CPU architecture

CPU

8bit switch input

REGistry memory

8Bytes

8bit word

RAM memory

256Bytes

8bit word

8bit LED output

Arithmetic

Logic

Unit

Control Unit

SOFT memory

8KBytes

16bit instruction

CPU modules

1. Arithmetic Logic Unit (ALU)
2. Registers (REG)
3. Random Access Memory (RAM)
4. Control Unit (CU)
5. Software (SOFT)
6. Central Processing Unit (CPU)

**The Arithmetic Logic Unit (ALU module)**

The purpose of the ALU module is to compute arithmetic operations: addition, subtraction, multiplication and division;

Also it computes logical operations:

bitwise and, bitwise or, bitwise xor, bitwise not, equality and comparison.

The inputs are two 8bit registers and its output is an 8bit word.

**Registers (REG module)**

There are 8 8bit word registers that can be randomly accessed for either reading or writing. This module is addressable by a 3bit address.

**Random access memory (RAM module)**

The RAM is formatted as 256 8bit words addressable by an 8bit input, excluding one dedicated address, a total of 255 usable words. The last address is used for having memory for the 8 LED output built on the Xilinx Zedboard.

**Control Unit (CU module)**

The control unit has a 12bit counter that represents the instruction line which gets incremented unless a valid JMP instruction is found. The whole purpose of the CU is to separate the current instruction in usable flags and addresses.

**Software (SOFT module)**

The SOFT module is the programmable memory of the CPU, having a total of 8KBytes divided in 16bit instructions. It is randomly addressable by 12bit addresses and it outputs the respective 16bit instruction.

**Central Processing Unit (CPU module)**

The CPU module brings all of the above modules alike the architecture figure. Its inputs are a CLK input and 8bit control, i.e. the 8 switches built on the Xilinx Zedboard. This module outputs 8bits intended for the 8 LEDs built on the Zedboard.

Instruction set

xxxxyyy\*zzzzzzzz – arithmetic instruction format

xxxx – the arithmetic operation

0000: addition ADD

0001: subtraction SUB

0010: multiplication MUL

0011: division DIV

0100: equality EQ

0101: A greater than B GRT

0110: bitwise and AND

0111: bitwise or OR

1000: bitwise xor XOR

1001: bitwise not NOT

yyy – address of the registry (term A)

zzzzzzzz – value of term B

\* - unused

1010yyyyyyyyyyyy – JMP instruction format

Yyyyyyyyyyyy – address of the next program line

1011\*\*\*\*\*\*\*\*\*\*\*\* – unused instruction

1100\*\*\*\*\*\*\*\*\*\*\*\* – unused instruction

intended for DRW and BLT instructions in the case of having a pixel LED display

1101xxx\*yyyyyyyy – MOV instruction

xxx – registry address

yyyyyyyy – value

value is copied in the registry

1110xxx\*yyyyyyyy – MOV instruction

xxx – registry address

yyyyyyyy – RAM address

the RAM value at yyyyyyyy is copied in the registry

1111xxx\*yyyyyyyy – MOV instruction

xxx – registry address

yyyyyyyy – RAM address

the value in the registry at xxx is copied in the RAM memory