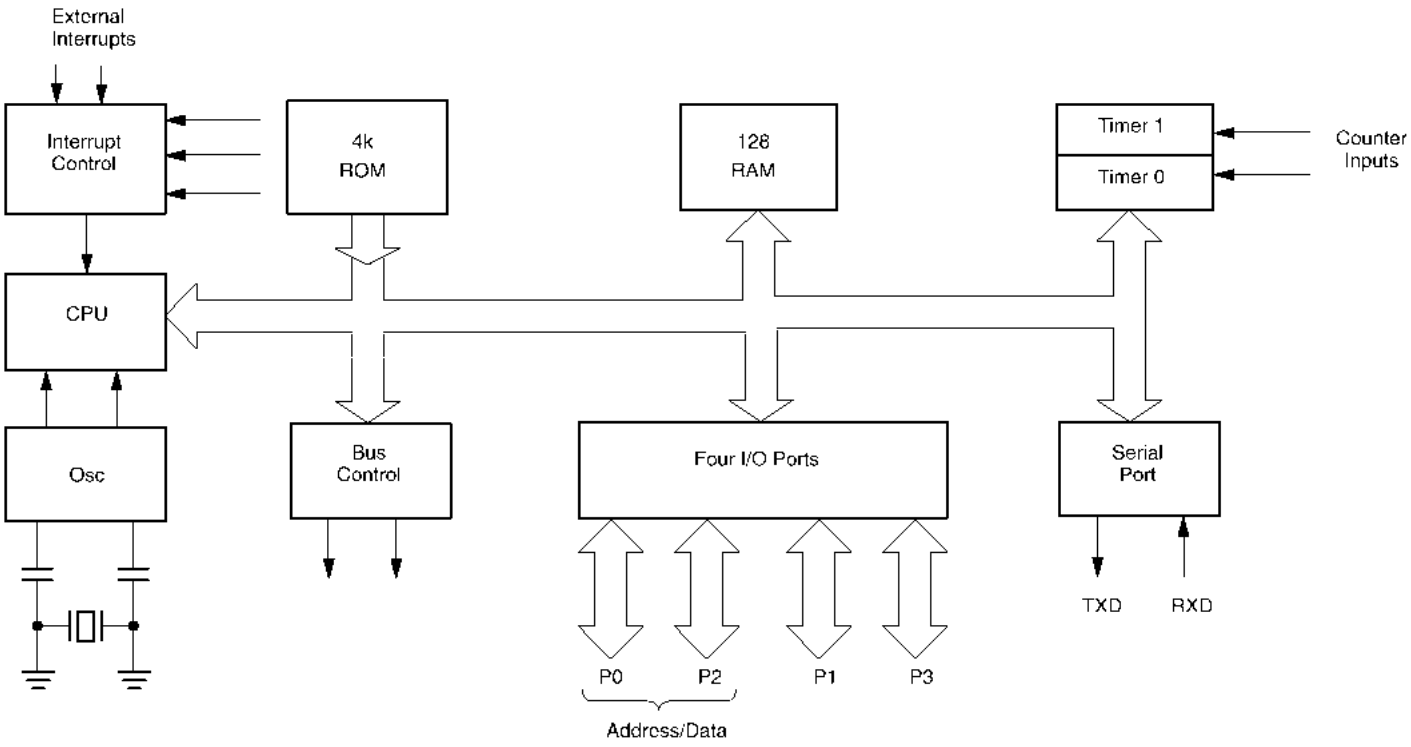
BUS INTERFACE UNIT (BIU)



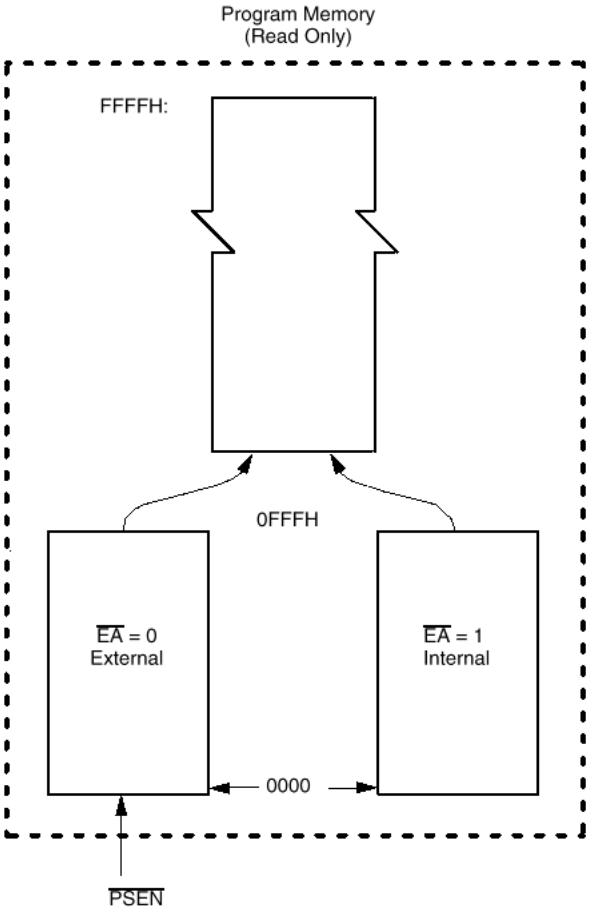


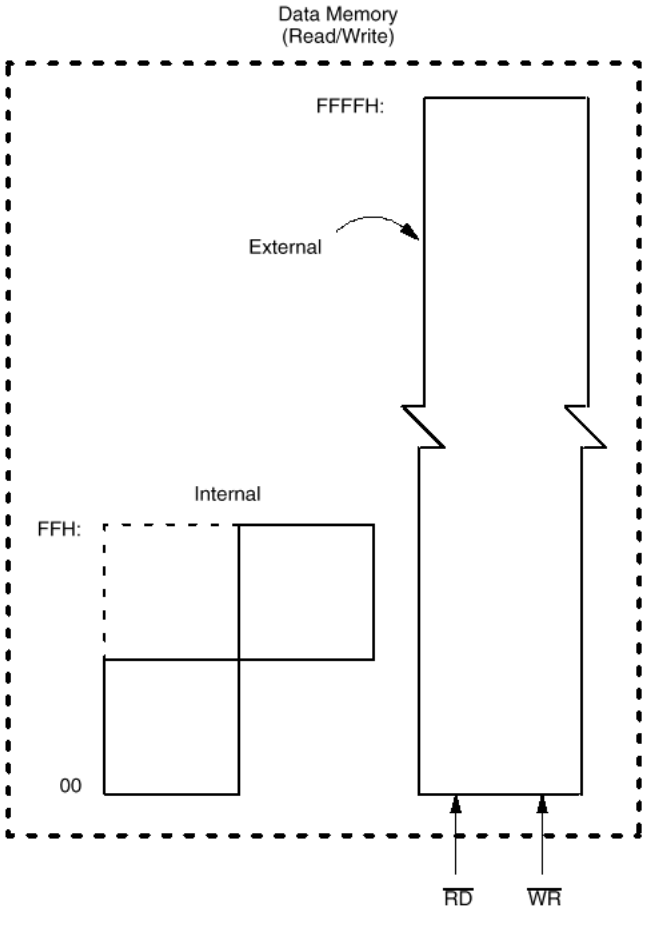
- 8086's 1-megabyte memory is divided into segments of up to 64K bytes each.
- The 8086 can directly address four segments (256 K bytes within the 1 M byte of memory) at a particular time.
- Programs obtain access to code and data in the segments by changing the segment register content to point to the desired segments.

80C51 BLOCK DIAGRAM



80C51 MEMORY





8051 MEMORY

The data width is 8 bits
Registers are 8 bits
Addresses are 8 bits
    i.e. addresses for only 256 bytes!
    PC is 16 bits (up to 64K program memory)
    DPTR is 16 bits (for external data - up to 64K)
C types
    char - 8 bits <-- use this if at all possible!
    short - 16 bits
    int - 16 bits
    long - 32 bits
    float - 32 bits
C standard **`signed/unsigned`**

ACCESSING EXTERNAL MEMORY

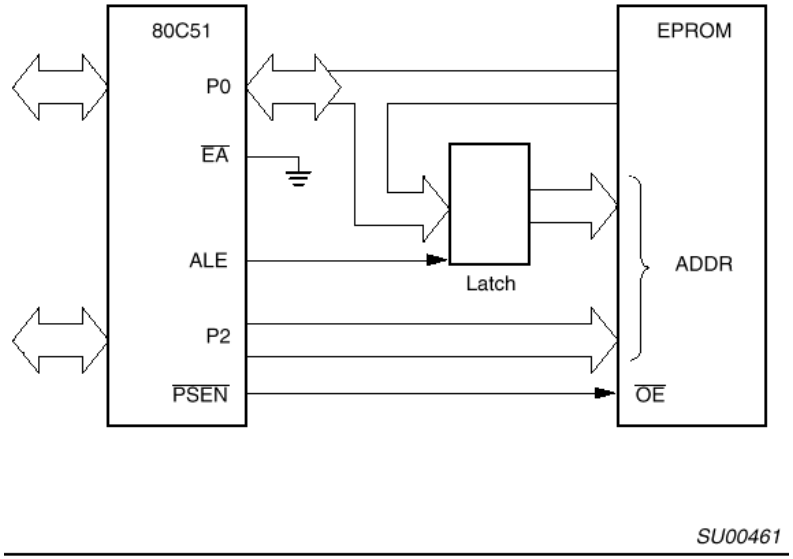


Figure 4. Executing from External Program Memory

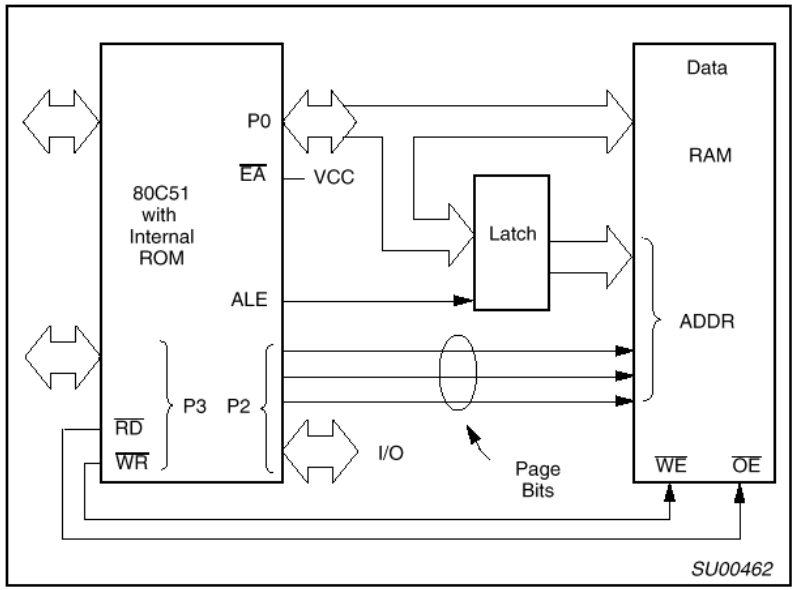


Figure 5. Accessing External Data Memory

PROGRAM MEMORY

Program and Data memory are separate

Can be internal and/or external

- 20K internal flash for the Atmel controller

Read-only

- Instructions
- Constant data

```
char code table[5] =
{'1','2','3','4','5'} ;
```

- Compiler uses instructions for moving “immediate” data

INTERNAL DATA MEMORY

Internal data memory contains all the processor state

- Lower 128 bytes: registers, general data
- Upper 128 bytes:
  - indirectly addressed: 128 bytes, used for the stack (small!)
  - directly addressed: 128 bytes for “special” functions

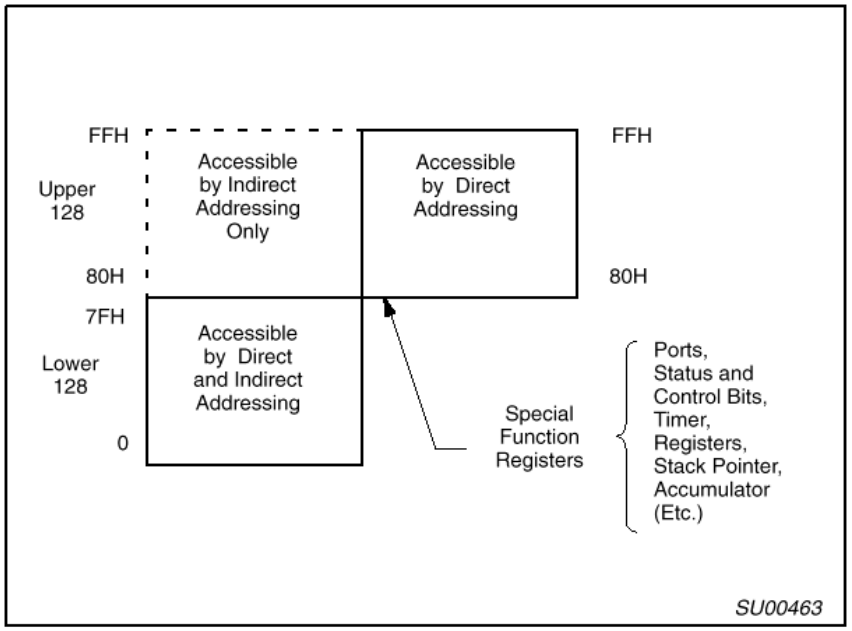


Figure 6. Internal Data Memory

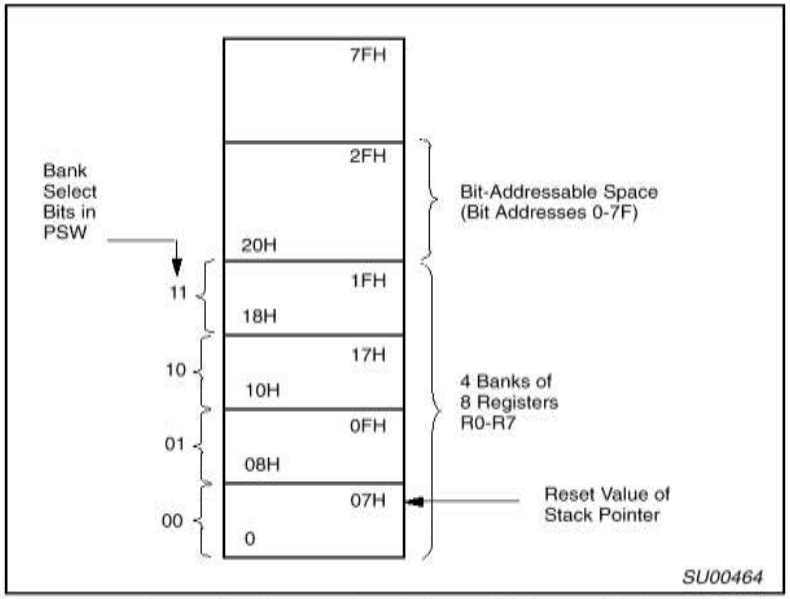


Figure 7. Lower 128 Bytes of Internal RAM

LOWER 128 BYTES

Register banks, bit addressable data, general data

you can address any register!

let the C compiler deal with details (for now)

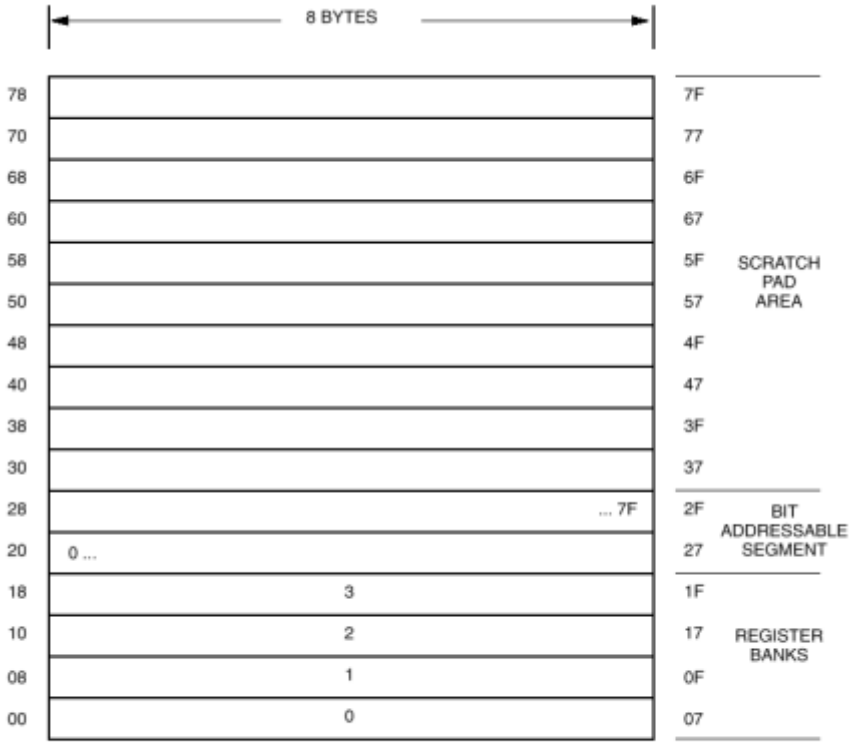


Figure 3. 128 Bytes of RAM Direct and Indirect Addressable

UPPER 128 BYTES: SFR AREA

Table 1. AT89LV55 SFR Map and Reset Values

0F8H								0FFH
0F0H	B 00000000							0F7H
0E8H								0EFH
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000							0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH
0C0H								0C7H
0B8H	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0A8H	IE 0X000000							0AFH
0A0H	P2 11111111							0A7H
98H	SCON 00000000	SBUF XXXXXXXX						9FH
90H	P1 11111111							97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000		8FH
80H	P0 11111111	SP 00000111	DPL 00000000	DPH 00000000			PCON 0XXX0000	87H

SPECIAL FUNCTION RESGISTERS

Table 1. 80C51 Special Function Registers

SYMBOL	DESCRIPTION	DIRECT ADDRESS	BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION MSB							LSB	RESET VALUE
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
DPTR	Data pointer (2 bytes)										
DPH	Data pointer high	83H									00H
DPL	Data pointer low	82H									00H
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*	Interrupt enable	A8H	EA	–	–	ES	ET1	EX1	ET0	EX0	0x000000B
			BF	BE	BD	BC	BB	BA	B9	B8	
IP*	Interrupt priority	B8H	–	–	–	PS	PT1	PX1	PT0	PX0	xx000000B
			87	86	85	84	83	82	81	80	
P0*	Port 0	80H	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	FFH
			97	96	95	94	93	92	91	90	
P1*	Port 1	90H	–	–	–	–	–	–	T2EX	T2	FFH
			A7	A6	A5	A4	A3	A2	A1	A0	
P2*	Port 2	A0H	A15	A14	A13	A12	A11	A10	A9	A8	FFH
			B7	B6	B5	B4	B3	B2	B1	B0	
P3*	Port 3	B0H	$\overline{RD}$	$\overline{WR}$	T1	T0	$\overline{INT1}$	$\overline{INT0}$	TxD	Rxd	FFH
PCON†	Power control	87H	SMOD	–	–	–	GF1	GF0	PD	IDL	0xxxxxxxB
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	–	P	00H
SBUF	Serial data buffer	99H									xxxxxxxxB
			9F	9E	9D	9C	9B	9A	99	98	
SCON*	Serial controller	98H	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	00H
SP	Stack pointer	81H									07H
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
TH0	Timer high 0	8CH									00H
TH1	Timer high 1	8DH									00H
TL0	Timer low 0	8AH									00H
TL1	Timer low 1	8BH									00H
TMOD	Timer mode	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00H

PORTS

Port 0 - external memory access
low address byte/data
Port 2 - external memory access
high address byte
Port 1 - general purpose I/O
pins 0, 1 for timer/counter 2
Port 3 - Special features
0 - RxD: serial input
1 - TxD: serial output
2 - INT0: external interrupt
3 - INT1: external interrupt
4 - T0: timer/counter 0 external input
5 - T1: timer/counter 1 external input
6 - WR: external data memory write strobe
7 - RD: external data memory read strobe

PORTS

- Port 0 - true bi-directional

Port 1-3 - have internal pullups that will source current

Output pins:

Just write 0/1 to the bit/byte

Input pins:

Output latch must have a 1 (reset state)

Turns off the pulldown

pullup must be pulled down by external driver

Just read the bit/byte

PROGRAM STATUS WORD

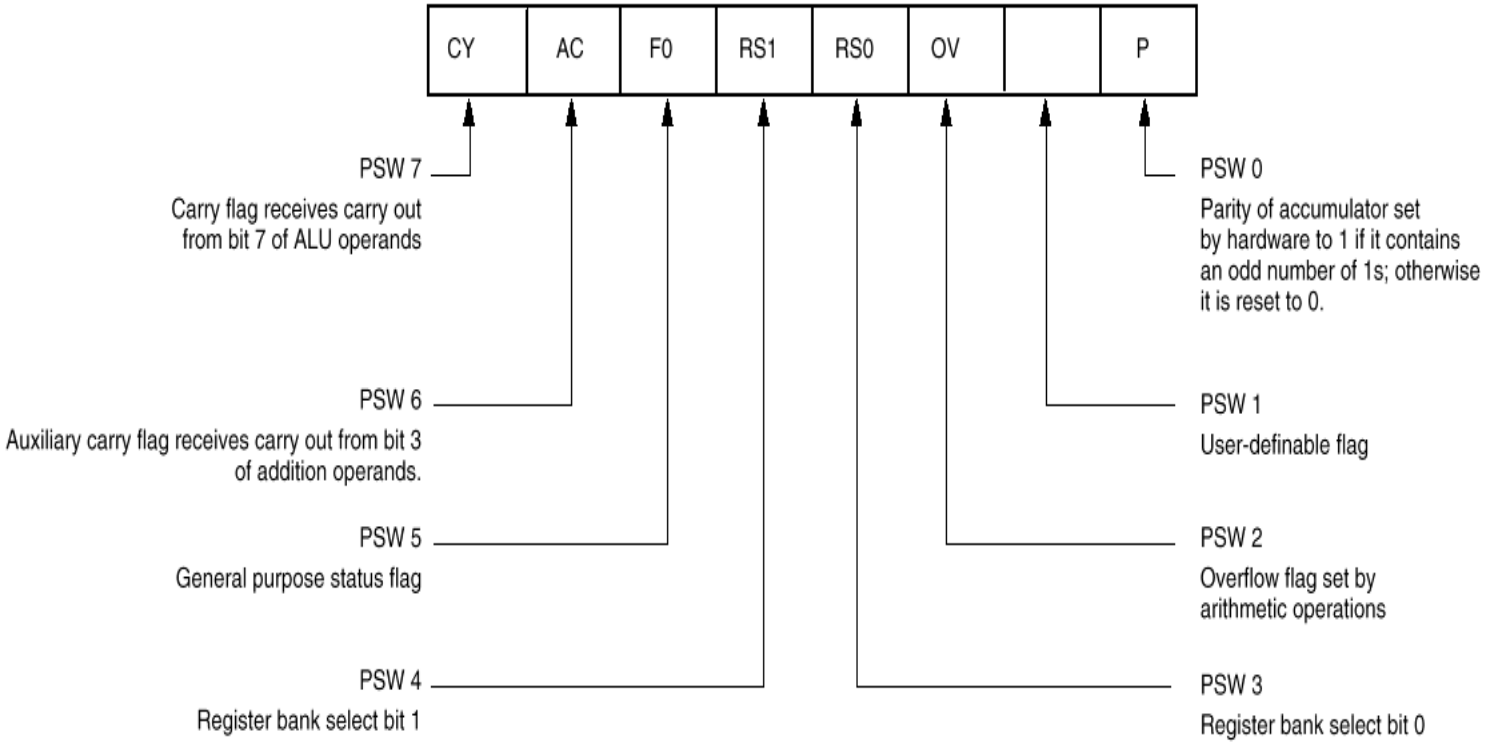


Figure 10. PSW (Program Status Word) Register in 80C51 Devices

TIMERS

Base 8051 has 2 timers
- we have 3 in the Atmel 89C55

Timer mode
- Increments every machine cycle (12 clock cycles)

Counter mode
- Increments when T0/T1 go from 1 - 0 (external signal)

Access timer value directly

Timer can cause an interrupt

Timer 1 can be used to provide programmable baud rate for serial communications

Timer/Counter operation
- Mode control register (TMOD)
- Control register (TCON)

MODE CONTROL REGISTER (TMOD)

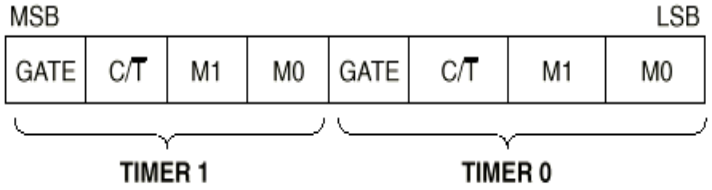
Modes 0-3

GATE - allows external pin to enable timer (e.g. external pulse)

0: INT pin not used

1: counter enabled by INT pin (port 3.2, 3.3)

C/T - indicates timer



	GATE	Gating control when set. Timer/Counter "x" is enabled only while "INTx" pin is high and "TRx" control pin is set. when cleared Timer "x" is enabled whenever "TRx" control bit is set.
	C/T	Timer or Counter Selector cleared for Timer operation (input from in=ternal system clock.) Set for Counter operation (input from "Tx" input pin).

M1	M0	OPERATING
0	0	8048 Timer "TLx" serves as 5-bit prescaler.
0	1	16-bit Timer/Counter "THx" and "TLx" are cascaded; there is no prescaler.
1	0	8-bit auto-reload Timer/Counter "THx" holds a value which is to be reloaded into "TLx" each time it overflows.
1	1	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only controlled by Timer 1 control bits.
1	1	(Timer 1) Timer/Counter 1 stopped.

SU00534

Figure 6. Timer/Counter Mode Control (TMOD) Register

INTERRUPTS

Allow parallel tasking
- Interrupt routine runs in “background”

Allow fast, low-overhead interaction with environment
- Don’t have to poll
- Immediate reaction

An automatic function call
- Easy to program

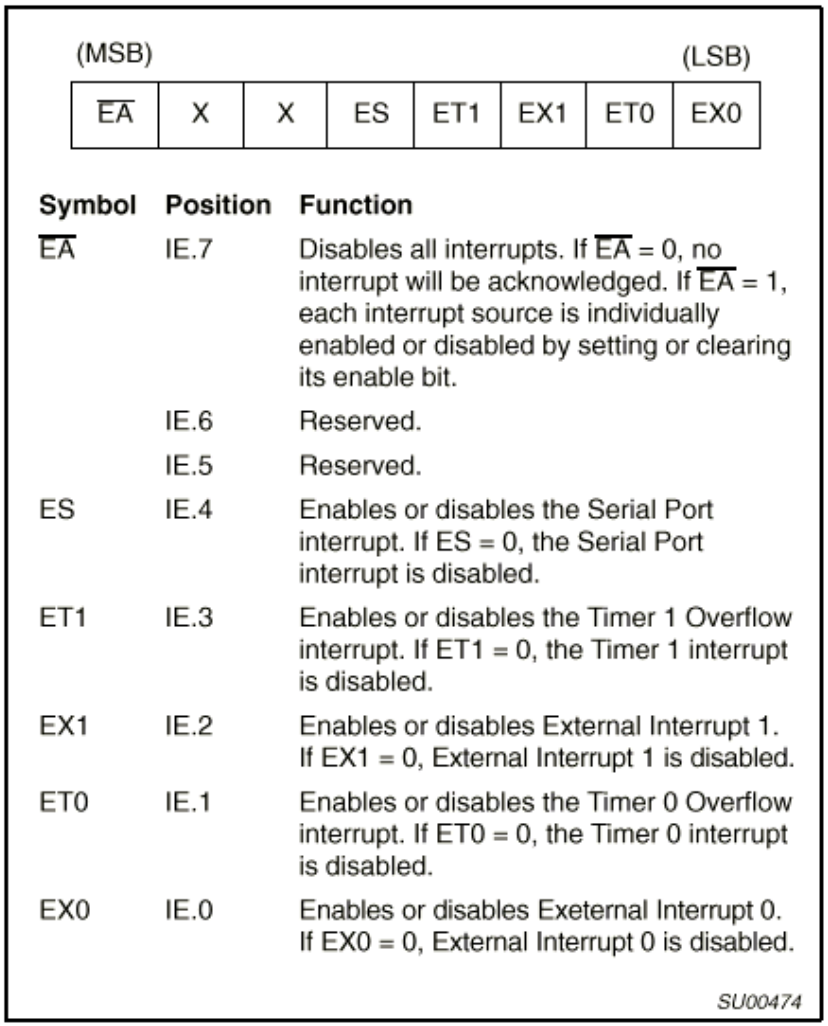
8051 Interrupts
- Serial port - wake up when data arrives/data has left
- Timer 0 overflow
- Timer 1 overflow
- External interrupt 0
- External interrupt 1

INTERRUPT PRIORITIES

Two levels of priority
- Set an interrupt priority using the interrupt priority register
- A high-priority interrupt can interrupt an low-priority interrupt routine
- In no other case is an interrupt allowed
- An interrupt routine can always disable interrupts explicitly
  - But you don't want to do this

Priority chain within priority levels
- Choose a winner if two interrupts happen simultaneously
- Order shown on previous page

CONTROLLING INTERRUPTS: ENABLES AND PRIORITY



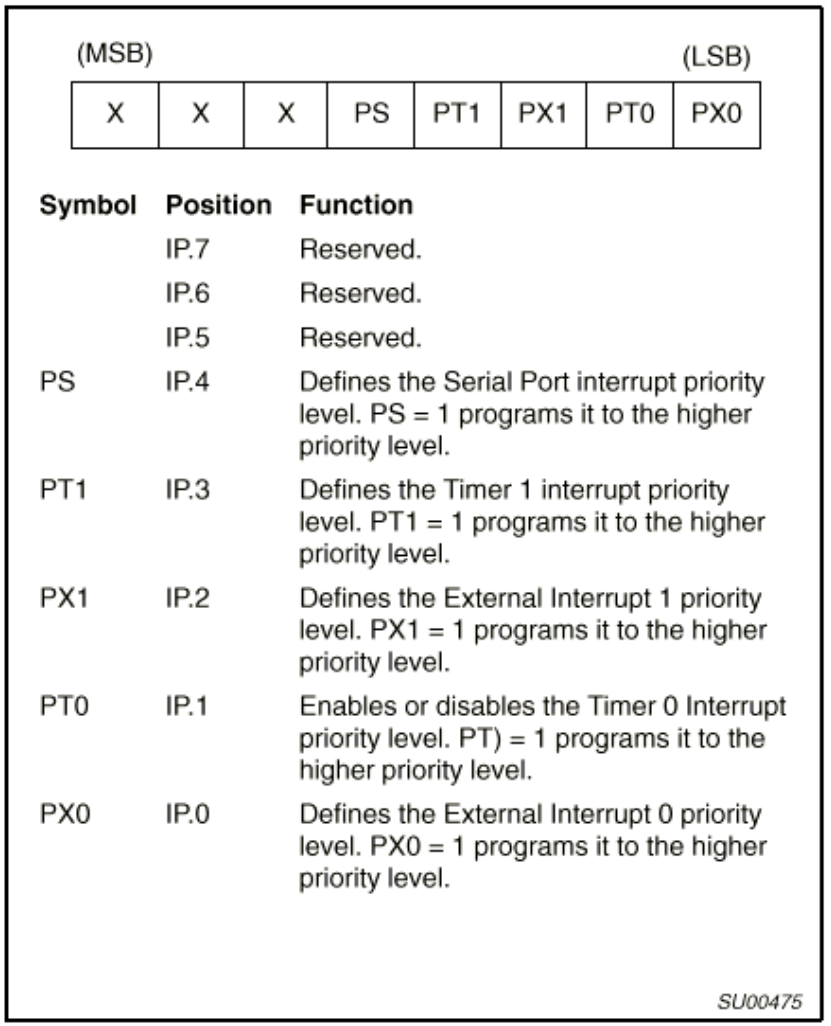
(MSB) (LSB)

$\overline{EA}$	X	X	ES	ET1	EX1	ET0	EX0

Symbol	Position	Function
$\overline{EA}$	IE.7	Disables all interrupts. If $\overline{EA}$ = 0, no interrupt will be acknowledged. If $\overline{EA}$ = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
	IE.6	Reserved.
	IE.5	Reserved.
ES	IE.4	Enables or disables the Serial Port interrupt. If ES = 0, the Serial Port interrupt is disabled.
ET1	IE.3	Enables or disables the Timer 1 Overflow interrupt. If ET1 = 0, the Timer 1 interrupt is disabled.
EX1	IE.2	Enables or disables External Interrupt 1. If EX1 = 0, External Interrupt 1 is disabled.
ET0	IE.1	Enables or disables the Timer 0 Overflow interrupt. If ET0 = 0, the Timer 0 interrupt is disabled.
EX0	IE.0	Enables or disables Exeternal Interrupt 0. If EX0 = 0, External Interrupt 0 is disabled.

SU00474

Figure 17. Interrupt Enable (IE) Register



(MSB) (LSB)

X	X	X	PS	PT1	PX1	PT0	PX0

Symbol	Position	Function
	IP.7	Reserved.
	IP.6	Reserved.
	IP.5	Reserved.
PS	IP.4	Defines the Serial Port interrupt priority level. PS = 1 programs it to the higher priority level.
PT1	IP.3	Defines the Timer 1 interrupt priority level. PT1 = 1 programs it to the higher priority level.
PX1	IP.2	Defines the External Interrupt 1 priority level. PX1 = 1 programs it to the higher priority level.
PT0	IP.1	Enables or disables the Timer 0 Interrupt priority level. PT) = 1 programs it to the higher priority level.
PX0	IP.0	Defines the External Interrupt 0 priority level. PX0 = 1 programs it to the higher priority level.

SU00475

Figure 18. Interrupt Priority (IP) Register

EXTERNAL INTERRUPTS

Can interrupt using the INT0 or INT1 pins (port 3: pin 2,3)

- Interrupt on level or falling edge of signal (TCON specifies which)
- Pin is sampled once every 12 clock cycles
  - for interrupt on edge, signal must be high 12 cycles, low 12 cycles
- Response time takes at least 3 instuctions cycles
  - 1 to sample
  - 2 for call to interrupt routine
  - more if a long instruction is in progress (up to 6 more)

Unit-II

INSTRUCTION SET AND PROGRAMMING OF 8051

ADDRESSING MODE

The CPU can access data in various ways, which are called addressing modes

- Immediate
- Register
- Direct
- Register indirect
- Indexed
  - The last three modes actually access memory

IMMEDIATE ADDRESSING MODE

The source operand is a constant

- The immediate data must be preceded by the pound sign, “#”
- Can load information into any registers, including 16-bit DPTR register

```
MOV A,#25H          ;load 25H into A
MOV R4,#62          ;load the decimal value 62 into R4
MOV B,#40H          ;load 40H into B
MOV DPTR,#4521H     ;DPTR=4512H
```

REGISTER ADDRESSING MODE

Use registers to hold the data to be manipulated

```
MOV A,R0  ;copy the contents of R0 into A
MOV R2,A  ;copy the contents of A into R2
ADD A,R5  ;add the contents of R5 to contents of A
ADD A,R7  ;add the contents of R7 to contents of A
MOV R6,A  ;save accumulator in R6
```

DIRECT ADDRESSING MODE

It is most often used the direct addressing mode to access RAM locations 30 – 7FH
The entire 128 bytes of RAM can be accessed

```
MOV R0,40H    ;save content of RAM location 40H in R0
MOV 56H,A     ;save content of A in RAM location 56H
MOV R4,7FH    ;move contents of RAM location 7FH to R4

MOV A,4          ;is same as
MOV A,R4         ;which means copy R4 into A

MOV A,7          ;is same as
MOV A,R7         ;which means copy R7 into A
```

SFR REGISTERS AND THEIR ADDRESSES

The SFR (Special Function Register) can be accessed by their names or by their addresses

The SFR registers have addresses between 80H and FFH

Not all the address space of 80 to FF is used by SFR

The unused locations 80H to FFH are reserved and must not be used by the 8051 programmer

```
MOV 0E0H,#55H  ;is the same as
MOV A,#55H     ;which means load 55H into A (A=55H)

MOV 0F0H,#25H  ;is the same as
MOV B,#25H     ;which means load 25H into B (B=25H)

MOV 0E0H,R2    ;is the same as
MOV A,R2       ;which means copy R2 into A

MOV 0F0H,R0    ;is the same as
MOV B,R0       ;which means copy R0 into B

MOV P1, A      ;is the same as
MOV 90H,A      ;which means copy reg A to P1
```

SFR REGISTERS AND THEIR ADDRESSES

Table 5-1: 8051 Special Function Register (SFR) Addresses

Symbol	Name	Address
ACC*	Accumulator	0E0H
B*	B register	0F0H
PSW*	Program status word	0D0H
SP	Stack pointer	81H
DPTR	Data pointer 2 bytes	
DPL	Low byte	82H
DPH	High byte	83H
P0*	Port 0	80H
P1*	Port 1	90H
P2*	Port 2	0A0H
P3*	Port 3	0B0H
IP*	Interrupt priority control	0B8H
IE*	Interrupt enable control	0A8H

SFR REGISTERS AND THEIR ADDRESSES

TMOD	Timer/counter mode control	89H
TCON*	Timer/counter control	88H
T2CON*	Timer/counter 2 control	0C8H
T2MOD	Timer/counter mode control	0C9H
TH0	Timer/counter 0 high byte	8CH
TL0	Timer/counter 0 low byte	8AH
TH1	Timer/counter 1 high byte	8DH
TL1	Timer/counter 1 low byte	8BH
TH2	Timer/counter 2 high byte	0CDH
TL2	Timer/counter 2 low byte	0CCH
RCAP2H	T/C 2 capture register high byte	0CBH
RCAP2L	T/C 2 capture register low byte	0CAH
SCON*	Serial control	98H
SBUF	Serial data buffer	99H
PCON	Power control	87H

* Bit-addressable

STACK AND DIRECT ADDRESSING MODE

Only direct addressing mode is allowed for pushing or popping the stack PUSH A is invalid Pushing the accumulator onto the stack must be coded as PUSH 0E0H

Example 5-2

Show the code to push R5 and A onto the stack and then pop them back them into R2 and B, where B = A and R2 = R5

Solution:

```
PUSH 05        ;push R5 onto stack
PUSH 0E0H      ;push register A onto stack
POP  0F0H      ;pop top of stack into B
               ;now register B = register A
POP  02        ;pop top of stack into R2
               ;now R2=R6
```

REGISTER INDIRECT ADDRESSING MODE

A register is used as a pointer to the data Only register R0 and R1 are used for this purpose R2 – R7 cannot be used to hold the address of an operand located in RAM When R0 and R1 hold the addresses of RAM locations, they must be preceded by the “@” sign

```
MOV A,@R0  ;move contents of RAM location whose
           ;address is held by R0 into A
MOV @R1,B  ;move contents of B into RAM location
           ;whose address is held by R1
```

INDEXED ADDRESSING MODE AND ON-CHIP ROM ACCESS

Indexed addressing mode is widely used in accessing data elements of look-up table entries located in the program ROM The instruction used for this purpose is

MOVC A,@A+DPTR

Use instruction MOVC, “C” means codeThe contents of A are added to the 16-bit register DPTR to form the 16-bit address of the needed data

I/O PORT BIT ADDRESSES

While all of the SFR registers are byte-addressable, some are also bit-addressable

The P0 – P3 are bit addressable We can access either the entire 8 bits or any single bit of I/O ports P0, P1, P2, and P3 without altering the rest

- When accessing a port in a single-bit manner, we use the syntax SETB X.Y
- X is the port number P0, P1, P2, or P3
- Y is the desired bit number from 0 to 7 for data bits D0 to D7
- ex. SETB P1.5 sets bit 5 of port 1 high

REGISTERS BIT-ADDRESSABILITY

- Only registers ACC (A), B, PSW, IP, IE, SCON, and TCON are bit-addressable While all I/O ports are bit-addressable

- In PSW register, two bits are set aside for the selection of the register banks Upon RESET, bank 0 is selected

- We can select any other banks using the bit-addressability of the PSW

CY	AC	--	RS1	RS0	OV	--	P

Unit-III

8051 MICROCONTROLLER DESIGN

8051 INTERFACING TO EXTERNAL MEMORY

- Memory Capacity

The number of bits that a semiconductor memory chip can store Called chip capacity. While the memory capacity of a memory IC chip is always given bits, the memory capacity of a computer system is given in bytes.

- 16M memory chip – 16 megabits
- A computer comes with 16M memory – 16 megabytes

Memory Organization

Memory chips are organized into a number of locations within the IC.Each location can hold 1 bit, 4 bits, 8 bits, or even 16 bits.

- A memory chip contain 2^x location, where x is the number of address pins
- Each location contains y bits, where y is the number of data pins on the chip
- The entire chip will contain $2^x \times y$ bits

PROBLEMS ON MEMORY

Example 14-1

A given memory chip has 12 address pins and 4 data pins. Find:
(a) the organization, and (b) the capacity.

Solution:

(a) This memory chip has 4096 locations ($2^{12} = 4096$), and each location can hold 4 bits of data. This gives an organization of 4096×4, often represented as 4Kx4.
(b) The capacity is equal to 16K bits since there is a total of 4K locations and each location can hold 4 bits of data.

Example 14-2

A 512K memory chip has 8 pins for data. Find:
(a) the organization, and (b) the number of address pins for this memory chip.

Solution:

(a) A memory chip with 8 data pins means that each location within the chip can hold 8 bits of data. To find the number of locations within this memory chip, divide the capacity by the number of data pins. 512K/8 = 64K; therefore, the organization for this memory chip is 64Kx8.
(b) The chip has 16 address lines since $2^{16} = 64K$.

PROBLEMS ON MEMORY

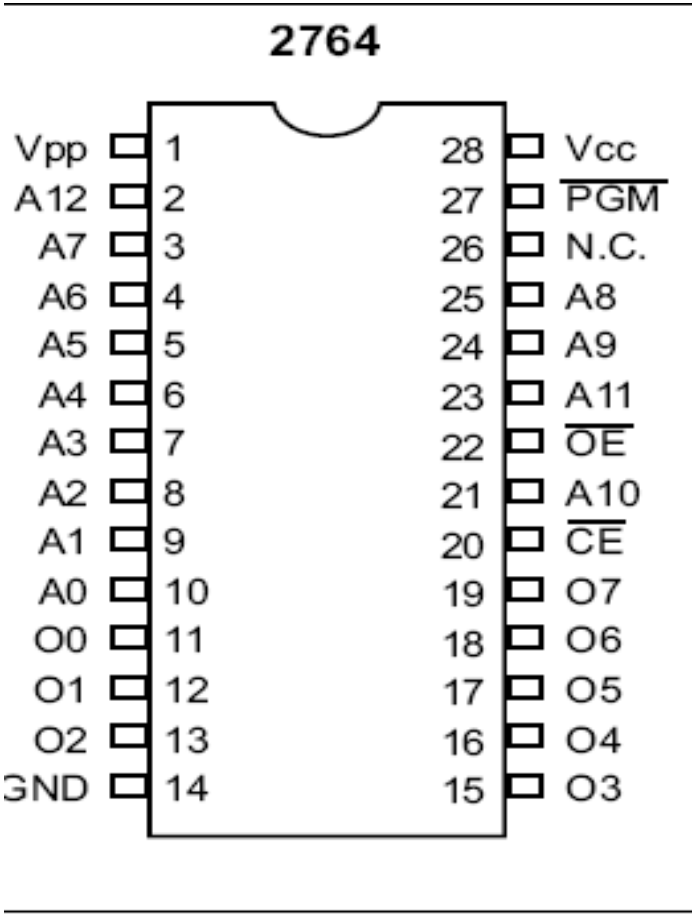
Example 14-3

For ROM chip 27128, find the number of data and address pins.

Solution:

The 27128 has a capacity of 128K bits. It has 16Kx8 organization (all ROMs have 8 data pins), which indicates that there are 8 pins for data and 14 pins for address (2^{14} = 16K).

EPROM (Erasable Programmable ROM)



SRAM (Static RAM)

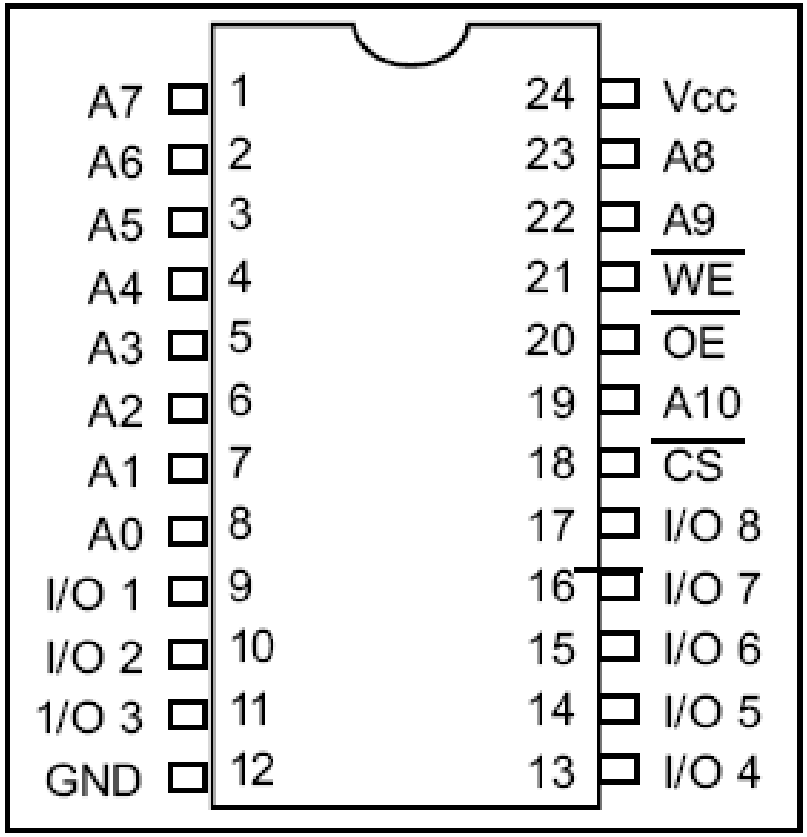


Figure 14-2. 2Kx8 SRAM Pins

DRAM (Dynamic RAM)

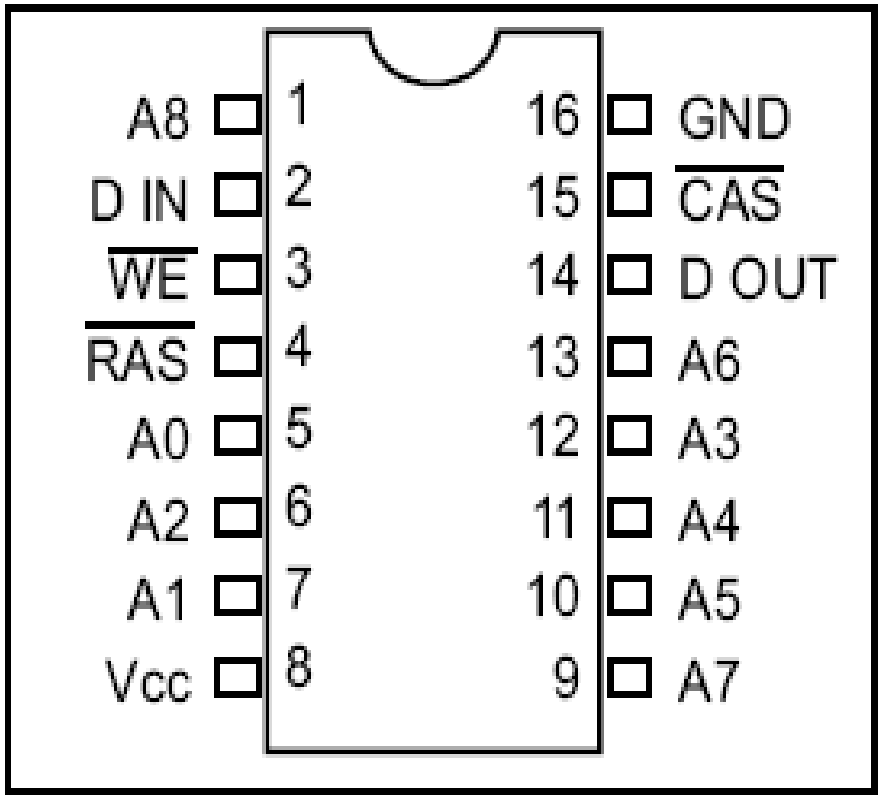


Figure 14-3. 256Kx1 DRAM

PROBLEMS ON MEMORY

Example 14-5

Discuss the number of pins set aside for addresses in each of the following memory chips. (a) 16Kx4 DRAM (b) 16Kx4 SRAM

Solution:

Since $2^{14} = 16K$:

(a) For DRAM we have 7 pins (A0 - A6) for the address pins and 2 pins for RAS and CAS.

(b) For SRAM we have 14 pins for address and no pins for RAS and CAS since they are associated only with DRAM. In both cases we have 4 pins for the data bus.

MEMORY SPACE DECODING

The CPU provides the address of the data It is the job of the decoding circuitry to locate the selected memory block Memory chips have one or more pins called CS (chip select) Must be activated for the memory's contents to be accessed Sometimes the chip select is also referred to as chip enable (CE).

In connecting a memory chip to the CPU, note the following points

- The data bus of the CPU is connected directly to the data pins of the memory chip
- Control signals RD (read) and WR (memory write) from the CPU are connected to the OE (output enable) and WE (write enable) pins of the memory chip

In the case of the address buses:

- The lower bits of the address from the CPU go directly to the memory chip address pins
- The upper ones are used to activate the CS pin of the memory chip

MEMORY SPACE DECODING

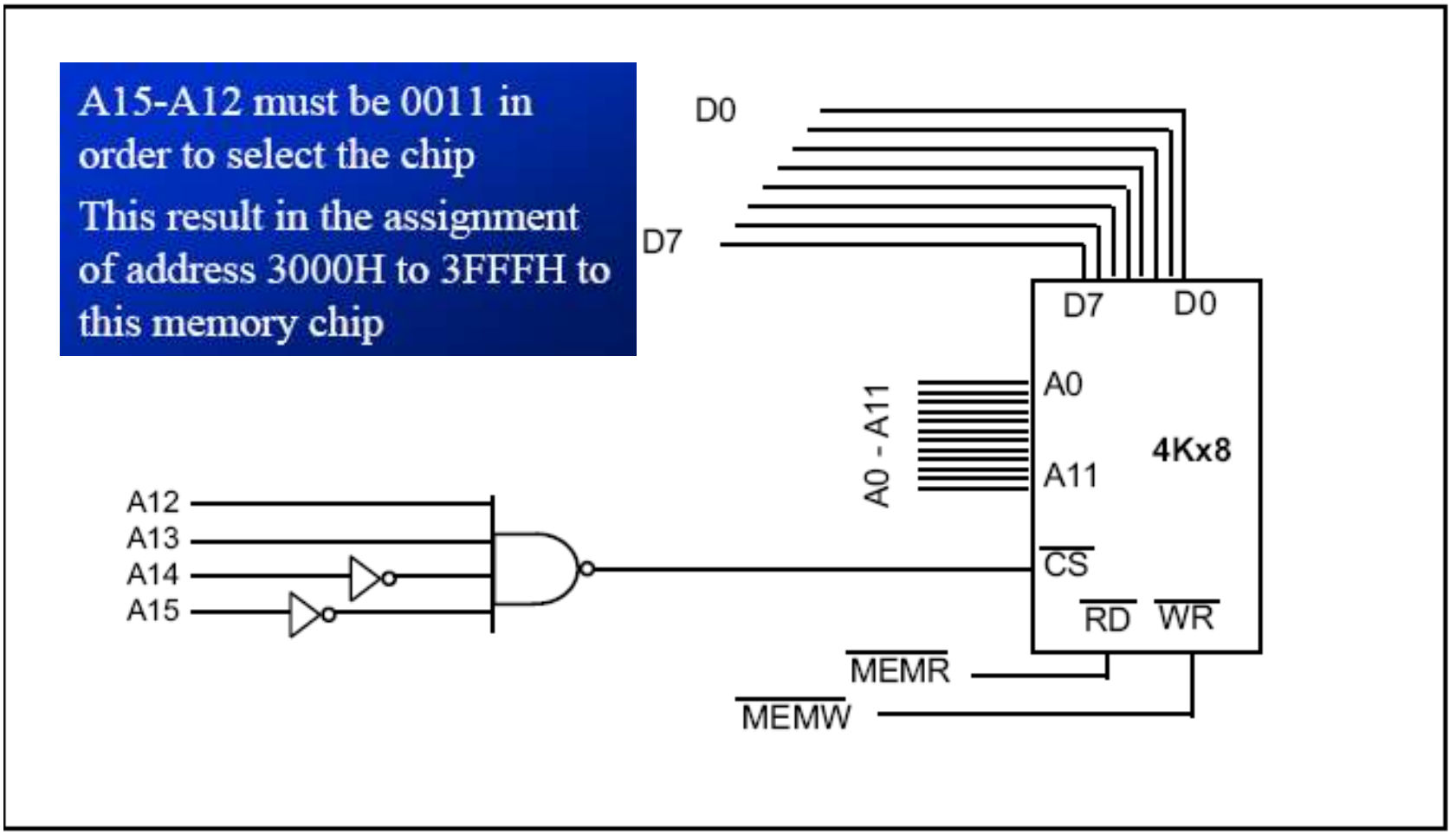


Figure 14-4. Logic Gate as Decoder

USING 74LS138 3-8 DECODER

This is one of the most widely used address decoders The 3 inputs A, B, and C generate 8 active-low outputs Y0 – Y7 Each Y output is connected to CS of a memory chip

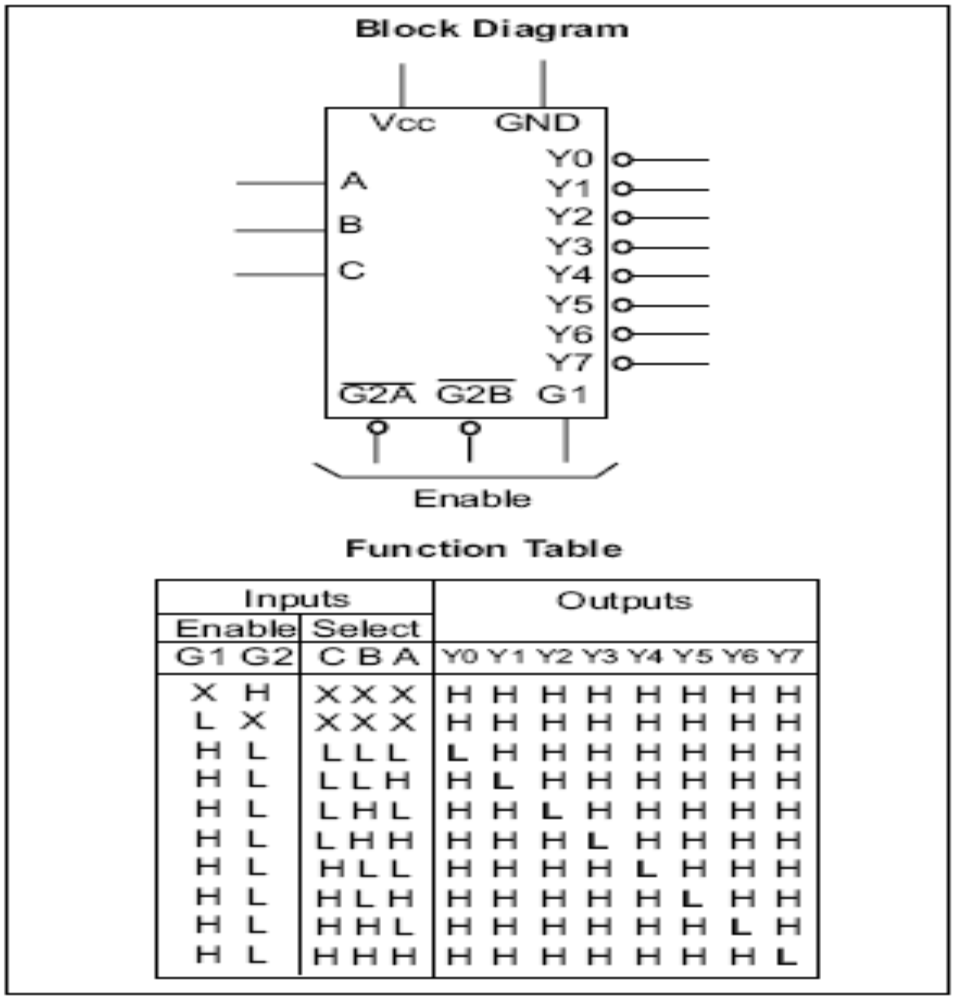
Allowing control of 8 memory blocks by a single 74LS138

- A, B, and C select which output is activate
- There are three additional inputs, G2A, G2B, and G1
- G2A and G2B are both active low, and G1 is active high

This is one of the most widely used address decoders

- The 3 inputs A, B, and C generate 8 active-low outputs Y0 – Y7
- Each Y output is connected to CS of a memory chip
- Allowing control of 8 memory blocks by a single 74LS138
- A, B, and C select which output is activated
- There are three additional inputs, G2A, G2B, and G1
- G2A and G2B are both active low, and G1 is active high

USING 74LS138 3-8 DECODER



Function Table

Inputs				Outputs							
Enable		Select									
G1	G2	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

Figure 14-5. 74LS138 Decoder

(Reprinted by permission of Texas Instruments, Copyright Texas Instruments, 1988)

PROBLEM

Example 14-6

Looking at the design in Figure 14-6, find the address range for the following.
(a) Y4, (b) Y2, and (c) Y7.

Solution:

(a) The address range for Y4 is calculated as follows.

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1

The above shows that the range for Y4 is 4000H to 4FFFH. In Figure 14-6, notice that A15 must be 0 for the decoder to be activated. Y4 will be selected when A14 A13 A12 = 100 (4 in binary). The remaining A11 - A0 will be 0 for the lowest address and 1 for the highest address.

(b) The address range for Y2 is 2000H to 2FFFH.

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1

(c) The address range for Y7 is 7000H to 7FFFH.

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

ON-CHIP AND OFF-CHIP CODE ROM

In an 8751 (89C51) system we could use on-chip ROM for boot code and an external ROM will contain the user's program

We still have EA = Vcc,

Upon reset 8051 executes the on-chip program first

When it reaches the end of the on-chip ROM, it switches to external ROM for rest of program

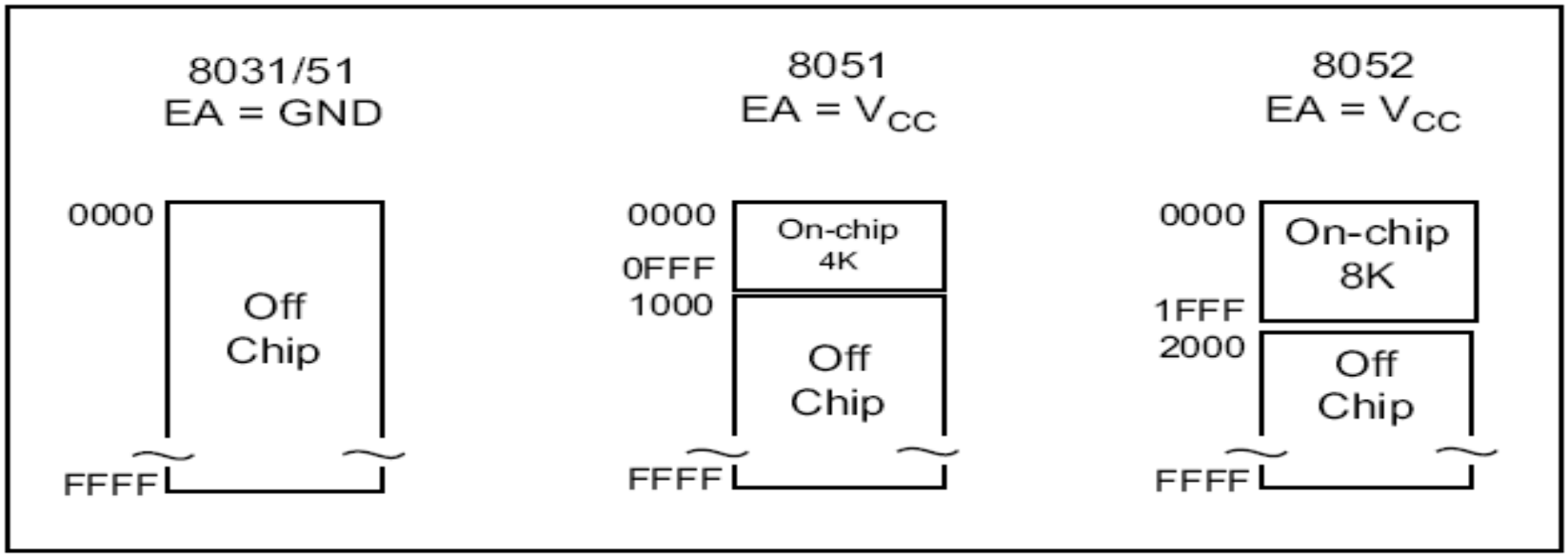


Figure 14-12. On-chip and Off-chip Program Code Access

ON-CHIP AND OFF-CHIP CODE ROM

Discuss the program ROM space allocation for each of the following cases.

(a) EA = 0 for the 8751 (89C51) chip.

(b) EA = V_{cc} with both on-chip and off-chip ROM for the 8751.

(c) EA = V_{cc} with both on-chip and off-chip ROM for the 8752.

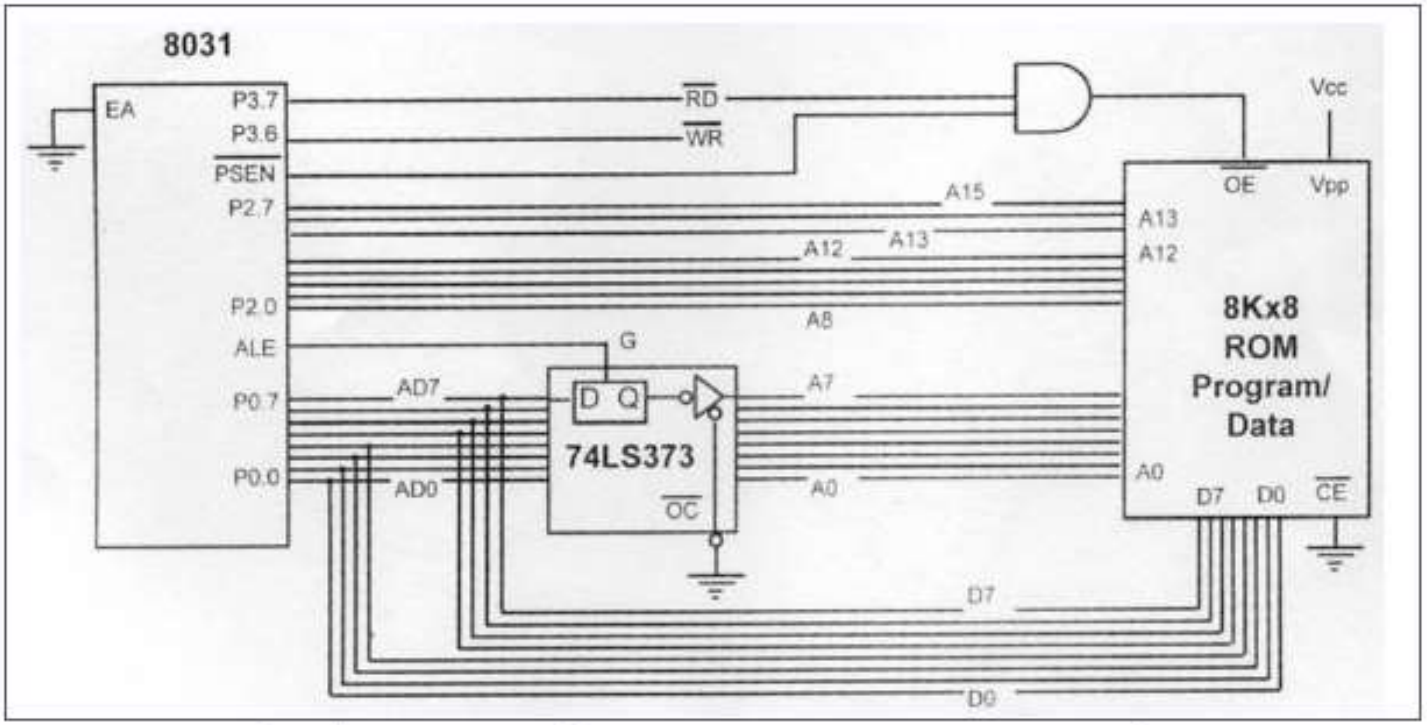
Solution:

(a) When EA = 0, the EA pin is strapped to GND, and all program fetches are directed to external memory regardless of whether or not the 8751 has some on-chip ROM for program code. This external ROM can be as high as 64K bytes with address space of 0000 – FFFFH. In this case an 8751(89C51) is the same as the 8031 system.

(b) With the 8751 (89C51) system where EA=V_{cc}, it fetches the program code of address 0000 – 0FFFH from on-chip ROM since it has 4K bytes of on-chip program ROM and any fetches from addresses 1000H – FFFFH are directed to external ROM.

(c) With the 8752 (89C52) system where EA=V_{cc}, it fetches the program code of addresses 0000 – 1FFFH from on-chip ROM since it has 8K bytes of on-chip program ROM and any fetches from addresses 2000H – FFFFH are directed to external ROM

SINGLE EXTERNAL ROM FOR CODE AND DATA



A Single ROM for BOTH Program and Data

SINGLE EXTERNAL ROM FOR CODE AND DATA

Example 14-13

Assume that we need an 8031 system with 16KB of program space, 16KB of data ROM starting at 0000, and 16K of NV-RAM starting at 8000H. Show the design using a 74LS138 for the address decoder.

Solution:

The solution is diagrammed in Figure 14-17. Notice that there is no need for a decoder for program ROM, but we need a 74LS138 decoder for data ROM and RAM. Also notice that G1 = V_{cc}, G2A = GND, G2B = GND, and the C input of the 74LS138 is also grounded since we use Y0 - Y3 only.

SINGLE EXTERNAL ROM FOR CODE AND DATA

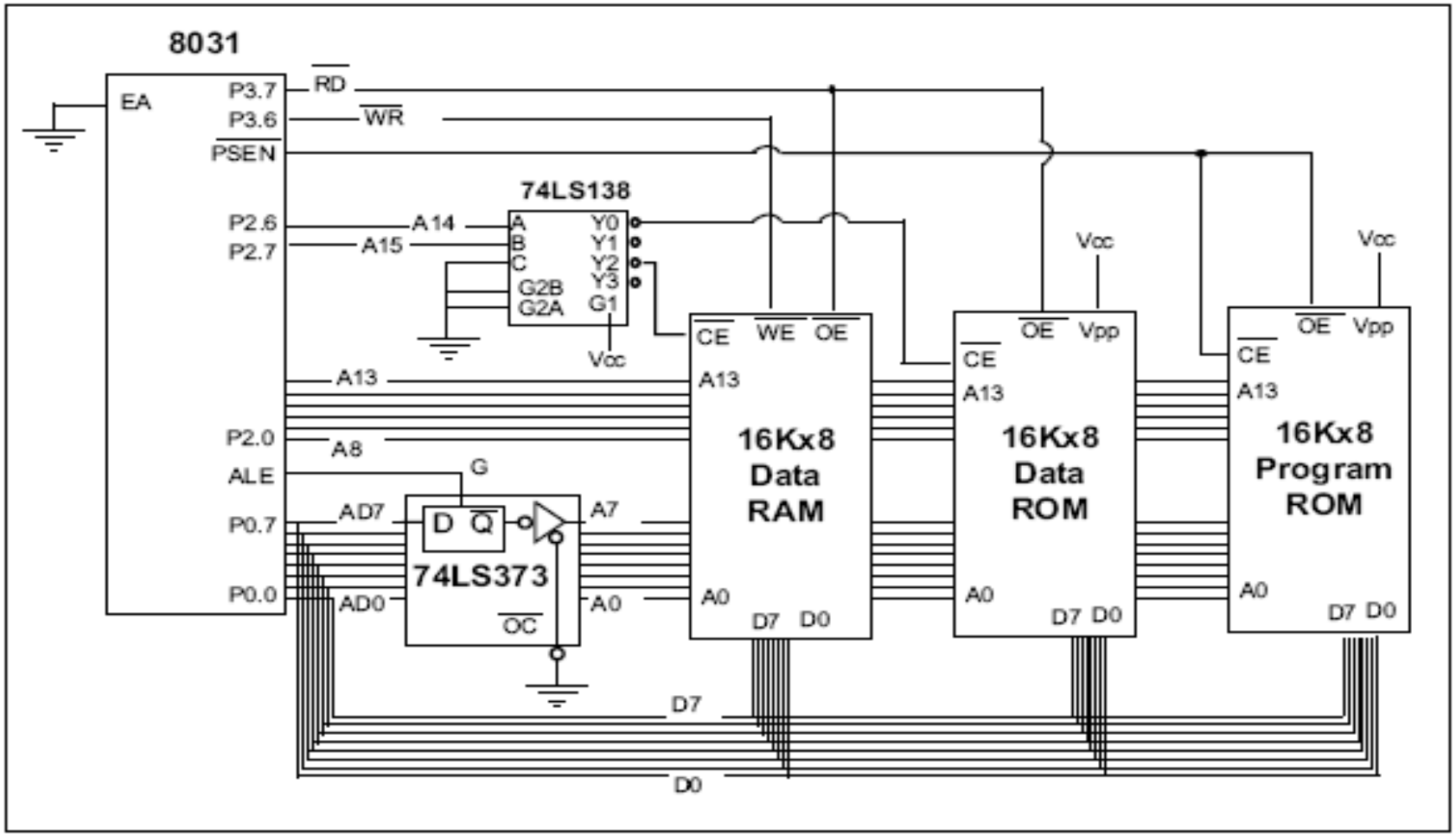


Figure 14-17. 8031 Connection to External Program ROM, Data RAM, and Data ROM

PROBLEMS

In a certain application, we need 256K bytes of NV-RAM to store data collected by an 8051 microcontroller. (a) Show the connection of an 8051 to a single 256K×8 NV-RAM chip. (b) Show how various blocks of this single chip are accessed

Solution:

(a) The 256K×8 NV-RAM has 18 address pins (A0 – A17) and 8 data lines. As shown in Figure 14-18, A0 – A15 go directly to the memory chip while A16 and A17 are controlled by P1.0 and P1.1, respectively. Also notice that chip select of external RAM is connected to P1.2 of the 8051.

(b) The 256K bytes of memory are divided into four blocks, and each block is accessed as follows :

Chip select P1.2	A17 P1.1	A16 P1.0	Block address space
0	0	0	00000H - 0FFFFH
0	0	1	10000H - 1FFFFH
0	1	0	20000H - 2FFFFH
0	1	1	30000H - 3FFFFH
1	x	x	External RAM disabled

....

PROBLEMS

....

For example, to access the 20000H – 2FFFFH address space we need the following :

```
CLR         P1.2      ;enable external RAM
MOV         DPTR,#0   ;start of 64K memory block
CLR         P1.0      ;A16 = 0
SETB        P1.1      ;A17 = 1 for 20000H block
MOV         A,SBUF    ;get data from serial port
MOVX        @DPTR,A
INC         DPTR      ;next location
...
```

Unit-IV

INTRODUCTION TO DIGITAL SIGNAL PROCESSING AND FAST FOURIER TRANSFORMS

SIGNAL PROCESSING

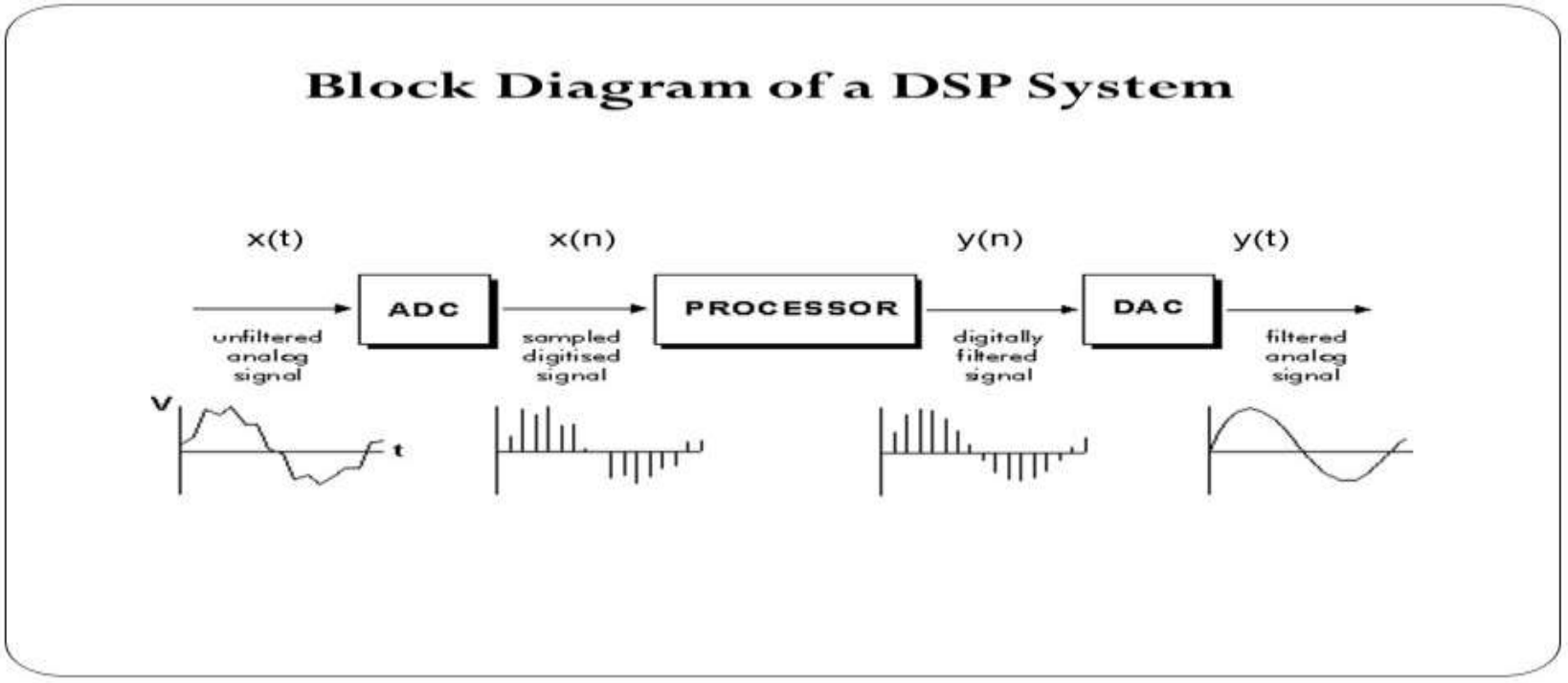
- Humans are the most advanced signal processors
  - speech and pattern recognition, speech synthesis,…
- We encounter many types of signals in various applications
  - Electrical signals: voltage, current, magnetic and electric fields,…
  - Mechanical signals: velocity, force, displacement,…
  - Acoustic signals: sound, vibration,…
  - Other signals: pressure, temperature,…
- Most real-world signals are analog
  - They are continuous in time and amplitude
  - Convert to voltage or currents using sensors and transducers
- Analog circuits process these signals using
  - Resistors, Capacitors, Inductors, Amplifiers,…
- Analog signal processing examples
  - Audio processing in FM radios
  - Video processing in traditional TV sets

LIMITATIONS OF ANALOG SIGNAL PROCESSING

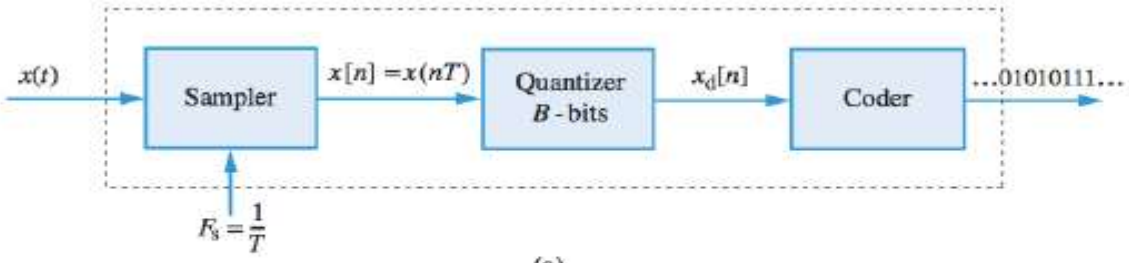
- Accuracy limitations due to
  - Component tolerances
  - Undesired nonlinearities
- Limited repeatability due to
  - Tolerances
  - Changes in environmental conditions
    - Temperature
    - Vibration
- Sensitivity to electrical noise
- Limited dynamic range for voltage and currents
- Inflexibility to changes
- Difficulty of implementing certain operations
  - Nonlinear operations
  - Time-varying operations
- Difficulty of storing information

DIGITAL SIGNAL PROCESSING

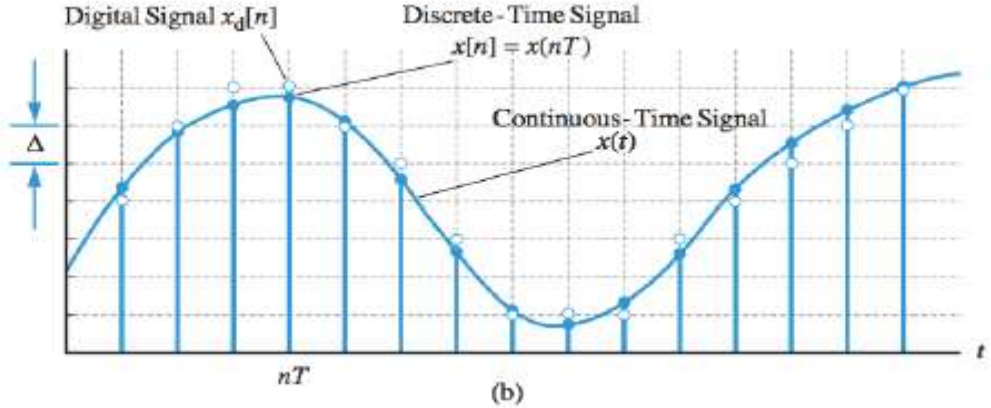
- Represent signals by a sequence of numbers
  - Sampling or analog-to-digital conversions
- Perform processing on these numbers with a digital processor
  - Digital signal processing
- Reconstruct analog signal from processed numbers
  - Reconstruction or digital-to-analog conversion



A/D CONVERTER



(a)

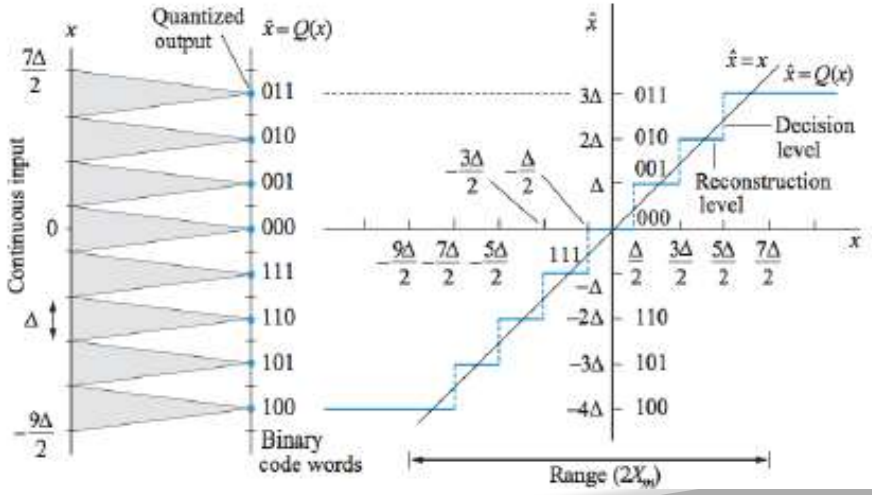


(b)

- *Quantization* converts a continuous-amplitude signal $x(t)$ into a discrete-amplitude signal $x_d[n]$.
- In theory, we are dealing with discrete-time signals; in practice, we are dealing with digital signals.

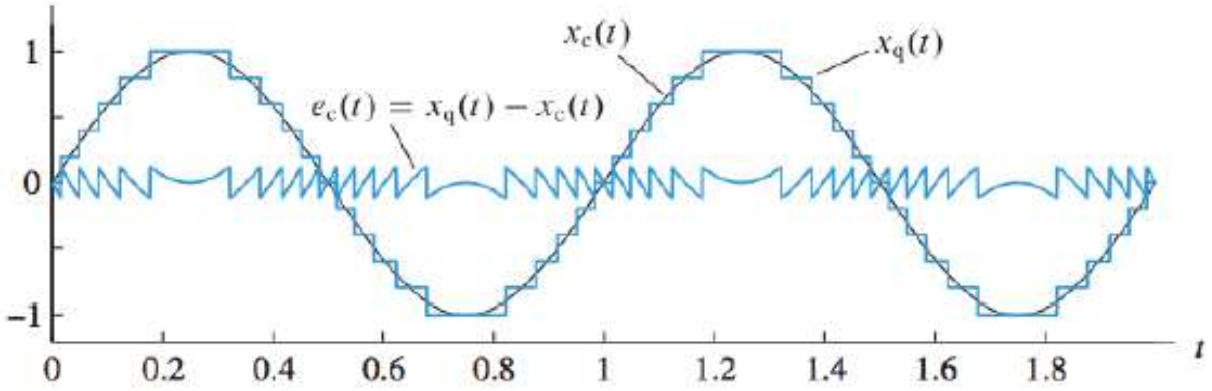
A/D CONVERTER

- The major difference between ideal and practical conversion is that an ADC generates sample values that are known with finite precision.
- The ADC is the device in which both quantization and binary coding of the sampled signal take place.
- A B-bit quantizer can represent 2^B different numbers.
- If the input amplitude range is divided into K quantization intervals of equal width Δ (*quantization step*) and the output levels are uniformly spaced, the resulting quantizer is called *uniform*.



QUANTIZATION NOISE

- The two major types of error introduced by an ADC are aliasing error and *quantization error*.



- Since quantization is a nonlinear operation, analysis of quantization error is done using statistical techniques.
- If there is a large number of small quantization intervals, the signal $x_c(t)$ can be assumed to be approximately linear between quantization levels. In this case:

$$e_c(t) \triangleq x_q(t) - x_c(t) = \frac{\Delta}{2\tau}t, \quad -\tau \le t \le \tau$$

- Then the mean squared quantization error power is

$$P_Q = \frac{1}{2\tau}\int_{-\tau}^{\tau} |e_c(t)|^2 dt = \frac{\Delta^2}{12}$$

D/A CONVERSION

- A band-limited signal can be reconstructed from a sequence of samples using the ideal DAC described by

$$x_r(t) = \sum_n x[n]\, g_{BL}(t - nT) = \sum_n x[n]\, \mathrm{sinc}(t/T - n)$$

- A system that implements the above formula, for an arbitrary function $g_r(t)$, is known as a *practical digital-to-analog converter (DAC)*.
- The function $g_r(t)$ is also known as the *characteristic pulse* of a DAC. At each sample time $t = nT$, the converter generates a pulse $g_r(t - nT)$ scaled by $x[n]$.
- In particular, the *switch-and-hold* DAC performs the following operation

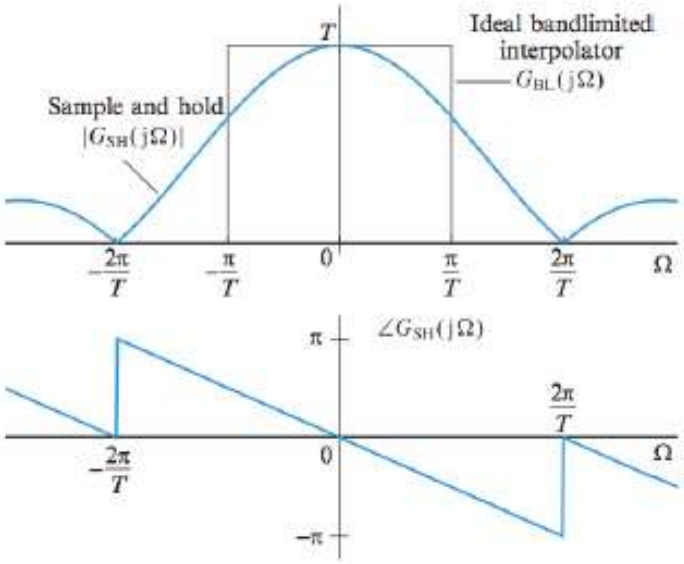
$$x_{SH}(t) = \sum_n x_q[n]\, g_{SH}(t - nT)$$

where

$$g_{SH}(t) = \begin{cases} 1, & 0 \le t \le T \\ 0, & \text{otherwise} \end{cases} \quad \Longleftrightarrow \quad G_{SH}(\jmath\Omega) = \frac{2\sin(\Omega T/2)}{\Omega} e^{-\jmath\Omega T/2}$$

D/A CONVERSION

- The S/H circuit cannot completely eliminate the spectral replicas introduced by the sampling process.
- Moreover, it introduces amplitude distortion in the Nyquist band $|F_s| < F_s/2$.



- To compensate for the effects of the S/H circuit, we can use an analog post-filter $H_r(j\Omega)$ so that $G_{SH}(j\Omega)\,H_r(j\Omega) = G_{BL}(j\Omega)$:

$$H_r(j\Omega) = \begin{cases} \dfrac{\Omega T/2}{\sin(\Omega T/2)} e^{j\Omega T/2}, & |\Omega| < \pi/T \\ 0, & \text{otherwise} \end{cases}$$

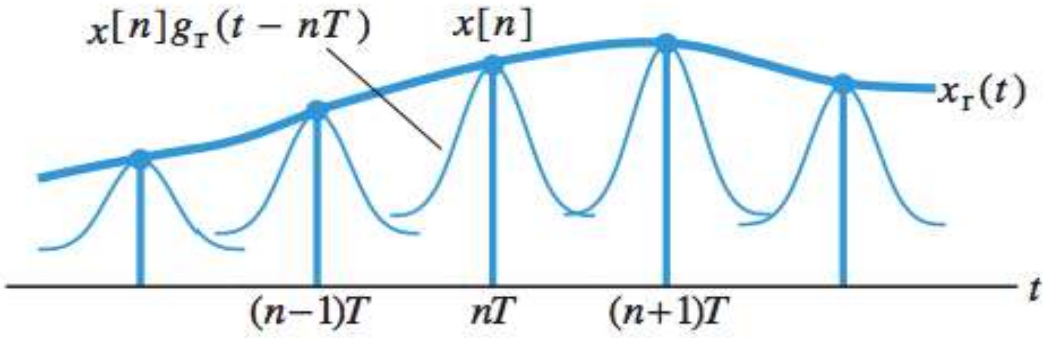
RECONSTRUCTION

- A general formula that describes a broad class of reconstruction processes is given by

$$x_r(t) = \sum_n x[n] g_r(t - nT)$$

where $g_r(t)$ is an interpolating reconstruction function.

- The process of fitting a continuous function to a set of samples is known as an *interpolation*.



- Thus, if the interpolation function has duration greater than or equal to T, the addition of the overlapping copies "fills the gaps" between samples.

RECONSTRUCTION

- Evaluating the inverse Fourier transform of $G_{BL}(j\Omega)$, we obtain

$$g_r(t) \triangleq g_{BL}(t) = \frac{\sin(\pi t/T)}{\pi t/T} = \text{sinc}(t/T)$$

- In this case we obtain:

The ideal interpolation formula

$$x_c(t) = \sum_n x[n] \frac{\sin[\pi(t - nT)/T]}{\pi(t - nT)/T}$$

- The system used to implement the ideal interpolation is known as an *ideal DAC*.

SIGNALS

Continuous-time signals are functions of a real argument

$x(t)$ where t can take any real value

$x(t)$ may be 0 for a given range of values of t

Discrete-time signals are functions of an argument that takes values from a discrete set

$x[n]$ where $n \in \{...-3,-2,-1,0,1,2,3...\}$

Integer index n instead of time t for discrete-time systems

$\boldsymbol{x}$ may be an array of values (multi channel signal)

Values for $\boldsymbol{x}$ may be real or complex

DISCRETE-TIME SIGNALS AND SYSTEMS

- *Continuous-time signals* are defined over a continuum of times and thus are represented by a continuous independent variable.
- *Discrete-time signals* are defined at discrete times and thus the independent variable has discrete values.
- *Analog signals* are those for which both time and amplitude are continuous. *Digital signals* are those for which both time and amplitude are discrete.

PERIODIC (UNIFORM) SAMPLING

Sampling is a continuous to discrete-time conversion

$$x[n] = x_c(nT) \quad -\infty < n < \infty$$

Most common sampling is periodic

T is the sampling period in second

$f_s = 1/T$ is the sampling frequency in Hz

Sampling frequency in radian-per-second $\Omega_s = 2\pi f_s$ rad/sec

Use [.] for discrete-time and (.) for continuous time signals

This is the ideal case not the practical but close enough

- In practice it is implement with an analog-to-digital converters
- We get digital signals that are quantized in amplitude and time

IMPULSE FUNCTION

•The *impulse function*, also known as Dirac's delta function, is used to represented quantities that are highly localized in space. Examples include point optical sources and electrical charges.

•The *impulse function*, also known as Dirac's delta function, is used to represented quantities that are highly localized in space. Examples include point optical sources and electrical charges.

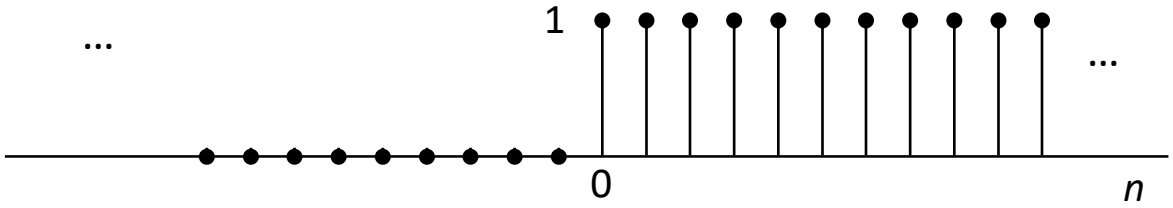
$$\delta(x - x_0) = 0, \quad x \neq x_0$$

$$\int_{x_1}^{x_2} f(x)\,\delta(x - x_0)\,dx = f(x_0), \quad x_1 < x_0 < x_2$$

UNIT STEP SEQUENCE

$u[n] = 1, n \geq 0$

$= 0, \ n < 0.$



$$u[n] = \delta[n] + \delta[n-1] + \delta[n-2] + \cdots$$

$$u[n] = \sum_{k=0}^{\infty} \delta[n-k]$$

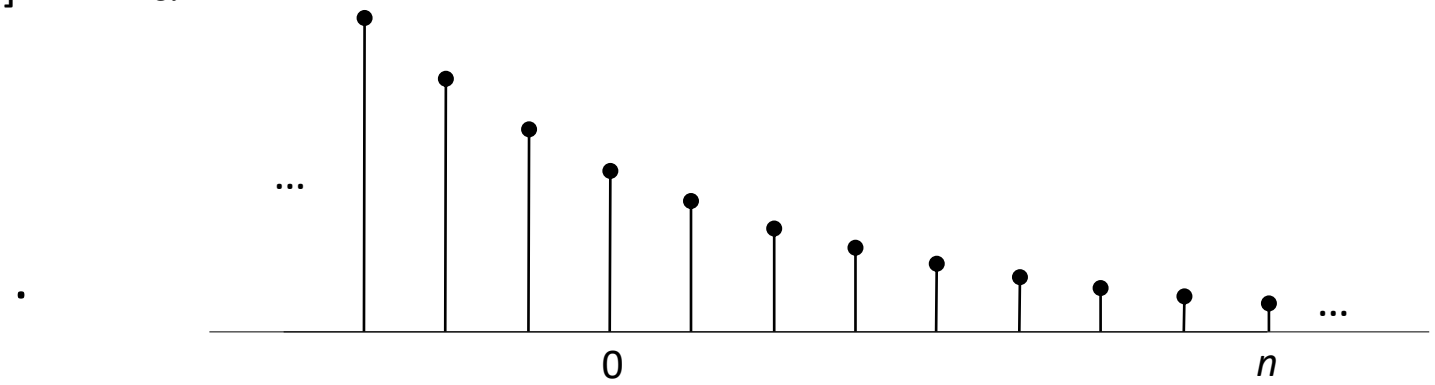
$$u[n] = \sum_{k=-\infty}^{n} \delta[k]$$

Conversely, the impulse sequence can be expressed as the first backward difference of the unit step sequence:

$$\delta[n] = u[n] - u[n-1]$$

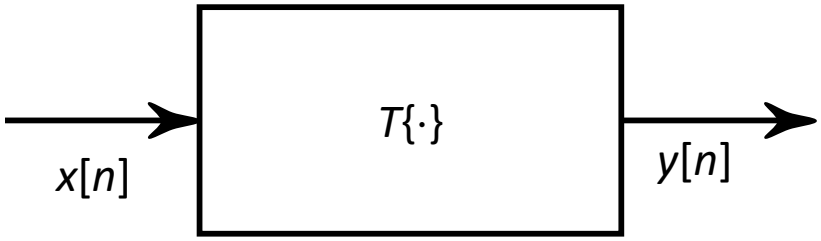
EXPONENTIAL SEQUENCE

$x[n] = A\,\alpha^{n}$



If we want an exponential sequence that is zero for $n < 0$, we can write this as:

SYSTEMS



A discrete-time system is a transformation that maps an input sequence $x[n]$ into an output sequence $y[n]$.

SYSTEM CHARACTERISTICS

1. Linear vs. non-linear
2. Causal vs. non-causal
3. Time invariant

LINEARITY

A **linear** system is one that obeys the principle of superposition,

$$T\{a_1 x_1[n] + a_2 x_2[n]\} = a_1 y_1[n] + a_2 y_2[n]$$

where the output of a linear combination of inputs is the same linear combination applied to the individual outputs. This result means that a complicated system can be decomposed into a linear combination of elementary functions whose transformation is known, and then taking the same linear combination of the results. Linearity also implies that the behavior of the system is independent of the magnitude of the input.

CAUSALITY

A system is *causal* if, for every choice of n_0, the output sequence at the index $n = n_0$ depends only on the input sequence values for $n \leq 0$.

All physical time-based systems are causal because they are unable to look into the future and anticipate a signal value that will occur later.

A system is causal it's output is a function of only the current and previous samples

Backward Difference $y[n] = x[n] - x[n-1]$

Forward Difference $y[n] = x[n+1] + x[n]$

STABILITY

A system is stable in the bounded-input, bounded-output (BIBO) sense if and only it every bounded input produces a bounded output sequence.

The input $x[n]$ is bounded if there exists a fixed positive finite value B_x such that

$$|x[n]| \leq B_x < \infty$$

Stability requires that for any possible input sequence there exist a fixed positive value B_y such that

$$|y[n]| \leq B_y < \infty$$

PASSIVE AND LOSSLESS SYSTEMS

A system is said to be *passive* if, for every finite energy input sequence $x[n]$, the output sequence has at most the same energy

$$\sum_{n=-\infty}^{\infty} |y[n]|^2 \leq \sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty$$

If the above inequality is satisfied with an equal sign for every input signal, the system is said to be *lossless*.

EXAMPLES OF SYSTEMS

1. Ideal Delay System $y[n] = x[n - n_d]$

2. Moving Average System $y[n] = \frac{1}{M_2 + M_1 + 1} \sum_{k=-M_1}^{M_2} x[n-k]$

3. Memoryless non-linear System $y[n] = x[n]^2$

4. Accumulator System $y[n] = \sum_{k=-\infty}^{n} x[k]$

5. Compressor System $y[n] = x[Mn]$

6. Backward Difference System $y[n] = x[n] - x[n-1]$

Z-TRANSFORM DEFINITION

Counterpart of the Laplace transform for discrete-time signals
-Generalization of the Fourier Transform
-Fourier Transform does not exist for all signals

The *z*-transform of a sequence *x*[*n*] is defined as

$$Z\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n]z^{-n} = X(z)$$

The inverse z-transform is given by

$$x[n] = \frac{1}{2\pi j}\oint_C X(z)z^{n-1}dz$$

RELATIONSHIP TO FOURIER TRANSFORM

The *z*-transform of a sequence *x*[*n*] is defined as

$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}$$

The Fourier transform of a sequence *x*[*n*] is defined as

$$X\left(e^{j\omega}\right) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$$

For $z = e^{j\omega}$ the *z*-transform reduces to the Fourier transform

MODULE-V

IIR AND FIR DIGITAL FILTERS

BASIC STRUCTURES FOR IIR SYSTEMS:

- Direct Forms
- Cascade Form
- Parallel Form
- Feedback in IIR Systems

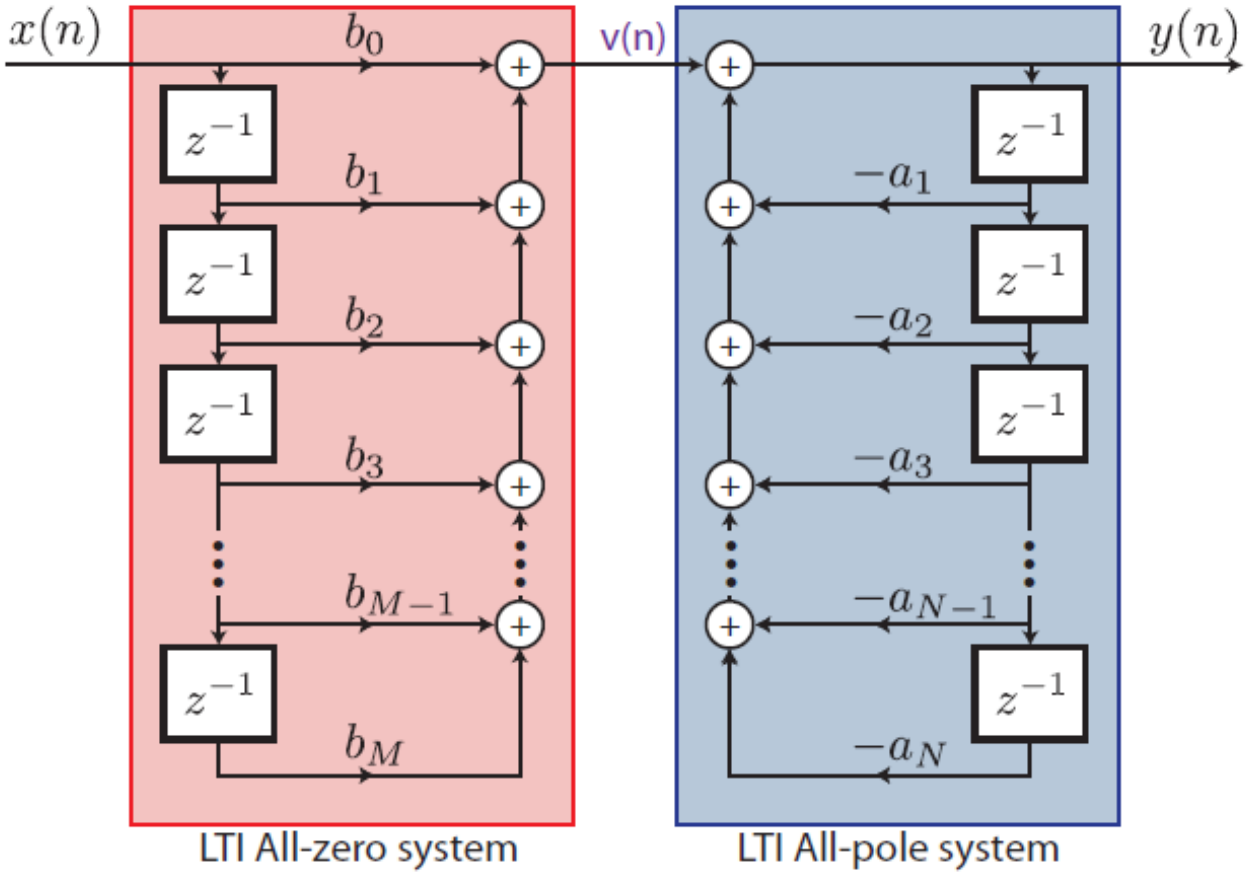
BASIC STRUCTURES FOR IIR SYSTEMS:

- **DIRECT FORMS**

$$y[n] - \sum_{k=1}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k]$$

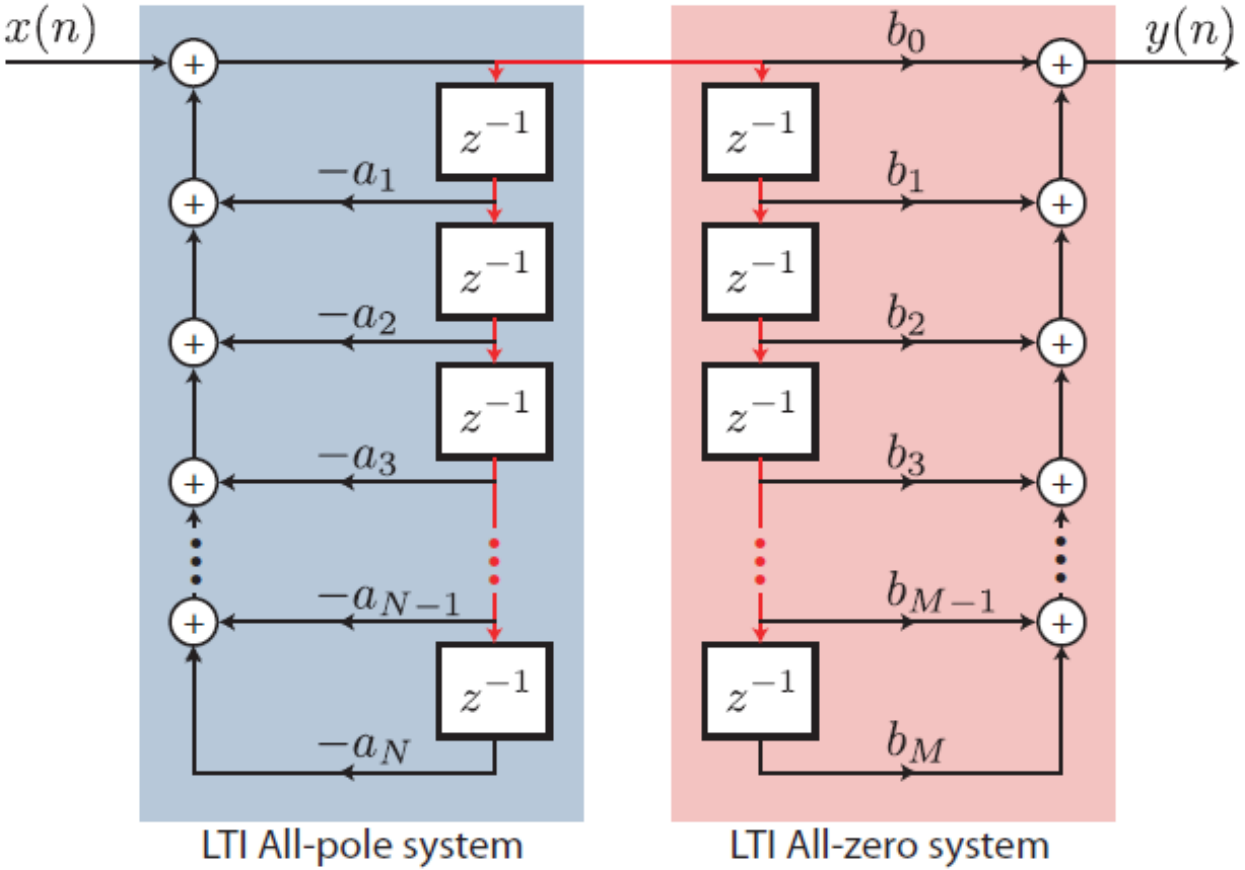
$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 - \sum_{k=1}^{N} a_k z^{-k}}$$

BASIC STRUCTURES FOR IIR SYSTEMS:



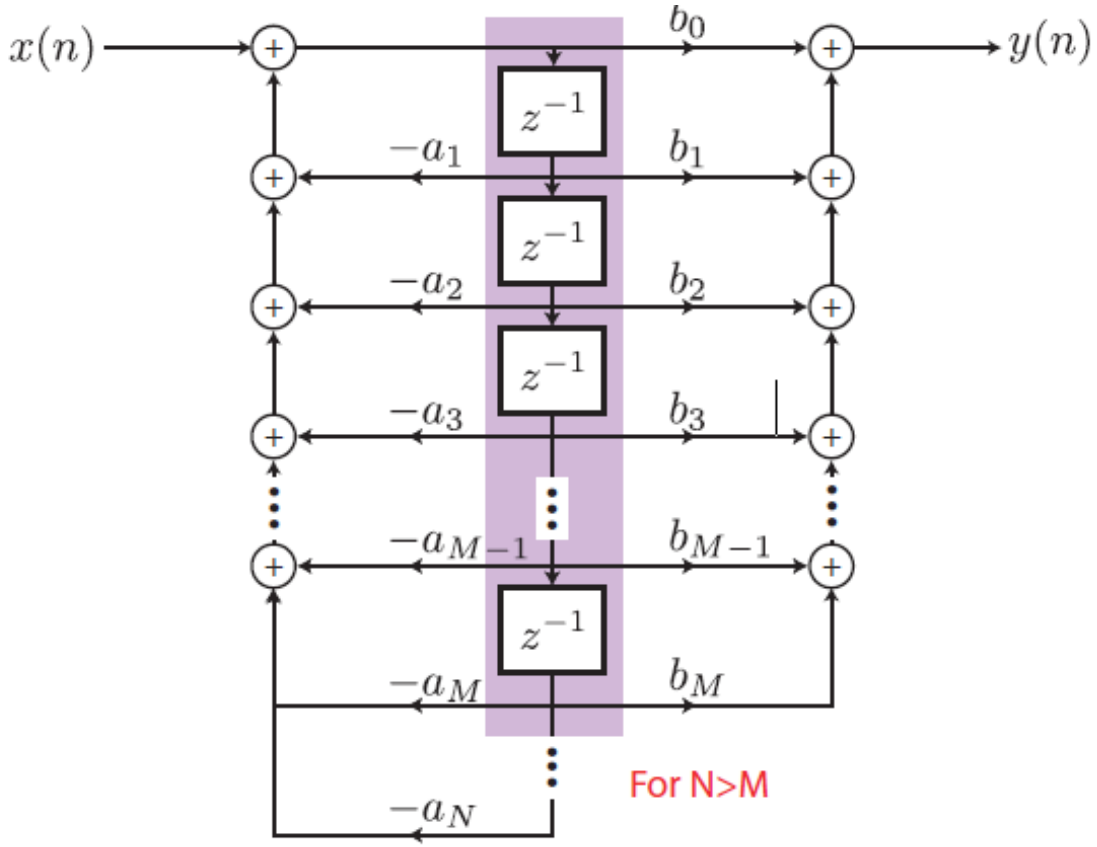
Requires: $M + N + 1$ multiplications, $M + N$ additions, $M + N$ memory locations

BASIC STRUCTURES FOR IIR SYSTEMS:



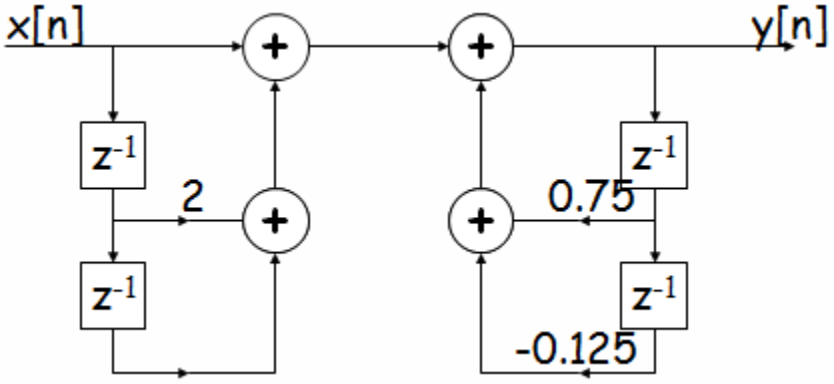
Requires: $M + N + 1$ multiplications, $M + N$ additions, $M + N$ memory locations

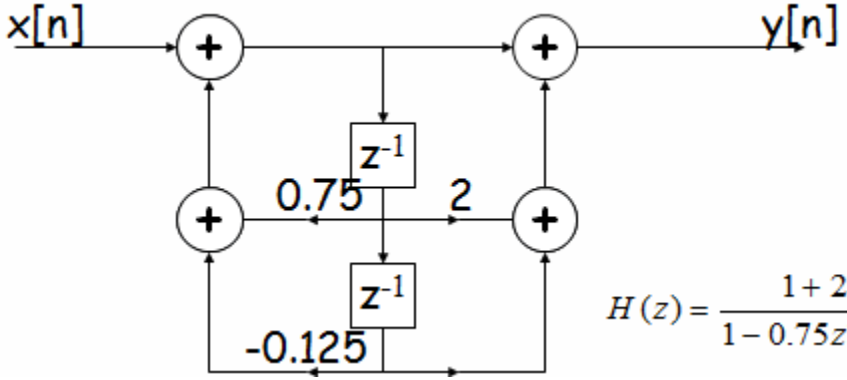
Direct Form II



Requires: $M + N + 1$ multiplications, $M + N$ additions, $\max(M, N)$ memory locations

DIRECT FORM II



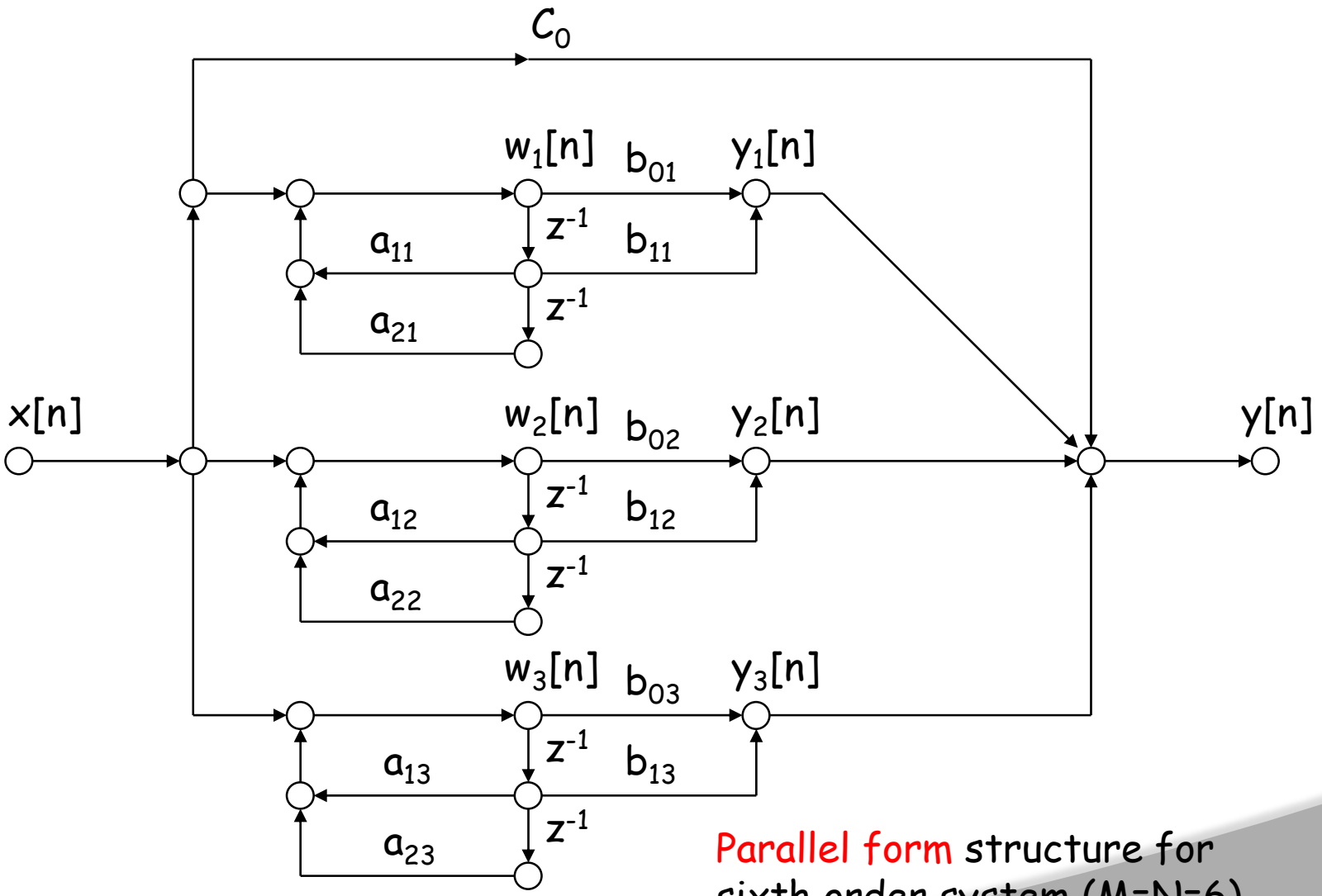


$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}}$$

- **CASCADE FORM**

$$H(z) = A\frac{\prod_{k=1}^{M_1}(1-g_k z^{-1})\prod_{k=1}^{M_2}(1-h_k z^{-1})(1-h_k^* z^{-1})}{\prod_{k=1}^{N_1}(1-c_k z^{-1})\prod_{k=1}^{N_2}(1-d_k z^{-1})(1-d_k^* z^{-1})}$$

BASIC STRUCTURES FOR IIR SYSTEMS



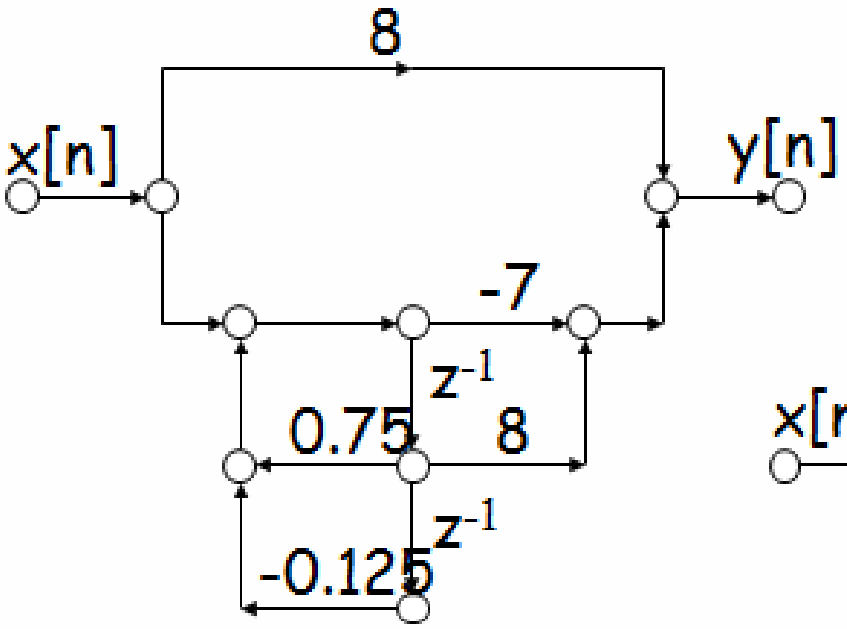
Parallel form structure for sixth order system (M=N=6).

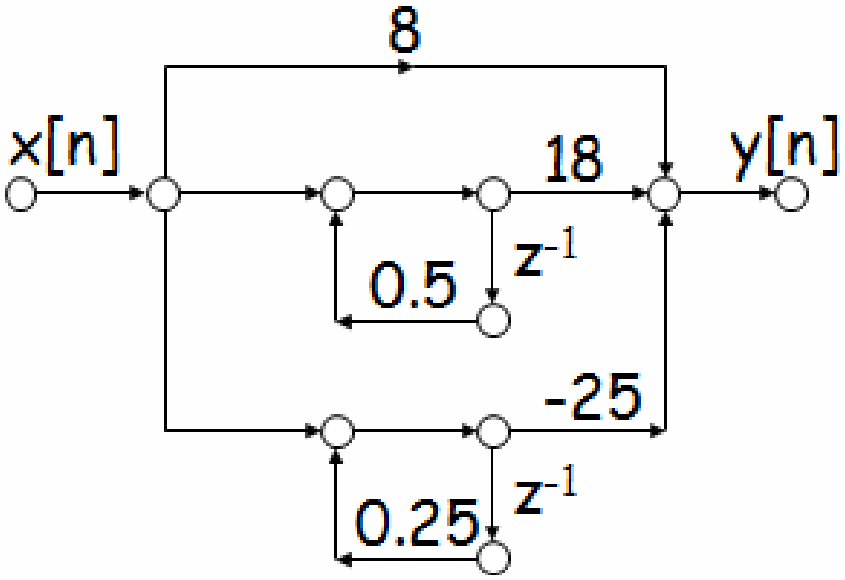
BASIC STRUCTURES FOR IIR SYSTEMS

.

$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 - 0.75z^{-1} + 0.125z^{-2}} = 8 + \frac{-7 + 8z^{-1}}{1 - 0.75z^{-1} + 0.125z^{-2}}$$

$$= 8 + \frac{18}{1 - 0.5z^{-1}} - \frac{25}{1 - 0.25z^{-1}}$$





BASIC STRUCTURES FOR FIR SYSTEMS

.

- Direct Form

It is also referred to as a tapped delay line structure or a transversal filter structure.

- Transposed Form
- Cascade Form

$$H(z) = \sum_{n=0}^{M} h[n] z^{-n} = \prod_{k=1}^{M_S} (b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2})$$

where MS is the largest integer contained in (M + 1)/2. If M is odd, one of coefficients b2k will be zero.

BASIC STRUCTURES FOR FIR SYSTEMS

.

- Direct Form

It is also referred to as a tapped delay line structure or a transversal filter structure.

- Transposed Form
- Cascade Form

$$H(z) = \sum_{n=0}^{M} h[n] z^{-n} = \prod_{k=1}^{M_S} (b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2})$$

where MS is the largest integer contained in (M + 1)/2. If M is odd, one of coefficients b2k will be zero.

BASIC STRUCTURES FOR FIR SYSTEMS

.

- Direct Form

It is also referred to as a tapped delay line structure or a transversal filter structure.

- Transposed Form
- Cascade Form

$$H(z) = \sum_{n=0}^{M} h[n] z^{-n} = \prod_{k=1}^{M_S} (b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2})$$

where MS is the largest integer contained in (M + 1)/2. If M is odd, one of coefficients b2k will be zero.

BASIC STRUCTURES FOR FIR SYSTEMS

.

- Direct Form

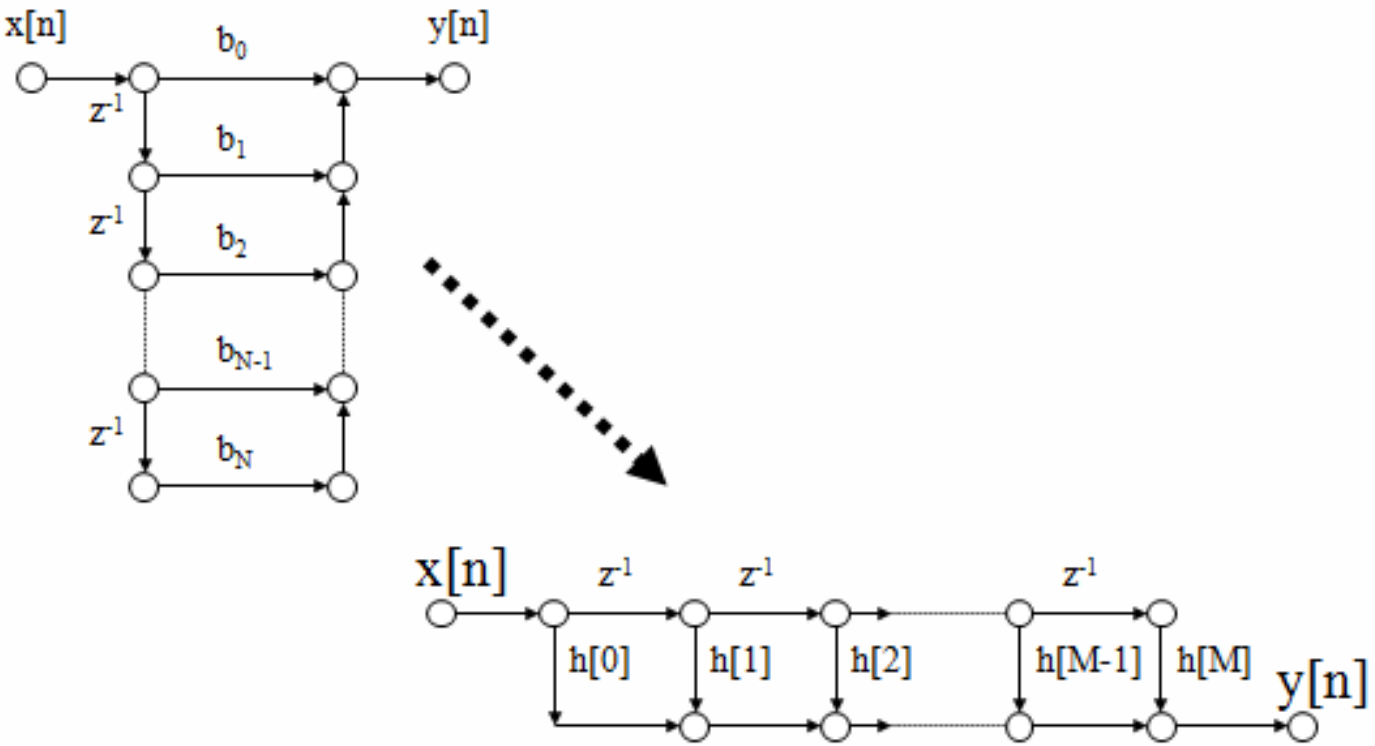
It is also referred to as a tapped delay line structure or a transversal filter structure.

- Transposed Form
- Cascade Form

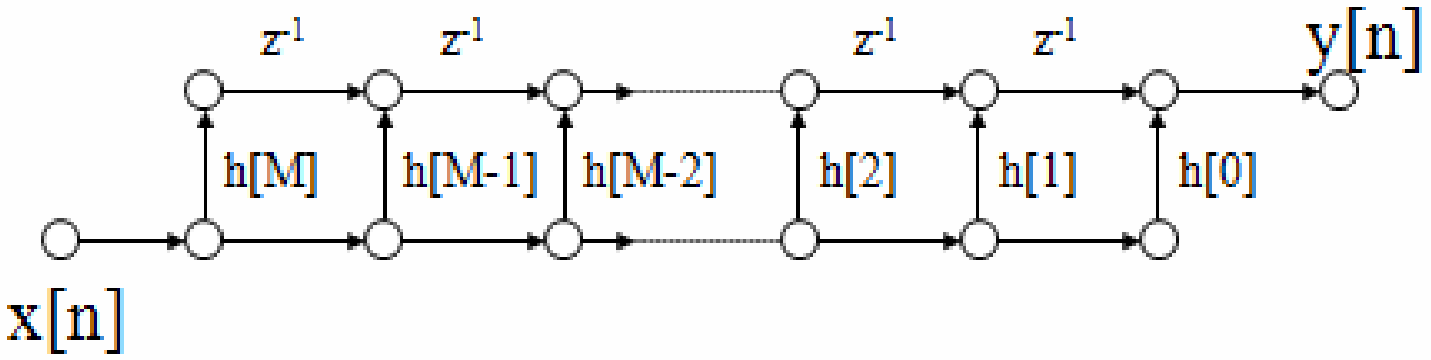
$$H(z) = \sum_{n=0}^{M} h[n] z^{-n} = \prod_{k=1}^{M_S} (b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2})$$

where MS is the largest integer contained in (M + 1)/2. If M is odd, one of coefficients b2k will be zero.

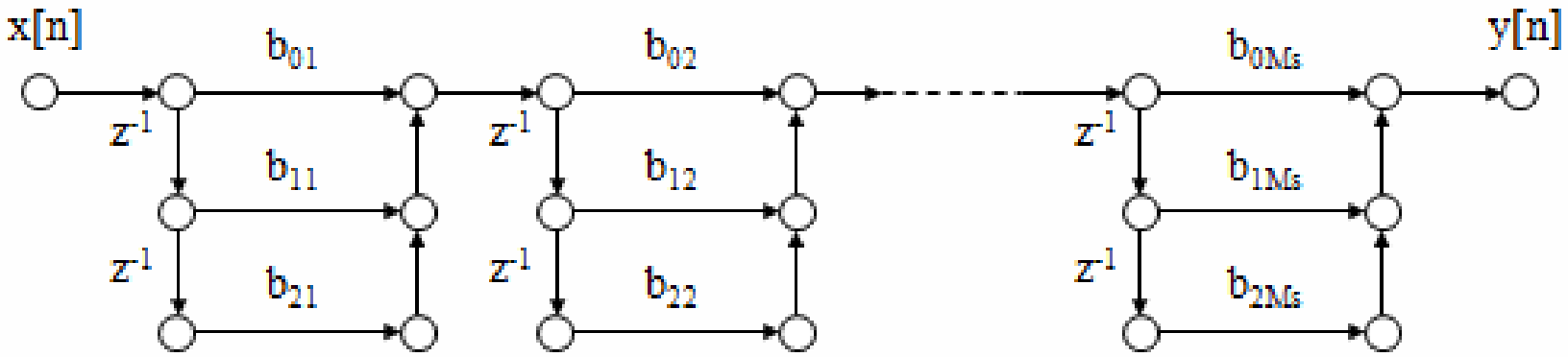
DIRECT FORM (TAPPED DELAY LINE)



TRANSPOSED FORM OF FIR NETWORK



CASCADE FORM OF A FIR SYSTEM



Thank you