EXPERIMENT - 3

Computer Organization and Architecture

Aim

To demonstrate all the data transfer instructions of GNUSim 8085 microprocessor.

Syeda Reeha Quasar
14114802719
4C7

EXPERIMENT – 2

Aim:

To demonstrate all the data transfer instructions of GNUSim 8085 microprocessor.

Theory:

Instructions Used-

- **MOV** (Copy from source to destination) - This instruction copies the contents of the source register into the destination register; the contents of the source register are not altered. If one of the operands is a memory location, its location is specified by the contents of the HL registers. Example: MOV B, C or MOV B, M
- **INX** SP instruction is used to increment the SP contents by 1. INX SP instruction is a special case of INX rp instruction which increases the content of the register pair. This instruction occupies only 1-Byte in memory.
- **ADD** (Add register or memory to accumulator) - The contents of the operand (register or memory) are added to the contents of the accumulator and the result is stored in the accumulator. If the operand is a memory location, its location is specified by the contents of the HL registers. All flags are modified to reflect the result of the addition. Example: ADD B or ADD M
- **LXI** (Load register pair immediate) - The instruction loads 16-bit data in the register pair designated in the operand. Example: LXI H, 2034H or LXI H, XYZ
- **LDA** or Load The accumulator loads the contents of a memory location, specified by a 16-bit address in the operand, are copied to the accumulator. Example − LDA 2034K
- **STA** - The contents of the accumulator are copied into the memory location specified by the operand. This is a 3-byte instruction, the second byte specifies the low-order address and the third byte specifies the high-order address. Example − STA 325K
- **IN** - Input data to accumulator from a port with 8-bit address. The contents of the input port designated in the operand are read and loaded into the accumulator. Example − IN5KL.
- **OUT** - The contents of the accumulator are copied into the I/O port specified by the operand. Example − OUT K9L.

- **LHLD** - The instruction copies the contents of the memory location pointed out by the address into register L and copies the contents of the next memory location into register H. Example − LHLD 3225K
- **SHLD** - The contents of register L are stored in the memory location specified by the 16-bit address in the operand and the contents of H register are stored into the next memory location by incrementing the operand. This is a 3-byte instruction, the second byte specifies the low-order address and the third byte specifies the high-order address. Example − SHLD 3225K
- **LDAX** - The contents of the designated register pair point to a memory location. This instruction copies the contents of that memory location into the accumulator. Example − LDAX K
- **PCHL** - it loads the program counter with HL data. the content of H placed into higher order byte and L placed at low order bytes.
- **STAX(Store accumulator indirect): -** The contents of the accumulator are copied into the memory location specified by the contents of the operand (register pair). The contents of the accumulator are not altered. Eg: - STAX B (the content of accumulator is stored into the memory location specified by the BC register pair.)
- **MVI** (Move immediate 8-bit) - The 8-bit data is stored in the destination register or M, data memory. If the operand is a memory location, its location is specified by the contents of the HL registers. Example: MVI B, 57H or MVI M, 57H
- **XCHG** - The contents of register H are exchanged with the contents of register D, and the contents of register L are exchanged with the contents of register E. Example – XCHG
- **STA** - The contents of the accumulator are copied into the memory location specified by the operand. This is a 3-byte instruction, the second byte specifies the low-order address and the third byte specifies the high-order address. Example − STA 325K

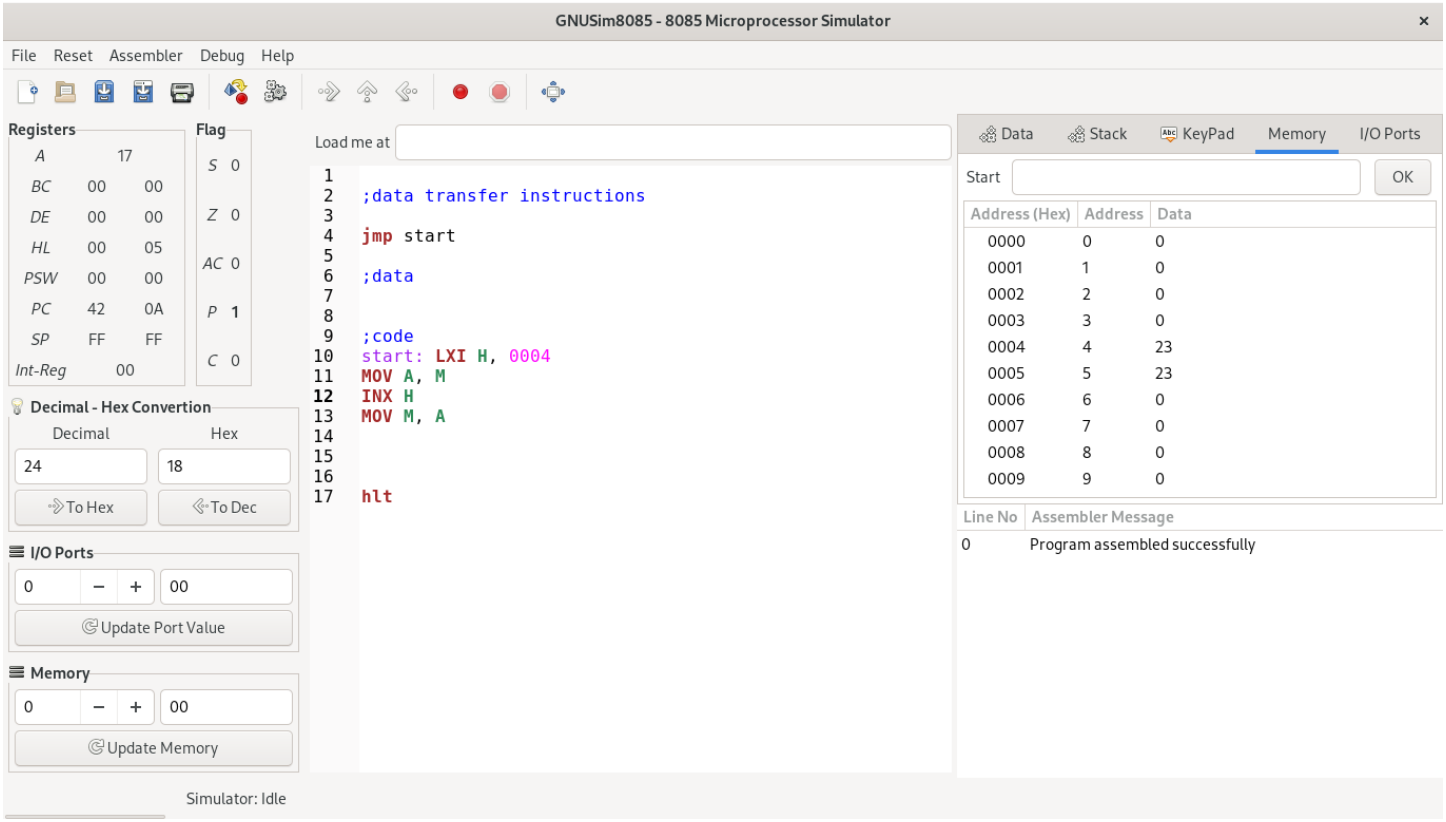
Source Codes:

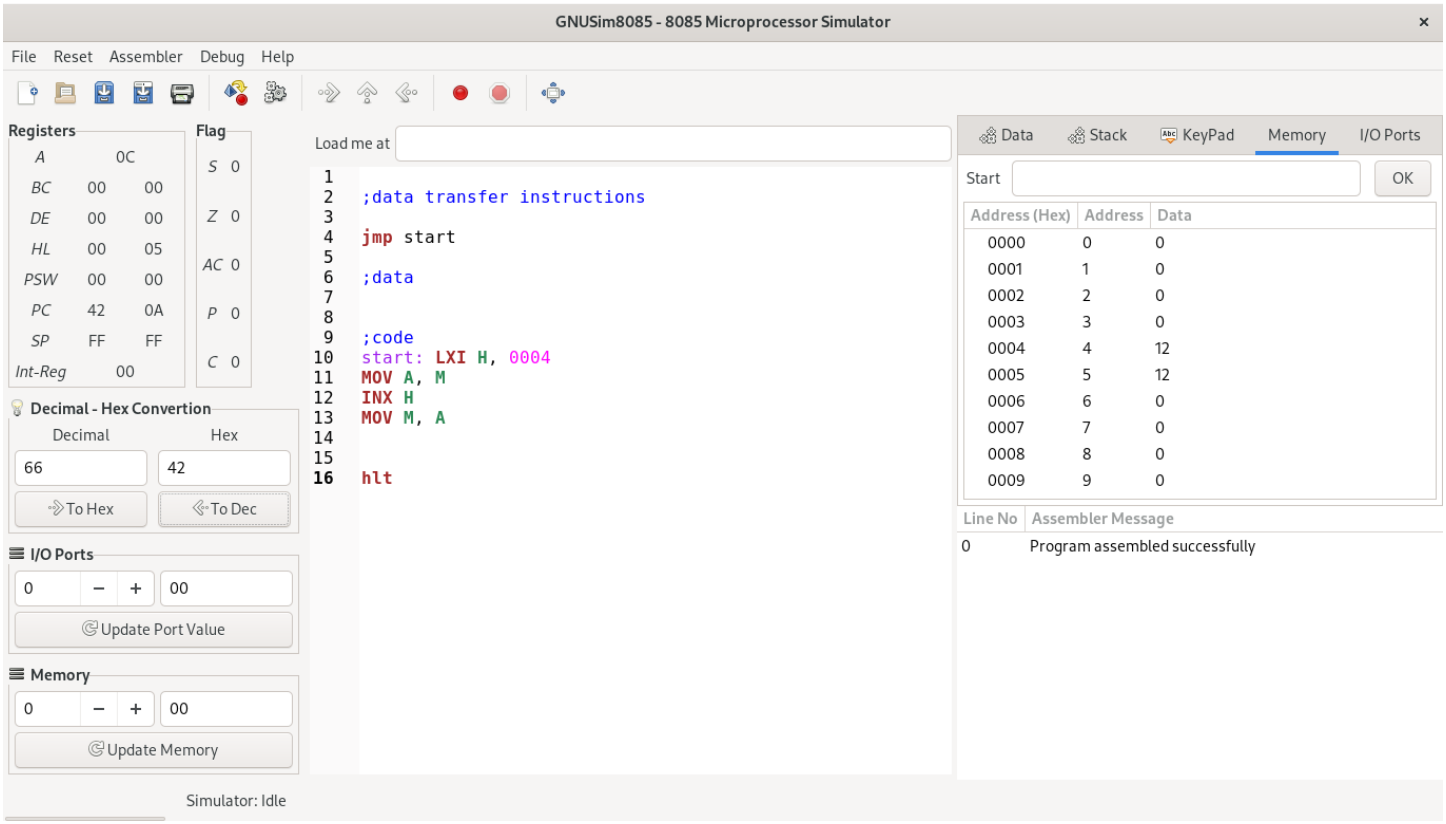
A) MOV

Source Code

```
;data transfer instructions

jmp start

;data


;code
start: LXI H, 0004
MOV A, M
INX H
MOV M, A


hlt
```

Output:


GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 17
BC 00 00
DE 00 00
HL 00 05
PSW 00 00
PC 42 0A
SP FF FF
Int-Reg 00
Flag
S 0
Z 0
AC 0
P 1
C 0
Decimal - Hex Convertion
Decimal
Hex
24
18
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: LXI H, 0004
10 MOV A, M
11 INX H
12 MOV M, A
13
14
15
16 hlt
Data Stack KeyPad Memory I/O Ports
Start
OK
Address (Hex) Address Data
0000 0 0
0001 1 0
0002 2 0
0003 3 0
0004 4 23
0005 5 23
0006 6 0
0007 7 0
0008 8 0
0009 9 0
Line No Assembler Message
0 Program assembled successfully
Simulator: Idle

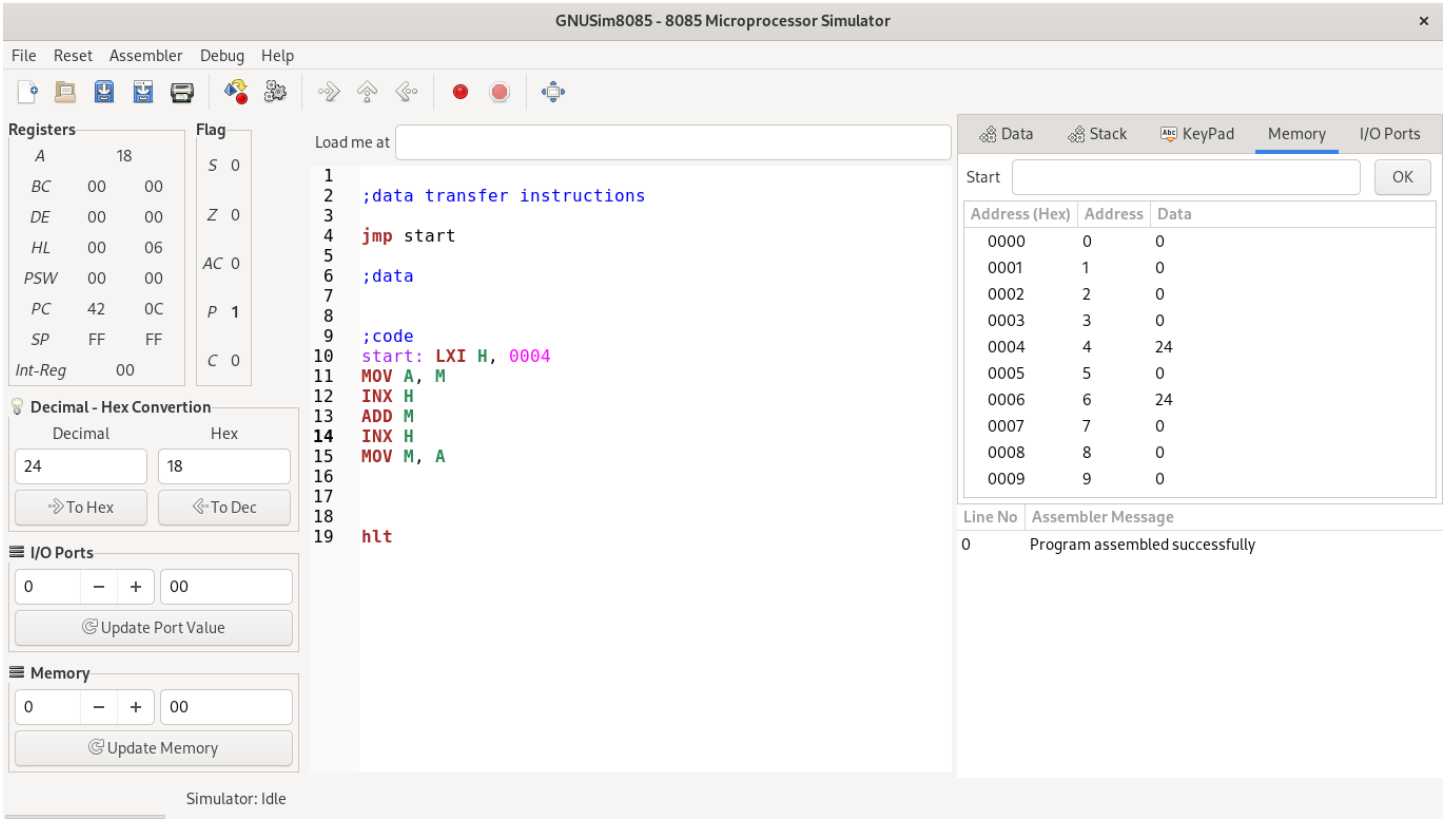

GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 0C
BC 00 00
DE 00 00
HL 00 05
PSW 00 00
PC 42 0A
SP FF FF
Int-Reg 00
Flag
S 0
Z 0
AC 0
P 0
C 0
Decimal - Hex Convertion
Decimal
Hex
66
42
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: LXI H, 0004
10 MOV A, M
11 INX H
12 MOV M, A
13
14
15 hlt
Data Stack KeyPad Memory I/O Ports
Start
OK
Address (Hex) Address Data
0000 0 0
0001 1 0
0002 2 0
0003 3 0
0004 4 12
0005 5 12
0006 6 0
0007 7 0
0008 8 0
0009 9 0
Line No Assembler Message
0 Program assembled successfully
Simulator: Idle

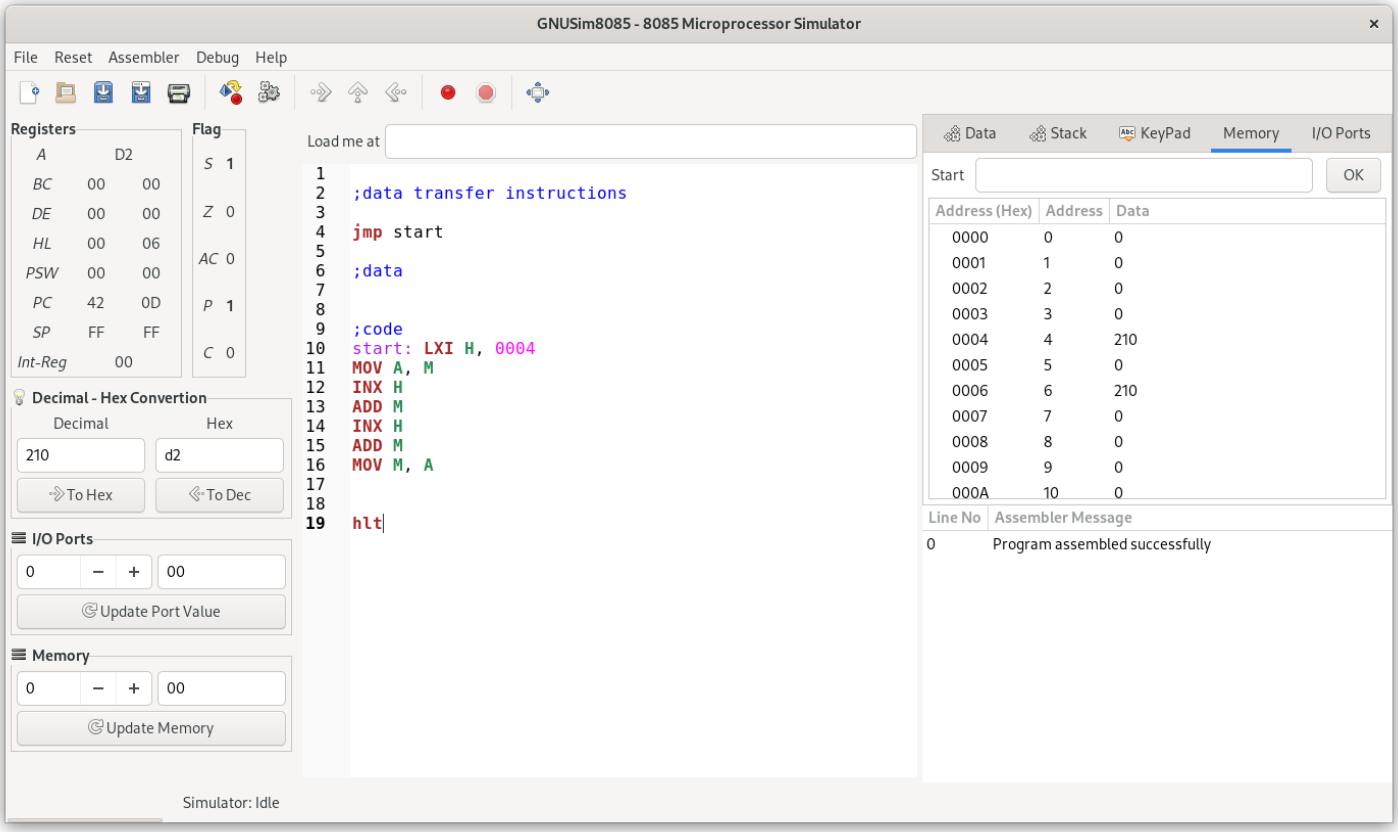
A) Memory Transfer and LXI and ADD

Source Code

```
;data transfer instructions

jmp start

;data


;code
start: LXI H, 0004
MOV A, M
INX H
ADD M
INX H
ADD M
MOV M, A


hlt
```

Output:


GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 18
BC 00 00
DE 00 00
HL 00 06
PSW 00 00
PC 42 0C
SP FF FF
Int-Reg 00
Flag
S 0
Z 0
AC 0
P 1
C 0
Decimal - Hex Convertion
Decimal
Hex
24
18
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: LXI H, 0004
10 MOV A, M
11 INX H
12 ADD M
13 INX H
14 MOV M, A
15
16
17 hlt
Data
Stack
KeyPad
Memory
I/O Ports
Start
OK
Address (Hex) Address Data
0000 0 0
0001 1 0
0002 2 0
0003 3 0
0004 4 24
0005 5 0
0006 6 24
0007 7 0
0008 8 0
0009 9 0
Line No Assembler Message
0 Program assembled successfully
Simulator: Idle

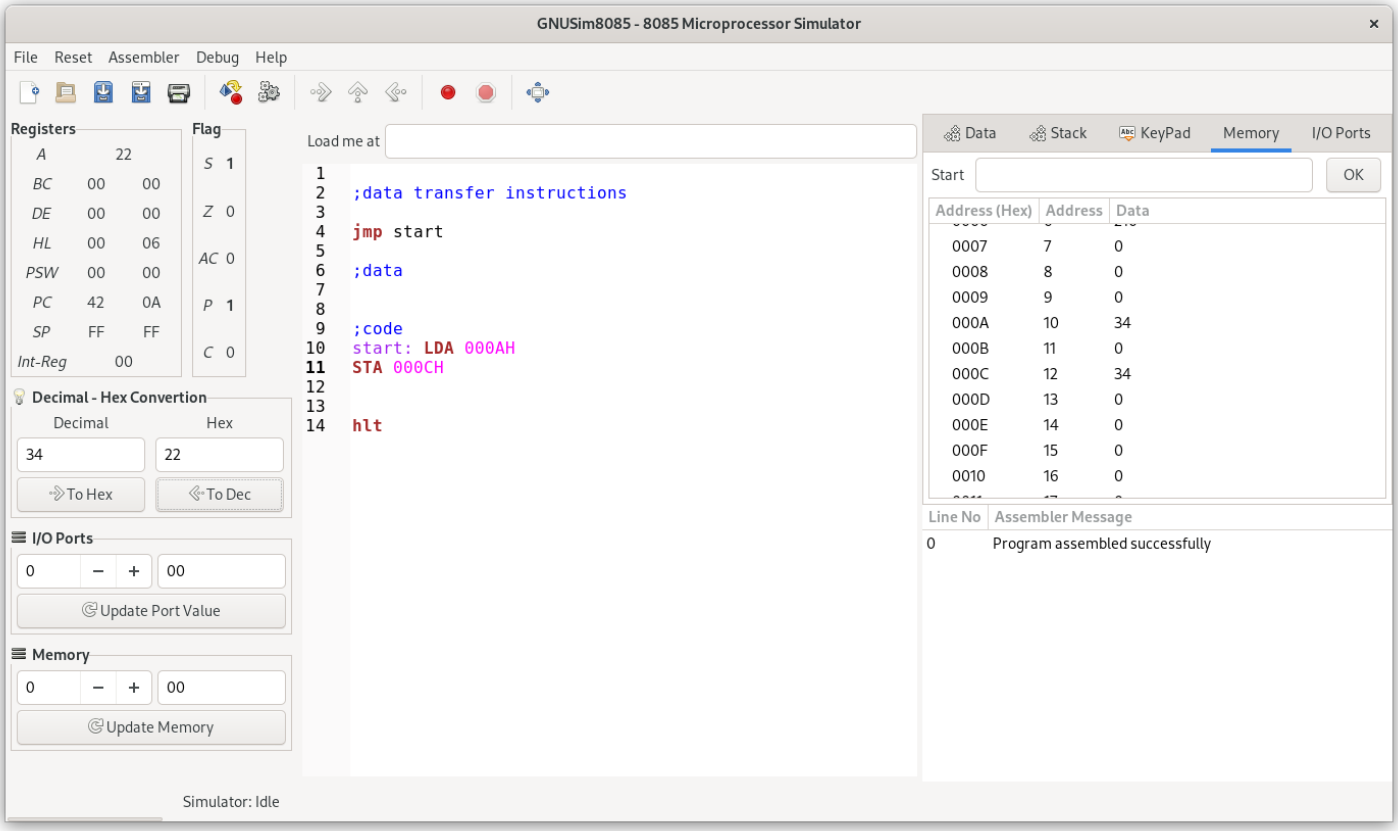

GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A D2
BC 00 00
DE 00 00
HL 00 06
PSW 00 00
PC 42 0D
SP FF FF
Int-Reg 00
Flag
S 1
Z 0
AC 0
P 1
C 0
Decimal - Hex Convertion
Decimal
Hex
210
d2
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1
2 ;data transfer instructions
3
4 jmp start
5
6 ;data
7
8
9 ;code
10 start: LXI H, 0004
11 MOV A, M
12 INX H
13 ADD M
14 INX H
15 ADD M
16 MOV M, A
17
18
19 hlt
Data
Stack
KeyPad
Memory
I/O Ports
Start
OK
Address (Hex) Address Data
0000 0 0
0001 1 0
0002 2 0
0003 3 0
0004 4 210
0005 5 0
0006 6 210
0007 7 0
0008 8 0
0009 9 0
000A 10 0
Line No Assembler Message
0 Program assembled successfully
Simulator: Idle

B) Memory Transfer and LXI and ADD

Source Code

```
;LDA -> ACC TO DATA
;STA A TO MEMORY

;MEMORY OPERATIONS

;data transfer instructions

jmp start

;data


;code
start: LDA 000AH
STA 000CH


hlt
```

Output:

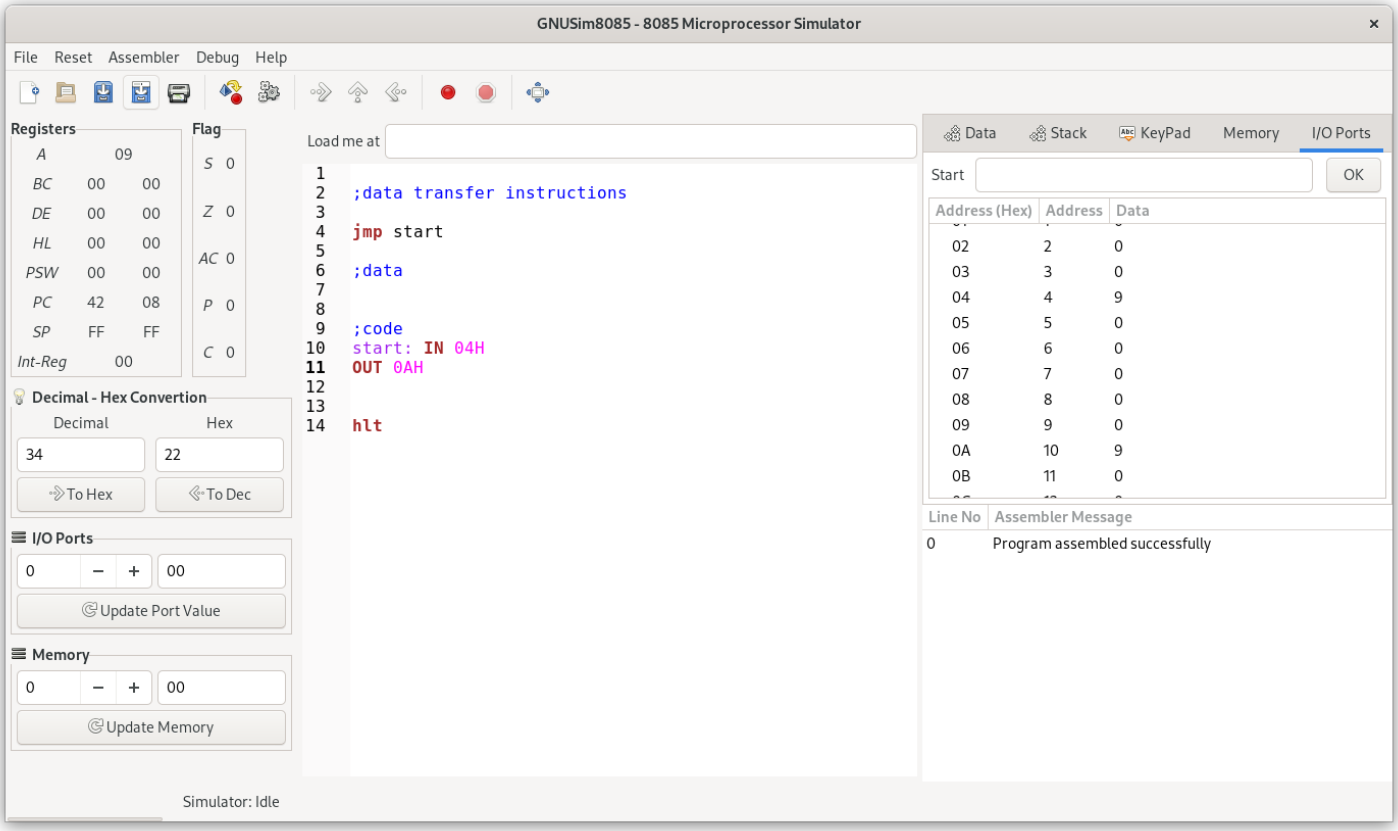

GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 22
BC 00 00
DE 00 00
HL 00 06
PSW 00 00
PC 42 0A
SP FF FF
Int-Reg 00
Flag
S 1
Z 0
AC 0
P 1
C 0
Decimal - Hex Convertion
Decimal
Hex
34
22
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: LDA 000AH
10 STA 000CH
11
12
13 hlt
Data
Stack
KeyPad
Memory
I/O Ports
Start
OK
Address (Hex)
Address
Data
0007 7 0
0008 8 0
0009 9 0
000A 10 34
000B 11 0
000C 12 34
000D 13 0
000E 14 0
000F 15 0
0010 16 0
Line No
Assembler Message
0
Program assembled successfully
Simulator: Idle

C) I/P TO MEMORY COMMUNICATION , IN and OUT instructions

Source Code

```
;I/P TO MEMORY COMMUNICATION


;data transfer instructions

jmp start

;data


;code
start: IN 04H
OUT 0AH


hlt
```

Output:

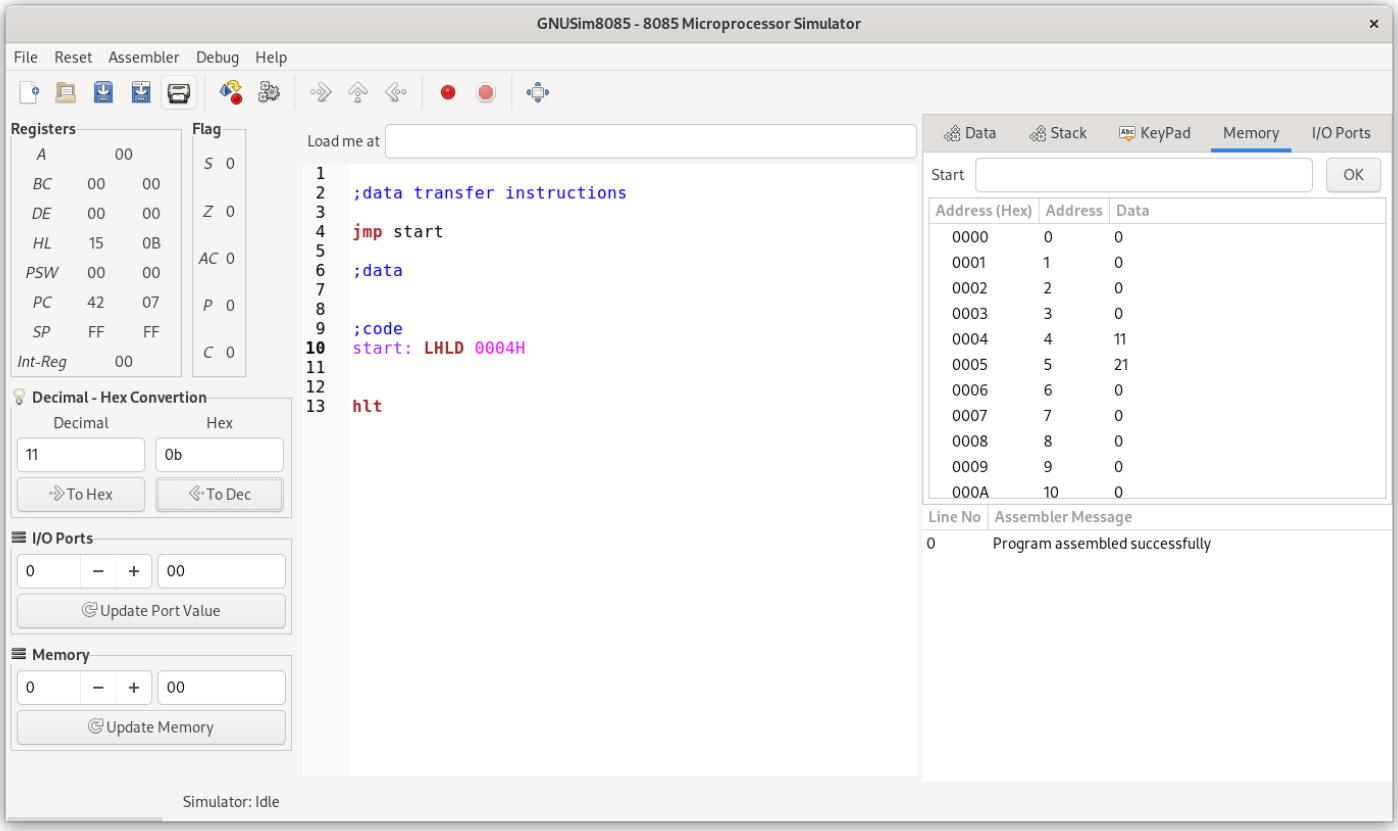

GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 09
BC 00 00
DE 00 00
HL 00 00
PSW 00 00
PC 42 08
SP FF FF
Int-Reg 00
Flag
S 0
Z 0
AC 0
P 0
C 0
Decimal - Hex Convertion
Decimal
Hex
34
22
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: IN 04H
10 OUT 0AH
11
12
13 hlt
Data
Stack
KeyPad
Memory
I/O Ports
Start
OK
Address (Hex)
Address
Data
02 2 0
03 3 0
04 4 9
05 5 0
06 6 0
07 7 0
08 8 0
09 9 0
0A 10 9
0B 11 0
Line No
Assembler Message
0
Program assembled successfully
Simulator: Idle

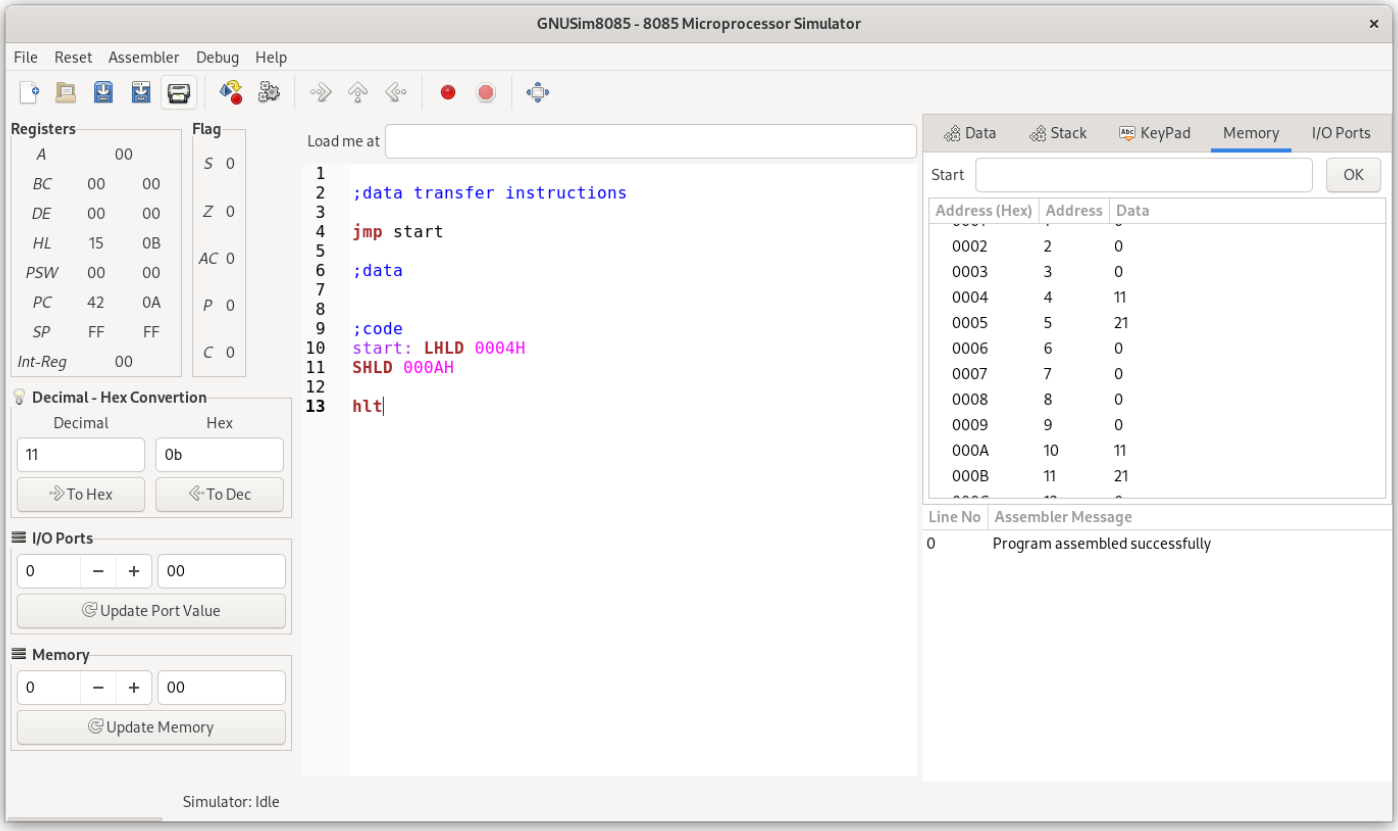
D) LHLD and SHLD instructions

Source Code

```
;data transfer instructions

jmp start

;data


;code
start: LHLD 0004H
SHLD 000AH

hlt
```

Output:


GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 00
BC 00 00
DE 00 00
HL 15 0B
PSW 00 00
PC 42 07
SP FF FF
Int-Reg 00
Flag
S 0
Z 0
AC 0
P 0
C 0
Decimal - Hex Convertion
Decimal
Hex
11
0b
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: LHLD 0004H
10
11 hlt
Data
Stack
KeyPad
Memory
I/O Ports
Start
OK
Address (Hex)
Address
Data
0000 0 0
0001 1 0
0002 2 0
0003 3 0
0004 4 11
0005 5 21
0006 6 0
0007 7 0
0008 8 0
0009 9 0
000A 10 0
Line No
Assembler Message
0
Program assembled successfully
Simulator: Idle

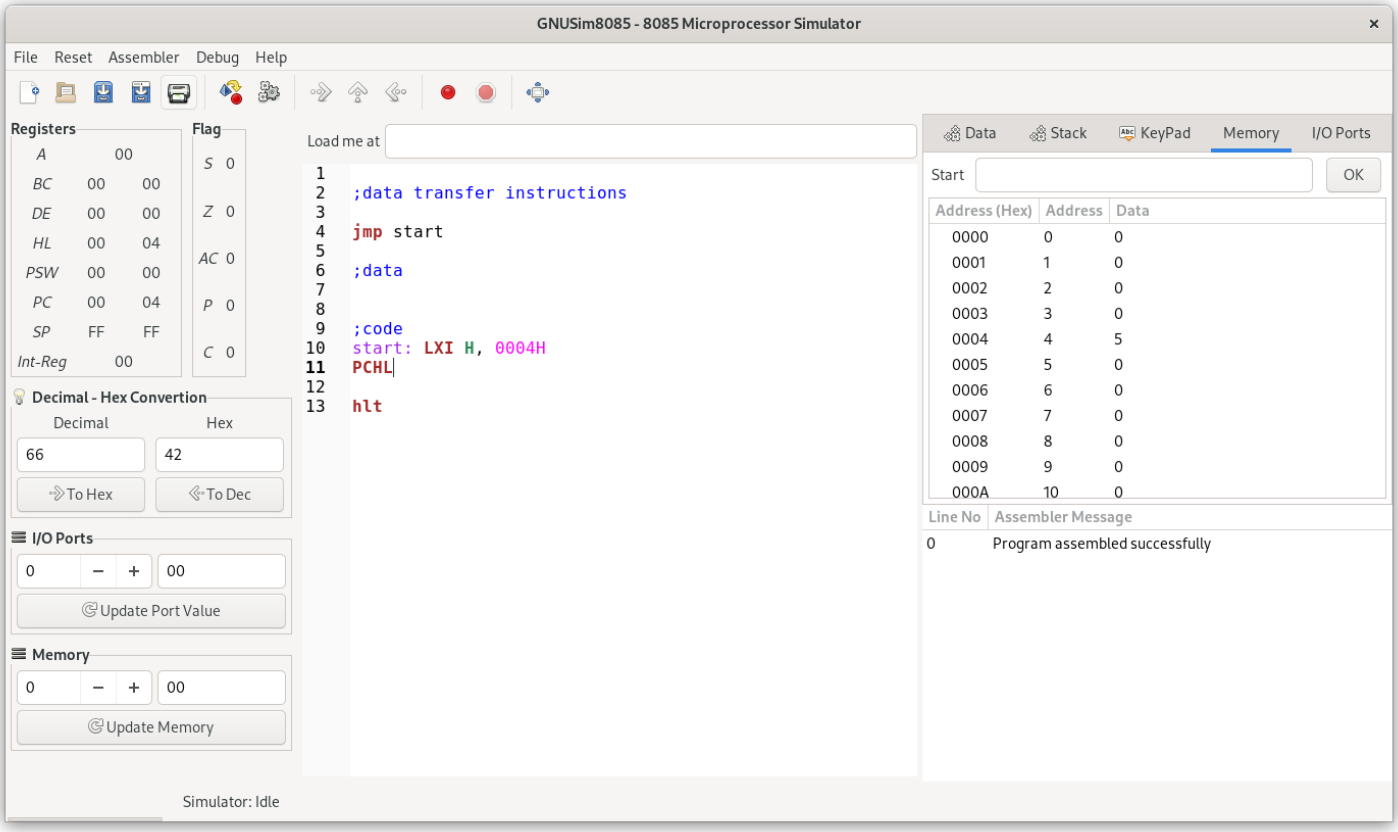

GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 00
BC 00 00
DE 00 00
HL 15 0B
PSW 00 00
PC 42 0A
SP FF FF
Int-Reg 00
Flag
S 0
Z 0
AC 0
P 0
C 0
Decimal - Hex Convertion
Decimal
Hex
11
0b
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: LHLD 0004H
10 SHLD 000AH
11
12 hlt
Data
Stack
KeyPad
Memory
I/O Ports
Start
OK
Address (Hex)
Address
Data
0002 2 0
0003 3 0
0004 4 11
0005 5 21
0006 6 0
0007 7 0
0008 8 0
0009 9 0
000A 10 11
000B 11 21
Line No
Assembler Message
0
Program assembled successfully
Simulator: Idle

E) PCHL instruction

Source Code

```
;data transfer instructions

jmp start

;data


;code
start: LXI H, 0004H
PCHL

hlt
```

Output:


GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Registers
A 00
BC 00 00
DE 00 00
HL 00 04
PSW 00 00
PC 00 04
SP FF FF
Int-Reg 00
Flag
S 0
Z 0
AC 0
P 0
C 0
Decimal - Hex Convertion
Decimal
Hex
66
42
To Hex
To Dec
I/O Ports
0
00
Update Port Value
Memory
0
00
Update Memory
Load me at
1 ;data transfer instructions
2
3 jmp start
4
5 ;data
6
7
8 ;code
9 start: LXI H, 0004H
10 PCHL
11
12 hlt
13
Data
Stack
KeyPad
Memory
I/O Ports
Start
OK
Address (Hex)
Address
Data
0000 0 0
0001 1 0
0002 2 0
0003 3 0
0004 4 5
0005 5 0
0006 6 0
0007 7 0
0008 8 0
0009 9 0
000A 10 0
Line No
Assembler Message
0
Program assembled successfully
Simulator: Idle