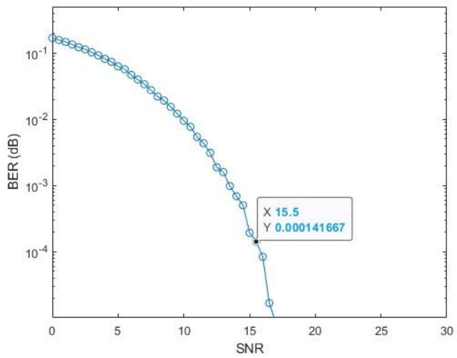
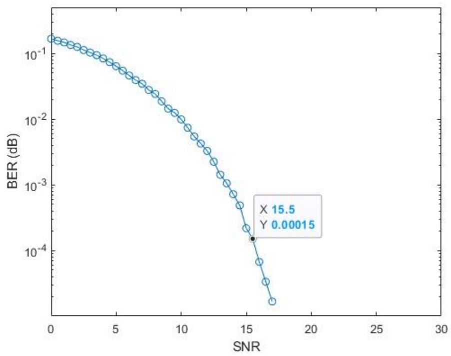
MIMO Decoder for 4 by 4 MIMO Systems with 8-PSK Modulation

Name: 黃琮凱、賴柏睿 ID：R13942116、R13942121

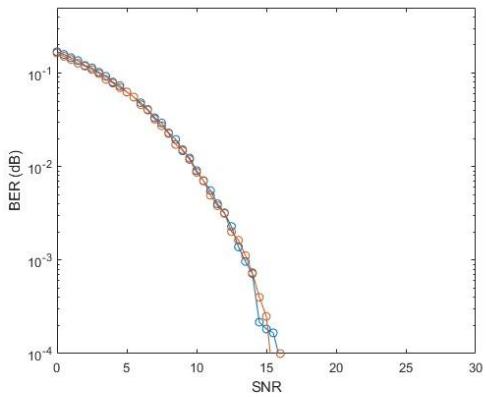
1. BER to SNR plot

Below are the figures of BER to SNR of using ML (left) and that of using fixed point 4-best algorithm (right).





We can easily see that the SNR degradation of the two algorithm is almost zero. Below is the figure of ML (blue line) and 4-best floating point (red line) algorithm. We can verify that the performance of two algorithm are very similar.



2. Algorithm Explanation

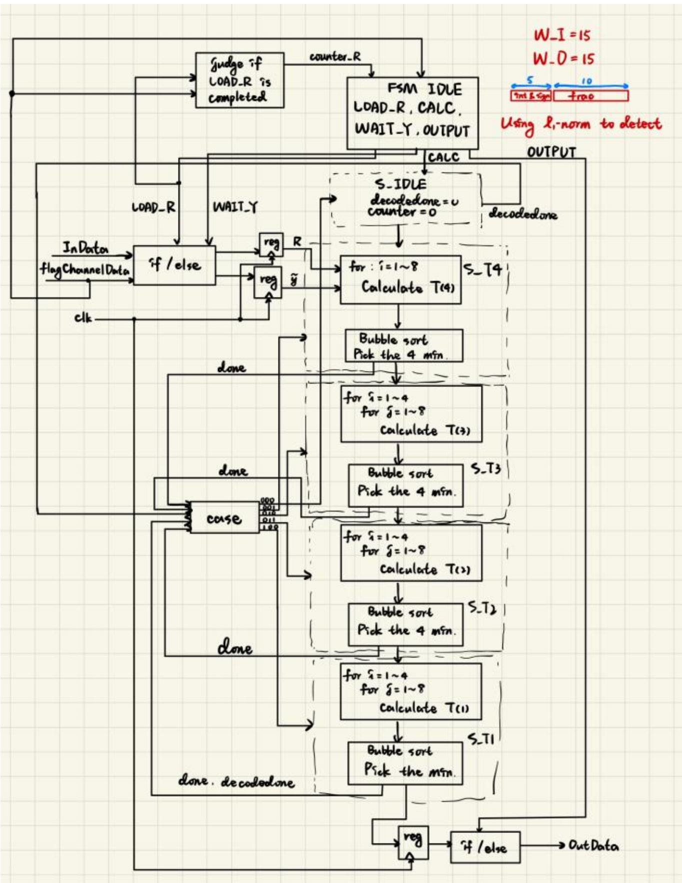
We adopted 4-best algorithm to design our decoder. First, generate random 11 4×1 transmitted signal. Calculating $y = Hx + v$, then we get $\tilde{y} = Q^H y$, where Q is derived from the QR decomposition of H.

Then we calculate the 4 $\bar{x}_4$ such that $T(4) = \left|z^{(4)} - \sum_{j=1}^{4} r_{4,j}\bar{x}_4{}^{(j)}\right|^2$ is the 4 minimum value, where these 4 $\bar{x}_4$ are 4×1 vector, and $\bar{x}_4{}^{(j)}$ is the j-th entry of $\bar{x}_4$, which is one of the 8PSK constellation. Next, along these 4 $\bar{x}_4$, we evaluate the value of the combination of these 4 $\bar{x}_4$ and 8 constellation points to get the 4 $\bar{x}_3$ such that T(3) are the 4 minimum values. Following this regulation, we finally get $\bar{x}_1$ such that T(1) + T(2) + T(3) + T(4) is minimum, then $x_{4B} = min_{\bar{x}_1}\Phi(\bar{x}_1)$.

Below is the diagram of our BER to SNR using 4-best algorithm and ML algorithm. We can clearly see that the performance of these two algorithms are very similar, which also illustrate the effectiveness of 4-best algorithms.

4. Block Diagram

Below is the block diagram we created, using 5 bits (integer + sign) plus 10 fractional bits, making both W_I and W_O equal to 15. Notably, we adopt the L1 norm to reduce both computation time and complexity while retaining sufficient accuracy, which using a small technique from one of the reference paper provided by teacher.



5.(a)

Setting:

SNR:10dB

Constellation point:

```
// 8-PSK 星座圖的實部 (cos(θ))
PSK_8_lut_real[0] <= 15'b000010000000000;  // cos(0) = 1
PSK_8_lut_real[1] <= 15'b000001011010100;  // cos(π/4)=0.707
PSK_8_lut_real[2] <= 15'b000000000000000;  // cos(π/2) = 0
PSK_8_lut_real[3] <= 15'b111110100101100;  // cos(3π/4)=-0.707
PSK_8_lut_real[4] <= 15'b111110000000000;  // cos(π) = -1
PSK_8_lut_real[5] <= 15'b111110100101100;  // cos(5π/4)=-0.707
PSK_8_lut_real[6] <= 15'b000000000000000;  // cos(3π/2) = 0
PSK_8_lut_real[7] <= 15'b000001011010100;  // cos(7π/4)=0.707

// 8-PSK 星座圖的虛部 (sin(θ))
PSK_8_lut_imag[0] <= 15'b000000000000000;  // sin(0) = 0
PSK_8_lut_imag[1] <= 15'b000001011010100;  // sin(π/4)=0.707
PSK_8_lut_imag[2] <= 15'b000010000000000;  // sin(π/2) = 1
PSK_8_lut_imag[3] <= 15'b000001011010100;  // sin(3π/4)=0.707
PSK_8_lut_imag[4] <= 15'b000000000000000;  // sin(π) = 0
PSK_8_lut_imag[5] <= 15'b111110100101100;  // sin(5π/4)=-0.707
PSK_8_lut_imag[6] <= 15'b111110000000000;  // sin(3π/2) = -1
```

```
PSK_8_lut_imag[7] <= 15'b111110100101100;  // sin(7π/4)=-0.707
```

Word length:

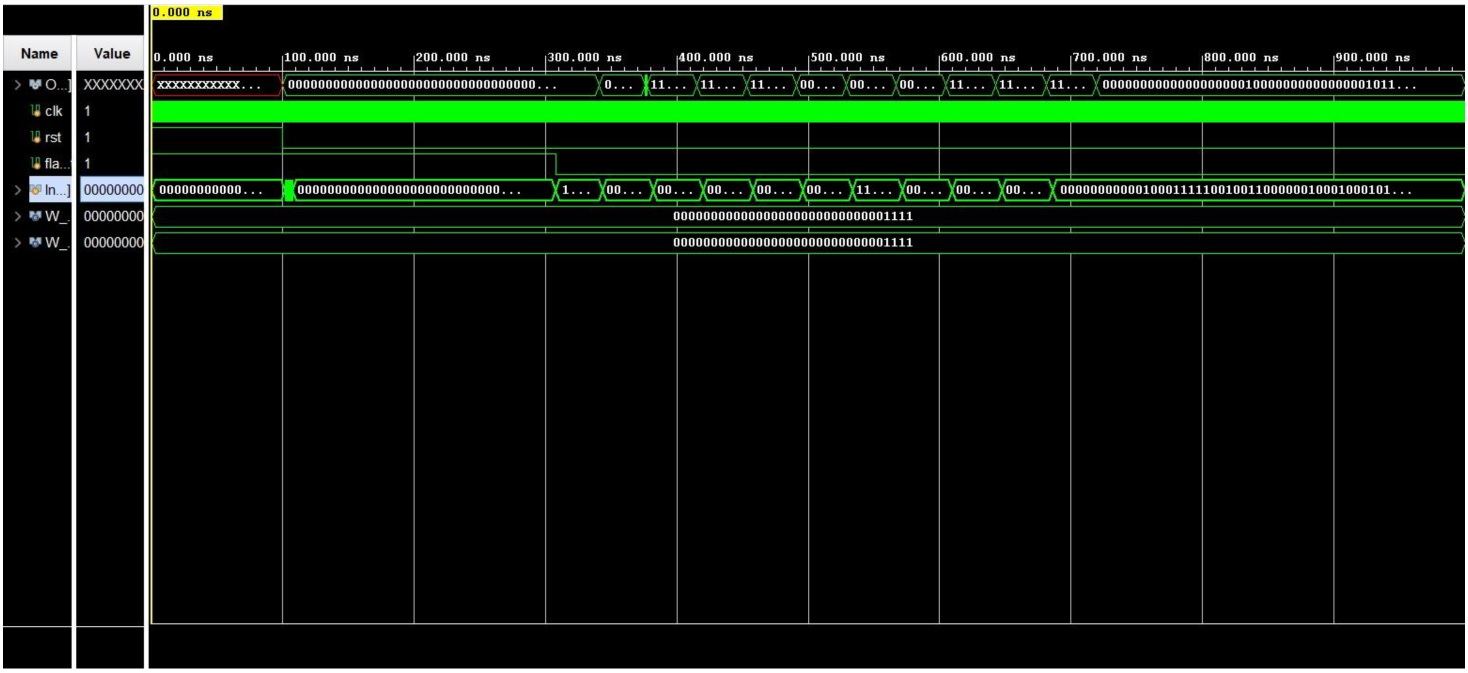
Total:15bits

Sign bit:1 bit

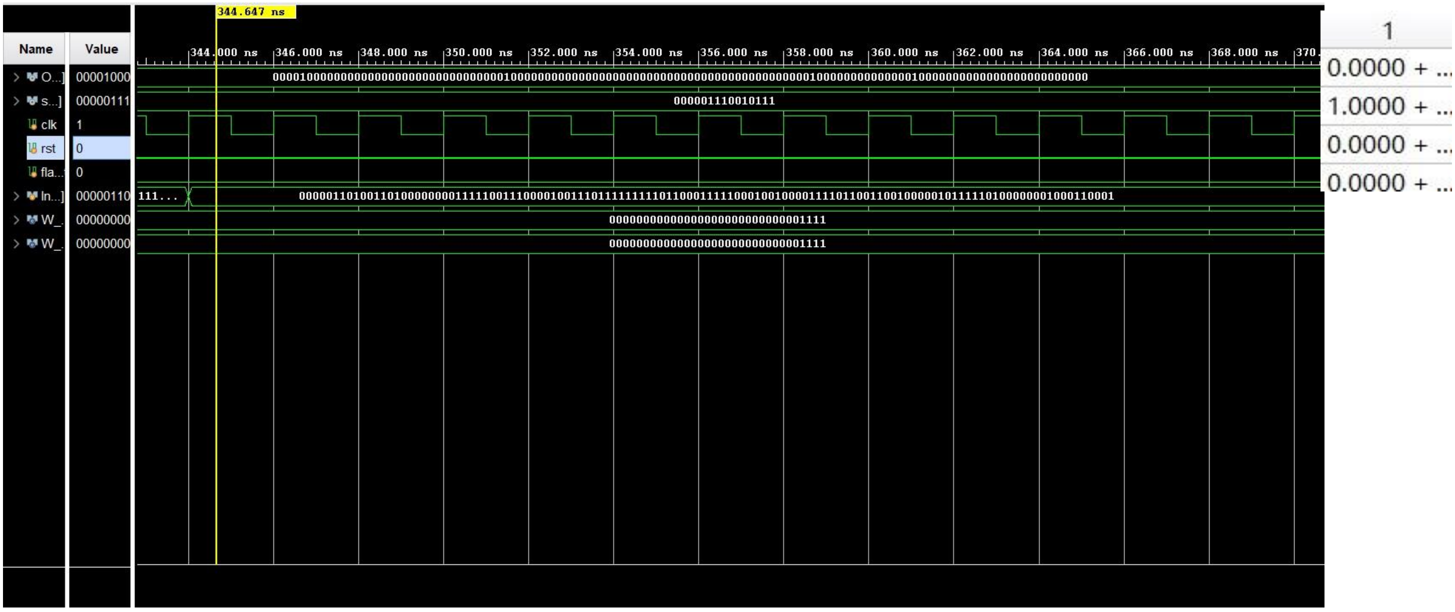
Integer part: 4 bits

Fractional part: 10 bits

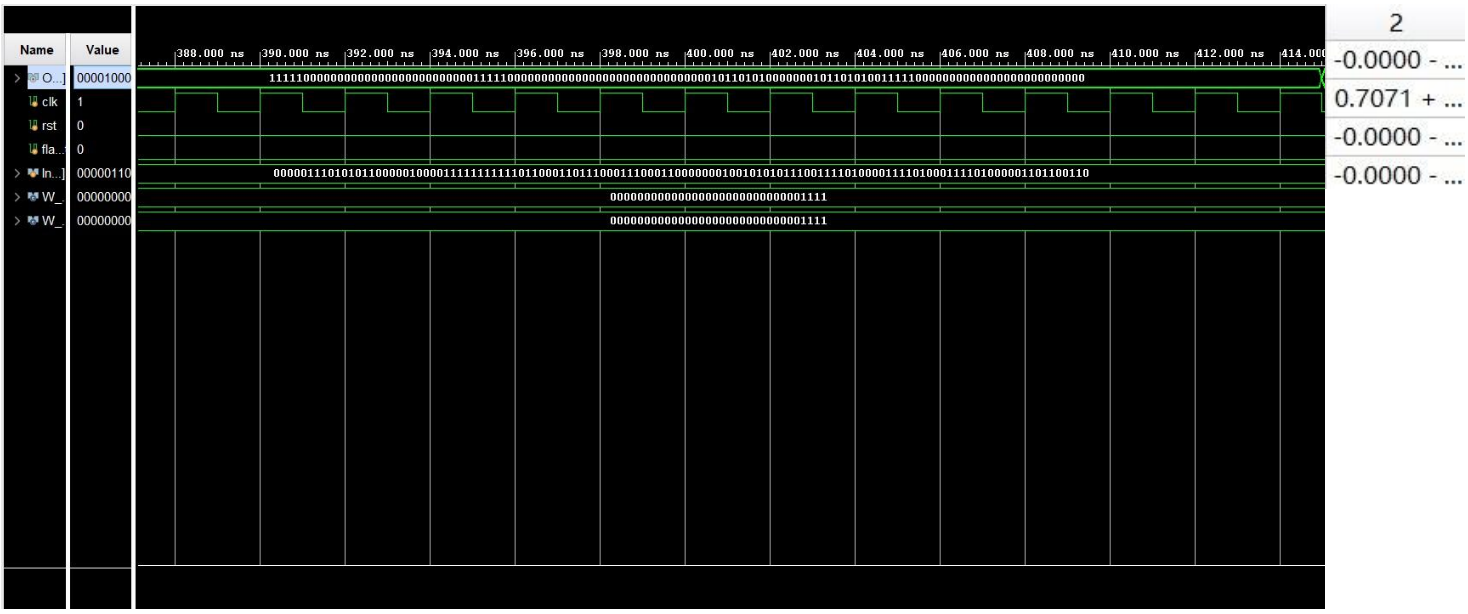
Simulation Results:



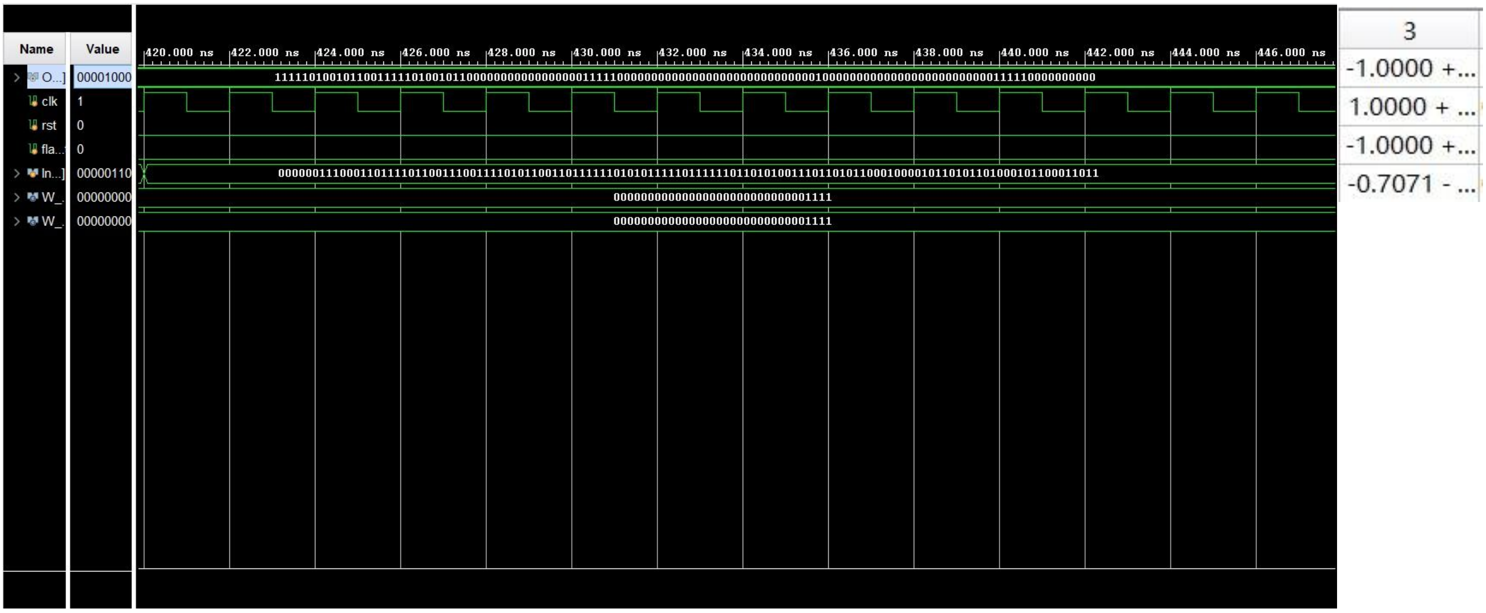
(1)Entire timingdiagram



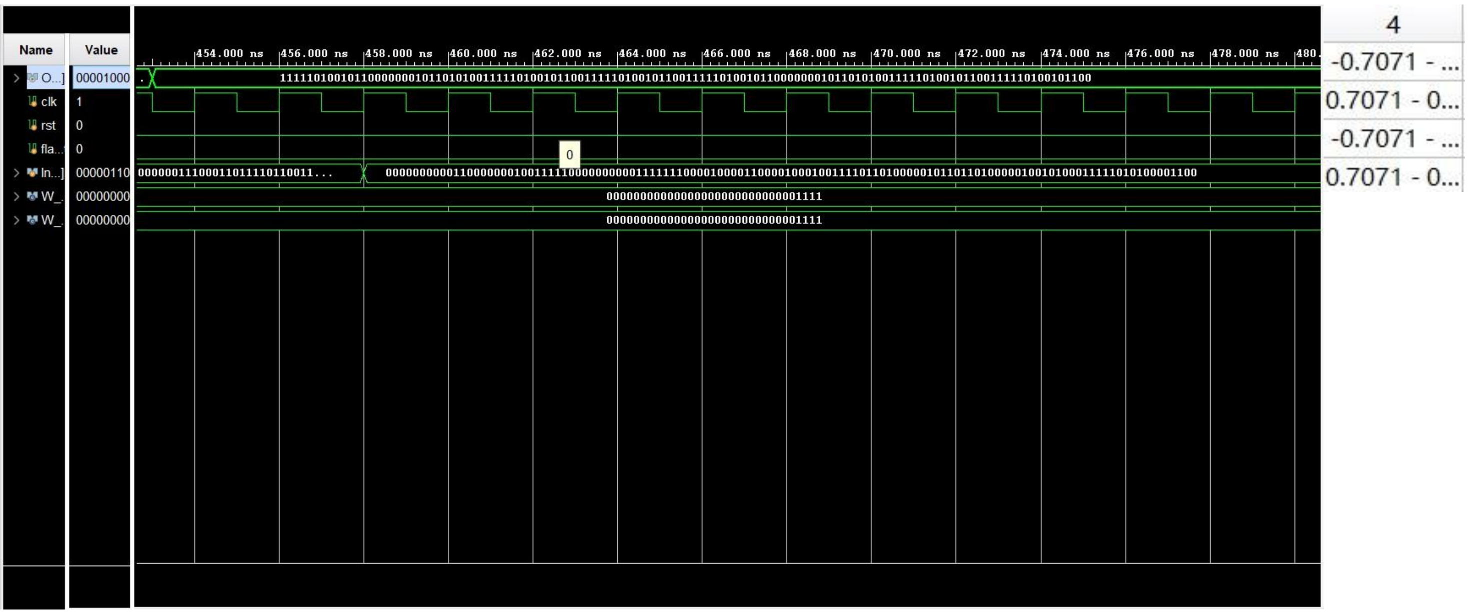
(2)Symbol 1



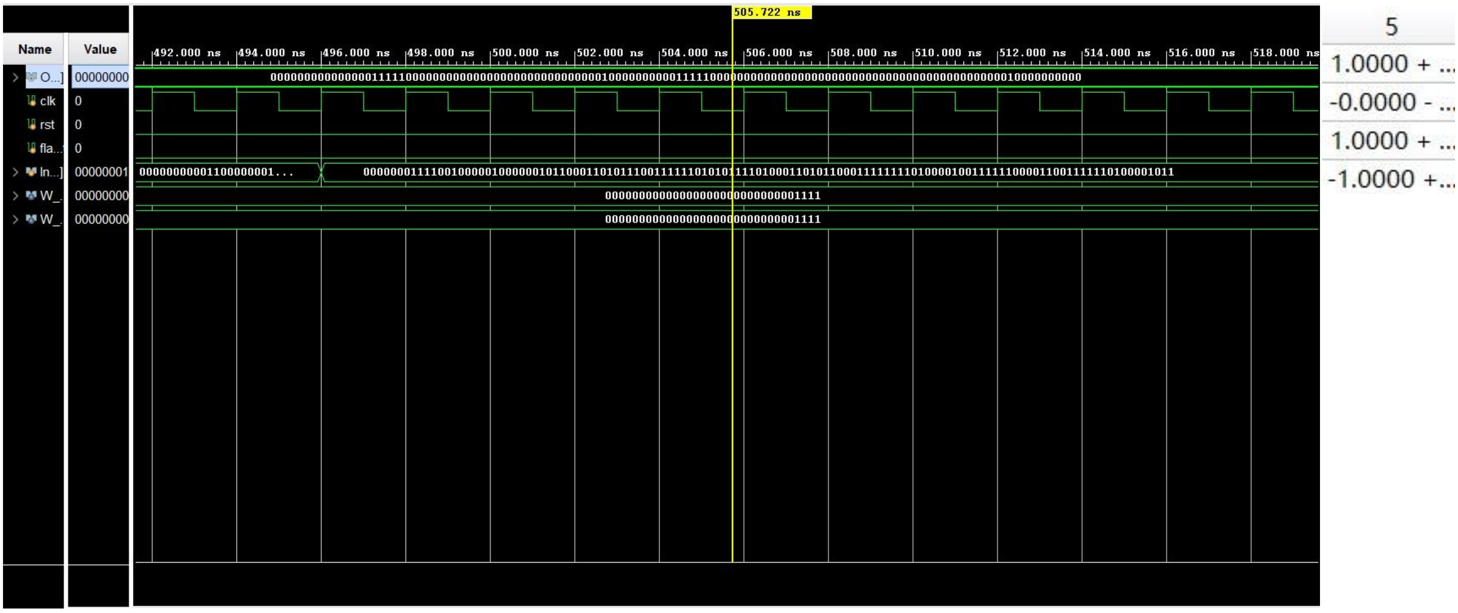
(3)Symbol 2



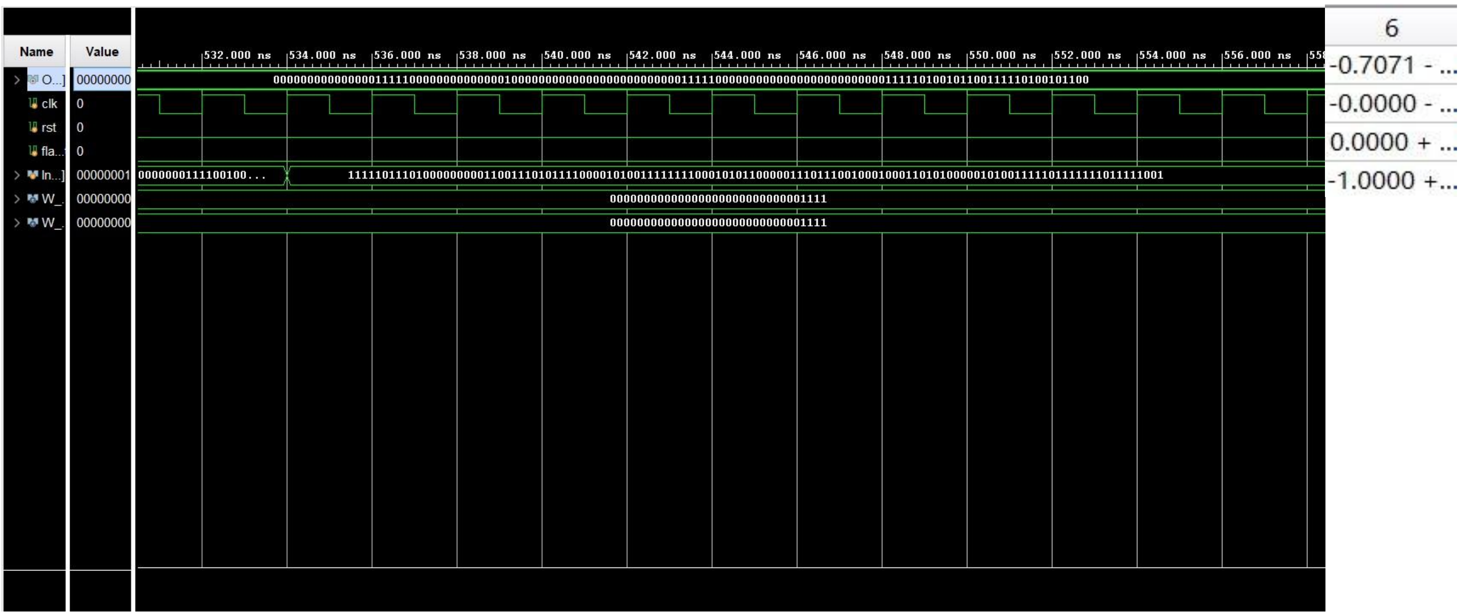
(4)Symbol 3



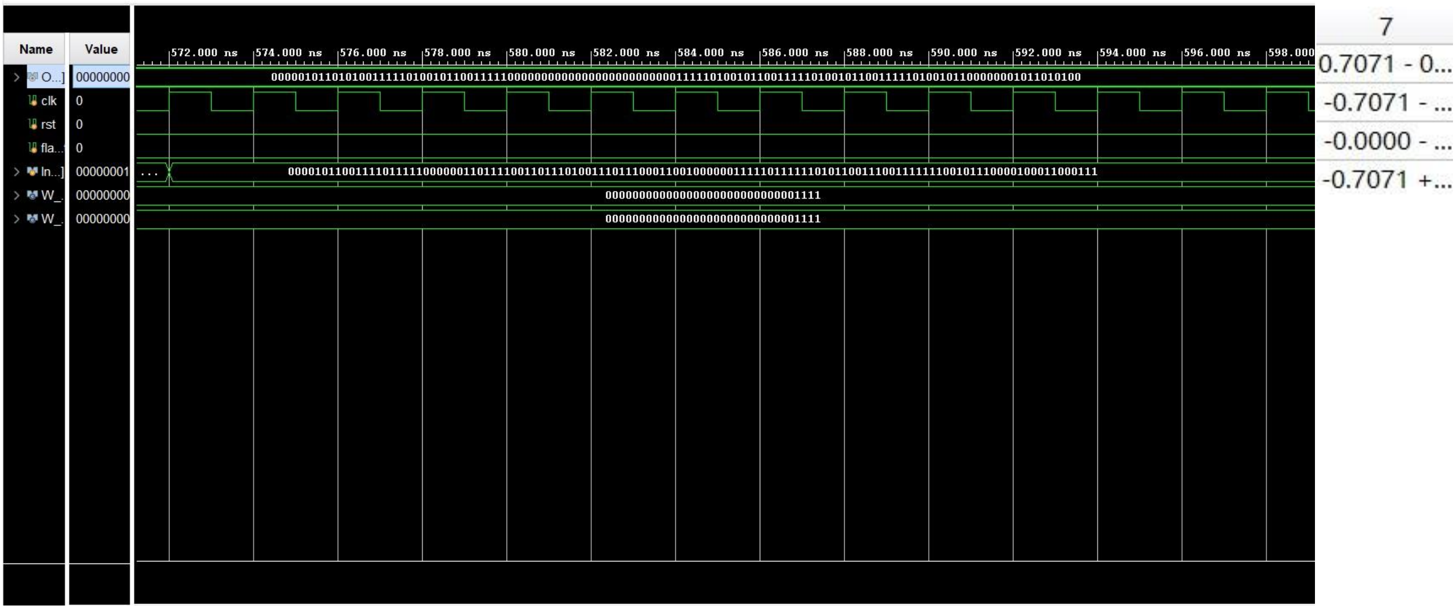
(5)Symbol 4



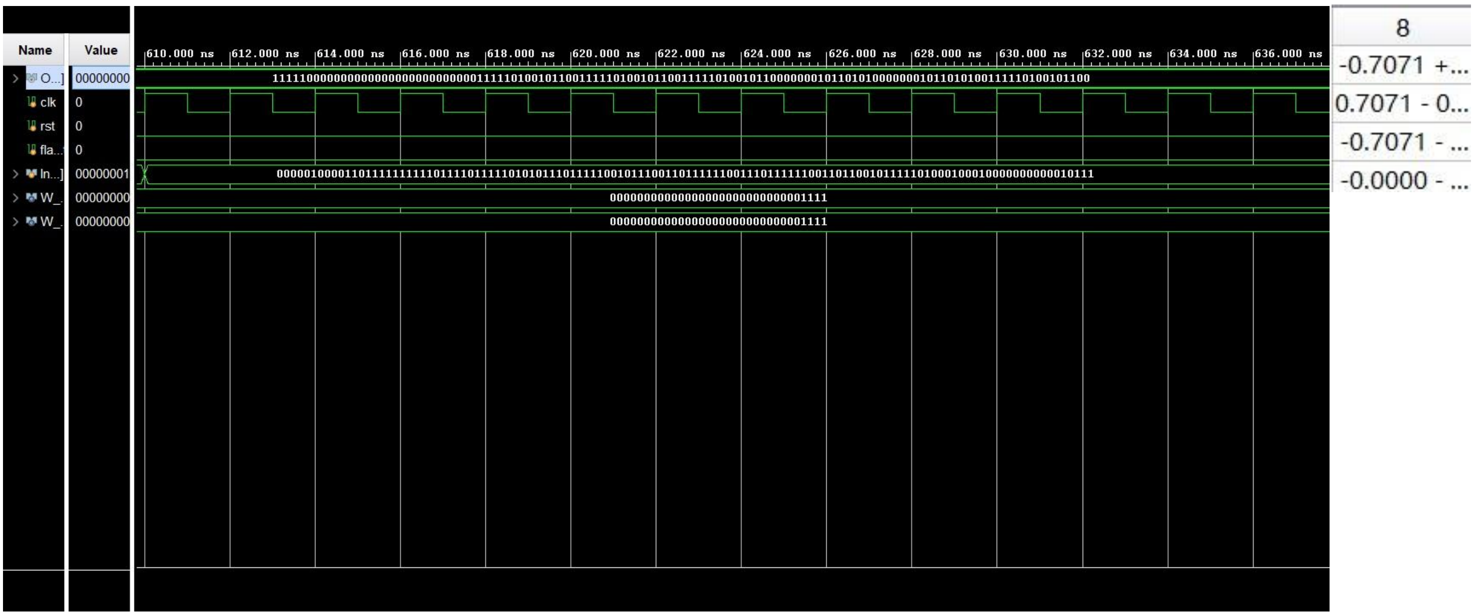
(6)Symbol 5



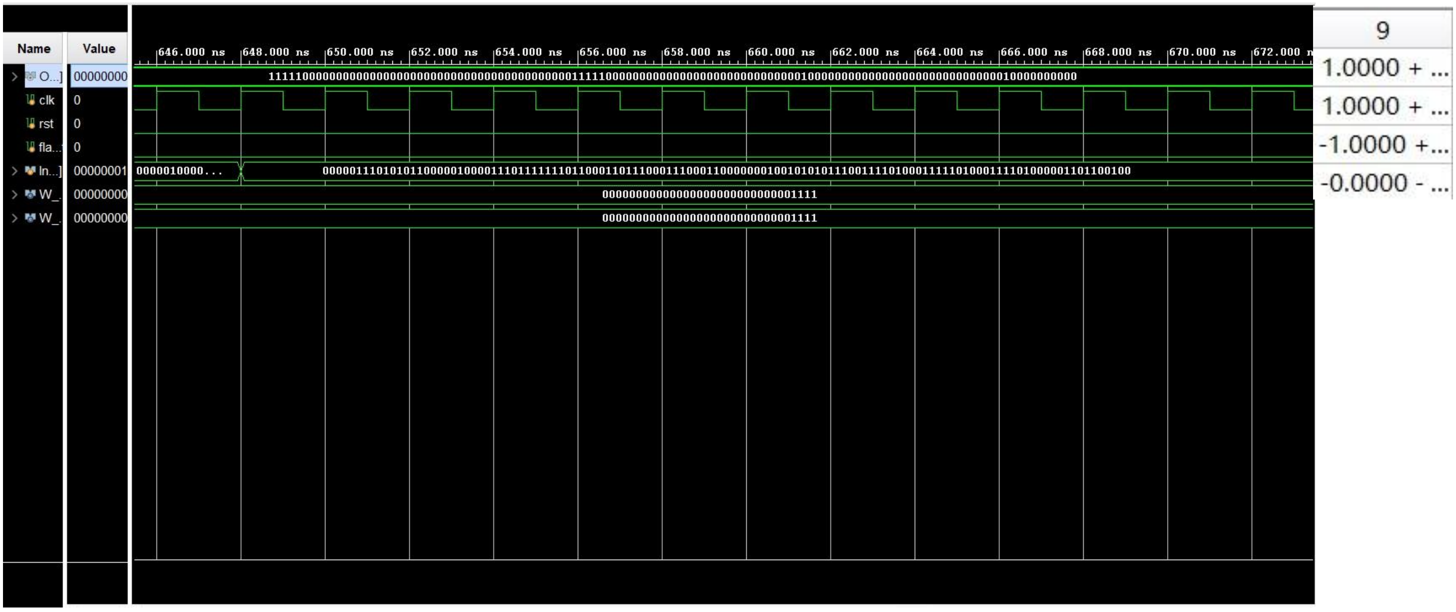
(7)Symbol 6



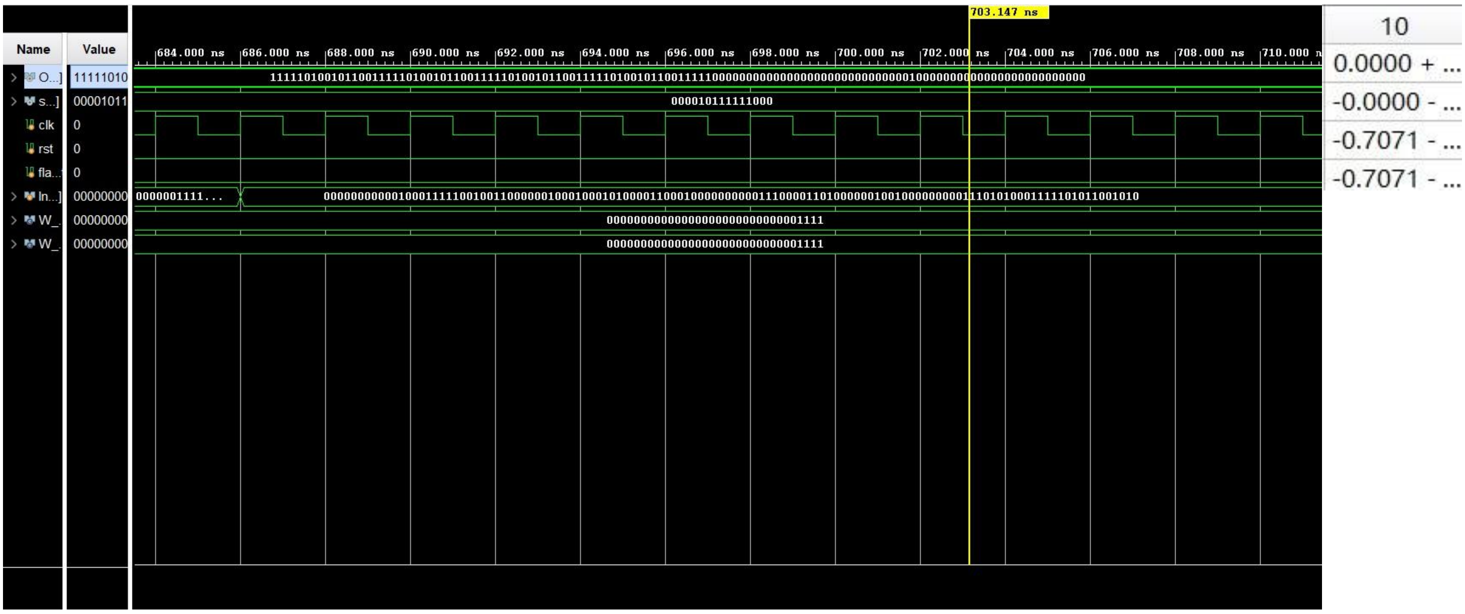
(8)Symbol 7



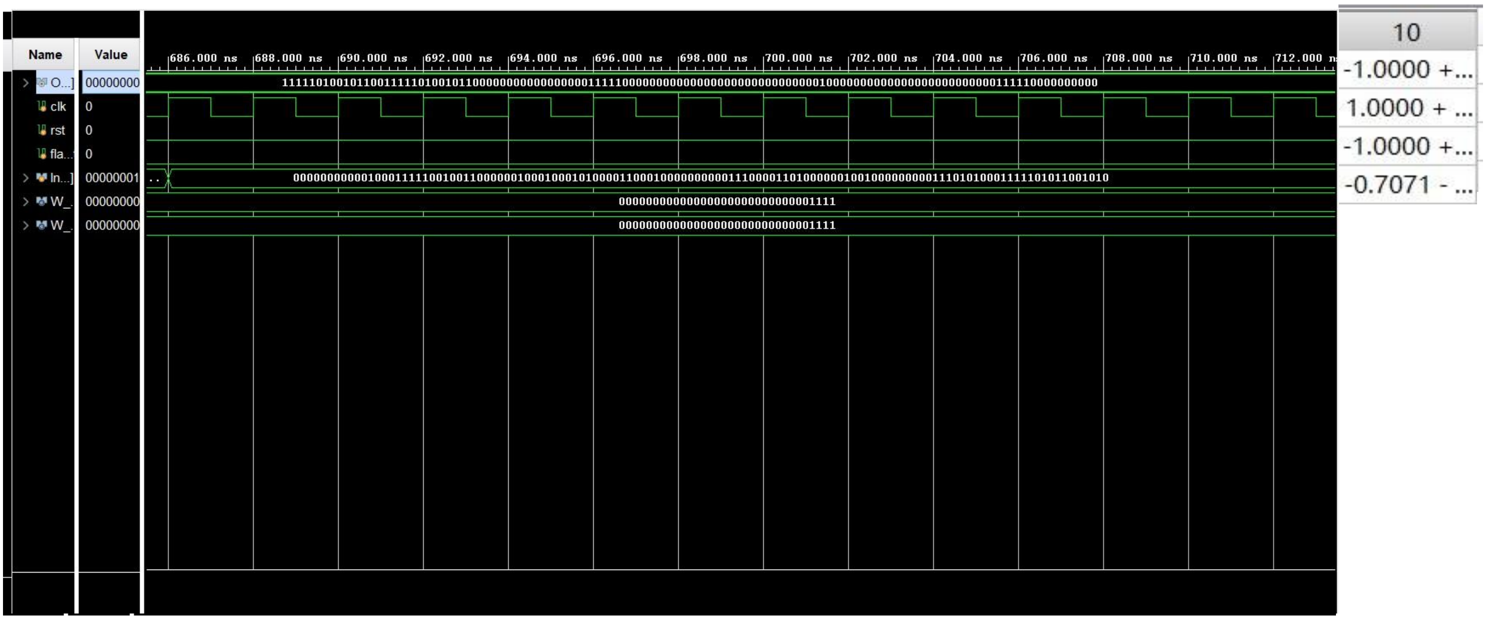
(9)Symbol 8



(10)Symbol 9



(11)Symbol 10



(12)Symbol 11

We can obviously observe that all detected symbol are corresponded with the original symbol.

5.(b)

Setting:

Ts:2ns

Constellation point:

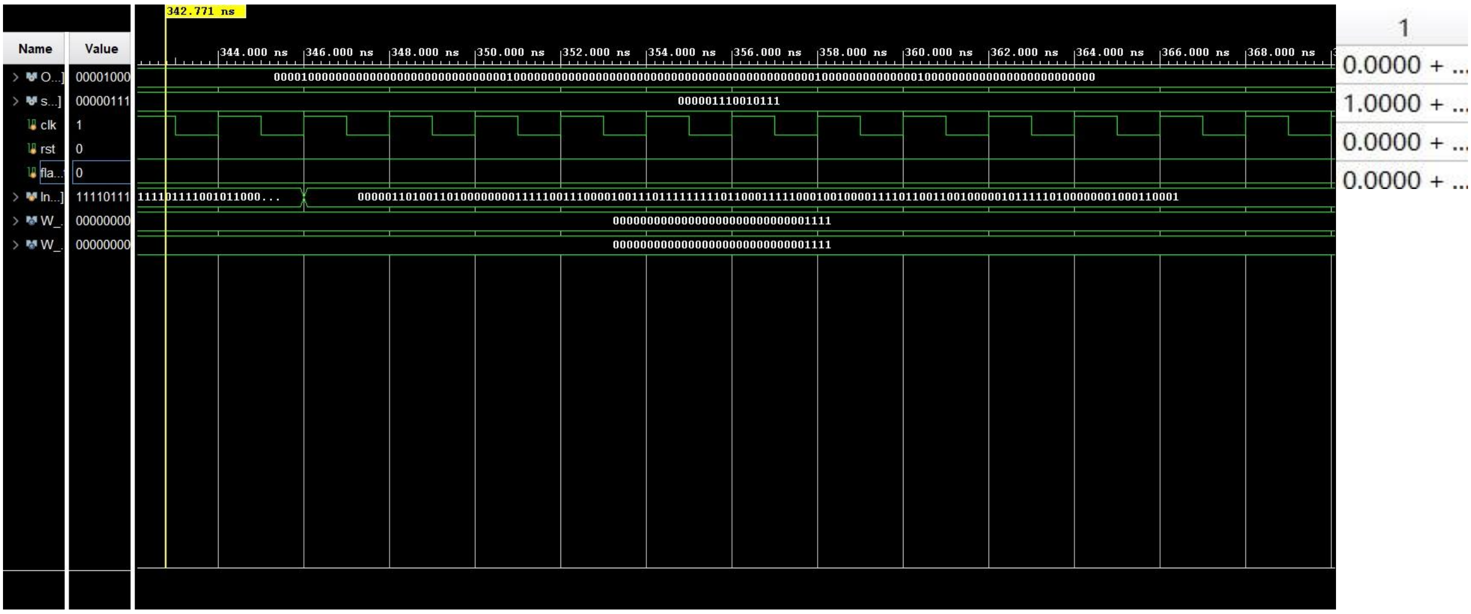
```
    // 8-PSK 星座圖的實部 (cos(θ))
PSK_8_lut_real[0] <= 15'b000010000000000;  // cos(0) = 1
PSK_8_lut_real[1] <= 15'b000001011010100;  // cos(π/4)=0.707
PSK_8_lut_real[2] <= 15'b000000000000000;  // cos(π/2) = 0
PSK_8_lut_real[3] <= 15'b111110100101100;  // cos(3π/4)=-0.707
PSK_8_lut_real[4] <= 15'b111110000000000;  // cos(π) = -1
PSK_8_lut_real[5] <= 15'b111110100101100;  // cos(5π/4)=-0.707
PSK_8_lut_real[6] <= 15'b000000000000000;  // cos(3π/2) = 0
PSK_8_lut_real[7] <= 15'b000001011010100;  // cos(7π/4)=0.707

        // 8-PSK 星座圖的虛部 (sin(θ))
PSK_8_lut_imag[0] <= 15'b000000000000000;  // sin(0) = 0
PSK_8_lut_imag[1] <= 15'b000001011010100;  // sin(π/4)=0.707
PSK_8_lut_imag[2] <= 15'b000010000000000;  // sin(π/2) = 1
PSK_8_lut_imag[3] <= 15'b000001011010100;  // sin(3π/4)=0.707
PSK_8_lut_imag[4] <= 15'b000000000000000;  // sin(π) = 0
PSK_8_lut_imag[5] <= 15'b111110100101100;  // sin(5π/4)=-0.707
PSK_8_lut_imag[6] <= 15'b111110000000000;  // sin(3π/2) = -1
```

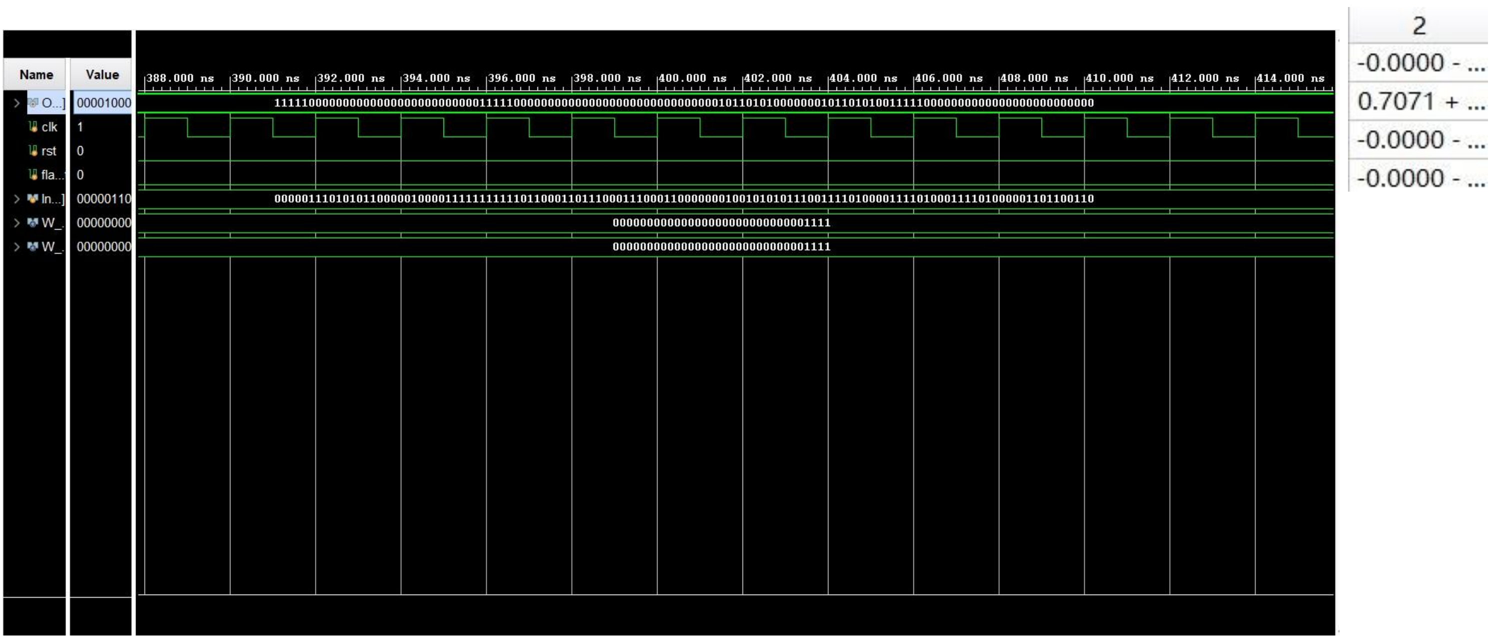
```
PSK_8_lut_imag[7] <= 15'b111110100101100;  // sin(7π/4)=-0.707
```

Simulation Results:

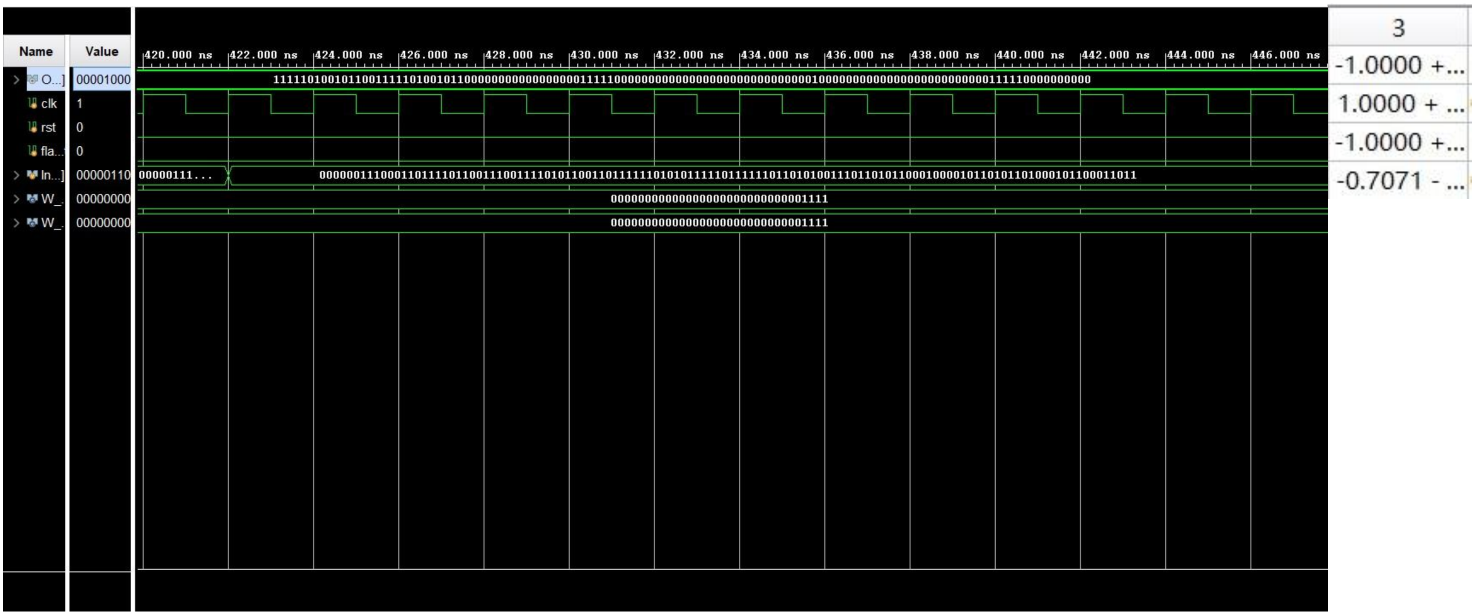
(1)Entire timingdiagram



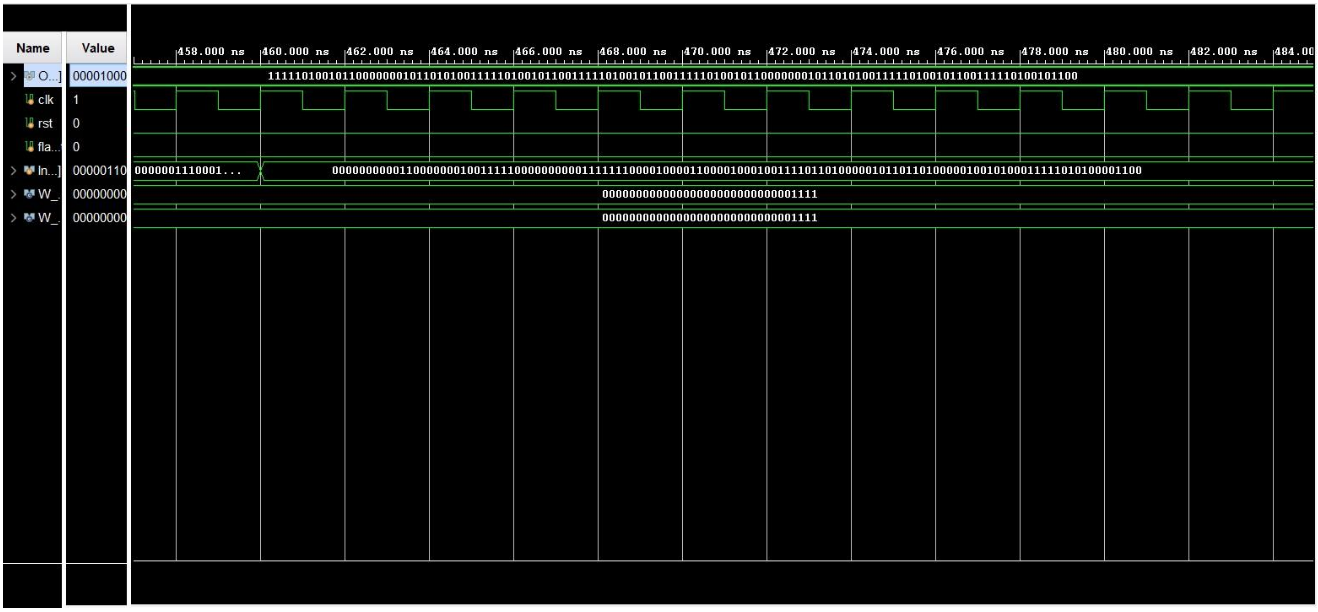
(2)Symbol 1



(3)Symbol 2



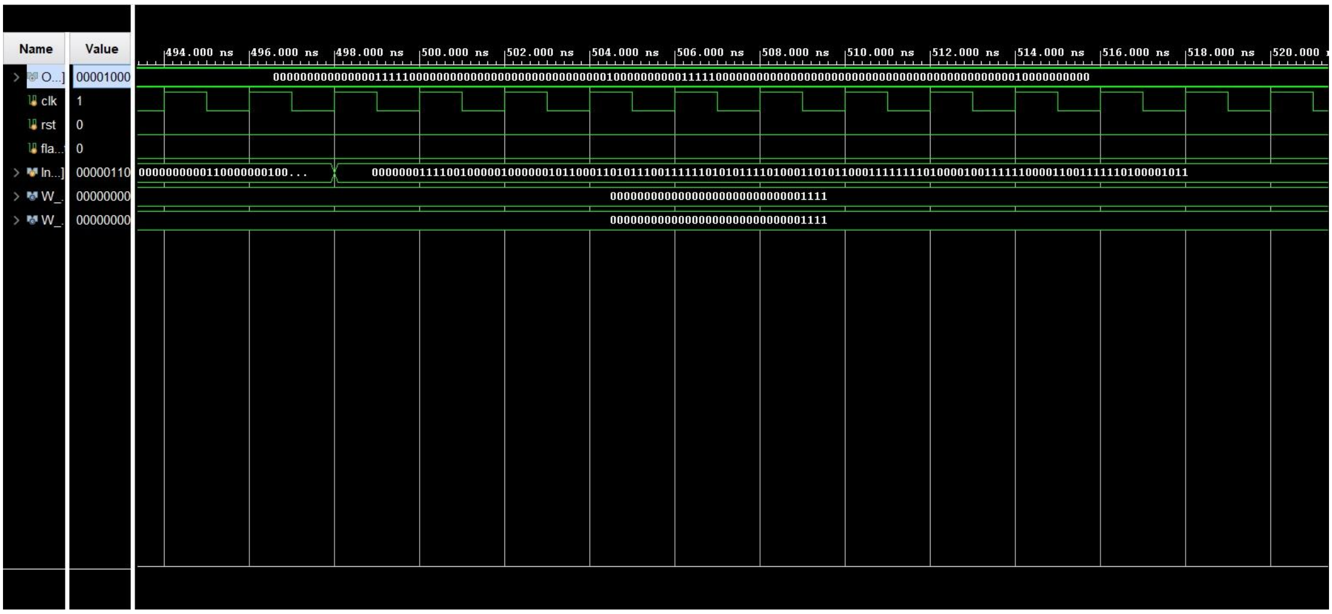
(4)Symbol 3



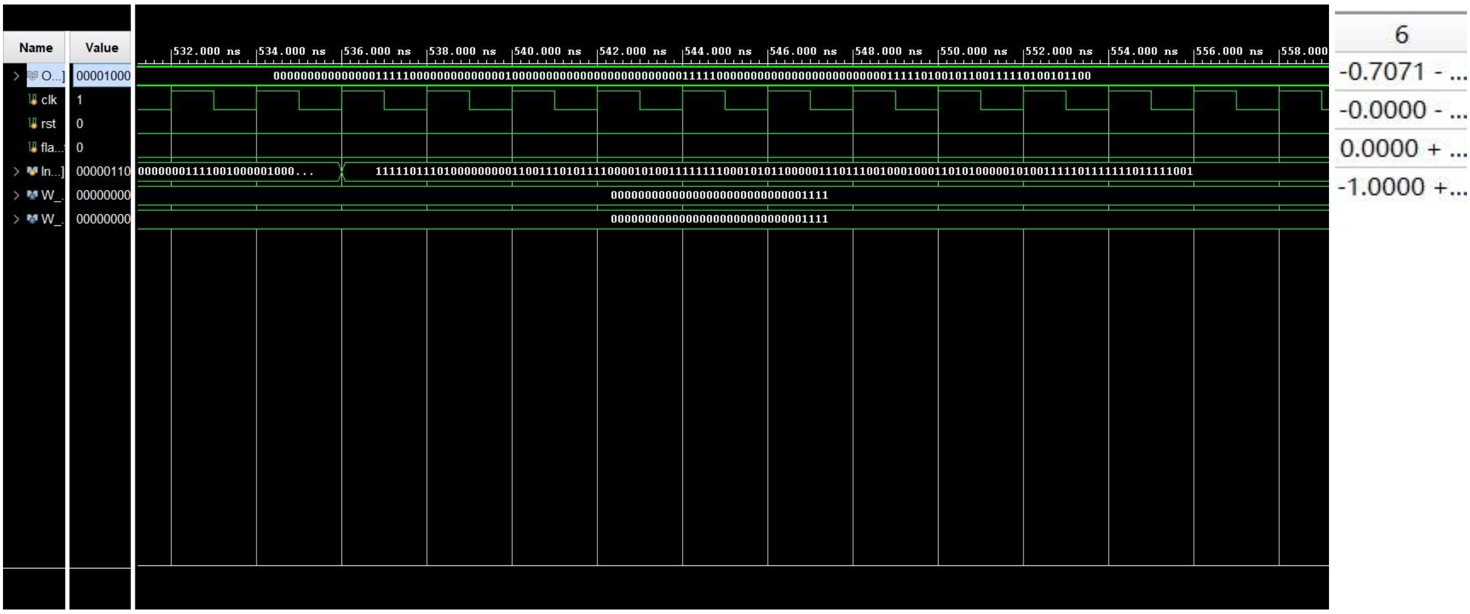
(5)Symbol 4

4
-0.7071 - ...
0.7071 - 0...

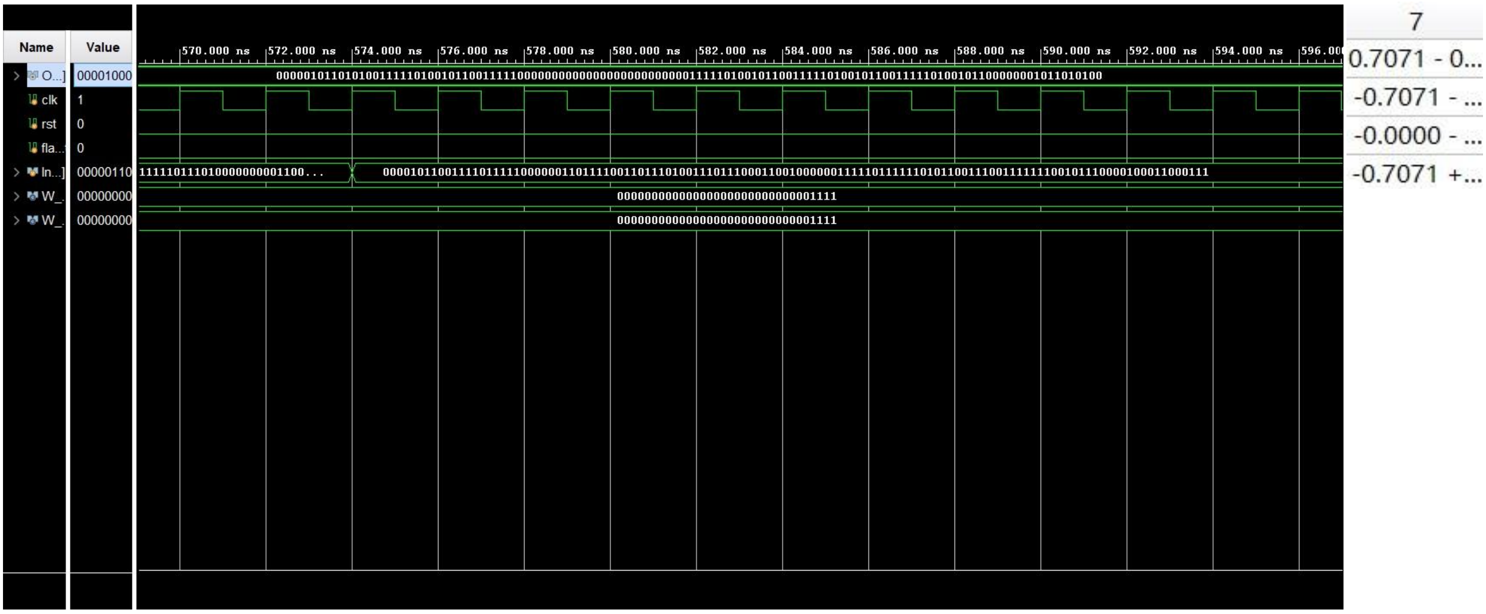
5
1.0000 + ...
-0.0000 - ...
1.0000 + ...
-1.0000 +...



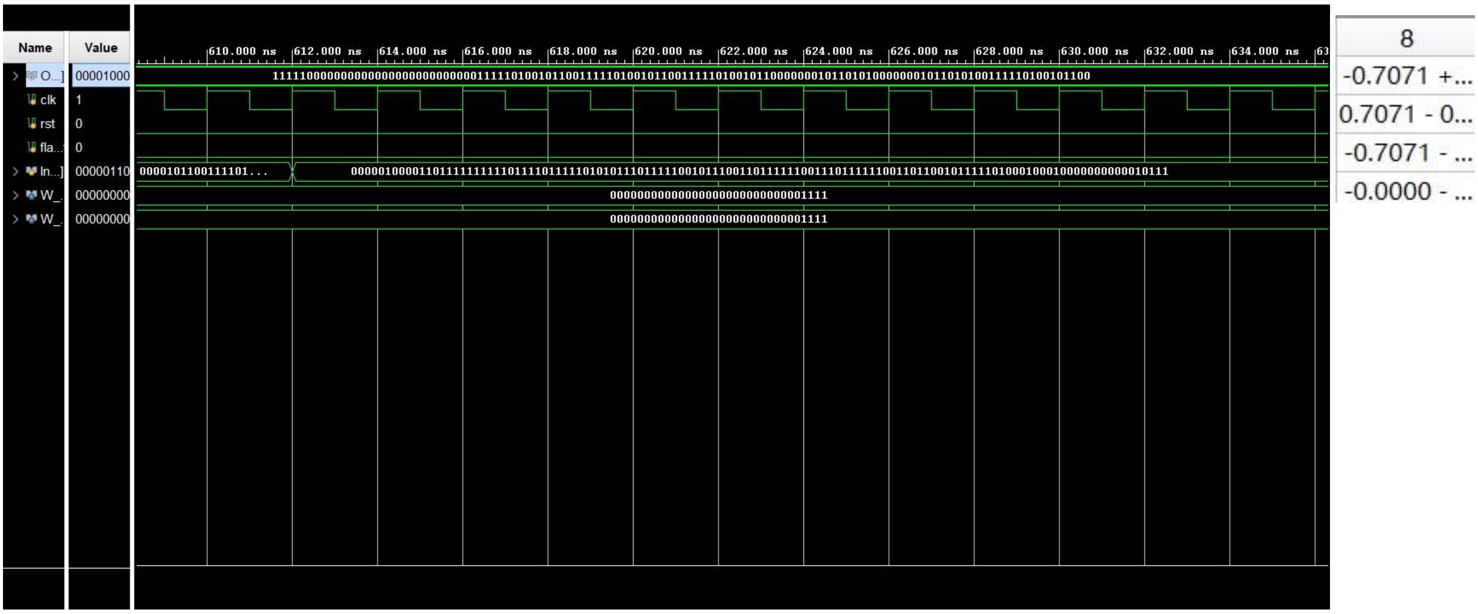
(6)Symbol 5



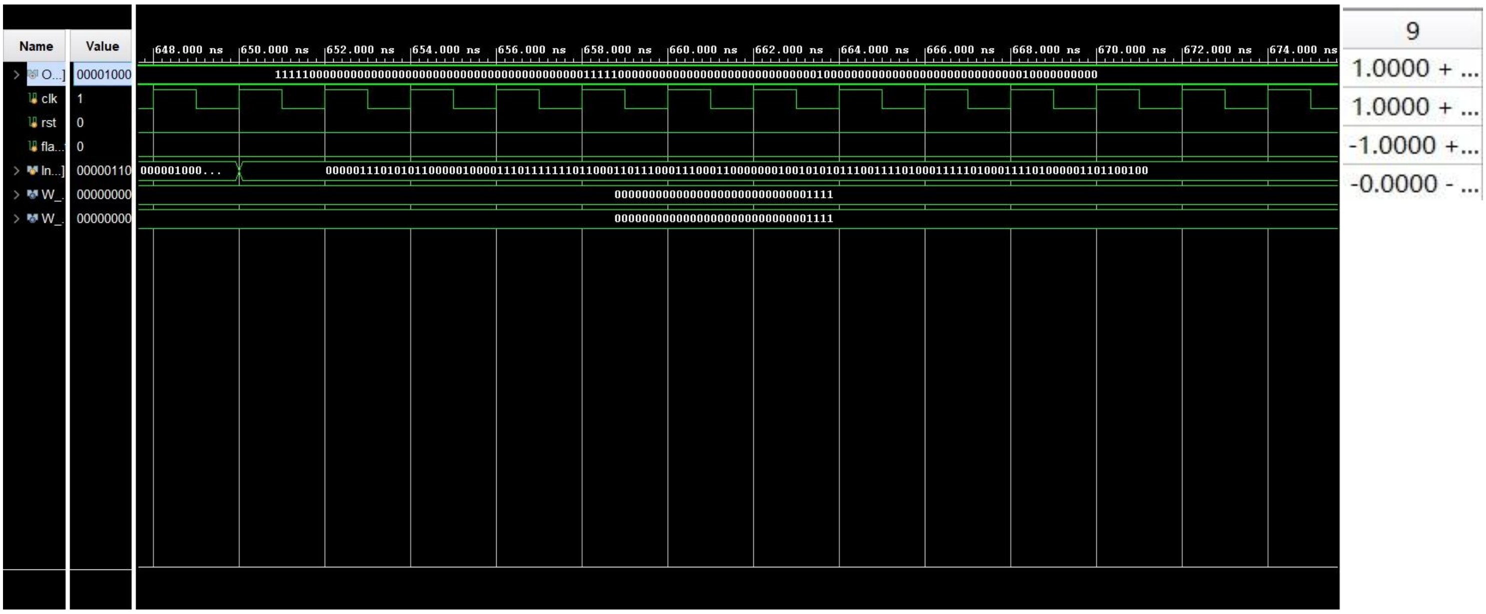
(7)Symbol 6



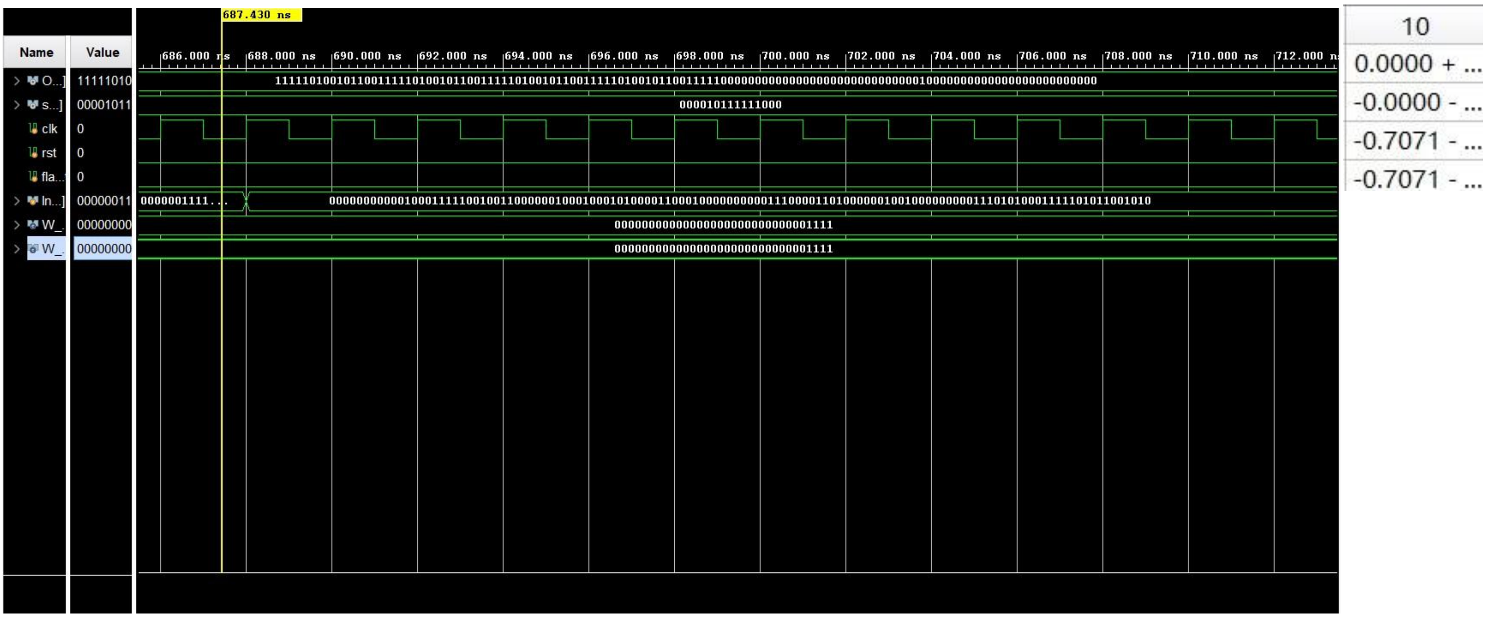
(8)Symbol 7



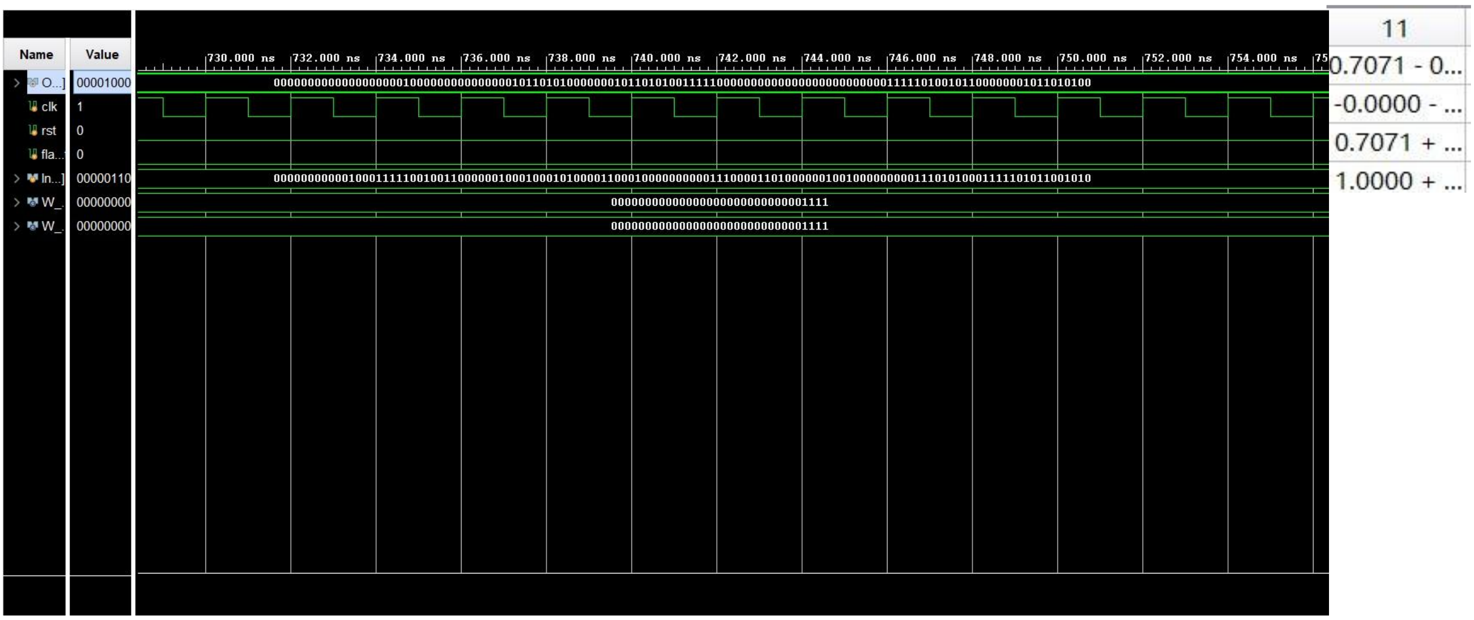
(9)Symbol 8



(10)Symbol 9



(11)Symbol 10



(12)Symbol 11

We can obviously observe that all detected symbol are corresponded with the original symbol.

L=16.5 clock cycles

M=190 clock cycles

M_PRIME=206.5 clock cycles

5.(c)

Ts=2ns

0.1MTs=0.1*190*2ns=38ns

6

We use L1 norm to calculate distance in our method

1	'0000001010111100'	0000001110010111
2	'0000010001110000'	0000010111001010
3	'0000010000110010'	0000010100110010
4	'0000010100101111'	0000011011011100
5	'0000001110111001'	0000010010110110
6	'0000010001011000'	0000010111010001
7	'0000001111000100'	0000010100101000
8	'0000100110001100'	0000101010100010
9	'0000010100001010'	0000011000001001
10	'0000010001011111'	0000010111111000
11	'0000010000101111'	0000010111111000

Left side is fixed simulation,right side is hardware simulation

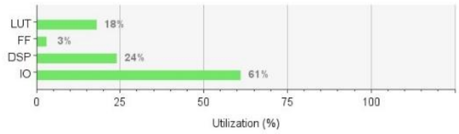
It can be observed that the integer parts are the same, but the decimal parts differ, which is presumed to be caused by precision issues.

7

Utilization:

Summary

Resource	Utilization	Available	Utilization %
LUT	24177	134600	17.96
FF	7370	269200	2.74
DSP	180	740	24.32
IO	243	400	60.75



Timing report:

Setup

Worst Negative Slack (WNS):	-107.593 ns
Total Negative Slack (TNS):	-60746.846 ns
Number of Failing Endpoints:	18542
Total Number of Endpoints:	21398

Hold

Worst Hold Slack (WHS):	-0.163 ns
Total Hold Slack (THS):	-41.663 ns
Number of Failing Endpoints:	610
Total Number of Endpoints:	21398

Pulse Width

Worst Pulse Width Slack (WPWS):	-0.155 ns
Total Pulse Width Negative Slack (TPWS):	-0.155 ns
Number of Failing Endpoints:	1
Total Number of Endpoints:	7267

Timing constraints are not met.

NA:

Name	Slice LUTs (134600)	Slice Registers (269200)	F7 Muxes (67300)	DSPs (740)	Bonded IOB (400)	BUFGCTRL (32)
test5	24177	7370	1	180	243	2
t0 (decoder)	24066	6208	1	180	0	0

$$NA=24177+180*280+7370=81947$$

8

$$AT\_product=81947*38*10^{(-9)}=3.113986*10^{(-3)}$$

9

Innavation of our design:

In this final report, we adopted the 4_best BFS algorithm. However, unlike the original, we used the 1-norm to compare distances instead of the 2-norm. At the same time, we also updated the scheduling method, simplifying the time required for comparisons from 32**32 to 4**31 . Additionally, we used only 15 bits to store the symbol values, instead of the IEEE standard 32 bits. This approach helps to simplify the computation time.

working items and weightings of two members.

賴柏睿:matlab algorithm design,Verilog sorting circuit design(50%)

黃琮凱:matlab algorithm design,Verilog circuit design(50%)