CSE 4224

1607098 Roll: 1607098, 1607032

Project Name: "SAP-1 computer design using logisim"

objectives:

The aim of this project are given below.

i) To know about SAP-1 computer.

ii) To know different functionality of logisim

iii) To know different components of SAP-1 computer

iv) 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 necessities for a functional Micro-Processor. Its primary purpose is to develop basic understanding of how a microprocessor works, interacts 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 towards modern computers.

Fig-1 shows the architecture of SAP-1, a bus

organised computer. All registers outputs to the W bus are three-state, this allows orderly transfer of data. All other register outputs are two state, these outputs continuously drive the boxes they are connected to.

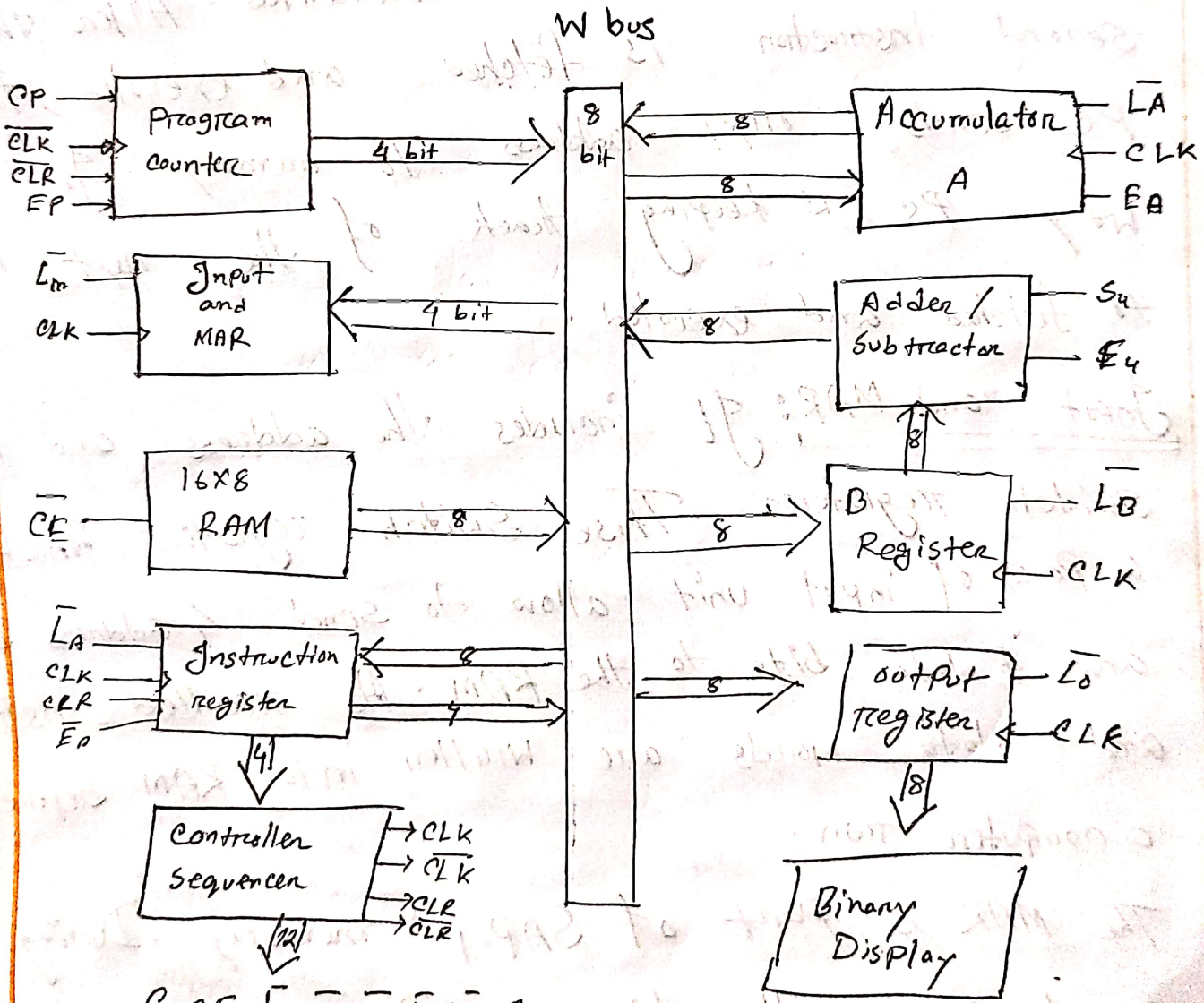


Fig: 1: SAP-1 architecture.

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 address 0001 to memory. Again the PC is incremented. After the second instruction is fetched and executed, the PC sends next address to memory. In this way, PC is keeping track of the next instruction to fetched and executed.

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. A bit later, the MAR applies this 4-bit address to RAM where a read operation is performed.

The 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 receives 4-bit address from MAR and a read operations 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 contents of the IR are split into two nibbles. The upper nibble is a two-state output that goes directly to the block labeled "controller sequencer". The other lower nibble is a three-state output that is read into W bus when needed.

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 wipes out the last instruction to the IR. A clock signal CLK is sent to all buffer registers. This synchronizes the operation of the computer, ensuring that all register transfer occur on the positive edge of common CLK signal.

The 12 bits that come out of controller from a word controlling the rest of the computer. The 12 wires carrying the control word are called the control bus.

The control word has the format of

$$CON = C_P\ E_P\ \overline{L_M}\ \overline{C_E}\ \ \overline{L_I}\ \overline{E_I}\ \overline{L_A}\ \overline{E_A}\ \ S_U\ E_U\ \overline{L_B}\ \overline{L_O}$$

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

Accumulator: A accumulator (A) is a buffer register that stores intermediate 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 EA is high.

The Adder Subtractor: SAP-1 uses a 2's complement adder-Subtractor. When S_u is low, the sum output of adder-Subtractor is

$$S = A+B$$

When S_u is high, the difference appears:

$$A = A+B'$$

It is asynchronous. When E_u is high the contents appear on the W bus.

B register: The B register is another buffer register. It is used in arithmetic operations. A Low $\overline{L_B}$ and positive clock edge load the word on W bus into B register drives the adder-Subtractor supplying the number to be added or Subtracted from the contents of the accumulator.

160 7032

Output Register:

At the end of computer num. the accumulator contains the answer to the problem being solved when E_A is high and L_o is low the next positive clock edge loads the accumulator word into the output register.

The output register is often called an output port because processed data can leave computer through this register.

Binary display:

The binary display is a row of eight LEDs. Because each LED connects to one flip flop of the output

160732

port, the binary display shows the contents of the output port.

Instruction set of SAP-1:

The SAP-1 instruction set follows:

① LDA: LDA stands for "Load the Accumulator" A complete LDA instruction includes the hexa decimal address of the data to be loaded.

② Add: A complete ADD instruction includes the address of the word to be added. for instance, 'ADD 9H' means: add the elements of memory location 9H to accumulator

160732

replace the original contents of the accumulator.

③ SUB

A complete sub instruction includes the address of word to be substracted for example SUB 9H means substract the contents of memory location 9H from the contents of the accumulator.

④ OUT:

The out instruction tells the SAP-1 computer to transfer the accumulator contents to the output port. OUT is complete by itself. you do not have to include an address when using out because the instruction doesn't involve

data in the memory

5. HLT:

HLT stands for Halt. It tells the computer to stop processing data. HLT makes the end of a program. You must use a HLT instruction at the end of every SAP-1 program.

Conclusion:

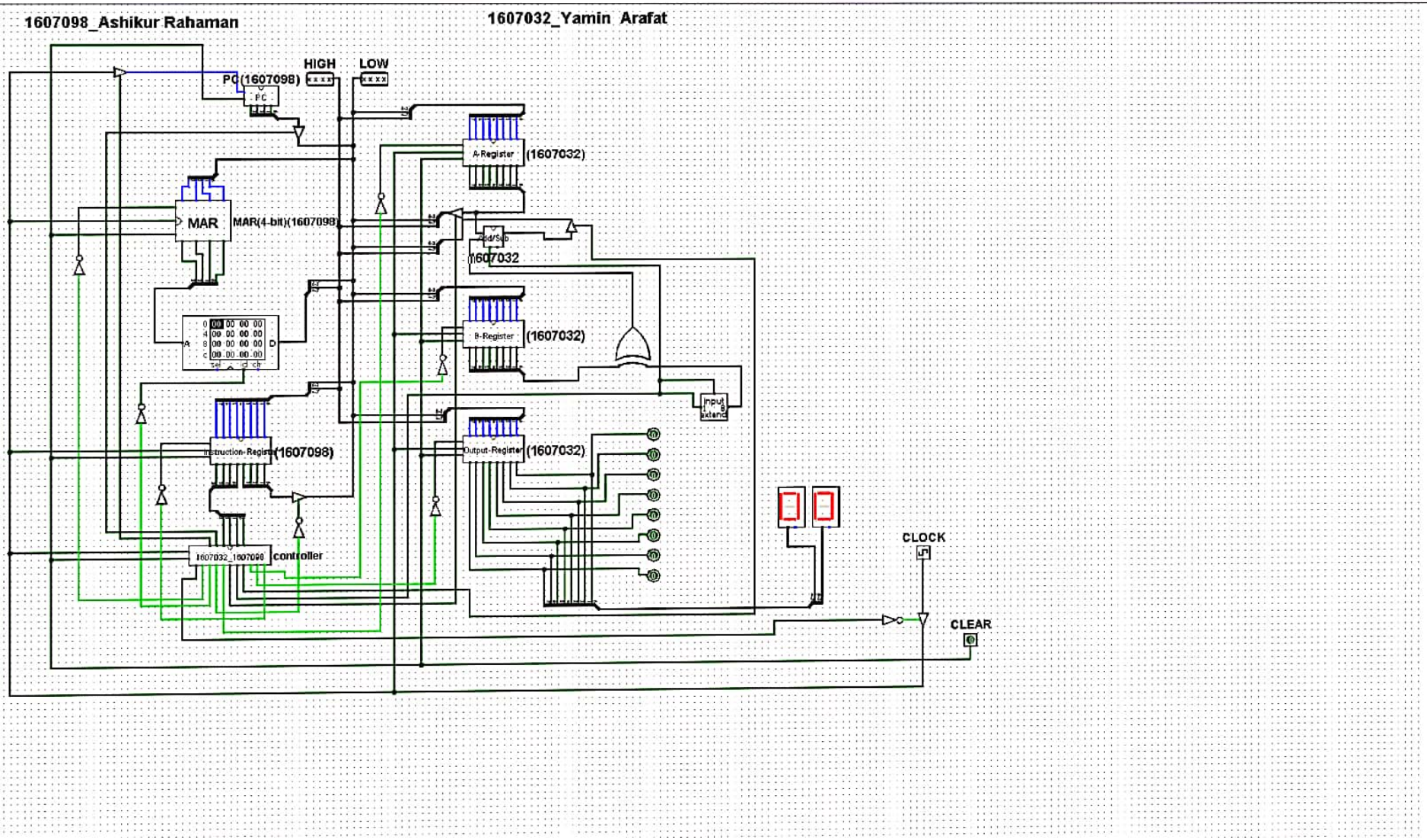
In this project we learnt about SAP-1 computer design process. We also learnt how to design every components of SAP-1 computer in logisim. We implemented

Logisim: main of 1607032_1607098

le Edit Project Simulate Window Help

1607032_1607098
main
MAR(4-bit)(1607098)
A-Register(1607032)
program counter(1607098) (synchronous)
Full Adder(1607032)
Adder subtractor(1607032)
controller(1607032_1607098)
ring counter(1607098)
instruction decode(1607032)
C39(1607098)
C40(1607098)
C41(1607098)
C43(1607098)
B-Register(1607032)
Output-Register(1607032)
Instruction-Registe(1607098)
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
Base

Circuit: main	
rcuit Name	main
ared Label	
ared Label Facing	East
ared Label Font	SansSerif Plain 12



160703 2

SAP-1 computer logisim using those components finally we ran a program in logisim and checked the output using LEDs and display module.

Discussion:

In this main project we learnt about a mini sap -1 computer. We combined all the knowledge we previously gatherd from the lab. We combined all the circuit designed individually and connected them to the main circuit. We faced different problem while combining all the circuit. Minor bugs were fixed and the SAP-1 runs perfectly.