# Instruction Memory

Each instruction has its own address, marked by the 32-bit vector of the program counter. When a given instruction is needed, the Instruction Memory delivers a 32-bit wide instruction. The following considerations have to be taken into account for the Instruction Memory:

• Each instruction is one word long (32 bits).

• Each instruction is addressed by a multiple of 4 bytes (1 word).

• Each instruction lies at a multiple of 4 bytes. By contrast, the program counter counts in terms of bytes. To avoid systematic “jumps” by four instructions when the new program counter is proposed, the program counter is divided by 4.

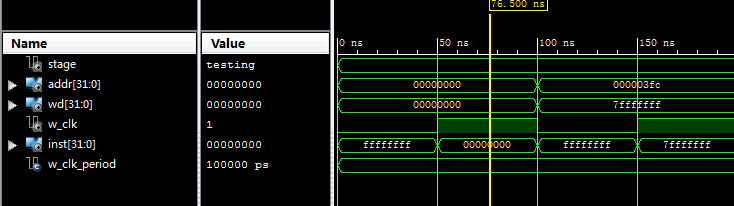
• At the rising edge of w\_clk, we write the value of the wd into instruction memory. In this way, it can support changing the program while our processor is running on the FPGA.

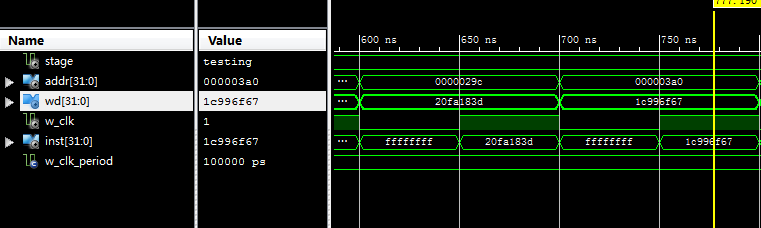
Test:

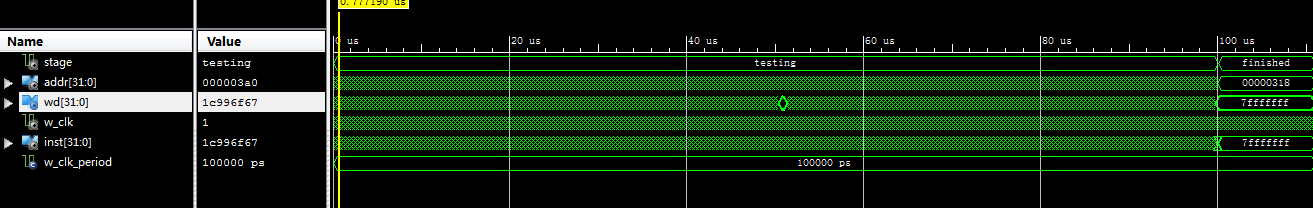
Random generate 1000 test cases.

1. Generate random variable to change addr(9 downto 2) and random variable to change instruction
2. When stage equals to finished, all the test case passed.

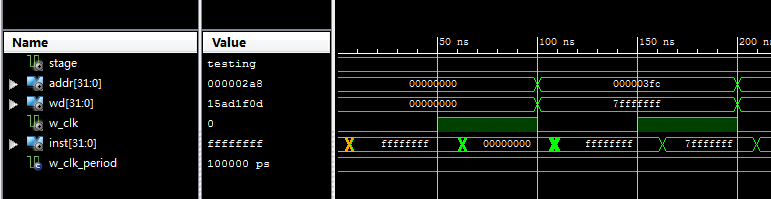
Function Simulation:

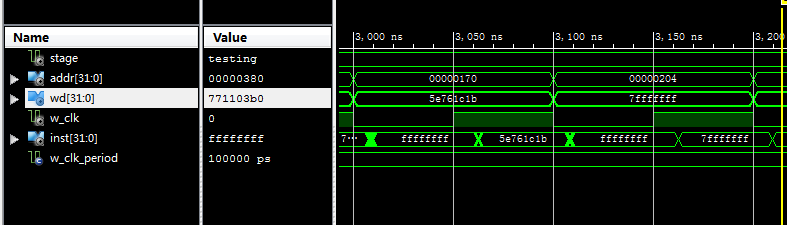


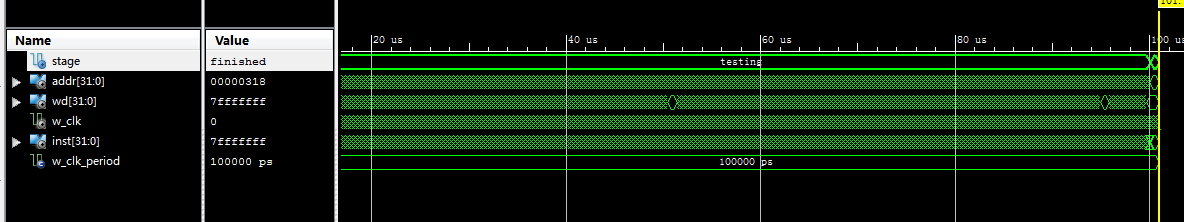




Timing Simulation:







Latency:

It takes 100000ns to get the result of 1000 test case. Clock period=100ns, so it’s 1000 clock cycles.

RC5 part:

(1)

ukey = 0x91cea91001a5556351b241be19465f91

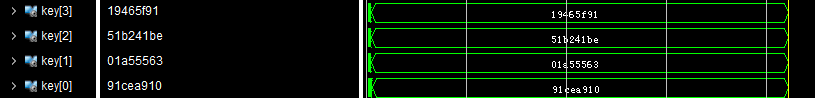
A\_in = 0xeedba521

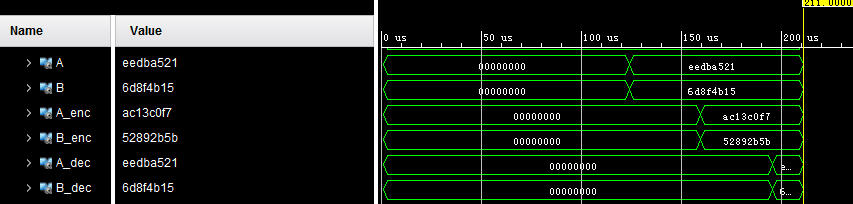
B\_in = 0x6d8f4b15

Functional simulation:

A\_enc = AC13C0F7

B\_ enc = 52892B5B





period: 10ns

It takes 160000ns to encode, about 16000 cycles.

It takes 196825ns to decode, about 19682 cycles.

(2)

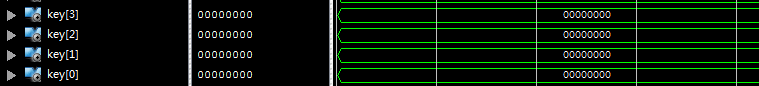
ukey = 0x00000000000000000000000000000000

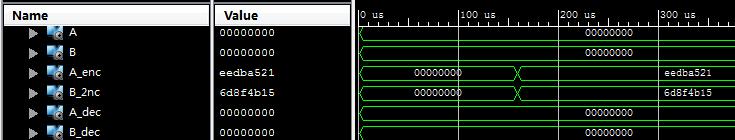
A\_in = 0x00000000

B\_in = 0x00000000

A\_enc = eedba521

B\_enc = 6d8f4b15





(3)

ukey = 8767C18DBB69B1D72F0FEB5AE7483378

A\_in = AC13C0F7

B\_in = 52892B5B

A\_enc = B7B3422F

B\_enc = 92FC6903

