SRI SIVASUBRAMANIYA NADAR COLLEGE OF ENGINEERING

(An Autonomous Institution – Affiliated to Anna University)

Rajiv Gandhi Salai, Kalavakkam – 603 110

LABORATORY RECORD

Name : Sai Prakash.R

Reg.No : 195002098

Dept. : IT Sem. : IV Sec: B

SRI SIVASUBRAMANIYA NADAR COLLEGE OF ENGINEERING

(An Autonomous Institution – Affiliated to Anna University)

Rajiv Gandhi Salai, Kalavakkam – 603 110

BONAFIDE CERTIFICATE

Certified that this is the bonafide record of the practical work done for the

Microprocessor and Microcontroller Lab

by

Name: Sai Prakash. R

Register Number: 195002098

of Department of Information Technology

SSN College of Engineering , kalavakkam, Chennai.

during the academic year 2020-2021

Staff In_Charge Head of Department, IT

Submitted for the End Semester Examination held at SSN College of Engineering on ----------

Internal Examiner External Examiner

INDEX

Name: Sai Prakash. R Reg.No: 195002098

Sem: IV Sec: B

8 Bit Arithmetic Operations using Trainer Kit

Ex No. : 1 Date: 02/02/2021

a. 8 bit Hexadecimal Addition

Aim:

To write an assembly language program to implement 8 bit hexadecimal addition.

Algorithm:

(a) Load input 1 into AL register
(b) Load input 2 into BL register
(c) Add the contents of AL & BL
(d) Move the result from AL register into memory
(e) Stop

Program:

```
MOV   AL,44H

MOV   BL,55H

ADD   AL,BL

MOV   [8500],AL

HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AL,44H	B0,44	Move 44H into AL register
8002		MOV	BL,55H	B3,55	Move 55H into BL register
8004		ADD	AL,BL	02,C3	Add the contents of AL&BL
8006		MOV	[8500],AL	A2,00,85	Move the sum from AL to memory
8009		HLT		F4	Stop

Sample input & output:

Input:

Input 1--- 44H

Input 2--- 55H

Output:

[8500] --- 99

Result:

Thus 8 bit hexadecimal addition has been performed successfully and the output is displayed.

b. 8 bit Hexadecimal Subtraction

Aim:

To write an assembly language program to implement 8 bit hexadecimal subtraction.

Algorithm:

(a) Load input 1 into AL register
(b) Load input 2 into BL register
(c) Subtract the contents of BL from AL
(d) Move the result from AL register into memory
(e) Stop

Program:

```
MOV   AL,55H

MOV   BL,44H

SUB    AL,BL

MOV   [8500],AL

HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AL,55H	B0,55	Move 44H into AL register
8002		MOV	BL,44H	B3,44	Move 55H into BL register
8004		SUB	AL,BL	2A,C3	Subtract BL from AL
8006		MOV	[8500],AL	A2,00,85	Move the contents of AL to memory
8009		HLT		F4	Stop

Sample input & output:

Input:

Input 1--- 55H

Input 2--- 44H

Output:

[8500] --- 11

Result:

Thus 8 bit hexadecimal subtraction has been performed successfully and the output is displayed.

c. 8 bit Hexadecimal Multiplication

Aim:

To write an assembly language program to implement 8 bit hexadecimal multiplication.

Algorithm:

(a) Load input 1 into AL register
(b) Load input 2 into BL register
(c) Multiply the contents of AL and BL
(d) Move the result from AX register into memory
(e) Stop

Program:

```
MOV   AL,22H
MOV   BL,01H
MUL   BL
MOV   [8500],AX
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AL,22H	B0,22	Move 22H into AL register
8002		MOV	BL,01H	B3,01	Move 01H into BL register
8004		MUL	BL	F6,E3	Multiply contents of AL&BL
8006		MOV	[8500],AX	A3,00,85	Move the product from AX to memory
8009		HLT		F4	Stop

Sample input & output:

Input:

Input 1--- 22H

Input 2--- 01H

Output:

[8500] --- 22

[8501] --- 00

Result:

Thus 8 bit hexadecimal multiplication has been performed successfully and the output is displayed.

d. 8 bit Hexadecimal Division

Aim:

To write an assembly language program to implement 8 bit hexadecimal division.

Algorithm:

(a) Load 00H into AH register
(b) Load input 1 into AL register
(c) Load input 2 into BL register
(d) Divide the contents of AX by BL
(e) Move the quotient from AL register into memory
(f) Move the remainder from AH to memory
(g) Stop

Program:

```
MOV   AH,00H
MOV   AL,55H
MOV   BL,01H
DIV   BL
MOV   [8500],AL
MOV   [8501],AH
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AH,00H	C6,C6,00	Move 00H into AH register
8002		MOV	AL,55H	B0,55	Move 55H into AL register
8005		MOV	BL,01H	B3,01	Move 01H into BL register
8007		DIV	BL	F6,F3	Divide AX by BL
8009		MOV	[8500],AL	A2,00,85	Move quotient from AL to memory
800C		MOV	[8501],AH	88,26,01,85	Move remainder from AH to memory
8010		HLT		F4	Stop

Sample input & output:

Input:

AH --- 00H

Input 1(AL) --- 55H

Input 2(BL) --- 01H

Output:

[8500] --- 55 (Quotient)

[8501] --- 00 (Remainder)

Result:

Thus 8 bit hexadecimal division has been performed successfully and the output is displayed.

16 Bit Arithmetic Operations using Trainer Kit

Ex No. : 2 Date: 09/02/2021

a. 16 bit Hexadecimal Addition

Aim:

To write an assembly language program to implement 16 bit hexadecimal addition.

Algorithm:

(a) Load input 1 into AX register
(b) Load input 2 into BX register
(c) Add the contents of AX & BX
(d) Move the result from AX register into memory
(e) Stop

Program:

```
MOV   AX, 1234H
MOV   BX, 1234H
ADD   AX,BX
MOV   [8500], AX
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AX,1234H	B8,34,12	Move 1234H into AX register
8003		MOV	BX,1234H	BB,34,12	Move 1234H into BX register
8006		ADD	AX,BX	03,C3	Add the contents of AX&BX
8008		MOV	[8500],AX	A3,00,85	Move the sum from AX to memory
800B		HLT		F4	Stop

Sample input & output:

Input:

Input 1--- 1234H

Input 2--- 1234H

Output:

[8500] --- 68

[8501] --- 24

Result:

Thus 16 bit hexadecimal addition has been performed successfully and the output is displayed.

b. 16 bit Hexadecimal Subtraction

Aim:

To write an assembly language program to implement 16 bit hexadecimal subtraction.

Algorithm:

(a) Load input 1 into AX register
(b) Load input 2 into BX register
(c) Subtract the contents of AX & BX
(d) Move the result from AX register into memory
(e) Stop

Program:

```
MOV   AX, 1234H
MOV   BX, 1234H
SUB    AX,BX
MOV   [8500], AX
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AX,1234H	B8,34,12	Move 1234H into AX register
8003		MOV	BX,1234H	BB,34,12	Move 1234H into BX register
8006		SUB	AX,BX	2B,C3	Subtract the contents of AX&BX
8008		MOV	[8500],AX	A3,00,85	Move the contents of AX to memory
800B		HLT		F4	Stop

Sample input & output:

Input:

Input 1--- 1234H

Input 2--- 1234H

Output:

[8500] --- 00

[8501] --- 00

Result:

Thus 16 bit hexadecimal subtraction has been performed successfully and the output is displayed.

c. 16 bit Hexadecimal Multiplication

Aim:

To write an assembly language program to implement 16 bit hexadecimal multiplication.

Algorithm:

(a) Load input 1 into AX register
(b) Load input 2 into BX register
(c) Load 0000 into DX register
(d) Multiply the contents of AX & BX
(e) Store the value in AX register (lower 16 bit) into memory
(f) Store the value in DX register (upper 16 bit) into memory
(g) Stop

Program:

```
MOV   AX,1234H
MOV   BX, 1234H
MOV   DX, 0000H
MUL   BX
MOV   [8500],AX
MOV   [8502],DX
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AX,1234H	B8,34,12	Move 1234H into AX register
8003		MOV	BX,1234H	BB,34,12	Move 1234H into BX register
8006		MOV	DX,0000H	BA,00,00	Move 0000H into DX register
8009		MUL	BX	F7,E3	Multiply the contents of AX&BX
800B		MOV	[8500],AX	A3,00,85	Move the value in AX to memory
800E		MOV	[8502],DX	89,16,02,85	Move the value in DX to memory
8012		HLT		F4	Stop

Sample input & output:

Input:

Input 1--- 1234H

Input 2--- 1234H

Output:

[8500] --- 90

[8501] --- 5A

[8502] --- 4B

[8503] --- 01

Result:

Thus 16 bit hexadecimal multiplication has been performed successfully and the output is displayed.

d. 16 bit Hexadecimal Division

Aim:

To write an assembly language program to implement 16 bit hexadecimal division.

Algorithm:

(a) Load input 1 into AX register
(b) Load input 2 into BX register
(c) Load 0000 into DX register
(d) Divide the contents of DX(upper 16 bit) & AX (lower 16 bit) by BX
(e) Store the value in AX register (quotient) into memory
(f) Store the value in DX register (remainder) into memory
(g) Stop

Program:

```
MOV   AX,1234H
MOV   BX, 1234H
MOV   DX, 0000H
DIV     BX
MOV   [8500],AX   //quotient
MOV   [8502],DX   //remainder
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AX,1234H	B8,34,12	Move 1234H into AX register
8003		MOV	BX,1234H	BB,34,12	Move 1234H into BX register
8006		MOV	DX,0000H	BA,00,00	Move 0000H into DX register
8009		DIV	BX	F7,F3	Divide DX&AX by BX
800B		MOV	[8500],AX	A3,00,85	Move the value in AX to memory
800E		MOV	[8502],DX	89,16,02,85	Move the value in DX to memory
8012		HLT		F4	Stop

Sample input & output:

Input:

Numerator --- 00001234H

Denominator --- 1234H

Output:

[8500] --- 01

[8501] --- 00

[8502] --- 00

[8503] --- 00

//[8500]&[8501] — Quotient

//[8502]&[8503] — Remainder

Result:

Thus 16 bit hexadecimal division has been performed successfully and the output is displayed.

8 Bit Decimal Operations using Trainer Kit

Ex No. : 3 Date: 16/02/2021

a. 8 bit decimal Addition

Aim:

To write an assembly language program to implement 8 bit decimal addition.

Algorithm:

(a) Load input 1 into AL register
(b) Load input 2 into BL register
(c) Add the contents of AL & BL
(d) Decimal adjust after addition on AL
(e) Move the result from AL register into memory
(f) Stop

Program:

```
MOV  AL,12H
MOV  BL,12H
ADD  AL,BL
DAA
MOV  [8500],AL
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AL,12H	B0,12	Move 12H into AL register
8002		MOV	BL,12H	B3,12	Move 12H into BL register
8004		ADD	AL,BL	02,C3	Add the contents of AL&BL
8006		DAA		27	Decimal adjust after addition on AL
8007		MOV	[8500],AL	A2,00,85	Move the sum from AL to memory
800A		HLT		F4	Stop

Sample input & output:

Input:

Input 1 --- 12H

Input 2 --- 12H

Output:

[8500] --- 24

Result:

Thus 8 bit decimal addition has been performed successfully and the output is displayed.

b. 8 bit decimal Subtraction

Aim:

To write an assembly language program to implement 8 bit decimal subtraction.

Algorithm:

(a) Load input 1 into AL register
(b) Load input 2 into BL register
(c) Subtract the contents of BL from AL
(d) Decimal adjust after subtraction on AL
(e) Move the result from AL register into memory
(f) Stop

Program:

```
MOV   AL,12H
MOV   BL,12H
SUB    AL,BL
DAS
MOV   [8500],AL
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AL,12H	B0,12	Move 12H into AL register
8002		MOV	BL,12H	B3,12	Move 12H into BL register
8004		SUB	AL,BL	2A,C3	Subtract the contents of BL from AL
8006		DAS		2F	Decimal adjust after subtraction on AL
8007		MOV	[8500],AL	A2,00,85	Move the difference from AL to memory
800A		HLT		F4	Stop

Sample input & output:

Input:

Input 1 --- 12H

Input 2 --- 12H

Output:

[8500] --- 00

Result:

Thus 8 bit decimal subtraction has been performed successfully and the output is displayed.

c. 8 bit decimal Multiplication

Aim:

To write an assembly language program to implement 8 bit decimal multiplication.

Algorithm:

(a) Load input 1 into AL register
(b) Load input 2 into BL register
(c) Multiply the contents of AL&BL
(d) Ascii adjust after multiplication
(e) Move the result from AX register into memory
(f) Stop

Program:

```
MOV   AL,12H
MOV   BL,03H
MUL   BL
AAM
MOV   [8500],AX
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AL,12H	B0,12	Move 12H into AL register
8002		MOV	BL,03H	B3,03	Move 03H into BL register
8004		MUL	BL	F6,E3	Multiply the contents of AL & BL
8006		AAM		D4,0A	Ascii adjust after multiplication
8008		MOV	[8500],AX	A3,00,85	Move the product from AX to memory
800B		HLT		F4	Stop

Sample input & output:

Input:

Input 1 --- 12H

Input 2 --- 03H

Output:

[8500] --- 36

[8501] --- 00

Result:

Thus 8 bit decimal multiplication has been performed successfully and the output is displayed.

c. 8 bit decimal Division

Aim:

To write an assembly language program to implement 8 bit decimal division.

Algorithm:

(a) Load 00H into AH register
(b) Load input 1 into AL register
(c) Load input 2 into BL register
(d) Ascii adjust after division
(e) Divide the contents of AX by BL
(f) Move the result from AX register into memory
(g) Stop

Program:

```
MOV   AH,00H
MOV   AL,12H
MOV   BL,12H
AAD
DIV    BL
MOV   [8500],AX
HLT
```

Table:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	AH,00H	C6,C6,00	Move 00H into AH register
8003		MOV	AL,12H	B0,12	Move 12H into AL register
8005		MOV	BL,03H	B3,12	Move 12H into BL register
8007		AAD		DS,0A	Ascii adjust after division
8009		DIV	BL	F6,F3	Divide the contents of AX by BL
800B		MOV	[8500],AX	A3,00,85	Move the contents from AX to memory
800E		HLT		F4	Stop

Sample input & output:

Input:

AH --- 00H

Input 1 --- 12H

Input 2 --- 12H

Output:

[8500] --- 01 //Quotient

[8501] --- 00 //Remainder

Result:

Thus 8 bit decimal division has been performed successfully and the output is displayed.

8 Bit Arithmetic Operations using MASM

Ex No. : 4 Date: 23/02/2021

a. 8 bit hexadecimal Addition

Aim:

To write an assembly language program to implement 8 bit hexadecimal addition.

Algorithm:

(a) Define all the necessary constants and the values to be added.
(b) Load the first value into AL.
(c) Add the two 8-bit numbers using add instruction.
(d) Convert the binary answer to human readable format using xlat.
(e) Display the sum.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     db  005h

n2     db  006h

result db  000h

msg    db  'The result is '

asciir db  2 dup(?)

       db  cr, lf, '$'

data ends


code segment

       assume cs:code, ds:data

start:
```

```
        mov ax, data
        mov ds, ax
        mov al, n1
        add al, n2
        lea bx, table
        mov result, al
        lea si, asciir
        add si, 1
        mov al, result
        and al, 0fh
        xlat
        mov [si], al
        dec si
        mov al, result
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg
        int 21h
quit:   mov al, 00h
        mov ah, 04ch
        int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 005h

n2 --- 006h

Output:

The result is 0B

Result:

Thus 8 bit hexadecimal addition has been performed successfully and the output is displayed.

b. 8 bit hexadecimal Subtraction

Aim:

To write an assembly language program to implement 8 bit hexadecimal subtraction.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Load the first value into the AL register.
(d) Perform the operation using sub instruction.
(e) Convert the result to human readable format using xlat instruction.
(f) Display the result

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     db  006h

n2     db  005h

result db  000h

msg    db  'The result is '

asciir db  2 dup(?)

       db  cr, lf, '$'

data ends


code segment

       assume cs:code, ds:data

start:

       mov ax, data

       mov ds, ax
```

```
        mov al, n1
        sub al, n2
        lea bx, table
        mov result, al
        lea si, asciir
        add si, 1
        mov al, result
        and al, 0fh
        xlat
        mov [si], al
        dec si
        mov al, result
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg
        int 21h
quit:   mov al, 00h
        mov ah, 04ch
        int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 006h

n2 --- 005h

Output:

The result is 01

Result:

Thus 8 bit hexadecimal subtraction has been performed successfully and the output is displayed.

c. 8 bit hexadecimal Multiplication

Aim:

To write an assembly language program to implement 8 bit hexadecimal multiplication.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Load the first value into the al register.
(d) Perform the operation using mul instruction.
(e) Convert AX to human readable format using xlat instruction.
(f) Display the result

Program:

```
cr      equ     0dh
lf      equ     0ah
data    segment
table   db      '0123456789abcdef'
n1      db  005h
n2      db  006h
res     dw  00000h
msg     db      'the result is '
as_res  db      4 dup(?)
        db      cr,lf,'$'
data    ends
code    segment
        assume cs:code, ds:data
start:  mov ax,data
        mov ds,ax
        mov al, n1
        mov bh, n2
        mul bh
```

```
mov res,ax
lea bx,table
lea si, as_res
add si,3
mov ax,res
and ax,0000fh
xlat
mov [si],al
dec si
mov ax,res
and ax,000f0h
mov cl,04h
shr ax,cl
xlat
mov [si],al
dec si
mov ax,res
and ax,00f00h
mov cl,08h
shr ax,cl
xlat
mov [si],al
dec si
mov ax,res
and ax,0f000h
mov cl,0ch
shr ax,cl
```

```
        xlat
        mov [si],al
        mov ah,09h
        lea dx,msg
        int 21h
        mov ah,04ch
        int 21h

code    ends
end     start
```

Sample input & output:

Input:

n1 --- 005h

n2 --- 006h

Output:

The result is 1E

Result:

Thus 8 bit hexadecimal multiplication has been performed successfully and the output is displayed.

d. 8 bit hexadecimal Division

Aim:

To write an assembly language program to implement 8 bit hexadecimal division.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Load dividend to AL.
(d) Load AH with 0.
(e) Perform the operation using div instruction.
(f) Convert AH and AL to human readable format using xlat instruction.
(g) Display AH as remainder and AL as quotient.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     dw  02Ah

n2     db  006h

rem    db  000h

quo    db  000h

msg1   db  'The quotient is '

asquo  db  2 dup(?)

       db  cr, lf, '$'

msg2   db  'The remainder is '

asrem  db  2 dup(?)

       db  cr, lf, '$'

data ends
```

```
code segment
        assume cs:code, ds:data
start:
        mov ax, data
        mov ds, ax
        mov ax, n1
        div n2
        lea bx, table
        mov quo, al
        mov rem, ah
        lea si, asquo
        add si, 1
        mov al, quo
        and al, 0fh
        xlat
        mov [si], al
        dec si
        mov al, quo
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg1
        int 21h
```

```
        lea si, asrem
        add si, 1
        mov al, rem
        and al, 0fh
        xlat
        mov [si], al
        dec si
        mov al, rem
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg2
        int 21h
quit:   mov al, 00h
        mov ah, 04ch
        int 21h

code ends
end start
```

Sample input & output:

Input:

n1 --- 02Ah

n2 --- 006h

Output:

The quotient is 07

The remainder is 00

Result:

Thus 8 bit hexadecimal division has been performed successfully and the output is displayed.

16 Bit Arithmetic Operations using MASM

Ex No. : 5 Date: 02/03/2021

a. 16 bit Hexadecimal Addition

Aim:

To write an assembly language program to implement 16 bit hexadecimal addition.

Algorithm:

(a) Define all the necessary constants and the values to be added.
(b) Add the two 16-bit numbers using add instruction.
(c) Convert the binary answer to human readable format using xlat.

Program:

```
cr equ 0dh
lf equ 0ah

data segment
table  db  '0123456789ABCDEF'
n1     dw  05555h
n2     dw  05555h
result dw  00000h
msg    db  'The result is '
asciir db  4 dup(?)
       db  cr, lf, '$'
data ends

code segment
       assume cs:code, ds:data
start:
       mov ax, data
```

```
mov ds, ax
mov ax, n1
add ax, n2
lea bx, table
mov result, ax
lea si, asciir
add si, 3
mov ax, result
and ax, 0000fh
xlat
mov [si], al
dec si
mov ax, result
and ax, 000f0h
mov cl, 04h
shr al, cl
xlat
mov [si], al
dec si
mov ax, result
and ax, 00f00h
mov cl, 08h
shr ax, cl
xlat
mov [si], al
dec si
mov ax, result
```

```
        and ax, 0f000h
        mov cl, 0ch
        shr ax, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg
        int 21h
        mov al, 00h
        mov ah, 04ch
        int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 05555h

n2 --- 05555h

Output:

The result is AAAA

Result:

Thus 16 bit hexadecimal addition has been performed successfully and the output is displayed.

b. 16 bit Hexadecimal Subtraction

Aim:

To write an assembly language program to implement 16 bit hexadecimal subtraction.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Perform the operation using sub instruction.
(d) Convert the result to human readable format using xlat instruction.
(e) Display the result.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     dw  05555h

n2     dw  05555h

result dw  00000h

msg    db  'The result is '

asciir db  4 dup(?)

       db  cr, lf, '$'

data ends


code segment

       assume cs:code, ds:data

start:

       mov ax, data

       mov ds, ax
```

```
mov ax, n1
sub ax, n2
lea bx, table
mov result, ax
lea si, asciir
add si, 3
mov ax, result
and ax, 0000fh
xlat
mov [si], al
dec si
mov ax, result
and ax, 000f0h
mov cl, 04h
shr al, cl
xlat
mov [si], al
dec si
mov ax, result
and ax, 00f00h
mov cl, 08h
shr ax, cl
xlat
mov [si], al
dec si
mov ax, result
and ax, 0f000h
```

```
        mov cl, 0ch
        shr ax, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg
        int 21h
        mov al, 00h
        mov ah, 04ch
        int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 05555h

n2 --- 05555h

Output:

The result is 0000

Result:

Thus 16 bit hexadecimal subtraction has been performed successfully and the output is displayed.

c. 16 bit Hexadecimal Multiplication

Aim:

To write an assembly language program to implement 16 bit hexadecimal multiplication.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Perform the operation using mul instruction.
(d) Convert DX:AX to human readable format using xlat instruction.
(e) Display the result.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     dw  05555h

n2     dw  05555h

result dw  00000h

msg    db  'The result is '

asciir db  8 dup(?)

       db  cr, lf, '$'

data ends


code segment

       assume cs:code, ds:data

start:

       mov ax, data

       mov ds, ax
```

```
mov ax, n1
mul n2
lea bx, table
mov result, ax
lea si, asciir
add si, 7
mov ax, result
and ax, 0000fh
xlat
mov [si], al
dec si
mov ax, result
and ax, 000f0h
mov cl, 04h
shr al, cl
xlat
mov [si], al
dec si
mov ax, result
and ax, 00f00h
mov cl, 08h
shr ax, cl
xlat
mov [si], al
dec si
mov ax, result
and ax, 0f000h
```

```
mov cl, 0ch
shr ax, cl
xlat
mov [si], al
dec si
mov result, dx
mov ax, result
and ax, 0000fh
xlat
mov [si], al
dec si
mov ax, result
and ax, 000f0h
mov cl, 04h
shr al, cl
xlat
mov [si], al
dec si
mov ax, result
and ax, 00f00h
mov cl, 08h
shr ax, cl
xlat
mov [si], al
dec si
mov ax, result
and ax, 0f000h
```

```
    mov cl, 0ch
    shr ax, cl
    xlat
    mov [si], al
    mov ah, 09h
    lea dx, msg
    int 21h
    mov al, 00h
    mov ah, 04ch
    int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 05555h

n2 --- 05555h

Output:

The result is 1C718E39

Result:

Thus 16 bit hexadecimal multiplication has been performed successfully and the output is displayed.

d. 16 bit Hexadecimal Division

Aim:

To write an assembly language program to implement 16 bit hexadecimal division.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Load dividend to AX.
(d) Load DX with 0.
(e) Perform the operation using div instruction.
(f) Convert DX and AX to human readable format using xlat instruction.
(g) Display DX as remainder and AX as quotient.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table   db  '0123456789ABCDEF'

lsbdiv  dw  05555h

msbdiv  dw  00000h

divisor dw  05000h

quo     dw  0000h

rem     dw  0000h

msg     db  'The quotient is '

asquo   db  4 dup(?)

        db  cr, lf, '$'

msg1    db  'The remainder is '

asrem   db  4 dup(?)

        db  cr, lf, '$'

data ends
```

```
code segment
        assume cs:code, ds:data
start:
        mov ax, data
        mov ds, ax
        mov ax, lsbdiv
        mov dx, msbdiv
        div divisor
        mov quo, ax
        mov rem, dx
        lea bx, table
        lea si, asquo
        add si, 3
        mov ax, quo
        and ax, 0000fh
        xlat
        mov [si], al
        dec si
        mov ax, quo
        and ax, 000f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
        dec si
        mov ax, quo
        and ax, 00f00h
```

```
mov cl, 08h
shr ax, cl
xlat
mov [si], al
dec si
mov ax, quo
and ax, 0f000h
mov cl, 0ch
shr ax, cl
xlat
mov [si], al
mov ah, 09h
lea dx, msg
int 21h
mov ax, rem
lea si, asrem
add si, 3
and ax, 0000fh
xlat
mov [si], al
dec si
mov ax, rem
and ax, 000f0h
mov cl, 04h
shr al, cl
xlat
mov [si], al
```

```
        dec si
        mov ax, rem
        and ax, 00f00h
        mov cl, 08h
        shr ax, cl
        xlat
        mov [si], al
        dec si
        mov ax, rem
        and ax, 0f000h
        mov cl, 0ch
        shr ax, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg1
        int 21h
quit:   mov al, 00h
        mov ah, 04ch
        int 21h
code ends
end start
```

Sample input & output:

Input:

dividend --- 00005555

divisor --- 5000

Output:

The quotient is 0001

The remainder is 0555

Result:

Thus 16 bit hexadecimal division has been performed successfully and the output is displayed.

8 Bit Decimal Operations using MASM

Ex No. : 6 Date: 09/03/2021

a. 8 bit decimal Addition

Aim:

To write an assembly language program to implement 8 bit decimal addition.

Algorithm:

(a) Define all the necessary constants and the values to be added.
(b) Load the first value into AL.
(c) Add the two 8-bit numbers using add instruction.
(d) Convert the answer in AX to decimal using DAA instruction.
(e) Convert the binary answer to human readable format using xlat.
(f) Display the sum.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     db  005h

n2     db  006h

result db  000h

msg    db  'The result is '

asciir db  2 dup(?)

       db  cr, lf, '$'

data ends


code segment
```

```
        assume cs:code, ds:data
start:
        mov ax, data
        mov ds, ax
        mov al, n1
        add al, n2
        daa
        lea bx, table
        mov result, al
        lea si, asciir
        add si, 1
        mov al, result
        and al, 00fh
        xlat
        mov [si], al
        dec si
        mov al, result
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
disp:   mov ah, 09h
        lea dx, msg
        int 21h
quit:   mov al, 00h
        mov ah, 04ch
```

```
    int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 005h

n2 --- 006h

Output:

The result is 11

Result:

Thus 8 bit decimal addition has been performed successfully and the output is displayed.

b. 8 bit decimal Subtraction

Aim:

To write an assembly language program to implement 8 bit decimal subtraction.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Load the first value into the AL register.
(d) Perform the operation using sub instruction.
(e) Convert the result to decimal using DAS instruction.
(f) Convert the result to human readable format using xlat instruction.
(g) Display the result.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     db  016h

n2     db  006h

result db  000h

msg    db  'The result is '

asciir db  2 dup(?)

       db  cr, lf, '$'

data ends


code segment

       assume cs:code, ds:data

start:

       mov ax, data
```

```
        mov ds, ax
        mov al, n1
        sub al, n2
        das
        lea bx, table
        mov result, al
        lea si, asciir
        add si, 1
        mov al, result
        and al, 00fh
        xlat
        mov [si], al
        dec si
        mov al, result
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
disp:   mov ah, 09h
        lea dx, msg
        int 21h
quit:   mov al, 00h
        mov ah, 04ch
        int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 016h

n2 --- 006h

Output:

The result is 10

Result:

Thus 8 bit decimal subtraction has been performed successfully and the output is displayed.

c. 8 bit decimal Multiplication

Aim:

To write an assembly language program to implement 8 bit decimal multiplication.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Load the first value into the al register.
(d) Perform the operation using mul instruction.
(e) Convert the result to ASCII format using AAM instruction.
(f) Convert AX to human readable format using xlat instruction.
(g) Display the result.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

   table db '0123456789'

   number1 db 009h

   number2 db 009h

   message1 db ' the result is'

   asciiresult db 2 dup(?)

   db cr, lf, '$'

data ends


code segment

    assume cs:code, ds: data

start: mov ax, data

       mov ds, ax

       mov al, number1

       mul number2
```

```
        aam
        lea bx,table
        lea si, asciiresult
        inc si
        and al,00fh
        xlat
        mov [si], al
        dec si
        mov al, ah
        and al,00fh
        xlat
        mov [si], al
        mov ah,09h
        lea dx, message1
        int 21h
 quit: mov al,0
       mov ah,04ch
       int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 005h

n2 --- 006h

Output:

The result is 30

Result:

Thus 8 bit decimal multiplication has been performed successfully and the output is displayed.

d. 8 bit decimal Division

Aim:

To write an assembly language program to implement 8 bit decimal division.

Algorithm:

(a) Define the necessary constants.
(b) Define the values of the inputs.
(c) Load dividend to AL.
(d) Load AH with 0.
(e) Convert the input to appropriate format using AAD instruction.
(f) Perform the operation using div instruction.
(g) Convert AH and AL to human readable format using xlat instruction.
(h) Display AH as remainder and AL as quotient.

Program:

```
cr equ 0dh

lf equ 0ah


data segment

table  db  '0123456789ABCDEF'

n1     dw  02Dh

n2     db  006h

rem    db  000h

quo    db  000h

msg1   db  'The quotient is '

asquo  db  2 dup(?)

       db  cr, lf, '$'

msg2   db  'The remainder is '

asrem  db  2 dup(?)

       db  cr, lf, '$'


data ends
```

```
code segment
        assume cs:code, ds:data
start:
        mov ax, data
        mov ds, ax
        mov ax, n1
        aad
        div n2
        lea bx, table
        mov quo, al
        mov rem, ah
        lea si, asquo
        add si, 1
        mov al, quo
        and al, 0fh
        xlat
        mov [si], al
        dec si
        mov al, quo
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg1
```

```
        int 21h

        lea si, asrem
        add si, 1
        mov al, rem
        and al, 0fh
        xlat
        mov [si], al
        dec si
        mov al, rem
        and al, 0f0h
        mov cl, 04h
        shr al, cl
        xlat
        mov [si], al
        mov ah, 09h
        lea dx, msg2
        int 21h
quit:   mov al, 00h
        mov ah, 04ch
        int 21h
code ends
end start
```

Sample input & output:

Input:

n1 --- 02Dh

n2 --- 006h

Output:

The quotient is 07

The remainder is 03

Result:

Thus 8 bit decimal division has been performed successfully and the output is displayed.

Block Transfer of Data Using MASM

Ex No. : 7 Date: 16/03/2021

Aim:

To perform block transfer of data using MASM.

Algorithm:

(a) Initialize the Data Segment
(b) Move data to DS
(c) Load the SI with the effective starting address of the Block of Data
(d) Load the DI with the effective starting address of the Result
(e) Initialize CX with 5
(f) Repeatedly move the block of data as byte by byte

Program:

```
assume ds:data, cs:code, es:extra
data segment
blk     db      10h,20h,30h,40h,50h
data    ends
extra segment
result  db      5 dup(?)
extra   ends
code    segment
start:
        mov     bx,data
        mov     ds,bx
        mov     bx,extra
        mov     es,bx
        lea     si, blk
        lea     di,result
        mov     cx,0005h
        rep  movsb
        mov     al,00h
        mov     ah,4ch
        int     21h
code    ends
end     start
```

Sample input & output:

```
D:\BIN>BLOCK_TR.EXE

D:\BIN>DEBUG BLOCK_TR.EXE
-u
076C:0000 BB6A07        MOV     BX,076A
076C:0003 8EDB          MOV     DS,BX
076C:0005 BB6B07        MOV     BX,076B
076C:0008 8EC3          MOV     ES,BX
076C:000A 8D360000      LEA     SI,[0000]
076C:000E 8D3E0000      LEA     DI,[0000]
076C:0012 B90500        MOV     CX,0005
076C:0015 F3            REPZ
076C:0016 A4            MOVSB
076C:0017 B000          MOV     AL,00
076C:0019 B44C          MOV     AH,4C
076C:001B CD21          INT     21
076C:001D 0000          ADD     [BX+SI],AL
076C:001F 0000          ADD     [BX+SI],AL
```

```
-g = CS:0000 001B

AX=4C00  BX=076B  CX=0000  DX=0000  SP=0000  BP=0000  SI=0005  DI=0005
DS=076A  ES=076B  SS=0769  CS=076C  IP=001B   NV UP EI PL NZ NA PO NC
076C:001B CD21          INT     21
-d DS:0000 0004
076A:0000  10 20 30 40 50                                       . 0@P
-d ES:0000 0004
076B:0000  10 20 30 40 50                                       . 0@P
```

Result:

The Program was executed successfully in debugging mode and the result was verified.

Sorting numbers using MASM

Ex No. : 8 Date: 23/03/2021

Aim:

To perform sorting of numbers using MASM.

Algorithm:

(a) Initialize the Data Segment
(b) Move data to DS
(c) Load CH and CL with the (total count of numbers -1)
(d) Load the SI with the effective starting address of the string of numbers
(e) Compare every two numbers from the first till the second last number
(f) Swap the two numbers if they are not in order
(g) Repeatedly perform steps v and vi till the numbers are sorted

Program:

```
data    segment
string1 db      40h,10h,50h,20h,99h,12h,56h,45h,36h
data    ends
code    segment
assume  cs:code, ds:data
start:  mov     ax,data
        mov     ds,ax
        mov     ch,08h
up2:    mov     cl,08h
        lea     si,string1
up1:    mov     al,[si]
        mov     bl,[si+1]
        cmp     al,bl
        jc      down
        mov     dl, [si+1]
        xchg    [si],dl
        mov     [si+1],dl
down:   inc     si
        dec     cl
        jnz     up1
        dec     ch
```

```
        jnz     up2
quit:   mov     al,0h
        mov     ah,04ch
        int     21h
code    ends
end     start
```

Sample input & output:

```
D:\BIN>DEBUG SORT_ASC.EXE
-G

Program terminated normally
-R
AX=FFFF  BX=0000  CX=003D  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0000   NV UP EI PL NZ NA PO NC
076B:0000 B86A07        MOV     AX,076A
-D 0769:0000
0769:0000  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 07   ................
0769:0010  10 12 20 36 40 45 50 56-99 00 00 00 00 00 00 00   .. 6@EPV........
0769:0020  B8 6A 07 8E D8 B5 08 B1-08 8D 36 00 00 8A 04 8A   .j........6.....
0769:0030  5C 01 3A C3 72 08 8A 54-01 86 14 88 54 01 46 FE   \.:.r..T....T.F.
0769:0040  C9 75 EA FE CD 75 E0 B0-00 B4 4C CD 21 EB 68 C7   .u...u....L.!.h.
0769:0050  06 FA 14 00 00 C7 06 FC-14 00 00 80 FC 3F 74 05   .............?t.
0769:0060  80 FC 40 75 08 8B 0E F2-14 89 0E FA 14 E8 76 00   ..@u..........v.
0769:0070  1E 07 EA A0 0A 58 00 E8-22 04 A1 F4 14 E8 3A 01   .....X.."......:.
-Q
```

Result:

The Program was executed successfully in debugging mode and the result was verified.

Stepper Motor Interface

Ex No. : 9 Date: 30/03/2021

Aim:

To interface stepper motor with 8086 microprocessor using 8255 interface

Algorithm:

(a) Initialise the control register by loading a value
(b) Load the address of Port A into DX.
(c) Load the value 88H into AL.
(d) Out the value and mark it as Label L1
(e) Call the label 'Delay'
(f) Rotate the value right by 1-bit.
(g) Jump to label L1
(h) Load 4000 into CX and mark it as label 'DELAY'
(i) Loop the label 'Return'

Program:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	DX, FFE6H	BA,E6,FF	CWR port address
8003		MOV	AL, 80H	B0, 80	
8005		OUT	DX , AL	EE	
8006		MOV	DX, FFE0H	BA , E0, FF	
8009		MOV	AL, 88H	B0, 88	Load Step sequence
800B	L1:	OUT	DX , AL	EE	
800C		CALL	DELAY	E8, 04,00	
800F		ROR	AL,01H	D0, C0	Clock wise/ Anti clockwise Rotation
8011		JMP	L1	EB, F8	
8013	DELAY:	MOV	CX, 4000H	B9,00,40	
8016	REPEAT:	LOOP	REPEAT	E2, FE	
8018		RET		C3	

Result:

The stepper motor is interfaced successfully with the 8086 microprocessor using 8255 interface.

Generation Of Square, Sawtooth And Triangular Waveform

Ex No. : 10 Date: 06/04/2021

a. Squarewave Generator

Aim:

To generate squarewave waveform using 8086 microprocessor with 8255 interface

Algorithm:

(a) Initialise the interface 8255 by loading control word register to DX register
(b) Load the address of Port A into DX register and mark it as label 'L1'
(c) Load the value '00' into AL
(d) OUT the value of DX and AL
(e) Call the label 'DELAY'
(f) Move the value 'FF' into AL register
(g) OUT the value of DX and AL
(h) Call the label 'DELAY' again
(i) Jump to label L1
(j) Load the value '0FF0' into BX and label it as 'DELAY'
(k) Decrement the value of BX and mark it as L2
(l) If BX not equal to zero , jump to label L2

Program:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	DX, FFE6H	BA, E6, FF	
8003		MOV	AL , 80	B0, 80	
8005		OUT	DX , AL	EE	
8006	L1:	MOV	DX, FFE0H	BA,E0,FF	
8009		MOV	AL, 00H	B0, 00	
800B		OUT	DX, AL	EE	
800C		CALL	DELAY	E8, 08, 00	
800F		MOV	AL, FFH	B0,FF	
8011		OUT	DX , AL	EE	
8012		CALL	DELAY	E8, 02,00	
8015		JMP	L1	EB,F2	
8017	DELAY:	MOV	BX,0FF0H	BB, F0, 0F	
801A	L2:	DEC	BX	4B	
801B		JNZ	L2	75,FD	
801D		RET		C3	

Result:

The square wave is generated by 8086 microprocessor with the help of 8255 Interface.

b. Sawtooth Wave Generator

Aim:

To generate sawtooth waveform using 8086 microprocessor with 8255 interface and DAC

Algorithm:

(a) Initialise the interface 8255 by loading control word register to DX register
(b) Load the value 80 into AL and OUT the value of DX and AL
(c) Move the value '00' into AL and label it as 'L1'
(d) Move the address of Port A into DX and label it as 'INCR'
(e) OUT the value of DX,AL
(f) Increment the value of AL
(g) Compare the value of AL with 'FF'
(h) Jump to 'INCR' if CF=1
(i) Jump to label 'L1'

Program:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	DX, FFE6H	BA, E6, FF	
8003		MOV	AL , 80	B0, 80	
8005		OUT	DX , AL	EE	
8006	L1:	MOV	AL, 00H	B0, 00	
8008	INCR:	MOV	DX, FFE0H	BA, E0, FF	
800B		OUT	DX , AL	EE	
800C		INC	AL	FE, C0	
800E		CMP	AL, FFH	3C, FF	
8010		JB	INCR	72, F6	Jump if below: CF=1
8012		JMP	L1	EB, F2	

Result:

The saw tooth wave is generated using 8086 microprocessor,8255 interface and DAC

c. Triangular Waveform

Aim:

To generate triangular wave form using 8086 microprocessor with 8255 interface

Algorithm:

(a) Initialise the interface 8255 by loading control word register to DX register
(b) Load the value 80 into AL and OUT the value of DX and AL
(c) Move the value '00' into AL and label it as L1
(d) Move the address of Port A to DX
(e) Out the value of DX,AL and label it as L2
(f) Increment the value of AL
(g) Compare AL with FF
(h) If CF=1 ,Jump to label L2
(i) Out the value of DX and AL and label it as L3
(j) Decrement the value of AL and compare it with '00'
(k) Jump to L3 on NO ZERO
(l) Jump to label L1

Program:

Memory	Label	Mnemonics	Operand	Opcode	Comments
8000		MOV	DX, FFE6H	BA, E6, FF	
8003		MOV	AL , 80	B0, 80	
8005		OUT	DX , AL	EE	
8006	L1:	MOV	AL, 00H	B0, 00	
8008		MOV	DX, FFE0H	BA, E0, FF	
800B	L2:	OUT	DX , AL	EE	
800C		INC	AL	FE, C0	
800E		CMP	AL, FFH	3C, FF	
8010		JB	L2	72, F6	Jump if below: CF=1
8012	L3:	OUT	DX , AL	EE	
8013		DEC	AL	CE,C8	
8015		CMP	AL,00H	3C, 00	
8017		JNZ	L3	75,F9	
8019		JMP	L1	EB, EB	

Result:

The triangular wave is generated using 8086 microprocessor and 8255 interface

8 Bit Arithmetic Operations using 8051

Ex No. : 11 Date: 13/04/2021

a. 8-Bit Hexadecimal Addition

Aim:

To perform 8-bit hexadecimal addition using 8051.

Algorithm:

(a) Move first operand into R1.
(b) Move second operand into A.
(c) Add A and R1.
(d) Move A into memory.
(e) Set A to #000H
(f) Add with carry A and #000H
(g) Move A into memory

Program:

```
MOV R1,#0FFH
MOV A,#0FFH
ADD A,R1
MOV R3,A
MOV A,#000H
```

```
ADDC A,#000H
MOV R4,A
```

Sample input & output:

Input:
R1 --- 0FFH
A --- 0FFH

Output:
R3 --- 0FEH
R4 --- 001H

Result:

Thus 8 bit hexadecimal addition has been performed successfully and the output is displayed.

b. 8-Bit Hexadecimal Subtraction

Aim:

To perform 8-bit hexadecimal subtraction using 8051.

Algorithm:

(a) Move first operand into R1.
(b) Move second operand into A.
(c) Subtract with borrow A and R1.
(d) Move A into memory.
(e) Set A to #000H
(f) Add with carry A and #000H
(g) Move A into memory

Program:

```
MOV R1,#0FFH
MOV A,#0FEH
SUBB A,R1
MOV R3,A
MOV A,#000H
ADDC A,#000H
MOV R4,A
```

Sample input & output:
Input :
R1 --- 0FFH
A --- 0FEH

Output:
R3 --- 0FEH
R4 --- 001H

Result:

Thus 8 bit hexadecimal subtraction has been performed successfully and the output is displayed.

c. 8-Bit Hexadecimal Multiplication

Aim:

To perform 8-bit hexadecimal multiplication using 8051.

Algorithm:

(a) Move first operand into R1.
(b) Move second operand into R2.
(c) Move R1 into A.
(d) Move R2 into B.
(e) Multiply A and B.
(f) Move B into memory.
(g) Move A into memory

Program:

```
MOV R1,#0FFH
MOV R2,#0FFH
MOV A,R1
MOV B,R2
MUL AB
MOV R3,B
MOV R4,A
```

Sample input & output:

Input :
R1 --- 0FFH
A --- 0FFH

Output:
R3 --- 0FEH
R4 --- 001H

Result:

Thus 8 bit hexadecimal multiplication has been performed successfully and the output is displayed.

d. 8-Bit Hexadecimal Division

Aim:

To perform 8-bit hexadecimal division using 8051.

Algorithm:

(a)Move first operand into R1.
(b)Move second operand into R2.
(c)Move R1 into A.
(d)Move R2 into B.
(e)Divide A and B.
(f)Move A into memory.
(g)Move B into memory

Program:

```
MOV R1,#0FEH
MOV R2,#00EH
MOV A,R1
MOV B,R2
DIV AB
MOV R3,A
MOV R4,B
```

Sample input & output:

Input:
R1 --- 0FEH
A --- 00EH

Output:
R3 --- 012H
R4 --- 002H

Result:

Thus 8 bit hexadecimal division has been performed successfully and the output is displayed.

16 Bit Arithmetic Operations using 8051

Ex No. : 12 Date: 20/04/2021

a. 16-Bit Hexadecimal Addition

Aim:

To perform 16-bit hexadecimal addition using 8051.

Algorithm:

(a) Move first operand into R1 and R2.
(b)Move second operand into R3 and R4.
(c)Move R1 into A.
(d)Add A and R3.
(e)Move A into memory.
(f) Move R3 into A.
(g)Add with carry A and R4.
(h)Move A into memory.
(i) Set A to #000H
(j) Add with carry A and #000H
(k)Move A into memory

Program:

```
MOV R1,#0F0H
MOV R2,#0F0H
MOV R3,#0F0H
MOV R4,#0F0H
MOV A,R1
ADD A,R3
MOV 0010H,A
MOV A,R2
ADDC A,R4
MOV 0011H,A
MOV A,#000H
ADDC A,#000H
MOV 0012H,A
```

Sample input & output:

Input :
R1 --- 0F0H
R2 --- 0F0H
R3 --- 0F0H

R4 --- 0F0H

Output:
[0012H] --- 001H
[0011H] --- 0E1H
[0010H] --- 0E0H

Result:

Thus 16 bit hexadecimal addition has been performed successfully and the output is displayed.

b. 16-Bit Hexadecimal Subtraction

Aim:

To perform 16-bit hexadecimal subtraction using 8051.

Algorithm:

(a) Move first operand into R1 and R2.
(b)Move second operand into R3 and R4.
(c)Move R1 into A.
(d)Subtract A and R3.
(e)Move A into memory.
(f) Move R3 into A.
(g)Subtract A and R4.
(h)Move A into memory.
(i) Set A to #000H
(j) Add with carry A and #000H
(k)Move A into memory

Program:

```
MOV R1,#0FFH
MOV R2,#0FFH
MOV R3,#0F0H
MOV R4,#0F0H
MOV A,R1
SUBB A,R3
MOV 0010H,A
MOV A,R2
SUBB A,R4
MOV 0011H,A
MOV A,#000H
ADDC A,#000H
MOV 0012H,A
```

Sample input & output:
Input:
R1 --- 0FFH
R2 --- 0FFH
R3 --- 0F0H
R4 --- 0F0H

Output:
[0012H] --- 000H
[0011H] --- 00FH

[0010H] --- 00FH

Result:

Thus 16 bit hexadecimal subtraction has been performed successfully and the output is displayed

Sorting the number using 8051

Ex No. : 13 Date: 27/04/2021

Aim:

To perform sorting on numbers using 8051.

Algorithm:

(a) Move array of number into memory location [030H].
(b) Move size of array into R1.
(c) Move R1 into A and label this step as AGAIN.
(d) Move A into R2.
(e) Move #030H into R0.
(f) Move the array element present at address given in R0 into A.
(g) Increment R0 and label this step as UP.
(h) Move the array element present at address given in R0 into B.
(i) Clear carry flag.
(j) Subtract with borrow A and B.
(k) Jump to label SKIP if carry flag is set.
(l) Move the array element present at address given in R0 into B.
(m) Decrement R0.
(n) Move the array element present at address given in R0 into A.
(o) Move B into memory address given by R0.
(p) Increment R0.
(q) Move A into memory address given by R0.
(r) Decrement R2 and if resulting value is not zero, jump to label UP. Label this step as SKIP.
(s) Decrement R1 and if resulting value is not zero, jump to label AGAIN.

Program:

```
MOV 030H,#005H
MOV 031H,#006H
MOV 032H,#003H
MOV 033H,#004H
MOV 034H,#001H
MOV 035H,#002H
MOV R1,#06H

AGAIN :   MOV A,R1
          MOV R2,A
          MOV R0,#030H
          MOV A,@R0
```

```
UP :    INC R0
        MOV B,@R0
        CLR C
        SUBB A,B
        JC SKIP
        MOV B,@R0
        DEC R0
        MOV A,@R0
        MOV @R0,B
        INC R0
        MOV @R0,A

SKIP :  DJNZ R2,UP
        DJNZ R1,AGAIN
```

Sample input & output:

Input :

[030H] ~ [035H] --- 005H, 006H, 003H, 004H, 001H, 002H

R1 --- 006H

Output:

[030H] ~ [035H] --- 001H, 002H, 003H, 004H, 005H, 006H

Result:

The Program was executed successfully and the result was verified.