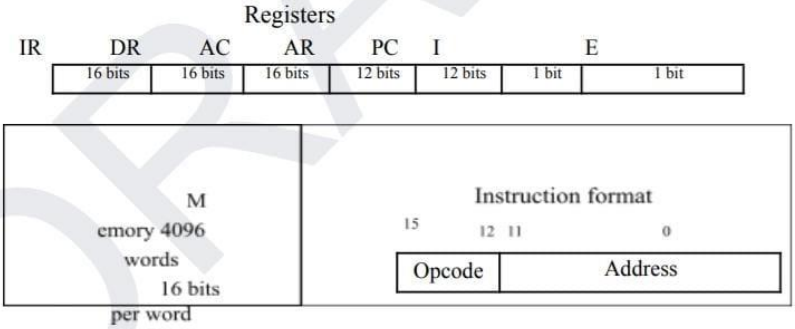
(Use Simulator – CPU Sim 3.6.9 or any higher version for the implementation)

1. Create a machine based on the following architecture:



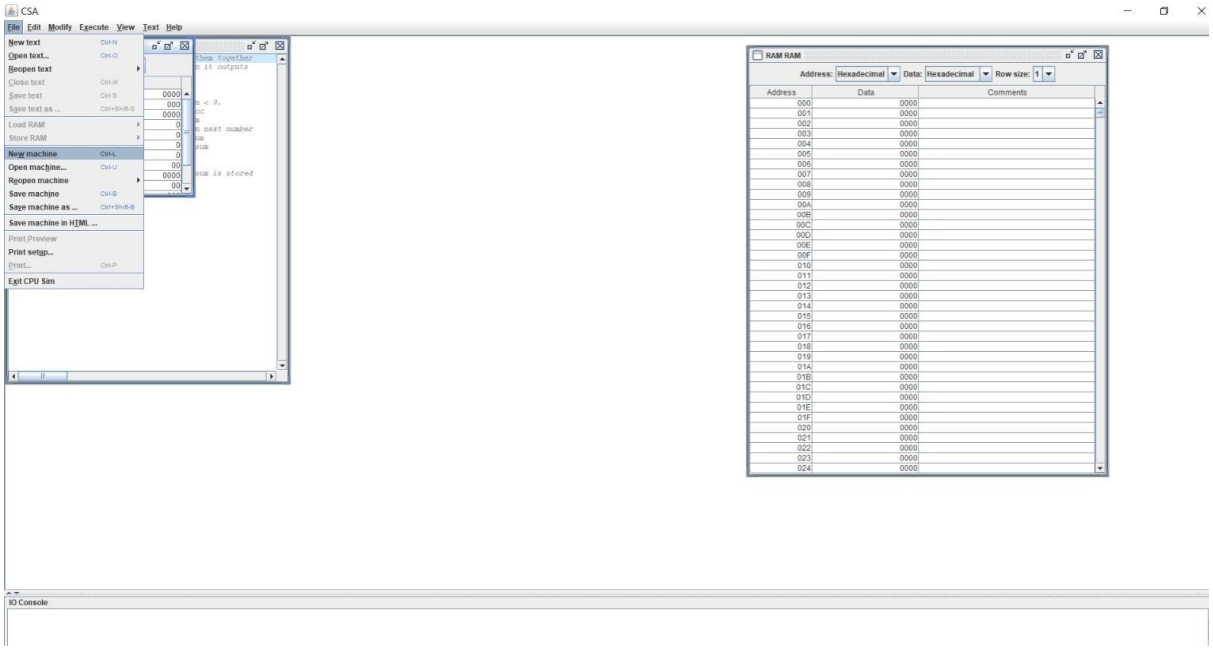

Basic Computer Instructions

| Memory Reference | | | Register Reference | |
|---|---|---|---|---|
| Symbol | Hex | | Symbol | Hex |
| AND | 0xxx | Direct | CLA | 7800 |
| ADD | 1xxx | | CLE | 7400 |
| LDA | 2xxx | | CMA | 7200 |
| STA | 3xxx | | CME | 7100 |
| BUN | 4xxx | | CIR | 7080 |

---

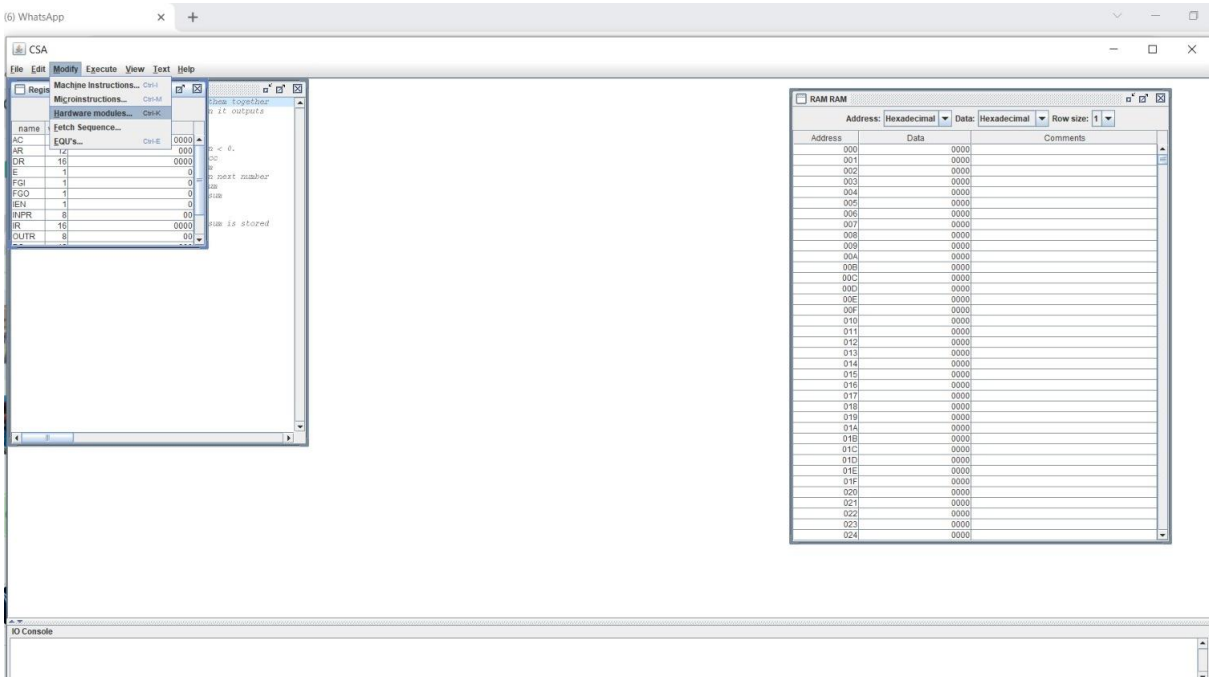
| | | | | |
|---|---|---|---|---|
| BSA | 5xxx | Addressing | CIL | 7040 |
| ISZ | 6xxx | | INC | 7020 |
| AND_I | 8xxx | Indirect Addressing | SPA | 7010 |
| ADD_I | 9xxx | | SNA | 7008 |
| LDA_I | Axxx | | SZA | 7004 |
| STA_I | Bxxx | | SZE | 7002 |
| BUN_I | Cxxx | | HLT | 7001 |
| BSA_I | Dxxx | | INP | F800 |
| ISZ_I | Exxx | | OUT | F400 |

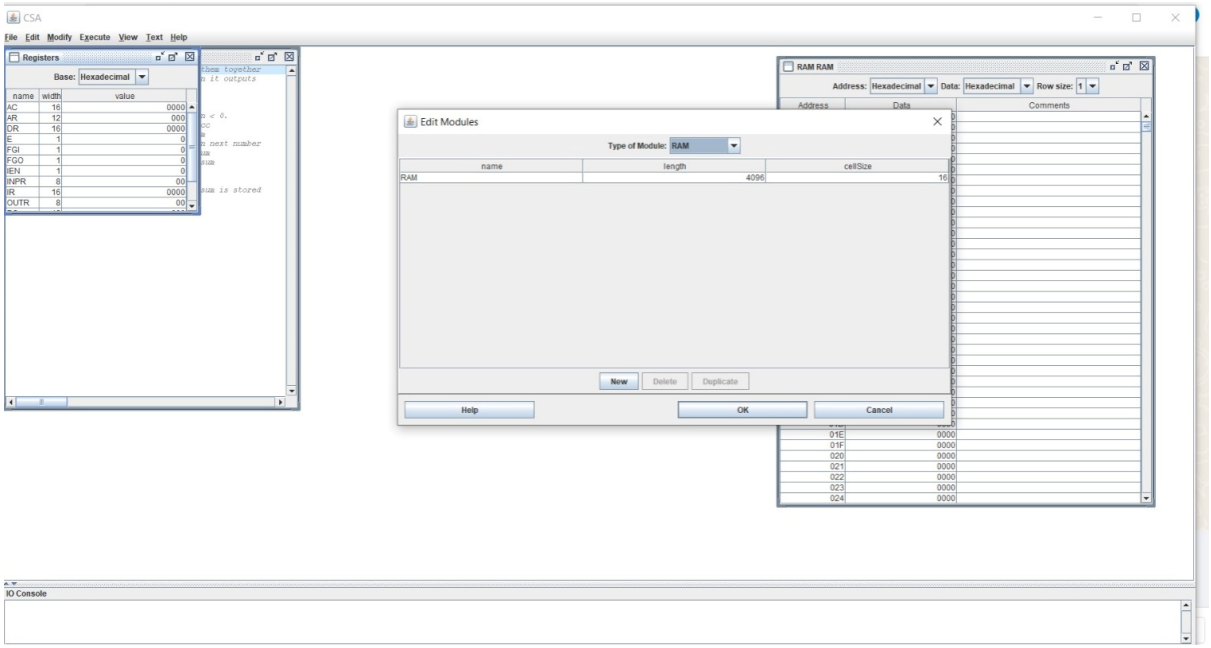
**Step 1. Go to file > new machine ,create a new machine and save it with (.cpu )extension.**

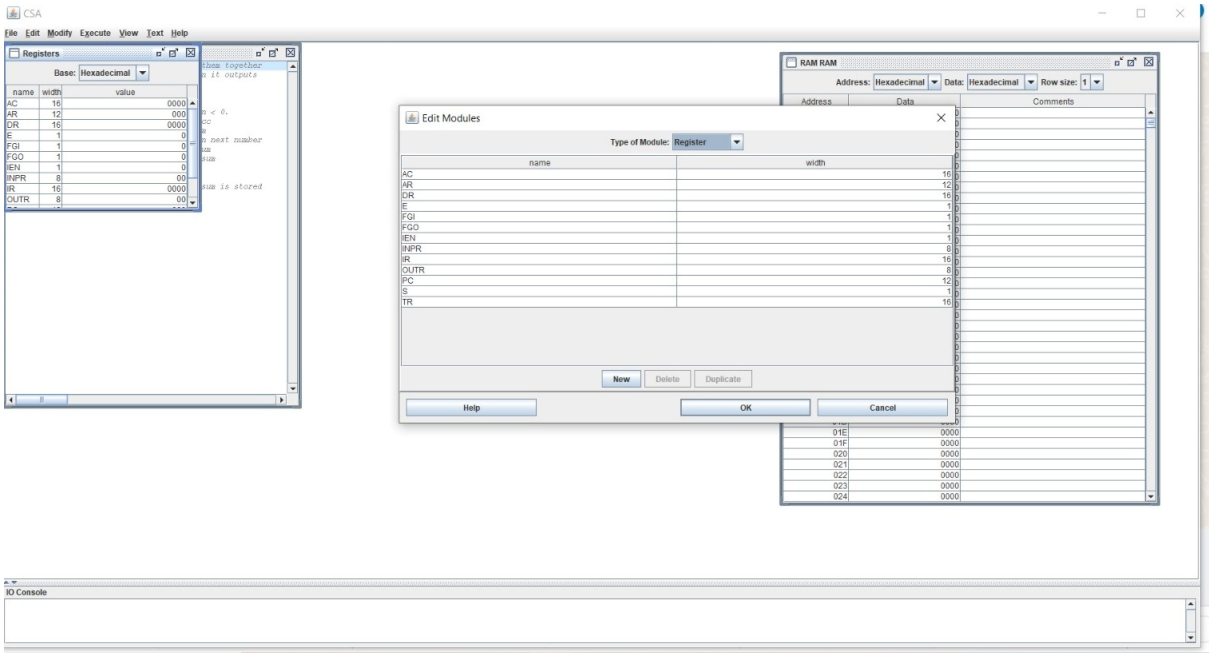


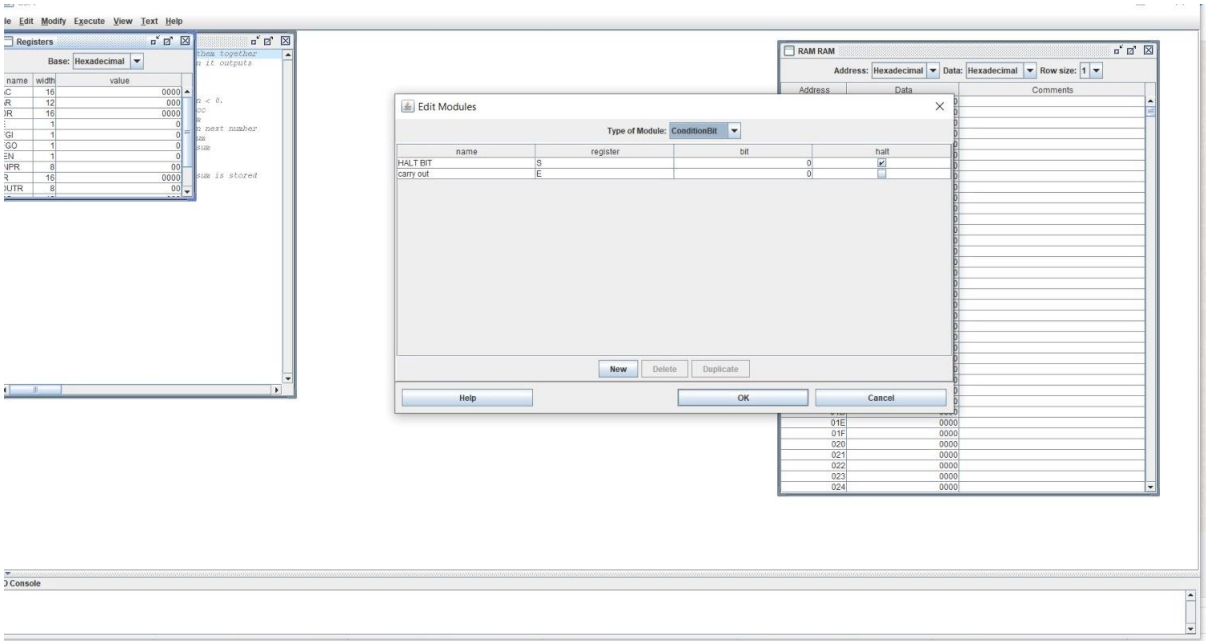

**Step 2 . GO TO MODIFY AND MAKE HARDWARE MODULES WHICH INCLUDES:**

(I)RAM (II) REGISTERS (4096*16 BITS) (III) CONDITIONAL BITS



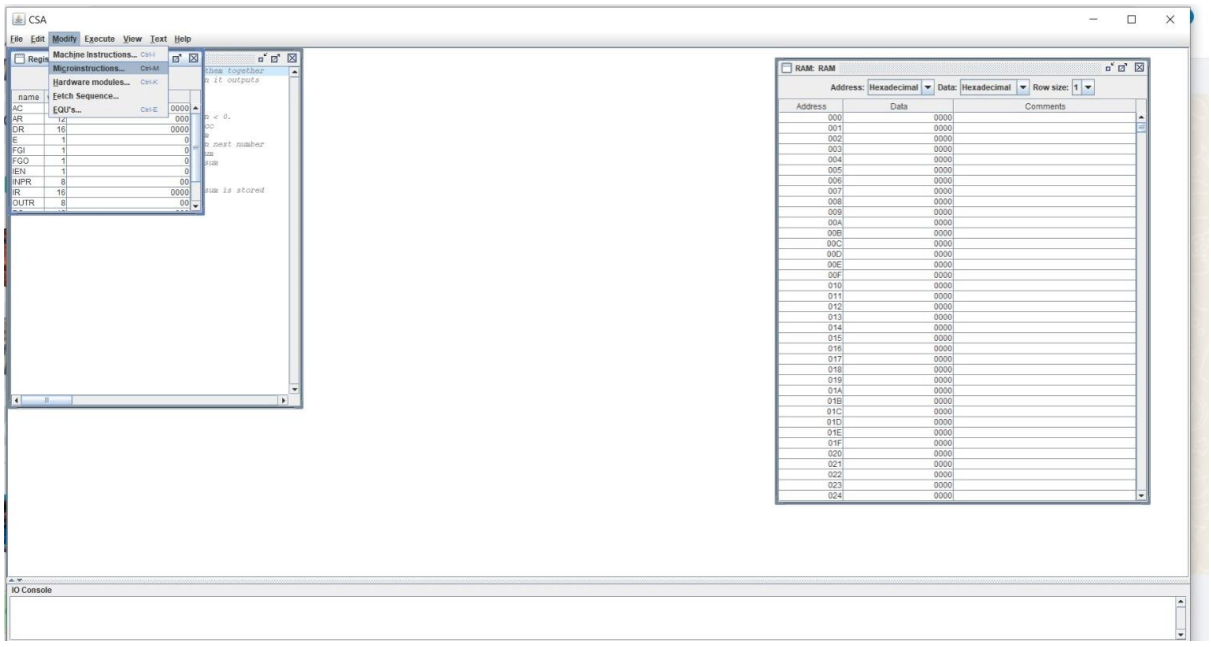

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
them together
it outputs
next number
num is stored
RAM RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
01E 0000
01F 0000
020 0000
021 0000
022 0000
023 0000
024 0000
Edit Modules
Type of Module: RAM
name length cellSize
RAM 4096 16
New Delete Duplicate
Help OK Cancel
IO Console

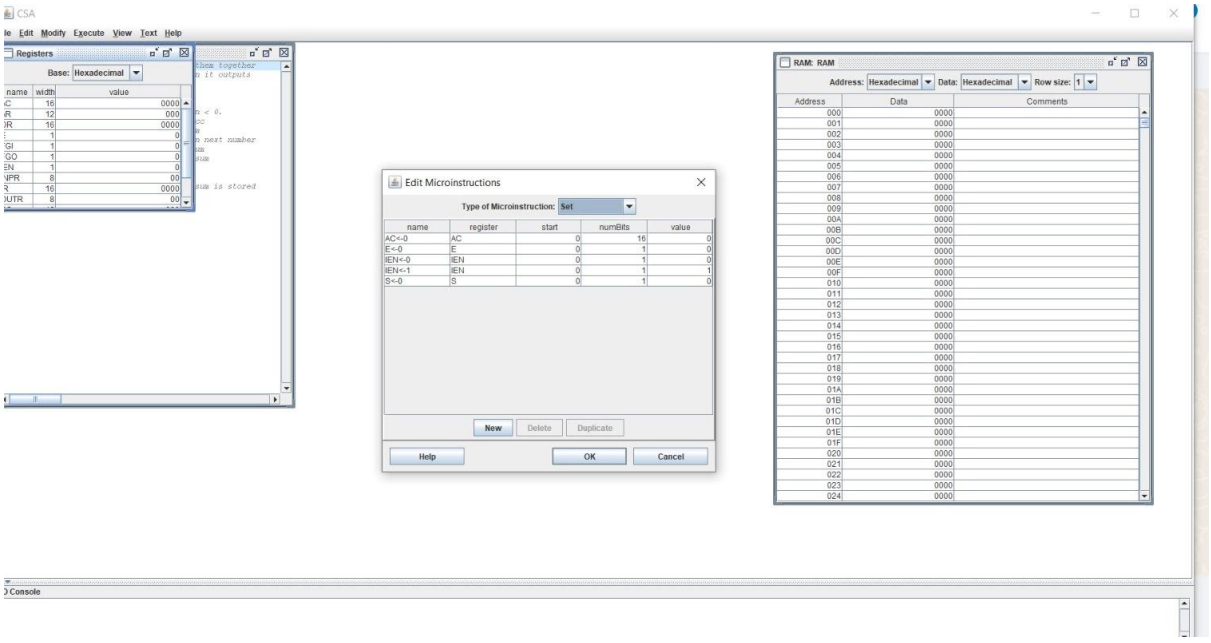

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
them together
it outputs
next number
num is stored
RAM RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
01E 0000
01F 0000
020 0000
021 0000
022 0000
023 0000
024 0000
Edit Modules
Type of Module: Register
name width
AC 16
AR 12
DR 16
E 1
FGI 1
FGO 1
IEN 1
INPR 8
IR 16
OUTR 8
PC 12
S 1
TR 16
New Delete Duplicate
Help OK Cancel
IO Console

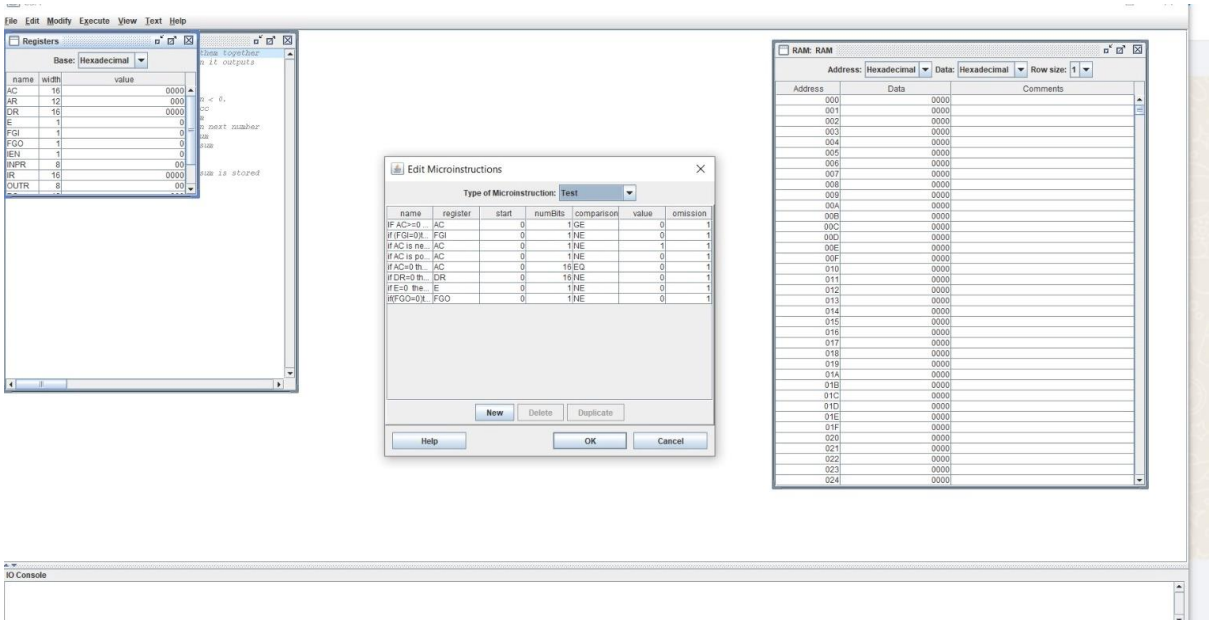


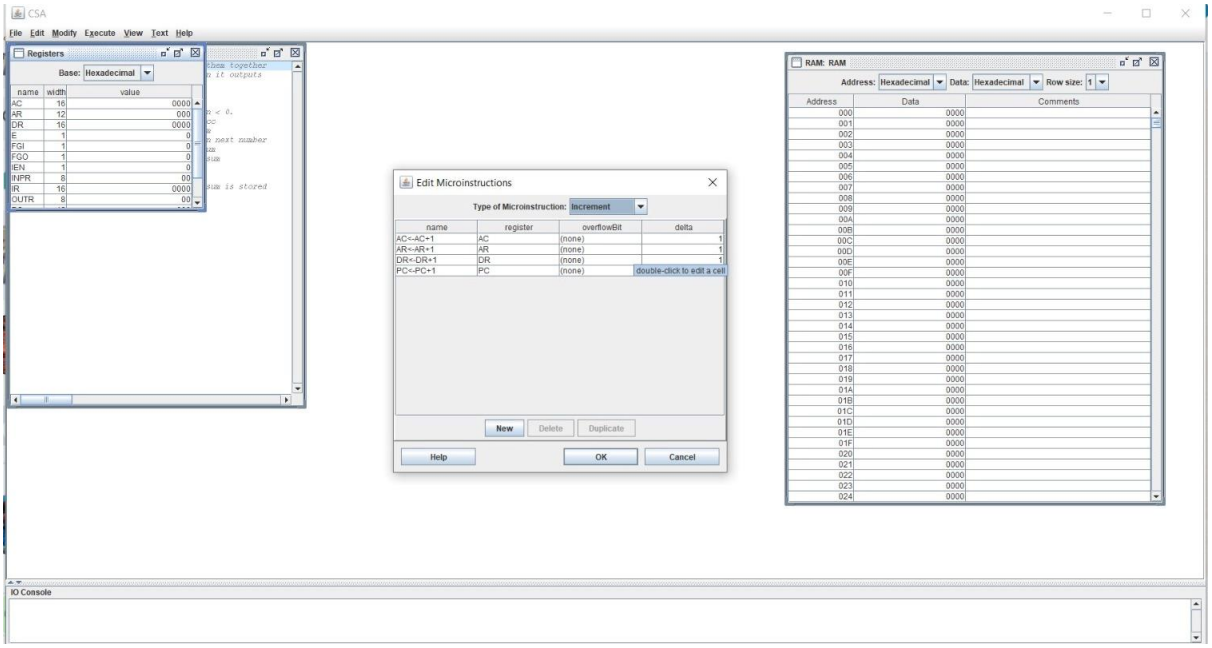

**STEP 3. GO TO MODIFY AND MAKE MICROINSTRUCTIONS WHICH INCLUDES:**

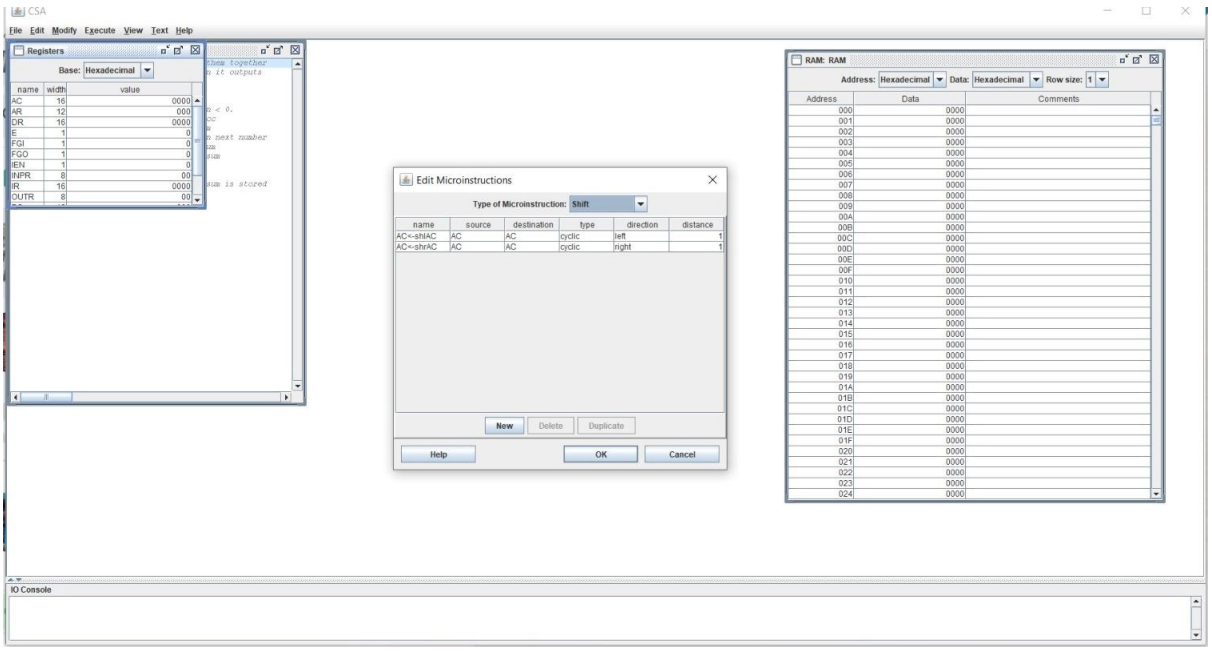
1.SET 2. TEST 3.INCREMENT 4.SHIFT 5. LOGICAL 6.AIRTHEMATIC 7. TRANSFER R TO R 8. DECODE 9. SET CONDITION BIT 10. I/0 11. MEMORY ACCESS

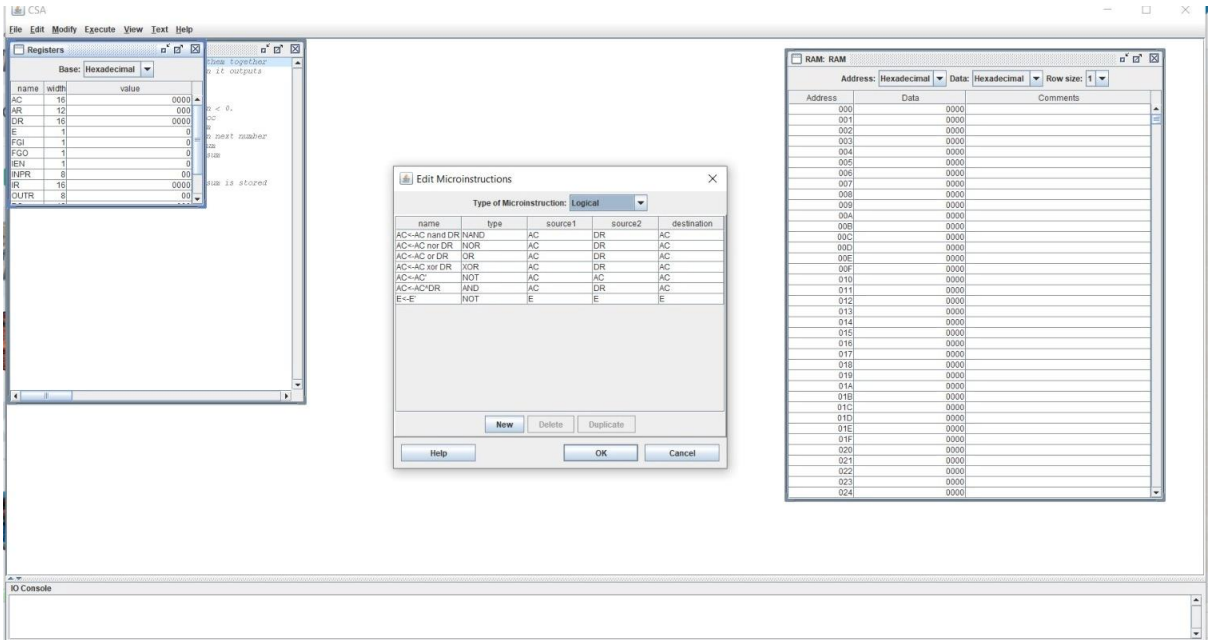


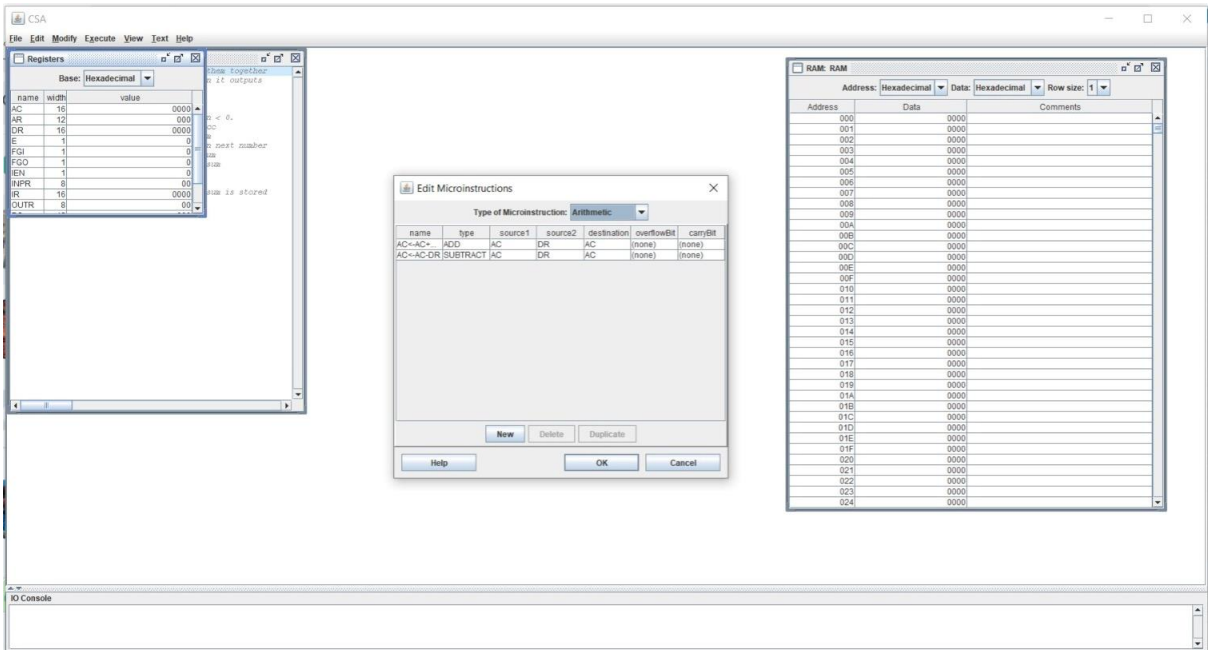

CSA
Edit Microinstructions
Type of Microinstruction: Set
New
Delete
Duplicate
Help
OK
Cancel
RAM: RAM
Address: Hexadecimal
Data: Hexadecimal
Row size: 1
Address
Data
Comments
Console

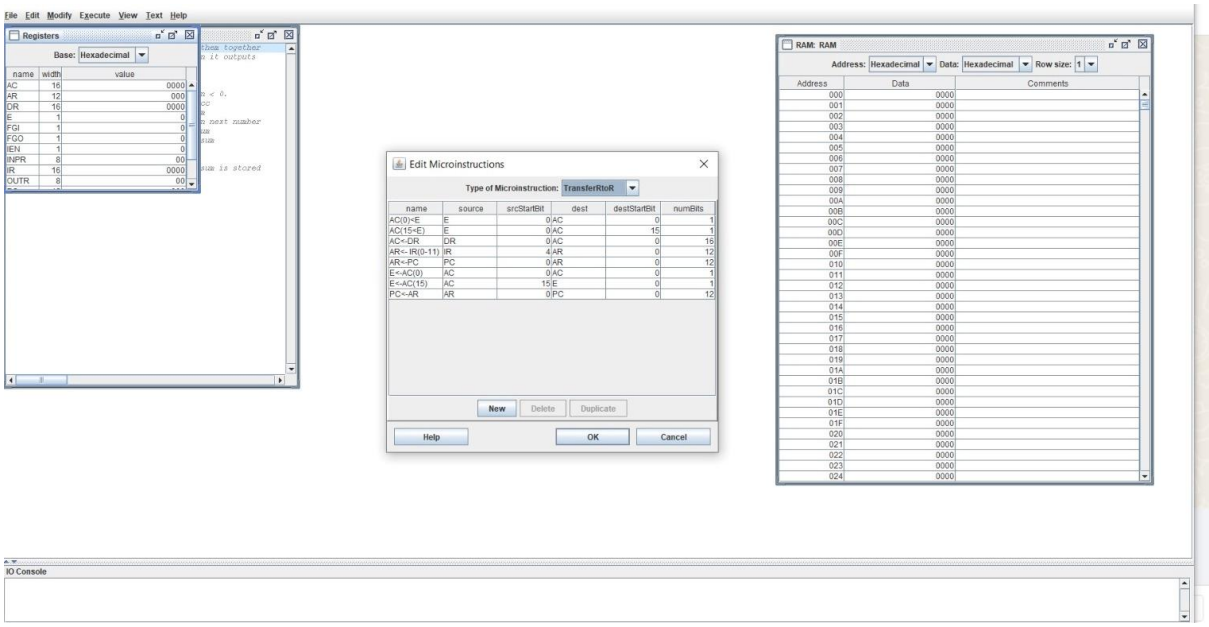

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
Edit Microinstructions
Type of Microinstruction: Test
New
Delete
Duplicate
Help
OK
Cancel
RAM: RAM
Address: Hexadecimal
Data: Hexadecimal
Row size: 1
Address
Data
Comments
IO Console

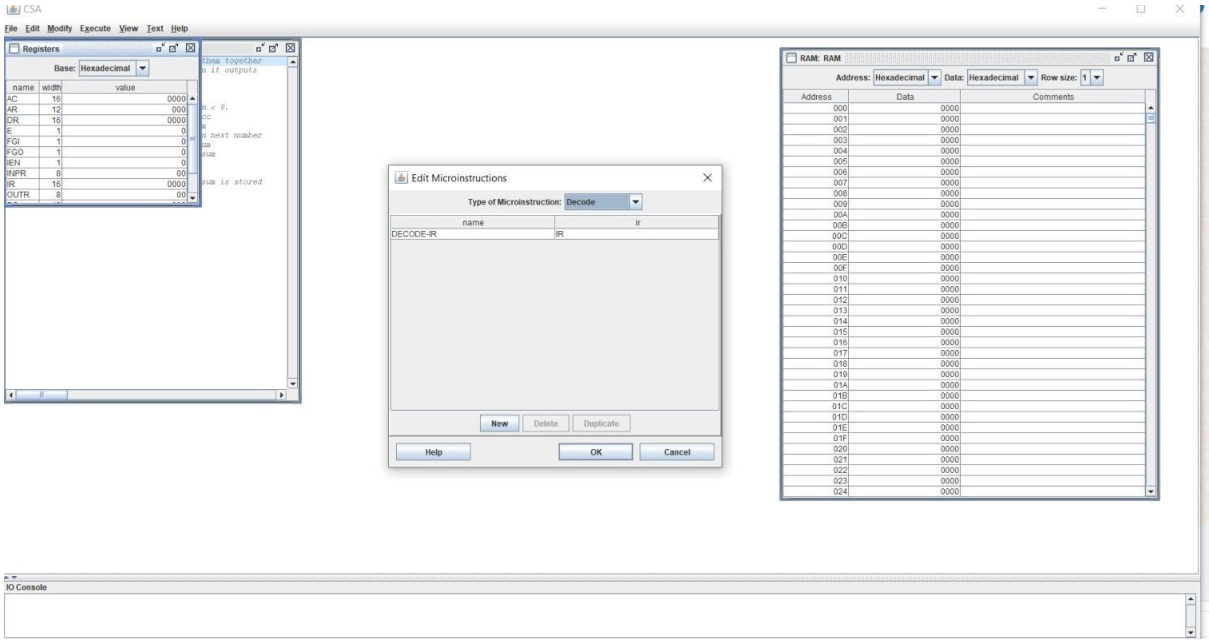

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: Increment
name register overflowBit delta
AC<-AC+1 AC (none) 1
AR<-AR+1 AR (none) 1
DR<-DR+1 DR (none) 1
PC<-PC+1 PC (none) 1
double-click to edit a cell
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console

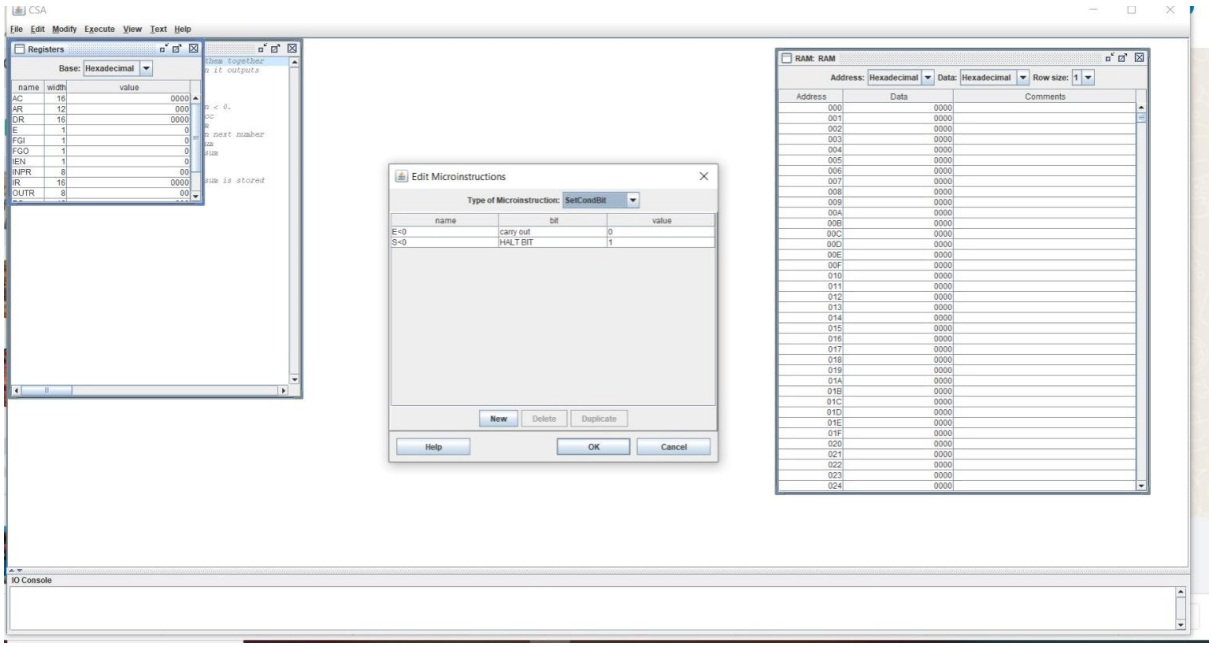

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: Shift
name source destination type direction distance
AC<-shlAC AC AC cyclic left 1
AC<-shrAC AC AC cyclic right 1
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console

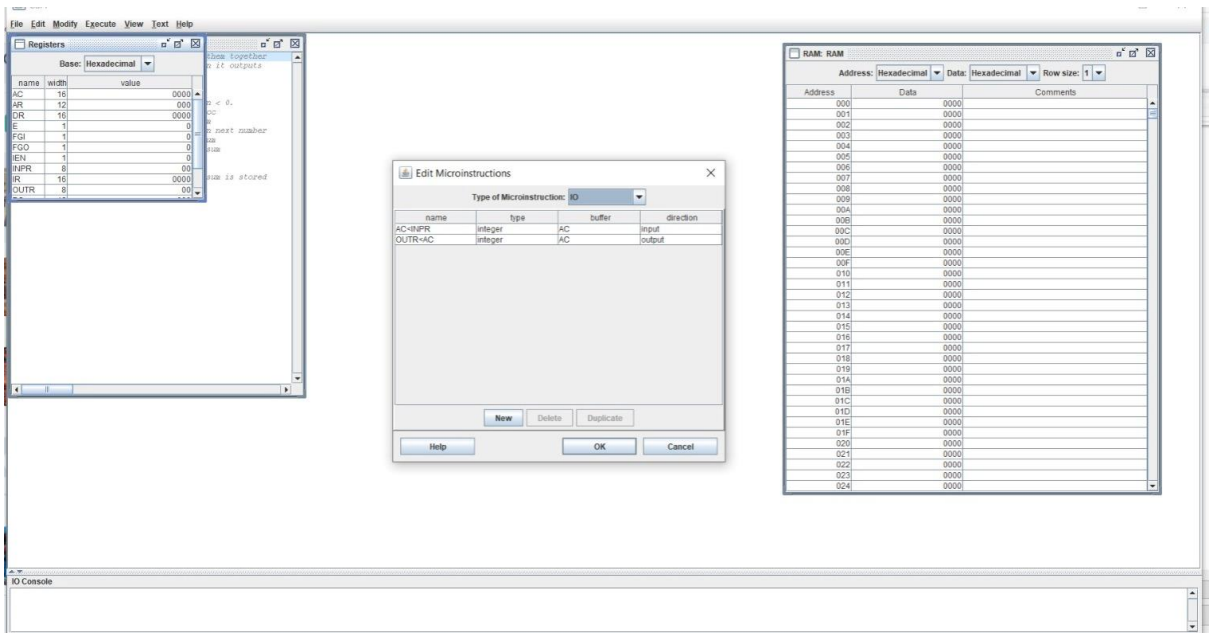

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: Logical
name type source1 source2 destination
AC<-AC nand DR NAND AC DR AC
AC<-AC nor DR NOR AC DR AC
AC<-AC or DR OR AC DR AC
AC<-AC xor DR XOR AC DR AC
AC<-AC' NOT AC AC AC
AC<-AC*DR AND AC DR AC
E<-E' NOT E E E
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console

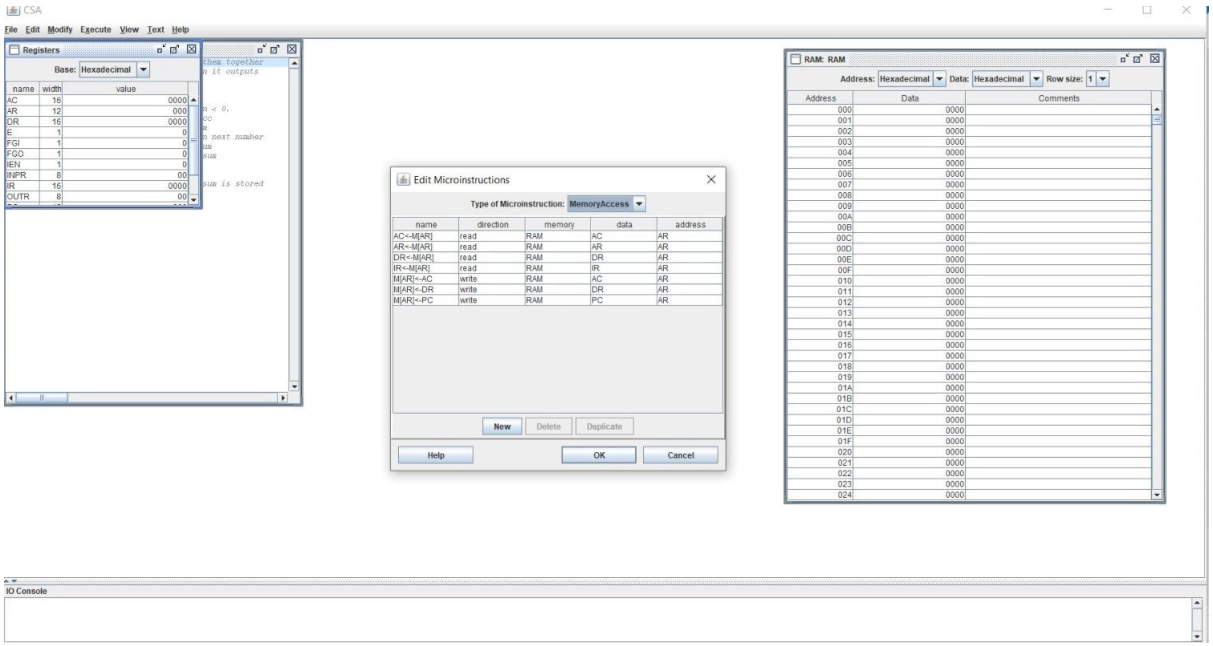

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: Arithmetic
name type source1 source2 destination overflowBit carryBit
AC<-AC+ ADD AC DR AC (none) (none)
AC<-AC-DR SUBTRACT AC DR AC (none) (none)
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console


File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: TransferRtoR
name source srcStartBit dest destStartBit numBits
AC(0)<E E 0 AC 0 1
AC(15<E) E 0 AC 15 1
AC<-DR DR 0 AC 0 16
AR<- IR(0-11) IR 4 AR 0 12
AR<-PC PC 0 AR 0 12
E<-AC(0) AC 0 AC 0 1
E<-AC(15) AC 15 E 0 1
PC<-AR AR 0 PC 0 12
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console

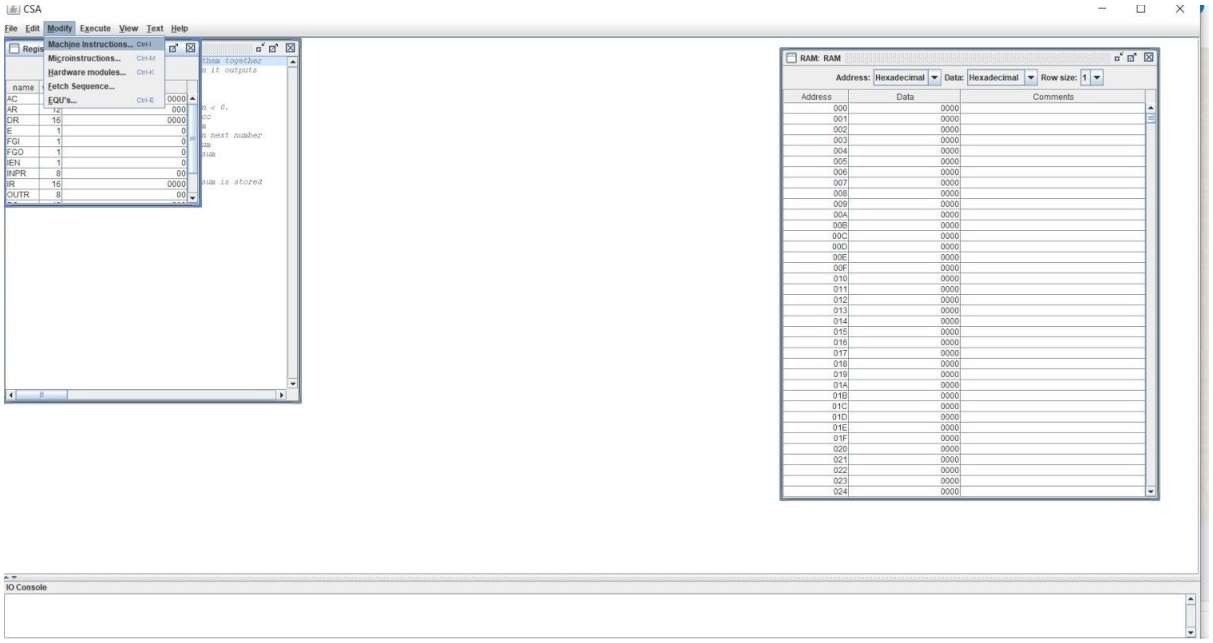

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: Decode
name ir
DECODE-IR IR
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console

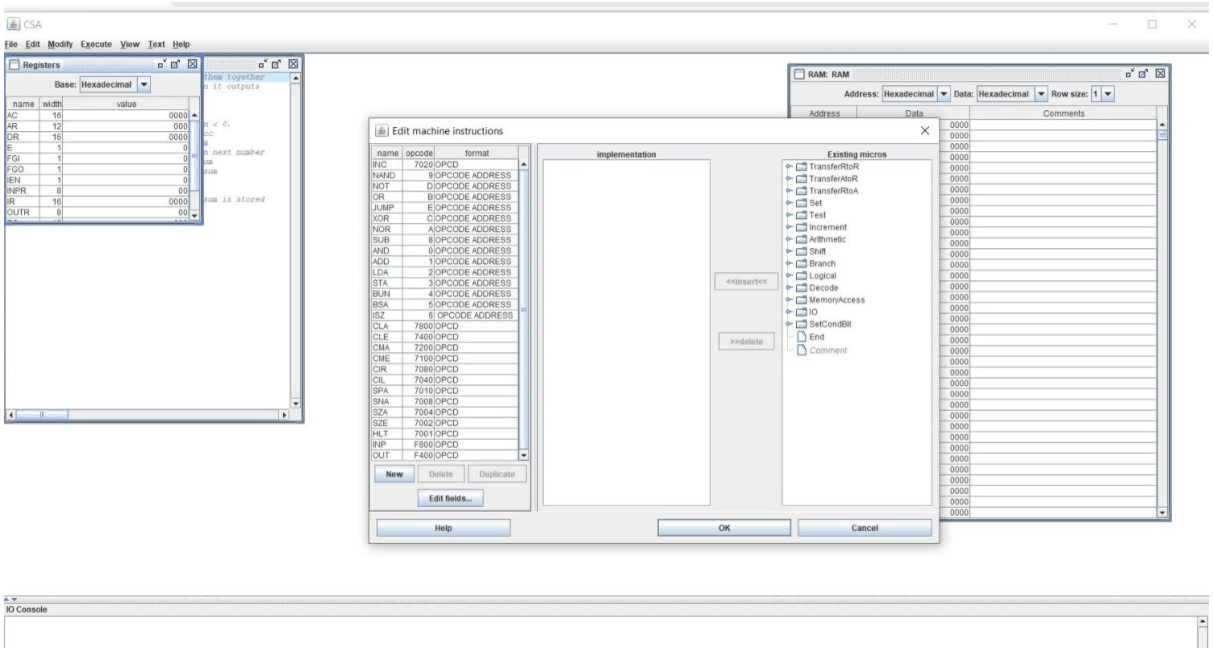

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: SetCondBit
name bit value
E=0 carry out 0
S=0 HALT BIT 1
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console

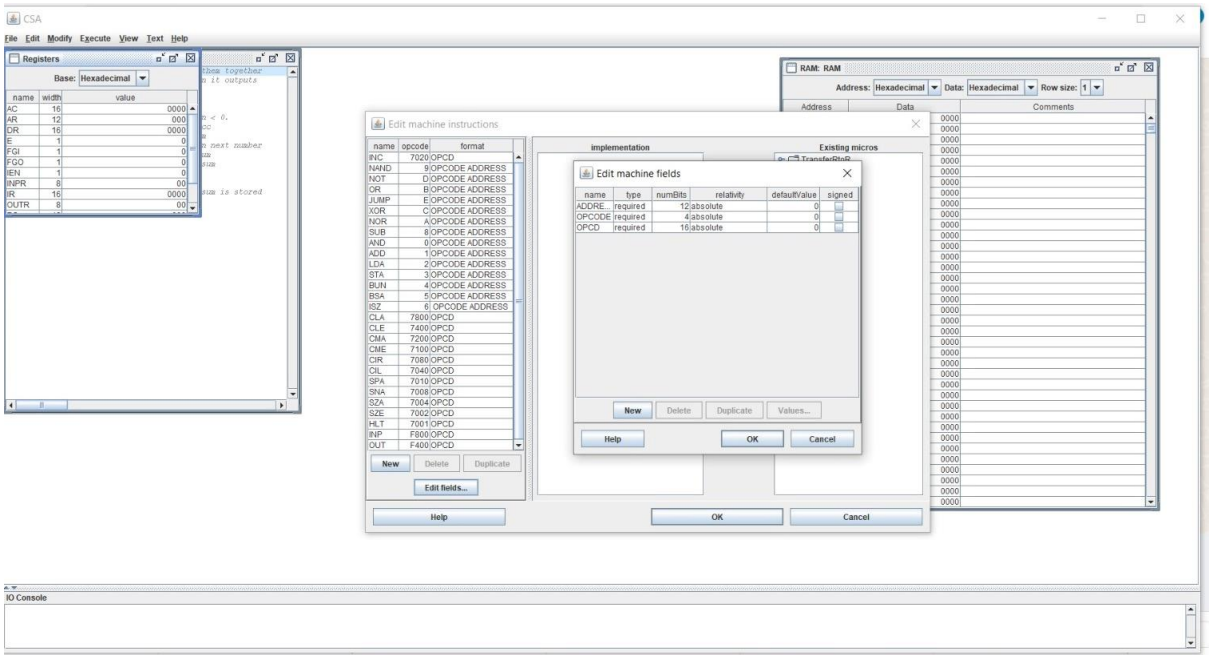

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 000
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 0000
OUTR 8 00
Edit Microinstructions
Type of Microinstruction: IO
name type buffer direction
AC=INPR integer AC input
OUTR=AC integer AC output
New Delete Duplicate
Help OK Cancel
RAM: RAM
Address: Hexadecimal Data: Hexadecimal Row size: 1
Address Data Comments
IO Console




## STEP 4. GO TO MODIFY AND MAKE MACHINE INSTRUCTIONS.

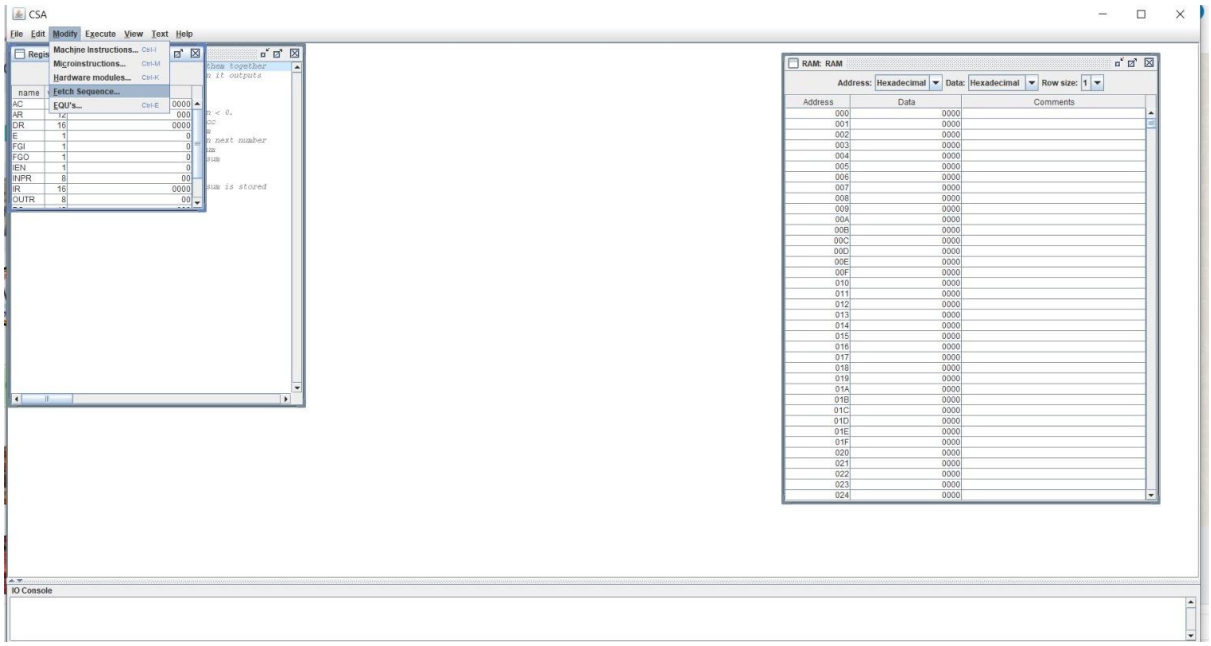


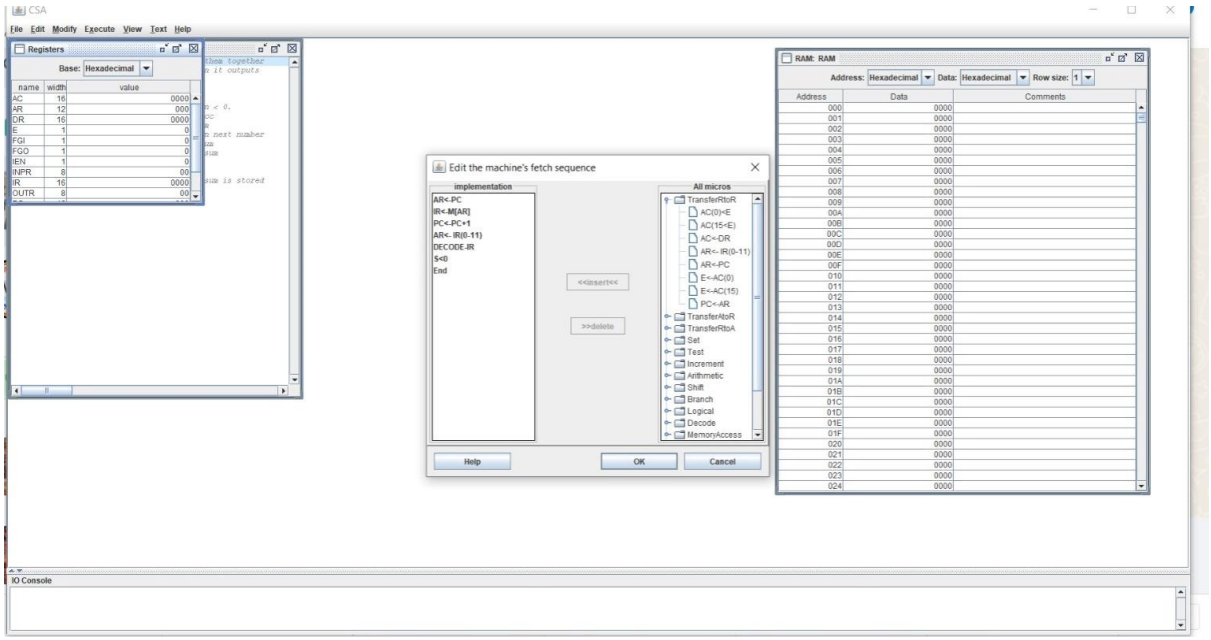


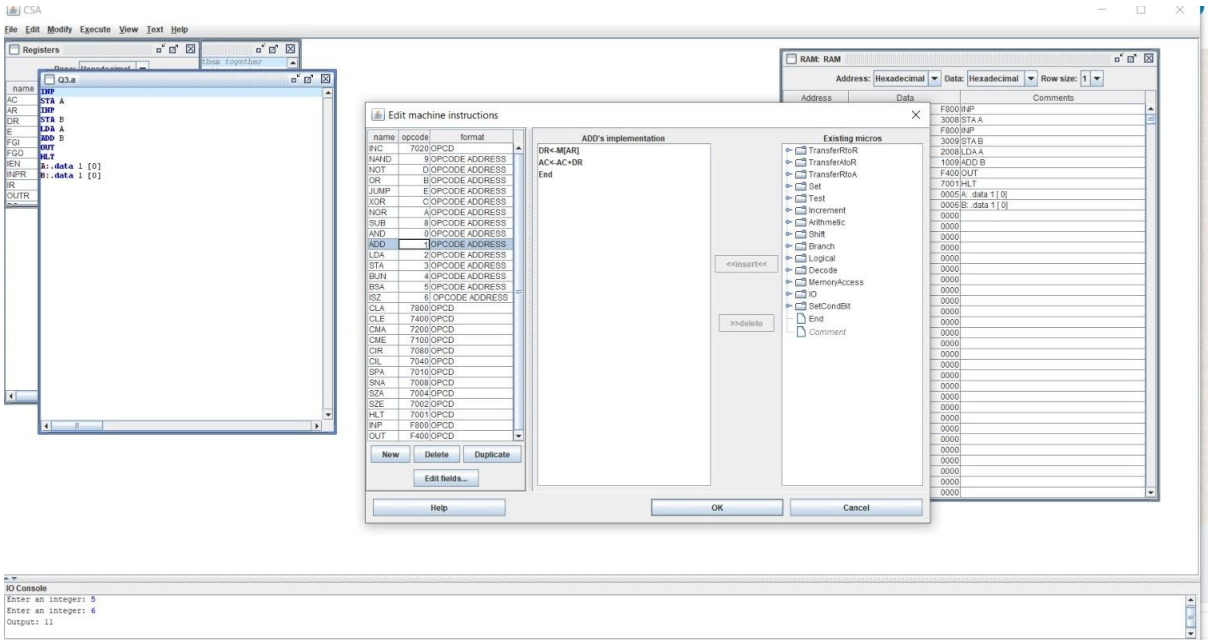

## 2. CREATE A FETCH ROUTINE OF THE INSTRUCTION CYCLE.

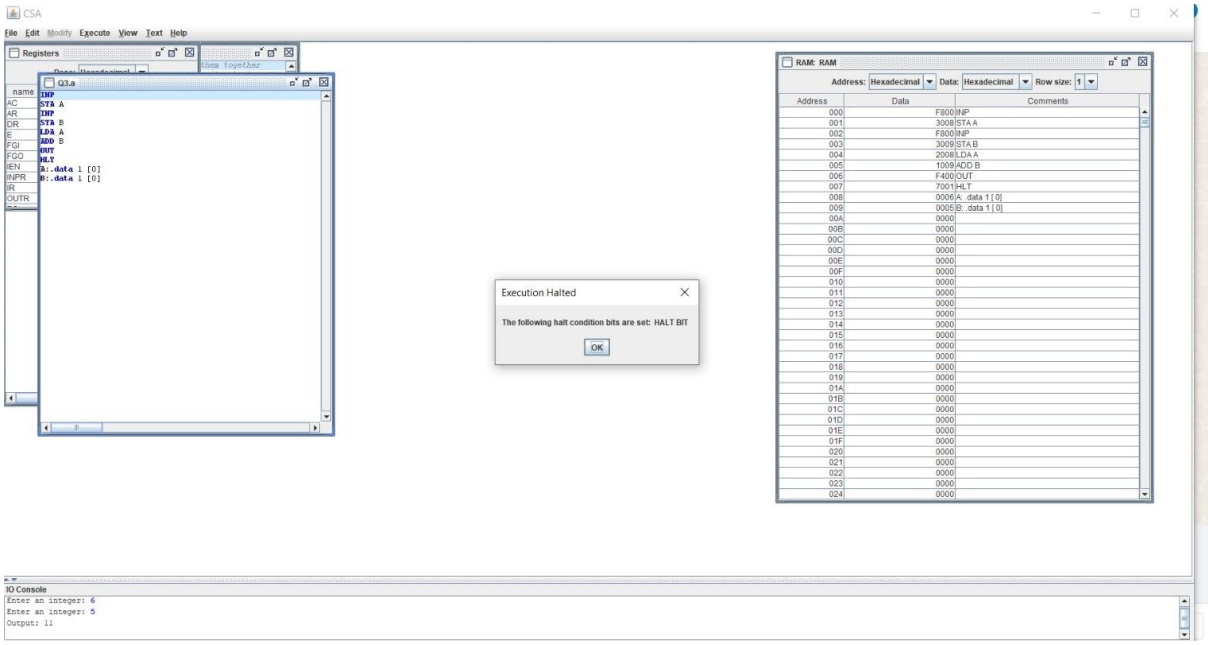
**Step 1. Go to modify and make machine fetch sequence.**



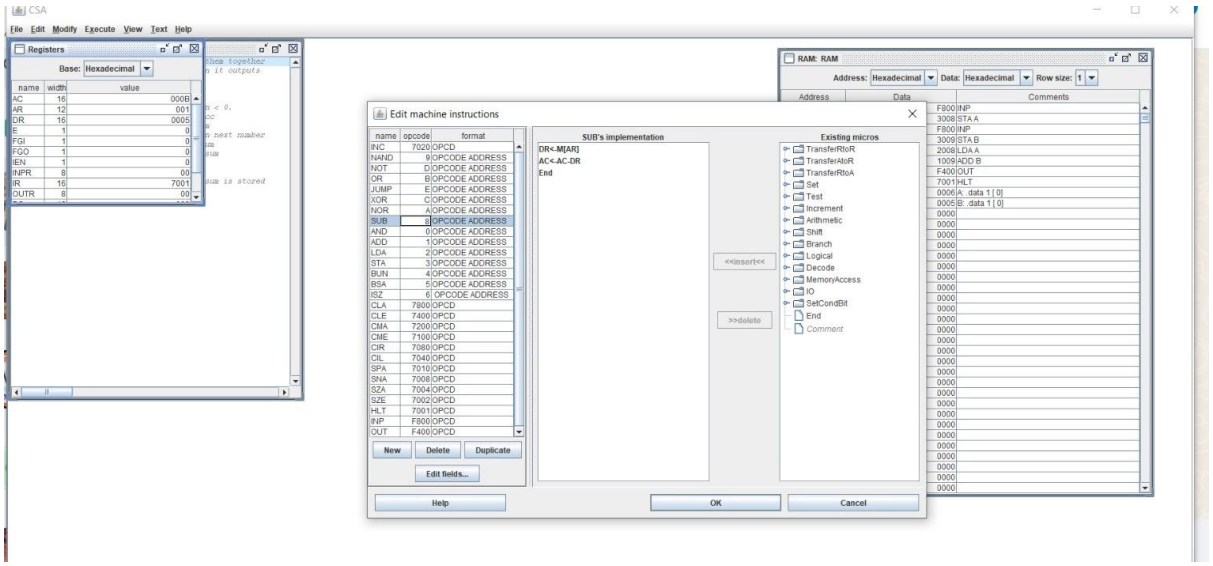



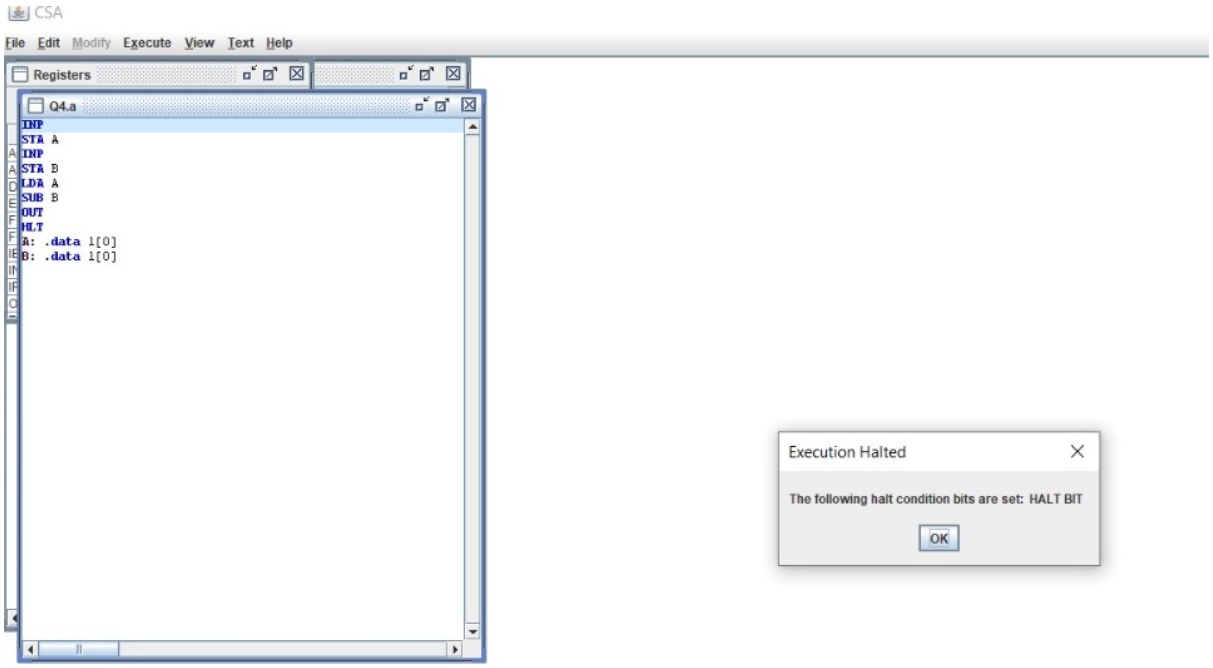

**3. write an assembly program to simulate ADD OPERATION ON TWO USER -ENTERED NUMBERS.**

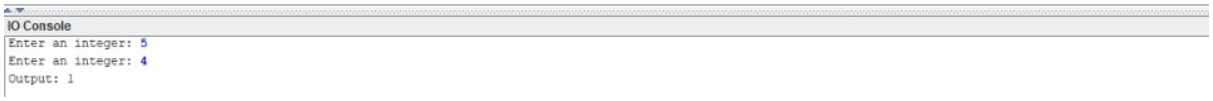



4. WRITE AN ASSEMBLY PROGRAM TO SIMULATE SUBSTRACT OPERATION ON TWO USER-ENTERED NUMBERS.



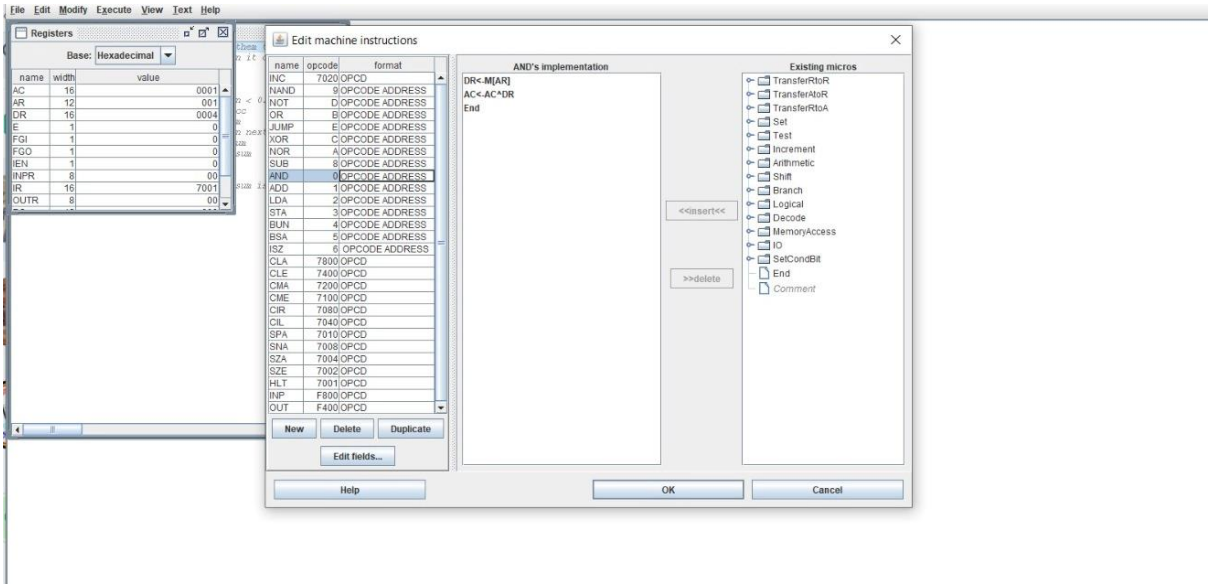



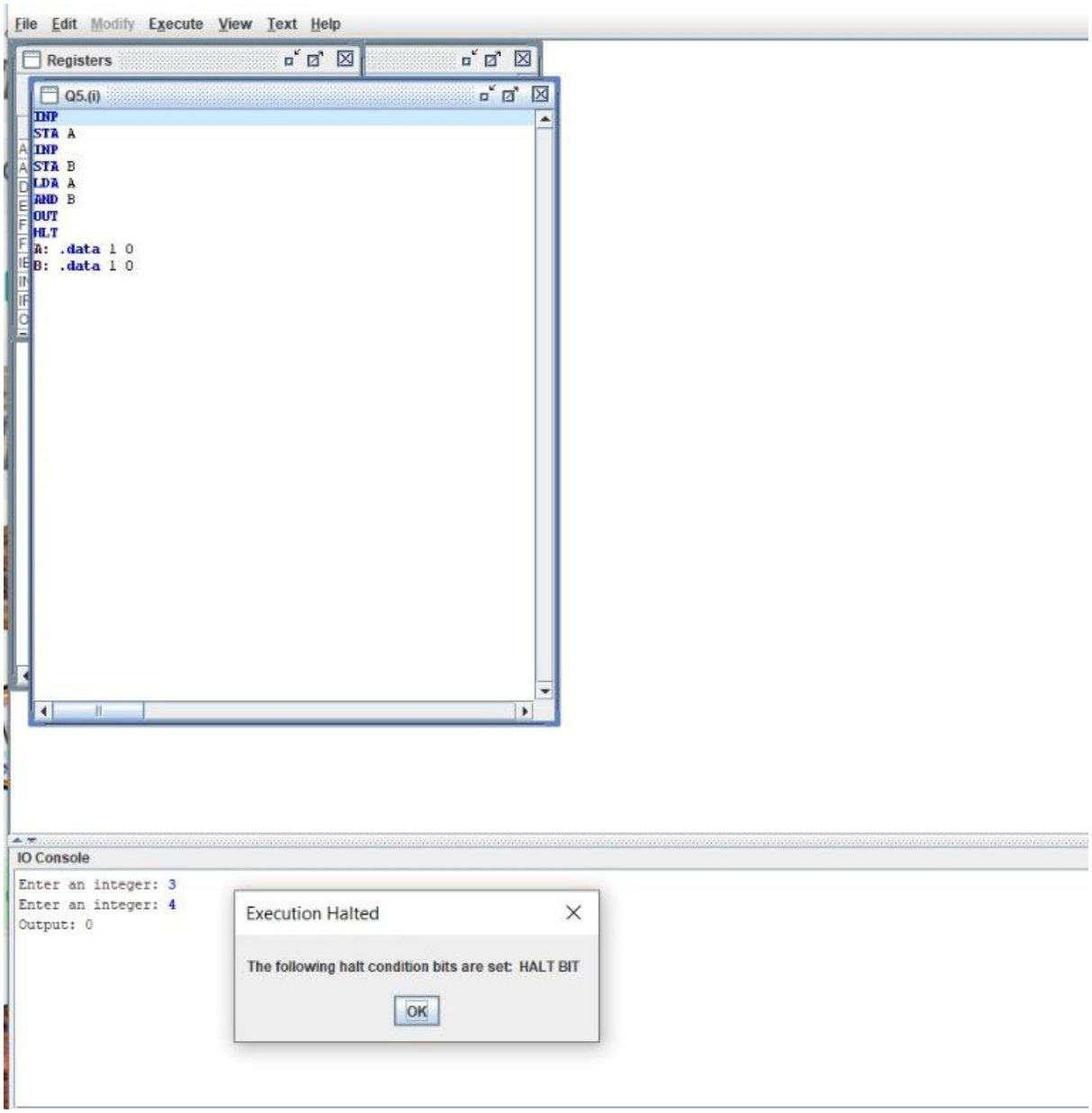




5. WRITE AN ASSEMBLY PROGRAM TO SIMULATE THE FOLLOWING LOGICAL OPERATIONS ON TWO USER ENTERED NUMBERS.

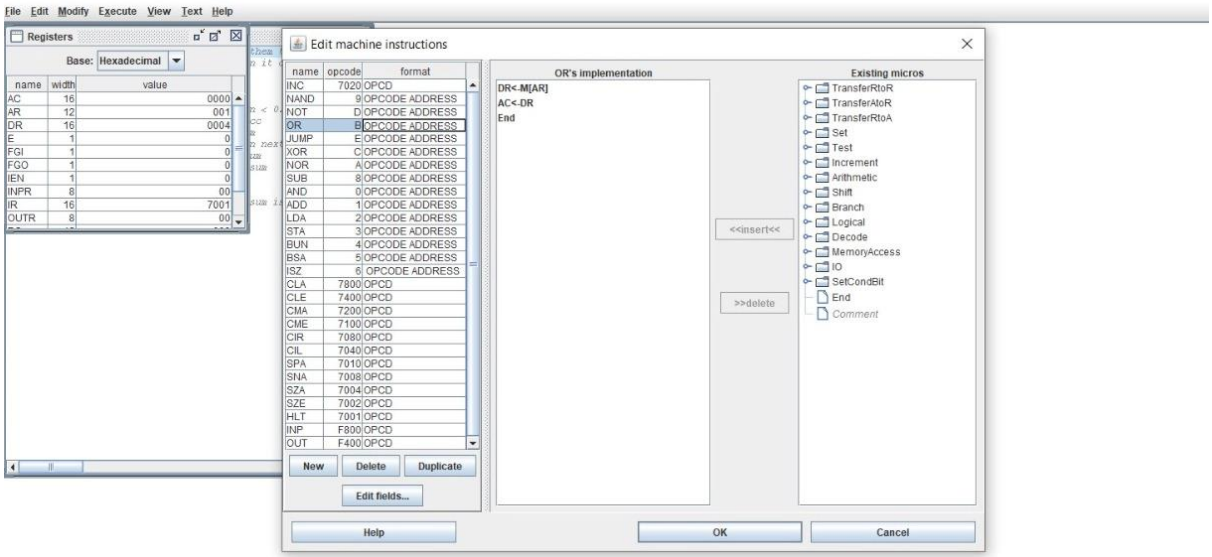
**AND,OR,NOT,XOR,NOR,NAND**

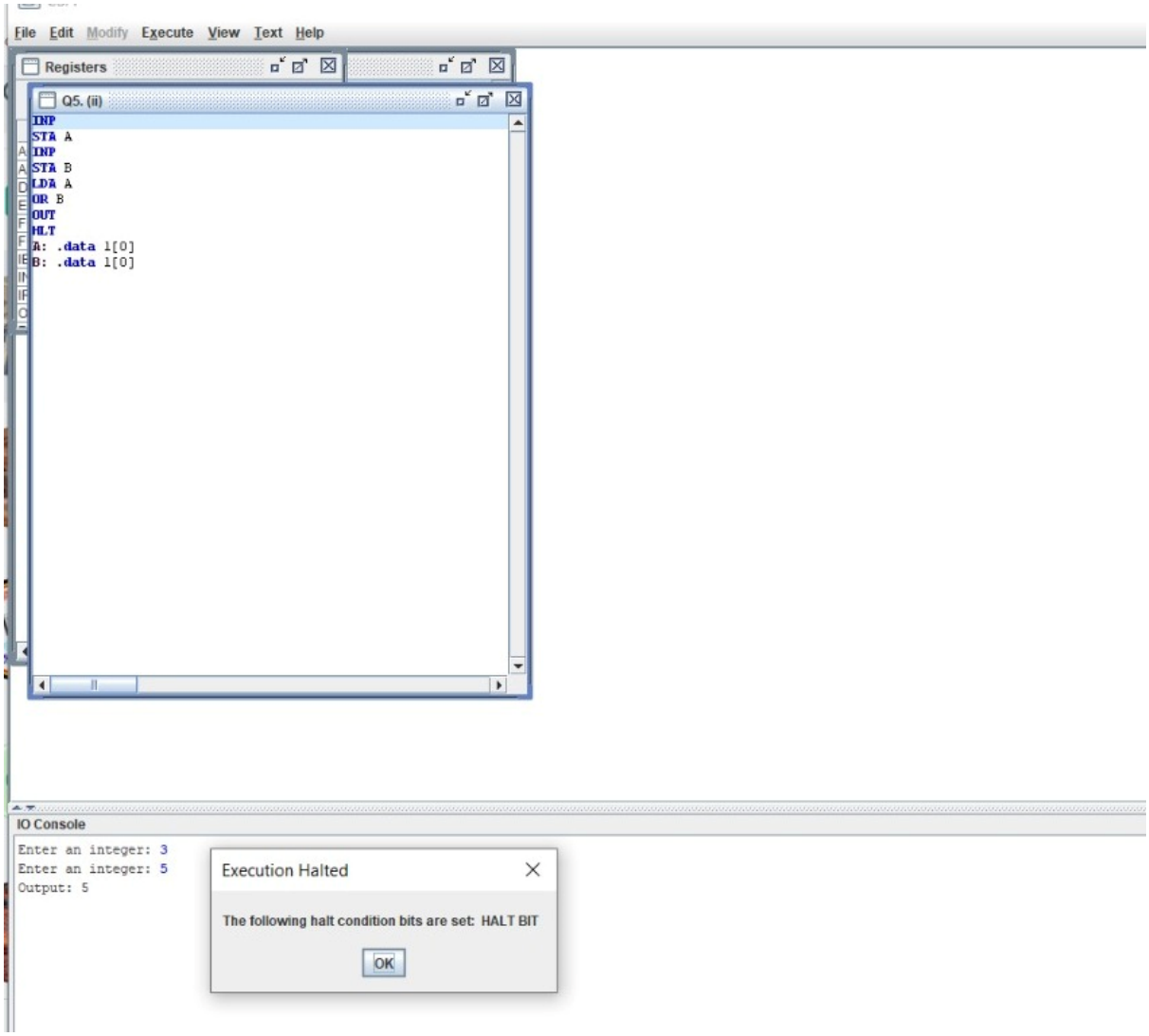
## 1. AND


File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0001
AR 12 001
DR 16 0004
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
AND's implementation
DR<-M[AR]
AC<-AC^DR
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel

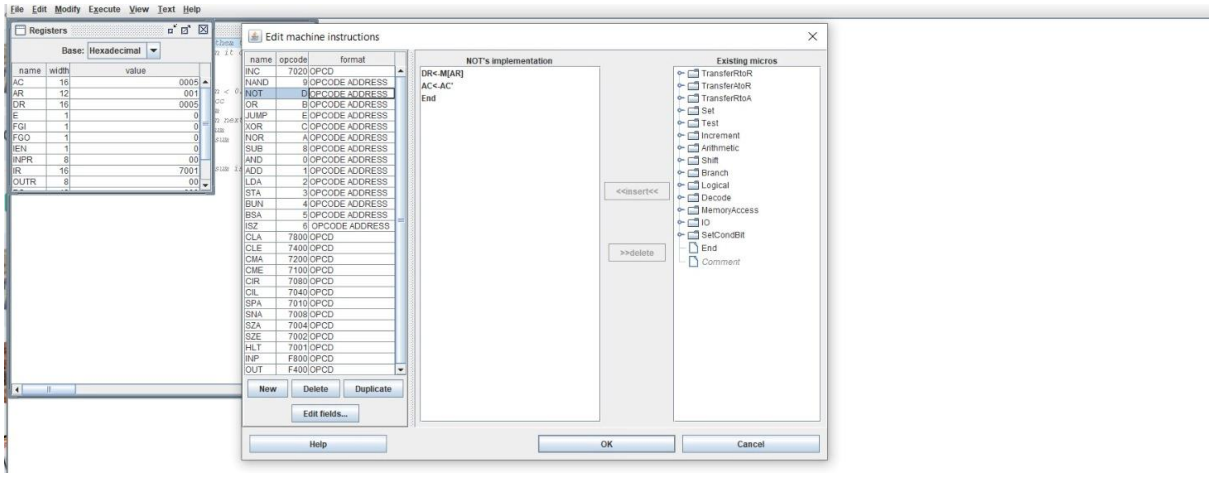

File Edit Modify Execute View Text Help
Registers
Q5.(i)
INP
STA A
INP
STA B
LDA A
AND B
OUT
HLT
A: .data 1 0
B: .data 1 0
IO Console
Enter an integer: 3
Enter an integer: 4
Output: 0
Execution Halted
The following halt condition bits are set: HALT BIT
OK

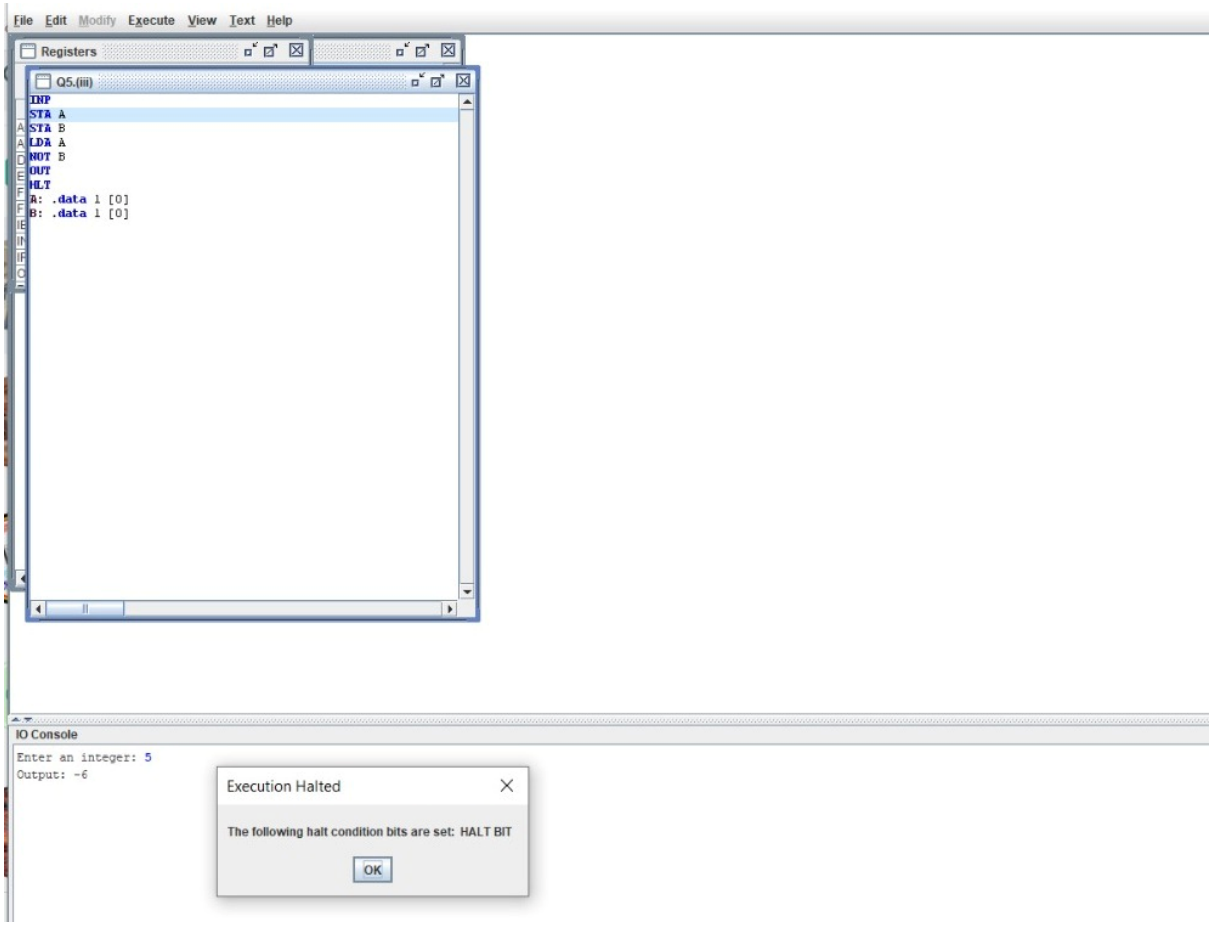
**2.OR**


File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 001
DR 16 0004
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
OR's implementation
DR<-M[AR]
AC<-DR
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel

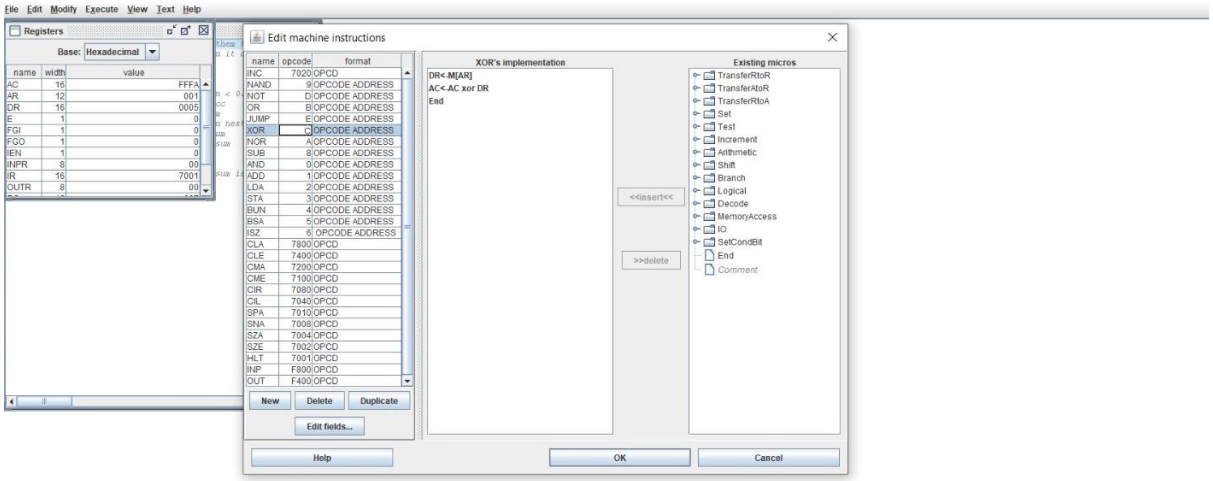

File Edit Modify Execute View Text Help
Registers
Q5. (ii)
INP
STA A
INP
STA B
LDA A
OR B
OUT
HLT
A: .data 1[0]
B: .data 1[0]
IO Console
Enter an integer: 3
Enter an integer: 5
Output: 5
Execution Halted
The following halt condition bits are set: HALT BIT
OK

**3.NOT**

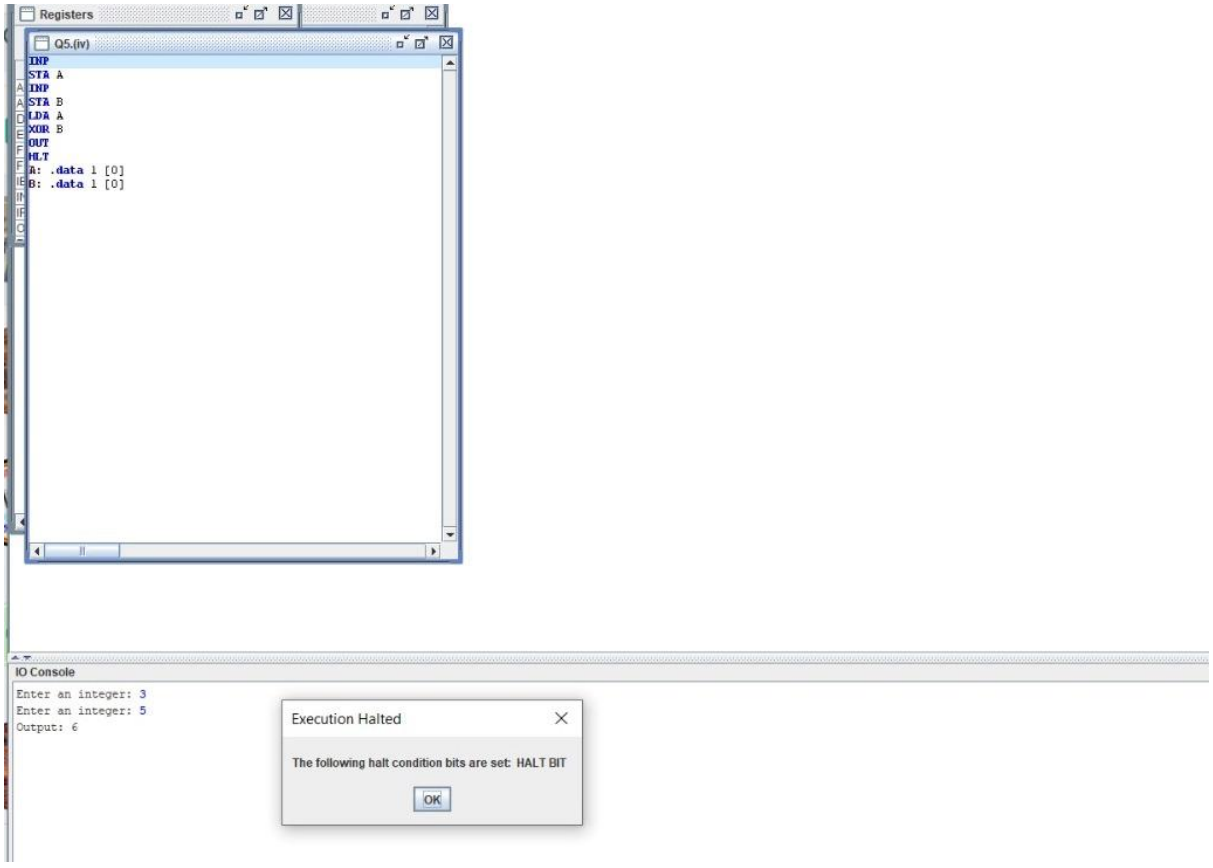

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0005
AR 12 001
DR 16 0005
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT B OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
NOT's implementation
DR<-M[AR]
AC<-AC'
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


File Edit Modify Execute View Text Help
Registers
Q5.(iii)
INP
STA A
STA B
LDA A
NOT B
OUT
HLT
A: .data 1 [0]
B: .data 1 [0]
IO Console
Enter an integer: 5
Output: -6
Execution Halted
The following halt condition bits are set: HALT BIT
OK

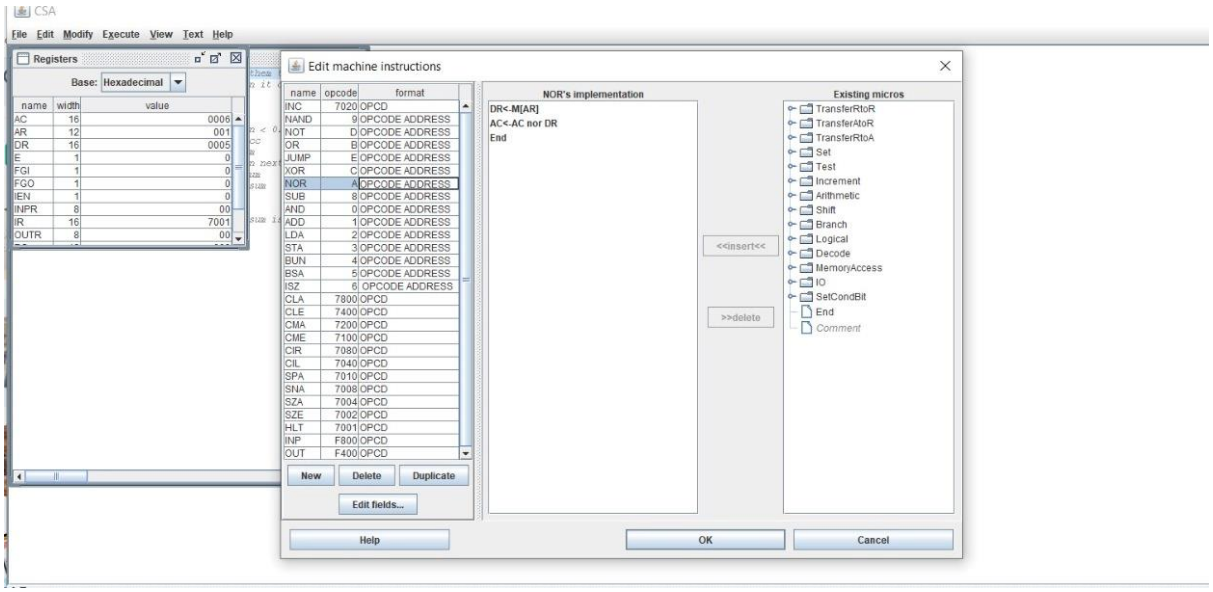
## 4.XOR

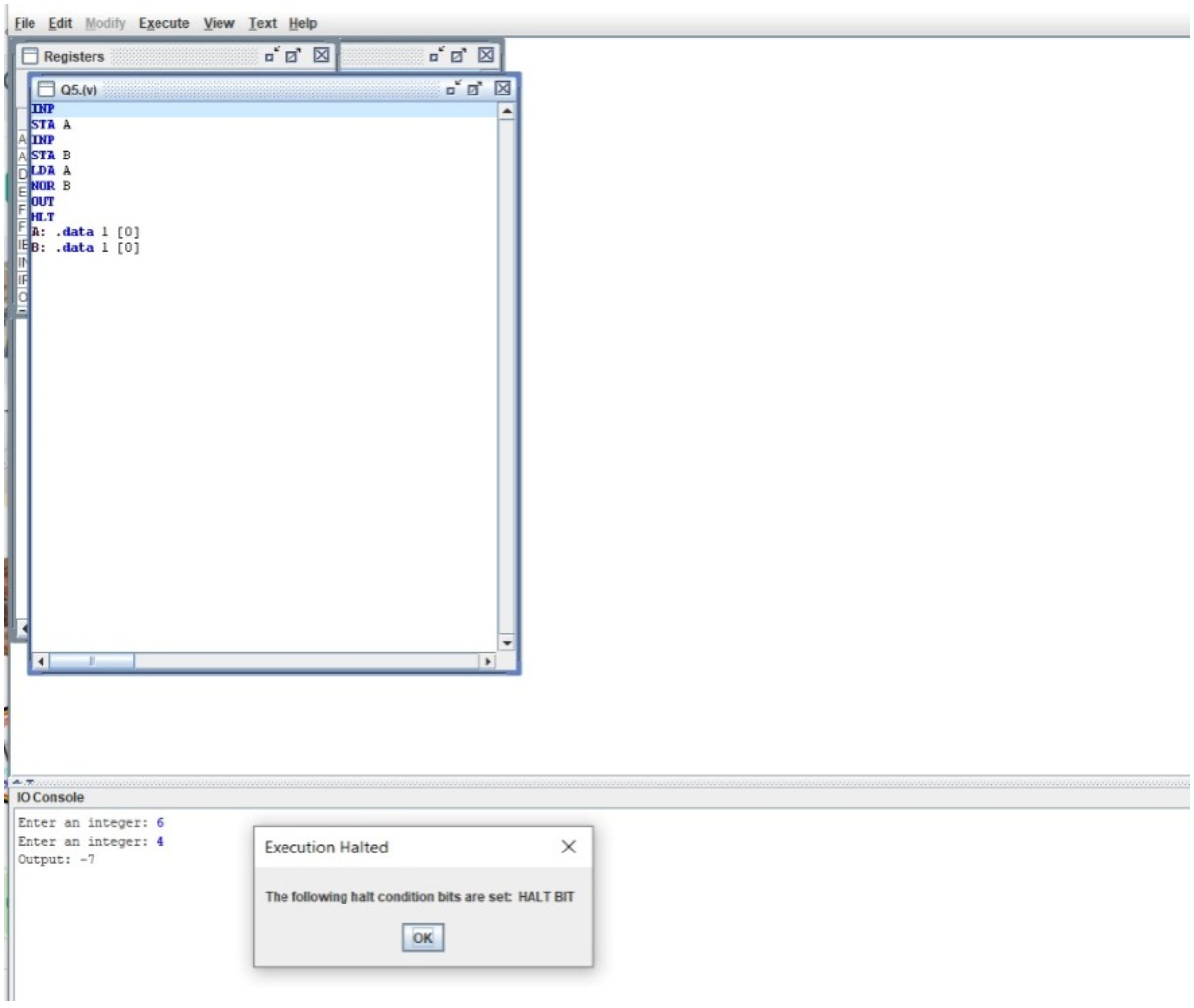

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 FFFA
AR 12 001
DR 16 0005
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New Delete Duplicate
Edit fields...
XOR's implementation
DR<-M[AR]
AC<-AC xor DR
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


IO Console
Enter an integer: 5
Output: -6

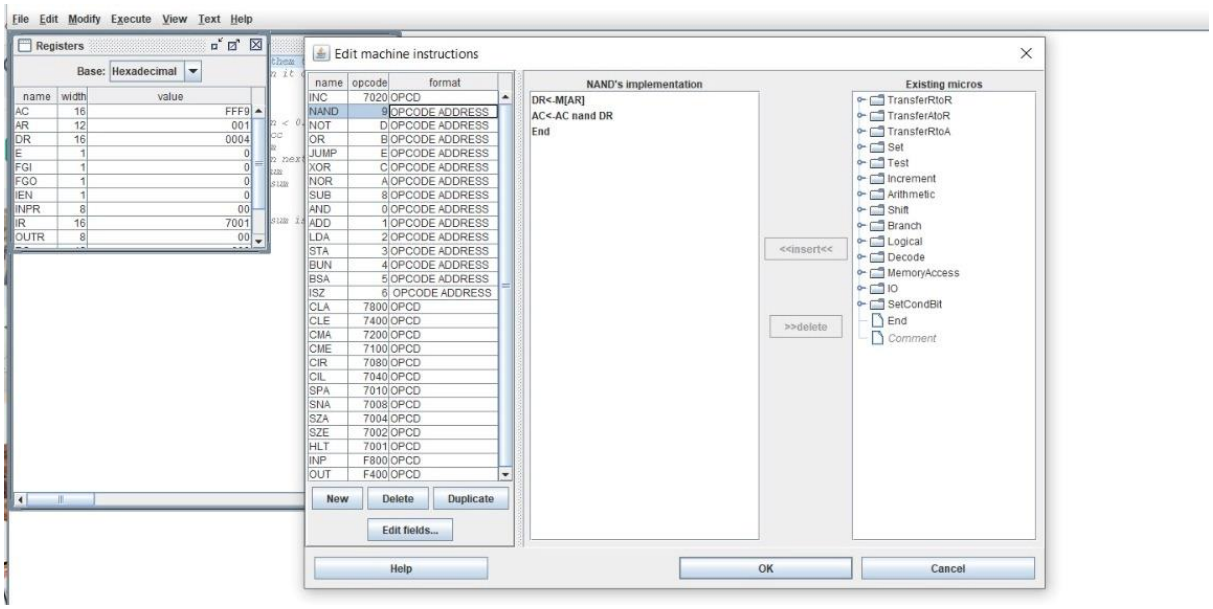

Registers
Q5.(iv)
INP
STA A
INP
STA B
LDA A
XOR B
OUT
HLT
A: .data 1 [0]
B: .data 1 [0]
IO Console
Enter an integer: 3
Enter an integer: 5
Output: 6
Execution Halted
The following halt condition bits are set: HALT BIT
OK

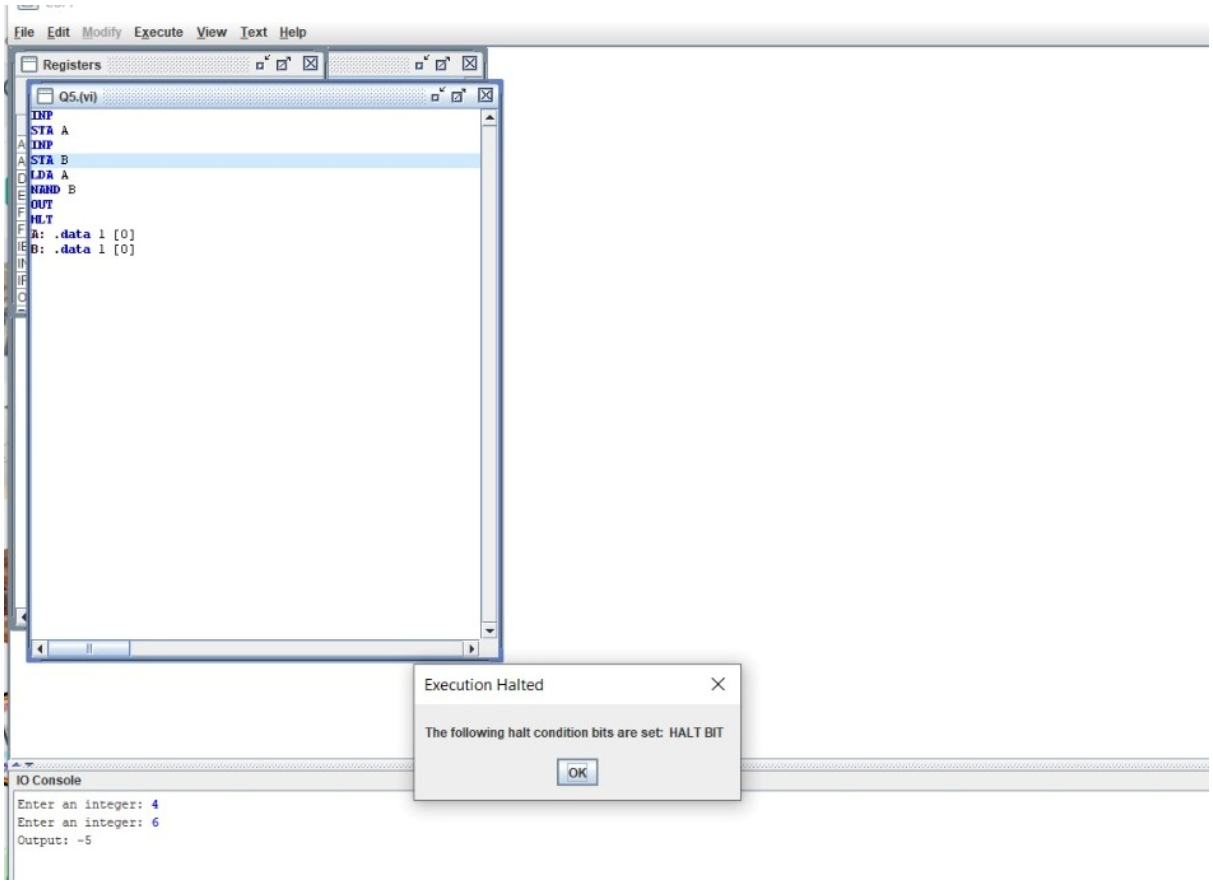
**5.NOR**


CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0006
AR 12 001
DR 16 0005
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New Delete Duplicate
Edit fields...
Help
NOR's implementation
DR<-M[AR]
AC<-AC nor DR
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
OK
Cancel


File Edit Modify Execute View Text Help
Registers
Q5.(v)
INP
STA A
INP
STA B
LDA A
NOR B
OUT
HLT
A: .data 1 [0]
B: .data 1 [0]
IO Console
Enter an integer: 6
Enter an integer: 4
Output: -7
Execution Halted
The following halt condition bits are set: HALT BIT
OK

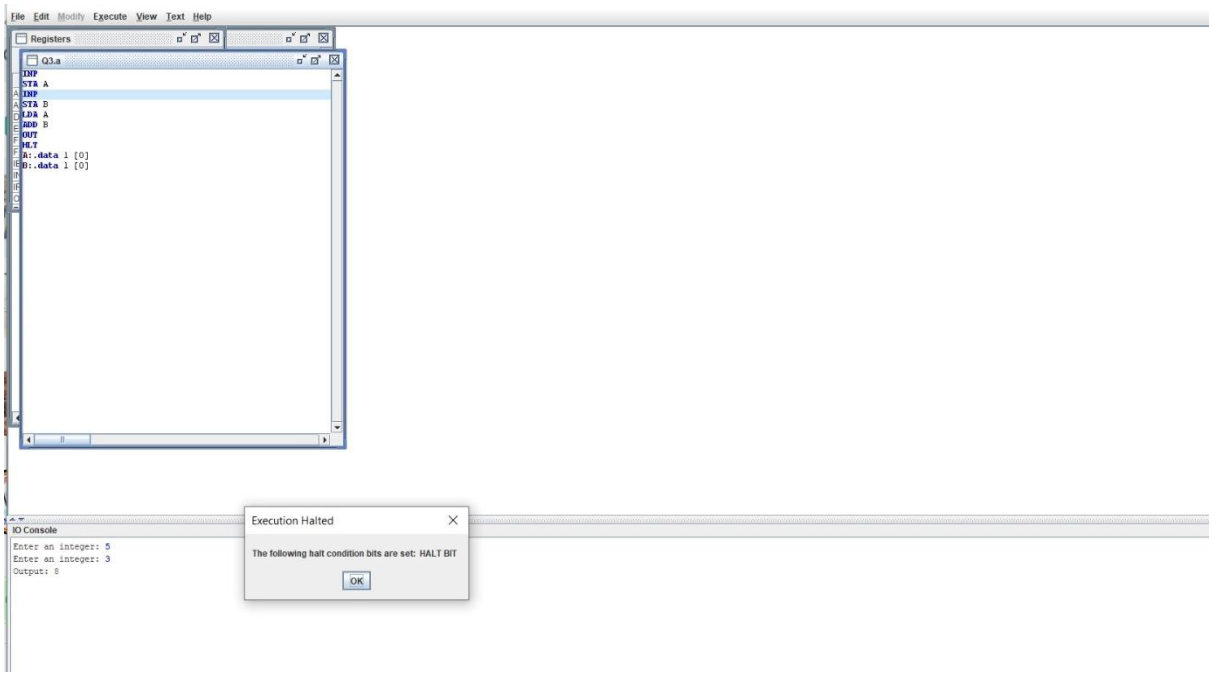
**6.NAND**



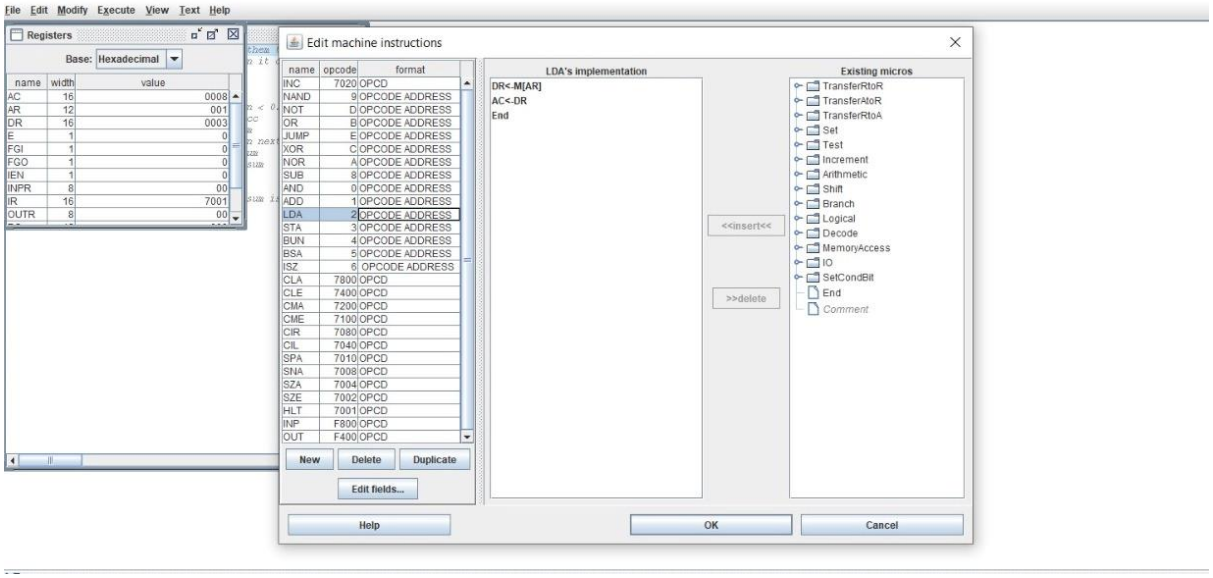



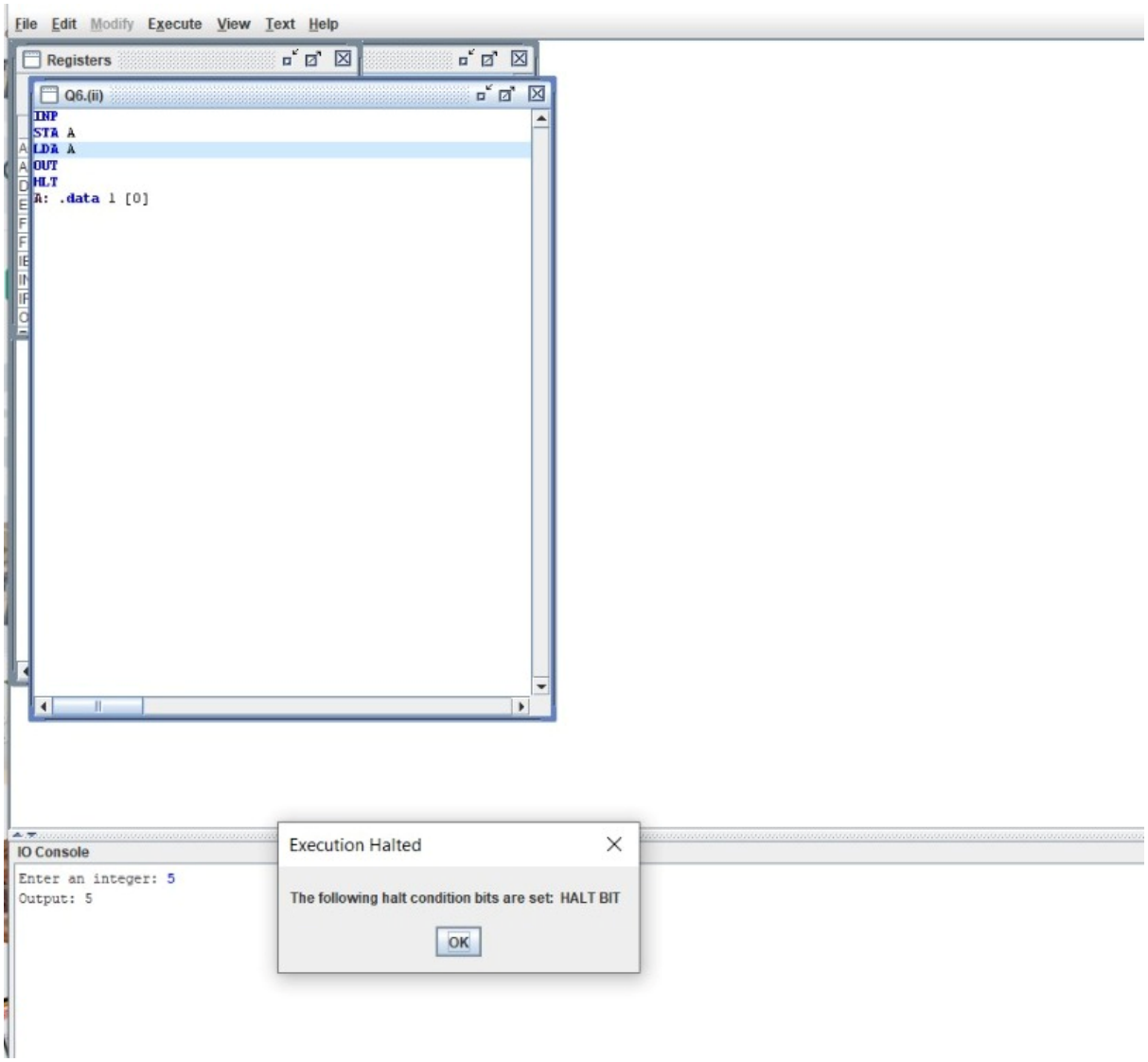

**6.** WRITE AN ASSEMBLY PROGRAM FOR SIMULATING FOLLOWING MEMORY REFERENCE INSTRUCTIONS.

1.**ADD**

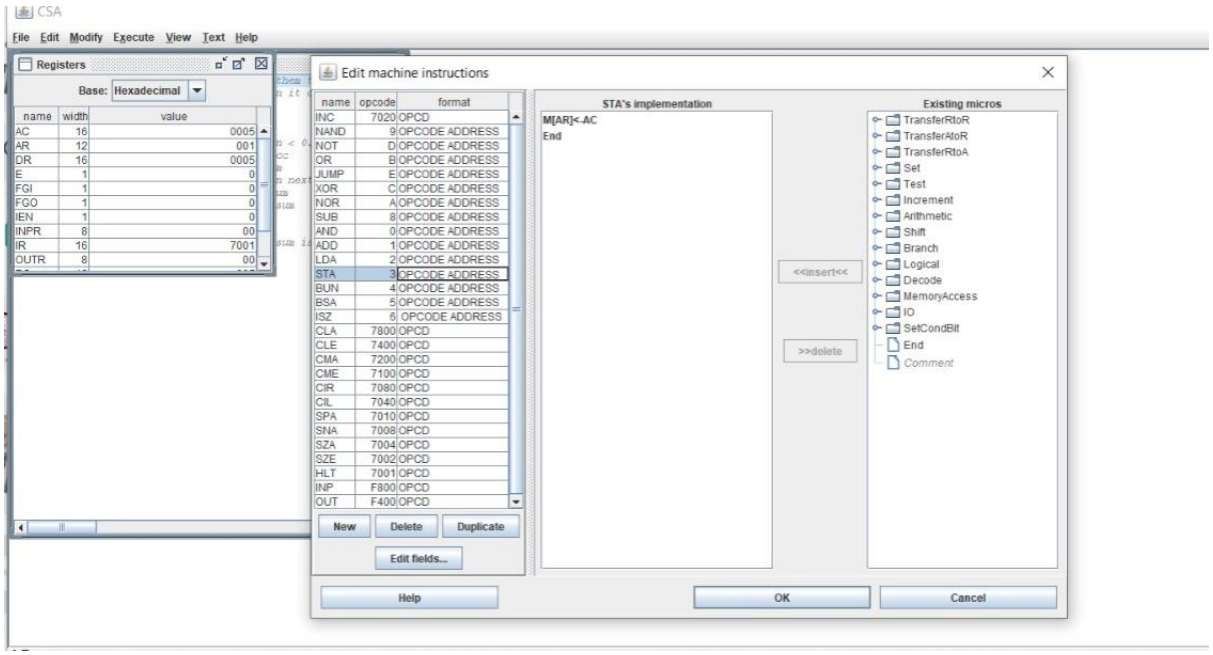

File Edit Modify Execute View Text Help
Registers
Q3.a
INP
STA A
INP
STA B
LDA A
ADD B
OUT
HLT
A:.data 1 [0]
B:.data 1 [0]
IO Console
Enter an integer: 5
Enter an integer: 3
Output: 8
Execution Halted
The following halt condition bits are set: HALT BIT
OK

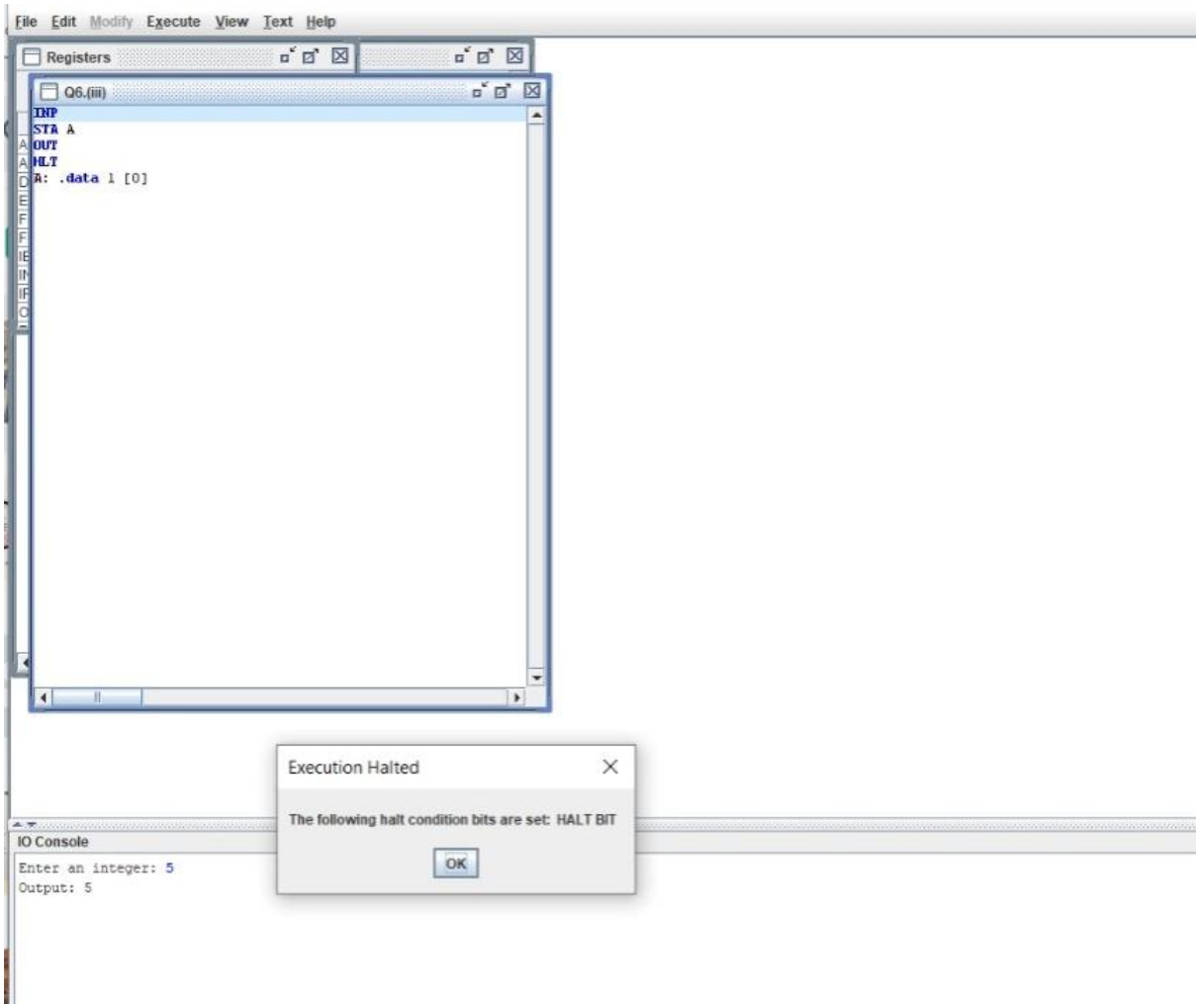
## 2.LDA


File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0008
AR 12 001
DR 16 0003
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
LDA's implementation
DR<-M[AR]
AC<-DR
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel

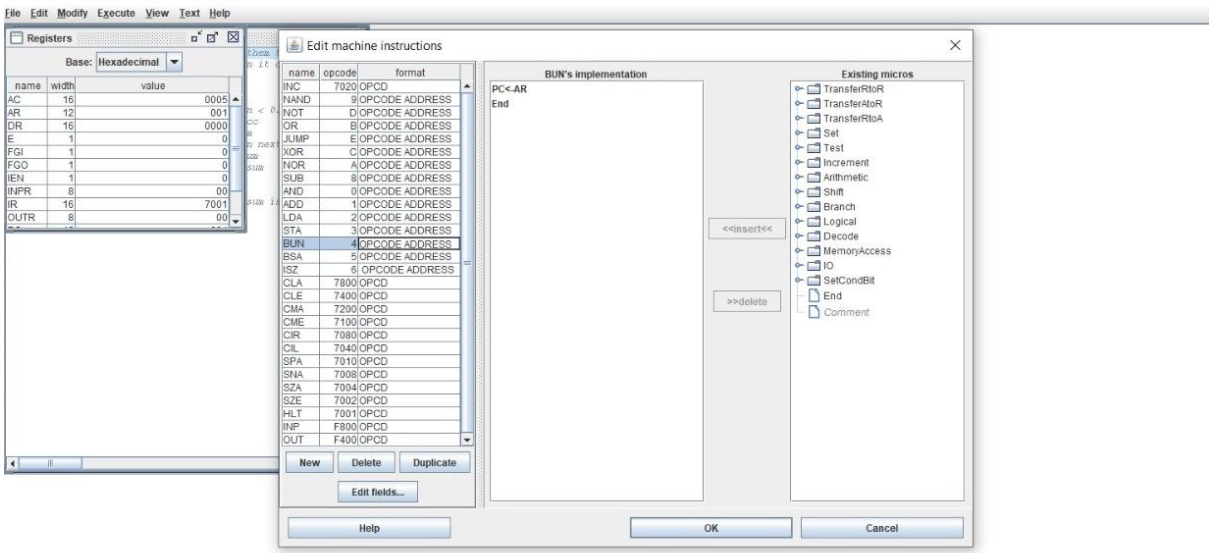

File Edit Modify Execute View Text Help
Registers
Q6.(ii)
INP
STA A
LDA A
OUT
HLT
A: .data 1 [0]
IO Console
Enter an integer: 5
Output: 5
Execution Halted
The following halt condition bits are set: HALT BIT
OK

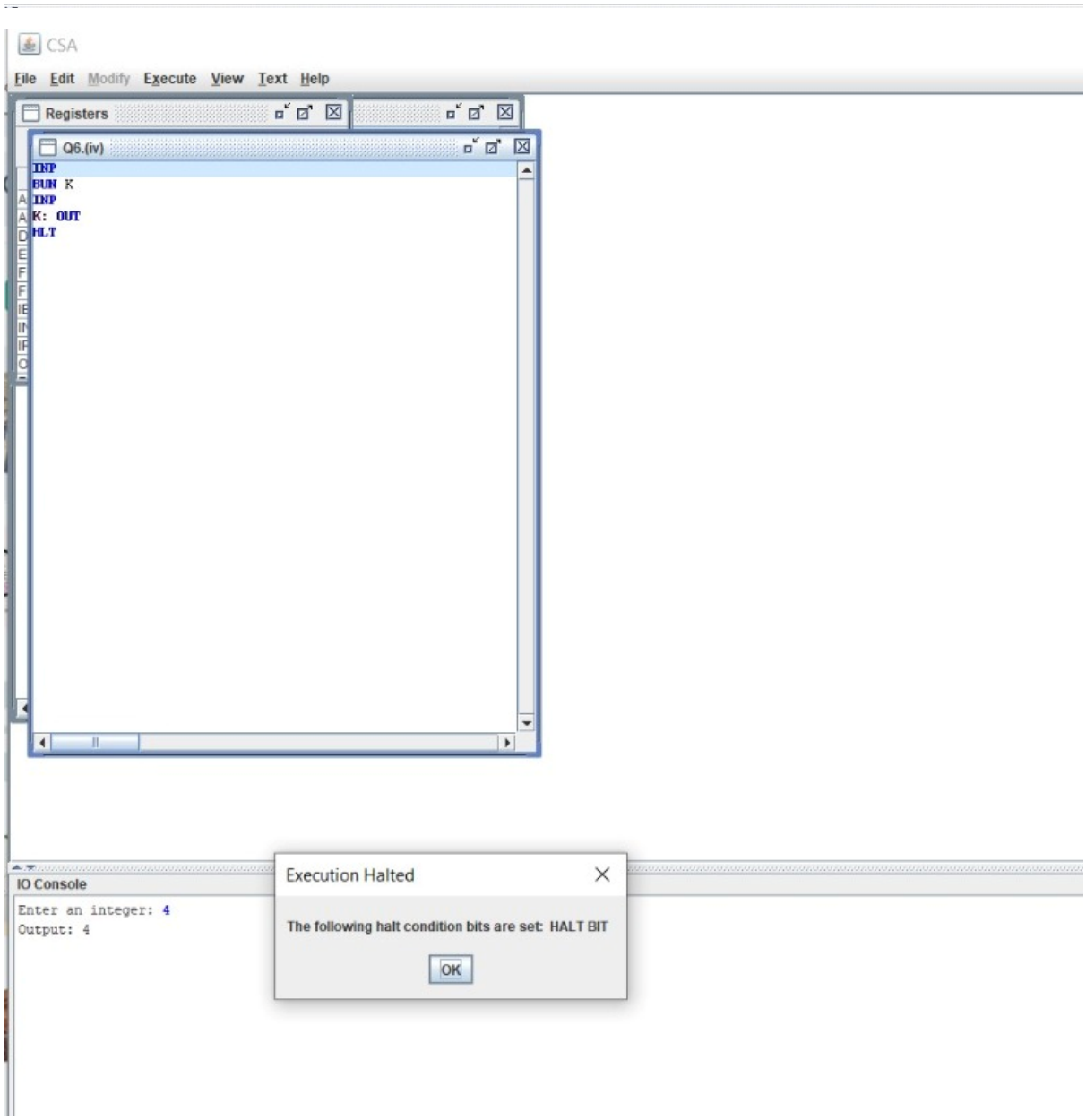
**3.STA**


CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0005
AR 12 001
DR 16 0005
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
STA's implementation
M[AR]<-AC
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel

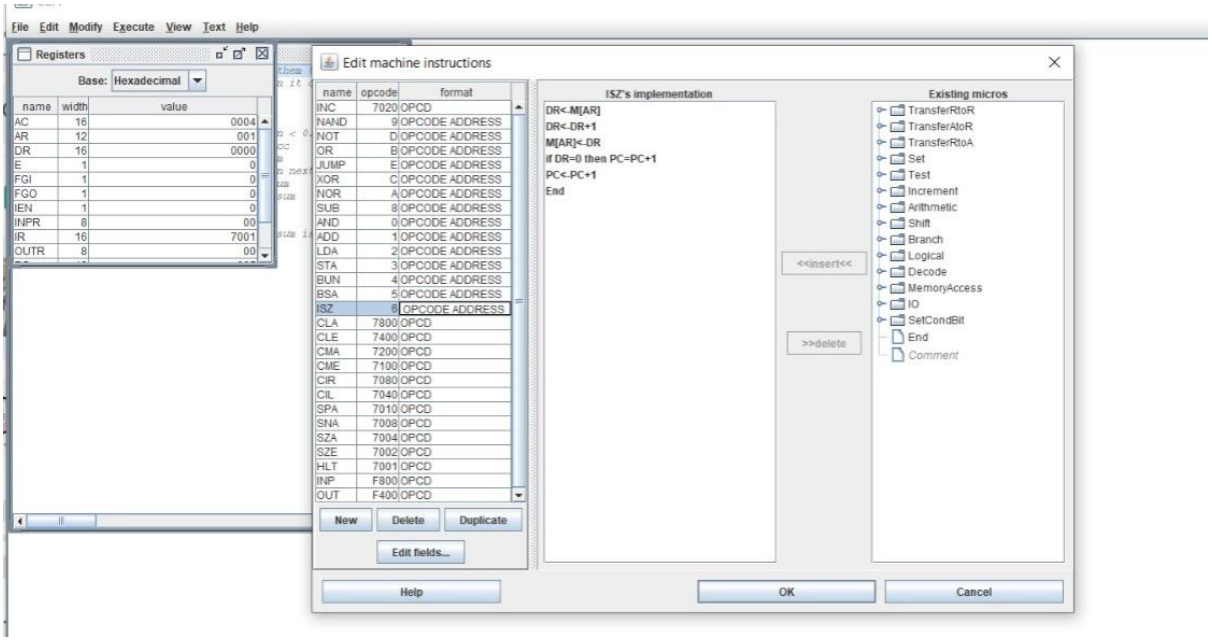

File Edit Modify Execute View Text Help
Registers
Q6.(iii)
INP
STA A
OUT
HLT
A: .data 1 [0]
Execution Halted
The following halt condition bits are set: HALT BIT
OK
IO Console
Enter an integer: 5
Output: 5

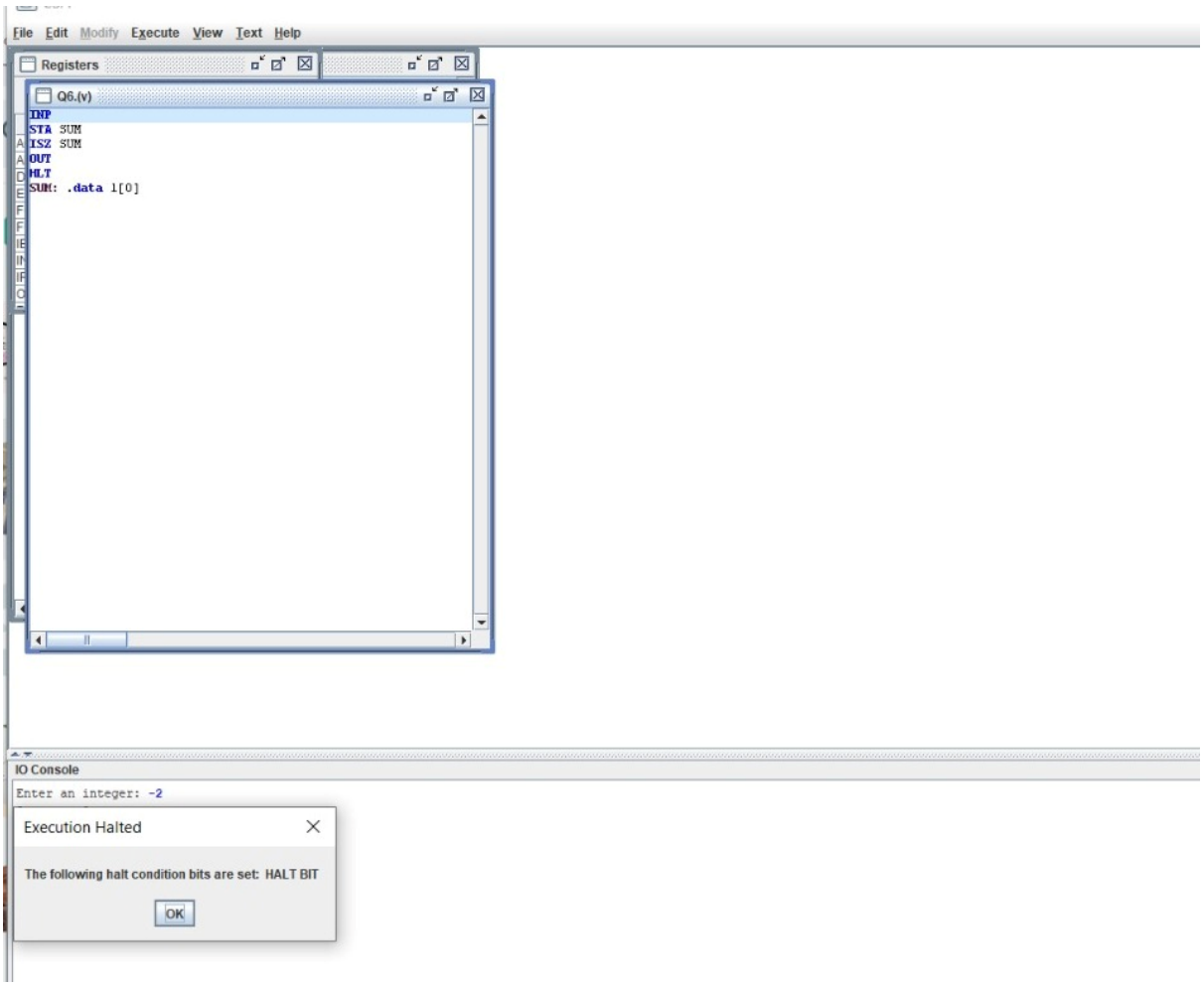
**4.BUN**


File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0005
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
BUN's implementation
PC<-AR
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


CSA
File Edit Modify Execute View Text Help
Registers
Q6.(iv)
INP
BUN K
INP
K: OUT
HLT
IO Console
Enter an integer: 4
Output: 4
Execution Halted
The following halt condition bits are set: HALT BIT
OK

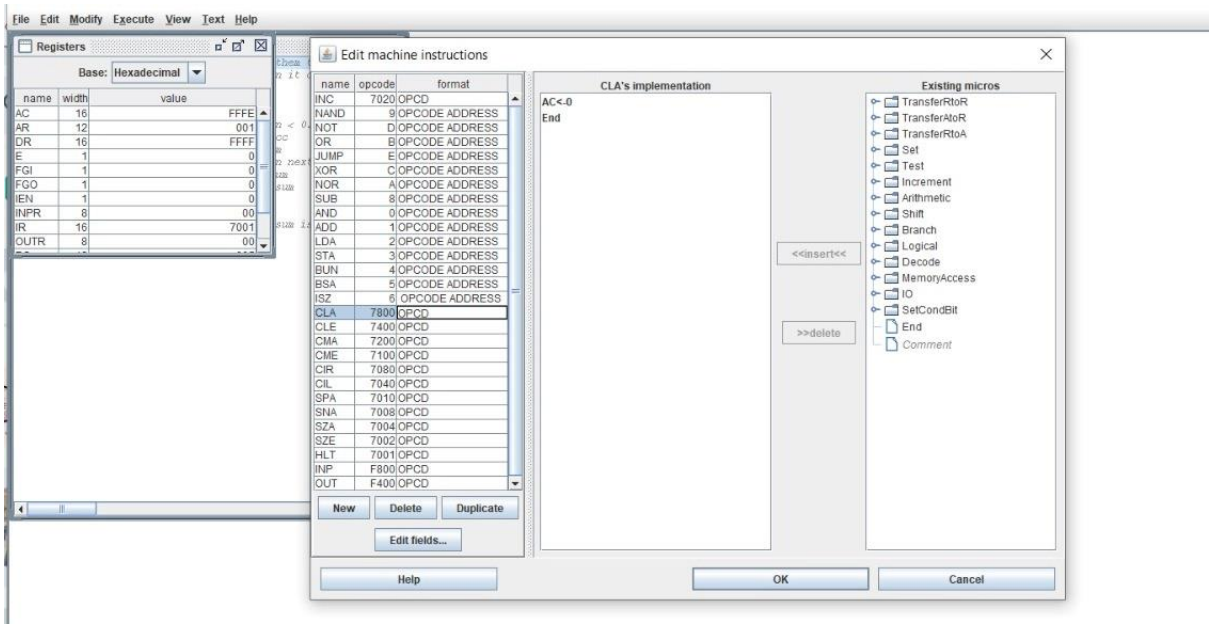
## 5.ISZ

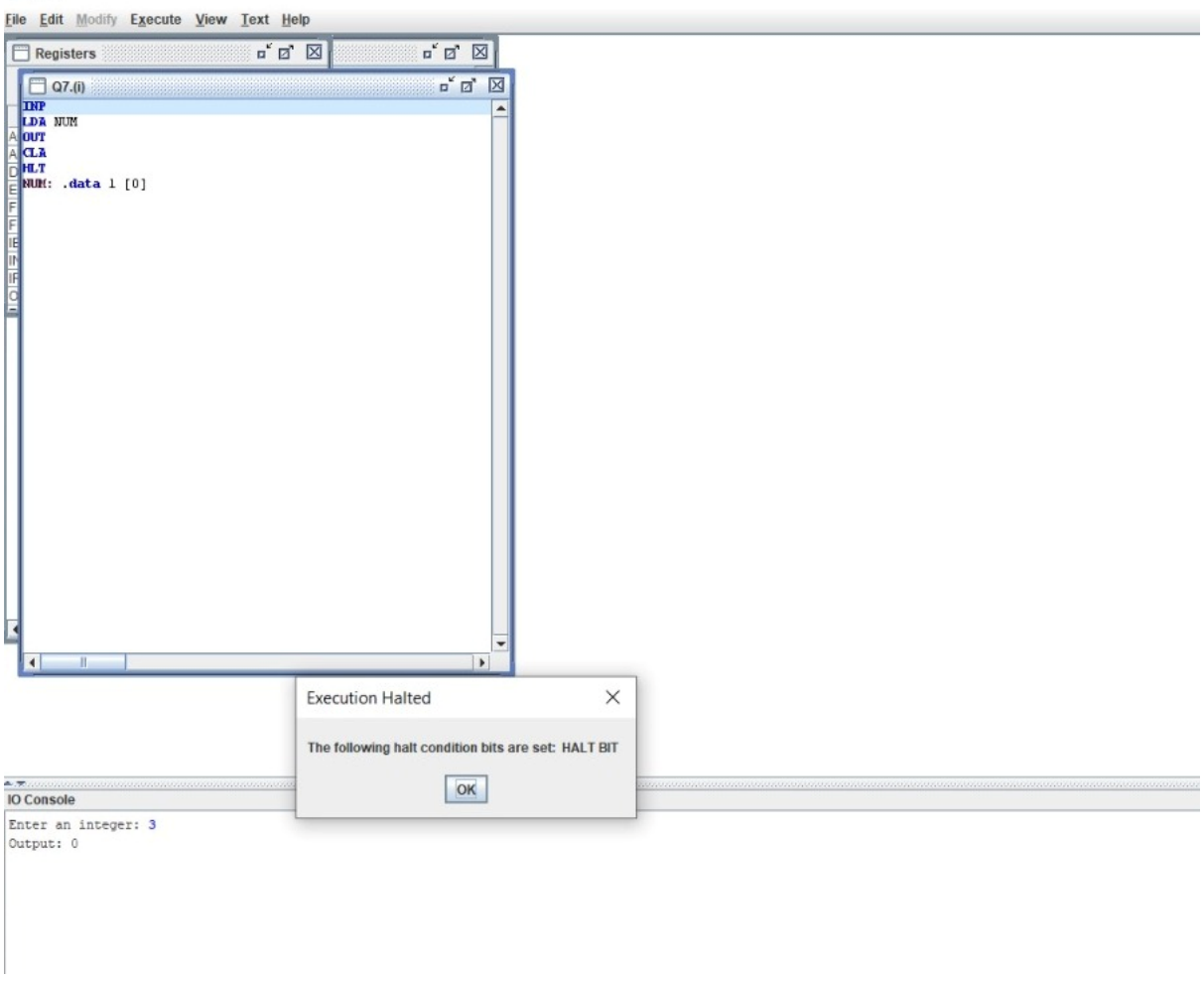

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0004
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
ISZ's implementation
DR<-M[AR]
DR<-DR+1
M[AR]<-DR
if DR=0 then PC=PC+1
PC<-PC+1
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


File Edit Modify Execute View Text Help
Registers
Q6.(v)
INP
STA SUM
ISZ SUM
OUT
HLT
SUM: .data 1[0]
IO Console
Enter an integer: -2
Execution Halted
The following halt condition bits are set: HALT BIT
OK

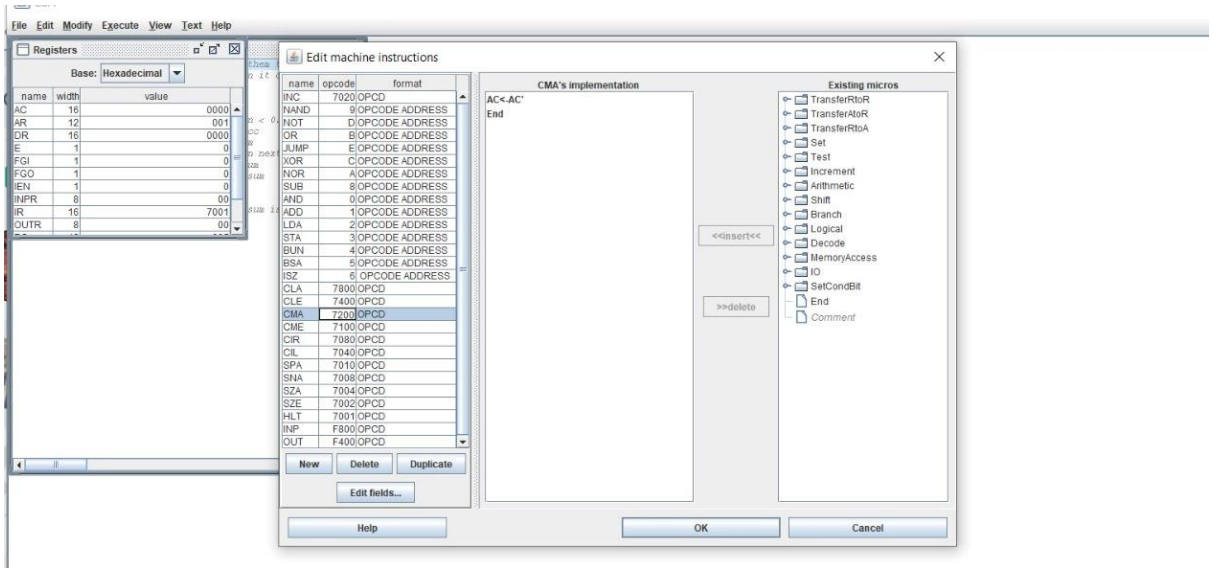
**7. WRITE AN ASSEMBLY LANGUAGE PROGRAM TO SIMULATE THE MACHINE FOR FOLLOWING REGISTER REFERENCE INSTRUCTIONS AND DETERMINE THE CONTENTS OF AC,E,PC,AR AND IR REGISTERS IN DECIMAL AFTER THE EXECUTION.**

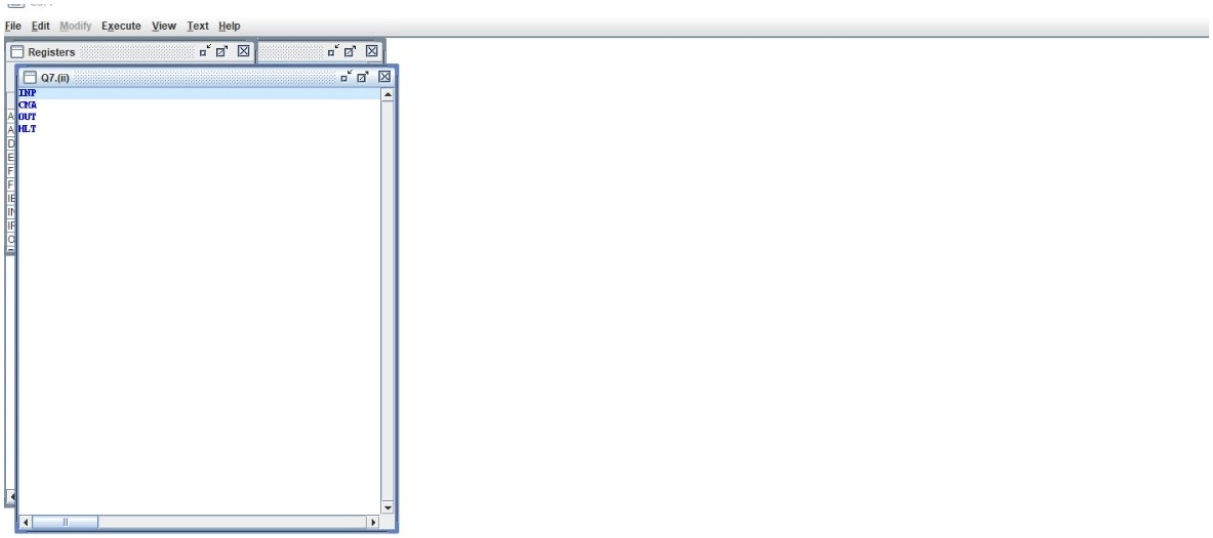
**1.CLA**



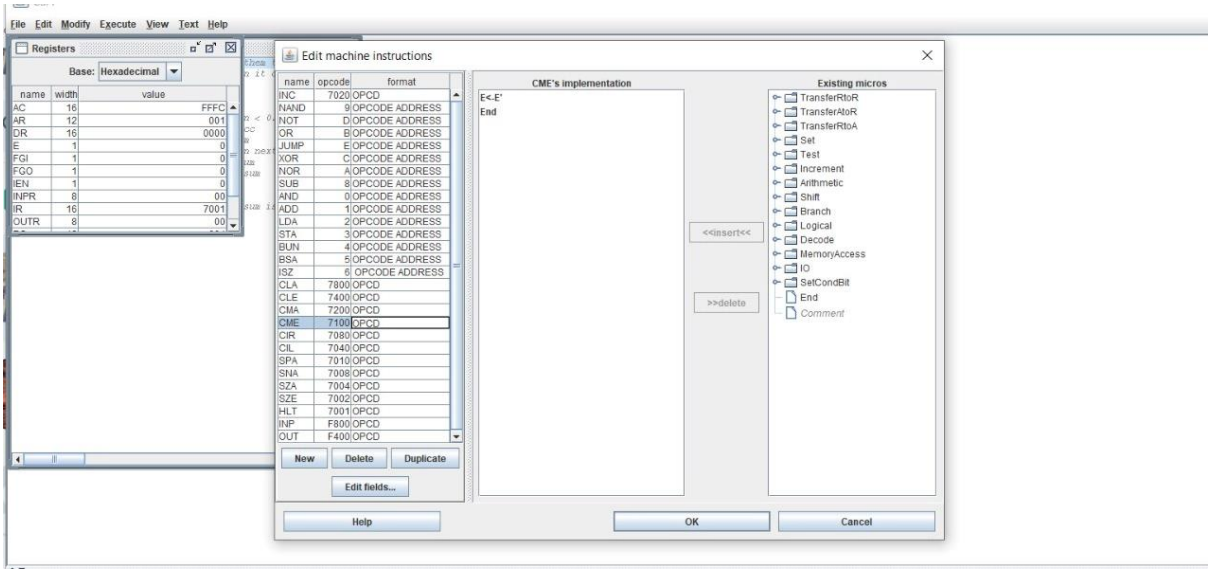
## 2.CMA

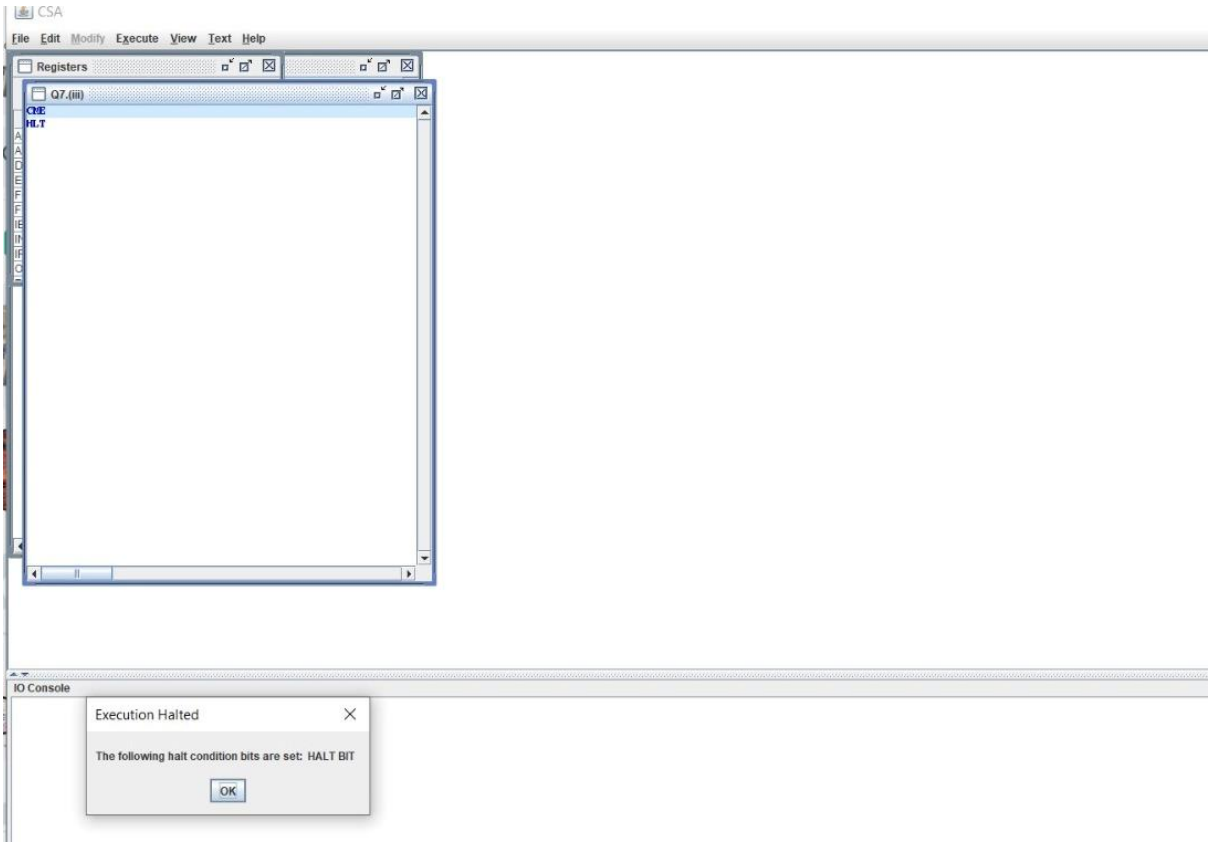

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0000
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
Help
CMA's implementation
AC<-AC'
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
OK
Cancel


File Edit Modify Execute View Text Help
Registers
Q7.(ii)
INP
CMA
OUT
HLT

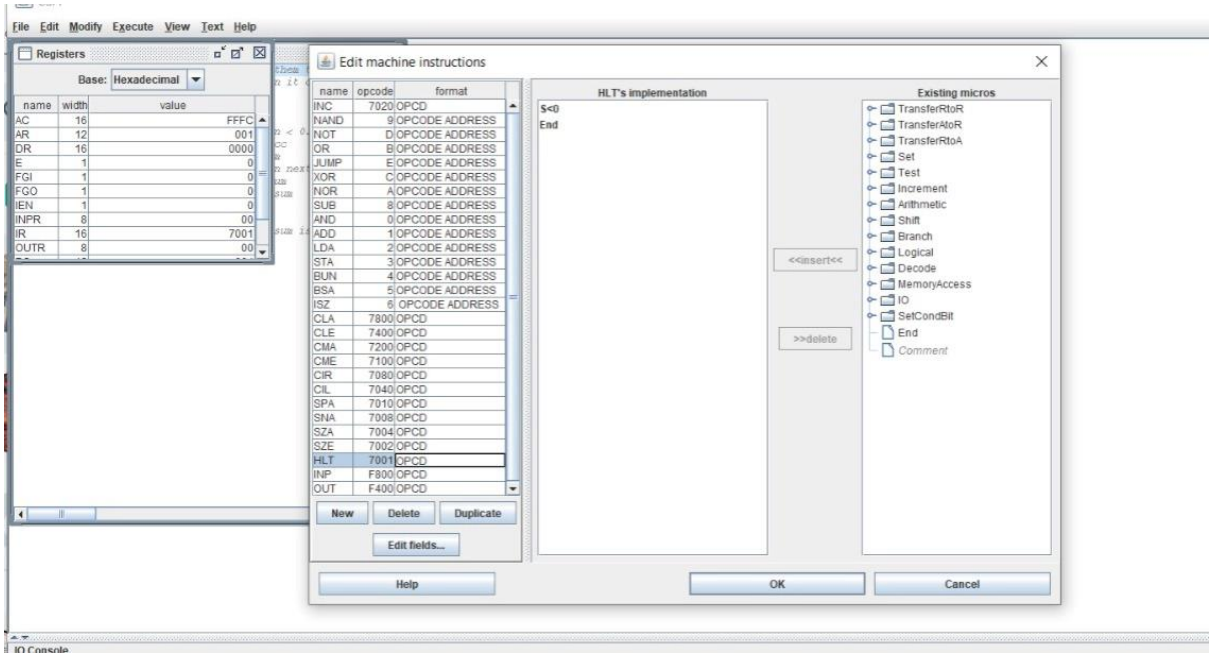

IO Console
Enter an integer: 3
Output: -4
Execution Halted
The following halt condition bits are set: HALT BIT
OK

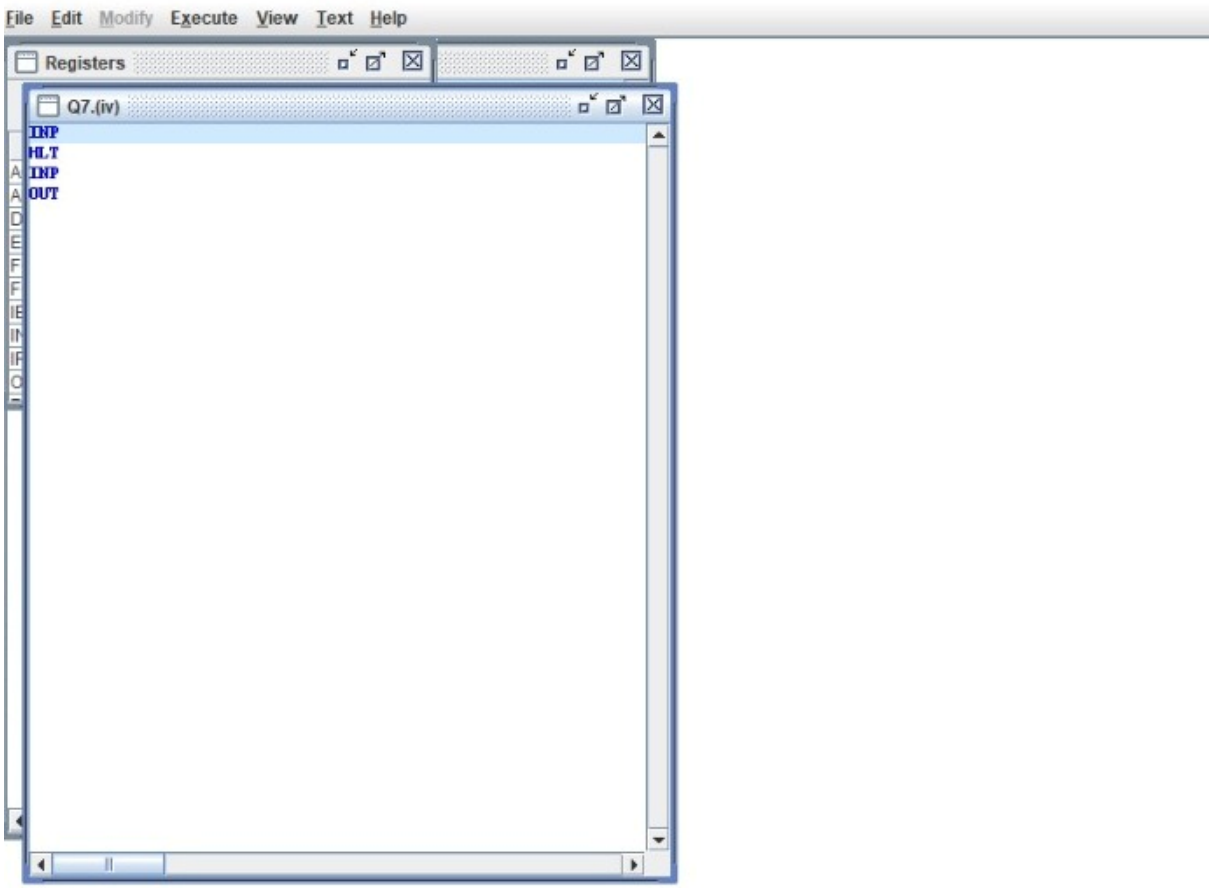
## 3.CME

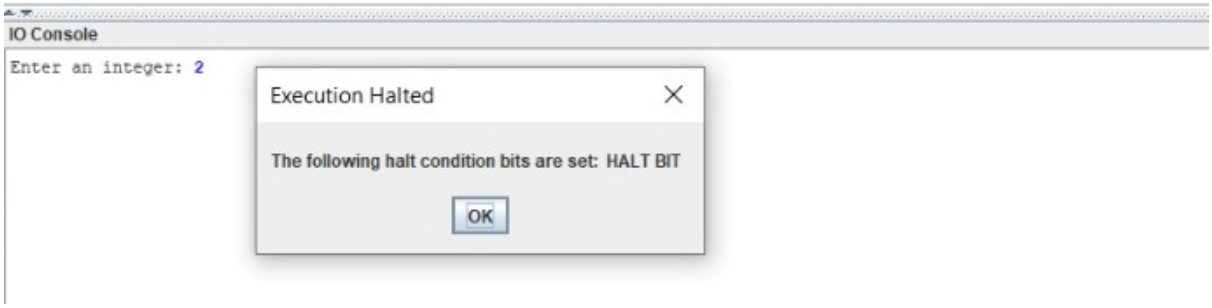

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 FFFC
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 8 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
CME's implementation
E<-E'
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


CSA
File Edit Modify Execute View Text Help
Registers
Q7.(iii)
CME
HLT
IO Console
Execution Halted
The following halt condition bits are set: HALT BIT
OK

## 4.HLT

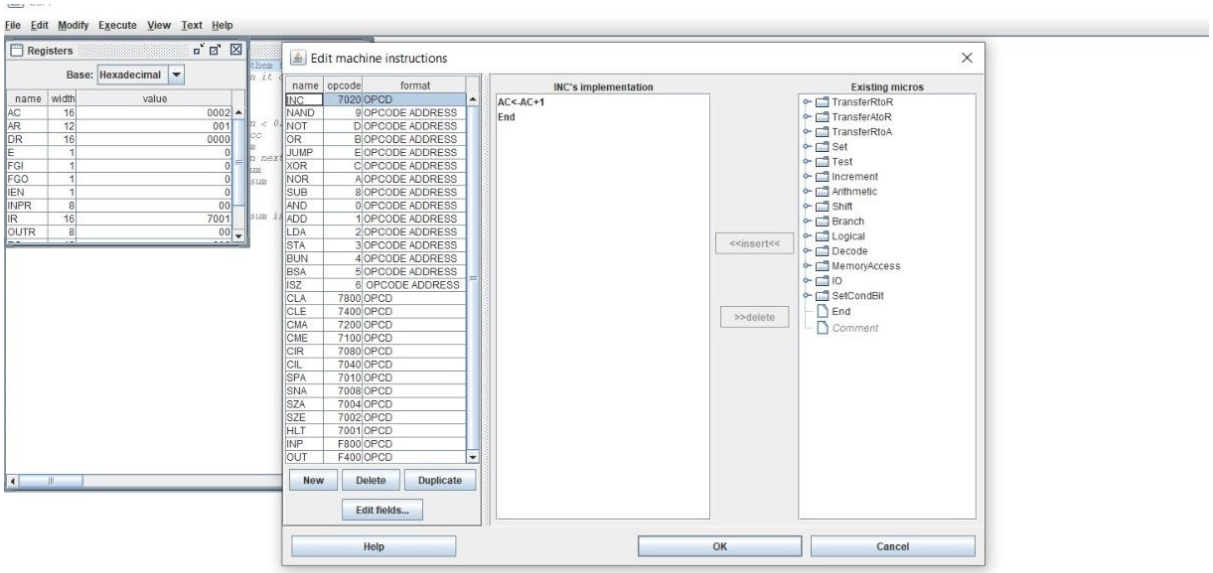

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 FFFC
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
HLT's implementation
S<0
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel
IO Console

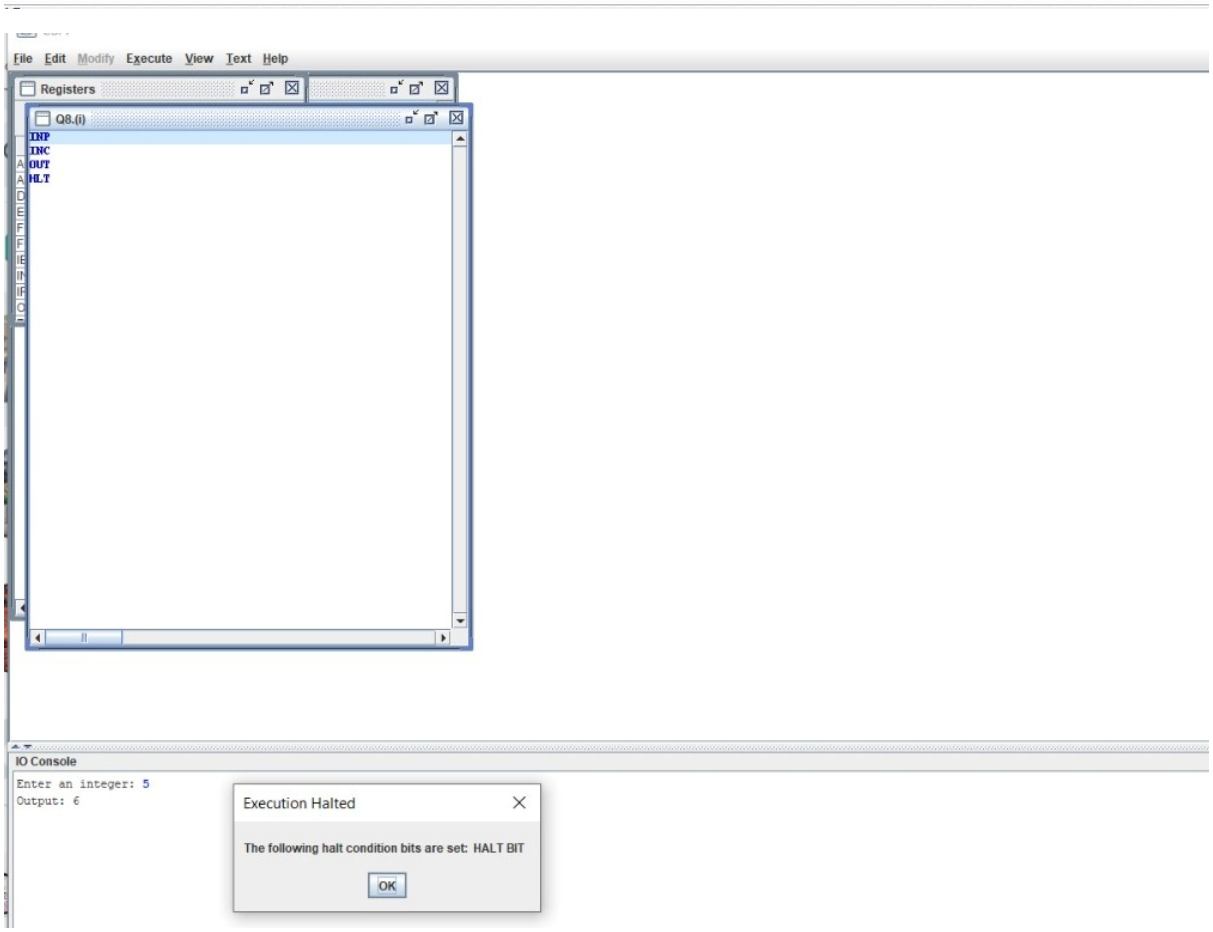


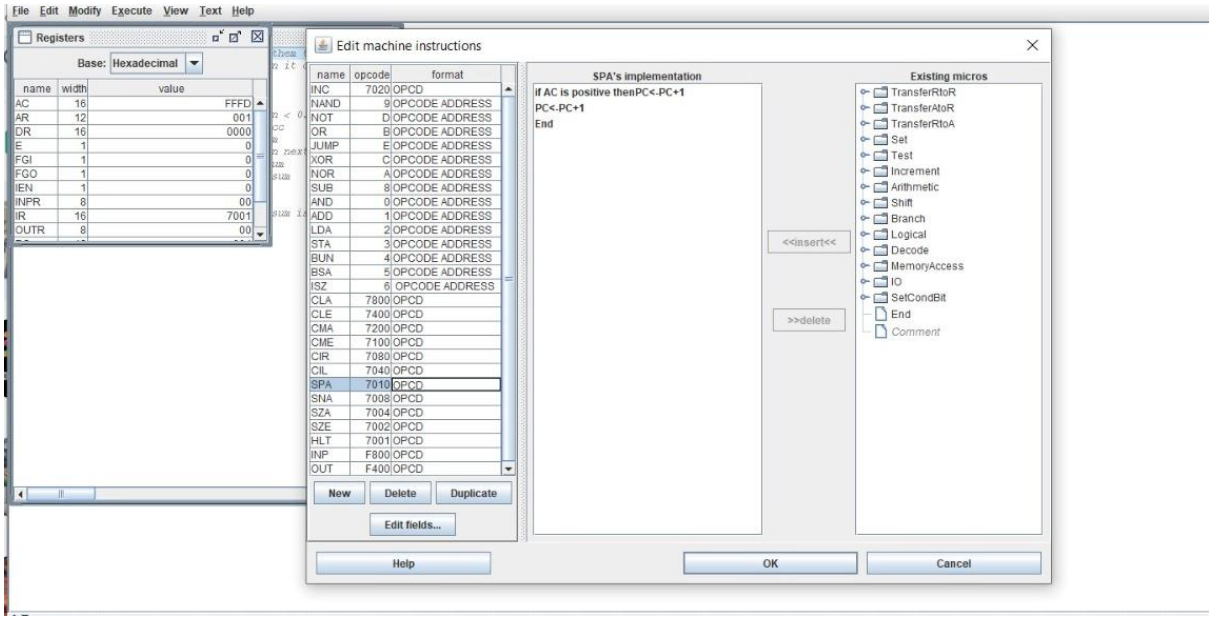

**8. WRITE AN ASSEMBLY LANGUAGE PROGRAM TO SIMULATE THE MACHINE FOR FOLLOWING REGISTER REFERENCE INSTRUCTIONS AND DETERMINE THE CONTENTS OF AC,E,PC,AR AND IR REGISTERS IN DECIMAL AFTER THE EXECUTION.**

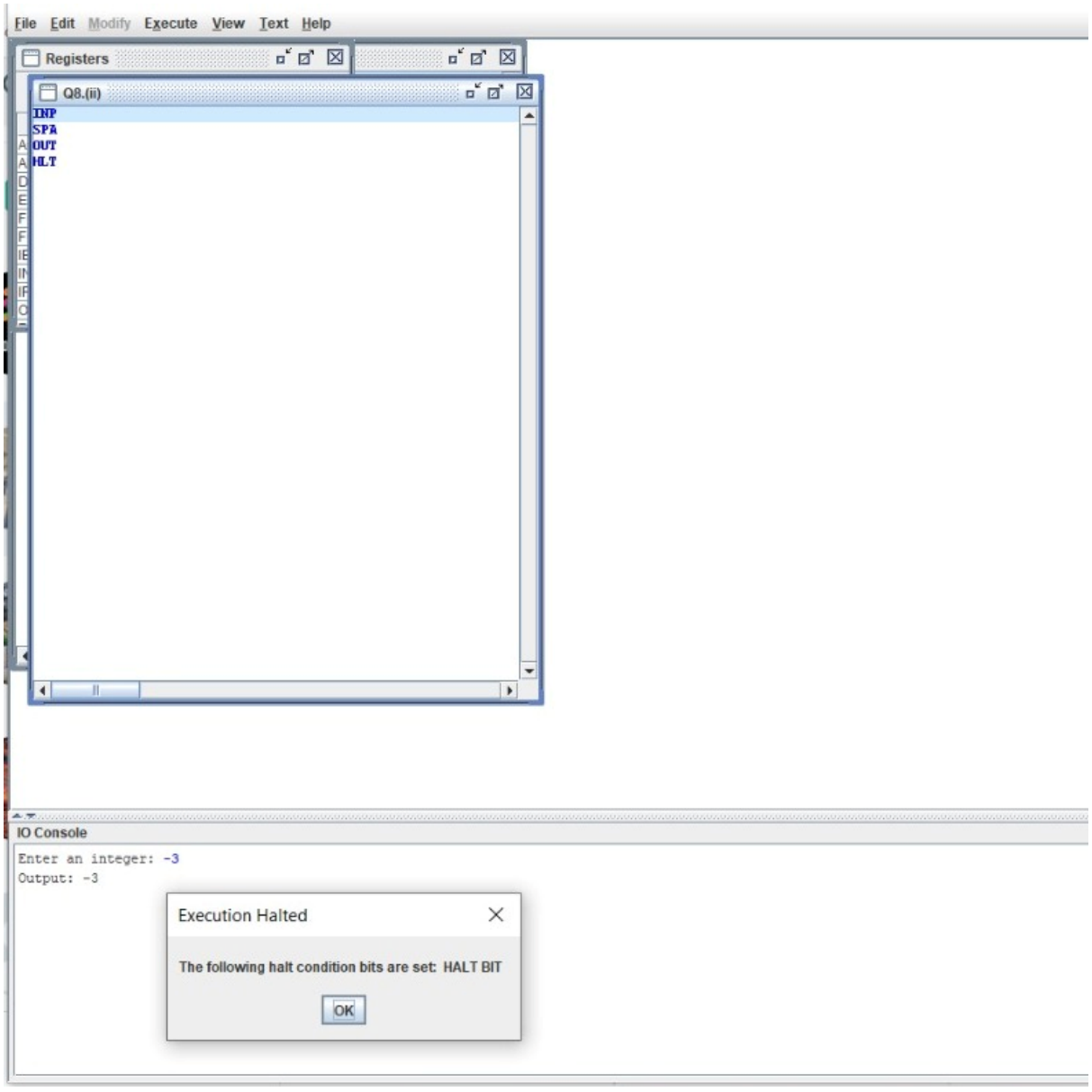
**1.INC**


File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 0002
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
Help
INC's implementation
AC<-AC+1
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
OK
Cancel

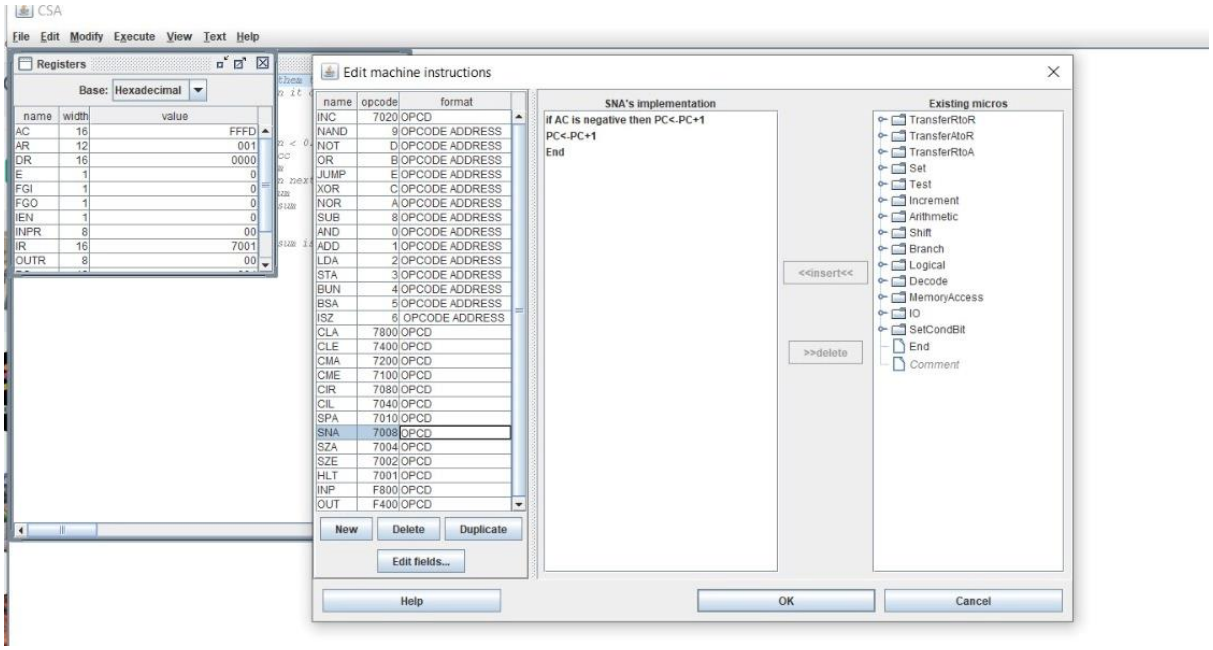

File Edit Modify Execute View Text Help
Registers
Q8.(i)
INP
INC
OUT
HLT
IO Console
Enter an integer: 5
Output: 6
Execution Halted
The following halt condition bits are set: HALT BIT
OK

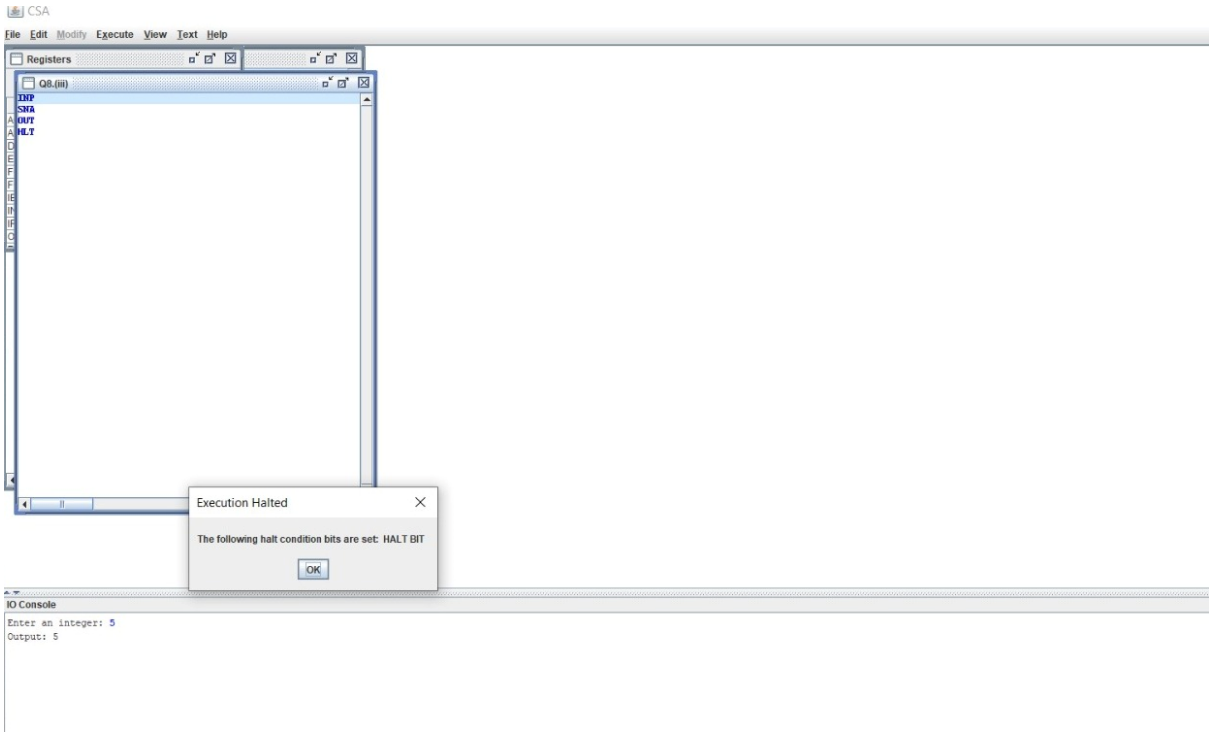
**2.SPA**


File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 FFFD
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
SPA's implementation
if AC is positive thenPC<-PC+1
PC<-PC+1
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel

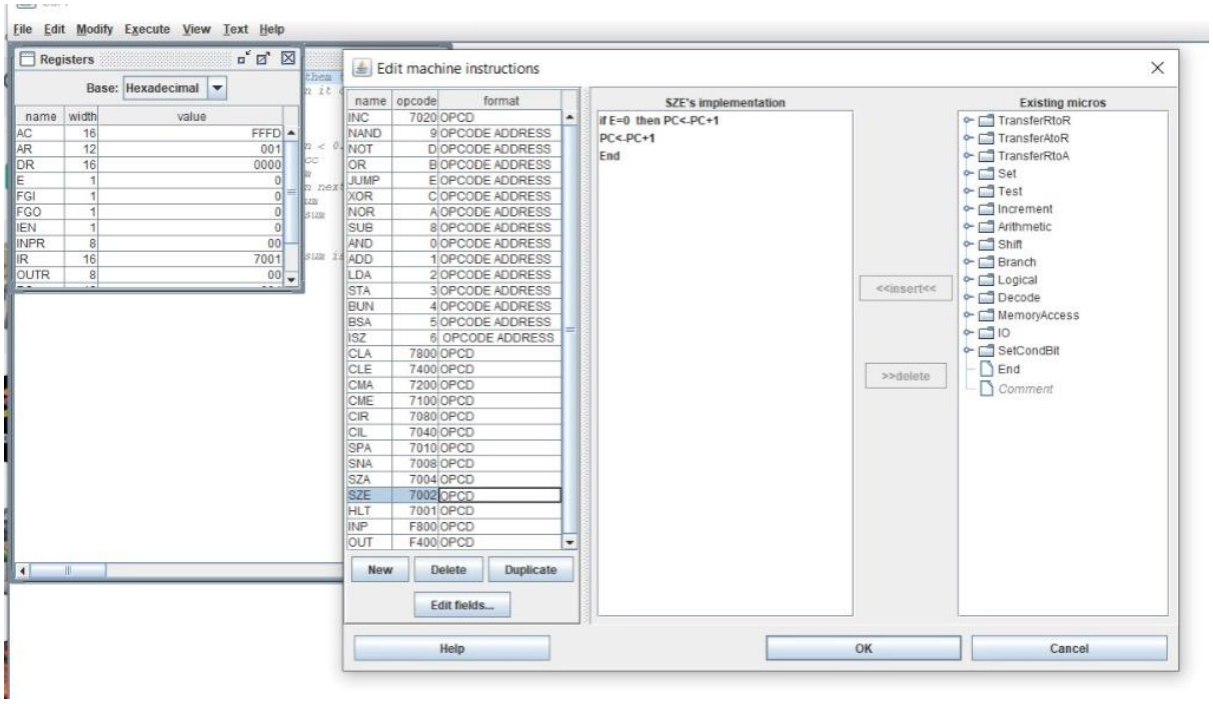

File Edit Modify Execute View Text Help
Registers
Q8.(ii)
INP
SPA
OUT
HLT
IO Console
Enter an integer: -3
Output: -3
Execution Halted
The following halt condition bits are set: HALT BIT
OK

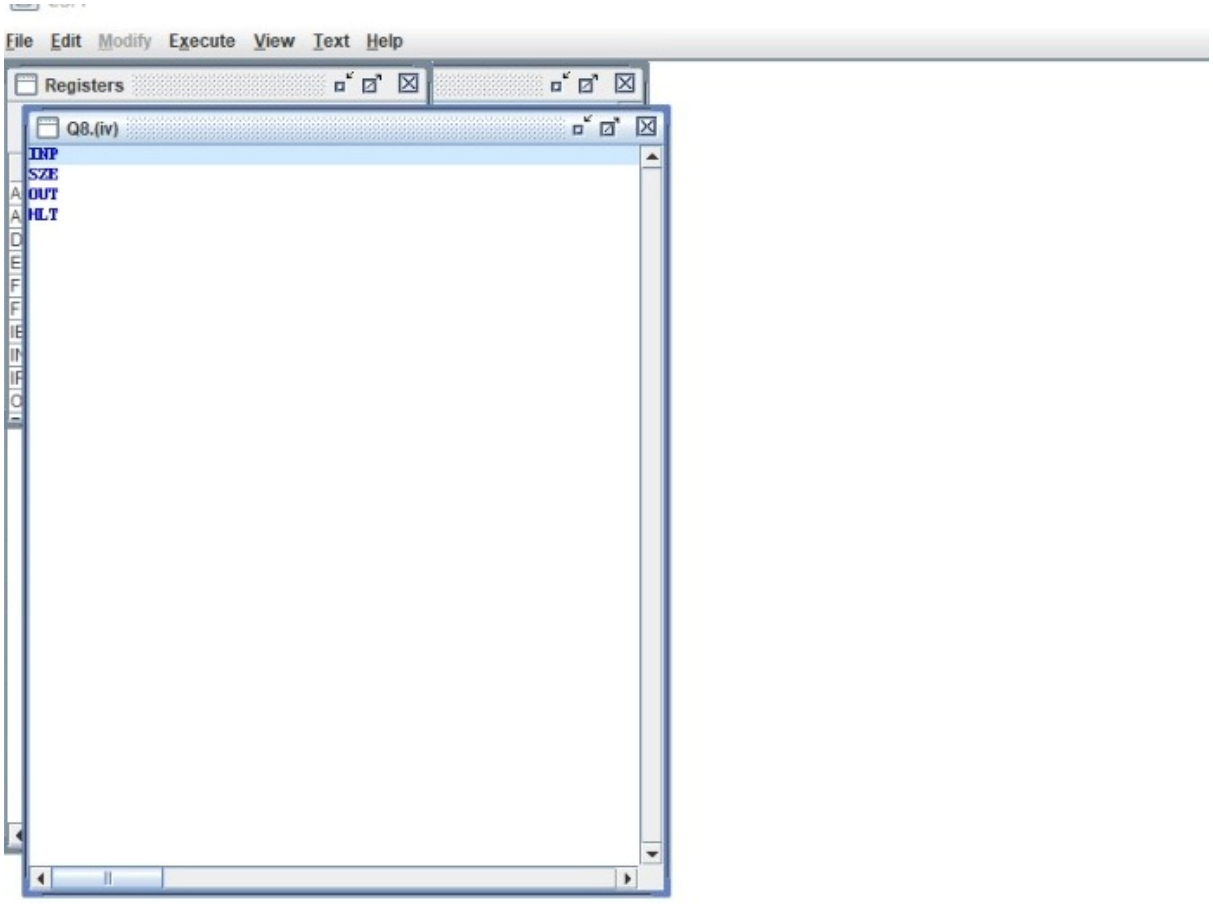
## 3.SNA

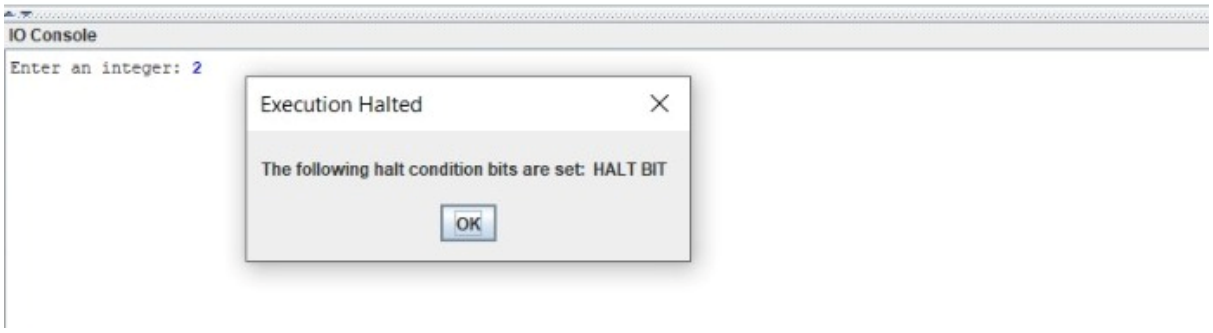

CSA
File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 FFFD
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
SNA's implementation
if AC is negative then PC<-PC+1
PC<-PC+1
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferRtoR
TransferAtoR
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


CSA
File Edit Modify Execute View Text Help
Registers
Q8.(iii)
INP
SNA
CMA
HLT
Execution Halted
The following halt condition bits are set: HALT BIT
OK
IO Console
Enter an integer: 5
Output: 5

## 4.SZE

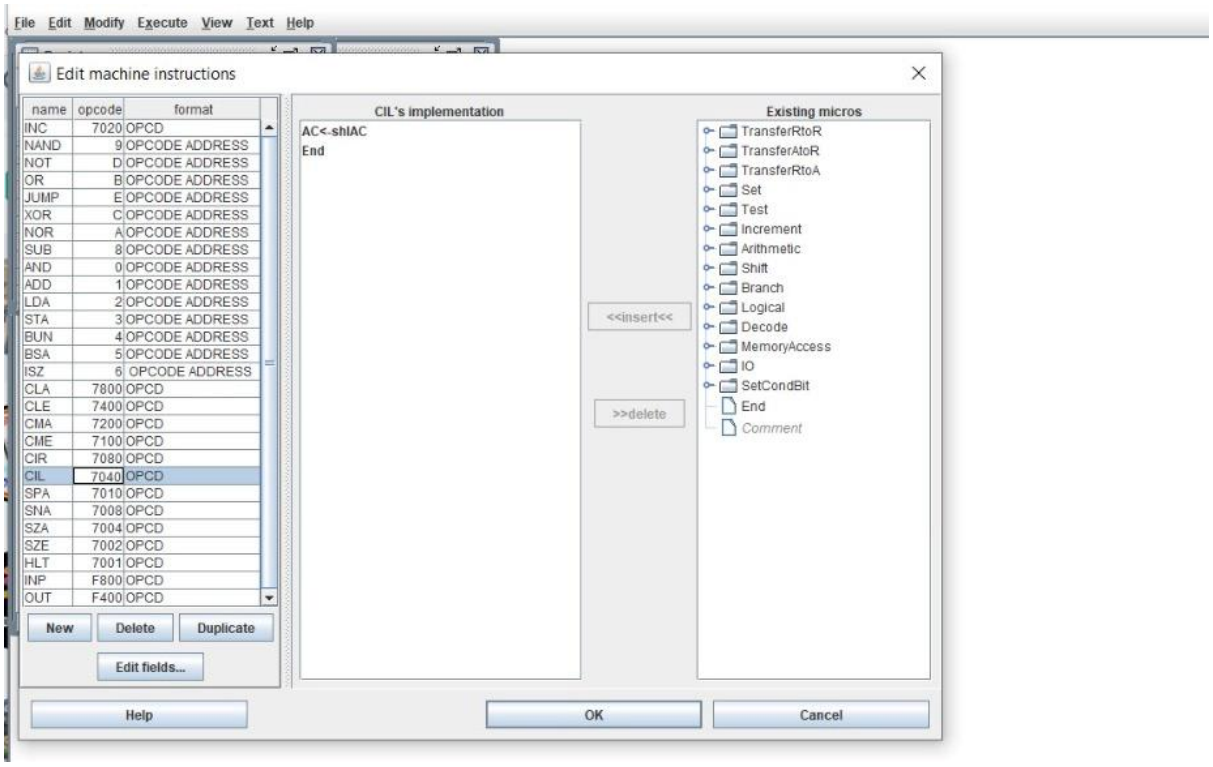

File Edit Modify Execute View Text Help
Registers
Base: Hexadecimal
name width value
AC 16 FFFD
AR 12 001
DR 16 0000
E 1 0
FGI 1 0
FGO 1 0
IEN 1 0
INPR 8 00
IR 16 7001
OUTR 8 00
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
SZE's implementation
if E=0 then PC<-PC+1
PC<-PC+1
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel

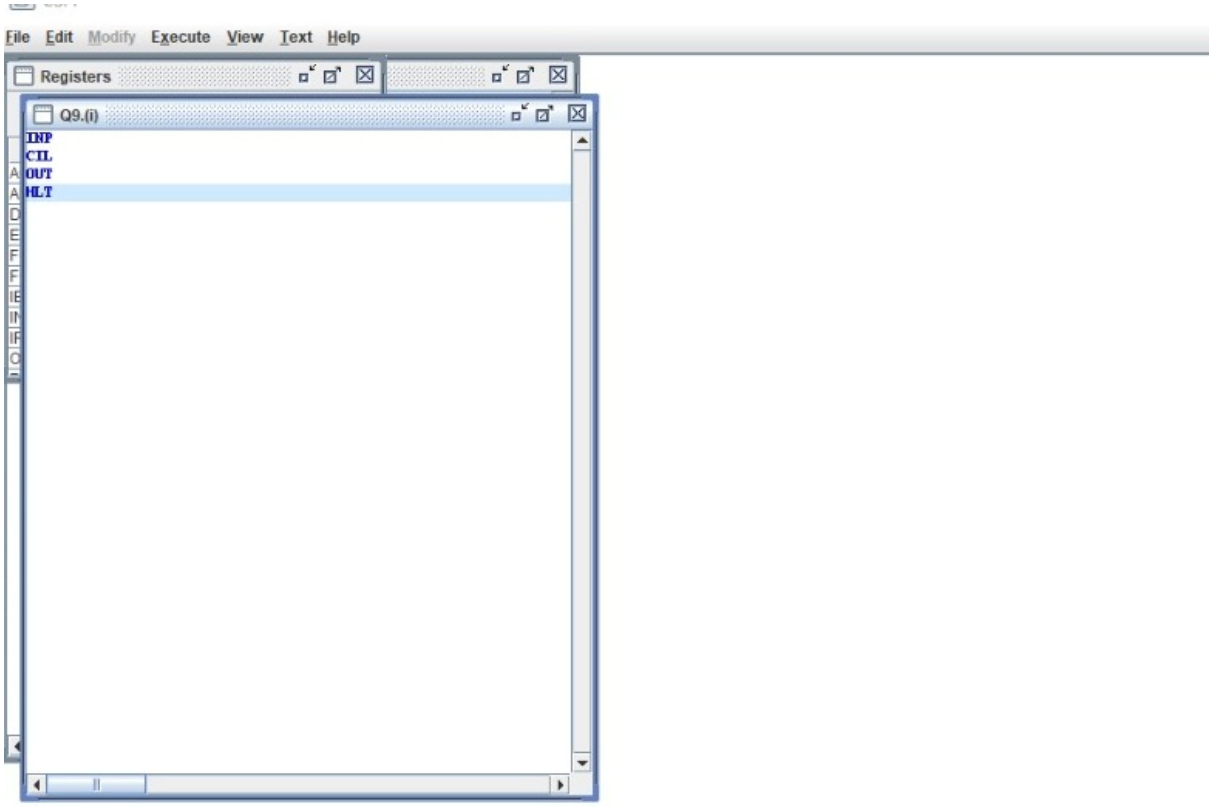


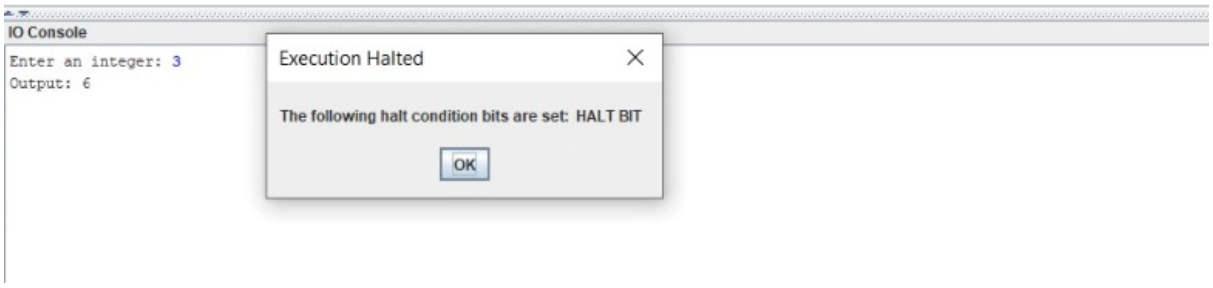




**9. WRITE AN ASSEMBLY LANGUAGE PROGRAM TO SIMULATE THE MACHINE FOR FOLLOWING REGISTER REFERENCE INSTRUCTIONS AND DETERMINE THE CONTENTS OF AC,E,PC,AR AND IR REGISTER IN DECIMAL AFTER THE EXECUTION.**

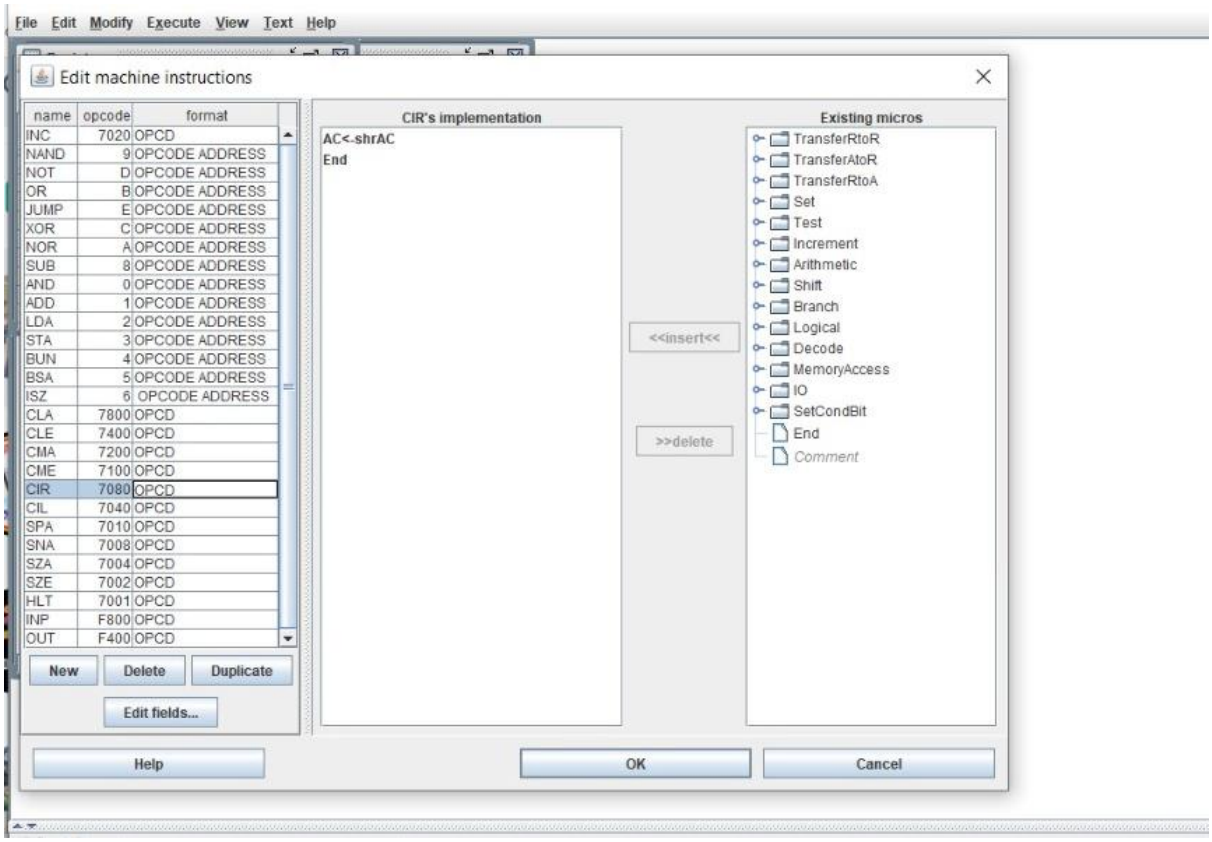
**1.CIL**

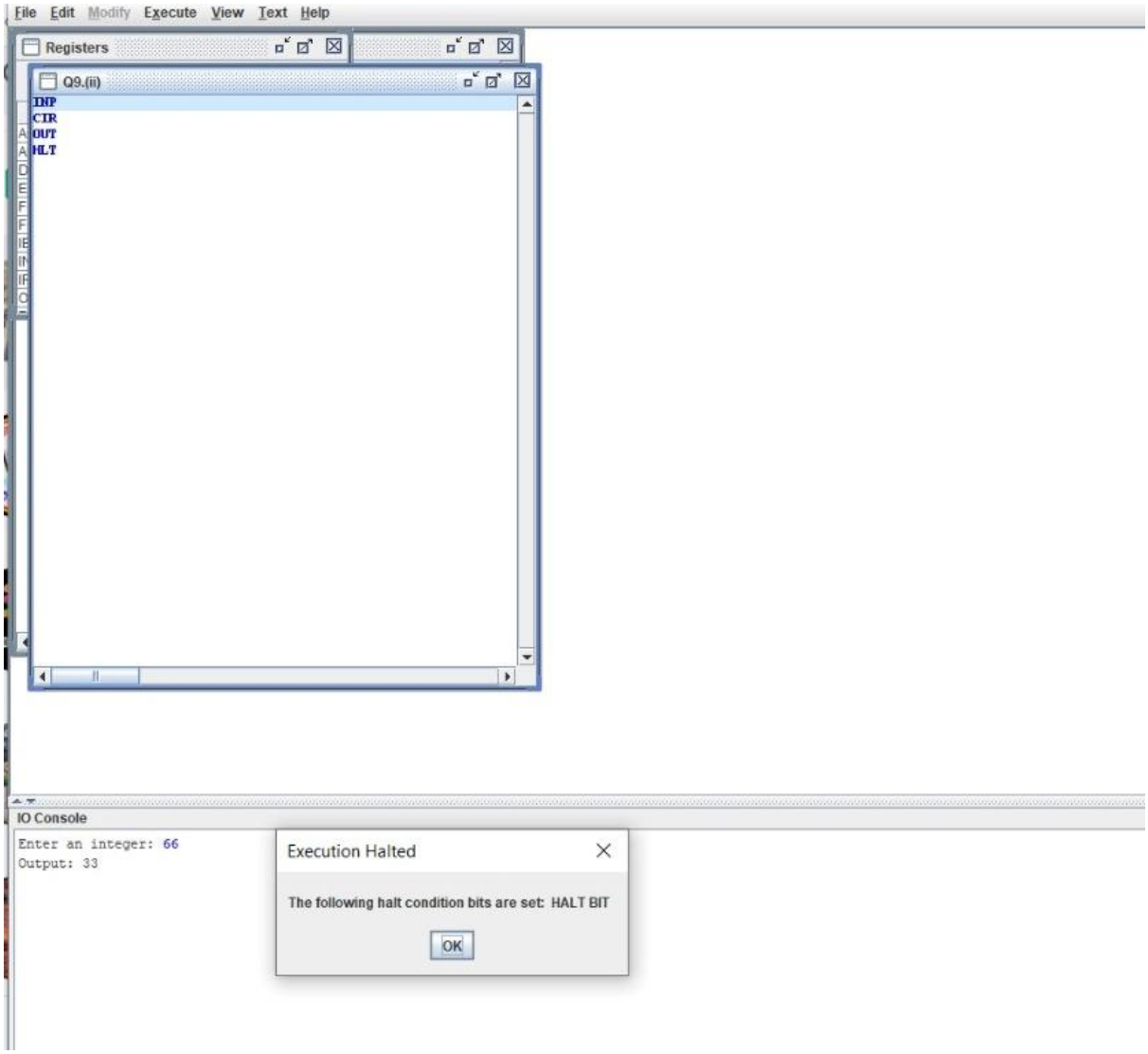

File Edit Modify Execute View Text Help
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
CIL's implementation
AC<-shlAC
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


File Edit Modify Execute View Text Help
Registers
Q9.(i)
INP
CIL
OUT
HLT

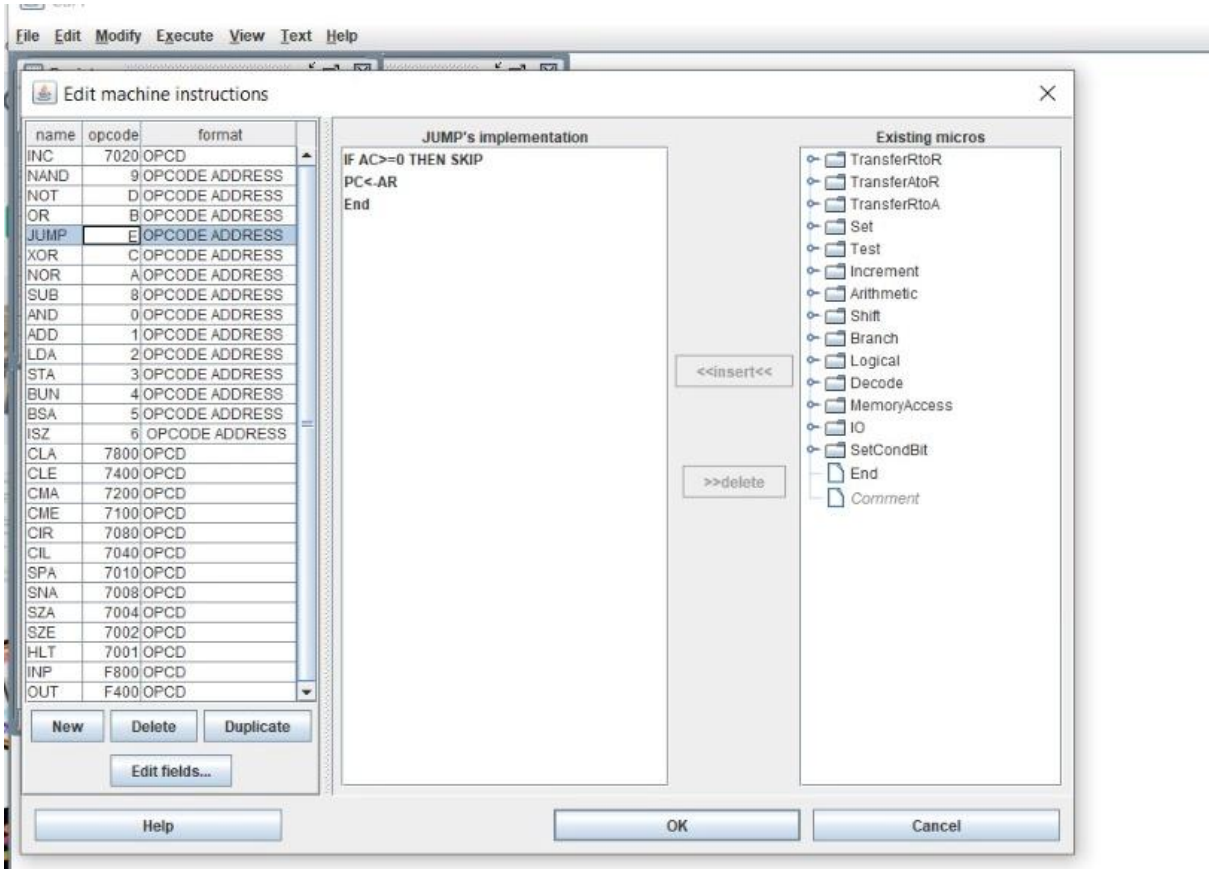

IO Console
Enter an integer: 3
Output: 6
Execution Halted
The following halt condition bits are set: HALT BIT
OK

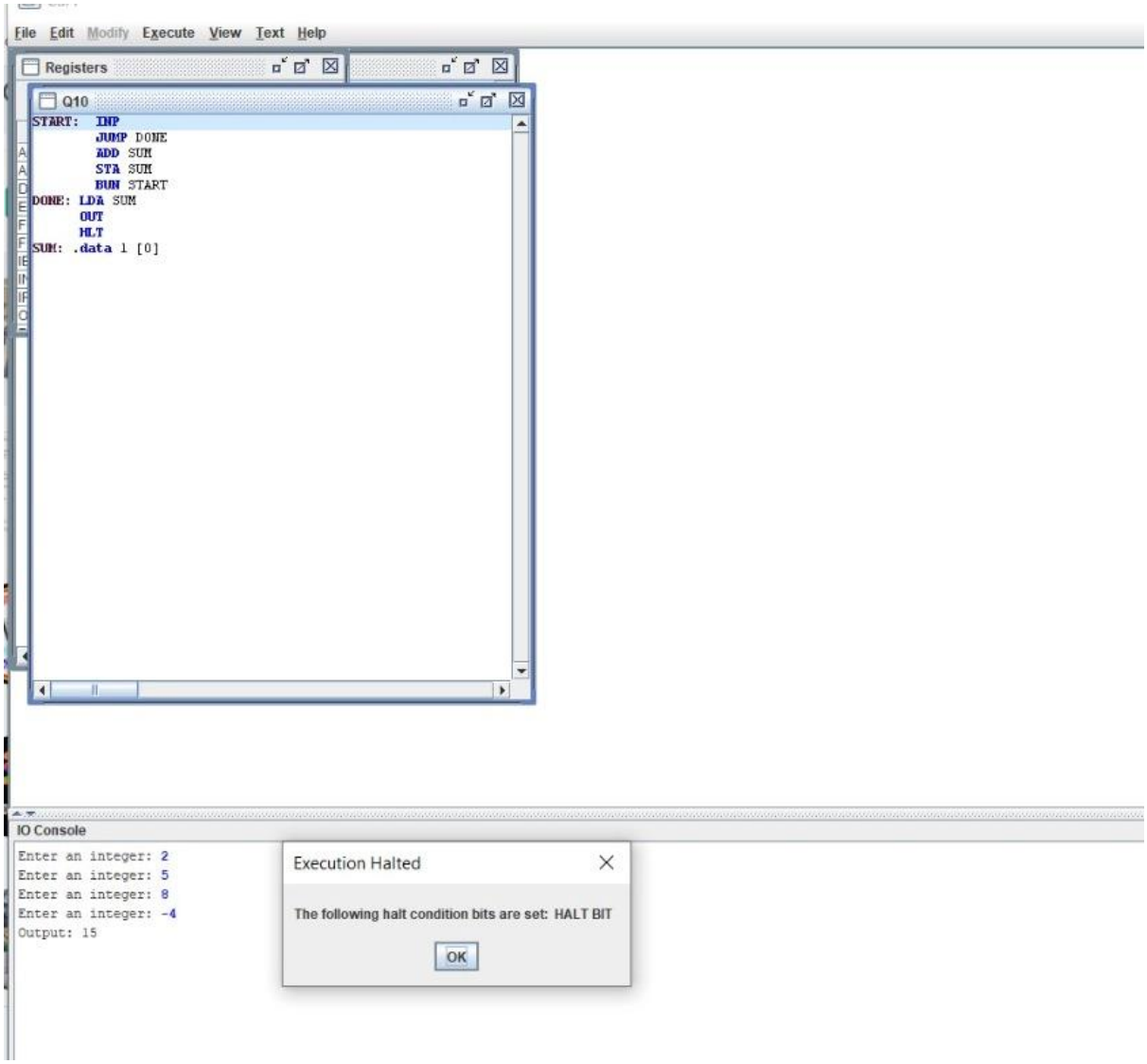
2. **CIR**


File Edit Modify Execute View Text Help
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
CIR's implementation
AC<-shrAC
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel

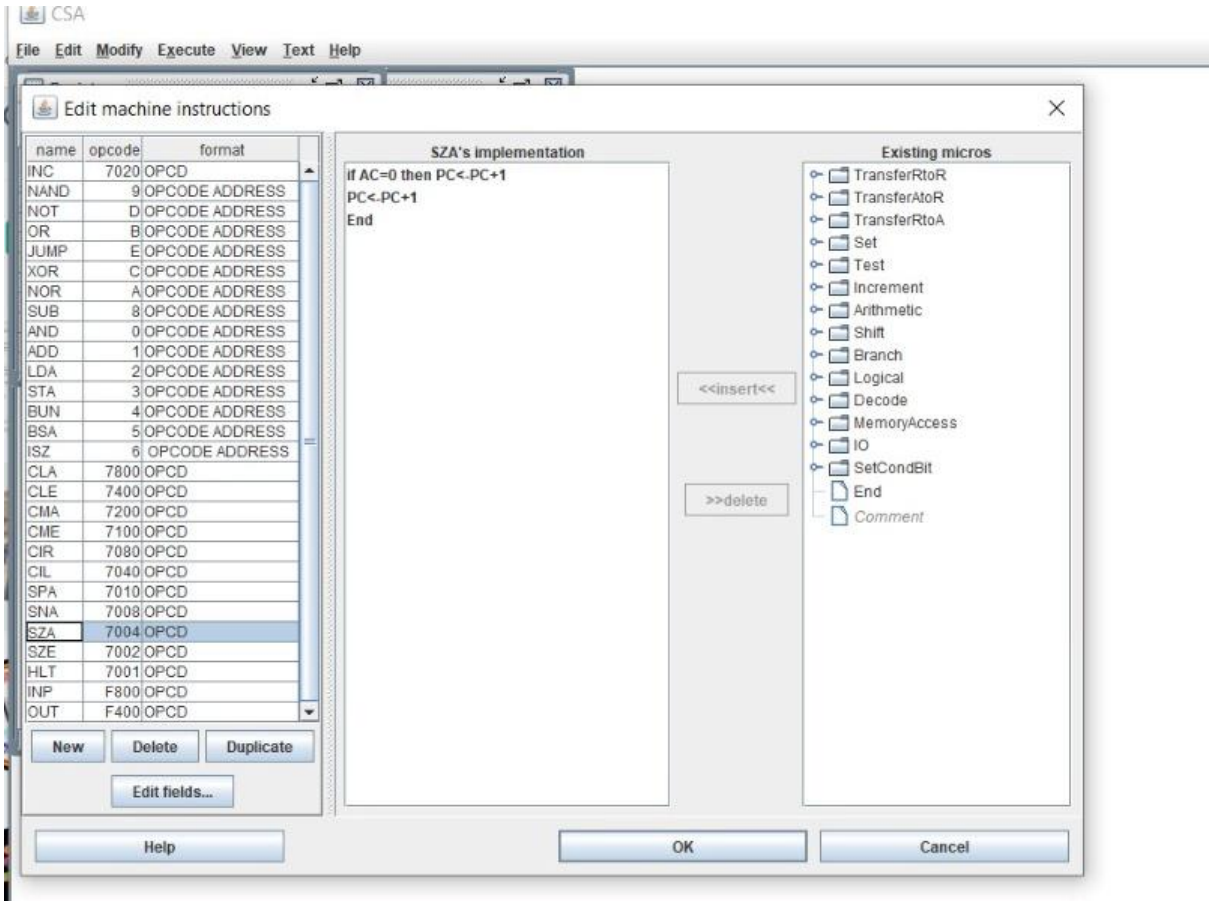


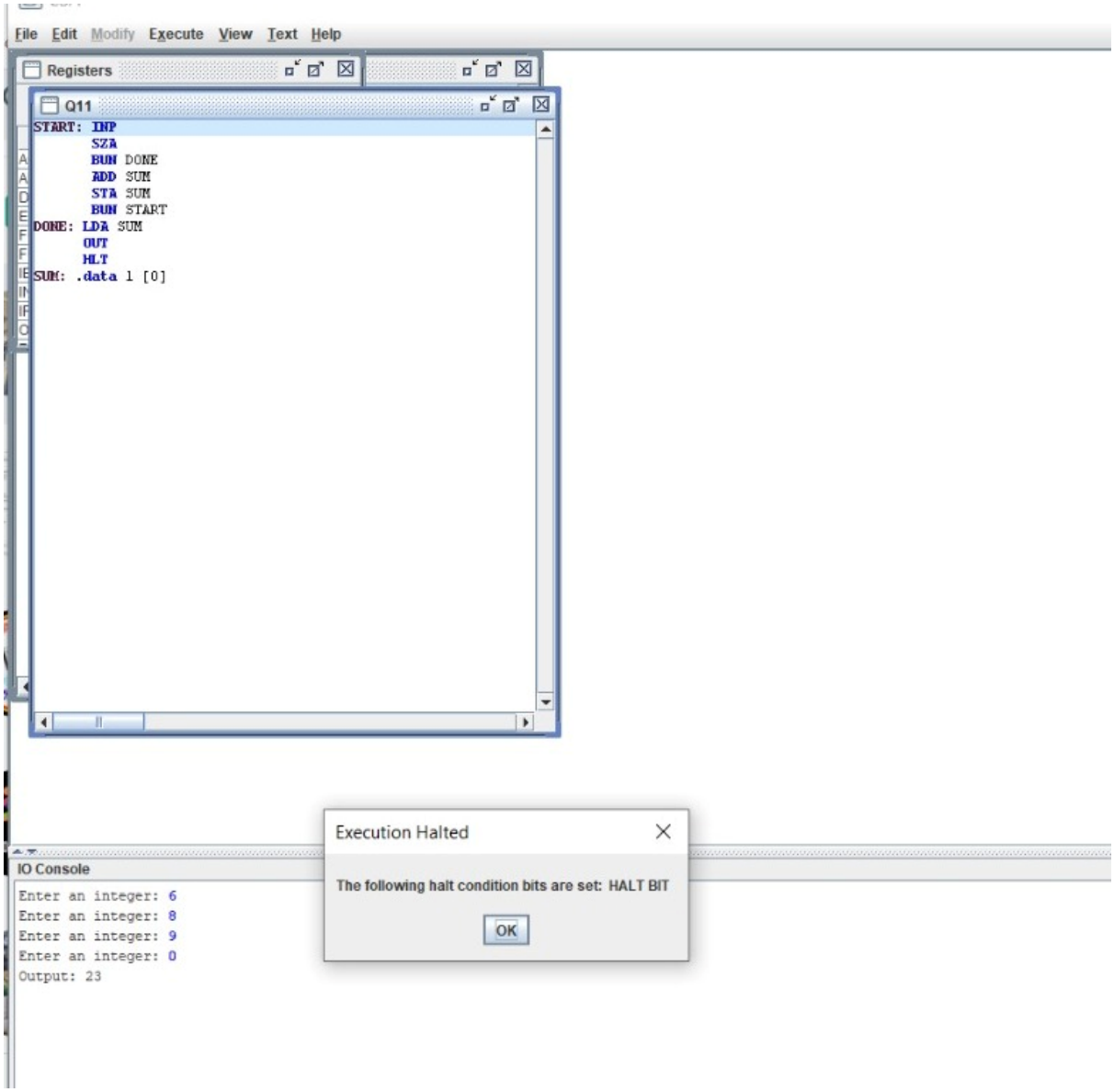

**10. WRITE AN ASSEMBLY PROGRAM THAT READS IN INTEGERS AND ADDS THEM TOGETHER ; UNTIL A NEGATIVE NON ZERO NUMBER IS READ IN .THEN IT OUTPUTS THE SUM (NOT INCLUDING THE LAST NUMBER).**


File Edit Modify Execute View Text Help
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
JUMP's implementation
IF AC>=0 THEN SKIP
PC<-AR
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel




**11. WRITE AN ASSEMBLY PROGRAM THAT READS IN INTEGERS AND ADDS THEM TOGETHER :UNTIL ZERO IS READ IN .THEN IT OUTPUTS THE SUM.**


CSA
File Edit Modify Execute View Text Help
Edit machine instructions
name opcode format
INC 7020 OPCD
NAND 9 OPCODE ADDRESS
NOT D OPCODE ADDRESS
OR B OPCODE ADDRESS
JUMP E OPCODE ADDRESS
XOR C OPCODE ADDRESS
NOR A OPCODE ADDRESS
SUB 8 OPCODE ADDRESS
AND 0 OPCODE ADDRESS
ADD 1 OPCODE ADDRESS
LDA 2 OPCODE ADDRESS
STA 3 OPCODE ADDRESS
BUN 4 OPCODE ADDRESS
BSA 5 OPCODE ADDRESS
ISZ 6 OPCODE ADDRESS
CLA 7800 OPCD
CLE 7400 OPCD
CMA 7200 OPCD
CME 7100 OPCD
CIR 7080 OPCD
CIL 7040 OPCD
SPA 7010 OPCD
SNA 7008 OPCD
SZA 7004 OPCD
SZE 7002 OPCD
HLT 7001 OPCD
INP F800 OPCD
OUT F400 OPCD
New
Delete
Duplicate
Edit fields...
SZA's implementation
if AC=0 then PC<-PC+1
PC<-PC+1
End
<<insert<<
>>delete
Existing micros
TransferRtoR
TransferAtoR
TransferRtoA
Set
Test
Increment
Arithmetic
Shift
Branch
Logical
Decode
MemoryAccess
IO
SetCondBit
End
Comment
Help
OK
Cancel


File Edit Modify Execute View Text Help
Registers
Q11
START: INP
SZA
BUN DONE
ADD SUM
STA SUM
BUN START
DONE: LDA SUM
OUT
HLT
SUM: .data 1 [0]
Execution Halted
The following halt condition bits are set: HALT BIT
OK
IO Console
Enter an integer: 6
Enter an integer: 8
Enter an integer: 9
Enter an integer: 0
Output: 23