MR. Chips
16-Bit MIPS CPU

Meghan Notkin (Student ID: 0161059) - *EE*
Anna Weisman (Student ID: 0167481) - *EE*

Design Explanation

The 16-bit MIPS architecture CPU is designed to execute instructions efficiently and quickly. The overall design consists of a control unit, arithmetic logic unit (ALU), registers, and memory. The CPU fetches instructions from memory and decodes them to determine the operation to be performed. The control unit generates signals to coordinate the various components of the CPU to execute the instruction. The ALU performs arithmetic and logic operations on the operands, and the result is stored in the destination register. The registers hold operands and results of instructions temporarily, and the memory stores instructions for the CPU to access.

This CPU uses 16-bit instructions and its ALU operands are 16-bits as well.

The following principals were kept in mind when designing the CPU.

- Smaller is faster.
- Simplicity favors regularity.
- Make the common case fast.
- Good design demands good compromise.

0.1 Design Diagrams

The following diagrams illustrate the components of the 16-bit MIPS architecture CPU. Figure 1 illustrates the process the CPU undergoes during each instruction cycle.

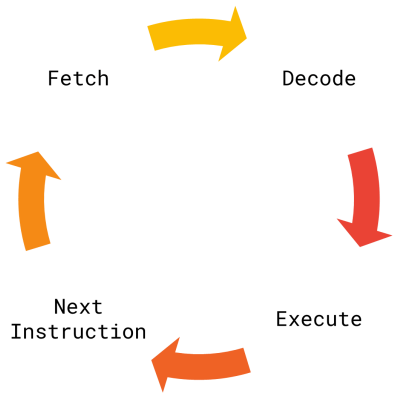

Figure 1: functions of a single instruction cycle

Figure 2 is a simplified block diagram of the CPU.

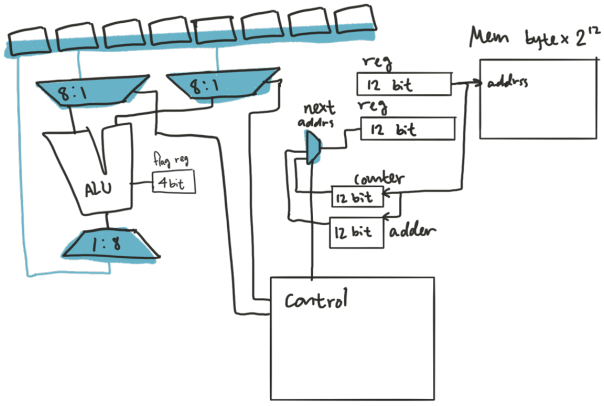

Figure 2: Functional Block Diagram

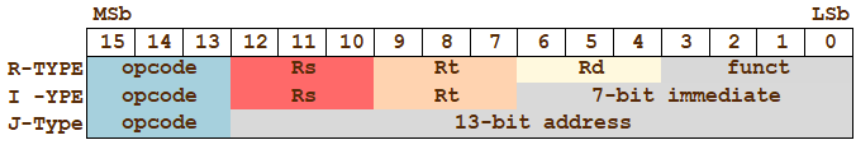

Figure 3: 16-bit instruction formats

R-type timing diagram:

The R-type instruction format is used for instructions that involve register operands and arithmetic or logical operations. The following timing diagram shows the various signals generated during the execution of an R-type instruction:

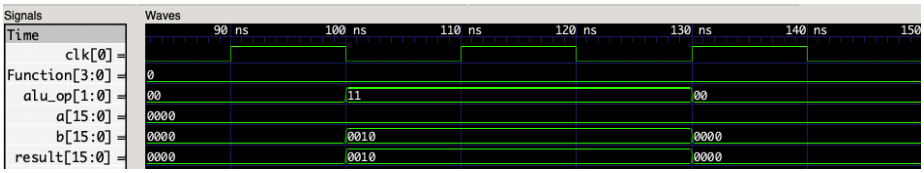

Figure 4: R-type instruction timing diagram

I-type timing diagram:

The I-type instruction format is used for instructions that involve immediate values and memory operations. The following timing diagram shows the various signals generated during the execution of an I-type instruction:

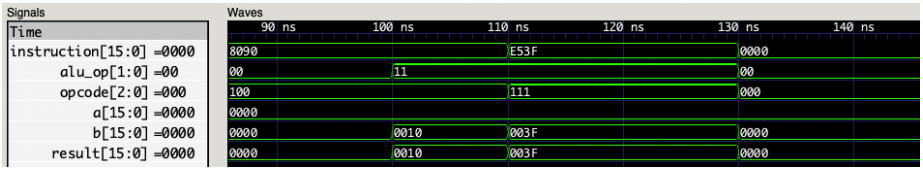

Figure 5: I-type instruction timing diagram

J-type timing diagram:

The J-type instruction format is used for jump instructions. The following timing diagram shows the various signals generated during the execution of a J-type instruction:

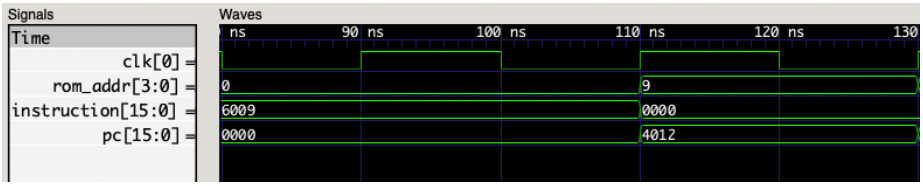

Figure 6: J-type instruction timing diagram

Here is an example program that implements the Fibonacci sequence using recursion:

```
1    1000000010000000    lw $1,0
2    1110000100000001    addi $2, $0, 1
3    0010101000100010    loop: stli $4,$1,21
4    1101000001011111    beq $4,$0,break
5    0000010100110000    add $3, $1, $2
6    0000010000100000    add $2, $1, $0
7    0000110000010000    add $1, $3, $0
8    1010101010000000    sw $2
9    1100000000000010    beq $0,$0,loop
10   0000000000000000    break
11   0000000000000000
12   0000000000000000
```

Figure 7: Machine language (left) and assembly code (right)

The blue waves in the following images show the output of the program.

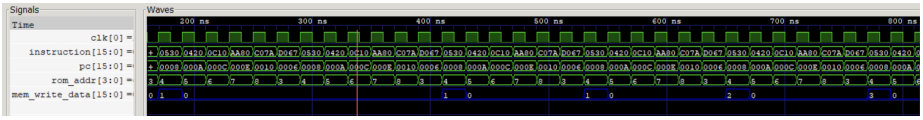

Figure 8: Snippet of implementation of Fibonacci

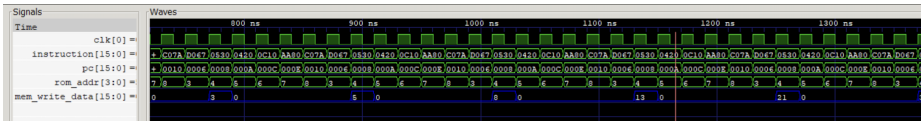

Figure 9: Second snippet of implementation of Fibonacci

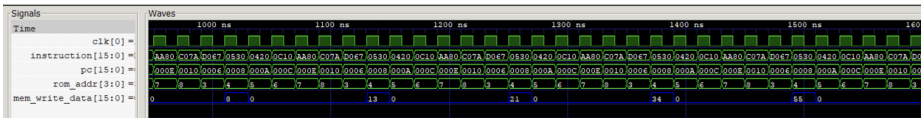

Figure 10: Third snippet of implementation of Fibonacci

References

D. A. Patterson and J. L. Hennessy, Computer Organization and Design MIPS Edition the Hardware/Software Interface. Morgan Kaufmann, 2021.