RC5 Timing Report

1. The Operations

The simulation test bench performs these operations:

* Reset the CPU for a while at the very beginning
* Perform a few necessary store & load operations to get Ukey
* Run Key-Generation
* Perform a few necessary store & load operations to get Din
* Run Encryption on the Din
* As soon as the encrypted value is valid, run Decryption on that value
* Finally get the decrypted value

If all the functions are correct, the final value we get should be the same as the initial Din.

Din (64 bits) is stored in data\_mem(40) and data\_mem(41), while the encrypted value is stored in data\_mem(42) and data\_mem(43), and the decrypted value is stored in data\_mem(44) and data\_mem(45). We will mainly look at these 6 values to check if our functions are correct.

For the simulation in this report, we will use:

* Ukey = 91CEA91001A5556351B241BE19465F91
* Din = EEDBA5216D8F4B15

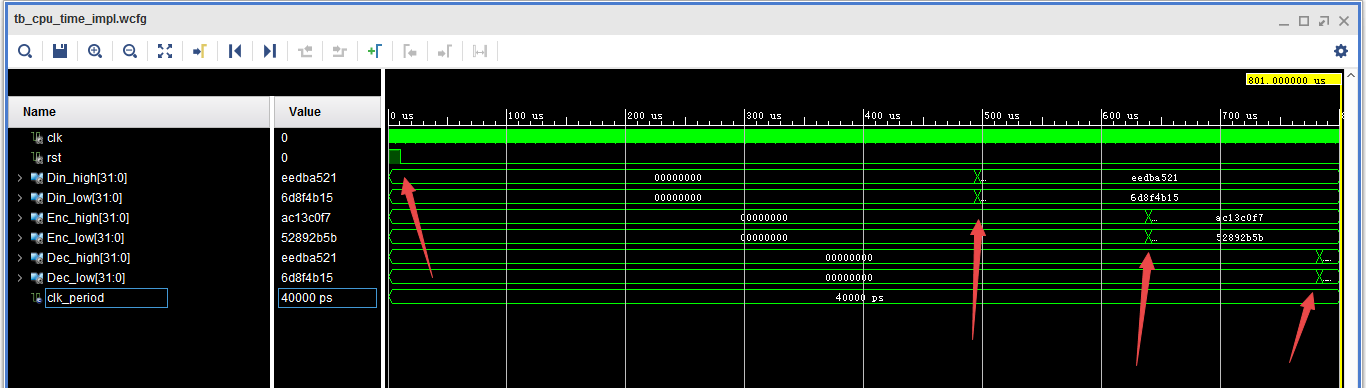
2. Timing Simulation

In this simulation, we will just show the values in data\_mem(40 to 45). We renamed those values so that they will be clearer in the waveform. They are labeled with Din(initial), Enc(encrypted), Dec(decrypted). Besides, a label of high/low indicates whether the 32-bit signal is the high 32 bits or the low 32 bits of the 64-bit value.

The sequence of the signals is:

clk, rst, Din\_high, Din\_low, Enc\_high, Enc\_low, Dec\_high, Dec\_low, clk\_period

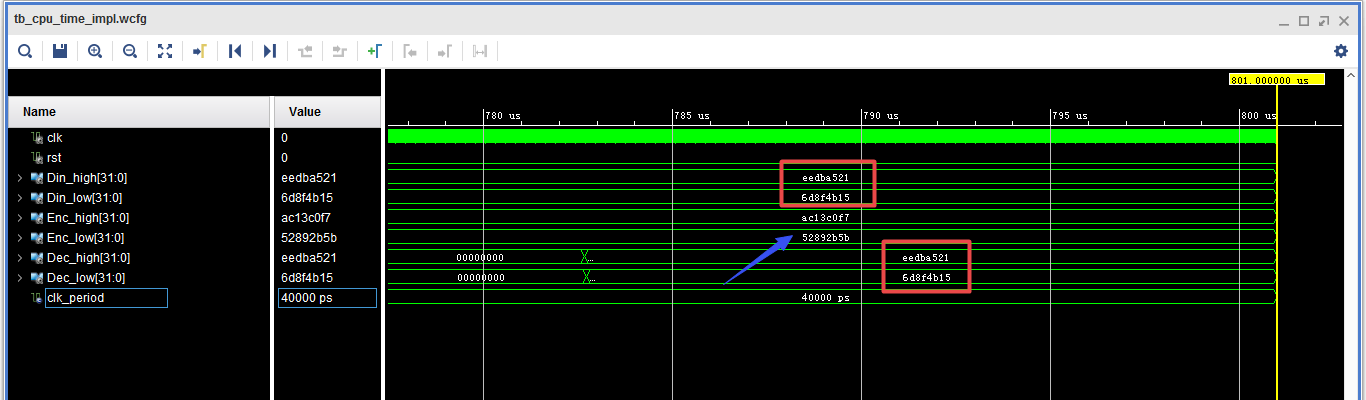
(1)



This is the whole waveform of the timing simulation. We add some arrows to mark the events in the process.

* At the 1st arrow, we let rst become low so that we can begin the Key-Generation.
* At the 2nd arrow, the key-generation is completed, and the value of Din is entered so that we will begin the Encryption.
* At the 3rd arrow, the encrypted value is obtained and saved, so we begin Decryption on this value.
* At the 4th arrow, the decrypted value is obtained and saved.

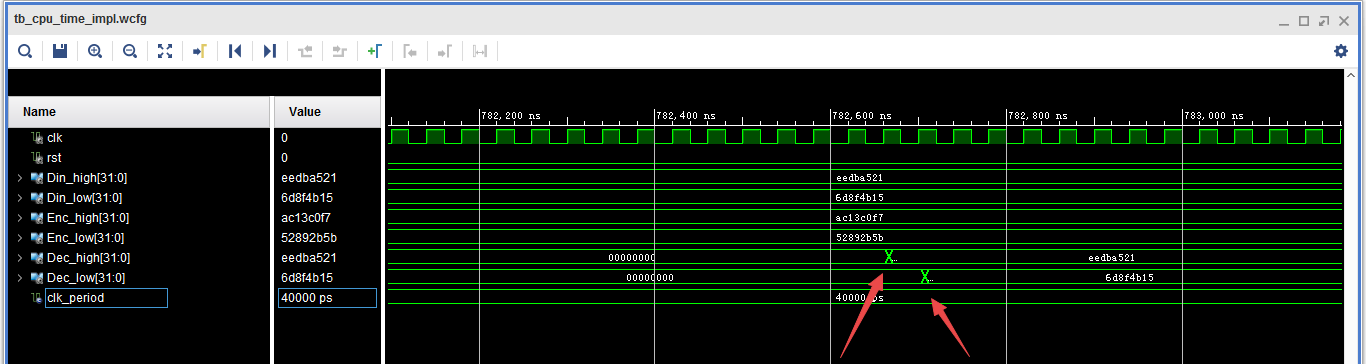
(2)



This show the final results. We can see that the encrypted-decrypted value is the same as the initial Din (the two red boxes). Besides, the encrypted value (pointed by the blue arrow) is also a correct encryption result:

* Enc\_high & Enc\_low = AC13C0F752892B5B

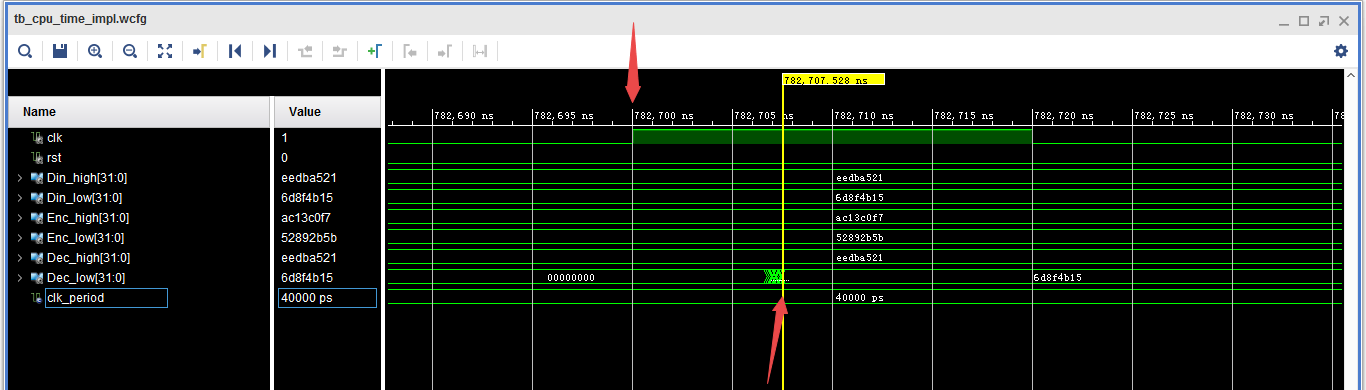
(3)



Here we go into some details about the time we get the final decrypted value. Notice that the low 32 bits are obtained one cycle later than the high 32 bits.

It’s easy to understand that: the CPU will perform one instruction in one cycle. So we have to first store the high 32 bits of the 64-bit output, then store the low 32 bits.

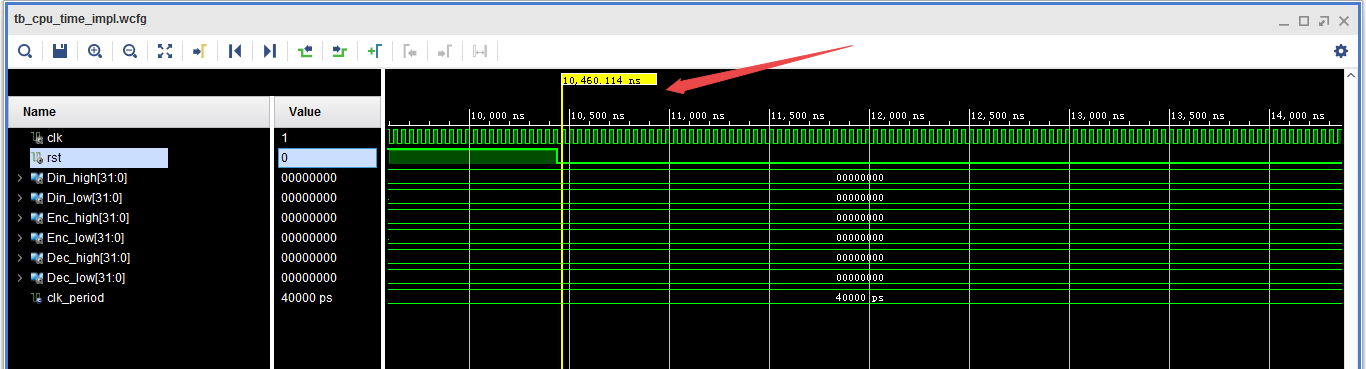
(4)



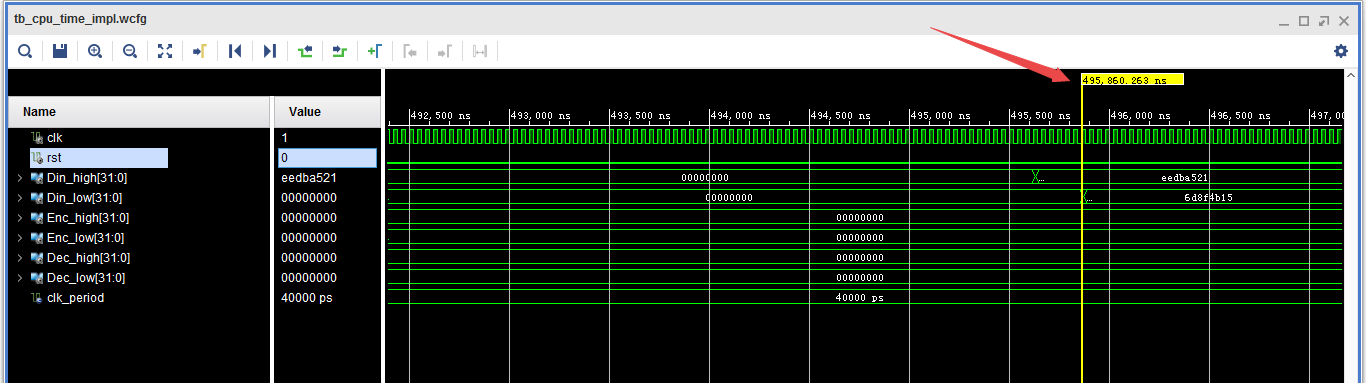
This figure is the more detailed capture of the time we get the final value (low 32 bits). As this is a timing simulation, there is a delay between the clock rising edge and the completion of the store operation (showed by the two arrows).

3. Latency

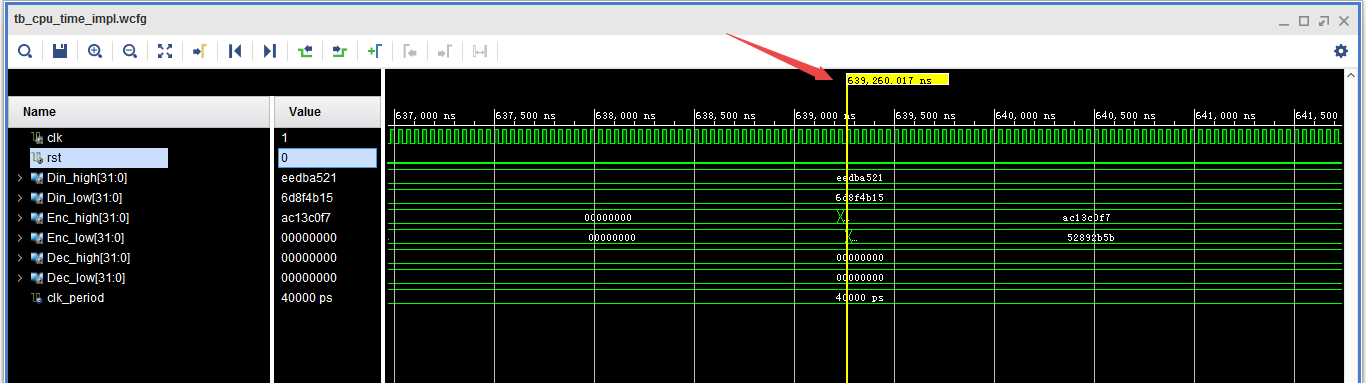
Here we will estimate the clock cycles that the CPU uses to run Key-Generation, Encryption, and Decryption. We can use the time points showed by the 4 arrows in PART 2 (1) to do that. We will show the detailed time.



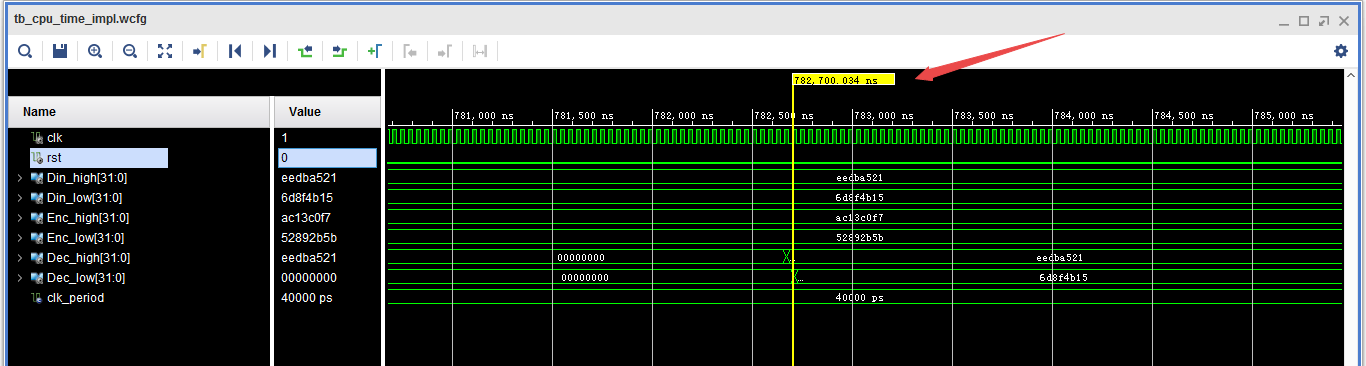
Time 1 = 10460ns



Time 2 = 495860ns



Time 3 = 639260ns



Time 4 = 782700ns

We can use this formula to calculate the clock cycles:

So:

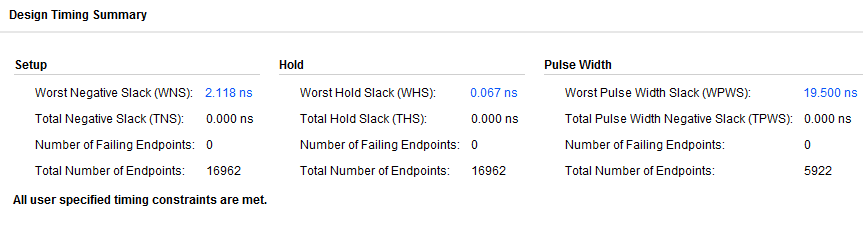
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Begin | End | Time-Diff | Latency | clk period |
| Key-Gen | 10460 | 495860 | 485400 | 12135 | 40 |
| Encryption | 495860 | 639260 | 143400 | 3585 |
| Decryption | 639260 | 782700 | 143440 | 3586 |

The Latency is presented in *number of cycles*.

Notice that there are a few read & write operations before or after each of these three functions, but the cycles they take can be ignored compared to the cycles taken by Key-Gen, Encryption and Decryption.

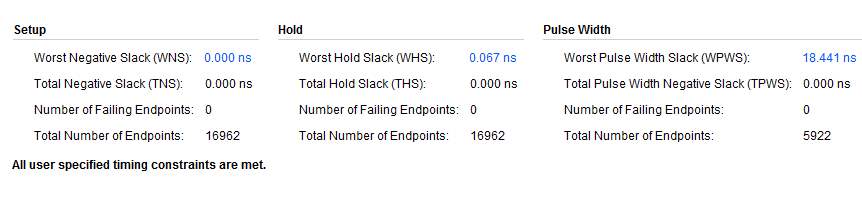
4. Delay

First we set clock period = 40ns and see the timing summary:



The WNS = 2.118ns. So we can calculate the minimum clock period:

Then we set clock period as this value to check the timing summary:

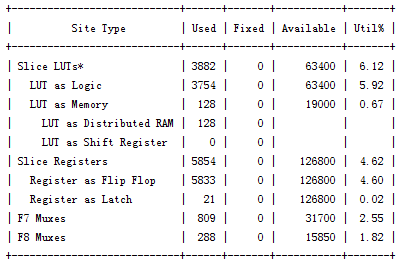


The WNS is just 0, and there’s no failed points. Thus we can regard 37.882ns as the Minimum Clock Period, which also indicates the critical path delay.

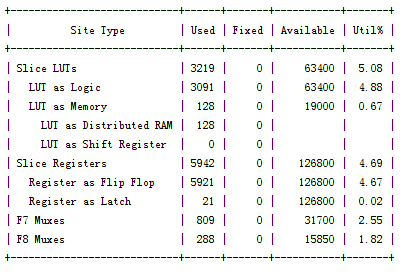
We further check this clock period in the simulation, and it works. So we can conclude:

|  |  |  |
| --- | --- | --- |
| General | Critical Path Delay | 37.882 ns |
| Maximum Frequency | 26.40 MHz |
| Key-Generation | Latency | 12135 cycles |
| Propagation Delay | 460 μs |
| Encryption | Latency | 3585 cycles |
| Propagation Delay | 136 μs |
| Decryption | Latency | 3586 cycles |
| Propagation Delay | 136μs |
| Total | Total Propagation Delay | 0.732 ms |

5. Resource Utilization



After Synthesis



After Post-Route Phase

|  |  |  |
| --- | --- | --- |
|  | Synthesis Stage | Place and Route Stage |
| Slice LUTs | 3882/63400 (6.12%) | 3219/63400 (5.08%) |
| FFs | 5833/126800 (4.60%) | 5921/126800 (4.67%) |
| IOB usage | 156/210 (74.29%) | 156/210 (74.29%) |