SAP-1 Digital Logic Design Project Report

GROUP NAME: AND-OR-NAUR

Group Members:

Lead	Abdullah Iqbal 26904
Co-Lead	Araiz Masood 26988
Member 1	Abdullah Ahmedani 25896
Member 2	Afnan Anwer Kiani 26900
Member 3	Adil Siddiqui 27147
Member 4	Ali Saeed 27153
Member 5	Ali Uzzama 26941
Member 6	Fatima Tanveer 25183

Contents

Chapter 1

Introduction

The Simple As Possible-1 (SAP-1) architecture is a keystone in the field of digital logic design, providing a basic investigation into the complexities of early computer design. It is based on the Von-Neuman Architecture. The SAP-1 framework is an innovative attempt to simplify intricate digital systems into a clear and instructive format. As future computer scientists and engineers, we set out to explore SAP-1 by dissecting its design, comprehending its instruction set, and running simulations of its functions.

The SAP-1 processor offers a perfect platform for understanding the fundamentals of digital computer design. With this project, we hope to demonstrate our competence in the real-world use of the SAP-1 architecture in addition to gaining an understanding of its internal workings.

1.1 Objectives

The primary objective of our SAP-1 digital logic design project is to gain a comprehensive understanding of the Simple As Possible-1 architecture and its implications in the realm of digital systems. Through a systematic exploration of SAP-1, we aim to achieve the following key objectives.

1. Assembly Language Programming: Demonstrate proficiency in SAP-1 assembly language programming. Craft and execute assembly language programs, showcasing the ability to manipulate data, perform arithmetic operations, and control the flow of execution within the SAP-1 architecture.

2. Simulation and Verification: Utilize Wokwi as a tool to model the SAP-1 processor's behavior and verify its correctness. Evaluate the outcomes of various programs and instructions, ensuring that the simulated results align with the expected behavior based on the architecture and instruction set.

3. Documentation and Reporting: Compile a comprehensive report documenting our exploration of SAP-1. This includes the project's background, the implementation of SAP-1 using Arduino Uno, simulation results, and a reflective analysis of the insights gained throughout the process.

1.2 Project Scope

The scope of the project encompasses the following phases:

1. SAP-1 Architecture Concept and initiation:

    - Develop a detailed understanding of the SAP-1 processor's architecture, including its main components such as the instruction register, arithmetic and logic unit (ALU), memory unit, and control unit.
    - Create a comprehensive block diagram illustrating the interconnections and functionalities of each module within the SAP-1 architecture.

2. Assembly Language Programming: Demonstrate proficiency in SAP-1 assembly language programming. Design and execute a set of assembly language programs that showcase the capabilities of the SAP-1 architecture in manipulating data, performing arithmetic operations, and controlling program flow.

3. Simulation and Verification: Using Arduino Uno as a controller on Wokwi to model the behavior of the SAP-1 processor. Verify the correctness of the architecture and instruction set by simulating the execution of various programs and comparing the results with expected outcomes.

4. Documentation and Reporting:

    - Compile a comprehensive project report that covers the background, objectives, methodology, and outcomes of the SAP-1 exploration.
    - Provide a detailed account of the challenges encountered during the project, along with reflective insights gained from the hands-on implementation and simulation of the SAP-1 architecture.

5. Presentation: Prepare and deliver a presentation summarizing the key findings and outcomes of the SAP-1 digital logic design project. This includes a walkthrough of the architecture, instruction set, assembly language programs, simulation results.

Through these objectives, the project seeks to bridge the gap between classroom learning and real-world application in the field of digital systems design.

Chapter 2

SAP-1 Concept and Initiation using Wokwi

2.1 SAP-1 Architecture

1) Program Counter: The program counter stores and sends out the memory address of the instructions to be collected and executed. It functions similarly to a pointer register, in the sense that it points to the next instructions address, i.e., the instruction that needs to be executed. The program counter starts counting from 0000 to 1111, signifying all the addresses present.

3) Random-Access Memory (RAM): By using the address and data switch registers, we can program the RAM. This enables us to load data and programs into memory prior to a computer running.

4) Instruction Register: The instruction register, temporarily, stores the instructions passed on by the RAM or the controller if not using RAM. After storing the instruction, it processes and divides the instruction into two nibbles(four-bit units), with each nibble serving a specific purpose in the instruction execution process.

5) Accumulator: It is a 8-bit buffer register that mainly stores the intermediate results while a particular instruction is being executed. It has two outputs: a two-state output and a three-state output. The two state output is directly passed to the Adder for arithmetic operations while the three-state

output goes to the bus. The conditional placement of the accumulator contents on the W bus is controlled by a control signal that decides whether the contents of the accumulator can or cannot be available on the bus.

6) Adder: It ,independently, performs addition or subtraction between the outputs from the accumulator and the arduino controller if not using RAM else uses the contents of RAM (operand) to perform addition.

7) B-Register: It is a buffer register that provides the value, that is to be added or subtracted in the output of the accumulator, in the adder-subtractor. It can receive and store data from the bus using the "Lb" signal. When the Lb signal is low and there is data available on the bus, the B-Register receives and stores that data.The synchronization with the clock edge ensures proper and reliable data transfer.

2.2 Simulation and Assembly Programming

SAP-1, which stands for Simple As Possible-1, is the initial stage in computer architecture learning that introduces the core concepts of computer operations. SAP-1 utilises a Von-Neumann architecture and works with 8-bit data, which means it handles information in small, 8-bit pieces. The SAP-1 processor operates on a simple instruction cycle:

1. Fetch: The instruction is fetched from memory into the Instruction.
2. Decode: The control unit decodes the instruction, determining the operation to be performed.
3. Execute: The ALU performs the specified operation, and the result is stored back in the accumulator or transferred to memory.

The key parts of SAP-1 include an Arithmetic Logic Unit (ALU) that does basic math and logic operations, a control unit that directs what the processor does, and some registers which are like small storage spaces for data. The way SAP-1 works is pretty straightforward: program counter holds the address of next memory location to fetched. The instruction at the address is divided into two parts opcode and operand. opcode is the instruction to be decoded and operand is the data value. Instruction register holds opcode which figures

out what these instructions mean, and directs what the instructions say. This process helps in understanding the basic steps a computer goes through to carry out commands. By breaking down these steps, SAP-1 makes it easier to understand how more complex computers work later on.

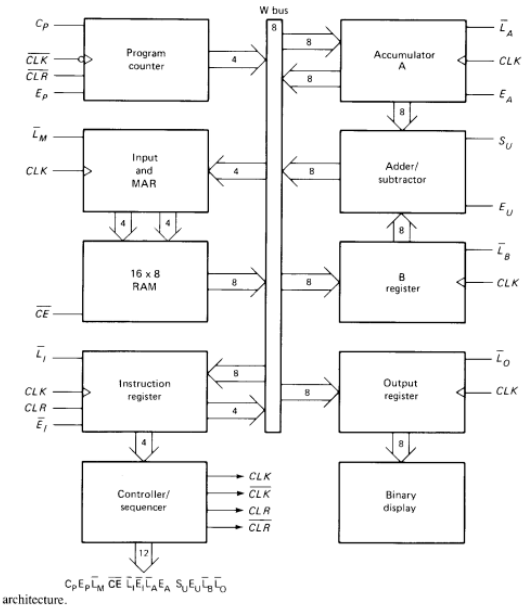

Fig. 10-1 SAP-1 architecture.

Figure 2.1: SAP-1 Architecture

Programming in SAP-1 Assembly Language: SAP-1 assembly language is relatively simple, consisting of instructions like LOAD (LDA), ADD, SUB, STORE (STA), and Halt (HLT). Programs are written using mnemonic codes, making it an accessible platform for learning assembly language programming.

Bibliography

[1] https://deeprajbhujel.blogspot.com/2015/12/sap-1-architecture.html.

[2] https://prezi.com/fv-ygb2jxu$_m$/$sap1 - architecture - instruction - set$/.

[3] https://dangrie158.github.io/SAP-1/logical-blocks/mar.html.

[1] [2] [3]