ByteSquad

November 17, 2023

Phase-1
Digital Logic Design

Zehra Ahmed 26965

Anusha Randhawa 27095

Muhammad Haaris 27083

Hamna Inam Abro 27113

Hussain Ali Shah 27131

Abdul Nafay Indrawala 26951

Saqlain Kazmi 26869

Zara Masood 26928

Contents

0.1 Introduction:

The Simple As Possible-1 (SAP-1) represents an innovative effort to simplify complex digital systems into a clear format. It serves as a straightforward introduction to some of the essential concepts behind computer operations. SAP-1 marks the initial stage in the evolution of computers, encompassing all the basic elements required for a functional computer. Our primary belief is that the main goal of this project is to develop a basic understanding of how computers function. It aims to illustrate how the core components taught in Digital Logic Design come together to construct the computers we use today. Additionally, the project guides us in combining various components and understanding how a computer interacts with memory and other system parts, like input and output. In SAP-1, the entire instruction set is simple, limited to a few operations, as detailed in the report below.

0.2 Project Objectives:

- Develop a comprehensive understanding of the design and architecture of the Simple As Possible-1 (SAP-1) computer.

- Explore the intricacies of the control unit and grasp the synchronization mechanisms within a computer system.

- Demonstrate proficiency in showcasing the functionality of a computer system through the implementation of an instruction set using a microprocessor.

- Acquire hands-on experience in integration and programming within the Arduino Uno environment to enhance practical knowledge in digital and logic design.

0.3 Note:

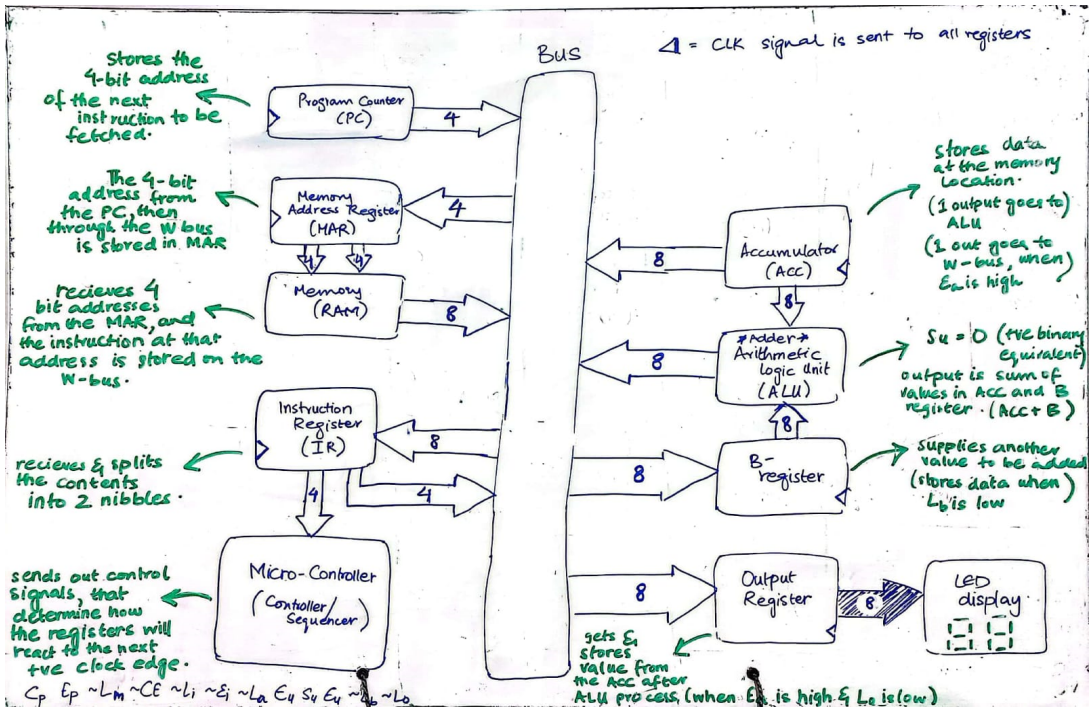
1. The Arithmetic Logic Unit (ALU) will focus on addition operations.
2. The SAP-1 model is designed to operate with 8 bits.

0.4 Overview:

SAP-1 is a bus-organized computer, with a minimal instruction set. This means it contains an 8 bit bus, which is connected to some components namely:

- 4 bit Program counter
- 4 bit MAR (Memory Address Register)
- 8 bit IR (Instruction Register)
- 8 bit Accumulator
- 8 bit B register
- 8 bit Output Register
- Microcontroller (manages the memory (RAM) and control unit)

SAP-1 computer makes use of the Von-Neumann Architecture.



0.5 Components:

0.5.1 Program Counter:

It is a kind of a register that contains the address of the next instruction to be fetched and executed. It counts up from 0 to 15, where 0 is 0000 and 15 is 1111. The program counter sends the address of the first instruction to the memory that is 0000, and is then incremented by 1 so that it points to the address of the next instruction to be fetched, provided that the instruction is stored in consecutive memory locations .The contents at the address contained in PC are then loaded into the Instruction Register and this is where the fetch-execute cycle begins. Every time the computer restarts, the program counter points to the memory address where the program begins.

0.5.2 Accumulator B-Register:

It is an 8-bit buffer register (small, fast storage element) within the computer's architecture, used to store and manage data that is actively being used or processed. It plays a central role in processing data, serving as a working register for intermediate calculations. It is where the results of arithmetic and logic operations are stored temporarily until they are transferred to another register or memory location.

The accumulator has two outputs:

1. A two-state output that goes directly to the adder-subtractor. The accumulator is used to continuously drive the adder-subtractor in ALU. It stores the output through the w-bus, when La BAR is low.

2. A three-state output that goes to the W bus through a tri-state buffer. Its value only appears on the W bus when Ea is high, which can then be read by the output register.

Operations such as Load, Add, Subtract, Output involve the accumulator.
An Accumulator operates on the principles of binary arithmetic.
An operand, which could be a value from memory or a register, is fetched and loaded into the accumulator. Arithmetic or logic operations are performed using the data in the accumulator. The result of the operation is then stored back in the accumulator. This new value in the accumulator can then be used in subsequent operations or transferred to another register or memory location.

The Accumulator and B register will be used to store the 8-bit data as input to ALU input, therefore it is important for the Accumulator and B register to be 8 bit as well.
The Accumulator has the following control bits:
Accumulator In (AI): stores the current value of the bus into Accumulator.
Accumulator Out (AO): Accumulator sends its stored content to the bus.
B Register In (BI) Stores the current values of the bus into B register

0.5.3 Memory Address Register:

The memory address register is used to handle the address of memory transferred to the memory unit. The program counter (PC) holds the address of the next instruction to be executed. The

content of the PC is transferred to the Memory Address Register (MAR) when an instruction fetch is initiated. The contents of the MAR are used to read data from a specific memory location. The MAR has a storage element, such as latches or flip-flops, to temporarily store the memory address during processing.

0.5.4 Memory Instruction register:

The 8 bit memory Instruction register is a crucial component.

It stores fetched instructions, facilitating their decoding and execution ,i.e. it holds the instructions currently being executed or decoded.When the Program counter fetches an instruction from the memory, it is loaded into the Instruction register.
The Instruction register typically holds:

1. the opcode
2. the address of the instruction.

Thus the contents of the Instruction register are split into two nibbles(half of 8-bit byte).

In simple terms, the instruction register serves as a temporary storage for the instruction being processed by the Control Unit. It allows the Control unit to decode the instruction, determine the necessary operations, and locate the required data in memory. The address portion of the instruction is used to specify the memory location of the data involved in the operation, while the opcode portion specifies the operation to be performed. This information is then used by the control unit to coordinate the execution of the instructions.

The components of the Control unit execute the instructions stored in the instruction register as per the requirement of the particular instruction's data value/values communicating via the w data bus.

Once the Execution is complete,the instruction register is updated with the next instruction to be executed.This is called the Instruction cycle.

0.5.5 Arithmetic Logic Unit:

Adder : The Adder asynchronously adds the value in the Accumulator (Register A) and the B register. Since this module is asynchronous, its contents can change as soon as the input changes. Among the many signals given out by the controller, Eu is one. When Eu is high, the contents of the Adder appear on the W bus.
A complete "Add" instruction includes the address of the value to be added . For example, ADD (mnemonic) 9H means "Add the contents of memory location 9H to the content in the Accumulator." This sum then replaces the original content of the Accumulator.

0.5.6 Microcontroller Processing (Controller/Sequencer Memory):

In a Simple As Possible (SAP) computer project, a microcontroller like Arduino can serve various roles: The Arduino can be used to simulate or construct the control unit of the SAP-1 computer.

It manages the operation of the computer by generating control signals to coordinate various components such as registers, ALU, memory, etc.
Using Arduino, you can simulate the functionalities of certain components within the SAP-1 computer using C Language. For instance, creating a memory, or arithmetic logic unit (ALU) can aid in testing and verifying the SAP-1's design and functionality.
Arduino in conjunction with the SAP-1 project allows for practical implementation, testing, and expansion of the basic computer architecture while providing a platform for learning and experimentation.

0.5.7 Output register:

Output register gets and stores the value stored in the accumulator usually after arithmetic operations. The transfer of the value stored in the accumulator to the output register is done through the W bus ,and happens in the next positive cycle. Further, it is connected to a 7 segment display through which the processed data is displayed to the outside world.

0.5.8 The instruction set:

The SAP-1 computer runs using an instruction set, which defines the basic operations that it can perform, the opcodes tell the computer about which operation is to be performed.:

INSTRUCTION (mnemonic + opcode)	**OPERATION**	**DESCRIPTION**
LDA (0000)	ACC = Memory [MAR]	Load data from the memory into the accumulator
ADD (0001)	ACC = ACC + B	Add data from the memory (B-register) to the data in the accumulator.
OUT (1110)	OUT = ACC	Load data from the accumulator to the output register.
HLT (1111)	CLK = 0	Halt processing. Marks the end of the program. Reset the clock.

0.6 Source:

digital-computer-electronics-albert-paul-malvino.pdf
https://karenok.github.io/SAP-1-Computer/
https://cellularnews.com/definitions/what-is-an-instruction-register-ir/
https://www.youtube.com/watch?v=kjdq1_l7cKw&list=PLk4sSiguONOW4v755N_O6Jk1WWrfWIGgm
https://slideplayer.com/slide/7275909/
https://www.slideshare.net/Jawad_Ahmad/sap-1-47400149