

Input

8100 : 02

8101 : 03

Output

8102 : 05

Aim: To write and execute a program for addition of two single byte numbers using 8051

Address	Instruction	opcode	Explanation
8000	MOV DPTR, #8100	90:81:00	DPTR points to memory location 8100
8003	MOVX A, @DPTR	F0	Content in DPTR is moved to accumulator
8004	MOV R0, A	F8	Content in A is moved to R0
8005	INC DPTR	A3	Increment DPTR
8006	MOVX A, @DPTR	E0	Content in DPTR is moved to accumulator
8007	ADD A, R0	28	Content of A and R0 and result is stored in A
8008	INC DPTR	A3	Increment DPTR
8009	MOVX @DPTR, A	F0	Value in A is moved to location pointed by DPTR
800A	MOV A, #00	44:00	Clear accumulator
800C	ADDC A, #0	34:00	Add accumulator with carry
800E	INC DPTR	A3	Increment DPTR
800F	MOVX @DPTR, A	F0	Value in A is moved to location pointed by DPTR
8010	SJMP 8010	80:FE	

Result

Program executed and output obtained successfully

## Input

8100:03

8101:05

## Output

8102:02

Aim: To write and execute a program for subtraction  
of two single byte numbers using 8051

Address	Instruction	Opcode	Explanation
8000	CLR C	C3	Clear carry
8001	Mov DPTR, #8100	90:81:00	DPTR points to memory location \$100
8004	MovX A, @DPTR	E0	Move content in Dptr to Accumulator
8005	Mov R0, A	F8	Move content in Accumulator to R0
8006	INC DPTR	A3	Increment DPTR
8007	MovX A, @DPTR	E0	Move content in DPTR to Accumulator
8008	SUBB A, R0	98	Subtract R0 from A and store in A
8009	INC DPTR	A3	Increment DPTR
800A	MovX @DPTR, A	F0	Move content in A to memory location pointed by DPTR
800B	SJMP 800B	80:FF	

## Result

Program was executed and output obtained successfully.

### Input

### EXPERIMENT-5

#### MULTIPLICATION OF TWO 8-BIT NUMBERS

\$100:04

\$101:02

Ans: To write and execute a program for multiplication  
of two 8-bit numbers using 8051

### Output

Address	Instruction	Opcode	Explanation
8000	MOV DPTR, #8100	90:81:00	DPTR points to memory location 8100
8003	MOVX A, @DPTR	E3	Move content of DPTR to Accumulator
8004	MOV F0,A	F5:F0	Move content of Accumulator to F0 (B)
8006	INC DPTR	A3	Increment DPTR
8007	MOVX A,@DPTR	E0	Move content of DPTR to Accumulator
8008	MUL AB	A4	Multiply contents of A and B and Result is stored in A
8009	INC DPTR	A3	Increment DPTR
800A	MOVX @DPTR,A	F0	Move content of A to memory location pointed by DPTR
800B	MOV A,F0	E5:F0	Move content of F0 (B) to Accumulator
800E	MOVX @DPTR,A	F0	Move content of A to memory location pointed by DPTR
800F	STCPL SOOF	80:fE	

### Result

Program was executed and output was obtained successfully.

### Input

8100: 03  
8101: 04

### Output

8102: 02  
8103: 01

Aim: To write and execute program for division of two 8-bit numbers using 8051

### EXPERIMENT- 5D

DIVISION OF TWO 8-BIT NUMBERS.

Address	Instruction	Opcode	Explanation
8000	MOV DPTR, #8100	90:81:00	DPTR points to memory location 8100
8003	Movx A, @DPTR	F3	Move content of DPTR to Accumulator
8004	MOV F0, A	F5:F0	Move content of Accumulator to F0
8006	INC DPTR	A3	Increment DPTR
8007	Movx A, @DPTR	E0	Move contents of DPTR to Accumulator
8008	DIV AB	84	Divide content of A by content of B
			and store quotient in A and remainder in B.
8009	INC DPTR	A3	Increment DPTR
800A	Movx @DPTR, A	F0	Move content of A to memory location pointed by DPTR
800B	MOV A, F0	E5:F0	Move content of F0 to A
800C	INC DPTR	A3	Increment DPTR
800E	Movx @DPTR, A	F0	Move content of A to memory location pointed by DPTR

Result  
Program was executed and output obtained successfully

19308  
SILVER BEECHES / 19308  
19308

104

0500:02

Output

0550:05

**EXPERIMENT - 6.4**

**ARITHMETIC OPERATIONS**

ADDITION OF TWO

卷之三

**Aim:** To write and execute program for addition of two 8-bit numbers using loops

Address	Instruction	Opcode	Explanation
0400	MOV DI, 0550	BF:5D:05	Move 0550 to DI register
0403	MOV SI, 0500	BE:00:05	Move 0500 to SI register
0406	MOV AL, [SI]	8A:04	Move content in SI location to AL register
0408	INC SI	46	Increment SI register
0409	Mov BL, [SI]	8A:1C	Move content in SI location to BL register
040B	ADD AL, BL	00:D8	Add AL and BL registers
040D	MOV [DI], AL	88:05	Move content of AL to location pointed by DI

Instruction	Opcode	Explanation
NOV DT, 0550	BF:5D:D5	Move 0550 to R
NOV ST, 0500	BE:00:05	Move 0500 to S
NOV AL, [ST]	8A:04	Move content in S to AL register
NC RT	46	Increment 81 register
NOV BL, [ST]	8A:1C	Move content in S to BL register

D40B	ADD AL,BL	00:08	Add AL and BL registers
D40D	MOV [DI],AL	88:05	Move content of AL to location pointed by DI
D40F	HLT	F4	

1

0500:10  
0501: 03  
0502: 20  
0503: 10

*Output*

0550:30  
0551:13

EXPERIMENT - 6.0

卷之三

**Aim:** To write and execute a program for addition of two 16-bit numbers using 5051

Address	Instruction	Opcode	Explanation
0400	MOV DI, 0550	BF:50:05	Move 0550 to DI register
0403	MOV ST, 0500	BE:00:05	Move 0500 to ST register
0406	MOV AX, [SI]	8B:04	Move the memory content of SI to AX register
0408	MOV BX, [SI+02]	8B:5C:02	Move the memory content of [SI+02] to BX register

040B	ADD AX,BX	D1:DS	Add AX and BX registers and store the result in AX
040D	MOV [017],AX	89:05	Move content of AX to memory location pointed by DX

CHOLESTEROL

## Result

Program was executed and output obtained successfully

卷之三

### Input

0500 : 05  
0501 : FA  
0502 : 2A  
0503 : E5

### Output

0550 : DF  
0551 : 5F  
0552 : 01

EXPERIMENT - 6C  
ADDITION OF TWO 16-BIT NUMBERS CONSIDERING CARRY  
Aim: To write and execute a program for addition of two 16-bit numbers considering carry

Address	Instruction	Opcode	Explanation
0400	CLC	F8	Clear carry flag
0401	Mov DI, 0550	BF:5D:05	Move 0550 to DI register
0404	Mov SI, 0500	BE:00:05	Move 0500 to SI Register
0407	Mov AX,[SI]	8B:04	Move The memory content of SI to AX register
0409	Mov BX,[SI+02]	8B:BC:02	Move the memory content of [SI+02] to BX register
040C	ADD AX,BX	01:D8	Add AX and BX register and store result in AX register
040E	Mov [DI],AX	89:05	Move content of AX register to memory location in DI register
0410	Mov AX,0000	BB:00:00	Clear AX register
0413	ADC AX,0000	15:00:00	Add AX with carry and store result in AX
0416	Mov [DI+02],AX	89:45:02	Move content of AX register to memory location in [DI+02]
0419	HLT	F4	

Result  
Program was compiled and output was obtained successfully

## Input

0500:66  
0501:66  
0502:11  
0503:11

## Output

0550:55  
0551:55

Aim: To write and execute a program for subtraction of two 16-bit numbers using戈86

Address	Instruction	Opcode	Explanation
0400	Mov D1,0550	BF3D:05	Move 0550 to DI register
0403	Mov ST,0500	BE:00:05	Move 0500 to ST register
0406	Mov AX,[ST]	8B:04	Move memory content of ST to AX register
0408	Mov BX,[ST+0]	8B:5C:02	Move memory content of [ST+0] to BX Register

Store result in AX register

040D	SUB AX,BX	89:D8	Subtract BX from AX and store result in AX register
040F	Mov [D1],AX	89:05	Move content of AX register to memory location in DI

to memory location in DI

040F	HLT	F4	
------	-----	----	--

Result			
--------	--	--	--

Program was executed and output was obtained successfully

## EXPERIMENT - 6D

### SUBTRACTION OF TWO 16-BIT NUMBERS

## EXPERIMENT - 6E

SUBTRACTION OF TWO 16-BIT NUMBERS  
(WITH CONSIDERING BORROW)

**Input**

```
0500:18
0501:08
0502:40
0503:10
```

Two 16-bit numbers  
0500:18 and 0501:08 are to be subtracted  
0502:40 is borrow value  
0503:10 is result

**Aim:** To write and execute a program for subtraction of  
two 16-bit numbers with borrow

Address	Instruction	Opcate	Explanation
0400	CLC	F8	Clear carry flag
0401	MOV DI, 0550	BF:50:05	Move 0550 to DI register
0404	MOV SI, 0500	BE:00:05	Move 0500 to SI register
0407	MOV AX, [SI]	8B:04	Move memory content of SI register to AX register
0409	MOV BX,[SI+02]	8B:5C:02	Move memory content of [SI+02] to AX register
040C	SBB AX,BX	19:D8	Subtract BX from AX considering the borrow and store result in AX register
040E	MOV [D7],AX	89:05	Move content of AX register to memory location in DI register
0410	HLT	F4	

Two 16-bit numbers, 0500:18 and 0501:08 are loaded in DI and SI registers  
0502:40 is borrow value

### Result

Program was executed and output obtained successfully

### Input

0500:1A  
0501:2B  
0502:4B  
0503:12

### Output

0550:9E  
0551:44  
0552:14  
0553:03

Aim: To write and execute a program for multiplication of two 16-bit numbers

Address	Instruction	Opcode	Explanation
0400	MOV DL,0550	BF:5D:05	Move 0550 to DL register
0403	MOV SI,0500	BE:00:D5	Move 0500 to SI register
0406	MOV AX,[SI]	8B:04	Move memory content of SI to AX register
0408	MOV CX,[SI+02]	8B:4C:02	Move memory content of SI+02 to CX register
040B	MUL CX	F1:E1	Multiply contents of AX and CX register and store result
040D	MOV [DI],AX	89:45:02	Move content of AX register to bit in AX and highest index
040F	MOV [DI+02],DX	89:45:02	Move content of DX register to memory location in DI
0412	HLT	F4	memory location in DI+02

### Result

Program was executed and output was obtained.

## EXPERIMENT - 5

DIVISION OF 16-BIT NUMBER BY 8-BIT NUMBER

NAME

Input

~~0B500:43~~  
~~0D50:12~~  
~~0050:321~~

~~0B500:43~~  
~~0D50:12~~  
~~0050:321~~

Output

~~0550:8D~~  
~~0551:16~~  
~~0400:00~~  
~~0403:0000~~  
~~0406:0000~~

Ans: To write and execute a program for division of a 16-bit number by an 8-bit number using R寄存器.

Address	Instruction	Opcode	Explanation
0400	MOV DL, 0550	BF:50:05	Move 0550 to DL register
0403	MOV SI, 0500	BE:00:05	Move 0500 to SI register
0406	MOV AX,[SI]	8B:04	Move memory content of SI into AX register

0408	MOV CL,[8H]	8A:4C:02	Move memory content of 8H to CL register
040B	DIV CL	F6:F1	Divide AX by CL and store quotient in AH and remainder in AL

0400	MOV [DI],AX	89:05	Move content of AX register to memory location in DI
040F	HLT	F4	

Program was executed and output obtained successfully

Result

Program was executed and output obtained successfully

## Input

0500:05  
0501:01  
0502:02  
0503:03  
0504:04  
0505:05

## Output

0601:01  
0602:04  
0603:09  
0604:10  
0605:19

SQUARES OF 8-BIT NUMBERS IN A BLOCK

Aim: To write and execute a program to find squares of 8-bit numbers stored in a block.

Address	Instruction	Opcode	Explanation
0400	Mov SI,0500	BE:00:05	Move 0500 to SI register
0403	Mov DI,0601	BF:01:06	Move 0601 to DI register
0406	Mov CL,[SI]	84:0C	Move memory content of SI register to CL
0408	INC SI	46	Increment SI
0409	Mov AL,[SI]	8A:04	Move memory content of SI to AL register
040B	MUL AL	F6:00	Multiply contents of AL register with SI and store the result in AL register
040D	Mov [DI],AL	88:05	Move content of AL to memory location in DI register
040F	INC DI	47	Increment DI register
0410	DEC CL	FE:09	Decrement CL register (count)
0412	JNZ 0408	75:54	If zero flag is not set then jump to location 0408
0414	HLT	F4	

Program was executed and output was obtained.

## Result

Program was executed and output was obtained.

114

0500:03

## Output

**Aim:** To write and execute a program to find sum of

Address	Instruction	Opcode	Explanation
0400	MOV ST,ES00	BED005	Move DS00 to ST registers
0403	Mov DI,0601	BF:01:06	Move 0601 to DI register
0406	Mov CX,[ESI]	8A:0C	Move memory content of ST to CX register
0408	Mov AL,01	BD:01	Move 01 to AL register
040A	Mul CL	FE:E1	Multiply CX register with AL register and store result in AL register
040C	DEC CL	FE:C9	Decrement CL register
040E	JNZ 040A	75:5A	If zero flag is not set then jump to location 040A
0410	Mov [DI],AL	88:05	Move content of AL register to memory location in DI
0412	HLT	F4	

## EXPERIMENT-9

### Input

0500:05

0501:01

0502:02

0503:03

0504:04

0505:05

### Output

0601:09

### SUM OF ODD NUMBERS FROM A SERIES OF NUMBERS

**Aim:** Write and execute a program to find sum of odd numbers from a series of numbers

Address	Instruction	Opcode	Explanation
0400	MOV BL,00	B3:00	Move 00 to BL register
0402	MOV SI,0500	BED005	Move 0500 to SI register
0405	MOV DI,0601	BF10106	Move 6601 to DI register
0408	MOV CL,[SI]	8A:0C	Move memory content of SI to CL register
040A	INC ST	46	Increment ST register
040B	MOV AL,[SI]	8A:04	Move memory content of SI to AL register
040D	TEST AL,01	A8:01	Perform bitwise AND operation between AL register and 01
040E	JZ 0415	74:04	and set the Flag accordingly
040F	JZ 0415	74:04	If the zero flag is set then jump to location 0415
0411	ADD AL,BL	00:08	Add contents of AL and BL registers and store result in AL
0413	Mov BL,01	88:C3	Move content of AL to BL
0415	DEC CL	F5:C9	Decrement CL register
0417	JNZ 040A	75:F1	If the zero flag is not set then jump to 040A
0419	Mov [DI],AL	88:05	Move content of AL to memory location in DI
041B	HLT	F4	

### Input

5555

0500:12  
0501:34

### Output

0500:43  
0501:21

Aim : To write and execute a program to reverse a 16-bit

number using 8086

Address	Instruction	Opcode	Explanation
0400	MOV SI,0500	BE:00:05	Move 0500 to SI register
0403	MOV AL,[SI]	8A:04	Move memory content at SI to AL register
0405	MOV CL,04	B1:04	Move 04 to CL register
0407	ROR AL,CL	D2:C8	Rotate the bits in AL register to right by value stored in CL times
0409	INC SI	A6	Increment SI register
040A	XCHG AL,[SI]	96:01: D2:C8	Exchange content of AL register with content in memory location ST
040C	RRR AL,CL	D2:C8	Rotate the bits in AL register to right side by value stored in CL number of times.
040E	DEC SI	4E	Decrement SI register
040F	Mov [SI],AL	88:04	Move contents of AL register to memory location in SI
0411	HLT	F4	

### Result

Program was executed and output obtained successfully

**Input 1**

0500 : 22

**Output 1**

0601 : 01

**Input 2**

0500 : 54

**Output 2**

0601 : 00

**Aim:** To write and execute a program to check whether an 8-bit number is palindrome or not

Address	Instruction	Opcode	Explanation
0400	MOV 8T,0500	BE:00:05	Move 0500 to 8T register
0403	MOV DI,0601	BE:01:06	Move 0601 to DI register
0406	MOV CL,B4	B1:04	Move 04 to CL register
0408	MOV AL,[5T]	8A:04	Move memory content of ST to AL register
040A	MOV BL,[8T]	8A:1C	Move memory content of ST to BL register
040C	ROR AL,CL	D4:CE	Rotate the bits in AL register to right side by the value in CL number of times
040E	CMP AL,BL	88:D8	Compare content of AL and BL registers and set flags accordingly
0410	JZ 0417	74:05	If zero flag is set then jump to memory location 0417
0412	Mov [B1],00	C6:05:00	Clear content of B1 register
0415	JMP 041A	FB:03	Jump to location 041A
0417	Mov [DI],01	C6:05:01	Move 01 to DI register
041A	HLT	F4	Halt

**Result**

Program was executed and output was obtained successfully

### Input

0500:02  
0501:05  
0502:03

**Output**  
 0500:02  
 0501:05  
 0502:03  
 Ans : To write and execute a program to find out  
 the discriminant of a quadratic equation.  
 Express the nature of roots (00: real and unequal,  
 01: real and distinct or: imaginary)

Address	Instruction	Opcode	Explanation
0400	MOV ST, 0500	BE:00:05	Move 0500 to ST register
0403	MOV DI, 0600	BF:00:06	Move 0600 to DI register
0406	MOV BL,[ST]	8A:1C	Move memory content of ST to BL register
0408	INC ST	46	Increment ST register
0409	MOV AL,[ST]	8A:04	Move memory content of ST to AL register
040B	MUL AL	F6:F0	Multiply AL register with itself and store result in AL
040D	MOV CL, AL	88:C1	Move content of AL to CL
040F	MOV AL,04	B0:D4	Move 04 to AL register
0411	MUL BL	F6:E3	Multiply content of BL register with AL and store result in AL
0413	INC SI	46	Increment SI register
0414	-MOV BL,[SI]	8A:1C	Move memory content of SI register with BL
0416	MUL BL	F6:E3	Multiply content of BL register with AL and store result in AL
0418	CMPL AL	38:C1	Compare contents of AL and BL registers and modify flags

### EXPERIMENT - 12

#### DISCRIMINANT OF A QUADRATIC EQUATION

041A	JNZ 0421	15:05	If zero flag is not set then
041C	MOV [DI],00	C6:05:00	jump to location 0421
041F	JMP 042B	- EB:0A	Move 00 to DI location
0421	JC 0428	42:05	Jump to location 042B
0423	MOV [B1],01	4E:05:01	If carry flag is set then
0426	JMP 042B	EB:03	jump to location 042B
0428	MOV [DI],02	C5:05:02	Move 02 to DI memory location
042B	HLT	F4	
<b>Result</b>			
Program was executed and output obtained successfully			

### Input

0501:05  
0501:06  
0502:04  
0503:02  
0504:09  
0505:01

### Output

0501:01  
0502:02  
0503:03  
0504:04  
0505:05

Aim: While and execute a program to sort an array of numbers in ascending order

Address	Instruction	Opcode	Description
0400	MOV ST,0500	BE:00:05	Move 0500 to ST register
0403	MOV CL,[ST]	8A:0C	Move memory content of ST register to CL
0405	DEC CL	FE:09	Decrement CL register
0407	MOV ST,0500	BE:00:05	Move 0500 to ST register
040A	MOV CH,[ST]	8A:2C	Move memory content of ST to CH register
040C	DEC CH	FE:CD	Decrement CH register
040E	INC ST	46	Increment ST register
040F	MOV AL,[ST]	8A:04	Move memory content of ST to AL register
0411	INC SI	3A:04	Increment SI register
0412	CMP AL,[SI]	3A:04	Compare AL and content of SI and set flags accordingly
0414	JC 041C	7A:06	Jump if carry flag is set then 041C
0416	XCHG AL,[SI]	86:04	Exchange content of AL and [SI]
0418	DEC ST	4E	Decrement ST register
0419	XCHG AL,[SI]	86:04	Exchange content of AL and [SI]
041B	INC ST	46	Increment ST register
041C	DEC CH	FE:CD	Decrement CH register
041E	JNZ 040F	75:EF	If zero flag is not set then jump

EXPERIMENT - 13

BORTE AN ARRIVED IN ASCENDING ORDER

0420	DEC CL	to location 040F
0422	JNZ 0407	FECQ Decrement CL register
0424	HLT	#4 If zero flag is not set then jump to location 0407

### Input

CONVERT

EXPERIMENT-111

FROM : 16 bit

TO : 8 bit

PROGRAM

0500: FF  
0501: 00

0400: 55  
0401: 02

0402: 00  
0403: 00

0404: 00  
0405: 00

0406: 00  
0407: 00

0408: 00  
0409: 00

040A: 00  
040B: 00

040C: 00  
040D: 00

040E: 00  
040F: 00

0410: 00  
0411: 00

0412: 00  
0413: 00

0414: 00  
0415: 00

0416: 00  
0417: 00

0418: 00  
0419: 00

041A: 00  
041B: 00

041C: 00  
041D: 00

041E: 00  
041F: 00

0420: 00  
0421: 00

0422: 00  
0423: 00

0424: 00  
0425: 00

0426: 00  
0427: 00

0428: 00  
0429: 00

042A: 00  
042B: 00

042C: 00  
042D: 00

042E: 00  
042F: 00

0430: 00  
0431: 00

0432: 00  
0433: 00

0434: 00  
0435: 00

0436: 00  
0437: 00

0438: 00  
0439: 00

043A: 00  
043B: 00

043C: 00  
043D: 00

043E: 00  
043F: 00

0440: 00  
0441: 00

0442: 00  
0443: 00

0444: 00  
0445: 00

0446: 00  
0447: 00

0448: 00  
0449: 00

044A: 00  
044B: 00

044C: 00  
044D: 00

044E: 00  
044F: 00

0450: 00  
0451: 00

0452: 00  
0453: 00

0454: 00  
0455: 00

0456: 00  
0457: 00

0458: 00  
0459: 00

045A: 00  
045B: 00

045C: 00  
045D: 00

045E: 00  
045F: 00

0460: 00  
0461: 00

0462: 00  
0463: 00

0464: 00  
0465: 00

0466: 00  
0467: 00

0468: 00  
0469: 00

046A: 00  
046B: 00

046C: 00  
046D: 00

046E: 00  
046F: 00

0470: 00  
0471: 00

0472: 00  
0473: 00

0474: 00  
0475: 00

0476: 00  
0477: 00

0478: 00  
0479: 00

047A: 00  
047B: 00

047C: 00  
047D: 00

047E: 00  
047F: 00

0480: 00  
0481: 00

0482: 00  
0483: 00

0484: 00  
0485: 00

0486: 00  
0487: 00

0488: 00  
0489: 00

048A: 00  
048B: 00

048C: 00  
048D: 00

048E: 00  
048F: 00

0490: 00  
0491: 00

0492: 00  
0493: 00

0494: 00  
0495: 00

0496: 00  
0497: 00

0498: 00  
0499: 00

049A: 00  
049B: 00

049C: 00  
049D: 00

049E: 00  
049F: 00

04A0: 00  
04A1: 00

04A2: 00  
04A3: 00

04A4: 00  
04A5: 00

04A6: 00  
04A7: 00

04A8: 00  
04A9: 00

04AA: 00  
04AB: 00

04AC: 00  
04AD: 00

04AE: 00  
04AF: 00

04B0: 00  
04B1: 00

04B2: 00  
04B3: 00

04B4: 00  
04B5: 00

04B6: 00  
04B7: 00

04B8: 00  
04B9: 00

04BA: 00  
04BB: 00

04BC: 00  
04BD: 00

04BE: 00  
04BF: 00

04C0: 00  
04C1: 00

04C2: 00  
04C3: 00

04C4: 00  
04C5: 00

04C6: 00  
04C7: 00

04C8: 00  
04C9: 00

04CA: 00  
04CB: 00

04CC: 00  
04CD: 00

04CE: 00  
04CF: 00

04D0: 00  
04D1: 00

04D2: 00  
04D3: 00

04D4: 00  
04D5: 00

04D6: 00  
04D7: 00

04D8: 00  
04D9: 00

04DA: 00  
04DB: 00

04DC: 00  
04DD: 00

04DE: 00  
04DF: 00

04E0: 00  
04E1: 00

04E2: 00  
04E3: 00

04E4: 00  
04E5: 00

04E6: 00  
04E7: 00

04E8: 00  
04E9: 00

04EA: 00  
04EB: 00

04EC: 00  
04ED: 00

04EE: 00  
04EF: 00

04F0: 00  
04F1: 00

04F2: 00  
04F3: 00

04F4: 00  
04F5: 00

04F6: 00  
04F7: 00

04F8: 00  
04F9: 00

04FA: 00  
04FB: 00

04FC: 00  
04FD: 00

04FE: 00  
04FF: 00

0400: 00  
0401: 00

0402: 00  
0403: 00

0404: 00  
0405: 00

0406: 00  
0407: 00

0408: 00  
0409: 00

040A: 00  
040B: 00

040C: 00  
040D: 00

040E: 00  
040F: 00

0410: 00  
0411: 00

0412: 00  
0413: 00

0414: 00  
0415: 00

0416: 00  
0417: 00

0418: 00  
0419: 00

041A: 00  
041B: 00

041C: 00  
041D: 00

041E: 00  
041F: 00

0420: 00  
0421: 00

0422: 00  
0423: 00

0424: 00  
0425: 00

0426: 00  
0427: 00

0428: 00  
0429: 00

042A: 00  
042B: 00

042C: 00  
042D: 00

042E: 00  
042F: 00

0430: 00  
0431: 00

0432: 00  
0433: 00

0434: 00  
0435: 00

0436: 00  
0437: 00

0438: 00  
0439: 00

043A: 00  
043B: 00

043C: 00  
043D: 00

043E: 00  
043F: 00

0440: 00  
0441: 00

0442: 00  
0443: 00

0444: 00  
0445: 00

0446: 00  
0447: 00

0448: 00  
0449: 00

044A: 00  
044B: 00

044C: 00  
044D: 00

044E: 00  
044F: 00

0450: 00  
0451: 00

0452: 00  
0453: 00

0454: 00  
0455: 00

0456: 00  
0457: 00

0458: 00  
0459: 00

045A: 00  
045B: 00

045C: 00  
045D: 00

045E: 00  
045F: 00

0460: 00  
0461: 00

0462: 00  
0463: 00

0464: 00  
0465: 00

0466: 00  
0467: 00

0468: 00  
0469: 00

046A: 00  
046B: 00

046C: 00  
046D: 00

046E: 00  
046F: 00

0470: 00  
0471: 00

0472: 00  
0473: 00

0474: 00  
0475: 00

0476: 00  
0477: 00

0478: 00  
0479: 00

047A: 00  
047B: 00

047C: 00  
047D: 00

047E: 00  
047F: 00

0480: 00  
0481: 00

0482: 00  
0483: 00

0484: 00  
0485: 00

0486: 00  
0487: 00

0488: 00  
0489: 00

048A: 00  
048B: 00

048C: 00  
048D: 00

048E: 00  
048F: 00

0490: 00  
0491: 00

0492: 00  
0493: 00

0494: 00  
0495: 00

0496: 00  
0497: 00

0498: 00  
0499: 00

049A: 00  
049B: 00

049C: 00  
049D: 00

049E: 00  
049F: 00

04A0: 00  
04A1: 00

04A2: 00  
04A3: 00

04A4: 00  
04A5: 00

04A6: 00  
04A7: 00

04A8: 00  
04A9: 00

04AA: 00  
04AB: 00

04AC: 00  
04AD: 00

04AE: 00  
04AF: 00

04B0: 00  
04B1: 00

04B2: 00  
04B3: 00

04B4: 00  
04B5: 00

Index

forward

Reverse

**Aim:** To write and execute a program to implement Stepper motor in 8086 HD kit

Address	Instruction	Opcode	Explanation
0400	MOV AL,80	B0:80	Move 80 to AL
0402	DTR 66,AL	E6:66	Out AL to port 66
0404	Mov CL,04	B1:D4	Move 04 to CL register
0406	MOV BX,1500	B8:0005	Move 500 to BX register
0409	MOV AL,[BX]	8A:04	Move memory content of BX to AH register
040B	DTR 60,HAL	E6:60	Out 4L to port 60
040D	CALL 0415	F8:05:00	Call subroutine at 0415 for day
0410	INC BX	H3	Increment BX register
0411	LOOPNZ 0409	ED:F6	If zero flag is not set then loop to location 0409
0413	JMP 040A	FB:EF	Jump to location 040A
0415	PUSH CX	51	Push CX register to stack
0416	MOV CX,0FFFF	B9:FF:FF	Move 0FFFF to CX register
0419	Loopnz 0419	E0:FE	If zero flag is not set then loop to location 0419
041B	Pop CX	59	Pop stack top to CX register
041C	RET	C3	Return from Subroutine

## Result

Program was executed and output was obtained successfully

**EXPERIMENT - 16**  
**8255 INTERFACE PROGRAM**

Aim: To write and execute a program to interface 8255 into 8086

Address	Instruction	Opcode	Description
0400	MOV AL,80	B0:80	Move 80 to AL register
0402	OUT 26,AL	E6:26	Out AL to Port 26
0404	MOV AL,FF	B0:FF	Move FF to AL register
0406	NOT AL	F6:D0	Logical Not performed to AL
0408	OUT 20,AL	F6:20	Out AL to port 20
040A	OUT 22,AL	E6:22	Out AL to port 22
040C	OUT 24,AL	E6:24	Out AL to port 24
040E	MOV CX,FFFF	B9:FF:FF	Move FFFF to CX register
0411	LOOPNZ 0411	E0:FE	If zero flag is not set then loop to location 0411
0413	JMP 040B	EB:F1	Jump to location 040B

Result  
 Program was executed and output obtained successfully

Output of program  
 Address Value  
 0400 80  
 0402 00  
 0404 FF  
 0406 00  
 0408 00  
 040A 00  
 040C 00  
 040E FFFF  
 0411 00  
 0413 F1

Conclusion  
 The output of the program shows that the 8255 has been interfaced with 8086.