KUZEY KIBRIS KAMPÜSÜ
ODTÜ
METU
NORTHERN CYPRUS CAMPUS

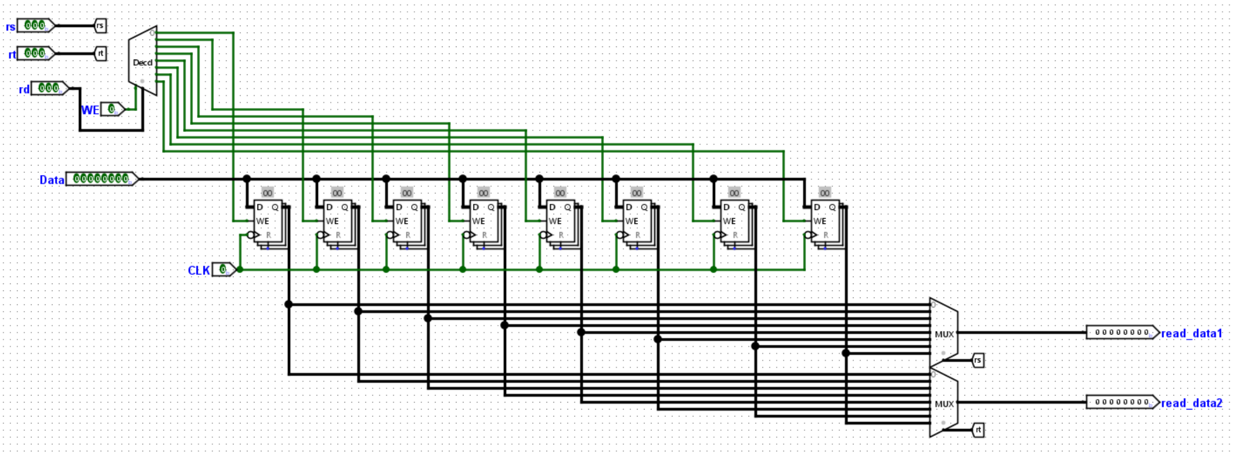
TERM PROJECT
#2 REPORT

Berke Diler

2401503

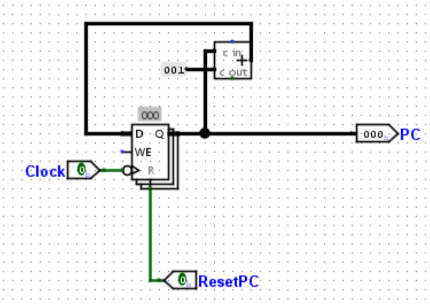
The Register File

The Schematic:



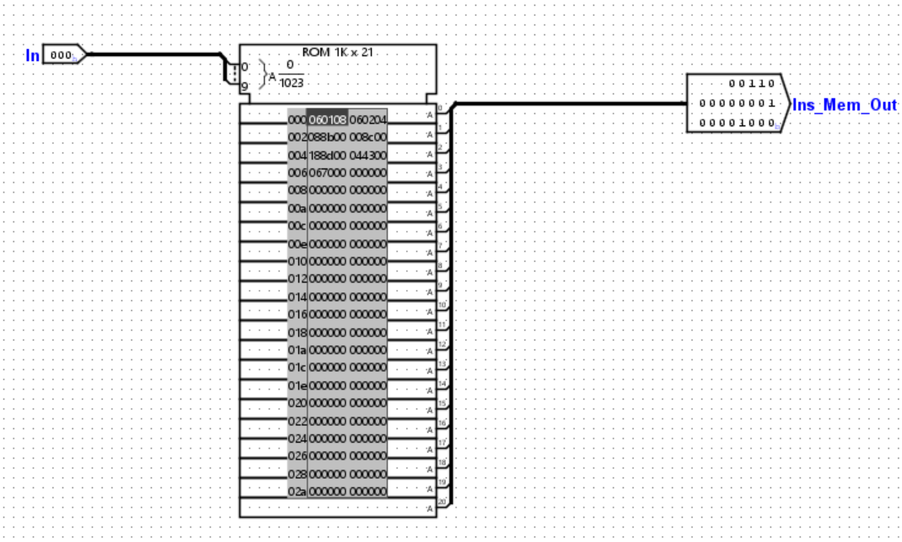
The register file is the component in the CPU that is responsible for getting the rs,rt,rd,WE,Data as inputs. And it either reads data from its registers or writes data to its registers according to to the inputs. If WE is not 1, then the register file reads from its registers and decides where to read the data from the rs and rt inputs which are connected to the output MUXes. If WE is 1, then the register file writes to the desired register which is decided by the input rd and the data that is written to the register is the Data input.

Program Counter (PC)



The program counter is a component in the CPU which ‘tells’ the Instruction Memory which instruction to take from the ROM each clock cycle. In this part of the project, we are incrementing the Program Counter by 1 in each clock cycle.

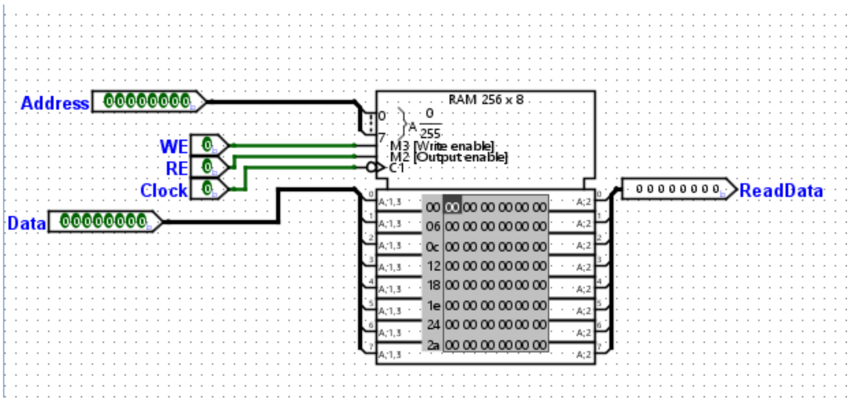
Instruction Memory



```
000 060108 060204
002 088b00 008c00
004 188d00 044300
006 067000 000000
008 000000 000000
00a 000000 000000
00c 000000 000000
00e 000000 000000
010 000000 000000
012 000000 000000
014 000000 000000
016 000000 000000
018 000000 000000
01a 000000 000000
01c 000000 000000
01e 000000 000000
020 000000 000000
022 000000 000000
024 000000 000000
026 000000 000000
028 000000 000000
02a 000000 000000
```

Instruction Memory is the part that is responsible of storing the instructions for the CPU which is built using a ROM, in each clock cycle the Instruction Memory decides which instruction to take according to the value that it recieves from the Program Counter (PC), in this specific project, each cycle the program counter increases by 1. So in every clock cycle, we take the next instruction. But if we had the jump function in our CPU, we might have taken the instruction that has been jumped to.

Data Memory



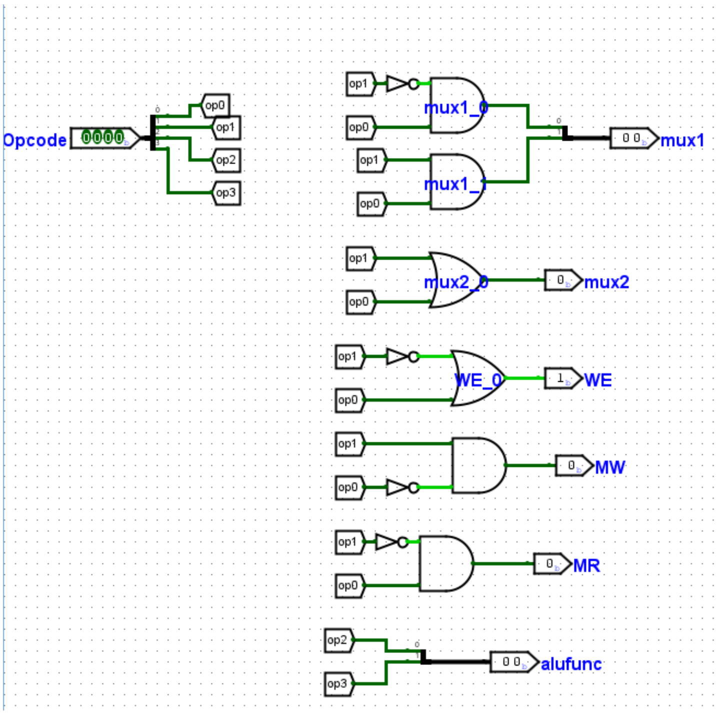
```
00 00 00 00 00 00 00
06 00 00 00 00 00 00
0c 00 00 00 00 00 00
12 00 00 00 00 00 00
18 00 00 00 00 00 00
1e 00 00 00 00 00 00
24 00 00 00 00 00 00
2a 00 00 00 00 00 00
```

Data Memory is a part of the CPU that is responsible for storing values to the RAM using WE signal so that these values can be fetched using RE signal. The Data Memory gets the data to store from the 'Data' input and the address to store or read the value from the 'Address' input. When RE signal is 1, we get the output from the 'ReadData'.

Difference Between RAM and ROM

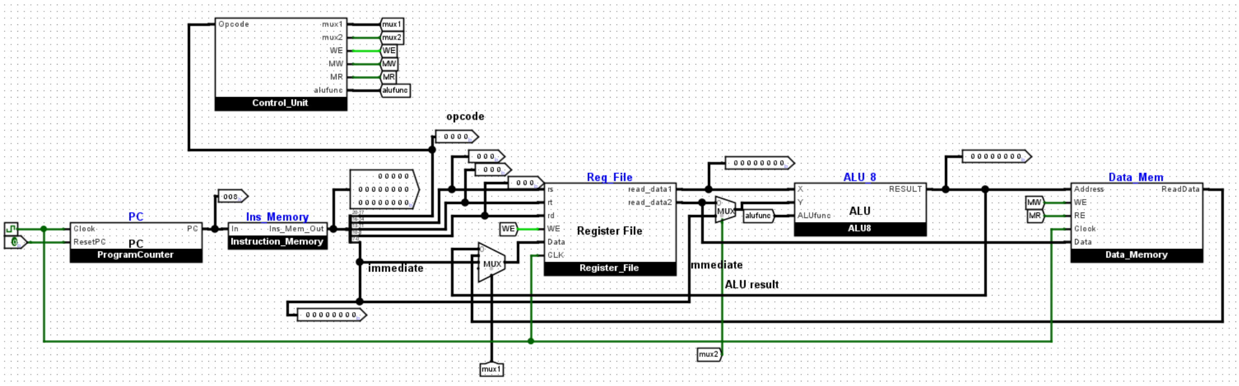
The main difference between RAM and ROM is that, RAM is a volatile memory and ROM is a non-volatile memory. Which means that RAM can store values as long as its powered but ROM can store instructions whether it is powered or not.

Control Unit

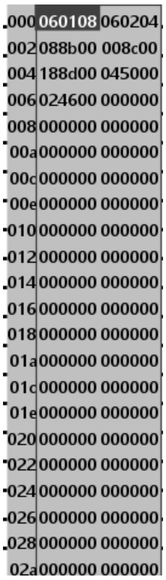


The control unit is the component in the CPU that takes the opcode of the given instruction and decides what operation is going to be done in the CPU. It sends signals to decide which value should a MUX give as the output, write to memory, write enable, read from memory and sends ALU the alufunc which tells the ALU which arithmetic or logic operation to implement.

Whole Schematic



The Test Part

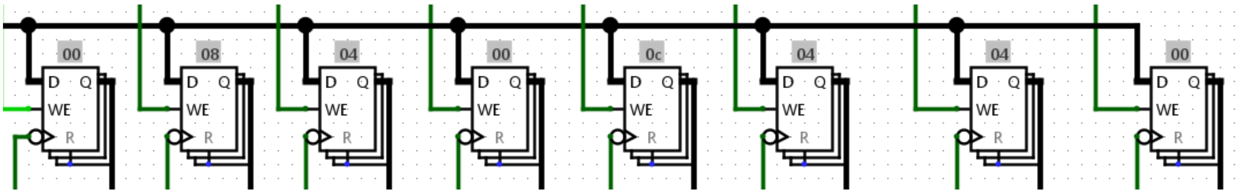


```
000 060108 060204
002 088b00 008c00
004 188d00 045000
006 024600 000000
008 000000 000000
00a 000000 000000
00c 000000 000000
00e 000000 000000
010 000000 000000
012 000000 000000
014 000000 000000
016 000000 000000
018 000000 000000
01a 000000 000000
01c 000000 000000
01e 000000 000000
020 000000 000000
022 000000 000000
024 000000 000000
026 000000 000000
028 000000 000000
02a 000000 000000
```

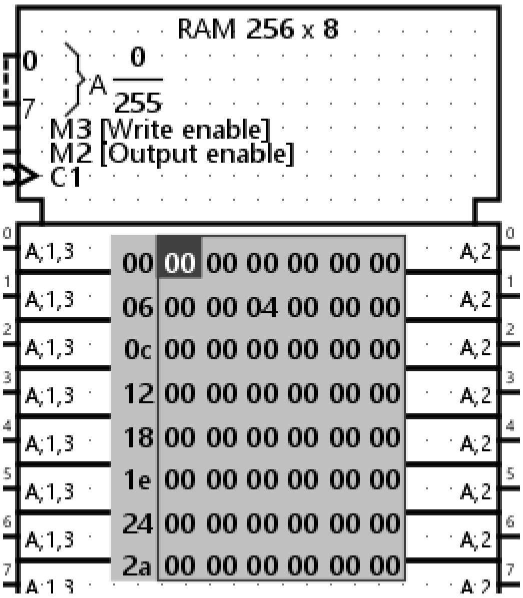
The figure shows the instructions inside the instruction memory after loading the image file given in ODTUClass.

The given instructions are converted to assembly as the following:
LI $001, IMM(0000 1000)
LI $010, IMM(0000 0100)
AND $011, $010, $001
ADD $100, $010, $001
ROTL $101, $010, $001
ST $010, IMM(0000 0100)
LD $110, IMM(0000 0100)

After loading the given image file and running the test, the register file registers resulted as following:



And the data memory resulted as the following:



```
00 00 00 00 00 00 00
06 00 00 04 00 00 00
0c 00 00 00 00 00 00
12 00 00 00 00 00 00
18 00 00 00 00 00 00
1e 00 00 00 00 00 00
24 00 00 00 00 00 00
2a 00 00 00 00 00 00
```