Homework 3 Report

Course:

IL2234 HT25 Digital Systems Design and Verification using Hardware Description Languages

Student Name:

Jinye Gong

GitHub Repository:

https://github.com/kth-ees/il2234ht25-hw3-jinye-gong

Submission Date:

2025-10-08

Question 1:

Design an FSM that can detect an `input_sequence` of 5 consecutive ones. Assume that a new input comes at every clock cycle. The output `detected` should be asserted at the same cycle when the 5th '1' is given to the input. It should remain asserted as long as the input remains '1'.

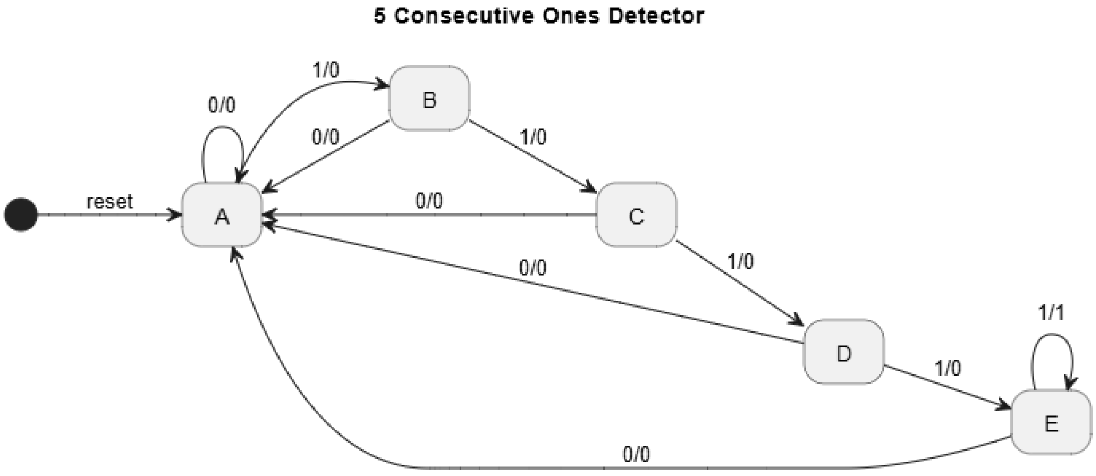
Module pinout

Name	Direction	Width	Control/Data	Description
clk	in	1	Control	Clock signal
rstn	in	1	Control	Asynchronous active low reset
input_sequence	in	1	Data	Input
detected	out	1	Data	1 if 5 consecutive ones are detected

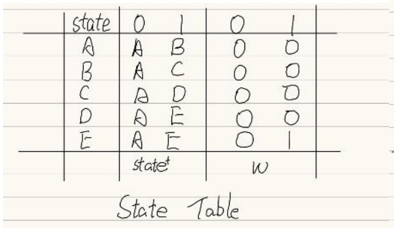
1. What should be the type of FSM machine?

This state machine belongs to the Mealy type because its output is jointly determined by both the current state and the current input.

2. Draw the state diagram for the FSM machine.

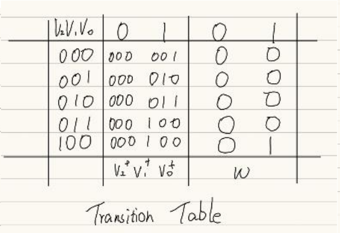


3. Derive the next state logic using D flops.



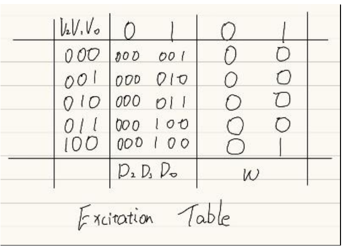
state	0	1	0	1
A	A	B	0	0
B	A	C	0	0
C	A	D	0	0
D	A	E	0	0
E	A	E	0	1
	state⁺		W	

State Table



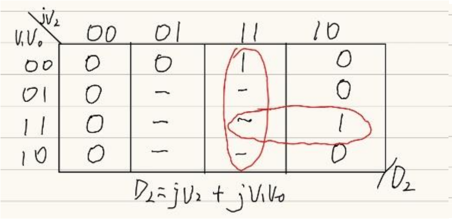
$V_2V_1V_0$	0	1	0	1
000	000	001	0	0
001	000	010	0	0
010	000	011	0	0
011	000	100	0	0
100	000	100	0	1
	$V_2^+ V_1^+ V_0^+$		W	

Transition Table



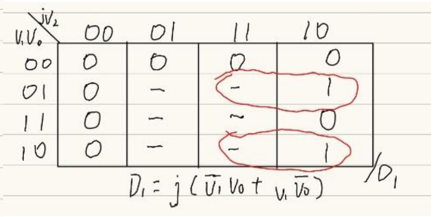
$V_2V_1V_0$	0	1	0	1
000	000	001	0	0
001	000	010	0	0
010	000	011	0	0
011	000	100	0	0
100	000	100	0	1
	$D_2 D_1 D_0$		W	

Excitation Table



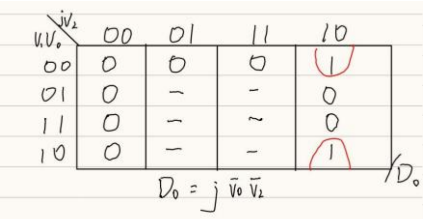
v_1v_0 \ jv_2	00	01	11	10
00	0	0	1	0
01	0	–	–	0
11	0	–	~	1
10	0	–	–	0

$D_2 = jv_2 + jv_1v_0$ /D_2



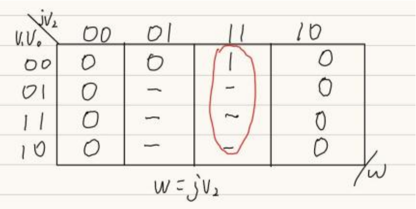
v_1v_0 \ jv_2	00	01	11	10
00	0	0	0	0
01	0	–	–	1
11	0	–	~	0
10	0	–	–	1

$D_1 = j(\bar{v}_1v_0 + v_1\bar{v}_0)$ /D_1



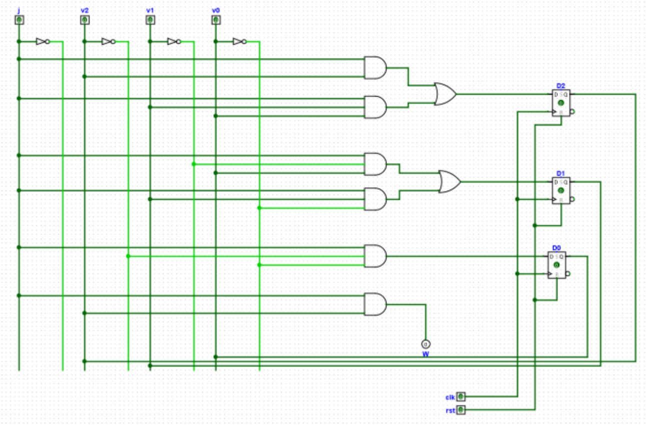
v_1v_0 \ jv_2	00	01	11	10
00	0	0	0	1
01	0	–	–	0
11	0	–	~	0
10	0	–	–	1

$D_0 = j\bar{v}_0\bar{v}_2$ /D_0

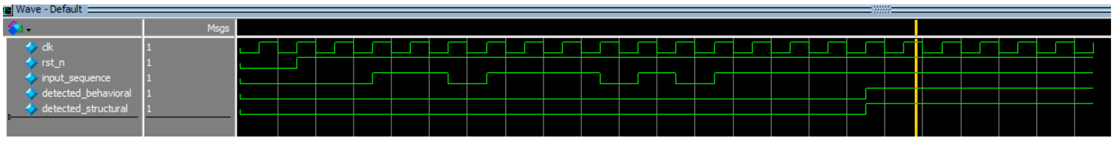


v_1v_0 \ jv_2	00	01	11	10
00	0	0	1	0
01	0	–	–	0
11	0	–	~	0
10	0	–	–	0

$W = jv_2$ /w



6. Verify both structural and behavioral models and compare their results and waveforms together in your report.



	Msgs
clk	1
rst_n	1
input_sequence	1
detected_behavioral	1
detected_structural	1

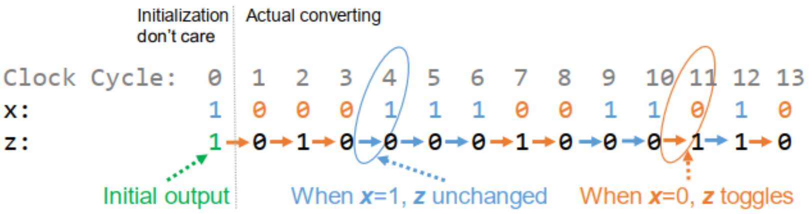
The testbench shows that the output waveforms of the structural and behavioral models are identical, and both correctly perform the detection.

Question 2:

Design a message conversion system with the conversion rule below:

For every cycle, if the input message bit is '0', toggle the current output bit; if the input message bit is '1', keep the output bit unchanged.

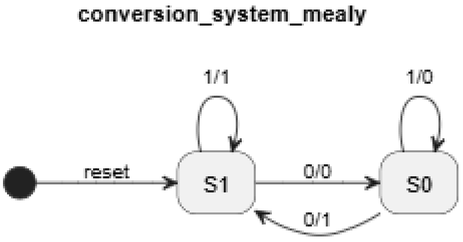
For example, let the input message be x, and the output message be z. Assuming z initially is '1'. Given the sequence of x = 10001110011010, the output sequence of z should be: 10100001000110. See the figure below.


Initialization don't care
Actual converting
Clock Cycle: 0 1 2 3 4 5 6 7 8 9 10 11 12 13
x: 1 0 0 0 1 1 1 0 0 1 1 0 1 0
z: 1→0→1→0→0→0→0→1→0→0→0→1→1→0
Initial output
When x=1, z unchanged
When x=0, z toggles

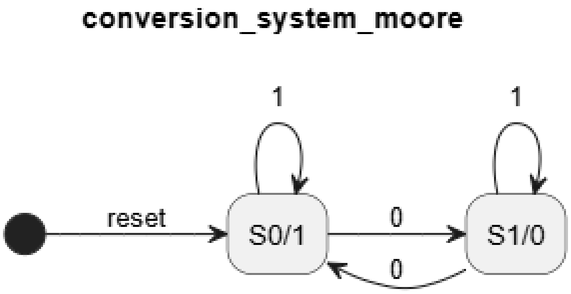
Module pinout

Name	Direction	Width	Control/Data	Description
clk	in	1	Control	Clock signal
rstn	in	1	Control	Asynchronous active low reset
x	in	1	Data	Input
z	out	1	Data	Output

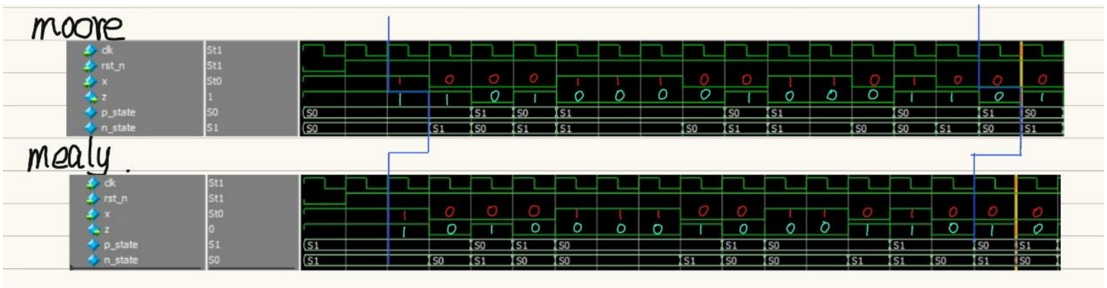
(1) Draw the state diagram of a **Mealy FSM** with the same behavior as described above.



(2) Draw the state diagram of a **Moore FSM** with the same behavior as described above.



(3) Write a testbench for both FSMs and compare the timing, number of states, and the output waveforms of Mealy and Moore FSMs in your report.



Number of states:

Both have 2 states (S0, S1).

Timing:

- **Mealy:**
  - When x=0, *z* toggles in the same cycle.
  - When x=1, z holds in the same cycle.
- **Moore:**
  - When x=0, z toggles in the **next cycle**.
  - When x=1, *z* holds in the **next cycle**.

Evaluations:

- In a **Mealy machine**, z responds in the **same cycle** as x.
- In a **Moore machine**, *z* is **delayed by one clock cycle** relative to x.

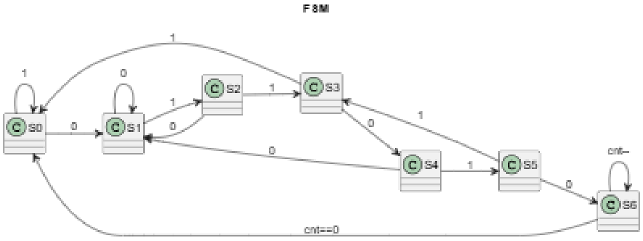
Question 3:

A serial communication device has a begin-sequence (011010) that leads to the transmission of 32 bits on its `serData` input. After receiving the begin-sequence, the `outValid` output becomes 1, and the next 32 bits on `serData` are regarded as valid serial data of the communication device. After 32 clock cycles, `outValid` becomes 0 and the device returns to the first state, where it searches for the begin-sequence again.

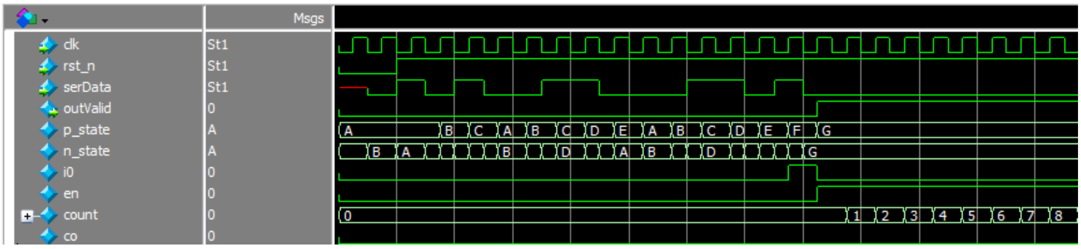
Module pinout

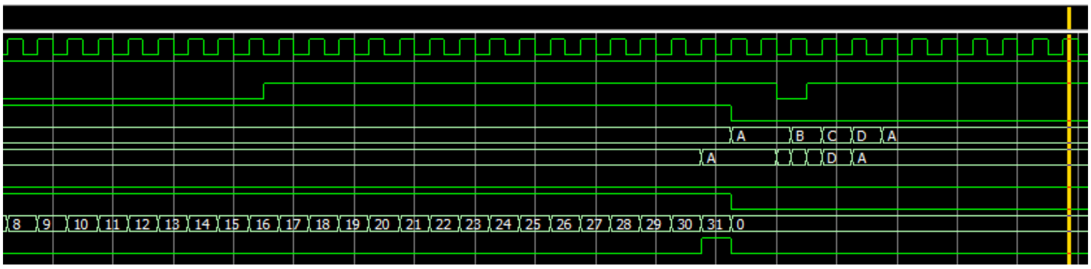
Name	Direction	Width	Control/Data	Description
clk	in	1	Control	Clock signal
rstn	in	1	Control	Asynchronous active low reset
serData	in	1	Data	Input
outValid	out	1	Data	Output

(1) Create a state diagram to represent the moore FSM for this system, clearly showing all states, transitions, and places where the output is issued. Draw the FSM in your report.



(3) Write a testbench and verify the functionality of your design.





The waveform demonstrates successful detection of the sequence 011010. Following this detection, the outValid signal is activated, enabling the complete transmission of the subsequent 32-bit data frame.

Question 4:

Design and implement an *average calculator* that computes the average of n m-bit input numbers, where n=2^k with $\boldsymbol{k} > \mathbf{1}$, and $\boldsymbol{m} > \mathbf{1}$. The module should begin computation when it receives a positive pulse on the `start` signal and assert the `done` signal once the average is ready.

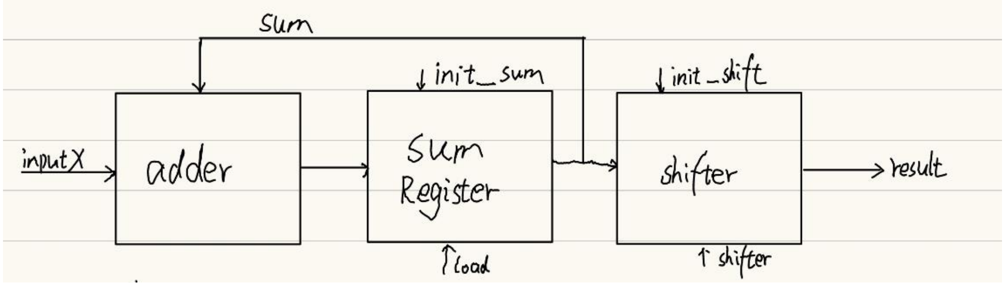
Module parameters

parameter	Description
m	input bit width
n = 2^k , $k > 1$	number of inputs

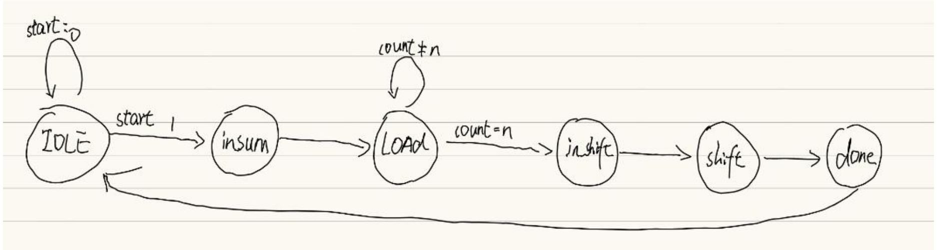
Module pinout

Name	Direction	Width	Control/Data	Description
clk	in	1	Control	Clock signal
rstn	in	1	Control	Asynchronous active low reset
start	in	1	Control	Input control signal
inputx	in	m	Data	Input data
done	out	1	Control	Output
result	out	m	Data	Output result

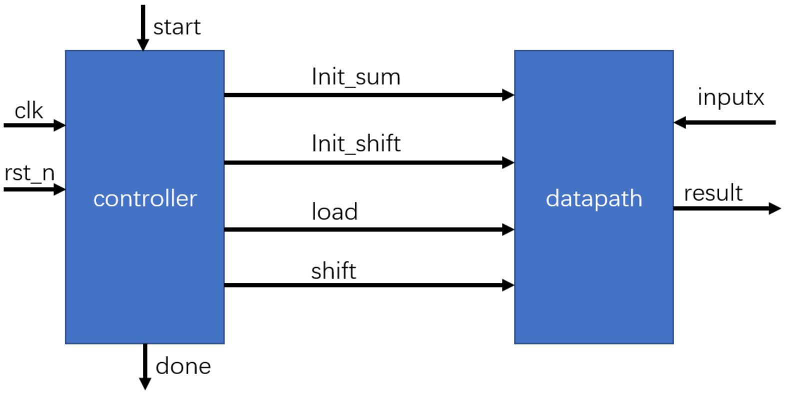
1. Draw a schematic of your datapath in your report, including the components, their interfaces, and necessary control signals.



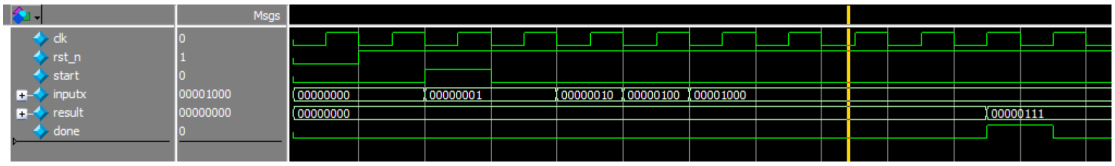
2. Draw a state diagram showing your controller's behavior in your report. In each state, show the control signals that are issued.



4. Show the wiring between the datapath and controller within the top-level module in your report.



5. Verify the functionality of your module with a testbench.



Question 5:

The Taylor series expansion of *sinx* is shown below.

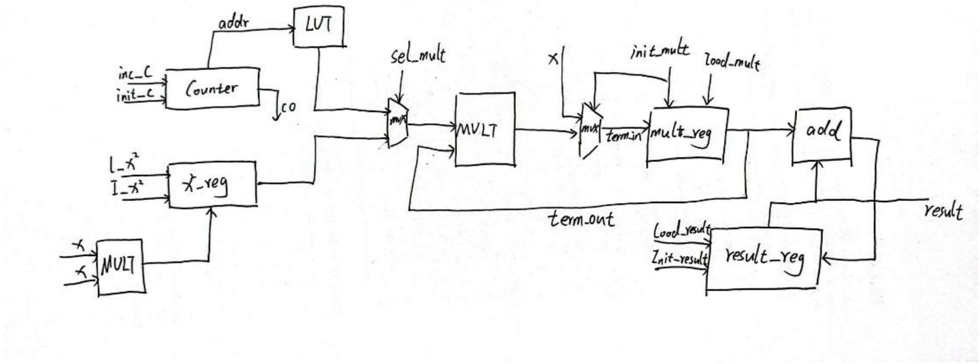
$$\sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \cdots$$

Design an accelerator that calculates the sin of x based on the Taylor series expansion of sinx. The x signal is a 15-bit fraction and 1-bit integer, $x\epsilon[0,\frac{\pi}{2})$. The accelerator starts calculating the sin of x when it receives a positive pulse on the `start` signal. Accelerator should issue 1 on the `done` signal when the result is calculated after eight iterations. The accelerator has a 16-bit fractional output signal called `result`, $result\epsilon[0,1)$.

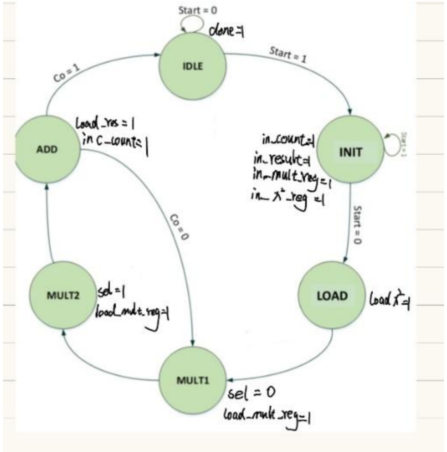
Module pinout

Name	Direction	Width	Control/Data	Description
clk	in	1	Control	Clock signal
rstn	in	1	Control	Asynchronous active low reset
start	in	1	Control	Input control signal
x	in	16	Data	Input data
done	out	1	Control	Output
result	out	16	Data	Output result

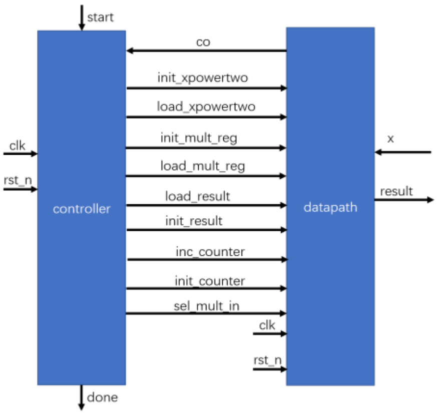
1. Draw a schematic of your datapath in your report, including the components, interfaces, and necessary control signals. (Use the sin_coeff_lut module that is given for the coefficients.)



2. Draw a state diagram that shows the behavior of your controller in your report. In each state, show the control signals that are issued.



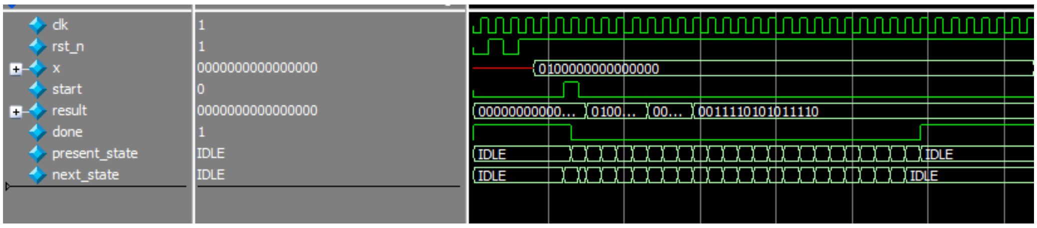
3. Show the wiring between the datapath and controller within the top-level module in your report.



6. Verify the functionality of your accelerator with a testbench.

X=0.5 0100 0000 0000 0000

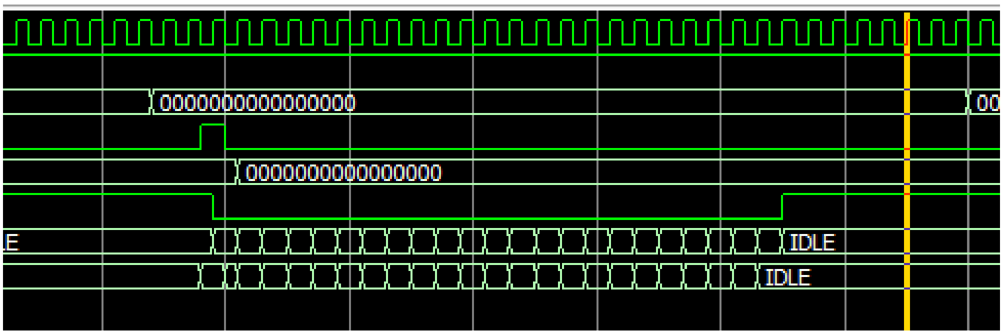
Sin(0.5)=0.47942554 0.011 1101 0101 1101



```
clk            1
rst_n          1
x              0000000000000000
start          0
result         0000000000000000
done           1
present_state  IDLE
next_state     IDLE
```

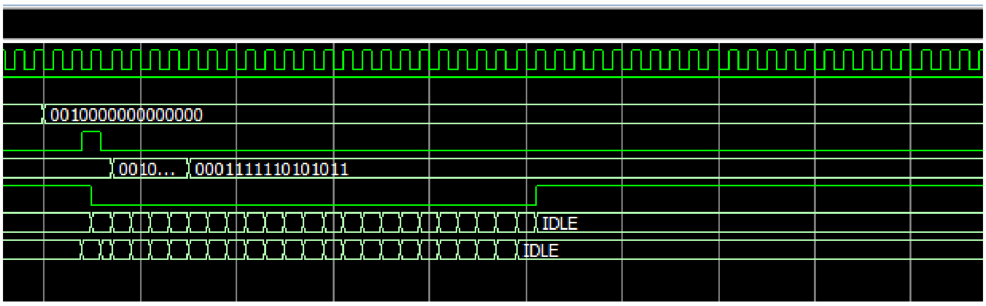
X=0 0000 0000 0000 0000

Sin(0)= 0 0.000 0000 0000 0000



X=0.25 0010 0000 0000 0000

Sin(0.25)= 0.24740396 0.001 1111 1010 1011


0010000000000000
0010...
0001111110101011
IDLE
IDLE