Description

A microprocessor based on Harvard architecture was designed, using Arduino code, with a certain set of instructions. To achieve this objective, it is first necessary to declare internal records (R0, R1, A and PC) and Carry and Zero flags that belong to the microprocessor, whose functionality is to store data in memory and the indexes of that memory, for the records, and show whether the A value overlapped the maximum allowed by the number of bits set for the microprocessor, in the case of the Carry flag, or whether the value of A is zero or not, in the case of the Zero flag. For one of the instructions, a constant 8-bit number is declared.

Each register needs a certain number of bits, in order to make sure that the encoding of the instructions is never the same for anyone. This number of bits will be transferred to the data and code memories, which, in turn, will be indicated and used by registers R0 and R1.

To make sure that the encoding is as simple as possible, it is mandatory that the instructions are encoded with the fewest bits possible.

The method of operation of the microprocessor is demonstrated through a functional module, a graph that is composed of:

- Multiplexers, filters that, depending on the Boolean value of an Enable register, have as output the malleable data received as input from the rest of the system, or a constant value received as input from the code itself;
- An ALU, a module that performs arithmetic operations, using two input values from the rest of the system, and which outputs the result of these operations and the new Carry and Zero flags resulting from the same operations;
- The code memory, which stores the instruction code in each of its registers, which in turn is sent through the system to carry out the same instructions;
- The data memory, which stores integer values in each register, which can be the same used in ALU arithmetic operations.
- Two D-Latch Flip-Flops, which receive the Boolean values of the Carry and Zero flags and, when the Master Clock is in RISING, send them to the control module and the Carry flag to the ALU;
- The control module, which, upon receiving certain input values, sends other values on the output that will control the various modules of the rest of the system.

After designing the functional module, a ROM is programmed, in this case an EPROM, which will contain the encodings for each microprocessor instruction, and that encoding will, in turn, determine the output values of the control module.