simge, sembol, yazı tipi, amblem, ticari marka içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, yazı tipi, logo, simge, sembol içeren bir resim

Açıklama otomatik olarak oluşturuldu

Marmara University

Faculty of Engineering

**CSE 315 DIGITAL LOGIC DESIGN TERM PROJECT**

|  |  |  |
| --- | --- | --- |
|  | Student Id Number | Name & Surname |
| 1 | 150120995 | Seyyid Ali Koldaş |
| 2 | 150120031 | Şükrü Can Mayda |
| 3 | 150121545 | Nurbetül Çakır |

Table of Contents

[1. INTRODUCTION -2 -](#_Toc61992405)

[2. ASSEMBLER - 3 -](#_Toc61992406)

[3. LOGISIM DESIGN - 4 -](#_Toc61992408)

[4. VERILOG DESIGN - 8 -](#_Toc61992408)

# Introduction

The objective of this project is to design and implement a processor with a specific instruction set architecture, supporting a range of instructions such as ADD, AND, NAND, NOR, ADDI, ANDI, LD, ST, CMP, JUMP, JE, JA, JB, JAE, and JBE.

This report dissects the design process of a processor, tracing its path from the blueprint of its instruction set architecture (ISA) to its final Verilog HDL implementation. Key components like registers, memory units, the control unit, and the ALU are examined in detail, illuminating the inner workings of this digital workhorse.metin, sayı, numara, paralel, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Instruction Format:**

**Opcode (4 bits):** Identifies the operation to be performed.

**Register (4 bits):** Specifies the registers involved in the operation.

**Immediate Value (6 bits):** Used in instructions involving immediate values.

**Address (10 bits)**

# Assembler

An assembler has been developed to convert human-readable assembly code into machine code suitable for the processor.

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

The assembler output consists of 5-digit hexadecimal numbers for each instruction, representing the 18-bit machine code.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Logisim Design

**Datapath Components:**

**Register File:** Register File contains 16 general purpose registers. Each register has a write signal, and these signals control writing data to a register. Read operations are triggered by a signal to read te contents of a particular register.

metin, diyagram, plan, şematik içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Instruction Memory:** Instruction Memory is a unit of memory where instructions are stored. Each instruction contains the opcode and associated register addresses. The PC (Program Counter) value is read incrementally from this memory.

diyagram, metin, çizgi, plan içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Data Memory:** Data Memory is a memory unit where data is stored. Data writing and reading operations are triggered by a specific address. The write-enable signal controls the writing of data to this address.

**Main:** When the clock signal arrives, the controller reads an instruction from memory. The instruction is interpreted and the necessary signals are generated. The ALU performs the arithmetic or logic operation specified in the instructions. The result is saved in the data memory. These processes ensure that our project's instructions are executed in order and correctly.

The main part of our Logisim project is an important part that ensures the basic functioning of our project. In this section, there are the basic components of our project such as the control unit, arithmetic logic unit (ALU), and data memory. The main part uses these components to execute instructions sequentially and correctly in each loop.

diyagram, plan, teknik çizim, şematik içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Control Unit:** It is a critical unit that controls and manages the functioning of other components. It is responsible for decoding commands and generating control signals. It manages the state of the processor and determines control signals based on the state. It produces signals that tell other components such as ALU, Register File, Data Memory what kind of operations they should perform. The Control Unit undertakes the correct processing and control of the program by ensuring the coordination of other components within the CPU.

diyagram, plan, teknik çizim, şematik içeren bir resim

Açıklama otomatik olarak oluşturuldu

**ALU (Arithmetic Logic Unit):** ALU is an important component that performs certain arithmetic and logical operations. The ALU performs the selected operation based on the opcode and writes the result to a register.

diyagram, plan, teknik çizim, şematik içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Half Adder:** A digital circuit that performs binary addition on two inputs, producing a sum and a carry-out.

diyagram, çizgi, plan, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Full Adder**: A more advanced digital circuit than a half adder, capable of adding three binary inputs with carry-in and producing a sum and carry-out.

diyagram, plan, teknik çizim, şematik içeren bir resim

Açıklama otomatik olarak oluşturuldu

**18-Bit Register:** A component that extends the sign bit of a binary number to a larger bit-size while maintaining the original value's sign in signed number representation.

diyagram, çizgi, plan, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Sign-Bit Extender:** A digital storage element in a computer's processor that stores binary data and can be used for various operations, such as arithmetic and logical calculations.

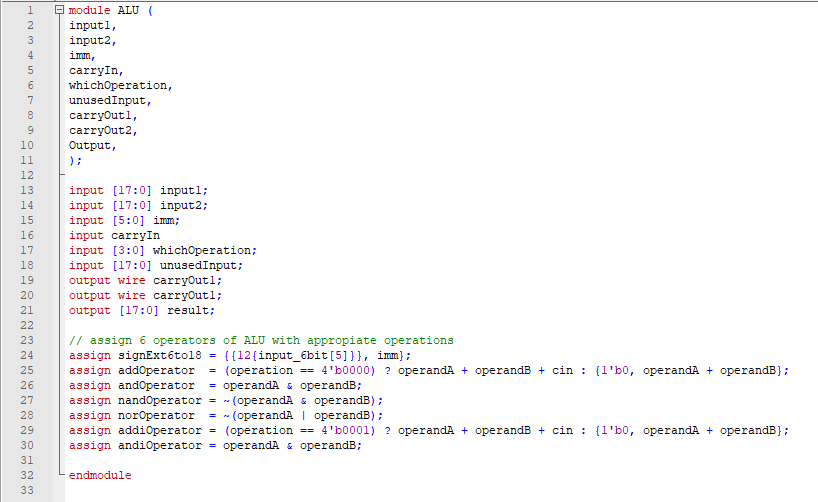
ekran görüntüsü, diyagram, metin, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

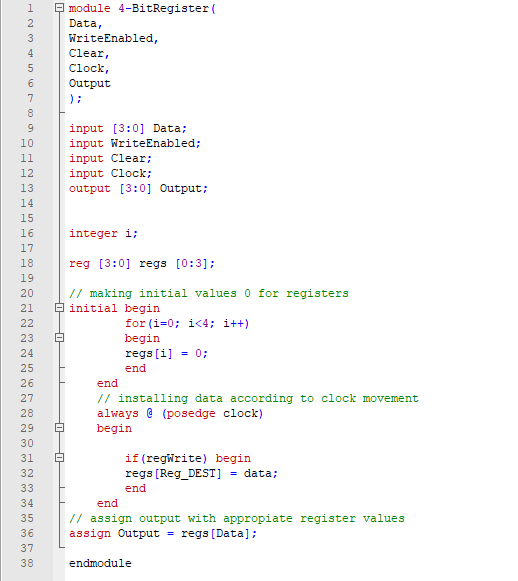
# Verilog Design

This part, we need to implement our design with Verilog language, based on our Logisim design.

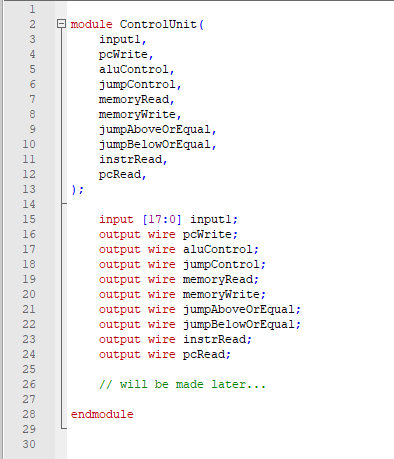
**ALU:**



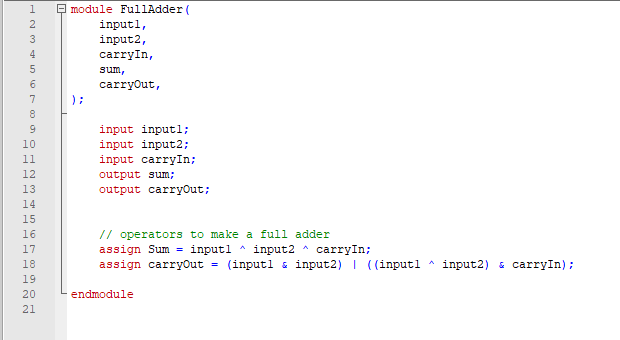
**4-Bit Register:**

****

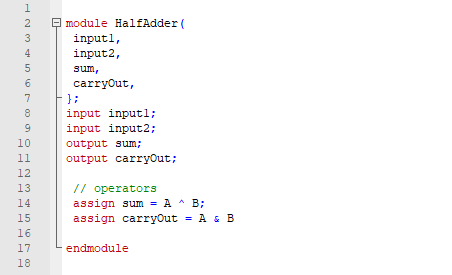
**Control Unit:**

****

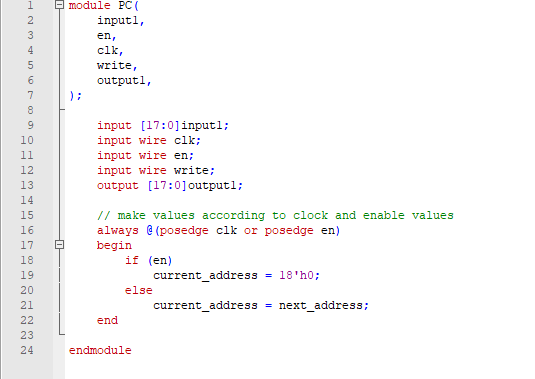
**Full Adder:**

****

**Half Adder:**

****

**PC:**



**Register File:**

