LAB ACTIVITY REPORT FILE

UCS617 : MICROPROCESSOR BASED SYSTEM DESIGN

8085 MICROPROCESSOR

SUBMITTED BY:

JATIN BAGGA - 102203713

KUSHAGR SHARMA - 102203714

SHAURYA DAMATHIA - 102203798

JIYA - 102203801

GROUP : 3CO18 (3C43)

SUBMITTED TO:

DR. ROHAN SHARMA

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY

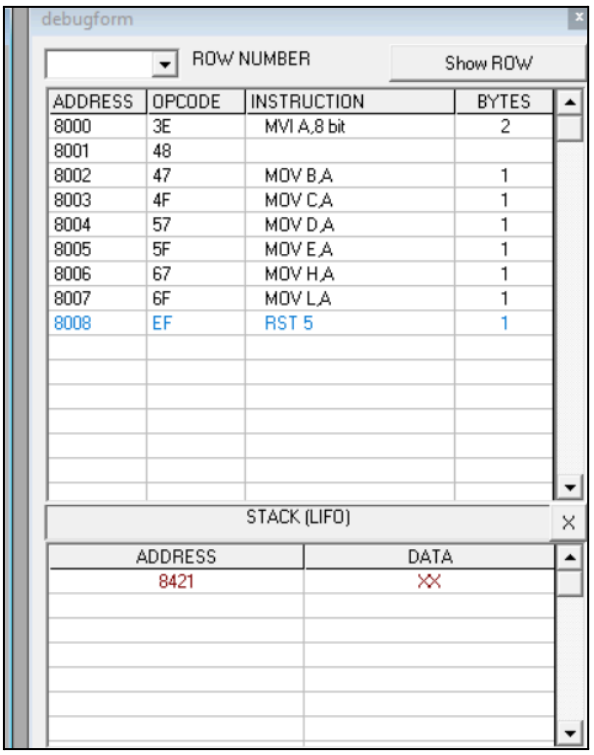
JAN-MAY 2025

TABLE OF CONTENTS

EXPERIMENT 1

1.1) Write a program to store 8-bit data into one register and then copy that to all registers

CODE:



debugform

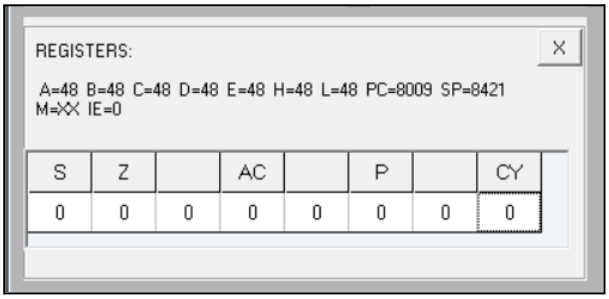
ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3E	MVI A,8 bit	2
8001	48		
8002	47	MOV B,A	1
8003	4F	MOV C,A	1
8004	57	MOV D,A	1
8005	5F	MOV E,A	1
8006	67	MOV H,A	1
8007	6F	MOV L,A	1
8008	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

OUTPUT:



REGISTERS:

A=48 B=48 C=48 D=48 E=48 H=48 L=48 PC=8009 SP=8421
M=XX IE=0

S	Z		AC		P		CY
0	0	0	0	0	0	0	0

1.2) Write a program for addition of two 8-bit numbers

CODE:

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3E	MVI A,8 bit	2
8001	19		
8002	06	MVI B,8 bit	2
8003	48		
8004	80	ADD B	1
8005	32	STA 16 bit	3
8006	00		
8007	90		
8008	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

OUTPUT:

REGISTERS:

A=61 B=48 C=00 D=59 E=90 H=85 L=03 PC=8009 SP=8421
M=00 IE=0

S	Z		AC		P		CY
0	0	0	1	0	0	0	0

9000 61

1.3) Write a program to add 8-bit numbers using direct and indirect addressing mode

CODE: DIRECT ADDRESSING

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	90		
8003	47	MOV B,A	1
8004	3A	LDA 16 bit	3
8005	01		
8006	90		
8007	80	ADD B	1
8008	32	STA 16 bit	3
8009	02		
800A	90		
800B	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT: [9000] - 05, [9001] - 09
OUTPUT:

REGISTERS:

A=0E B=05 C=00 D=00 E=00 H=00 L=00 PC=800C SP=8421
M=XX IE=0

S	Z		AC		P		CY
0	0	0	0	0	0	0	0

CODE: INDIRECT ADDRESSING

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	21	LXI H,16 bit	3
8001	00		
8002	90		
8003	7E	MOV A,M	1
8004	23	INX H	1
8005	86	ADD M	1
8006	23	INX H	1
8007	77	MOV M,A	1
8008	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT: [9000] - 05, [9001] - 09
OUTPUT:

REGISTERS:

A=0E B=05 C=00 D=00 E=00 H=00 L=00 PC=800C SP=8421
M=XX IE=0

S	Z		AC		P		CY
0	0	0	0	0	0	0	0

1.4) Write a program to add 16-bit numbers using direct and indirect addressing mode

CODE: DIRECT ADDRESSING MODE

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	2A	LHLD 16 bit	3
8001	00		
8002	90		
8003	EB	XCHG	1
8004	2A	LHLD 16 bit	3
8005	02		
8006	90		
8007	19	DAD D	1
8008	22	SHLD 16 bit	3
8009	04		
800A	90		
800B	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT: [9000] - 90, [9001] - 90, [9002] - 59, [9003] - 59
OUTPUT:

CODE: INDIRECT ADDRESSING MODE

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	01	LXI B,16 bit	3
8001	00		
8002	90		
8003	0A	LDAX B	1
8004	57	MOV D,A	1
8005	03	INX B	1
8006	0A	LDAX B	1
8007	82	ADD D	1
8008	32	STA 16 bit	3
8009	04		
800A	90		
800B	03	INX B	1
800C	0A	LDAX B	1
800D	57	MOV D,A	1
800E	03	INX B	1
800F	0A	LDAX B	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

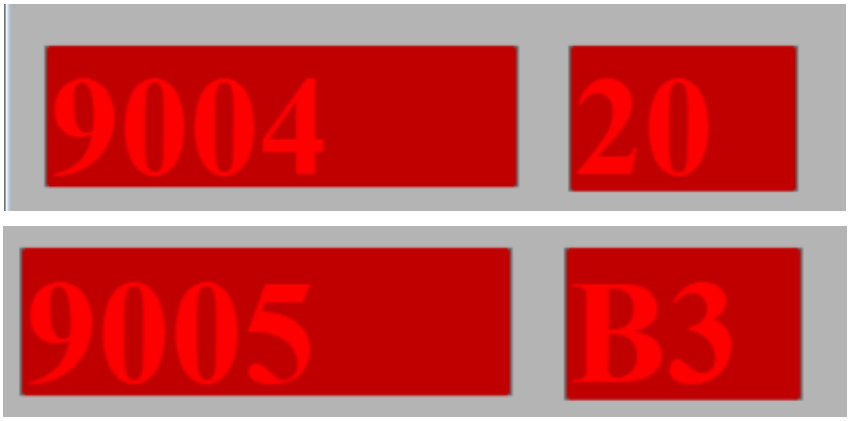
ADDRESS	OPCODE	INSTRUCTION	BYTES
800E	03	INX B	1
800F	0A	LDAX B	1
8010	8A	ADC D	1
8011	32	STA 16 bit	3
8012	05		
8013	90		
8014	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT: [9000] - 90, [9001] - 90, [9002] - 59, [9003] - 59

OUTPUT:



1.5) Write a program to add 8-bit numbers using carry. (using JNC instruction)

CODE:

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	0E	MVI C,8 bit	2
8001	00		
8002	21	LXI H,16 bit	3
8003	00		
8004	85		
8005	7E	MOV A,M	1
8006	23	INX H	1
8007	86	ADD M	1
8008	D2	JNC 16 bit	3
8009	0C		
800A	80		
		JUMPED TO 800C	
800C	23	INX H	1
800D	77	MOV M,A	1
800E	23	INX H	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8009	0C		
800A	80		
		JUMPED TO 800C	
800C	23	INX H	1
800D	77	MOV M,A	1
800E	23	INX H	1
800F	71	MOV M,C	1
8010	8A	ADC D	1
8011	32	STA 16 bit	3
8012	05		
8013	90		
8014	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT: [8500] - 58, [8501]-39

OUTPUT:

1.6) Write a program to find 1's complement and 2's complement of 8-bit number

CODE: 1's complement

```
Value at [9000h] = DBh
```

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	21	LXI H,16 bit	3
8001	00		
8002	90		
8003	7E	MOV A,M	1
8004	2F	CMA	
8005	23	INX H	1
8006	77	MOV M,A	1
8007	76	HLT	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

OUTPUT:

REGISTERS:

A=24 B=00 C=00 D=00 E=00 H=90 L=01 PC=8008 SP=8421
M=24 IE=0

S	Z		AC		P		CY
0	0	0	0	0	0	0	0

CODE: 2's complement

```
Value at [9000h] = DBh
```

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	21	LXI H,16 bit	3
8001	00		
8002	90		
8003	7E	MOV A,M	1
8004	2F	CMA	
8005	3C	INR A	1
8006	23	INX H	1
8007	77	MOV M,A	1
8008	76	HLT	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

OUTPUT:

REGISTERS:

A=25 B=00 C=00 D=00 E=00 H=90 L=01 PC=8009 SP=8421
M=25 IE=0

S	Z		AC		P		CY
0	0	0	0	0	0	0	0

EXPERIMENT 2

2) Write a program for the sum of series of numbers

CODE:

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	90		
8003	4F	MOV C,A	1
8004	97	SUB A	1
8005	21	LXI H,16 bit	3
8006	01		
8007	90		
8008	86	ADD M	1
8009	23	INX H	1
800A	0D	DCR C	1
800B	C2	JNZ 16 bit	3
800C	08		
800D	80		
		JUMPED TO 8008	

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8008	86	ADD M	1
8009	23	INX H	1
800A	0D	DCR C	1
800B	C2	JNZ 16 bit	3
800C	08		
800D	80		
		JUMPED TO 8008	
8008	86	ADD M	1
8009	23	INX H	1
800A	0D	DCR C	1
800B	C2	JNZ 16 bit	3
800C	08		
800D	80		
		JUMPED TO 8008	

8008	86	ADD M	1
8009	23	INX H	1
800A	0D	DCR C	1
800B	C2	JNZ 16 bit	3
800C	08		
800D	80		
		JUMPED TO 8008	
8008	86	ADD M	1
8009	23	INX H	1
800A	0D	DCR C	1
800B	C2	JNZ 16 bit	3
800C	08		
800D	80		
		JUMPED TO 8008	

8008	86	ADD M	1
8009	23	INX H	1
800A	0D	DCR C	1
800B	C2	JNZ 16 bit	3
800C	08		
800D	80		
800E	32	STA 16 bit	3
800F	00		
8010	91		
8011	EF	RST 5	1

INPUT: [9000] - 5, [9001] - 63, [9002] - 72, [9003] - 81, [9004] - 90, [9005] - 99
OUTPUT:

EXPERIMENT 3

3) Write a program for data transfer from memory block B1 to memory block B2

CODE:

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	0E	MVI C,8 bit	2
8001	05		
8002	21	LXI H,16 bit	3
8003	00		
8004	90		
8005	11	LXI D,16 bit	3
8006	00		
8007	91		
8008	7E	MOV A,M	1
8009	12	STAX D	1
800A	23	INX H	1
800B	13	INX D	1
800C	0D	DCR C	1
800D	C2	JNZ 16 bit	3
800E	08		

ADDRESS	OPCODE	INSTRUCTION	BYTES
800A	23	INX H	1
800B	13	INX D	1
800C	0D	DCR C	1
800D	C2	JNZ 16 bit	3
800E	08		
800F	80		
		JUMPED TO 8008	
8008	7E	MOV A,M	1
8009	12	STAX D	1
800A	23	INX H	1
800B	13	INX D	1
800C	0D	DCR C	1
800D	C2	JNZ 16 bit	3

800F	80		
		JUMPED TO 8008	
8008	7E	MOV A,M	1
8009	12	STAX D	1
800A	23	INX H	1
800B	13	INX D	1
800C	0D	DCR C	1
800D	C2	JNZ 16 bit	3
800E	08		
800F	80		
		JUMPED TO 8008	
8008	7E	MOV A,M	1
8009	12	STAX D	1

800E	08		
800F	80		
		JUMPED TO 8008	
8008	7E	MOV A,M	1
8009	12	STAX D	1
800A	23	INX H	1
800B	13	INX D	1
800C	0D	DCR C	1
800D	C2	JNZ 16 bit	3
800E	08		
800F	80		
8010	EF	RST 5	1

INPUT: [9000] - 5, [9001] - 63, [9002] - 72, [9003] - 81, [9004] - 90, [9005] - 99
OUTPUT:

EXPERIMENT 4

4) Write a program for multiply two 8-bit numbers

CODE:

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	85		
8003	5F	MOV E,A	1
8004	16	MVI D,8 bit	2
8005	00		
8006	3A	LDA 16 bit	3
8007	01		
8008	85		
8009	4F	MOV C,A	1
800A	21	LXI H,16 bit	3
800B	00		
800C	00		
800D	19	DAD D	1
800E	0D	DCR C	1
800F	C2	JNZ 16 bit	3

STACK (LIFO)

ADDRESS	DATA
8421	XX

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
800F	C2	JNZ 16 bit	3
8010	0D		
8011	80		
		JUMPED TO 800D	
800D	19	DAD D	1
800E	0D	DCR C	1
800F	C2	JNZ 16 bit	3
8010	0D		
8011	80		
8012	22	SHLD 16 bit	3
8013	00		
8014	86		
8015	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

OUTPUT:

8085

8500 B2

C D E F
8 9 A B
4 5 6 7
0 1 2 3

Reset | Kbint
Prev | exm Mem
Next | exm Reg
Go | Exec

REGISTERS:

A=03 B=00 C=00 D=00 E=B2 H=02 L=16 PC=8016 SP=8421
M=XX IE=0

S	Z		AC		P		CY
0	1	0	0	0	0	0	0

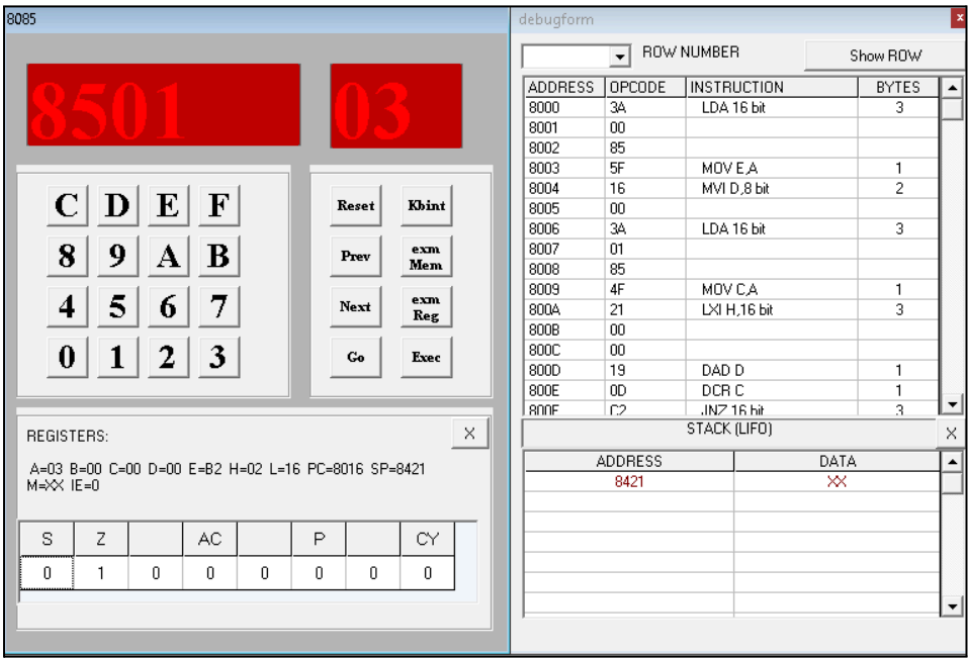
debugform

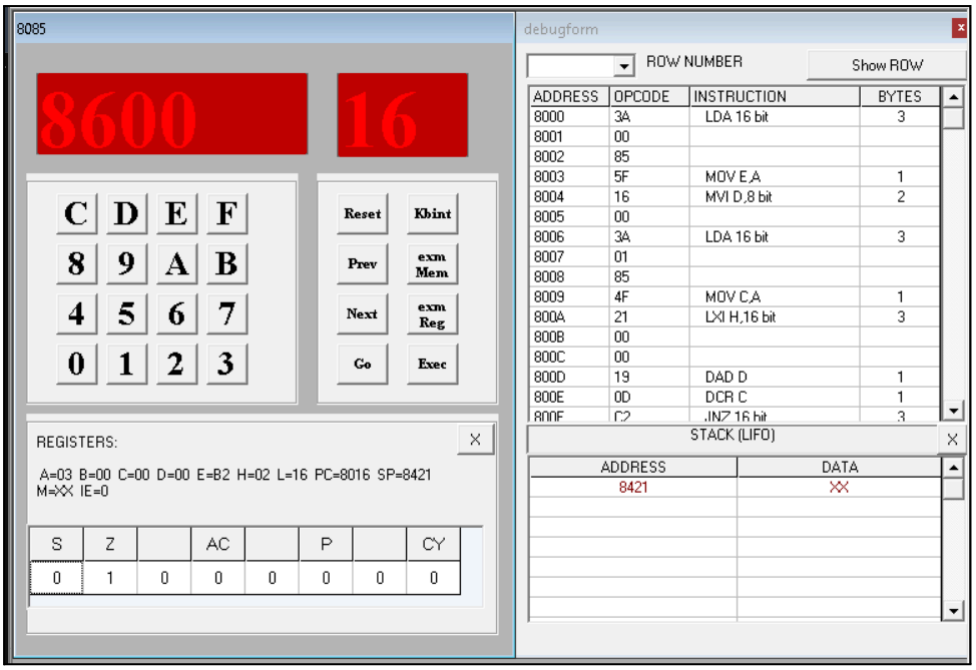
ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	85		
8003	5F	MOV E,A	1
8004	16	MVI D,8 bit	2
8005	00		
8006	3A	LDA 16 bit	3
8007	01		
8008	85		
8009	4F	MOV C,A	1
800A	21	LXI H,16 bit	3
800B	00		
800C	00		
800D	19	DAD D	1
800E	0D	DCR C	1
800F	C2	JNZ 16 bit	3

STACK (LIFO)

ADDRESS	DATA
8421	XX


8085
8501
03
C D E F
8 9 A B
4 5 6 7
0 1 2 3
Reset
Kbint
Prev
exm Mem
Next
exm Reg
Go
Exec
REGISTERS:
A=03 B=00 C=00 D=00 E=B2 H=02 L=16 PC=8016 SP=8421
M=XX IE=0
S Z AC P CY
0 1 0 0 0 0 0 0
debugform
ROW NUMBER
Show ROW
ADDRESS OPCODE INSTRUCTION BYTES
8000 3A LDA 16 bit 3
8001 00
8002 85
8003 5F MOV E,A 1
8004 16 MVI D,8 bit 2
8005 00
8006 3A LDA 16 bit 3
8007 01
8008 85
8009 4F MOV C,A 1
800A 21 LXI H,16 bit 3
800B 00
800C 00
800D 19 DAD D 1
800E 0D DCR C 1
800F C2 JNZ 16 bit 3
STACK (LIFO)
ADDRESS DATA
8421 XX


8085
8600
16
C D E F
8 9 A B
4 5 6 7
0 1 2 3
Reset
Kbint
Prev
exm Mem
Next
exm Reg
Go
Exec
REGISTERS:
A=03 B=00 C=00 D=00 E=B2 H=02 L=16 PC=8016 SP=8421
M=XX IE=0
S Z AC P CY
0 1 0 0 0 0 0 0
debugform
ROW NUMBER
Show ROW
ADDRESS OPCODE INSTRUCTION BYTES
8000 3A LDA 16 bit 3
8001 00
8002 85
8003 5F MOV E,A 1
8004 16 MVI D,8 bit 2
8005 00
8006 3A LDA 16 bit 3
8007 01
8008 85
8009 4F MOV C,A 1
800A 21 LXI H,16 bit 3
800B 00
800C 00
800D 19 DAD D 1
800E 0D DCR C 1
800F C2 JNZ 16 bit 3
STACK (LIFO)
ADDRESS DATA
8421 XX

EXPERIMENT 5

5) Write a program to add ten 8-bit numbers. Assume the numbers are stored in 8500-8509. Store the result in 850A and 850B memory address

CODE:

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	0E	MVI C,8 bit	2
8001	00		
8002	06	MVI B,8 bit	2
8003	09		
8004	21	LXI H,16 bit	3
8005	00		
8006	85		
8007	7E	MOV A,M	1
8008	23	INX H	1
8009	86	ADD M	1
800A	D2	JNC 16 bit	3
800B	0E		
800C	80		
800D	0C	INR C	1
800E	05	DCR B	1
800F	C2	JNZ 16 bit	3

STACK (LIFO)

ADDRESS	DATA
8421	XX

debugform

ROW NUMBER | Show ROW

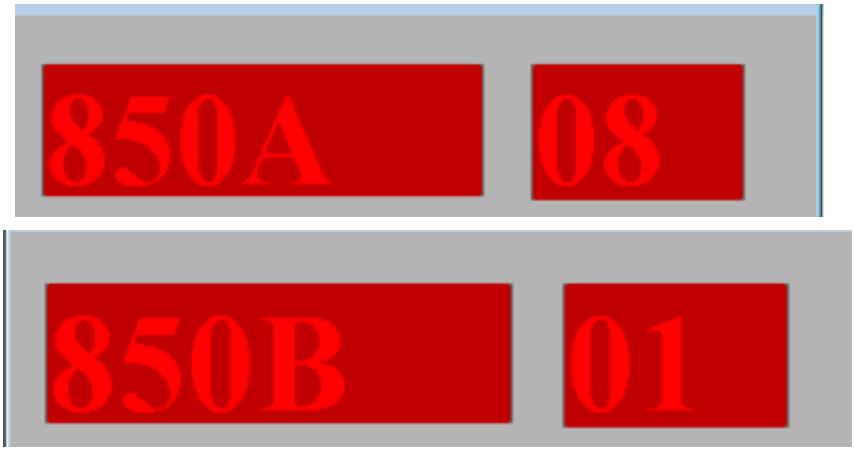
ADDRESS	OPCODE	INSTRUCTION	BYTES
		JUMPED TO 800E	
800E	05	DCR B	1
800F	C2	JNZ 16 bit	3
8010	08		
8011	80		
8012	23	INX H	1
8013	77	MOV M,A	1
8014	23	INX H	1
8015	71	MOV M,C	1
8016	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT- [8500] – FF, [8501] – 01, [8502] – 01, [8503] – 01, [8504] – 01, [8505] – 01, [8506] – 01, [8507] – 01, [8508] – 01, [8509] – 01

OUTPUT:



EXPERIMENT 6

6) Write a program to find the negative numbers in a block of data

CODE:

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	85		
8003	4F	MOV C,A	1
8004	06	MVI B,8 bit	2
8005	00		
8006	21	LXI H,16 bit	3
8007	01		
8008	85		
8009	7E	MOV A,M	1
800A	E6	ANI 8 bit	2
800B	80		
800C	CA	JZ 16 bit	3
800D	10		
800E	80		
		JUMPED TO 8010	

STACK (LIFO)

ADDRESS	DATA
8421	XX

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
800A	E6	ANI 8 bit	2
800B	80		
800C	CA	JZ 16 bit	3
800D	10		
800E	80		
800F	04	INR B	1
8010	23	INX H	1
8011	0D	DCR C	1
8012	C2	JNZ 16 bit	3
8013	09		
8014	80		
8015	78	MOV A,B	1
8016	32	STA 16 bit	3
8017	00		
8018	86		
8019	FF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT - [8500] – 04, [8501] – 56, [8502] – A9, [8503] – 73, [8504] – 82

OUTPUT

8600 02

REGISTERS:

A=02 B=02 C=00 D=00 E=00 H=85 L=05 PC=801A SP=8421
M=00 IE=0

EXPERIMENT 7

7) Write a program to count the number of one's in a number

CODE:

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	85		
8003	06	MVI B,8 bit	2
8004	08		
8005	16	MVI D,8 bit	2
8006	00		
8007	07	RLC	1
8008	D2	JNC 16 bit	3
8009	0C		
800A	80		
		JUMPED TO 800C	
800C	05	DCR B	1
800D	C2	JNZ 16 bit	3
800E	07		

STACK (LIFO)

ADDRESS	DATA
8421	XX

debugform

ROW NUMBER | Show ROW

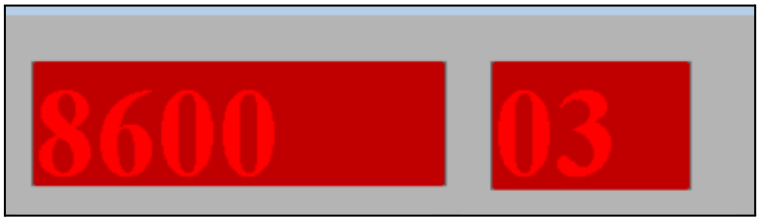
ADDRESS	OPCODE	INSTRUCTION	BYTES
8007	07	RLC	1
8008	D2	JNC 16 bit	3
8009	0C		
800A	80		
800B	14	INR D	1
800C	05	DCR B	1
800D	C2	JNZ 16 bit	3
800E	07		
800F	80		
8010	7A	MOV A,D	1
8011	32	STA 16 bit	3
8012	00		
8013	86		
8014	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT - [8500] - 25

OUTPUT:



REGISTERS:

A=03 B=00 C=00 D=03 E=00 H=00 L=00 PC=8015 SP=8421
M=XX IE=0

EXPERIMENT 8

8) Write a program to arrange numbers in Ascending order

CODE:

debugform

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	21	LXI H,16 bit	3
8001	00		
8002	85		
8003	4E	MOV C,M	1
8004	0D	DCR C	1
8005	51	MOV D,C	1
8006	21	LXI H,16 bit	3
8007	01		
8008	85		
8009	7E	MOV A,M	1
800A	23	INX H	1
800B	BD	CMP L	1
800C	DA	JC 16 bit	3
800D	14		
800E	80		
800F	46	MOV B,M	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

debugform

ROW NUMBER | Show ROW

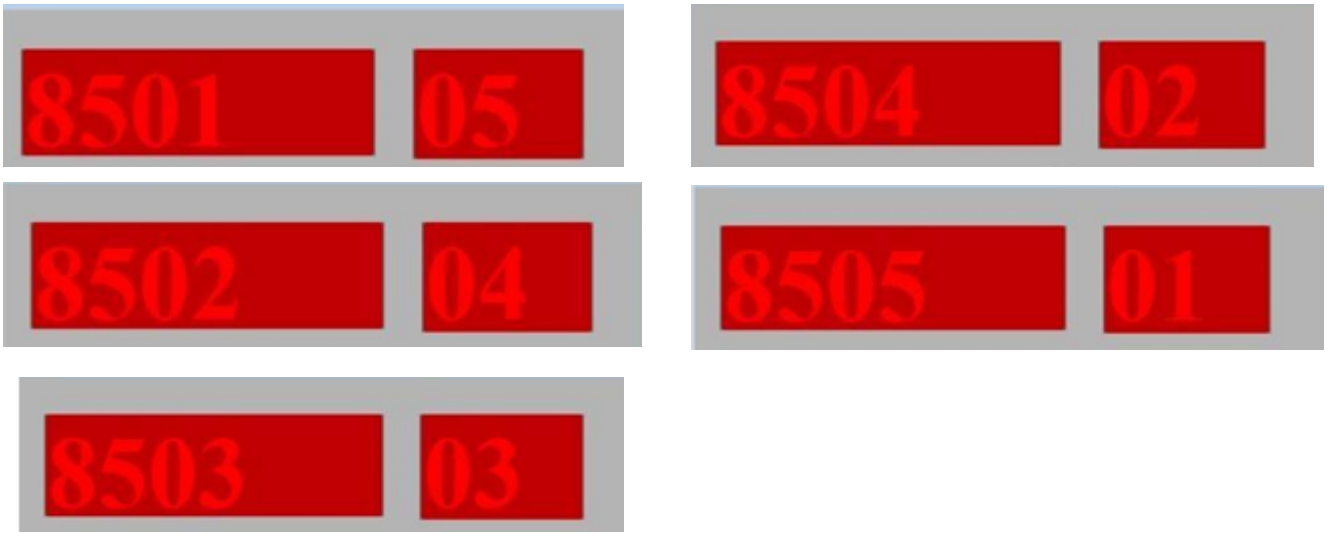
ADDRESS	OPCODE	INSTRUCTION	BYTES
		JUMPED TO 8013	
8013	15	DCR D	1
8014	C2	JNZ 16 bit	3
8015	09		
8016	80		
8017	0D	DCR C	1
8018	C2	JNZ 16 bit	3
8019	05		
801A	80		
801B	EF	RST 5	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT - [8500] – 05, [8501] – 05, [8502] – 04, [8503] – 03, [8504] – 02, [8505] – 01

OUTPUT:



EXPERIMENT 9

9) Calculate the sum of series of even numbers

CODE:

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	90		
8003	4F	MOV C,A	1
8004	06	MVI B,8 bit	2
8005	00		
8006	21	LXI H,16 bit	3
8007	01		
8008	90		
8009	7E	MOV A,M	1
800A	E6	ANI 8 bit	2
800B	01		
800C	C2	JNZ 16 bit	3
800D	12		
800E	80		
800F	78	MOV A,B	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8009	7E	MOV A,M	1
800A	E6	ANI 8 bit	2
800B	01		
800C	C2	JNZ 16 bit	3
800D	12		
800E	80		
800F	78	MOV A,B	1
8010	86	ADD M	1
8011	47	MOV B,A	1
8012	23	INX H	1
8013	0D	DCR C	1
8014	C2	JNZ 16 bit	3
8015	09		
8016	80		
		JUMPED TO 8009	

STACK (LIFO)

ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
800C	C2	JNZ 16 bit	3
800D	12		
800E	80		
		JUMPED TO 8012	
8012	23	INX H	1
8013	0D	DCR C	1
8014	C2	JNZ 16 bit	3
8015	09		
8016	80		
8017	32	STA 16 bit	3
8018	00		
8019	86		
801A	76	HLT	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT:

```
[9000h] - 05h
[9001h] - 14h
[9002h] - 24h
[9003h] - 33h
[9004h] - 45h
[9005h] - 23h
```

OUTPUT:

Sum stored in B register - 38h

REGISTERS:

A=01 B=38 C=00 D=00 E=00 H=90 L=06 PC=801B SP=8421
M=00 IE=0

S	Z		AC		P		CY
0	1	0	1	0	0	0	0

EXPERIMENT 10

10) Write an assembly language program to verify how many bytes are present in a given set, which resembles 10101101 in 8085

CODE: $(10101101)_2 = (AD)_{16}$

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	06	MVI B,8 bit	2
8001	0A		
8002	16	MVI D,8 bit	2
8003	AD		
8004	0E	MVI C,8 bit	2
8005	00		
8006	21	LXI H,16 bit	3
8007	00		
8008	90		
8009	7E	MOV A,M	1
800A	BA	CMP D	1
800B	C2	JNZ 16 bit	3
800C	0F		
800D	80		
800E	0C	INR C	1
800F	23	INX H	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
800B	C2	JNZ 16 bit	3
800C	0F		
800D	80		
800E	0C	INR C	1
800F	23	INX H	1
8010	05	DCR B	1
8011	C2	JNZ 16 bit	3
8012	09		
8013	80		
8014	79	MOV A,C	1
8015	32	STA 16 bit	3
8016	0A		
8017	90		
8018	76	HLT	1

STACK (LIFO)

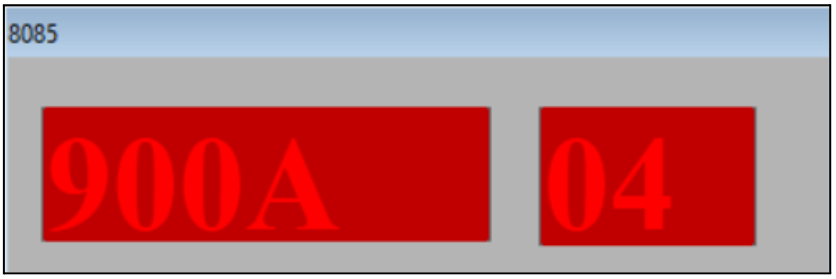
ADDRESS	DATA
8421	XX

INPUT:

```
[9000h] - ADh
[9001h] - 13h
[9002h] - 13h
[9003h] - ADh
[9004h] - 13h
[9005h] - 13h
[9006h] - 13h
[9007h] - 13h
[9008h] - ADh
[9009h] - ADh
```

OUTPUT:

```
[900Ah] - 04h
```



EXPERIMENT 11

11) Write an assembly language program to find the numbers of even parity in ten consecutive memory locations in 8085

CODE:

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	06	MVI B,8 bit	2
8001	0A		
8002	0E	MVI C,8 bit	2
8003	00		
8004	21	LXI H,16 bit	3
8005	00		
8006	90		
8007	7E	MOV A,M	1
8008	E6	ANI 8 bit	2
8009	FF		
800A	E2	JPO 16 bit	3
800B	0E		
800C	80		
800D	0C	INR C	1
800E	23	INX H	1
800F	05	DCR B	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8009	FF		
800A	E2	JPO 16 bit	3
800B	0E		
800C	80		
800D	0C	INR C	1
800E	23	INX H	1
800F	05	DCR B	1
8010	C2	JNZ 16 bit	3
8011	07		
8012	80		
8013	79	MOV A,C	1
8014	32	STA 16 bit	3
8015	0A		
8016	90		
8017	76	HLT	1

STACK (LIFO)

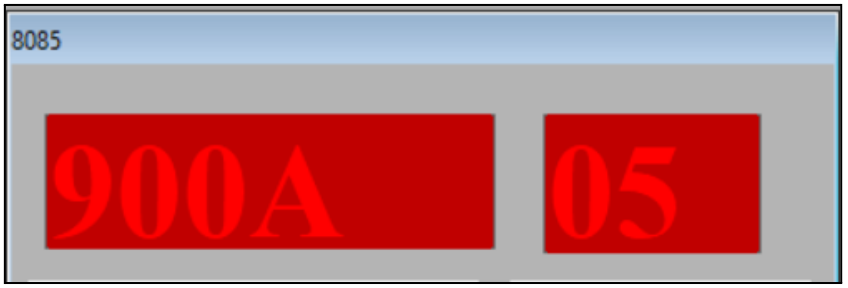
ADDRESS	DATA
8421	XX

INPUT:

```
[9000h] - ADh
[9001h] - 13h
[9002h] - 13h
[9003h] - ADh
[9004h] - 13h
[9005h] - 13h
[9006h] - 13h
[9007h] - 13h
[9008h] - ADh
[9009h] - ADh
```

OUTPUT:

```
[900Ah] - 05h
```



EXPERIMENT 12

12) Write an assembly language program to convert a BCD number into its equivalent binary in 8085

CODE:

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	90		
8003	47	MOV B,A	1
8004	E6	ANI 8 bit	2
8005	0F		
8006	4F	MOV C,A	1
8007	78	MOV A,B	1
8008	E6	ANI 8 bit	2
8009	F0		
800A	0F	RRC	1
800B	0F	RRC	1
800C	0F	RRC	1
800D	0F	RRC	1
800E	47	MOV B,A	1
800F	AF	XRA A	1

STACK (LIFO)

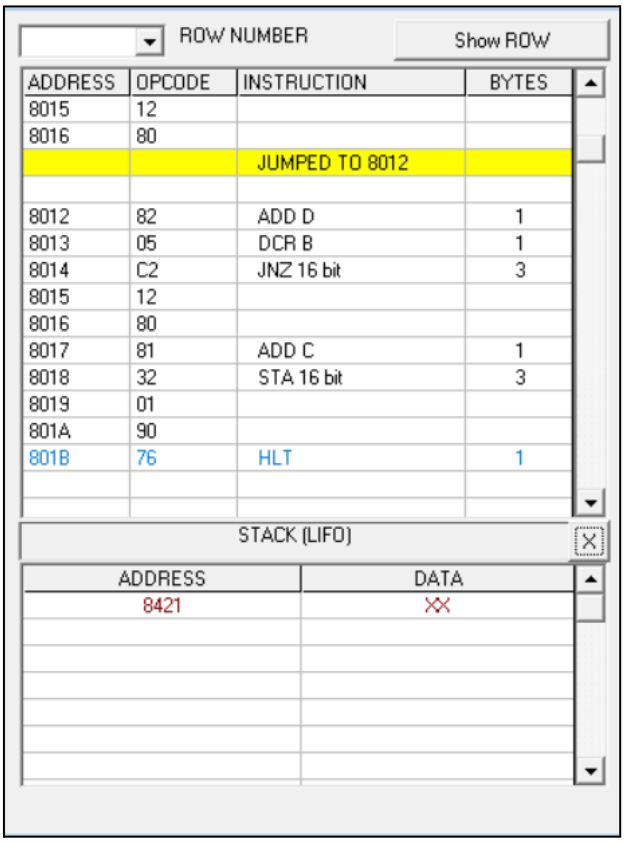
ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8009	F0		
800A	0F	RRC	1
800B	0F	RRC	1
800C	0F	RRC	1
800D	0F	RRC	1
800E	47	MOV B,A	1
800F	AF	XRA A	1
8010	16	MVI D,8 bit	2
8011	0A		
8012	82	ADD D	1
8013	05	DCR B	1
8014	C2	JNZ 16 bit	3
8015	12		
8016	80		
		JUMPED TO 8012	

STACK (LIFO)

ADDRESS	DATA
8421	XX

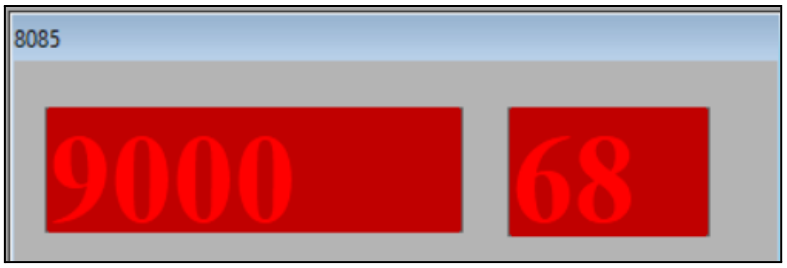


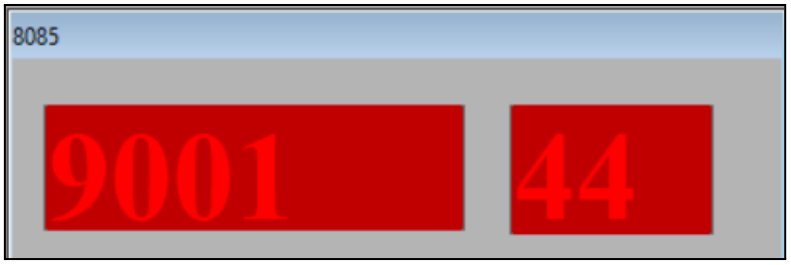
INPUT:

[9000h] - 68h

OUTPUT:

[9001h] - 44h





EXPERIMENT 13

13) Write an assembly language program for exchange the contents of memory location

CODE:

ROW NUMBER | Show ROW

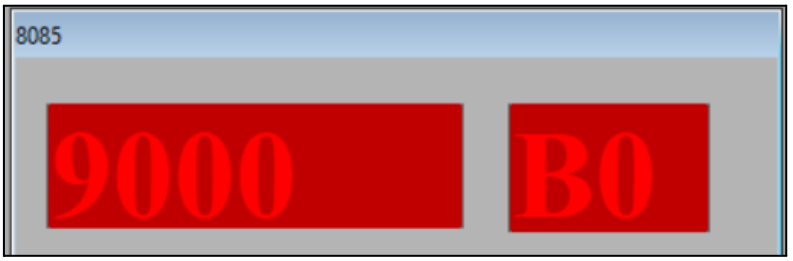
ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	3A	LDA 16 bit	3
8001	00		
8002	90		
8003	47	MOV B,A	1
8004	3A	LDA 16 bit	3
8005	01		
8006	90		
8007	32	STA 16 bit	3
8008	00		
8009	90		
800A	78	MOV A,B	1
800B	32	STA 16 bit	3
800C	01		
800D	90		
800E	76	HLT	1

STACK (LIFO)

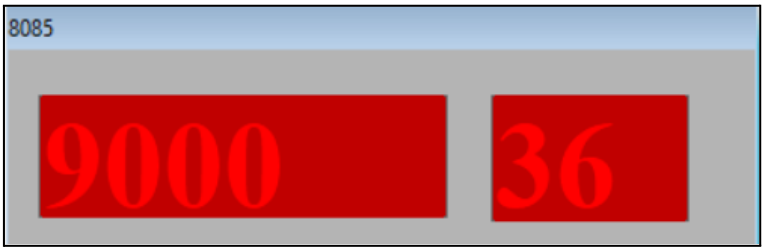
ADDRESS	DATA
8421	XX

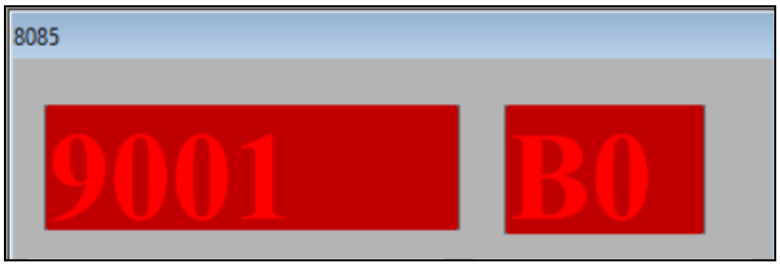
OUTPUT:

Before



After





EXPERIMENT 14

14) Write a program to find the largest number in an array of 10 elements

CODE:

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8000	06	MVI B,8 bit	2
8001	09		
8002	21	LXI H,16 bit	3
8003	00		
8004	90		
8005	7E	MOV A,M	1
8006	23	INX H	1
8007	BE	CMP M	1
8008	D2	JNC 16 bit	3
8009	0C		
800A	80		
800B	7E	MOV A,M	1
800C	23	INX H	1
800D	05	DCR B	1
800E	C2	JNZ 16 bit	3
800F	07		

STACK (LIFO)

ADDRESS	DATA
8421	XX

ROW NUMBER | Show ROW

ADDRESS	OPCODE	INSTRUCTION	BYTES
8007	BE	CMP M	1
8008	D2	JNC 16 bit	3
8009	0C		
800A	80		
800B	7E	MOV A,M	1
800C	23	INX H	1
800D	05	DCR B	1
800E	C2	JNZ 16 bit	3
800F	07		
8010	80		
8011	32	STA 16 bit	3
8012	0A		
8013	90		
8014	76	HLT	1

STACK (LIFO)

ADDRESS	DATA
8421	XX

INPUT:

```
[9000h] - 01h
[9001h] - 02h
[9002h] - 03h
[9003h] - 04h
[9004h] - 05h
[9005h] - 06h
[9006h] - 07h
[9007h] - 08h
[9008h] - 09h
[9009h] - 10h
```

OUTPUT:

```
[900Ah] - 10h
```