What is LedCPU

This simple CPU is one that has a simple mechanism that we will describe below.

The instructions are 16 bits. These are kept in a ROM containing up to 256 instructions. The CPU has 2 possible instructions:

- Output
- Jump

These operations are divided within the instructions as follows:

BIT #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output	OUTPUT PATTERN								DELAY≠0							
Jump	JUMP ADDRESS								DELAY=0							

So, we have the following logic:

1) DELAY = 0 => Do Jump operation BIT[15:8]

   is the address we jump to.

2) DELAY ≠0 => Do Output operation

   BIT[15:8] is the pattern shown in the LEDs.

   BIT[7:0] is the amount of time this particular pattern should be shown in the LEDs.

   Then it moves to the next address

The CPU simply starts from address 0 and executes the following instructions.

Example

Let's say we wish to make the rotating dot example with this CPU and consider the original version where the dot rotated from left to right.

Let's divide the rotating dot to LED outputs, so we should see:

BIT #	7	6	5	4	3	2	1	0
STAGE0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
STAGE1	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
STAGE2	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
STAGE3	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
STAGE4	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
STAGE5	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
STAGE6	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
STAGE7	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

So the correspoding LedCPU's ROM, i.e. its memory, should be:

BIT # / Address	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	7	6	5	4	3	2	1	0
0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1	0	0	0	0	0	0	0
1	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	1	0	0	0	0	0	0	0
2	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	1	0	0	0	0	0	0	0
3	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	1	0	0	0	0	0	0	0
4	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	1	0	0	0	0	0	0	0
5	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	1	0	0	0	0	0	0	0
6	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	1	0	0	0	0	0	0	0
7	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	1	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

So, between addresses 0 and 7, we give new LED patterns that are in the instruction's BIT[15:8] and ask the delay to be 8'b10000000 in BIT[7:0]. At the last address (address 8), the CPU jumps back to the instruction at the address 0.

We will fill the missing part in LedCPUcore.v