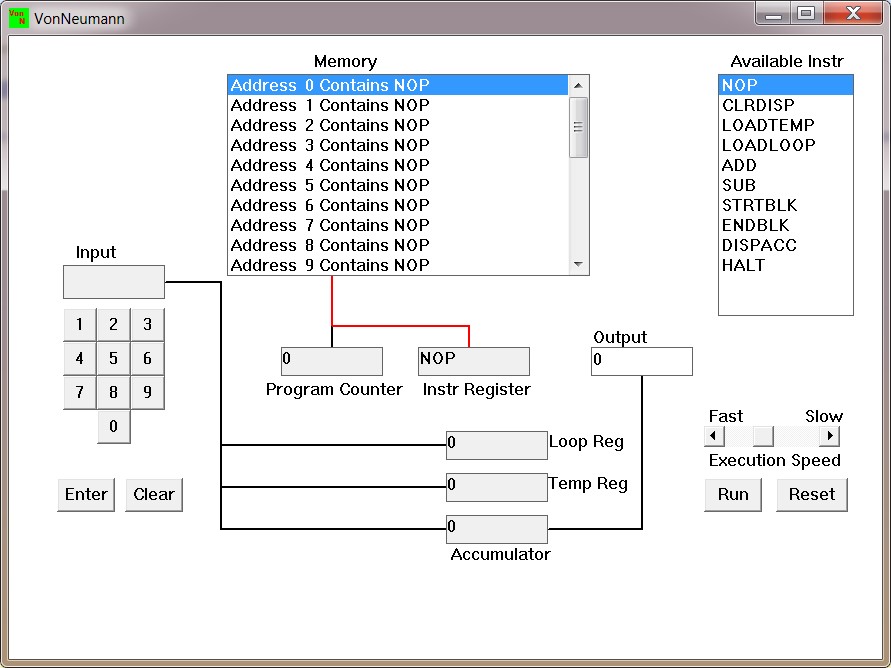
**4CS015 Fundamentals of Computing – Workshop-6**

**Workshop tasks:**

1. Von Neumann Simulator. This program simulates a very simple computer with the von Neumann architecture.

1. Download the von Neumann Simulator (VonNeumann.exe) program from WOLF in the Week 5 folder. Save it in your Documents folder and run it. You will see a window similar to this:



The simulator has a small program memory area which is available for programming. To enter your program instructions simply click on the “Available” instruction on the list on the right and then click on the “Memory” location you wish to put it in.

This simulator understands only the following ten instructions:

|  |  |
| --- | --- |
| NOP | No Operation, i.e. do nothing. |
| LOADTEMP | Get a number from the keypad, completed by the Enter key, into the Temporary Register. |
| LOADLOOP | Get a number from the keypad, completed by the Enter key, into the Loop Register. |
| CLRDISP | Clear the Display. |
| ADD | Add the Temporary Register to the  Accumulator |
| SUB | Subtract the Temporary Register from the  Accumulator |
| DISPACC | Display the contents of the Accumulator |
| STRTBLK | Start of Loop Block |
| ENDBLK | End of Loop Block |
| HALT | Halt. Stop Program |

1. Load the following program into the memory:

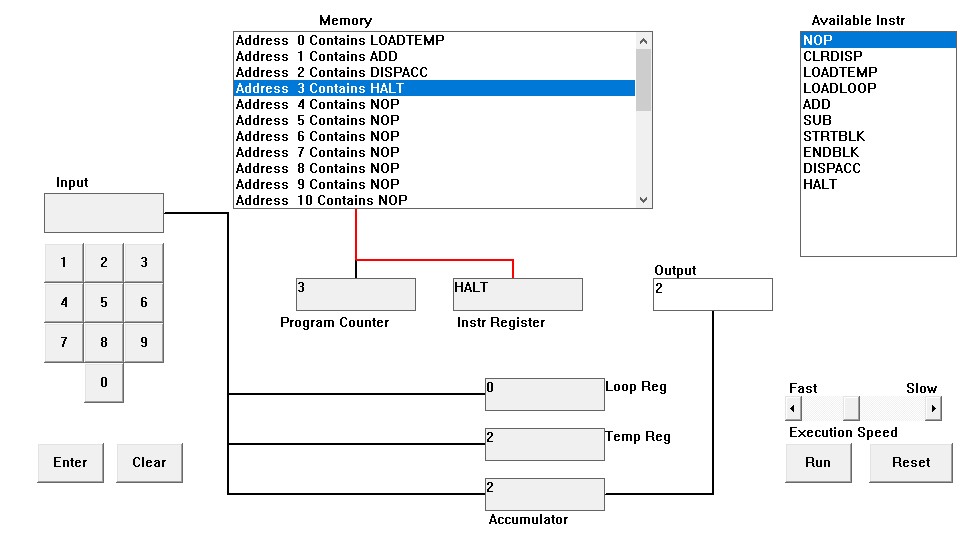
LOADTEMP

ADD

DISPACC

HALT

To do this, first click on the “LOADTEMP” in the list of instructions on the right of simulator window. Then click on Memory location with “Address 0 Contains NOP”. This will then change into “Address 0 Contains LOADTEMP”. Repeat the process with “Address 1” and so on until the whole program is loaded.

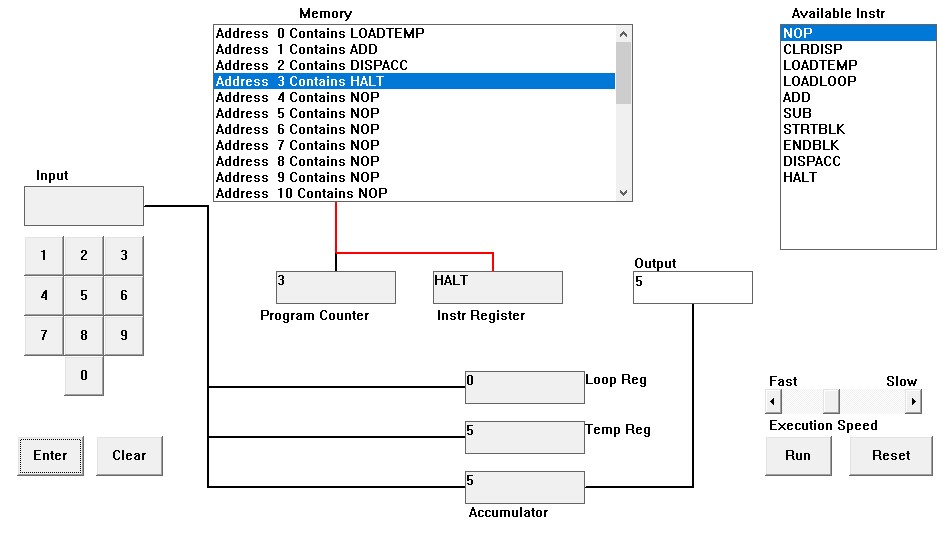


1. Run the program by clicking on the “Run” button. The simulator would highlight the Address 0 location and then pause. It is executing the instruction “LOADTEMP” which requires you to input a number into the keypad.

Click 2 or 3 numbers on the keypad and then click the “Enter” button. The simulator will then resume running the program and execute the instruction

“ADD”. This adds the number that you just entered, to the zero in the accumulator.

The next instruction is “DISPACC” which stands for “Display Accumulator”, and it does exactly that. After than the simulator stops running the program when it executes the instruction “HALT”.



1. Load the following program into the simulator:

LOADTEMP

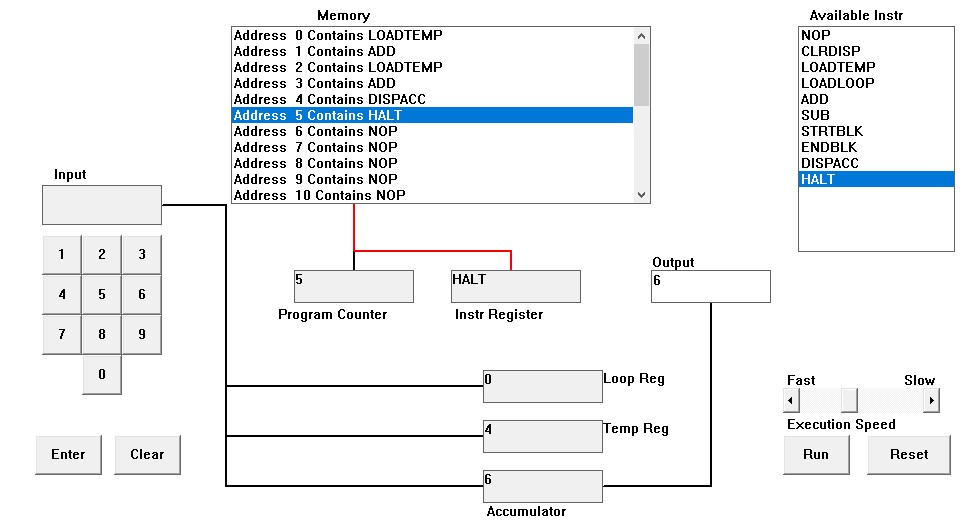
ADD

LOADTEMP

ADD

DISPACC

HALT



2+4 = 6

What do you think it does? Write your answer below (10 marks)

Ans: In the below program two number are added i.e., 2 and 4. Originally, instructions LOAD TEMP, ADD, LOAD TEMP, ADD, DISPAC, AND HAL Tare loaded to the memory independently. It also the input is taken as 2 from LOAD TEMP in address 0 which is placed in Temp Reg (Temporary Register). The number is then added with 0 due to ADD instruction in address 1. Again, the LOAD TEMP in address 3 is stimulated which motivates the user for the input. Input two is taken as 4 The ADD instruction in address 3 adds input one and input two GIVING THE RESULT 6. The added value is then stored in a collector. After all, DISPACC sends the data to the output and the program ends after HALT instruction.

e. Write a program to add 3 numbers together. List your program below (10 marks)

LOADTEMP

ADD

LOADTEMP

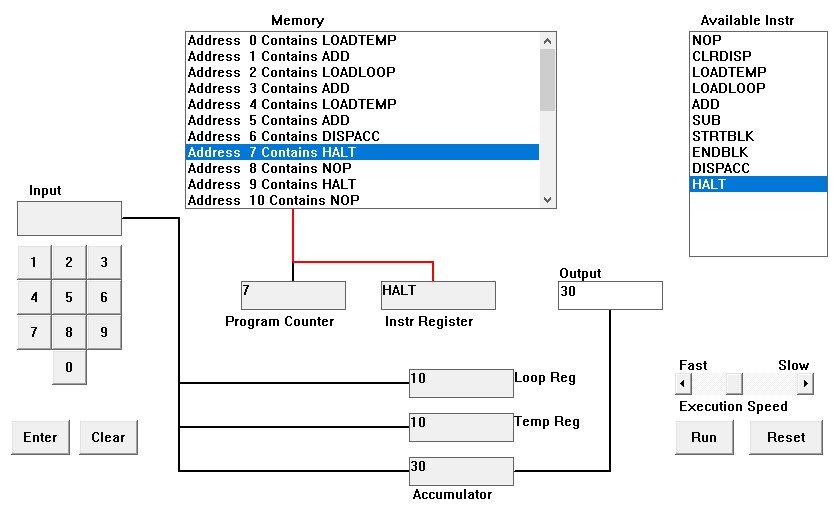
ADD

LOADTEMP

ADD

DISPACC

HALT



10+10+10=30

Ans: In the below Von Neumann Simulator, three numbers i.e., 10, 10, and 10 are added. This is done using LOADTEMP, ADD, DISPACC and HALT instructions in the memory. Originally, we use LOADTEMP in address 0 to input our first number i.e. 10 in the temporary register. The number is also added with 0 due to ADD instruction in Address 1. This process is again repeated, and we input 10 in the LOADTEMP in Address 2, which is also added with 10 using ADD instruction in Address 3. This operation results with 20 in the temporary worker register. We again repeat these instructions with LOADTEMP in Address 4 and ADD in Address 5 which adds our final input 10 with the output 20 . Eventually, the output 30 is shown using DISPACC in Address 6 and the program ends after HALT instruction in Address 6.

f. Write a program to subtract a number from another. List your program below (10 marks)

LOADTEMP

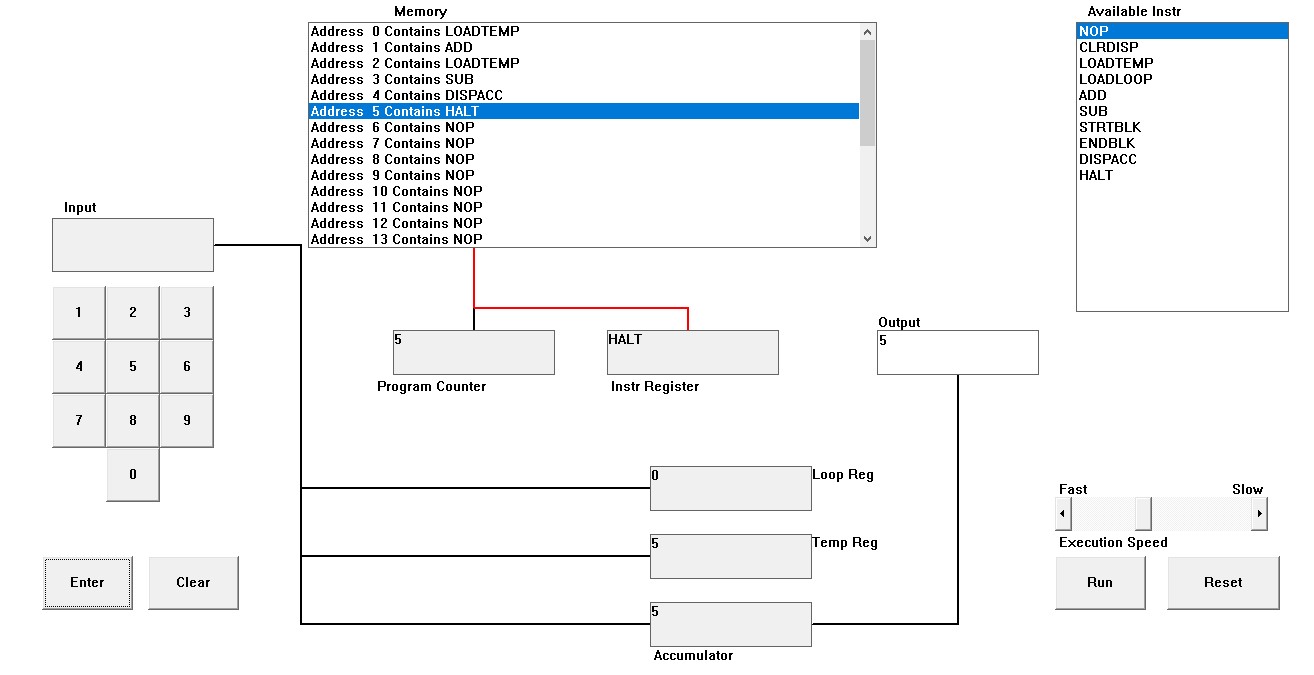
ADD

LOADTEMP

SUB

DISPACC

HALT



10-5 = 5

Ans: In the below program two numbers are i.e., 10 and 5 . Originally, instructions LOAD TEMP, ADD, LOAD TEMP, SUB , DISPAC, AND HAL Tare loaded to the

memory independently. It is also the input is taken as 10 from LOAD TEMP in address 0 which is placed in Temp Reg (Temporary Register). The number is then added with 0 due to ADD instruction in address 1. Again, the LOAD TEMP in address 3 is stimulated which motivates the user for the input. Input two is taken as The

ADD instruction in address 3 adds input one and input two GIVING THE RESULT

5 . The added value is then stored in a collector. After all, DISPACC sends the data to the output and the program ends after HALT instruction.

g. Load the following program into the simulator:

LOADTEMP

ADD

LOADLOOP

STRTBLK

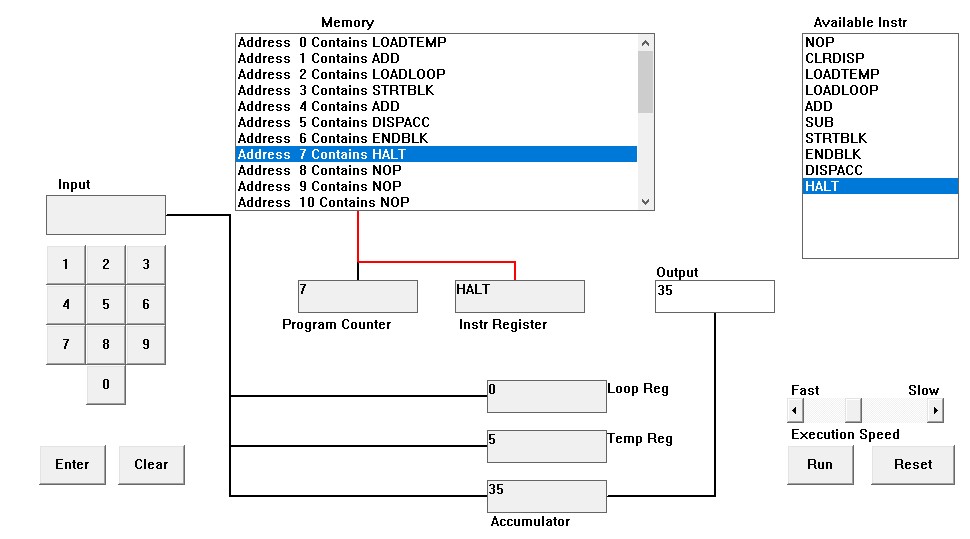
ADD

DISPACC

ENDBLK

HALT

Run it and when it reach the LOADTEMP instruction, enter 5 on the keypad and click the “Enter” button. When it reaches the LOADLOOP instruction, enter 6. What do you think the program does? Write your answer below in the form of an equation (10 marks)



Ans: First I have passed all given instructions from the available instruction to memory location. Then 5 is loaded in address 0 LOADTEMP and 6 is loaded in address 3 LOADLOOP to executes the loop 6 times. STRTBL and ENDBLK instructions are used to block the loop.

And address 5 DISPACC instruction is used to display the stored number of the accumulator. Program counter has stored the number enter at last in the address 7 and HALT instruction is stored in the instruction register. The number entered in last LOADTEMP is stored in temporary register.

Accumulator has stored the output 35 when the 5 is added with the 6 times of the 5. So, the related equation become +5+(5\*6) = 35.

h. Write a program that will let you add 5, or 10 or 20 numbers together. List your program below and explain how it works (25 marks)

# Add 5

2+2+2+2+2=10

LOADLOOP

STRTBLK

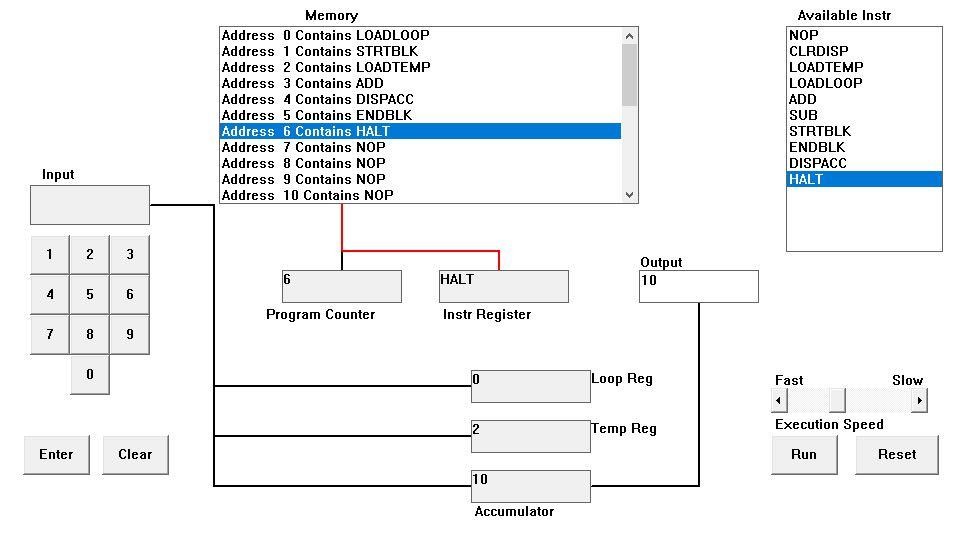
LOADTEMP

ADD

DISPACC

ENDBLK

HALT



Ans: In the below program we load 5 as we want to produce a program that can add 5 number. Now the STRTBLK instruction marks the beginning of the circle. It also we move onto address 2 which simply says to LOADTEMP or input a number in other words. We also load 2 as the first input .2 get stored in the temp regulation and gets added with 0 in the accumulation as there are not value stored in it. Also the program again circle for 4 times where we are asked to input a number. It we have input 2 for times . Eventually after the circle executes for 5 times , it eventually comes to the DISPACC Instruction where it displays the total sum contained in the accumulator which is (2 2 2 2 2 ), 5 . The program eventually stops due to HALT.

# Add 10

2+2+2+2+2+2+2+2+2+2=20

LOADLOOP

STRTBLK

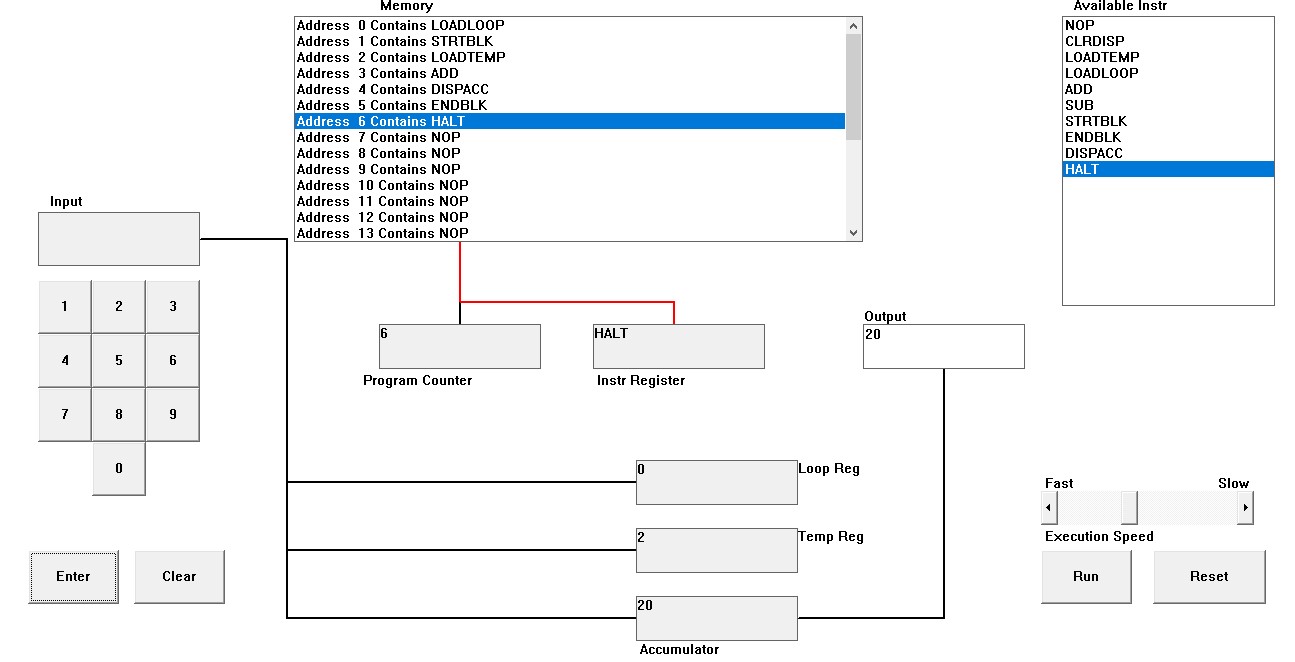
LOADTEMP

ADD

DISPACC

ENDBLK

HALT



Ans: At the below program we at first LOADLOOP has passed the 10 to execute the loop 10 time. So we can store random 10 number in the LOADTEMP . I have entered 2 in LOADTEMP for ten times. So the output becomes 20 which is stored in accumulator . The last number stored in LOADTEMP is stored in Temporary register r . In memory Location STRTBLK and ENDBLK are the instruction to block the loop of the program. And DISPACC

instructions display the content of the accumulator. So the output 20 is stored in accumulator. Finally HALT instruction has stopped and ended the program.

# Add 20

2+2+2+2+2+2+2+2+2+2+2+2+2+2+2+2+2+2+2+2=40

LOADLOOP

STRTBLK

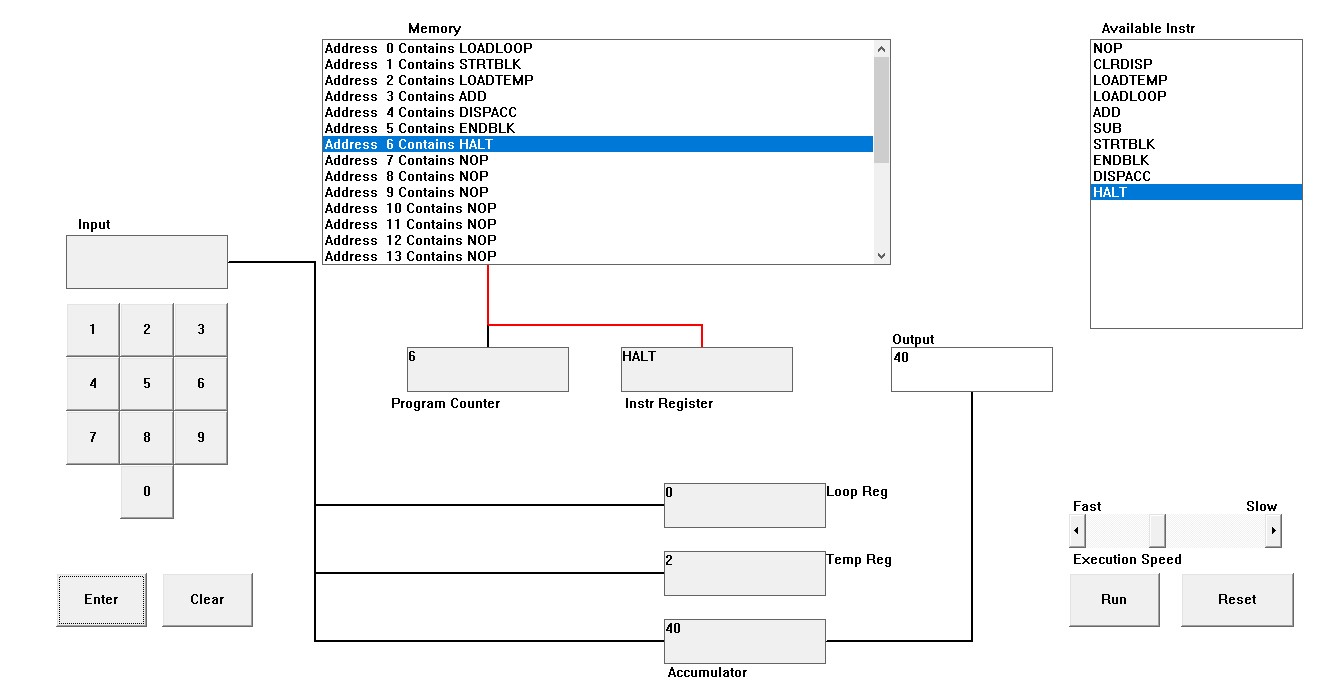
LOADTEMP

ADD

DISPACC

ENDBLK

HALT



Ans: At the below program we at first LOADLOOP has passed the 20 to execute the loop 20 time. So we can store random 20 number in the LOADTEMP . I have entered 2 in LOADTEMP for twenty times. So the output becomes 40 which is stored in accumulator . The last number stored in LOADTEMP is stored in Temporary register r . In memory

Location STRTBLK and ENDBLK are the instruction to block the loop of the program. And DISPACC instructions display the content of the accumulator. So the output 40 is stored in accumulator. Finally HALT instruction has stopped and ended the program.

i. Write a program that will let you multiply 2 numbers together. List your program below and explain how it works (35 marks)

4\*2=8

LOADLOOP

STRTBLK

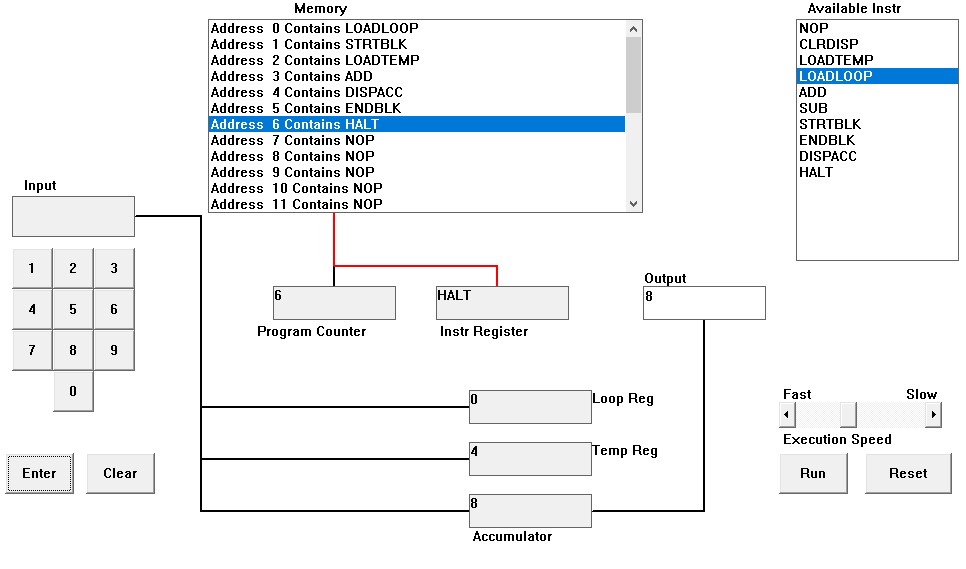
LOADTEMP

ADD

DISPACC

ENDBLK

HALT



Ans: First we are asked to load a circle. For this instruction we have loaded 2 in LOAD circle instruction as want to multiply 4 to 2 .Now the STRTBLK instruction help us on beginning the circle block. Now in Address 2 we have asked to LOADTEMP which simply means to input a number. As mentioned we will use 4 and 2 . Now we load 4 which gets stored in the Temp Reg and gets added with 0 in the accumulator as there are no values in the accumulator yet . as we have input 2 as the circle duration. We have to input the same number which is 4 . For three different times, other wise this addition program does not display the correct input. Finally after adding 4 for two different times. We will reach to the coming instruction which is DISPACC. This instruction displays the total value in the affair box. For this question, the answer is 8.r and affair.