A REPORT ON SIMPLE AS POSSIBLE (SAP-1) COMPUTER

By

Mirazul Islam

Roll:1607062

Md. Atik Shahariar

Roll:1607065

Course Name: Digital System Design Laboratory

Course No: CSE4224

Submitted to:

Dr. Md. Aminul Haque Akhand

Professor, Dept. of CSE, KUET

Jakaria Rabbi

Assistant Professor, Dept. of CSE, KUET

Khulna University of Engineering & Technology
Khulna 920300, Bangladesh

June 2021

Objectives: The purpose of this project is given below:

→ To learn about SAP-1 computer

→ To learn about different components of SAP-1 computer

→ To design a SAP-1 computer using logisim

Introduction: The simple-As-possible (SAP)-1 computer is a very basic model of a micro processor explained by Albert paul Malvino. The SAP-1 design contains the basic necessaties for a functional micro-processor. Its primary purpose is to develop basic understanding of how a micro-processor works, interrects with memory and other parts of the system like input and output. The instruction set is very limited and simple SAP-1 is the first state in the evaluation towords modern computers.

Fig - 1 shows the architecture of SAP - 1, a bus organized computer. All registers outputs towords to the w bus are three state, this allows orderly transfer of data

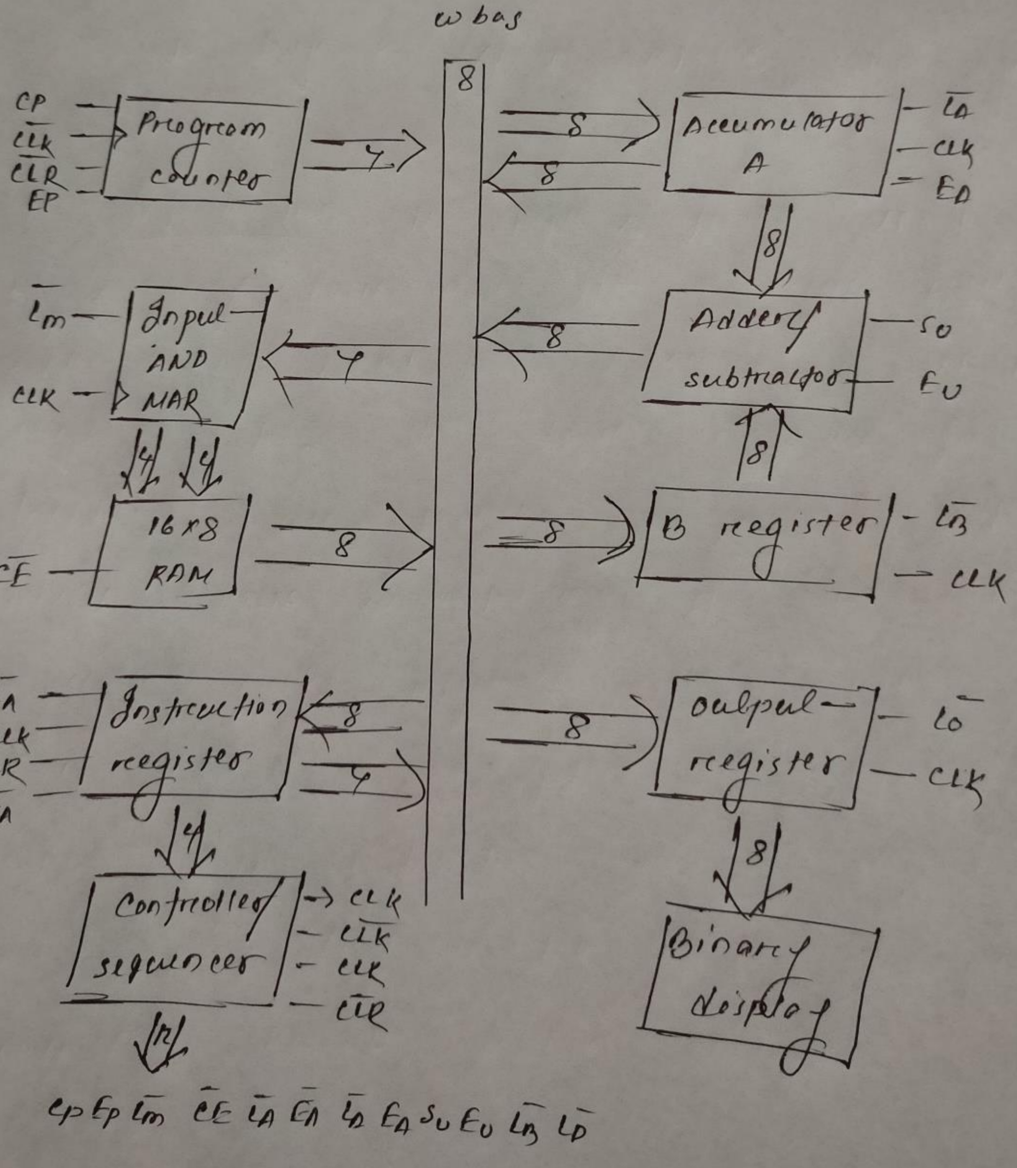
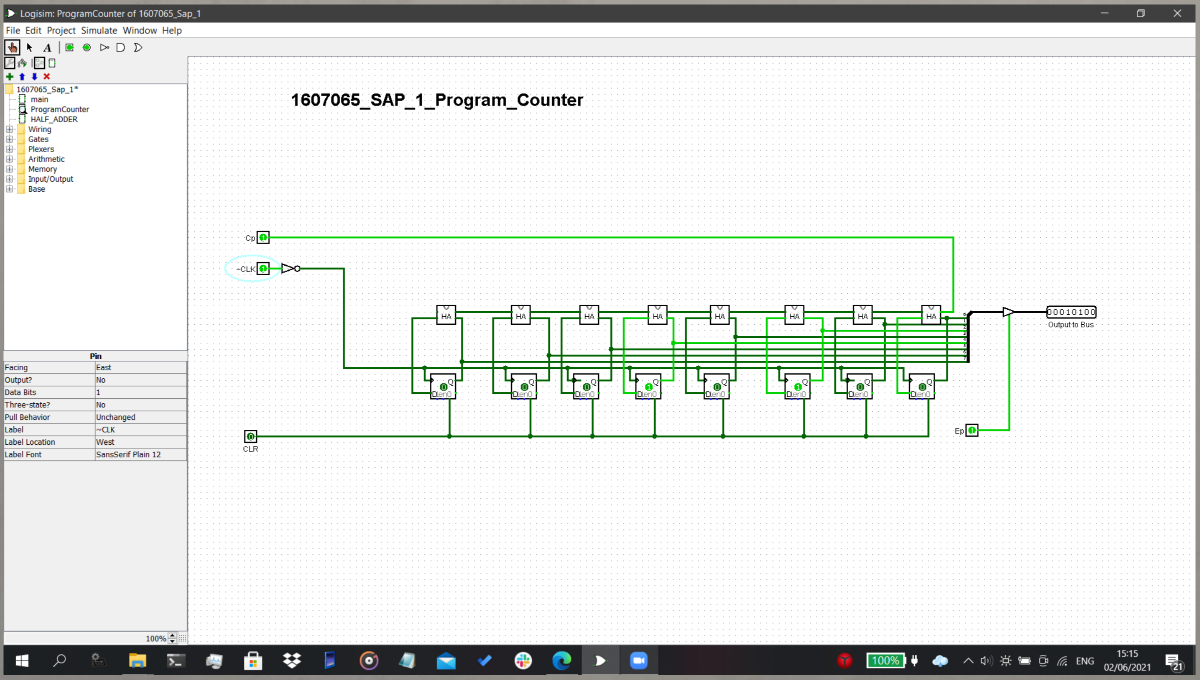


Fig 1. SAP-1 architecture

Program counter: The program counter is reset to 0000 before each computer run. when the computer run begins, the PC sends address 0000 to memory. The PC then is incremented to get 0001. After the first instruction is fetched and executed. the PC sends addess 0001 to memory again the PC is incremented. In this way. PC is keeping track of the next instruction to fetched and executed.

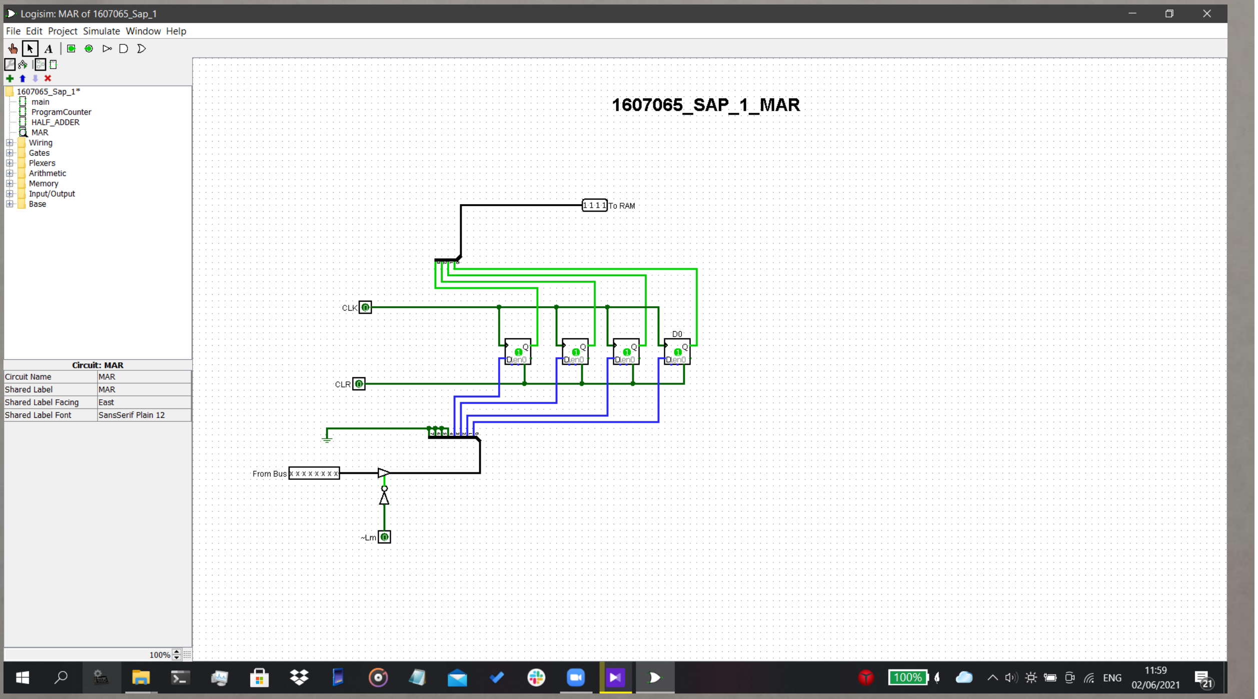


1607065_SAP_1_Program_Counter

Pin	
Facing	East
Output?	No
Data Bits	1
Three-state?	No
Pull Behavior	Unchanged
Label	~CLK
Label Location	West
Label Font	SansSerif Plain 12

Input and MAR: It includes the address and data switch registers. These switch registers which are a part of input unit, allow to send 4 address bits and 8 data bits to the RAM. As recalled, instruction and data words are written into RAM before a computer run. ~~&~~

The MAR is part of SAP-1 memory. During a computer run. The address in PC is latched into MAR.



Logisim: MAR of 1607065_Sap_1

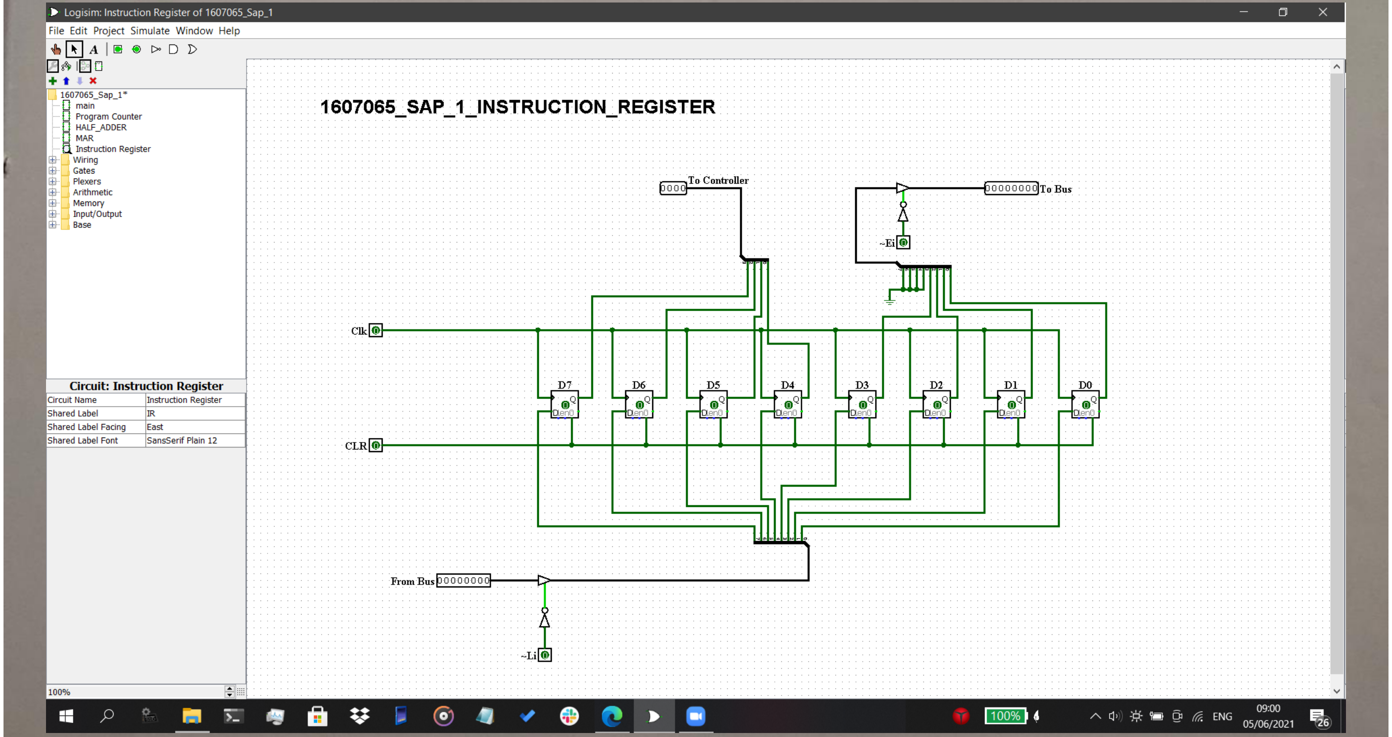
Circuit: MAR

Circuit Name	MAR
Shared Label	MAR
Shared Label Facing	East
Shared Label Font	SansSerif Plain 12

1607065_SAP_1_MAR

RAM: The RAM is a 16X8 static TTL RAM. This allows one to store a program and data in the memory before computer run. During a computer run. RAM recieves 4-bit address from MAR and a read of operation is performed. In this way. the data or instruction word in RAM. is placed on the w bus for use in some other part of the computer.

Instruction register (IR): IR is a part of control unit. To fetch an instruction The computer does a memory operation, This places the contents of addressed memory location on w bus. At the same time, IR is set-up for loading on the next positive clock edge. The upper nibble is a two state output and the lower nibble is a three -state output.



Logisim: Instruction Register of 1607065_Sap_1

1607065_Sap_1*
- main
- Program Counter
- HALF_ADDER
- MAR
- Instruction Register
- Wiring
- Gates
- Plexers
- Arithmetic
- Memory
- Input/Output
- Base

Circuit: Instruction Register

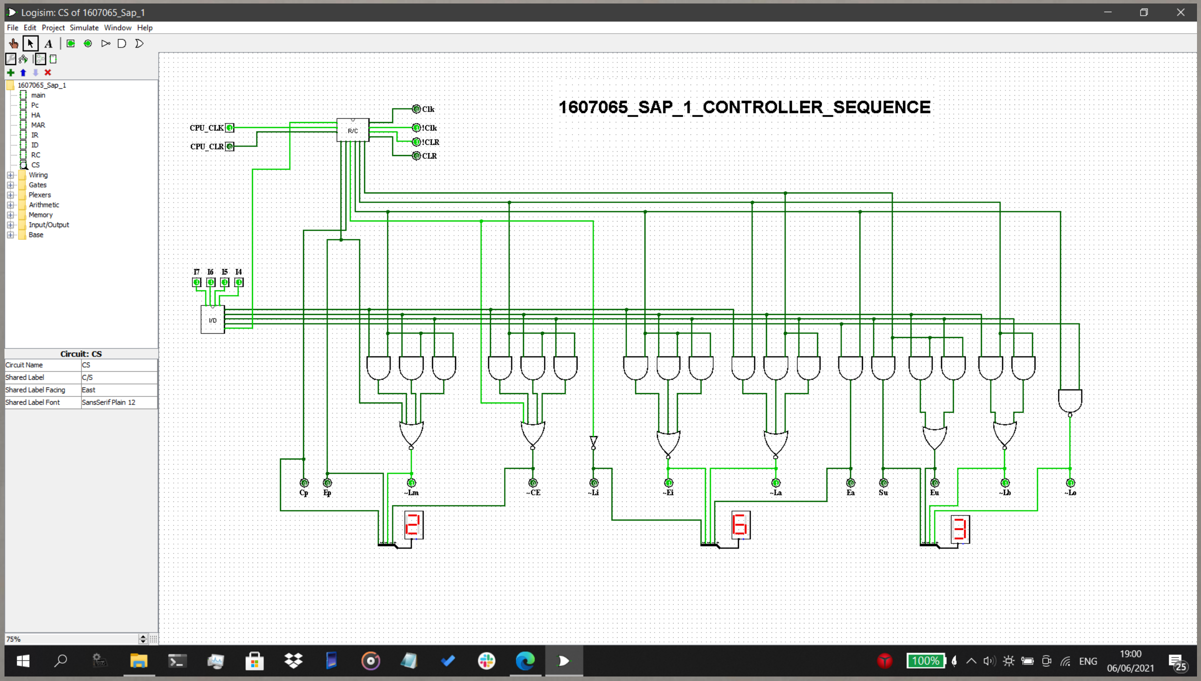
Circuit Name	Instruction Register
Shared Label	IR
Shared Label Facing	East
Shared Label Font	SansSerif Plain 12

1607065_SAP_1_INSTRUCTION_REGISTER

controller - sequencer: Before each computer run a $\overline{CLR}$ is sent to PC and a CLR signal to the IR. This resets the PC to 0000 and wipe out the last instruction to the IR, A clock signal CLK is sent to all buffer registers. The control-word has a format of

$$CON = C_P\ E_P\ \bar{L}_m\ \bar{C}_E\ \bar{L}_I\ \bar{E}_I\ \bar{L}_A\ E_A\ S_U\ E_U\ \bar{L}_B\ \bar{L}_O$$

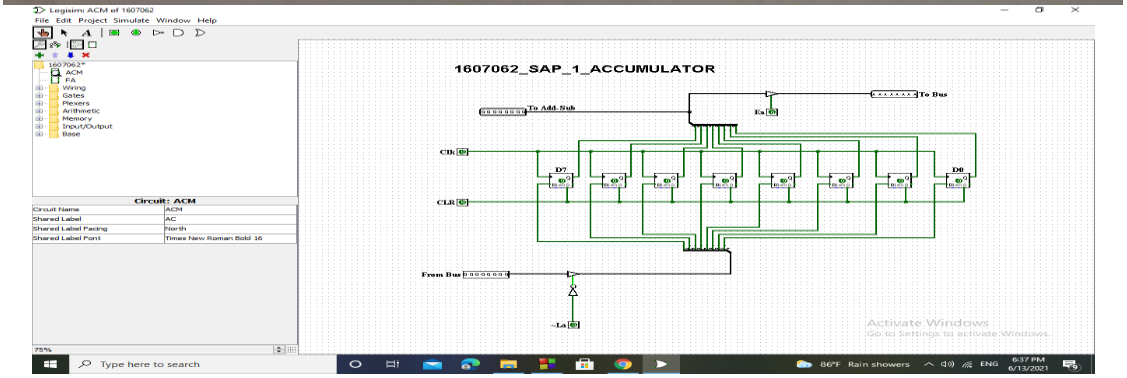
The word determines how the registers will to the next positive clock edge


Logisim: CS of 1607065_Sap_1

Circuit: CS

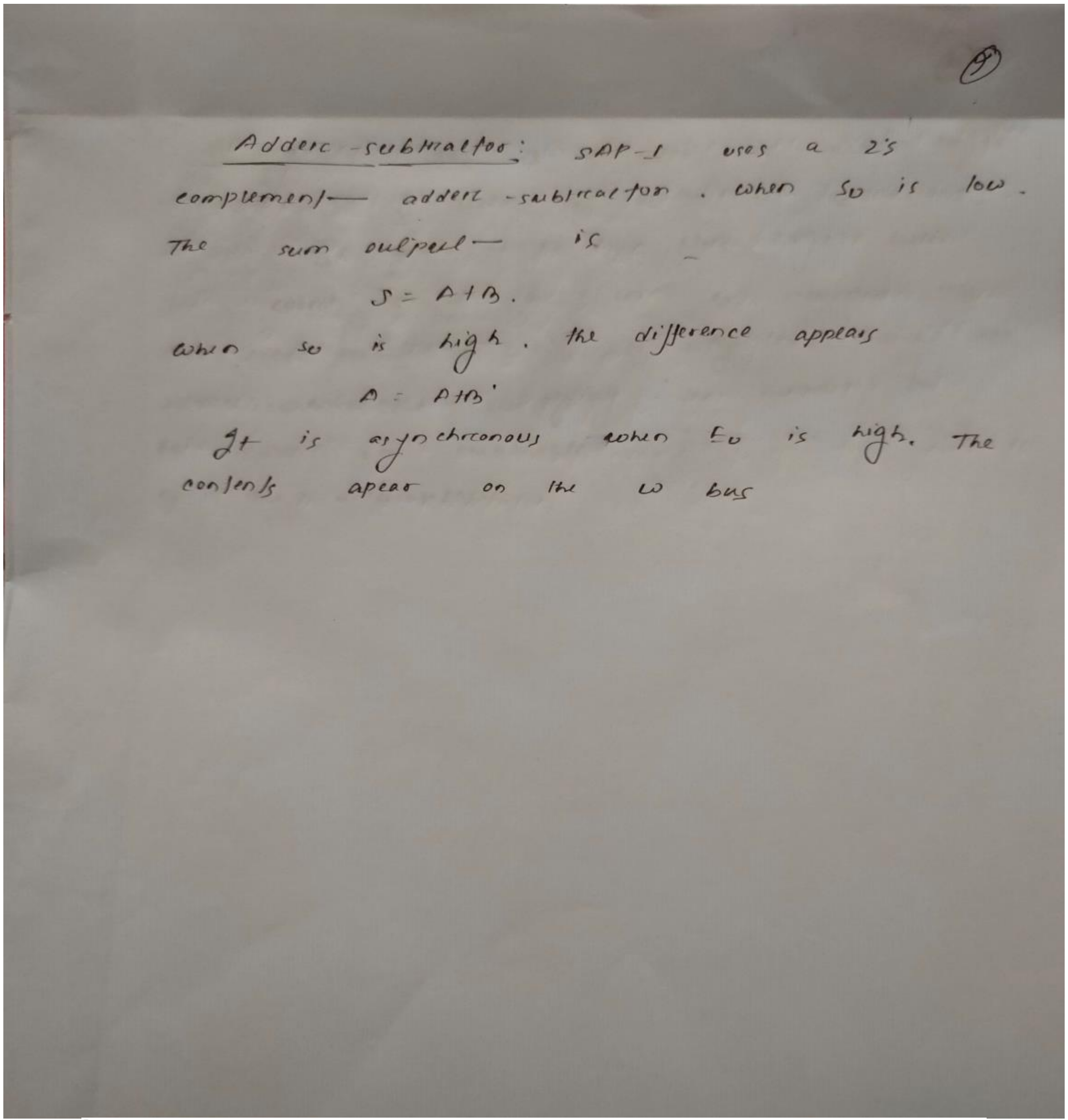
Circuit Name	CS
Shared Label	C/S
Shared Label Facing	East
Shared Label Font	SansSerif Plain 12

Accumulator: An accumulator (A) is a buffer register that stores intermidiate answer during a computer run. It has two outputs. The two state output goes directly to adder-subtractor. The three-state output goes to the w-bus. Therefore, the 8-bit accumulator word continously drives the adder-subtractor. the same word appears on w bus when Ep is high.



Circuit: ACM

Circuit Name	ACM
Shared Label	AC
Shared Label Facing	North
Shared Label Font	Times New Roman Bold 16



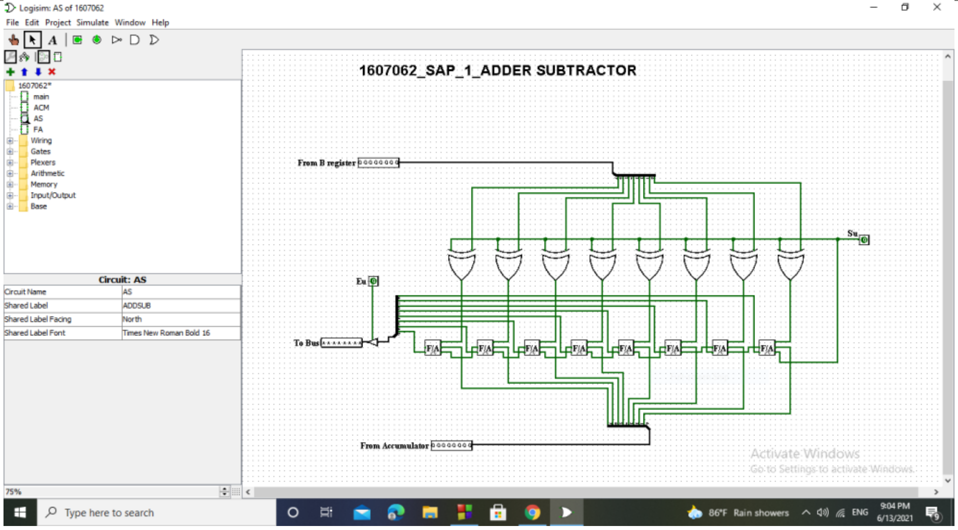
Adderc -subtraltor: SAP-1 uses a 2's complement— adderc -subtracltor. when Su is low. The sum outpeet— is

$$S = A+B.$$

when su is high. the difference appears

$$A = A+B'$$

It is asynchrconous when Eu is high. The contents apear on the w bus



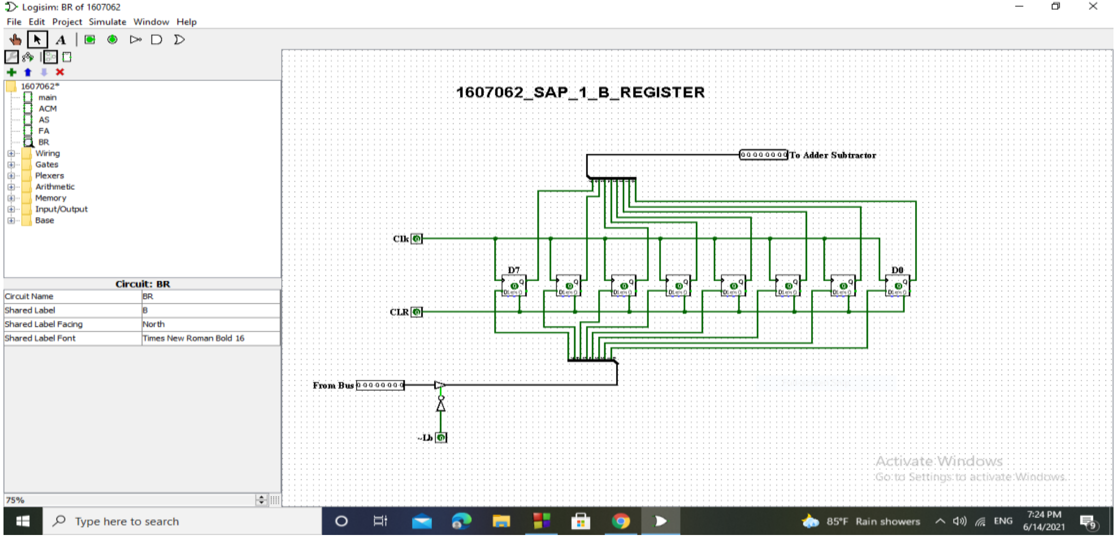
Logisim: AS of 1607062

1607062_SAP_1_ADDER SUBTRACTOR

Circuit: AS

Circuit Name	AS
Shared Label	ADDSUB
Shared Label Facing	North
Shared Label Font	Times New Roman Bold 16

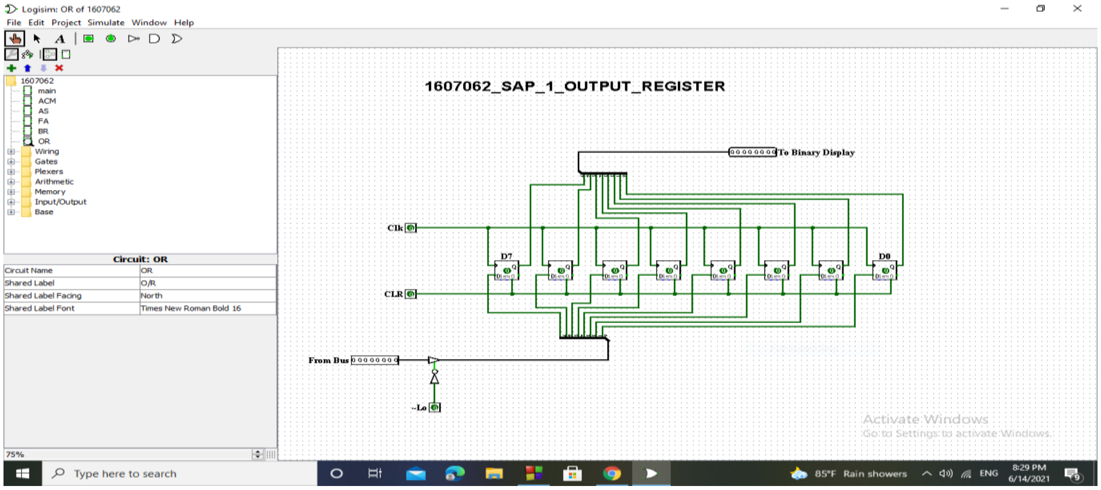
B register : The B register is another buffer register. It is used in arithmatic operations. A low $\overline{L_B}$ and positive clock edge load the word on W bus on B register. The two state output of B register drives the adder-subtractor, supplying the number to be added or subtracted from the contents of the accumulator.



Circuit: BR

Circuit Name	BR
Shared Label	B
Shared Label Facing	North
Shared Label Font	Times New Roman Bold 16

Output — register : At the end of — computer run. the accumulator contains the answer to the problem being solved. when E_A is high and $\bar{L}_o$ is low the next positive clock edge loads the accumulator word into the output — register.



Circuit: OR

Circuit Name	OR
Shared Label	O/R
Shared Label Facing	North
Shared Label Font	Times New Roman Bold 16

Binary display: The binary display is a row of eight LEDs. Because each LED connects to one flip-flop of the output port. the binary display shows the contents of the output port.

Instruction set of SAP-1:

The SAP-1 instruction set is given below:

1. **LDA**: LDA stands for "Load the accumulator". A complete LD'LED instruction includes the hexa-decimel adders of the data to be loaded

2. **ADD**: A complete ADD instruction includes the address of the word to be added. For instance add 10H means "add the contents of the memory location 10H to accumulator. replace the original contents of accumulator.

3. **SUB**: It includes the address of word to be subtracted. For example. SUB CH means, "subtract the contents of memory location CH from the contents of the accumulator.

4. OUT: It tells the SAP-1 computer to ~~transform~~ transfer the accumulator contents to the output port. Out is compute by itself..

5. HLT: HLT stands for HALT. It tells the computer to stop processing data HLT makes the end of the program. It must be used at the end of the SAP-1 program

Conclusion: By performing this project. SAP-1 computer design is learnt. we also learnt how to design every components of SAP-1 computer in logisim. we impemented SAP-1 computer in logisim using this component Finally. we ran a program in logisim and checked the output using LED and display module. So it can be said that, the project is done successfuly.