Quiz 4

Chapter2:
Memory, I/O addressing

Asst.Prof.Dr.Supakit Nootyaskool

Objective

- Explain the relationship between binary and hexadecimal systems.
- Understand the concepts of parity and error correction.
- Know the structure and content of a text file.
- Understand the roles of the address bus, data bus, and control signals.
- Explain devices selection in a decoder circuit.

Topics

- Relationship between binary and hexadecimal
- Parity and error correction
- Text data
- Address bus, Data bus, and Control signal
- Decoder

RELATIONSHIP BETWEEN BINARY AND HEXADECIMAL

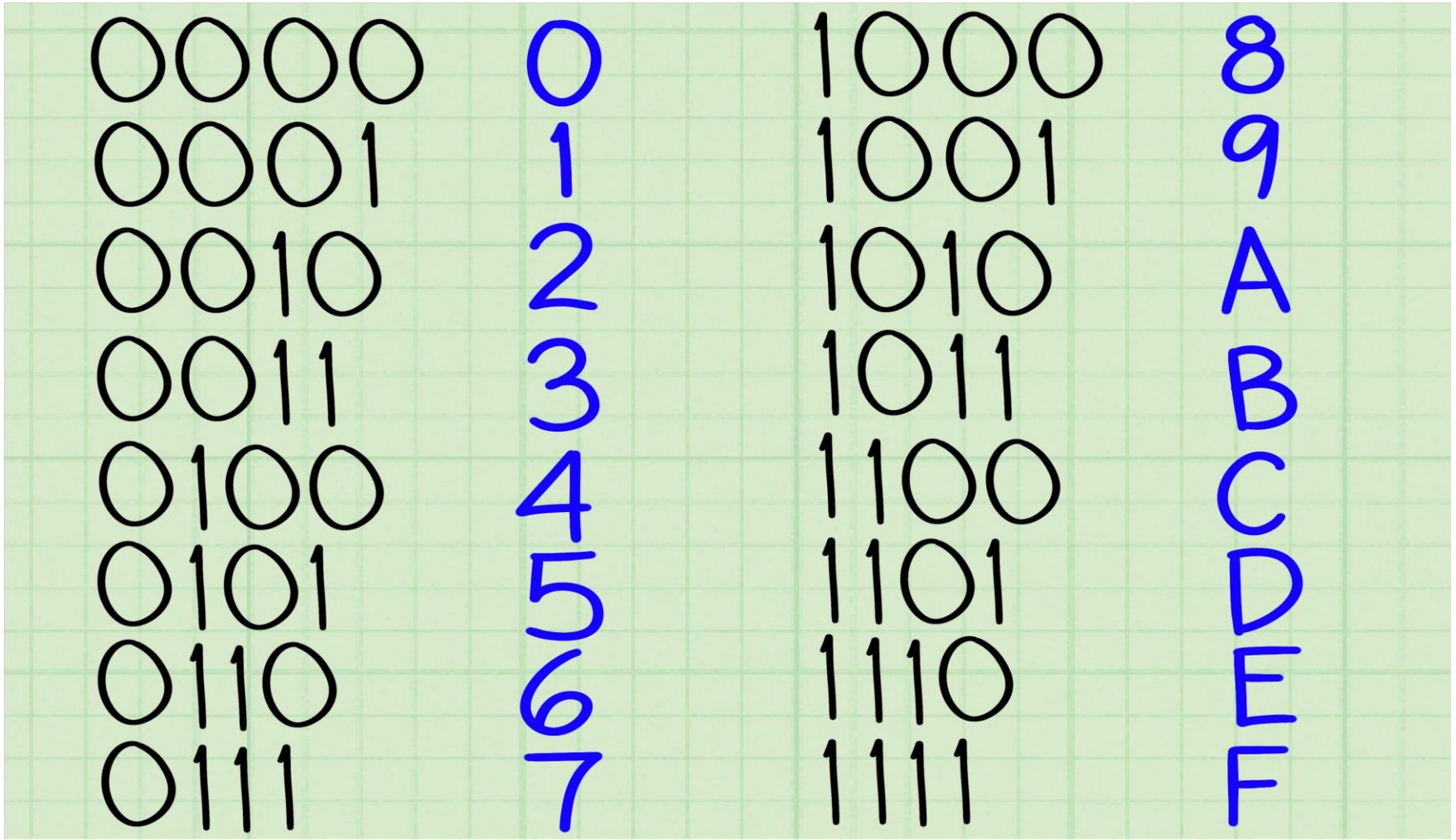
Student will be able to convert binary data to hexadecimal systems.

When you read a piece of computer code

```
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#define PORT1 0x3F8  /* Defines Serial Port Base Address (COM1 */
void main(void){
    unsigned char c = 0;
    unsigned char chrctr = 0;
    /*int exit = 1; */
    outportb(PORT1 + 1, 0); /* Turn off interrupts */
    /* PORT1 Communication Settings */
    outportb(PORT1 + 3, 0x80); /* Set DLAB ON */
    outportb(PORT1 + 0, 0x0C); /* Set the baud rate to 9600 */
    outportb(PORT1 + 1, 0x00); /* Set Baud - Divisor latch HIGH */
    outportb(PORT1 + 3, 0x03); /* 8 bits, no parity, 1 stop */
    outportb(PORT1 + 2, 0xC7); /* FIFO Control Register */
    outportb(PORT1 + 4, 0x0B); /* Turn on DTR, RTS, and OUT2) */
    printf("Waiting on transmission from source.\nPress ESC to quit.\n");
    while(chrctr != 27){      /* Execute the loop if ESC has been hit */
    c = inportb(PORT1 + 5);
    if (c & 0x01){
        chrctr = inportb(PORT1);
        printf("%d",chrctr);
        }
    if (kbhit()){
        chrctr = getch();
        outportb(PORT1, chrctr);
        }
    }
    }"
```

What is 0x3F8?

Conversion binary and hexadecimal systems



Activity2.1 Convert Binary and Hex

- 3F8H

  0011-____-____

0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

- 01101100111100111101

0 1 0 9 C F 3 D

Format of the number system

- We can conclude that

$$\text{Value} \times \text{Base}^{\text{position-1}}$$

- $101_2 = 1x2^2 + 0x2^1 + 1x2^0 = ?_{10}$
- $453_8 = 4x8^2 + 5x8^1 + 3x8^0 = ?_{10}$
- $453_{16} = 4x16^2 + 5x16^1 + 3x16^0 = ?_{10}$

Relationship between binary, octal, and hexadecimal

- 4310_{10} = 1000011100001
  divided in 4 = 0001 0000 1110 0001
  divided in 3 = 001 000 011 100 001

0001 0000 1110 0001

Hex = 1 0 E 1 = $10E1_{16}$

001 000 011 100 001

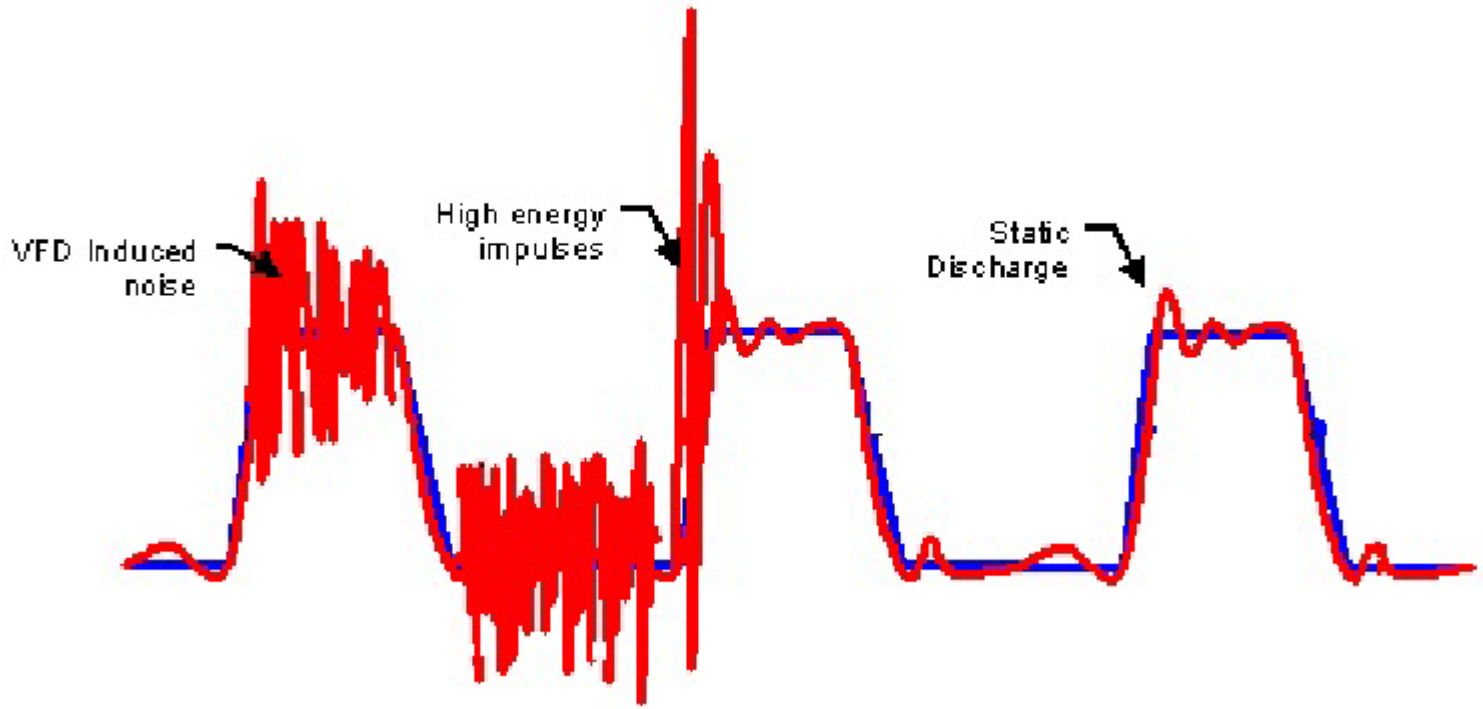
Oct = 1 0 3 4 1 = 10341_{8}

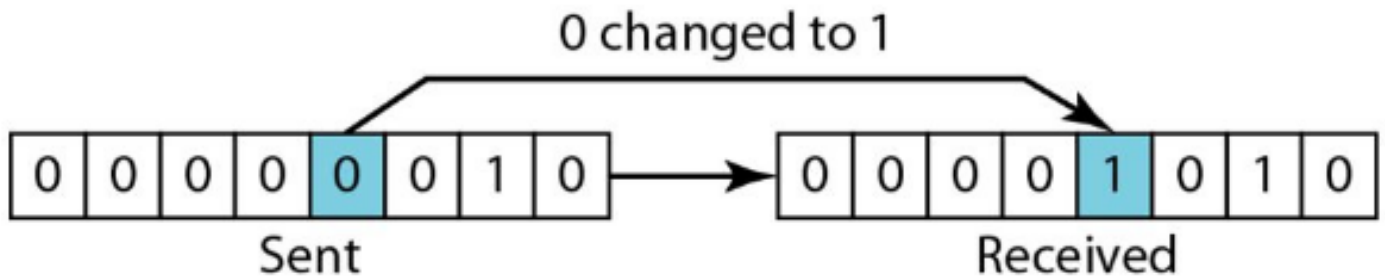
PARITY AND ERROR CORRECTION

Students will be able to generate binary data using odd or even parity formats.

Students will be able to explain the mechanism of error correction in data transmission.

Error in signal communication





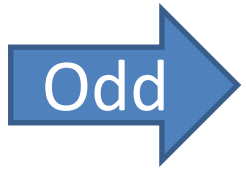
Parity bit and error correction

- A parity bit is a bit appended to a binary sequence in order to sum of the number in even or odd.

- Even = 0, Odd = 1

0100

01001



01000

Parity bit and error correction

0110

Even → 01100

Odd → 01101

001110

Even → 0011101

Odd → 0011100

Activity 2.2 write parity bit

101101

Even → 1011010

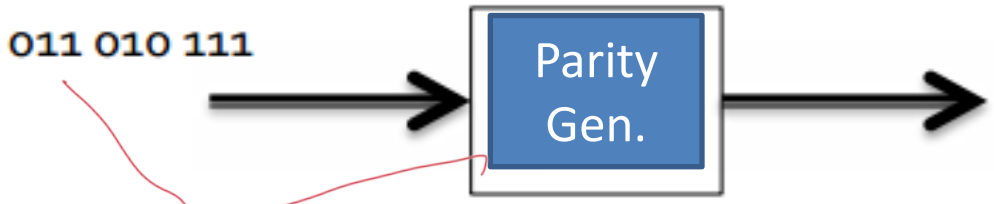
Odd → 1011011

0101110

Even → 0101101

Odd → 0101100

Error correction



0110 0011 1010 1111

1 1 1 1	
0 1 0 1	
1 1 0 0	→
0 1 1 0	

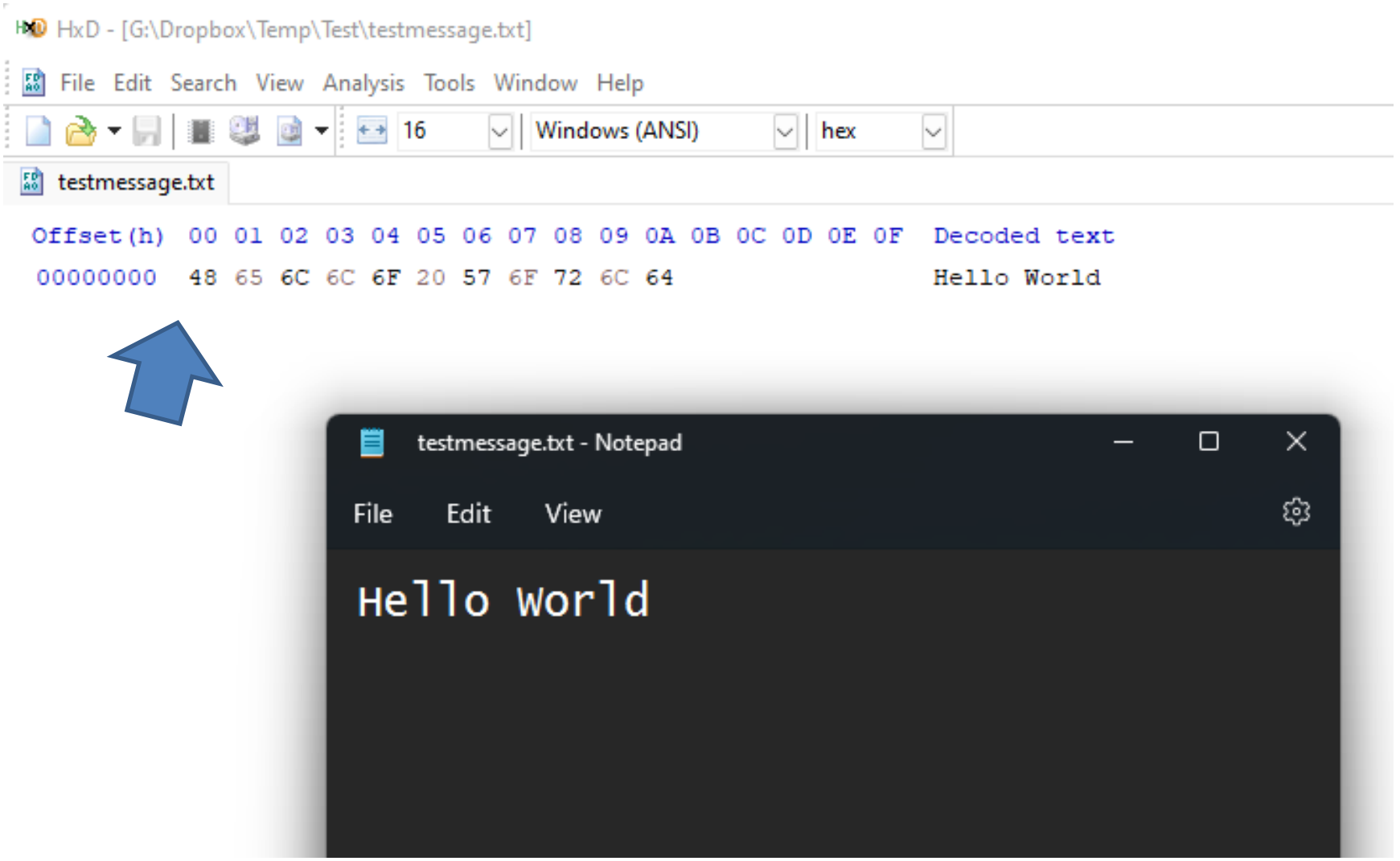
At receiver, if there is an error

Not even	
1 1 1 1	→ Even and checked
0 0 0 1	Not even
1 1 0 0	→ Even and checked
0 1 1 0	→ Even and checked

TEXT FILE

Student will be able to demonstrate by showing biary data inside a text file.

Text file



https://mh-nexus.de/en/

American Standard Code for Information Interchange (ASCII)

ASCII control characters

00	NULL	(Null character)
01	SOH	(Start of Header)
02	STX	(Start of Text)
03	ETX	(End of Text)
04	EOT	(End of Trans.)
05	ENQ	(Enquiry)
06	ACK	(Acknowledgement)
07	BEL	(Bell)
08	BS	(Backspace)
09	HT	(Horizontal Tab)
10	LF	(Line feed)
11	VT	(Vertical Tab)
12	FF	(Form feed)
13	CR	(Carriage return)
14	SO	(Shift Out)
15	SI	(Shift In)
16	DLE	(Data link escape)
17	DC1	(Device control 1)
18	DC2	(Device control 2)
19	DC3	(Device control 3)
20	DC4	(Device control 4)
21	NAK	(Negative acknowledgement)
22	SYN	(Synchronous idle)
23	ETB	(End of trans. block)
24	CAN	(Cancel)
25	EM	(End of medium)
26	SUB	(Substitute)
27	ESC	(Escape)
28	FS	(File separator)
29	GS	(Group separator)
30	RS	(Record separator)
31	US	(Unit separator)
127	DEL	(Delete)

ASCII printable characters

32	space	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

Extended ASCII characters

128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	░	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

USASCII code chart

Bits $b_7 b_6 b_5$ →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 b_3 b_2 b_1 ↓ Row / Column →	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

USASCII code chart

Bits $b_7 b_6 b_5$ →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 b_3 b_2 b_1 ↓ Row / Column →	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

A = 41H = 1000001

ADDRESS BUS, DATA BUS AND CONTROL SIGNAL

Student will able to describe the specific of the address bus, data bus and control signal in the computer system.

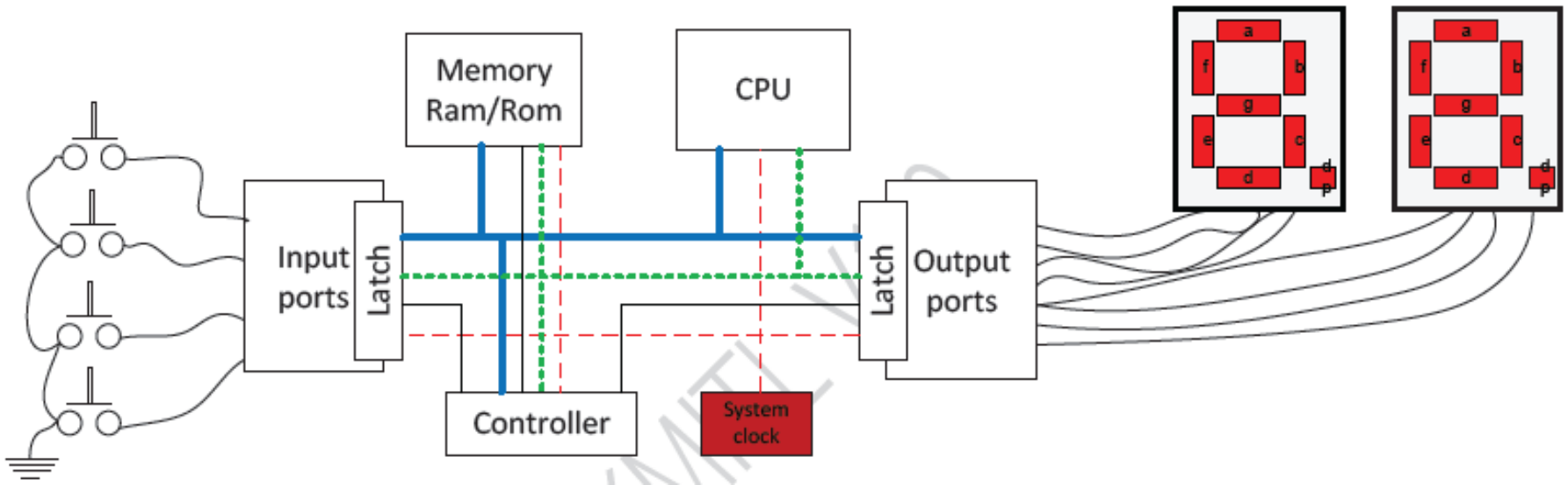
ADDRESS, DATA, and CONTROL

- **Address bus** is a sequence of binary data that points to the location of the data in the memory or I/O devices.
- **Control signal** is the signal sending to the devices such as the memory, the I/O to control reading, writing, enable, and disable.
- **Data bus** is the data that sending from a device to other device on the data wires. The data location relates to the address pointing the position.

Activity 2.3 Describe process of parcel express.

- What is an issue concern in the parcel post?

Data transferring in the computer system



Clock signal

Control signal

Address bus

Data bus

Device Address in computer

I/O port address	Description
060H – 064H	Keyboard controller
170H – 376H	Secondary IDE hard-disk controller
1F0H – 3F6H	Primary IDE hard-disk controller
220H	Sound card
300H	Network interface controller card (LAN card)
330H	SCSI adapter
3F2H	Floppy drive controller
2E8H, 2F8H, 3E8H, 3F8H	Communication port 1-4 (COM1-4)
278H, 378H, 3BCH	Line printer terminal port 1,2 (LPT1, 2, 3)

Example transferring data in the comport with C-language

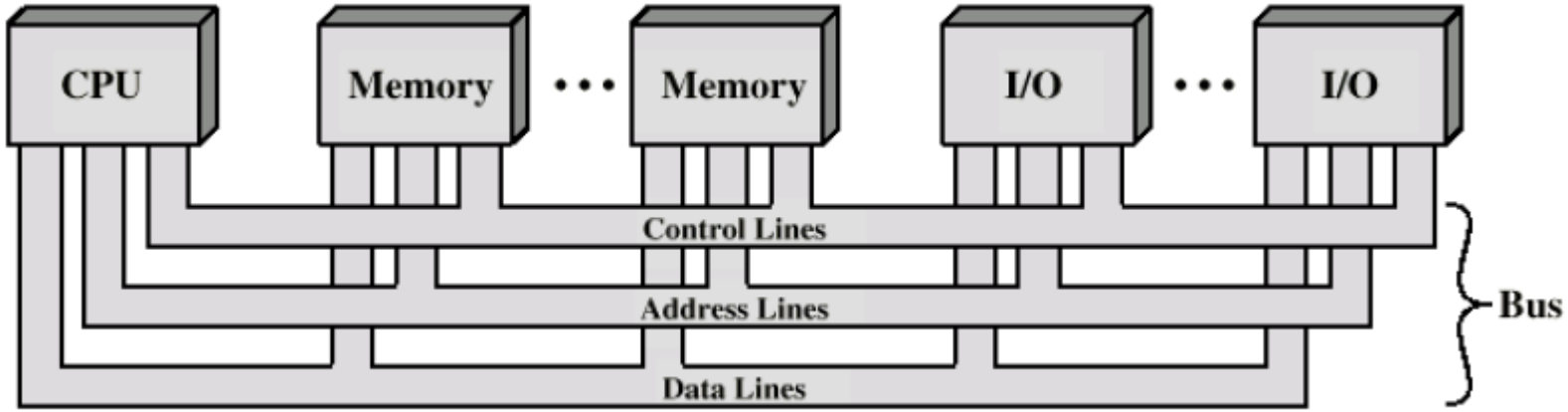
```
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#define PORT1 0x3F8  /* Defines Serial Port Base Address (COM1 */
void main(void){
    unsigned char c = 0;
    unsigned char chrctr = 0;
    /*int exit = 1; */
    outportb(PORT1 + 1, 0); /* Turn off interrupts */
    /* PORT1 Communication Settings */
    outportb(PORT1 + 3, 0x80); /* Set DLAB ON */
    outportb(PORT1 + 0, 0x0C); /* Set the baud rate to 9600 */
    outportb(PORT1 + 1, 0x00); /* Set Baud - Divisor latch HIGH */
    outportb(PORT1 + 3, 0x03); /* 8 bits, no parity, 1 stop */
    outportb(PORT1 + 2, 0xC7); /* FIFO Control Register */
    outportb(PORT1 + 4, 0x0B); /* Turn on DTR, RTS, and OUT2) */
    printf("Waiting on transmission from source.\nPress ESC to quit.\n");
    while(chrctr != 27){      /* Execute the loop if ESC has been hit */
    c = inportb(PORT1 + 5);
    if (c & 0x01){
        chrctr = inportb(PORT1);
        printf("%d",chrctr);
        }
    if (kbhit()){
        chrctr = getch();
        outportb(PORT1, chrctr);
        }
    }
    }"
```

Address

Digital data in computer

A computer has three bus

- Data bus:
- Address bus:
- Control bus:

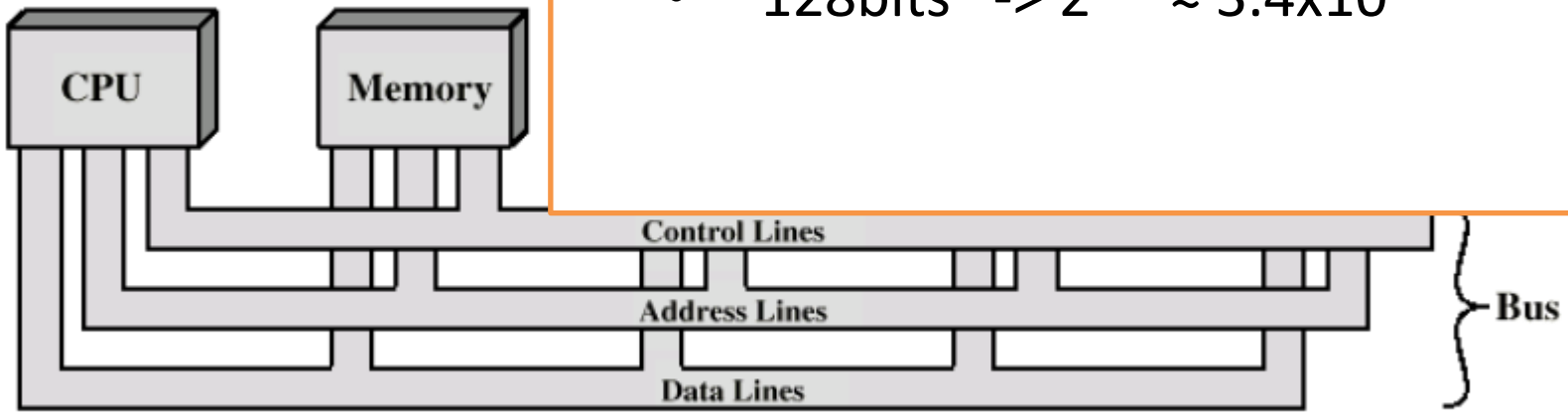


Digital data in com

A computer ha

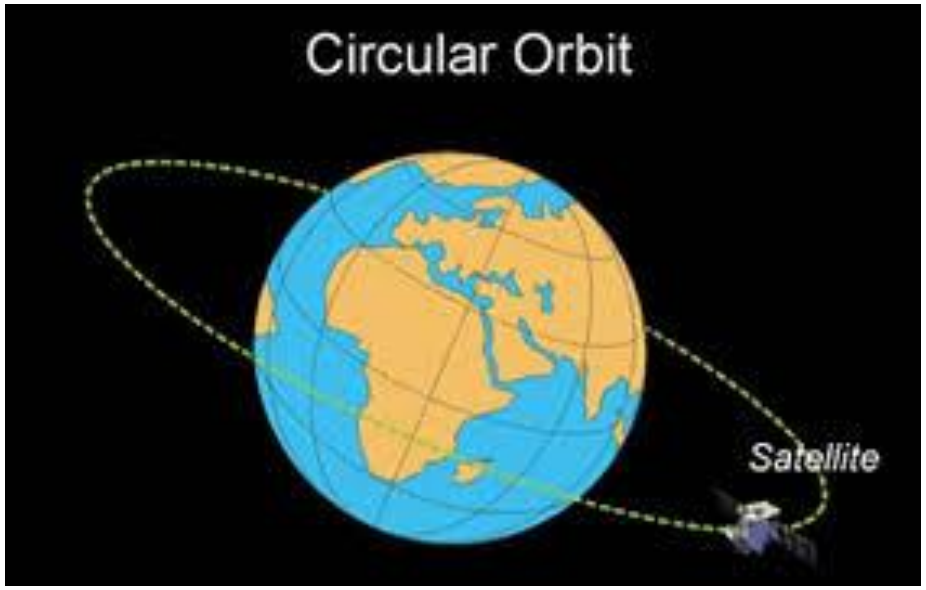
- Data bus:
- Address bus:
- Control bus:

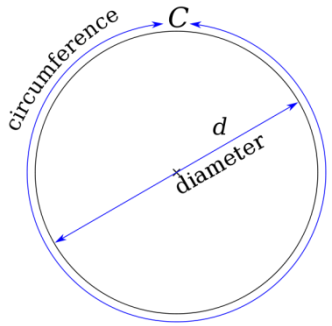
- Data bus carries data or information from CPU/a device to another device.
- Size of data bus
  - 8bits -> 2^8 = 256 levels
  - 16bits -> 2^{16} = 65,536 ≈64K
  - 32bits -> 2^{32} ≈ 4G
  - 64bits -> 2^{64} ≈ $1.8x10^9$
  - 128bits -> 2^{128} ≈ $3.4x10^{38}$



Why does the computer development from 8 bit to 128 bit?

- Calculation circumference between two objects





$C = 2\pi r$

Activity 2.4 Proof Float and Double keep PI

Approximation

A quick and easy approximation for **π** is 22/7

22/7 = **3.1428571...**

But as you can see, 22/7 is **not exactly right**. In fact **π** is not equal to the ratio of any two numbers, which makes it an irrational number.

A really good approximation, better than 1 part in 10 million, is:

355/113 = **3.1415929...**

(think "113355", slash the middle "113/355", then flip "355/113")

Summary:

22/7 = **3.14**28571...

355/113 = **3.1415929**...

π = **3.14159265**...

https://www.mathsisfun.com/numbers/pi.html

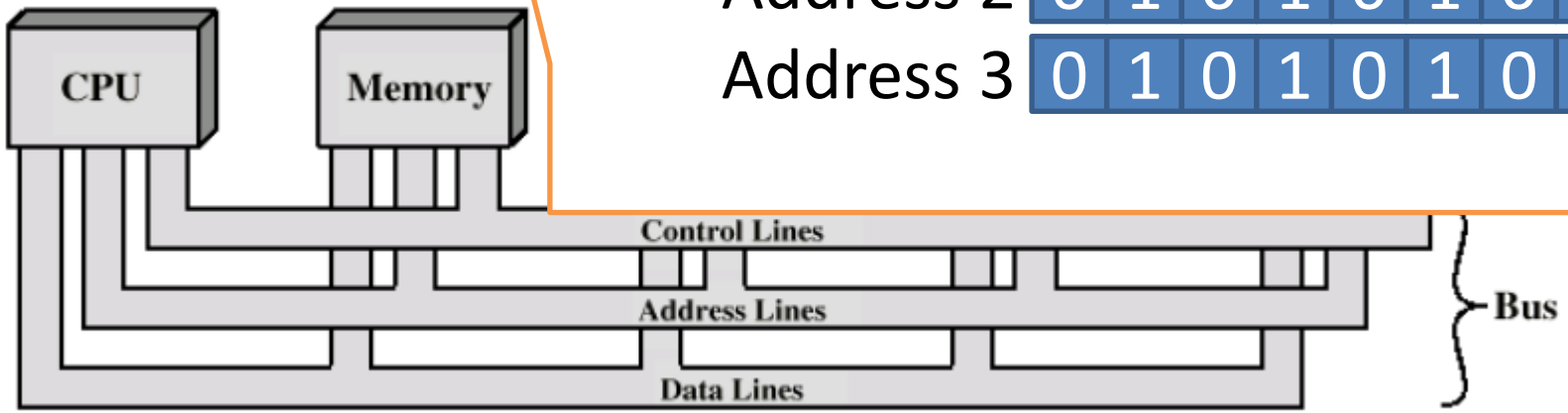
```c
#include <stdio.h>

int main()
{
    float pi_f = 355.0/113.0;
    double pi_d = 355.0/113.0;
    printf("%1.50e",pi_f);
    printf("\n");
    printf("%1.50e",pi_d);
    return 0;
}
```

A computer has three bu

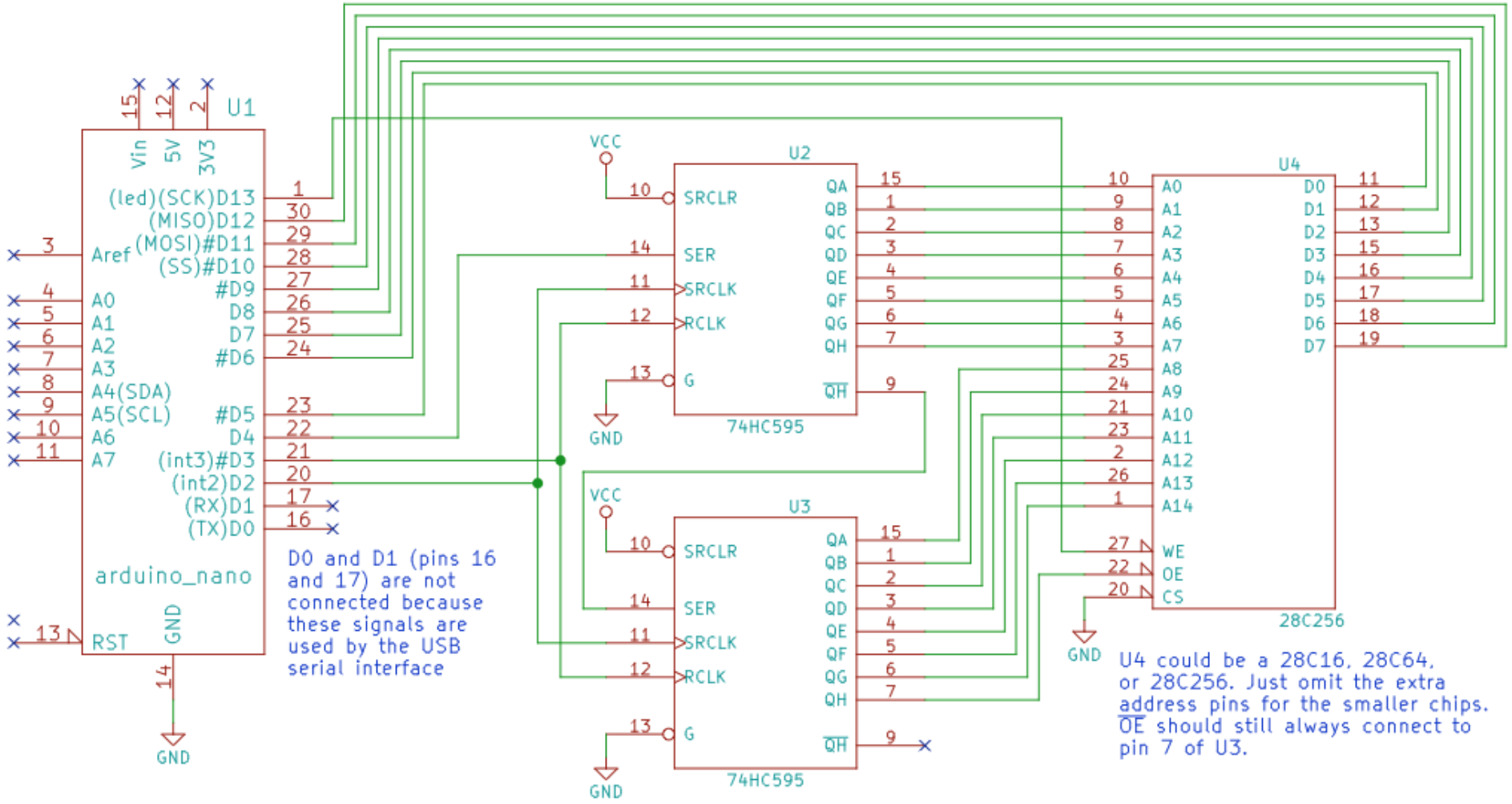
- Data bus:
- Address bus:
- Control bus:

- Address bus relates to position of devices.
- For example memory 4Byte at 8bit has

Address 0	0	1	0	1	0	1	0	1
Address 1	0	1	0	1	0	1	0	1
Address 2	0	1	0	1	0	1	0	1
Address 3	0	1	0	1	0	1	0	1

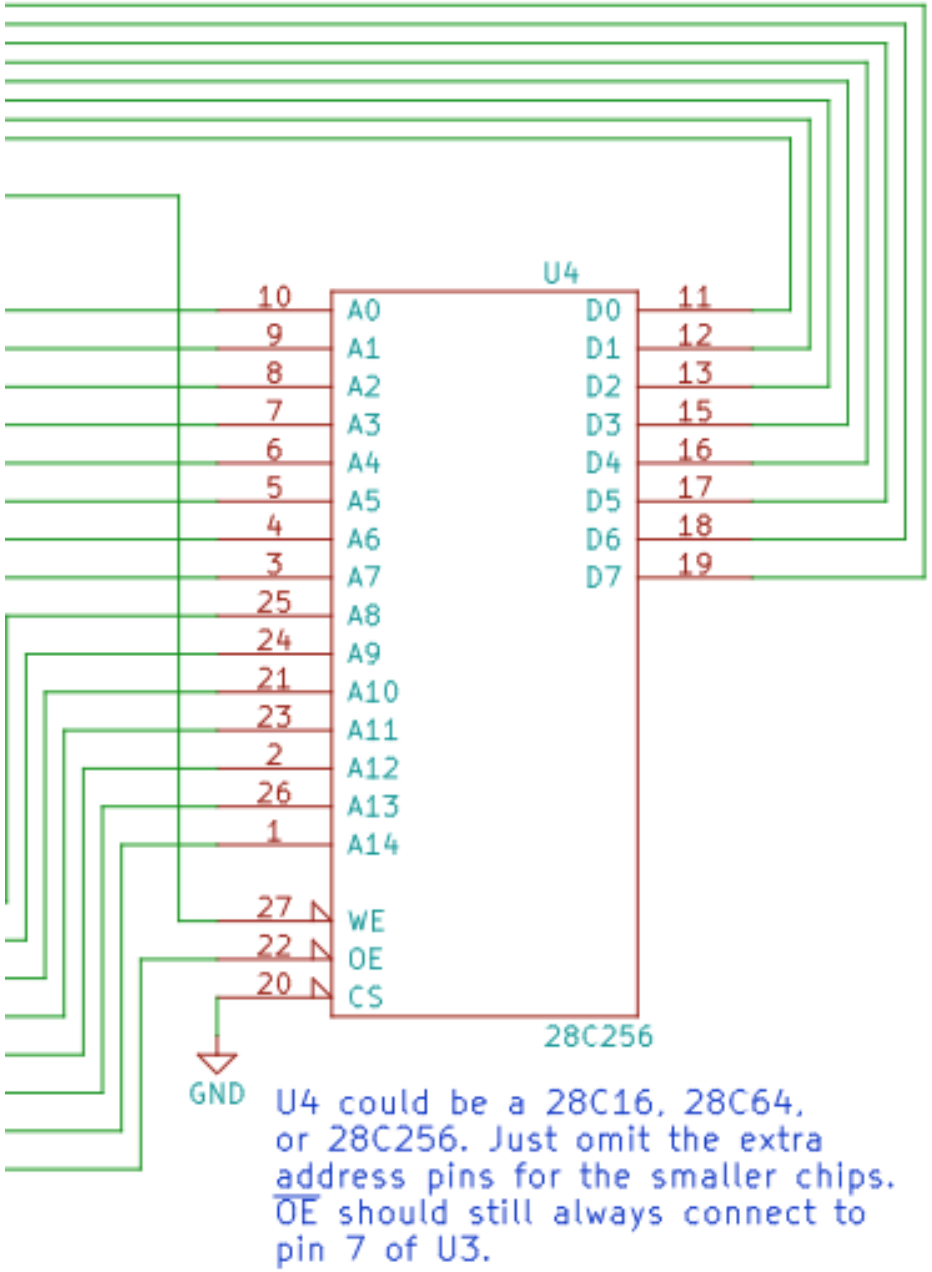


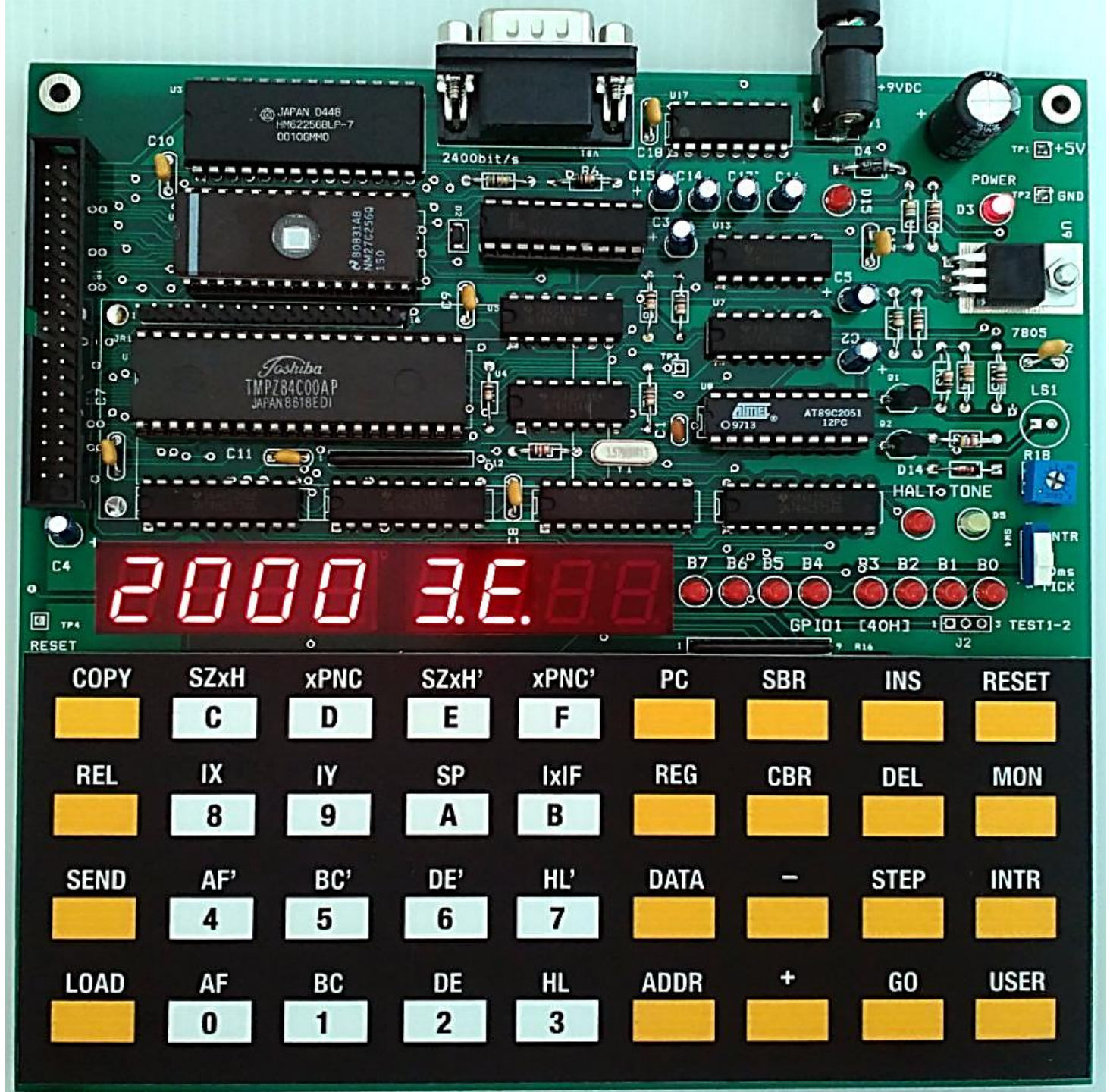
Activity 2.5, The circuit diagram shown below has 4 ICs, given you consider, which is the chip doing as the memory.

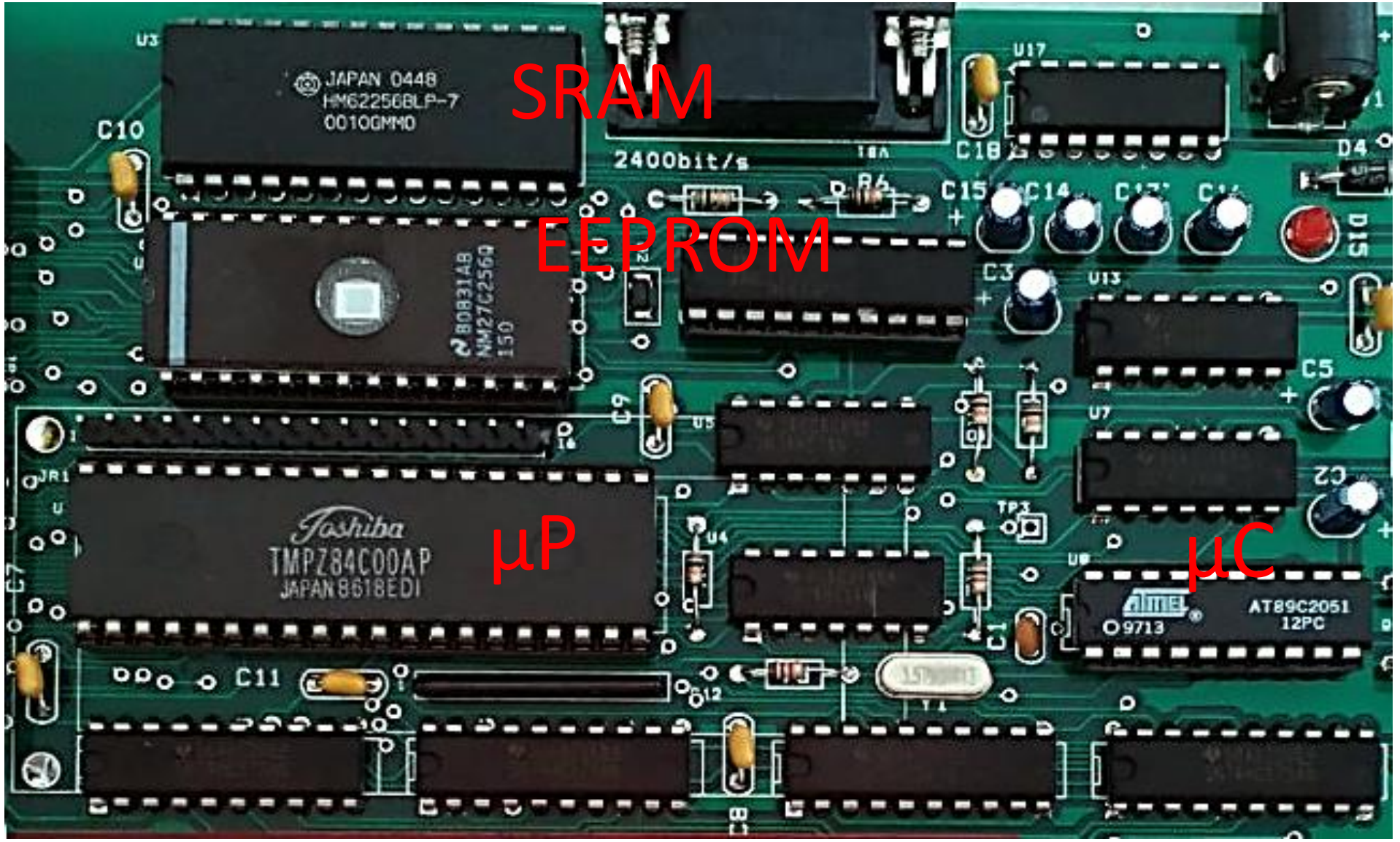


Activity 2.6, The memory IC has the pin name as the list below, given you write the full name of the pin.

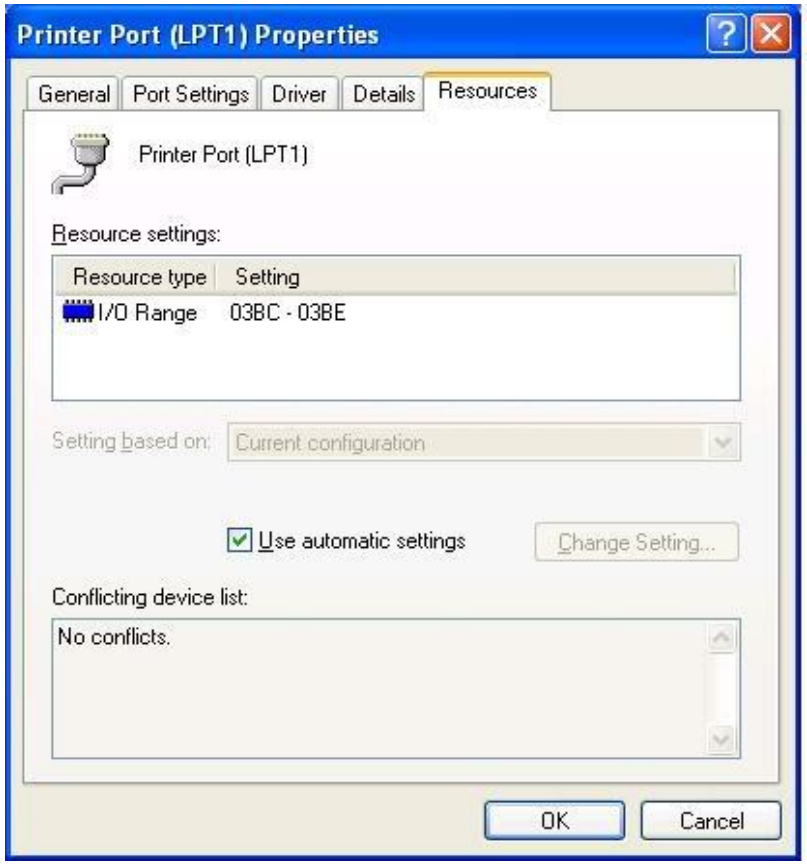
1) An Address
2) Dn Data
3) WE write able
4) OE output able
5) CS chip select

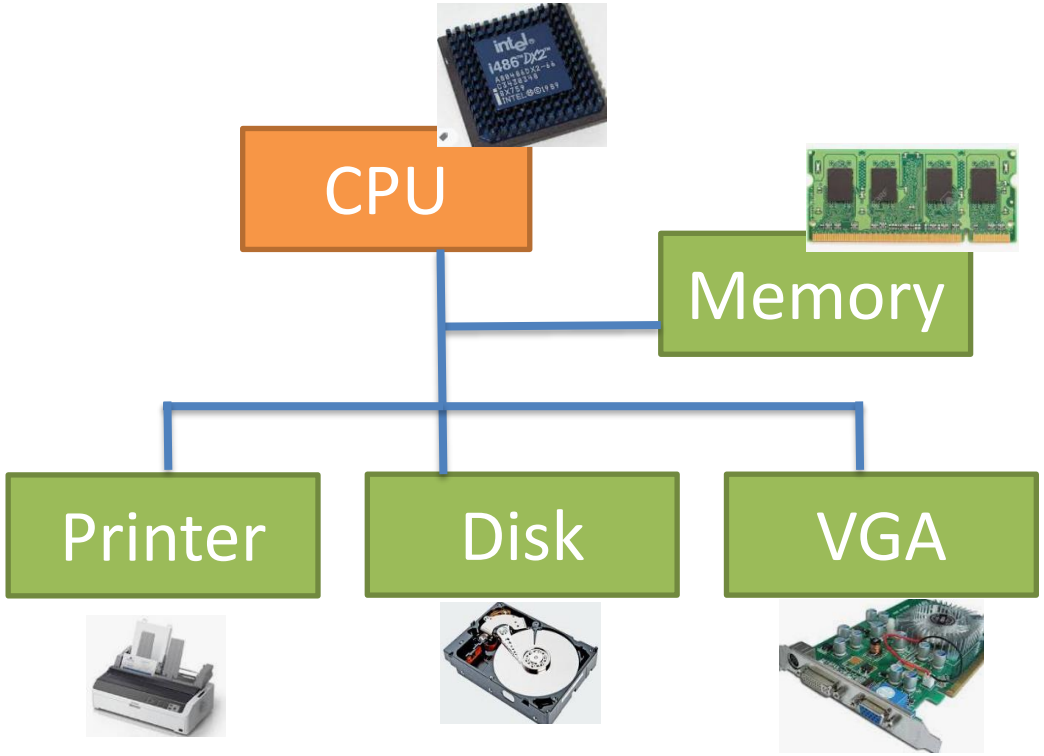



JAPAN 0448
HM62256BLP-7
0010GMMO
2400bit/s
+9VDC
+5V
GND
POWER
7805
LS1
R18
HALT
TONE
INTR
Toshiba
TMPZ84C00AP
JAPAN 8618EDI
AT89C2051
12PC
2000 3E
B7 B6 B5 B4 B3 B2 B1 B0
GPIO1 [40H]
TEST1-2
RESET
COPY SZxH xPNC SZxH' xPNC' PC SBR INS RESET
C D E F
REL IX IY SP IxIF REG CBR DEL MON
8 9 A B
SEND AF' BC' DE' HL' DATA − STEP INTR
4 5 6 7
LOAD AF BC DE HL ADDR + GO USER
0 1 2 3


SRAM
EEPROM
µP
µC
JAPAN 0448
HM62256BLP-7
0010GMM0
2400bit/s
Toshiba
TMPZ84C00AP
JAPAN 8618EDI
AT89C2051
12PC
9713

Example printer port address in Window XP

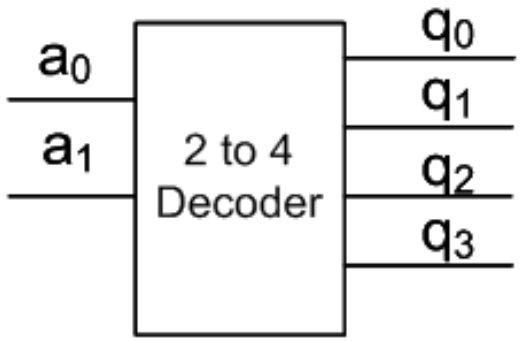




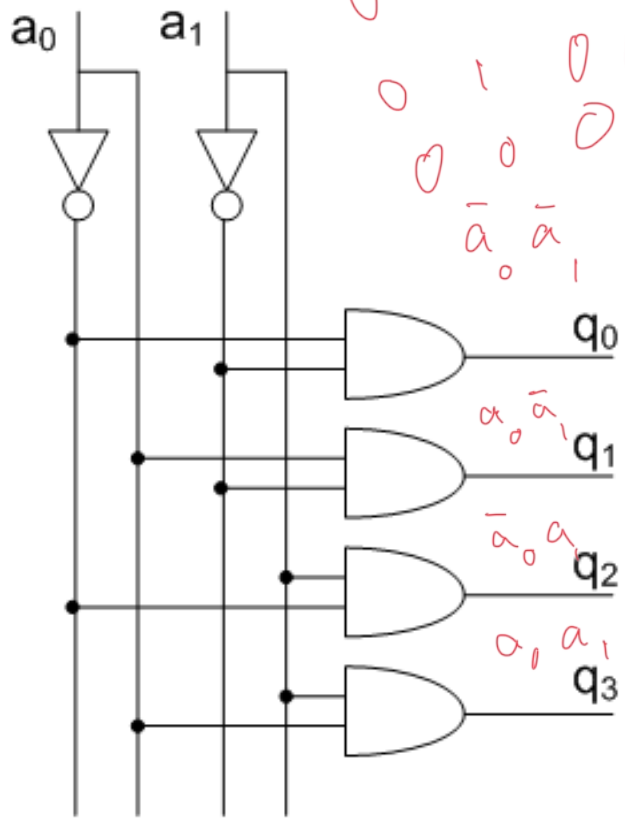
DECODER CIRCUIT

Student able to create a decode circuit based on the trouth table.

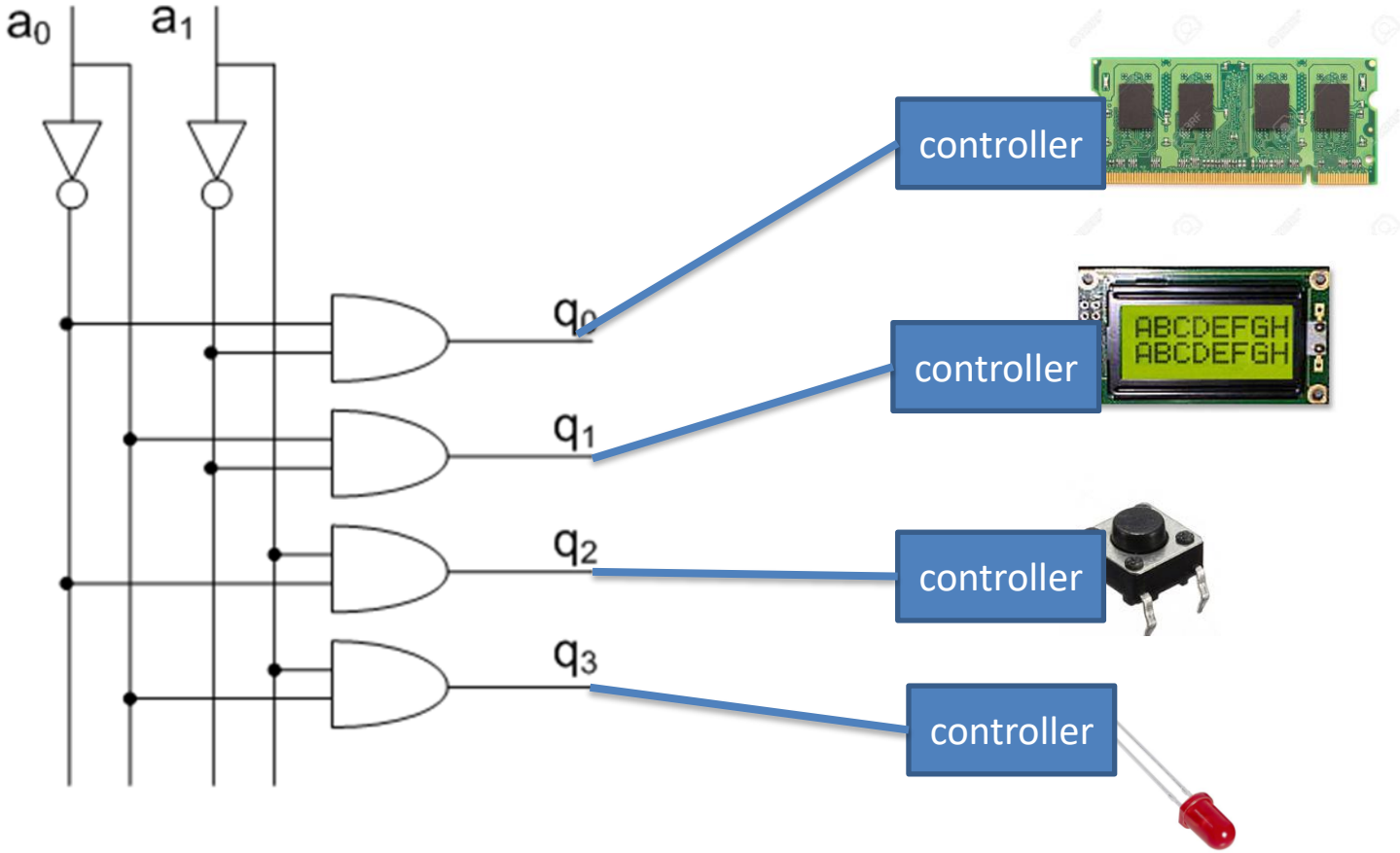
Decoder circuit



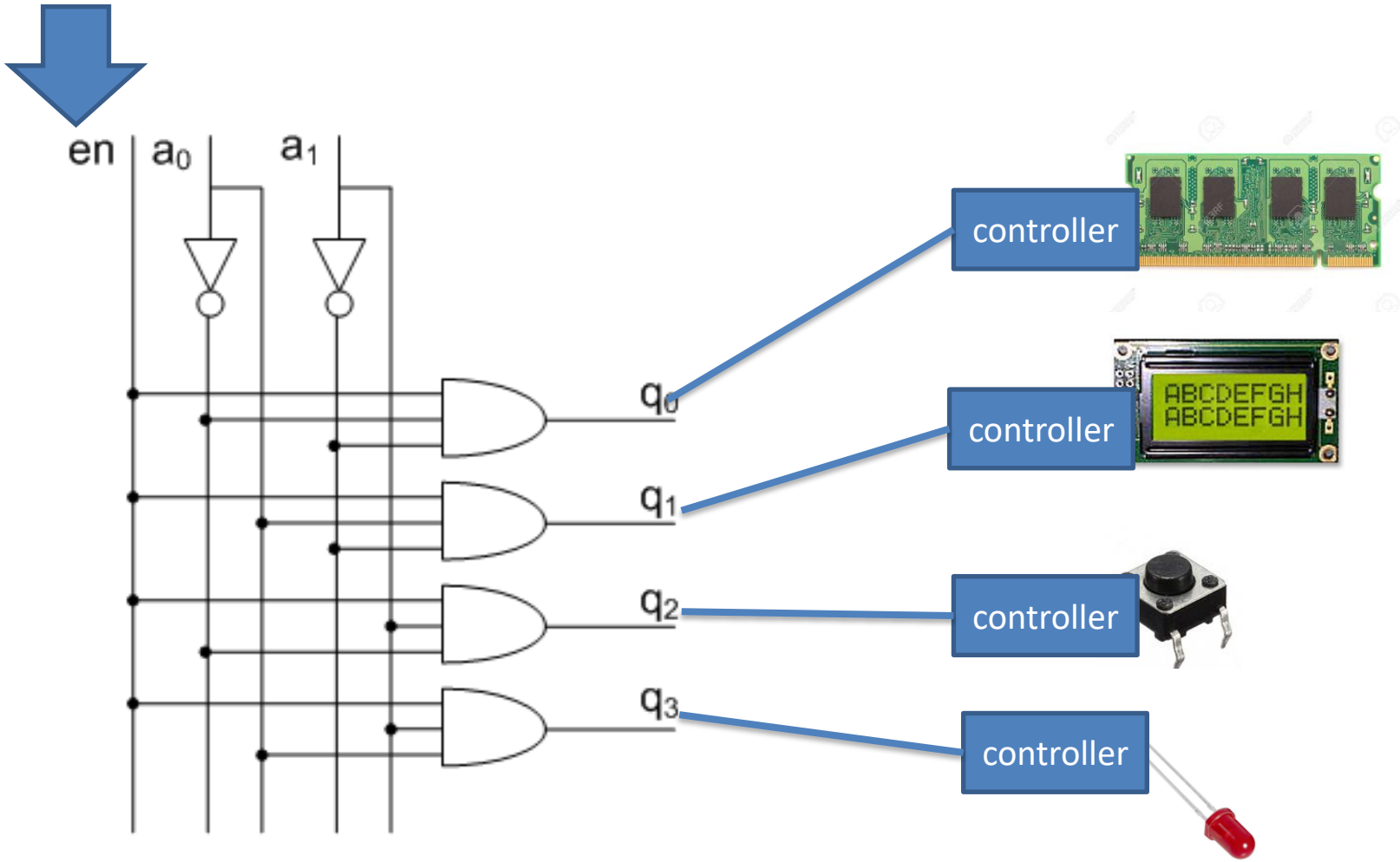
a0	a1	q0	q1	q2	q3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



Decoder circuit selects devices



Decoder circuit with enable

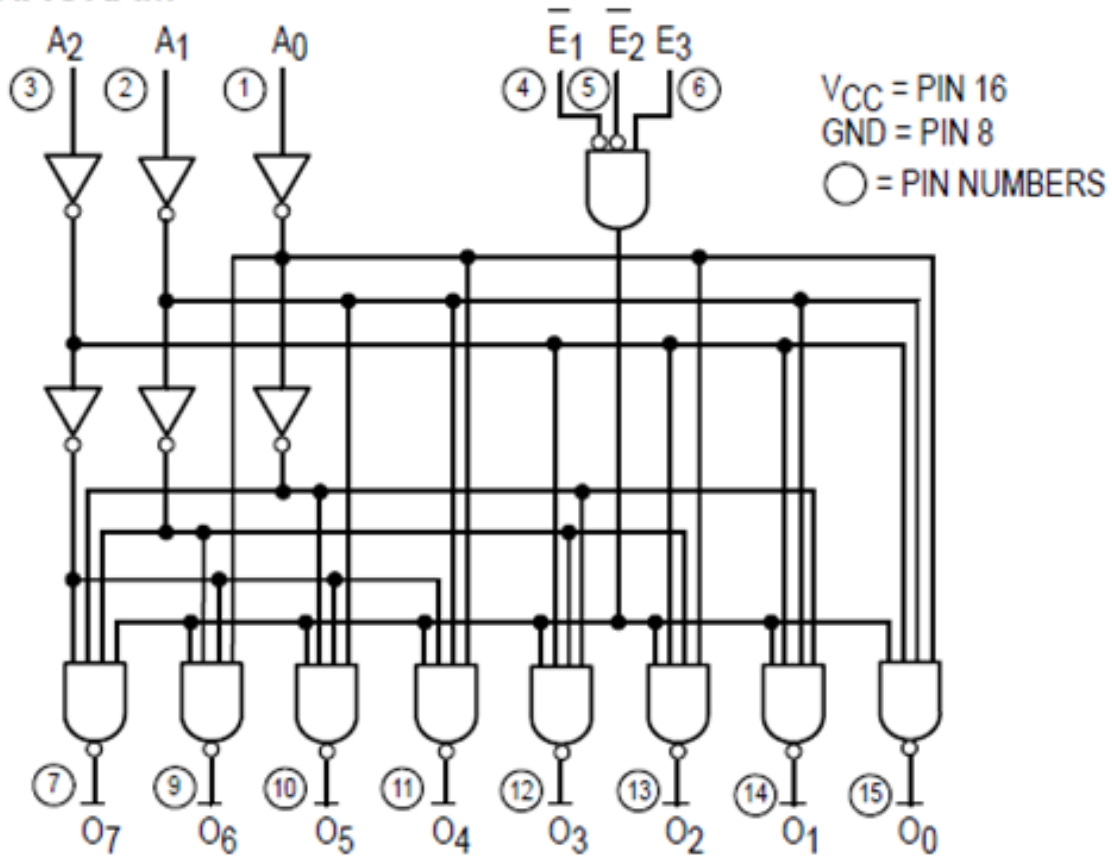


MOTOROLA

1-OF-8 DECODER/ DEMULTIPLEXER

SN54/74LS138

LOGIC DIAGRAM



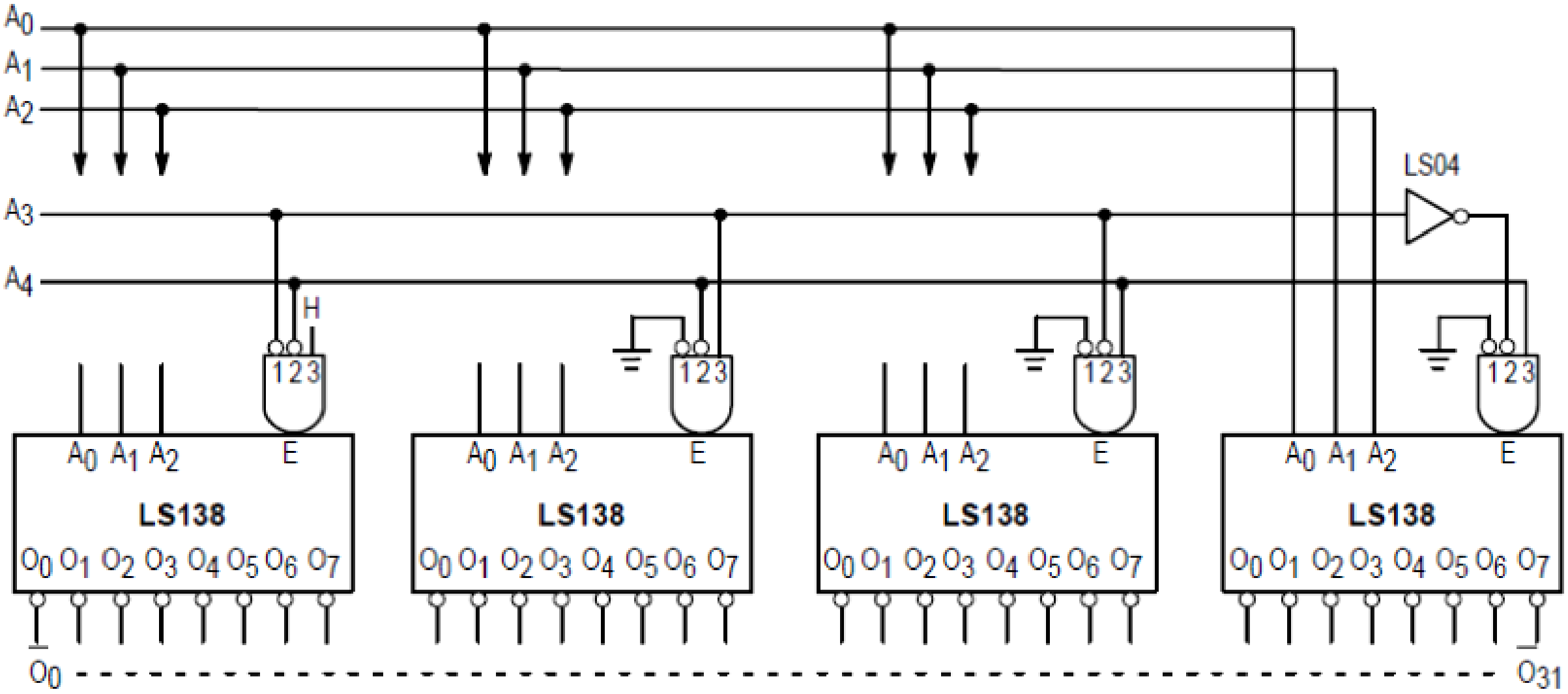
TRUTH TABLE

INPUTS						OUTPUTS							
$\overline{E_1}$	$\overline{E_2}$	$\overline{E_3}$	A_0	A_1	A_2	$\overline{O_0}$	$\overline{O_1}$	$\overline{O_2}$	$\overline{O_3}$	$\overline{O_4}$	$\overline{O_5}$	$\overline{O_6}$	$\overline{O_7}$
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

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

ใส่ขีดแท่งบนหรือ Active ที่ Low

Extended address



GUARANTEED OPERATING RANGES

Symbol	Parameter		Min	Typ	Max	Unit
V_{CC}	Supply Voltage	54 74	4.5 4.75	5.0 5.0	5.5 5.25	V
T_A	Operating Ambient Temperature Range	54 74	–55 0	25 25	125 70	°C
I_{OH}	Output Current — High	54, 74			–0.4	mA
I_{OL}	Output Current — Low	54 74			4.0 8.0	mA

DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE (unless otherwise specified)

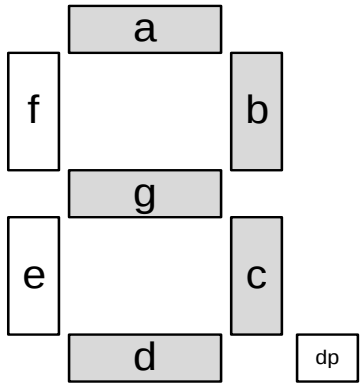
Symbol	Parameter		Limits Min	Limits Typ	Limits Max	Unit	Test Conditions	
V_{IH}	Input HIGH Voltage		2.0			V	Guaranteed Input HIGH Voltage for All Inputs	
V_{IL}	Input LOW Voltage	54			0.7	V	Guaranteed Input LOW Voltage for All Inputs	
		74			0.8			
V_{IK}	Input Clamp Diode Voltage			–0.65	–1.5	V	V_{CC} = MIN, I_{IN} = –18 mA	
V_{OH}	Output HIGH Voltage	54	2.5	3.5		V	V_{CC} = MIN, I_{OH} = MAX, V_{IN} = V_{IH} or V_{IL} per Truth Table	
		74	2.7	3.5		V		
V_{OL}	Output LOW Voltage	54, 74		0.25	0.4	V	I_{OL} = 4.0 mA	V_{CC} = V_{CC} MIN, V_{IN} = V_{IL} or V_{IH} per Truth Table
		74		0.35	0.5	V	I_{OL} = 8.0 mA	
I_{IH}	Input HIGH Current				20	µA	V_{CC} = MAX, V_{IN} = 2.7 V	
					0.1	mA	V_{CC} = MAX, V_{IN} = 7.0 V	
I_{IL}	Input LOW Current				–0.4	mA	V_{CC} = MAX, V_{IN} = 0.4 V	
I_{OS}	Short Circuit Current (Note 1)		–20		–100	mA	V_{CC} = MAX	
I_{CC}	Power Supply Current				10	mA	V_{CC} = MAX	

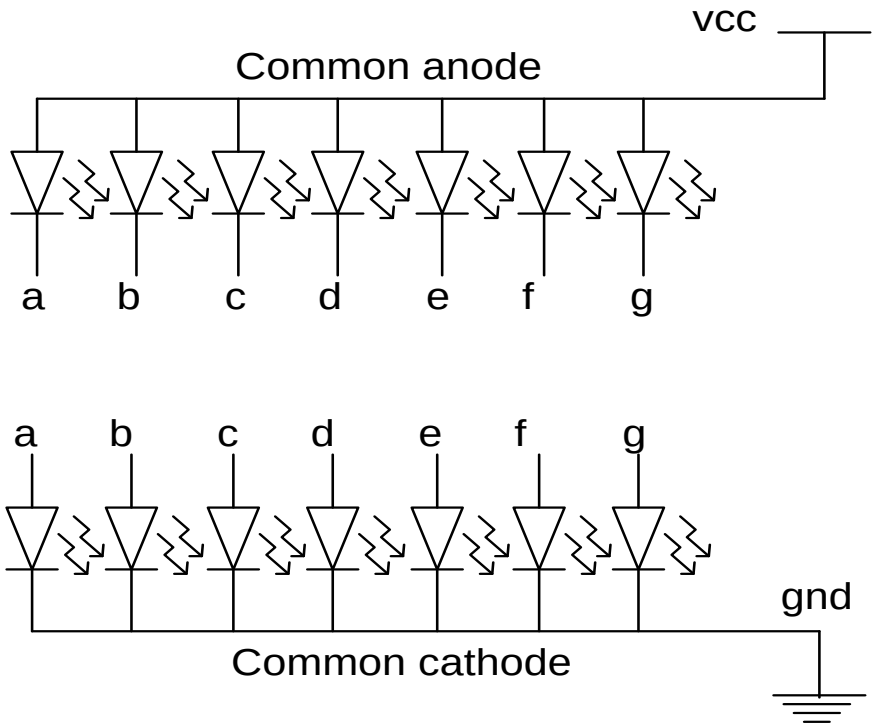
LED SEVEN SEGMENT
COMMON ANODE AND CATHODE

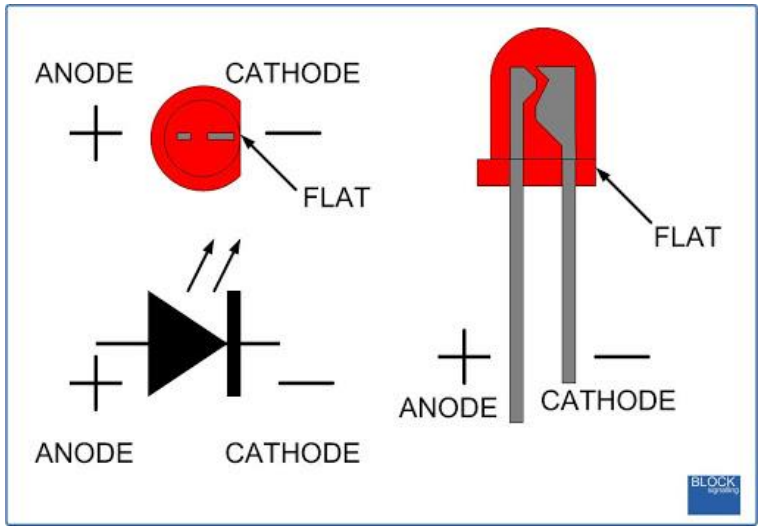
Student will able to explain the 7-segment between common cathode and anode.

Student will able to modify a new pattern displayed on the 7-segment.

Activity: 2.7 Plot your name on 7-segment







Alphabet present on 7-Segment

Once we have identified the decoded letter as a 5-bit value, we would like to display it. To do so, we can use a 7-segment LED display as shown in Fig. 1. For invalid addresses, we will display a dash (i.e. only segment G illuminated).

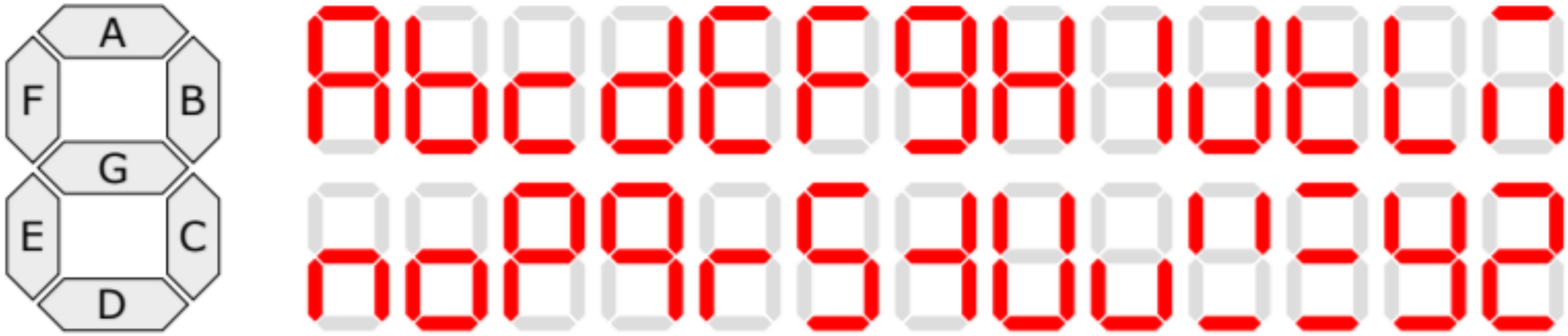
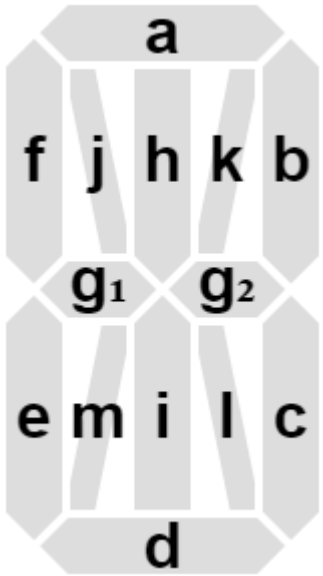
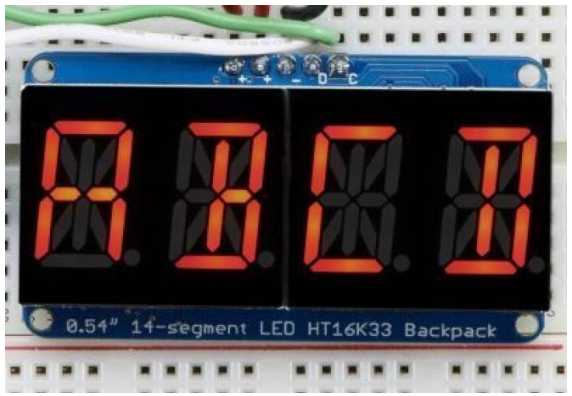
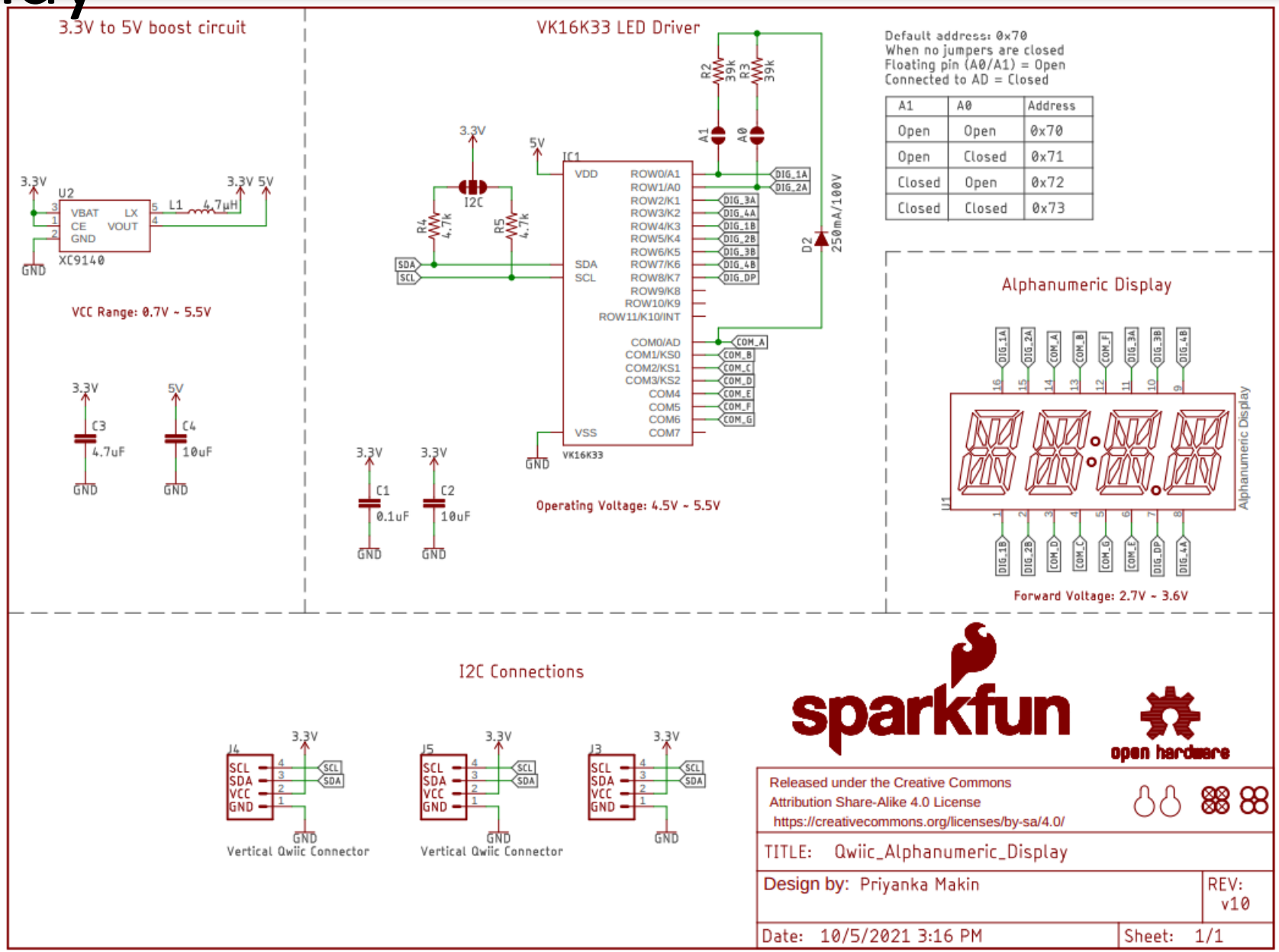


Figure 1: 7 segment display alphabet.

14-Segment display







Summary