CONTENTS

EXPERIMENTS USING KIT

CONTENTS

EXPERIMENTS USING MASM

Completed

10/10

10/9/10/19

8000		mov	AX, 1234H	B8, 34, 12	
8003		mov	BX, 1234H	BB, 34, 12	
8006		add	AX, BX	03, C3	add data in AX, BX
8008		mov	[8500], AX	A3, 0085	
800b		HLT		F4	

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	AX, 1234H	B8, 34, 12	
8003		mov	BX, 1234H	BB, 34, 12	
8006		sub	AX, BX		subtract the data in AX, BX
8008		mov	[8500], AX	A3, 00, 85	
800b		HLT		F4	

Sample I/P & O/P

Input AX - 1234
BX - 1234

Output 8500 - 00
8501 - 0D

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	AX, 1234H	B8, 34, 12	
8003		mov	BX, 1234H	BB, 34, 12	
8006		mov	DX, 0000H	BA, 00, 00	
8009		mul	BX	F7, F3	multiply data in DX, AX with BX
800b		mov	[8500], DX	A3, 00, 85	
800e		mov	[8502], DX	89, 16, 02, 85	
8012		HLT		F4	

Sample I/P & O/P

AX = 1234
BX = 1234

output.

8500 = ~~90~~
8501 = ~~005A~~
8502 = AB
8503 = ~~AB~~
8504 =

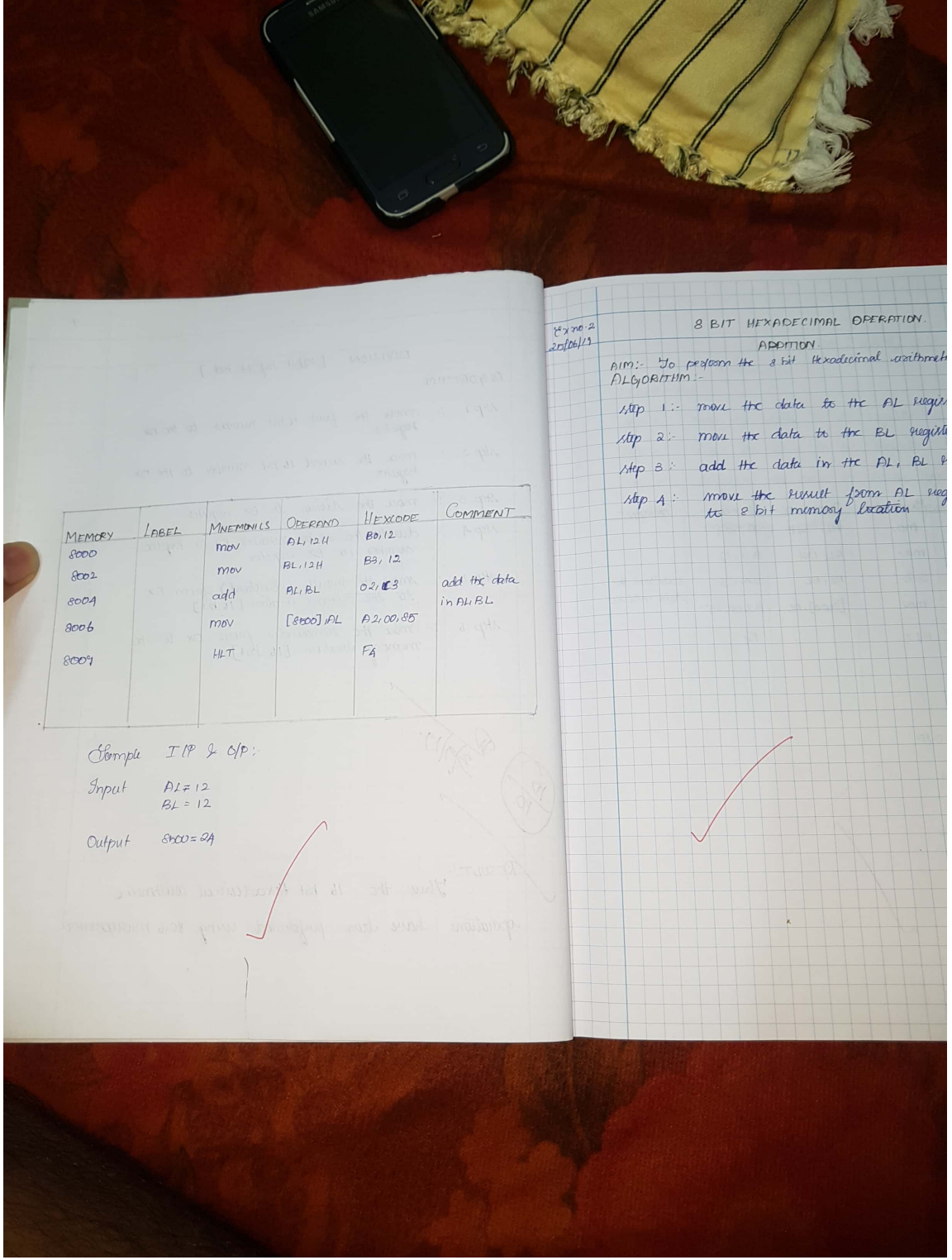
8500 = 90
8501 = 5A
8502 = AB
8503 = 01

LABEL	MEMORY	MNEMONICS	OPERAND	HEXCODE	COMMENT
	8000	mov	AX, 1234H	B8, 34, 12	
	8003	mov	BX, 1234H	BB, 34, 12	
	8006	mov	DX, 0000H	BA, 00, 00	
	8009	div	BX	F7, F3	divide data in DX, AX with BX
	800b	mov	[8500], AX	A3, 00, 85	
	800e	mov	[8502], DX	89, 16, 02, 85	
	8012	HLT		F4	

Sample I/P & O/P :

Input
AX - 1234
DX - 0006
BX - 1234

Output
AX - 0001 [Quotient]
DX - 0000 [Remainder]



Ex no: 2
20/08/19

8 BIT HEXADECIMAL OPERATION.

ADDITION.

AIM:- To perform the 8 bit Hexadecimal arithmeti

ALGORITHM:-

Step 1:- move the data to the AL regis

Step 2:- move the data to the BL regist

Step 3:- add the data in the AL, BL r

Step 4:- move the result from AL reg to 8 bit memory location

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	AL, 12H	B0, 12	
8002		mov	BL, 12H	B3, 12	
8004		add	AL, BL	02, C3	add the data in AL, BL
8006		mov	[8500], AL	A2, 00, 85	
8009		HLT		FA	

Sample I/P & O/P:

Input AL = 12
BL = 12

Output 8500 = 24

MEMORY	LABEL	MNEMONICS	OPERAND	HEX CODE	COMMENT
8000		mov	AL, 34H	B0, 34	
8002		mov	BL, 12H	B3, 12	
8004		sub	AL, BL	2A, C3	subtract the data in AL, BL
8006		mov	[8500], AL	A2, 00, 85	
8009		HLT		F4	

Sample I/P & O/P:

Input AL = 34
BL = 12

Output 8500 = 22

MEMORY	LABEL	MNEMONICS	OPERAND	HEX CODE	COMMENT
8000		MOV	AL, 12H	B0, 12	
8002		MOV	BL, 34H	B3, 34	
8004		mul	BL	F6, E3	multiply the data in BL with AL
8006		MOV	[8500], AX	A3, 00, 85	
8009		HLT		F4	

Sample I/P & O/P:-

Input AL = 12
BL = 34

Output 8500 = 08
8501 = 03

MEMORY	LABEL	MNEMONICS	OPERAND	HEX CODE	COMMENT
8000		mov	AH, 00H	B4, 00	
8002		mov	AL, 12H	B0, 12	
8004		mov	BL, 12H	B3, 12	
8006		DIV	BL	F6, F3	divides the data in AX with BL
8008		mov	[8500], AL	A2, 00, 85	
800b		mov	[8501], AH	86, 26, 01, 85	
800f		HLT		F4	

Sample I/P & O/P

Input
AH = 00
AL = 12
BL = 12

Output
AL = 01 [Quotient]
AH = 00 [Remainder]

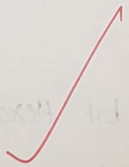
Ex. no. 3
29/06/19

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	AL, 12H	B0, 12	
8002		mov	BL, 12H	B3, 12	
8004		add	AL, BL	02, C3	
8006		daa		27	instruct for decimal operation (addition)
8007		mov	[8500], AL	A2, 00, 85	
800a		HLT		F4	

Sample I/P & O/P

Input AL = 12
BL = 12

Output 8500 = 24



MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	AL, 12H	B0, 12	
8002		mov	BL, 12H	B3, 12	
8004		sub	AL, BL	2A, C3	
8006		das		2F	convert the hexadecimal to decimal number in subtraction
8007		mov	[8500], AL	A2, 00, 85	
800a		HLT		F4	

Sample I/P & O/P

Input AL = 12
BL = 12

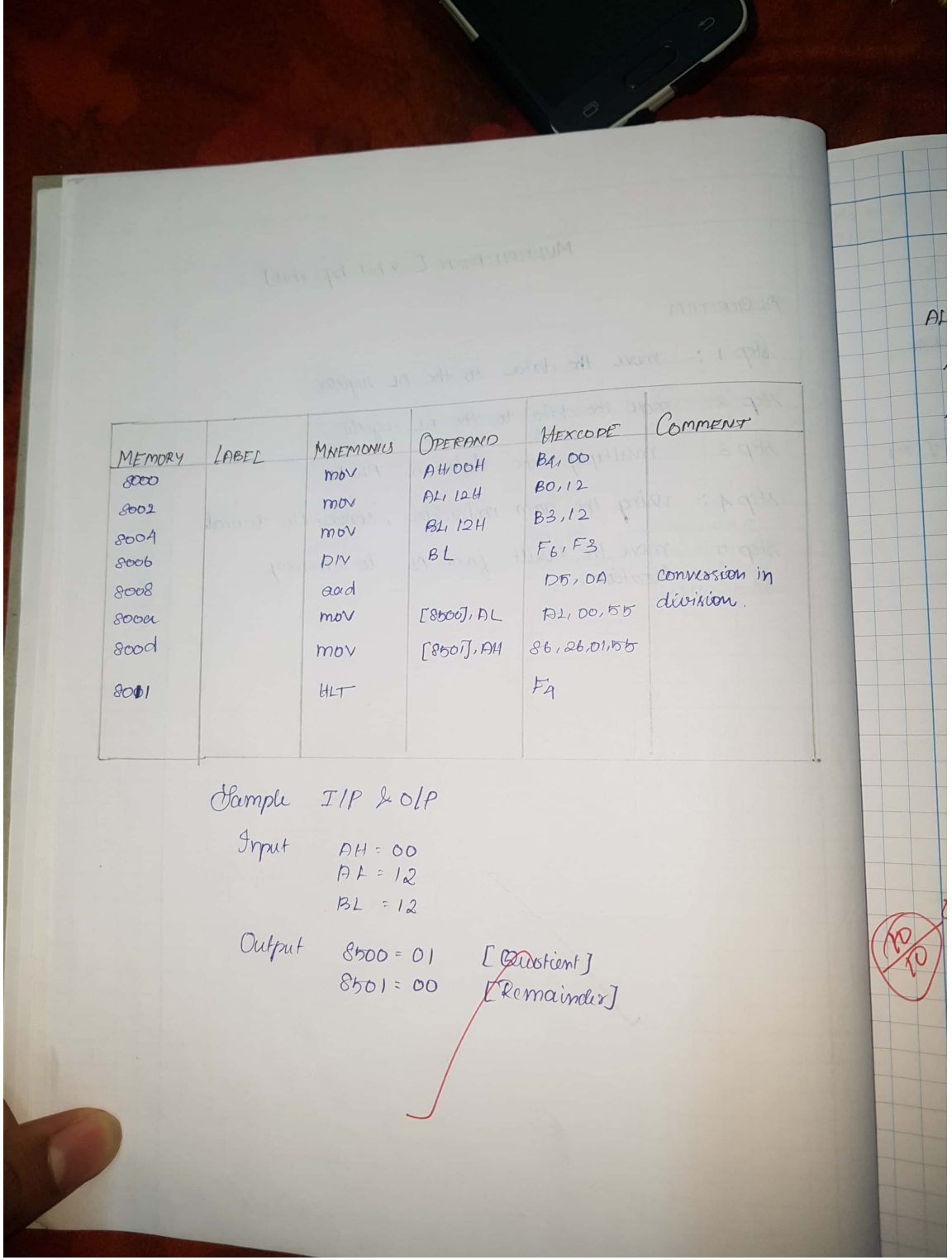
Output 8500 = 00

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	AL, 12H	B0, 12	
8002		mov	BL, 34H	B3, 34	
8004		mul	BL	F6, E3	
8006		aam		D4, 0A	Conversion of hexadecimal to decimal in multiplication
8008		mov	[8500], AX	A3, 00, 85	
800b		HLT		F4	

Sample I/P & O/P

Input AL = 12
BL = 34

Output 8500 = 08
8501 = 04



MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	AH, 00H	B4, 00	
8002		mov	AL, 12H	B0, 12	
8004		mov	BL, 12H	B3, 12	
8006		DIV	BL	F6, F3	
8008		aad		D5, 0A	conversion in division.
800a		mov	[8500], AL	A2, 00, 85	
800d		mov	[8501], AH	88, 26, 01, 85	
8011		HLT		F4	

Sample I/P & O/P

Input
AH = 00
AL = 12
BL = 12

Output
8500 = 01 [Quotient]
8501 = 00 [Remainder]

10/10

LARGEST NUMBER

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	SI, 8500	BE, 00, 85	
8002		mov	AL, [SI]	8A, 04	
8004		mov	CX, 0004H	B9, 04, 00	
8007		dec	CX	49	
8008	top:	inc	SI	~~3A, 04~~ 46	
8009		cmp	AL, [SI]	3A, 04	; compare the data in AL & [SI]
800B		JC	down	72, 04	
800D		Loop	top:	E2, F9	
800F		JMP	Last	EB, 04	
8011	down:	xCHG	AL, [SI]	86, 04	; exchanging if CY=1
8013		LOOP	TOP	E2, F3	
8015	Last:	mov	SI, 8600	BE, 00, 86	
8018		mov	[SI], AL	88, 04	
801A		HLT		F4	

Sample I/p & O/p:

8500 – 8503 ⇒ 05, 02, 07, 06

8600 ⇒ 07 ; largest number

SMALLEST NUMBER.

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	SI, 8500	BE, 00, 85	
8002		mov	AL, [SI]	8A, 04	
8004		mov	CX, 0004H	B9, 04, 00	
8007		dec	CX	49	
8008	top:	inc	SI	46	
8009		cmp	AL, [SI]	3A, 04	; compare the data in AL & [SI]
800B		JNC	down	73, 04	
800D		Loop	top	E2, F9	
800F		JMP	Last	EB, 04	
8011	down:	XCHG	AL, [SI]	86, 04	; exchanging if CY=0
8013		Loop	top	E2, F3	
8015	Last:	mov	SI, 8500	BE, 00, 86	
8018		mov	[SI], AL	88, 04	
801A		HLT		F4	

Sample I/p & O/p.

8500 - 8503 ⇒ 05, 02, 07, 06

8600 ⇒ 02 ; smallest number

ASCENDING ORDER.

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENTS
8000		mov	CH, 04H	B5, 04	
8002	UP2:	mov	CL, 04H	B1, 04	
8004		Lea	SI, 8500	8D, 36, 00, 85	
8008	UP1:	mov	AL, [SI]	8A, 04	
800A		mov	BL, [SI+1]	8A, 5C, 01	
800D		cmp	AL, BL	38, D8	; compare the data in AL, BL
800F		JC	down	72, 08	
8011		mov	DL, [SI+1]	8A, 54, 01	
8014		xchg	[SI], DL	86, 14	; exchange them if CY = 0
8016		mov	[SI+1], DL	88, 54, 01	
8019	down:	inc	SI	46	
801A		dec	CL	FE, C9	
801C		JNZ	UP1	75, EA	
801E		dec	CH	FE, CD	
8020		JNZ	UP2	75, E0	
8022		Hlt		F4	

INPUT:-

8500 - 8504 ⇒ 05, 04, 03, 02, 01

OUTPUT:-

8500 - 8504 ⇒ 01, 02, 03, 04, 05

DESCENDING ORDER

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENTS
8000		mov	CH, 04H	B5, 04	
8002	UP1:	mov	CL, 04H	B1, 04	
8004		Lea	SI, 8500	8d, 36, 00, 85	
8008	UP2:	mov	AL, [SI]	8A, 04	
800A		mov	BL, [SI+1]	8A, 5C, 01	
800D		cmp	AL, BL	38, D8	
800F		JNC	down	73, 08	
8011		mov	DL, [SI+1]	8A, 54, 01	
8014		XCHG	[SI], DL	86, 14	
8016		mov	[SI+1], DL	88, 54, 01	
8019	down	inc	SI	46	
801A		dec	CL	FE, C9	
801C		JNZ	UP1	75, EA	
801E		dec	CH	FE, CD	
8020		JNZ	UP2	75, E0	
8022		Hlt		F4	

INPUT

8500 - 8504 ⇒ 01, 02, 03, 04, 05

OUTPUT

8500 - 8504 ⇒ 05, 04, 03, 02, 01

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT
8000		mov	SI, 8100	BE, 00, 81	
8003		mov	DI, 8200	BF, 00, 82	
8006		mov	CX, 0005H	B9, 05, 00	
8009		CLD		FC	
800A		REP movsb		F3, A4	move the data from SI to DI
800C		HLT		FA	

SAMPLE I/P : & O/P

8100 → 55
8101 → 64
8102 → 70
8103 → 12
810A → 10

8200 → 55
8201 → 64
8202 → 70
8203 → 12
8204 → 10

MEMORY	LABEL	MNEMONICS	OPERAND	HEX CODE	COMMENT
8000		mov	SI, 9000	BE, 00, 90	
8003		mov	AL, [SI]	8A, 04	
8005		mov	BH, 01H	B7, 01	; multiplication factor
8007		and	AL, 000FH	24, 0F	
8009		mul	BH	F6, E7	
800B		mov	DX, AX	8B, D0	
800D		mov	BH, 0AH	B7, 0A	
800F		mov	AL, [SI]	8A, 04	
8011		~~mov~~ and	AL, ~~000~~F0H	24, F0	
8013		mul	BH	F6, E7	; multiplication factor
8015		add	AX, DX	03, C2	
8017		inc	SI	46	
8018		mov	[SI], AX	89, 04	; storing ~~the~~ hexadecimal value.
801A		HLT		F4	

Sample I/P + O/P

9000 → 89

9001 → 1D [output]

9002 → 00

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE	COMMENT.
8000		mov	SI, 9000	BE, 00, 90	
8003		mov	BL, 0AH	B3, 0A	
8005		mov	AH, 00H	B4, 00	
8007		mov	AL, [SI]	8A, 04	
8009	top:-	cmp	AL, BL	38, D8	; comparing AL, BL
800B		Jb	Store	72, 07	
800D		div	BL	F6, F3	
800F		inc	SI	46	
8010		mov	[SI], AH	88, 24	; Stor the remainder to SI
8012		Jmp	top	EB, F5	
8014	Store:	inc	SI	46	
8015		mov	[SI], AL	88, 04	Stor the quotient to SI
8017		HLT		F4	

Sample input & output.

9000 → 0A

9000 → 00

9002 → 01

HEXADECIMAL TO DECIMAL NUMBER.

step 1 :- move the data's address to SI
move 00H to AH, 0AH to BL

step 2 :- move the [SI] to AL

step 3 :- compare AL & BL

step 4 :- if AL < BL go to step ③, else goto step ⑤

step 5 :- divide AL by BL

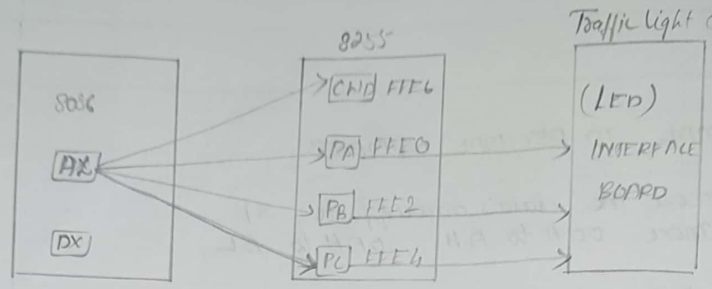
step 6 :- increment SI and move AH to [SI]

step 7 :- go to step ③

step 8 :- increment SI and move AL to [SI]

10/10 22/9/19

Thus the number conversions decimal to hexadecimal and hexadecimal to decimal have been performed using 8086 microprocessor.



MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		mov	AL, 80H	B0, 80
8002		mov	DX, FFE6H	BA, E6, FF
8005		out	DX, AL	EE
8006	Repeat Again :	mov	SI, 8038	BE, 38, 80
8009	next state:	mov	AL, [SI]	8A, 04
800B		mov	DX, FFE0H	BA, E0, FF
800E		out	DX, AL	EE
800F		inc	SI	46
8010		mov	DX, FFE2H	BA, E2, FF
8013		out mov	DX, AL, [SI]	8A, 04
8015		out	DX, AL	EE
8016		inc	SI	46
8017		mov	DX, FFE4H	BA, E4, FF
801A		mov	AL, [SI]	8A, 04
801C		out	DX, AL	EE
801D		inc	SI	46
801E		Call	delay	E8, 09, 00
8021		cmp	SI, 8056	81, FE, 56, 80
8025		jnz	nextstate	75, E2
8027		jmp	Repeat Again	E9, DC, FF
802A	delay :	mov	CX, 00FFH	B9, FF, 00
802D	Top1 :	push	CX	51
802E		mov	CX, 03FFH	B9, FF, 03
8031	Top2 :	NOP		90
8032		Loop	Top2	E2, FD
8034		pop	CX	59
8035		Loop	Top1	E2, F6
8037		Ret		C3

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8038		DB	10H, 81H, 7AH	10H, 81H, 7AH
803B		DB	44H, 44H, 44H	44H, 44H, 44H
803E		DB	08H, 11H, E5H	08H, 11H, E5
8041		DB	44H, 44H, 44H	44, 44, 44
8044		DB	81H, 10H, 00H	81, 10, 00
8047		DB	44H, 44H, 44H	44, 44, 44
804A		DB	11H, 08H, B5H	11, 08, B5
804D		DB	44H, 44H, 44H	44, 44, 44
8050		DB	88H, 88H, 00H	88, 88, 00
8053		DB	44H, 44H, 44H	44, 44, 44
8056		DB	00	00

Ex.no 8
3/9/12

WAVE GENERATION USING 8086 MICROPROCESSOR.

Aim:-

To generate wave forms such as square, triangle, sawtooth using 8086.

ALGORITHM:

[SQUARE WAVE GENERATION]

step 1 :- Initialise the control word.

step 2 :- set the 3 ports as output ports in control word register.

step 3 :- move 00 to AL as low state, move AL to the port A of address FFE0H

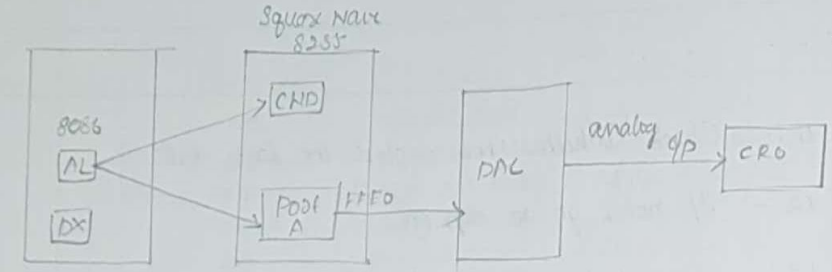
step 4 :- call delay for suitable frequency

step 5 :- move FFH to AL as high value. move AL to the port A of address FFE0H

step 6 :- call delay

step 7 :- declare the delay

step 8 :- Repeat the process again.



Square wave form →

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE
8000		mov	AL, 80H	B0, 80
8002		mov	DX, FFE6H	BA, E6, FF
8005		out	DX, AL	EE
8006	Repeat:	mov	AL, 00H	B0, 00
8008		mov	DX, FFE0H	BA, E0, FF
800B		out	DX, AL	EE
800C		call	delay	E8, 0B, 00
800F		mov	AL, FFH	B0, FF
8011		mov	DX, FFE0H	BA, E0, FF
8014		out	DX, AL	EE
8015		call	delay	E8, 02, 00
8018		jmp	Repeat	EB, FC
801A	delay:	mov	CX, 00FF	B9, FF, 00
801D	Top:	dec	CX	49
801E		jnz	Top	75, FD
8020		Ret		C3

TRIANGULAR WAVEFORM →

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		mov	AL, 80H	B0, 80
8002		mov	DX, FFE6H	BA, E6, FF
8005		Out	DX, AL	EE
8006	Repeat:	~~AH~~ mov	AL, 00H	B0, 00
8008		mov	DX, FFE0H	BA, E0, FF
800b	again1:	out	DX, AL	EE
800C		inc	AL	FE, C0
800e		Cmp	AL, FF	3C, FF
8010		JB	Again1	72, F9
8012	again2:	out	DX, AL	EE
8013		dec	AL	FE, C8
8015		Cmp	AL, 00H	3C, 00
8017		JNZ	Again2	75, F9
8019		jmp	Repeat	EB, EB

[TRIANGULAR WAVE GENERATION]

step 1:- Initialise the control word

step 2:- set the three ports as output port by its control register

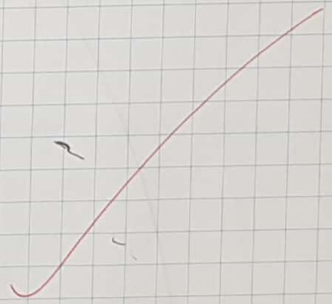
step 3:- move 00 H to AL as low state, move AL to port A of address FFE0 H

step 4:- inc AL and go to step 3, if AL ≠ FF H

step 5:- move the value in AL (FFH) to port A of address FFE0H as high state

step 6:- decrement AL and go to step 5 if AL ≠ 00 H

step 7:- Repeat once again



SAWTOOTH WAVEFORM -

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		mov	AL, 80H	B0, 80
8002		mov	DX, FFE6H	BA, E6, FF
8005		out	DX, AL	EE
8006	repeat:	mov	AL, 00H	B0, 00
8008	increment:	mov	DX, FFE0H	BA, E0, FF
800B		out	DX, AL	EE
800C		inc	AL	EE, C0
800E		cmp	AL, FFH	36, FF
8010		jb	increment	72, F6
8012		jmp	repeat	EB, F2

[SAWTOOTH WAVE GENERATION]

step 1 :- Initialise the control word.

step 2 :- set the 3 ports as the output ports in control word register.

step 3 :- move 00H to AL and AL to the port A of address OFFE0H

step 4 :- inc AL and move to port A if AL ≠ FFH

step 5 :- move FF to AL , move AL to port A.

step 6 :- Repeat the process if AL = FFH

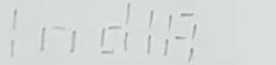
10/10 22/4/10/19

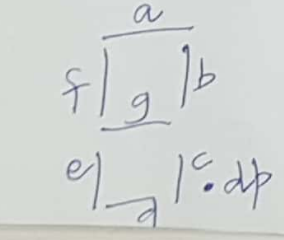
RESULT :-

Thus the wave forms has been generated using DAC

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE
8000	START:	mov	AL, 00H	B0, 00H
8002		mov	DX, FFEBH	BA, EB, FF
8005		out	DX, AL	EE
8006		mov	AL, C1H	B0, 90C1
8008		out	DX, AL	EE
8009		mov	AL, 90H	B0, 90
800B		out	DX, AL	EE
800C	Repeat:	mov	CX, 0005 H	B9, 0F, 00
800F		mov	SI, 8500	BE, 00, 85
8012		mov	DX, FFE9H	BA, E9, FF
8015	next:	mov	AL, [SI]	BA, 04
8017		out	DX, AL	EE
8018		Call	delay	EB, 05, 00
801B		inc	SI	46
801C		Loop	next	E2, F7
801E		JMP	Repeat/START	EB, EC / E0
8020	delay:	mov	BX, 0005H	BB, 05, 00
8023	Subagain:	mov	DI, FFFFH	BF, FF, FF
8026	Subagain1:	DEC dec	DI	4F
8027		JNZ	Subagain 1	75, FD
8029		dec	BX	AB
802A		JNZ	Subagain	75, F7
802C		Ret		C3

Sample I/p & o/p





Ex no. 9
3/9/19

KEYBOARD AND DISPLAY CONTROL - ROLLING FASHION

AIM :-

To display the characters in Rolling fashion. interfacing keyboard and display control

ALGORITHM :-

step 1 :- Initialise the control word.

step 2 :- move the control word to display and keyboard controller of address OFFEBH.

step 3 :- move C1 to OFFEBH and move 90 to FFEB

step 4 :- move th no of characters to CX

step 5 :- move the address to SI where the seven segment code of the characters are written.

step 6 :- move the [SI] to AL and AL to FFE9H.

step 7 :- call delay and increment SI, go to step (6)

step 8 :- If CX = 00H, go to step (4). for the repetition.

step 9 :- declare the delay

10/10 4/10/19

RESULT :-

Thus the characters are displayed in Rolling fashion.

Ex.no.10
17/9/15

STEPPER MOTOR WITH 8086

MEMORY	LABEL	MNEMONICS	OPERAND	HEXCODE
8000		mov	DX, FFE6H	BA, E6, FF
8003		mov	AL, 80H	B0, 80
8005		Out	DX, AL	EE
8006				
8009		mov	DX, FFE0H	BA, E0, FF
800B	back:	mov	AL, 88H	B0, 88
800C		Out	DX, AL	EE
800F		call	delay	E8, 04, 00
8011		ROR/ROL	AL, 01	D0, C8/D0
8013		Jmp	back	E8, F8
8016	delay:	mov	CX, 4000H	B9, 00, 40
8018	Repeat:	Repeat loop	loop Repeat.	E2, FE
		Ret		C3

AIM:-

To rotate the stepper motor in clockwise / anticlockwise direction continously.

ALGORITHM:

step 1:- Initialise the control word.

step 2:- Set the ports as output port

step 3:- move the control word 88 to port A

step 4:- call delay

step 5:- Rotate the motor in either clockwise or anticlockwise direction

step 7:- declare the delay.

RESULT:-

Thus the stepper motor interfaced with 8086 microprocessor

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		mov	DX, FFE6H	BA, E6, FF
8003		mov	AL, 89H	B0, B9
8005		mov	DX, AL	EE
8006		mov	CL, 00H	B1, 00
8008		mov	DX, FFE4H	BA, E4, FF
800B	LOOP1:	in	AL, DX	EC
800C		and	AL, 02H	24, 02
800E		JZ	LOOP1	74, FB
8010		call	DELAY	E8, 24, 00
8013	LOOP2:	in	AL, DX	EC
8014		AND	AL, 02H	24, 02
8016		JNZ	LOOP2	75, FB
8018	LOOP3:	mov	AL, CL	8A, C1
801A		mov	DX, FFE0H	BA, E0, FF
801D		out	DX, AL	EE
801E		call	delay	E8, 16, 00
8021		mov	DX, FFE4H	BA, E4, FF
8024		in	AL, DX	EC
8025		and	AL, 01H	24, 01
8027		JZ	FINISH	74, 04
8029		Inc	CL	FE, C1
802B		JMP	LOOP3	EB, EB
802D	FINISH:	mov	AL, CL	8A, C1
802F		mov	AH, 00H	B4, 00
8031		call	0B0A:FF00	9A, 0A, 0B, 00, FF
8036		HLT		F4
8037	DELAY:	mov	BX, 05FFH	BB, FF, 05
803A		DEC	BX	4B
803B		JNZ	LOOP4	75, FD
803D		RET		C3

Ex.no.11
8/9/19

ANALOG TO DIGITAL SIGNAL CONVERSION

AIM:

To perform the conversion from analog to digital signal using 8086 microprocessor.

ALGORITHM

step 1 :- Interface the ADC with 8086 microprocessor.

step 2 :- Initialise the ports A, B as output port and port C as input port by writing control word in register (CWR)

step 3 :- Read the input from C

step 4 :- And the result with 2

step 5 :- If the result of AND is 0, jump to Loop1 (step 3)

step 6 :- Else call delay

step 7 :- Read the input from ports

step 8 :- And with 2, if the result of AND is 0, jump to step 7

step 9 :- Call a delay

step 10 :- move the contents of CL to AL register

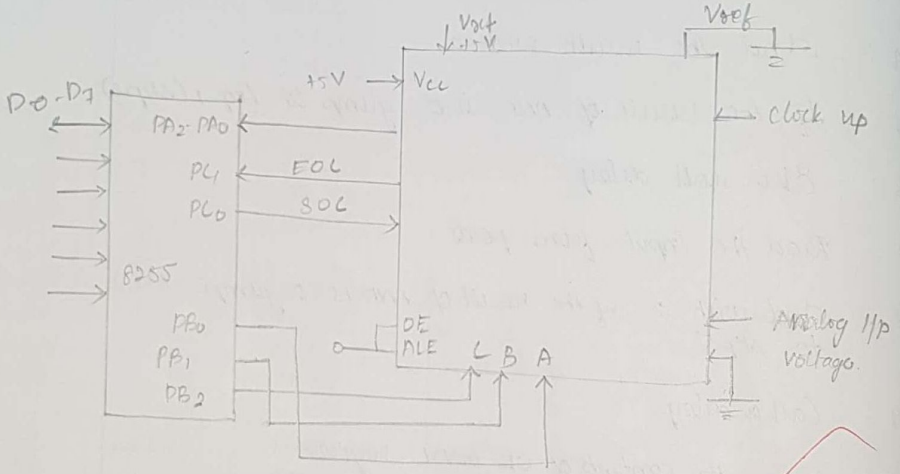
step 11 :- move the output to port A

step 12 :- call a delay. move the input from port C to AL

step 13 :- AND the AL with 01, if AND is 0 jump to finish. (step 15)

SAMPLE I/P-O/P

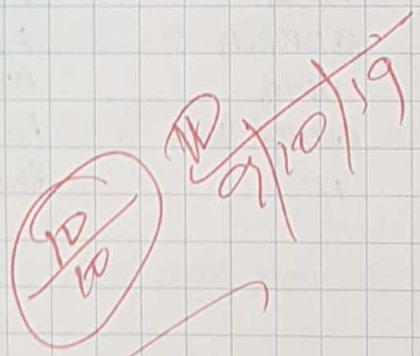
ANALOG I/P	DIGITAL O/P	HEX VALUE
0V	0000 0000	00
0.5V	0001 1100	1C
1.02V	0011 0100	34
1.51V	0101 0100	54
2.00V	0111 0000	70
2.50V	1000 1100	8C
3.00V	1010 1000	A8



step 14 :- the ~~more~~ increment CL and go to step 10

step 15 :- Call display function to show the output

step 16 :- delay is maintained for (05FFH) duration



RESULT :-

Thus the conversion from analog to digital signal has been executed using 8086 microprocessor.

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		mov	DPTR, #8500	90, 00, 85
8003		movx	A, @DPTR	E0
8004		mov	R0, A	F8
8005		inc	DPTR	A3
8006		movx	A, @DPTR	E0
8007		ADDC	A, R0	38
8008		mov	R1, A	F9
8009		inc	DPTR	A3
800A		movx	A, @DPTR	E0
800B		mov	R0, A	F8
800C		inc	DPTR	A3
800D		movx	A, @DPTR	E0
800E		addc	A, R0	38
800F		inc	DPTR	A3
8010		movx	@DPTR, A	F0
8011		inc	DPTR	A3
8012		movx	@DPTR, A	F0
8013		SJMP	FF	80, FF

SAMPLE I/P & O/P.

8500 - 44 - LSB operand 1
8501 - 22 - MSB operand 2
8502 - 00 - MSB operand 1
8503 - 22 - MSB operand 2
8504 - 88 - MSB result
8505 - 66 - LSB result

Ex.no:12
8/10/25

ARITHMETIC OPERATION, 2'COMPLEMENT, CONVERSION IN 8051 microcontroller

AIM:-

To perform the arithmetic operations and basic operations using 8051 microcontroller.

ALGORITHM:-

ADDITION

step 1 :- store the data starting from address 8500

step 2 :- move the data from #8500 to DPTR and to A

step 3 :- move the data from A to R0

step 4 :- increment DPTR and the move the data from DPTR to A

step 5 :- Add the data in A and R0 with carry

step 6 :- move the result to R1

step 7 :- increment DPTR and move the data from DPTR to A and to R0

step 8 :- increment DPTR and move the data from DPTR to A

step 9 :- add A, R0 with carry

step 10 :- increment DPTR move the MSB result to DPTR

step 11 :- increment DPTR move the LSB result to it

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		MOV	DPTR, #8500	90, 85, 00
8003		MOVX	A, @DPTR	E0
8004		MOV	R0, A	F8
8005		INC	DPTR	A3
8006		MOVX	A, @DPTR	E0
8007		SUBB	A, R0	98
8008		MOV	R1, A	F9
8009		INC	DPTR	A3
800A		MOVX	A, @DPTR	E0
800B		MOV	R0, A	F8
800C		INC	DPTR	A3
800D		MOVX	A, @DPTR	E0
800E		SUBB	A, R0	98
800F		INC	DPTR	A3
8010		MOVX	@DPTR, A	F0
8011		INC	DPTR	A3
8012		MOV	A, R1	E9
8013		MOVX	@DPTR, A	F0
8014		SJMP	FE	80, FF

Sample I/P & O/P

8500 - 02 - LSB operand1
8501 - 04 - LSB operand2
8502 - 00 - MSB operand1
8503 - 00 - MSB operand2
8504 - 00 - MSB Result
8505 - 02 - LSB Result.

SUBTRACTION.

step 1 :- move the starting address of data to DPTR.

step 2 :- move the data from DPTR to A and to R0

step 3 :- increment DPTR and move the data from DPTR to A

step 4 :- subtract the data in A & R0 with borrow

step 5 :- move the LSB result to R1

step 6 :- move the data from DPTR to A & to R0

step 7 :- move the data from DPTR to A

step 8 :- subtract the data in A & R0 with borrow

step 8 :- increment DPTR, move the MSB result to it

step 9 : increment DPTR, move the LSB result to it.

MULTIPLICATION

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000				
8003		mov	DPTR, # 8100	90, 81, 00
8004		movx	A, @DPTR	E0
8005		mov	F0, A	F5, F0
8006		mov	R0, A	F8
8007		mul	AB	A4
8008		inc	DPTR	A3
8009		movx	@DPTR, A	F0
800A		mov	F0, A	F5, F0
800B		mul	AB	E8
800C		inc	DPTR	A4
800D		movx	@DPTR, A	A3, F0
		SJMP	FE	80, FF

Sample i/p - o/p.

8100 - 02
8101 - 04 [square]
8102 - 08 [cube]

DIVISION

MEMORY	LABEL	MENMONICS	OPERAND	OPCODE
8000		mov	DPTR, # 8500	90, 85, 00
8003		movx	A, @DPTR	E0
8004		mov	F0, A	F5, F0
8005		inc	DPTR	A3
8006		movx	A, @DPTR	E0
8007		div	AB	84
8009		inc	DPTR	A3
800A		movx	@ DPTR, A	F0
800B		inc	DPTR	A3
800C		mov	A, F0	E5, F0
800E		movx	@DPTR, A	F0
800F		SJMP	FE	80, FE

Sample i/p - o/p

8500 - 02 [divisor]
8501 - 04 [dividend]
8502 - 02 [Quotient]
8503 - 00 [Remainder]

2' COMPLEMENT

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		MOV	DPTR, #8500	90, 85, 00
8003		MOVX	A, @ DPTR	F0
8004		CPL	A	F4
8005		ADD	A, #01	24, 01
8007		INC	DPTR	F0
8008		MOVX	@DPTR, A	F5, A0
800A		SJMP	FF	80, FE

Sample I/P + O/P

8500 → 02

8501 → FE [2' complement]

ASCII TO BCD

MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8000		MOV	DPTR, #8500	90, 85, 00
8003		MOV	A, #35	74, 35
8005		CLR	C	C3
8006		SUBB	A, #30	94, 30
8008		CLR	C	C3
8009		SUBB	A, #0A	94, 0A
800B		JC	STR	40, 0A
800D		MOV	A, #FF	74, FF
800F		SJMP	L1	80, 02
8011	STR:	ADD	A, #0A	24, 0A
8013	L1:	MOVX	@ DPTR, A	F0
8014		SJMP	FF	80, FF

Sample I/P + O/P

8500 → 35 [ASCII]

8501 → 05 [BCD]

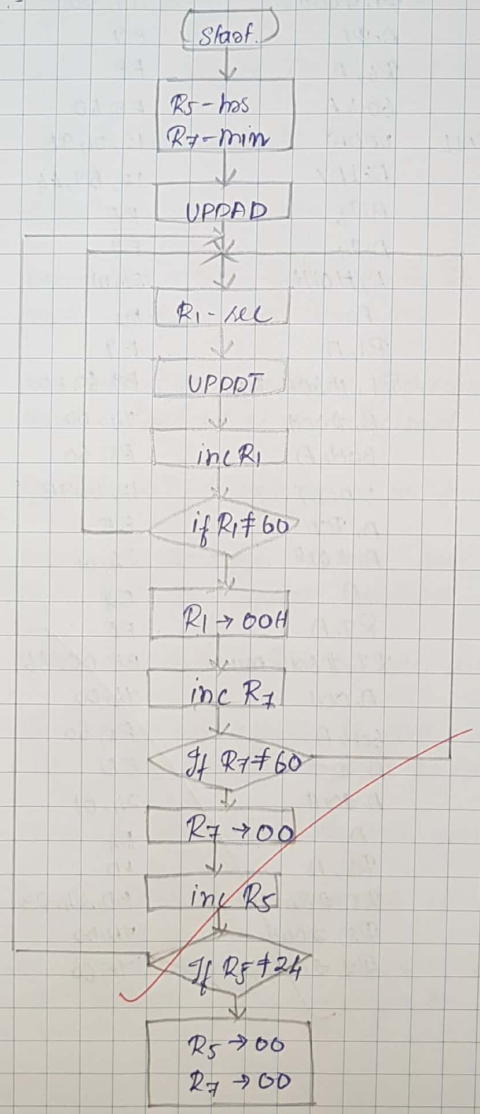
Ex.no.13
4/10/19

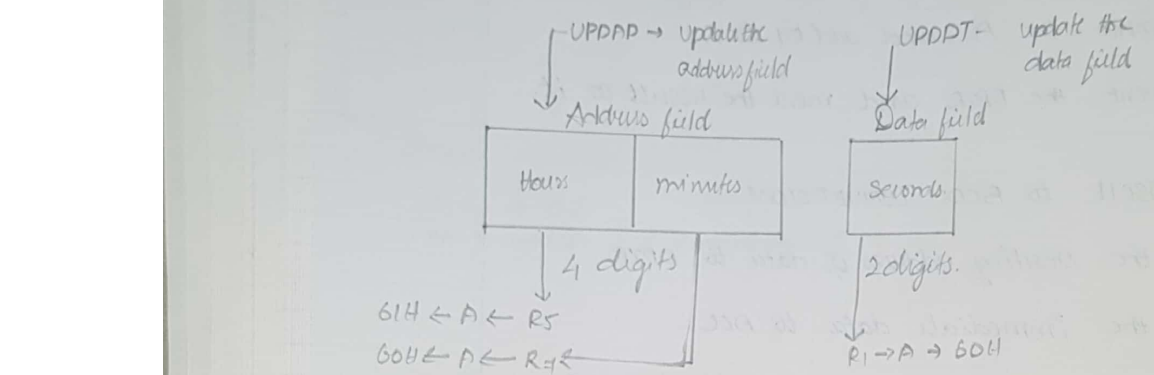
DIGITAL CLOCK IN 8051 MICROCONTROLLER.

AIM:

To display 24 hours digital clock in keyboard mode in 8051 microcontroller.

FLOWCHART :-





ADDRESS	LABEL	MNEMONICS	OPERAND	OPCODE
8900		MOV	R5, #23H	7D, 23
8902		MOV	R7, #58H	7F, 58
8904	LOOP:	MOV	A, R5	ED
8905		MOV	61H, A	F5, 61
8907		MOV	A, R7	EF
8908		MOV	60H, A	F5, 60
890A		LCALL	UPDAD	12, 02, 0B
890D	UP:	MOV	R1, #00H	79, 00
890F	RPT:	MOV	A, R1	E9
8910		MOV	R6, A	FE
8911		MOV	60H, A	F5, 60
8913		~~MOV~~ LCALL	UPDDT	12, 01, 9B
8916		LCALL	DELAY	12, B9, 44
8919		MOV	A, R6	EE
891A		MOV	A, R1	E9
891B		ADD	A, #01H	24, 01
891D		DA	A	D4
891E		MOV	R1, A	F9
891F		CJNE	R1, #60H, RPT	B9, 60, ED
8922		MOV	A, #00H	74, 00
8924		MOV	60H, A	F5, 60
8926		LCALL	UPDDT	12, 01, 9B
8929		MOV	A, R7	EF
892A		add	A, #01H	24, 01
892C		DA	A	D4
892D		MOV	R7, A	FF
892E		CJNE	R7, #60H, MIN	BF, 60, 36
8931		MOV	A, 00H	74, 00
8933		MOV	60H, A	F5, 60
8935		MOV	A, R5	ED
8937		add	A, #01H	24, 01
8938		DA	A	D4
8939		MOV	R5, A	FD
893A		CJNE	R5, #24, HRS	BD, 24, 23
893D		MOV	R5, #00H	7D, 00
893F		MOV	R7, #00H	7F, 00

ALGORITHM:-

step 1 :- move the hours to R5 register

step 2 :- move the minutes to R7 register

step 3 :- move the data in R5 to 61H to display it

step 4 :- move the data in R7 to 60H to display it

step 5 :- update the address field

step 6 :- store the seconds in R1

step 7 :- move the data from R1 to R6 and to 60H and update the data field

step 8 :- call delay for 1 second increment the second by 1 and convert it to decimal value using DA

step 9 :- If data in R1 is not equal to 60, goto step 7

step 10 :- move the data 00 to ACC and to location 60H and call the update ~~address~~ field data field

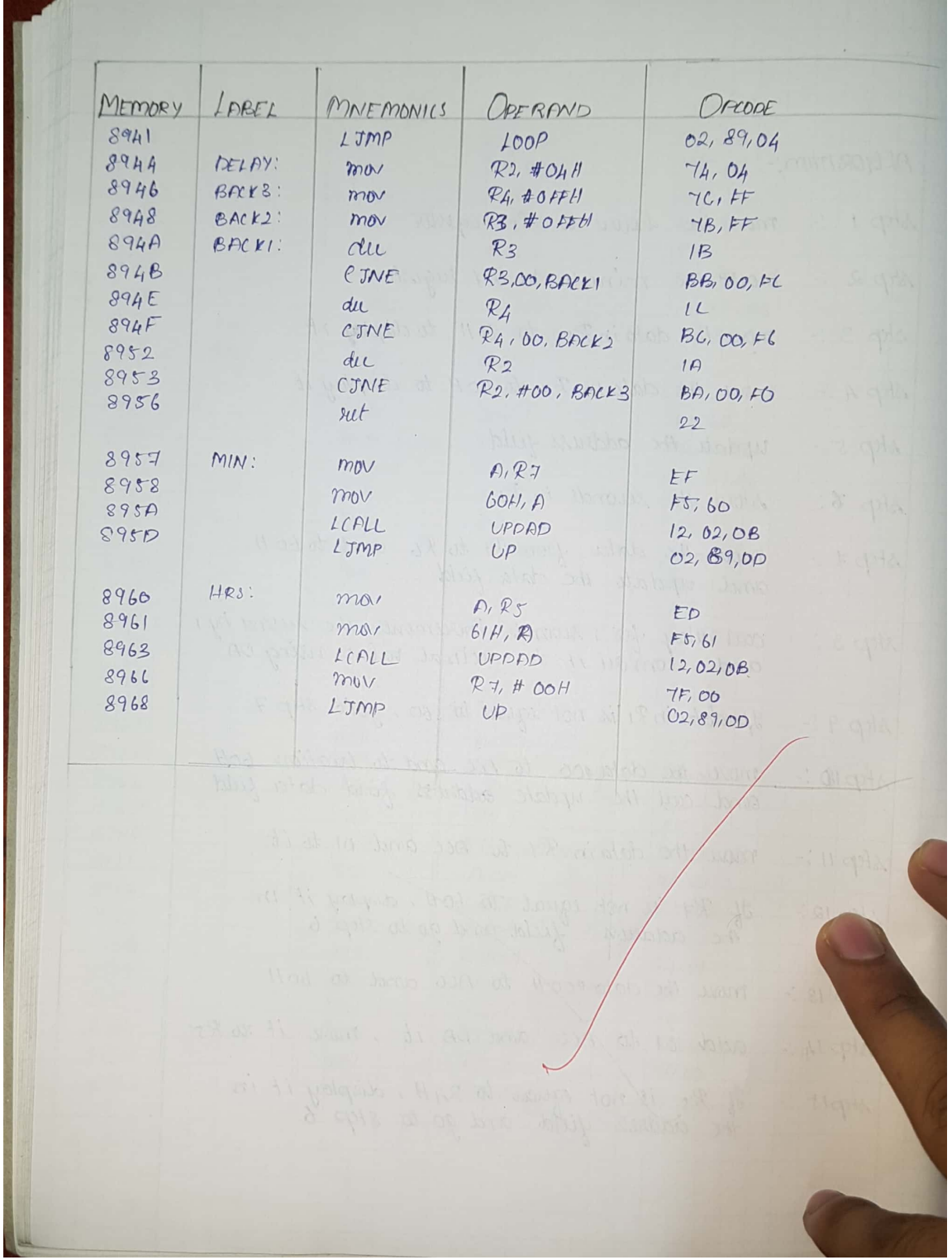
step 11 :- move the data in R7 to ACC and 01 to it

step 12 :- If R7 is not equal to 60H, display it in the address field and go to step 6

step 13 :- move the data 00H to ACC and to 60H

step 14 :- add 01 to ACC and DA it, move it to R5

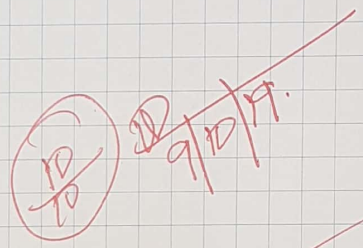
step 15 :- If R5 is not equal to 24H, display it in the address field and go to step 6



MEMORY	LABEL	MNEMONICS	OPERAND	OPCODE
8941		LJMP	LOOP	02, 89, 04
8944	DELAY:	mov	R2, #04H	7A, 04
8946	BACK3:	mov	R4, #0FFH	7C, FF
8948	BACK2:	mov	R3, #0FFH	7B, FF
894A	BACK1:	dec	R3	1B
894B		CJNE	R3, 00, BACK1	BB, 00, FC
894E		dec	R4	1C
894F		CJNE	R4, 00, BACK2	BC, 00, F6
8952		dec	R2	1A
8953		CJNE	R2, #00, BACK3	BA, 00, F0
8956		ret		22
8957	MIN:	mov	A, R7	EF
8958		mov	60H, A	F5, 60
895A		LCALL	UPDAD	12, 02, 0B
895D		LJMP	UP	02, 89, 0D
8960	HRS:	mov	A, R5	ED
8961		mov	61H, A	F5, 61
8963		LCALL	UPDAD	12, 02, 0B
8966		mov	R7, # 00H	7F, 00
8968		LJMP	UP	02, 89, 0D

step 16 :- Repeat the whole process.

step 17 :- declare the delay for 1 second.



RESULT :-

Thus the 24 hrs digital clock in keyboard mode is display in 8051 microcontroller.