**2.5 PS/2 Keyboard Functional Unit**

PS/2 is an interface for keyboards and mice to PC via a 6-pin Mini-DIN connector. The system must provide the keyboard or mouse with 5V source and ground connections. Communication occurs over two wires, a clock line and a data line. The clock has a frequency between 10 kHz and 16.7 kHz. The data has a start bit, which is set as logic low, one byte of data, a parity bit, and a stop bit, which is set as logic high. Each bit is read on the falling edge of the clock signal.

The input and output speciﬁcations are defined below, and a block diagram of the unit follows in Figure 1. In the figure, ps2\_data and ps2\_clk first synchronized and debounced. The data signal is then loaded to a shift register on falling edges of the PS/2 clock so that the flip flops can store the data. When the data transmission is finished and the clock remains high for more than 55us, the time longer than half of the worst-case PS/2 clock period, idle counter sends logic high to the AND gate. If there is no error and the signal from the idle counter is high, ps2\_code\_new becomes logic high indicating that the block is ready for the next signal. The ps2\_code[7..0] gets ps2\_word[8..1] because flip flops in that position contain the data. Figure 2 shows the block symbol of the PS/2 Keyboard functional unit and Figures from 3 to 10 show the simulation of the keyboard.

**• Inputs:** clk, ps2\_data, ps2\_clk

**• Outputs:** ps2\_code[7..0], ps2\_code\_new

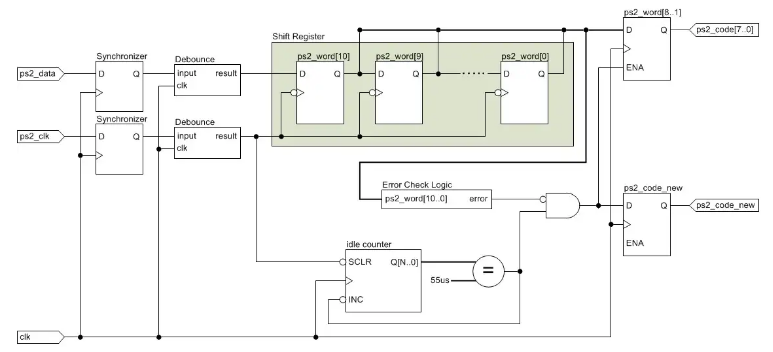


Figure 1: The logic design of the PS/2 Keyboard functional unit used in the ﬁnal design

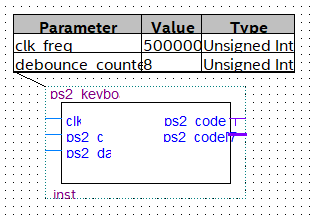


Figure 2. The block symbol of the ps2\_keyboard block used to further simulate the PS2 Keyboard functional unit

The waves indicate clock with 50MHz, ps2\_clk, ps2\_data, ps2\_code\_new, and ps2\_code[7..0] in sequence. The signals for ps2\_clk and ps2\_data are gotten from the csv file provided. After ps2\_word[10..0] stores the signal of ps2\_data at the falling edge of the ps2\_clk, ps2\_code\_new becomes 1 and ps2\_word[8..1] is sent to ps2\_code[7..0] if there is no error.

Each figure below shows the simulation of pushing the keys of ‘a’, ‘s’, ‘d’, ‘f’, ‘w’, ‘e’, ‘r’, and ‘t’. The keys in PS/2 Keyboard have make code in hexadecimal. The code for the keys used in this project is 0x1C, 0x1B, 0x23, 0x2B, 0x1D, 0x24, 0x2D, and 0x2C respectively. The codes are shown in the simulation in the binary format.

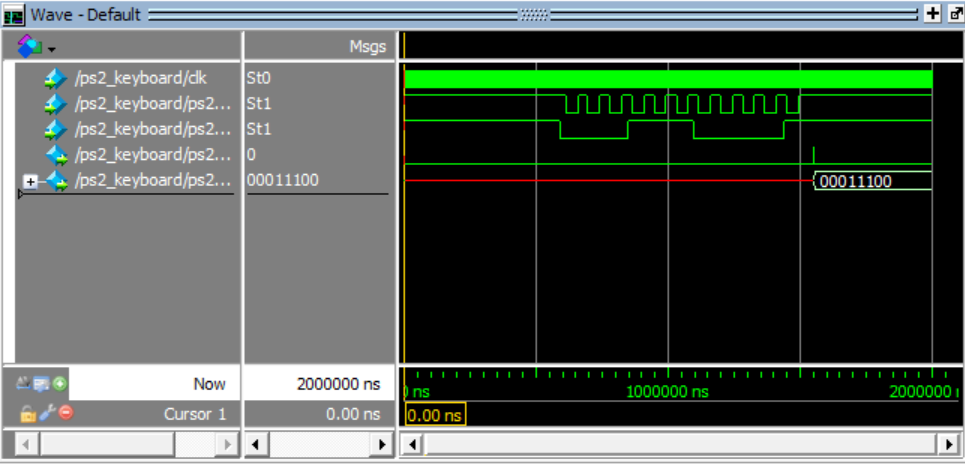


Figure 3: The simulation results of pushing ‘a’ on the keyboard.

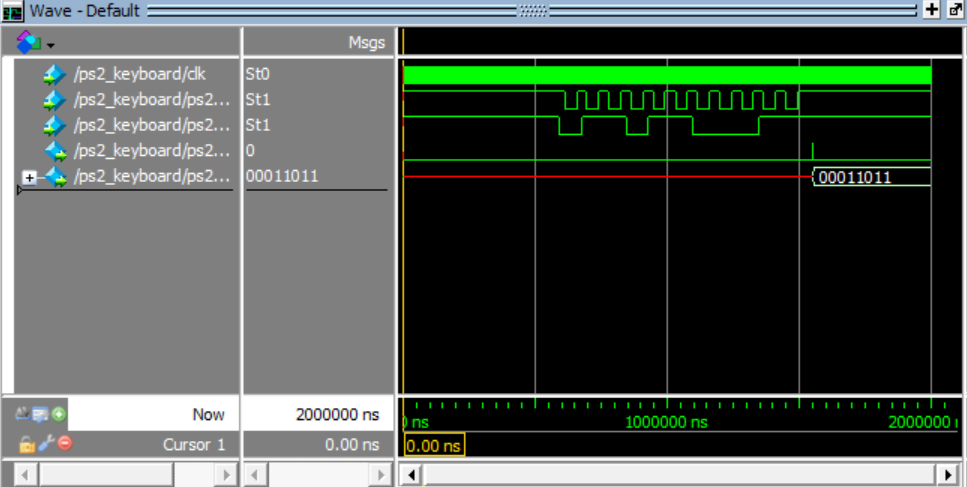


Figure 4: The simulation results of pushing ‘s’ on the keyboard

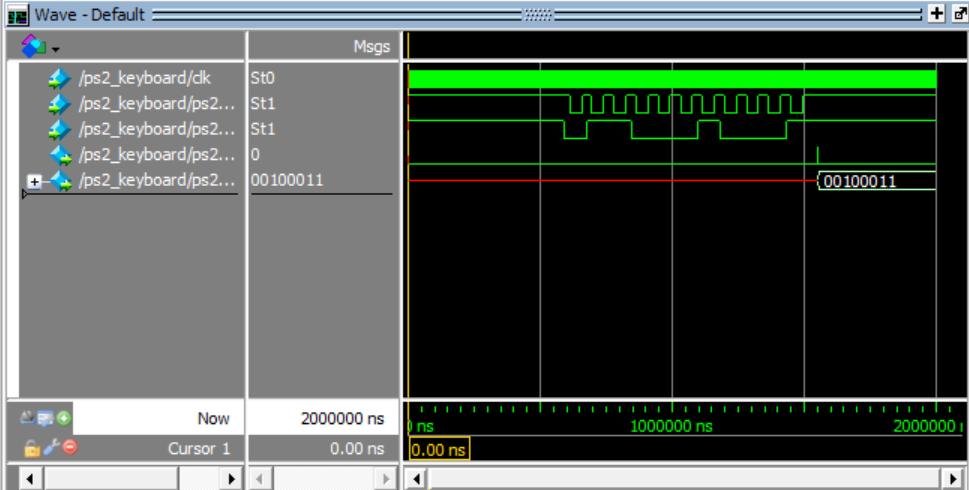


Figure 5: The simulation results of pushing ‘d’ on the keyboard

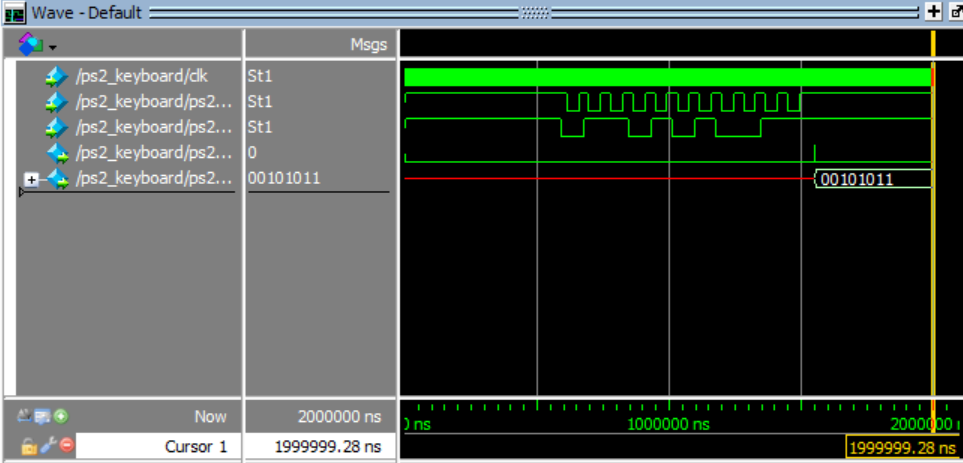


Figure 6: The simulation results of pushing ‘f’ on the keyboard

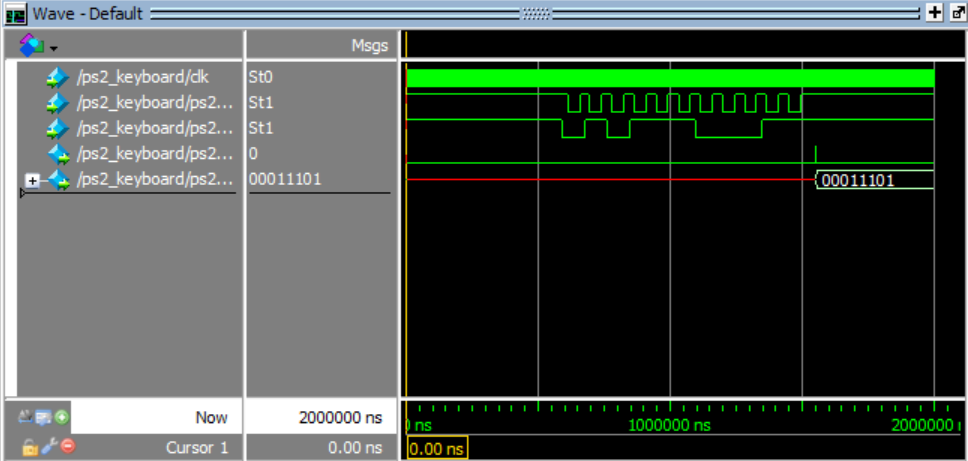


Figure 7: The simulation results of pushing ‘w’ on the keyboard

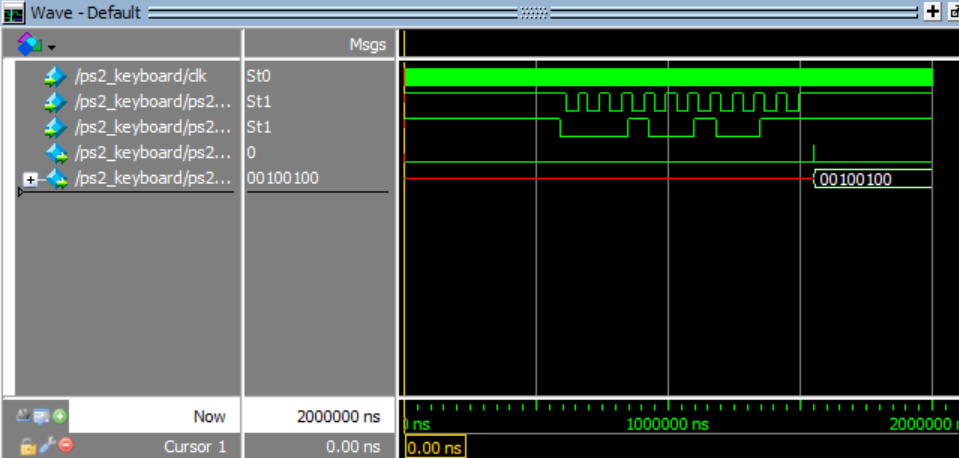


Figure 8: The simulation results of pushing ‘e’ on the keyboard

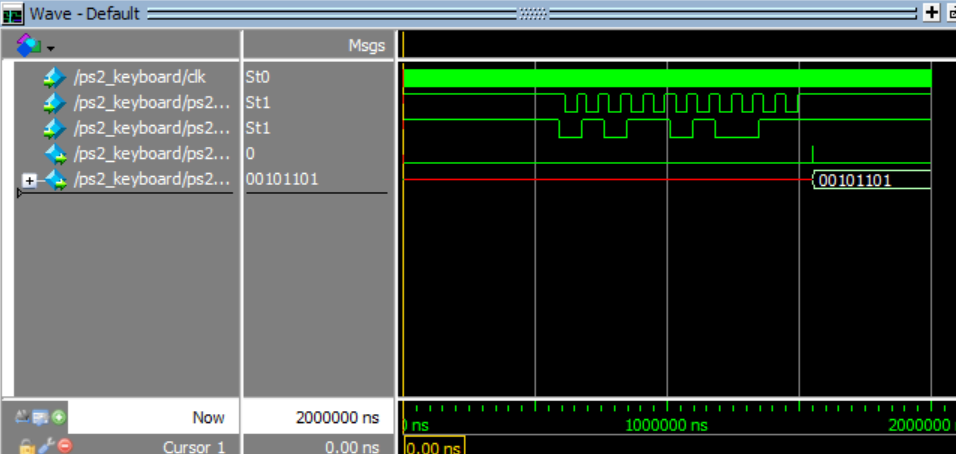


Figure 9: The simulation results of pushing ‘r’ on the keyboard

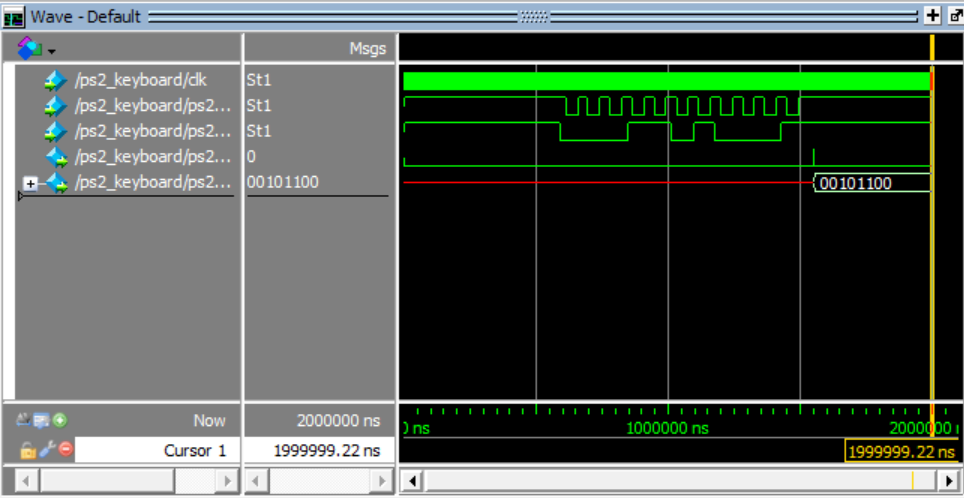


Figure 10: The simulation results of pushing ‘t’ on the keyboard

2.5.2 Debounce

When using mechanical switches, the signals often rebound, or bounce, off one another before settling into a stable state. The debounce component is used for successfully sending the data in a stable state. Two flip flops in the design check whether the data changes or not via the XOR gate. If the data changes, the counter block receives the signal to reset the counter. If not, the counter increases until it reaches the specified time and enables the output register.

The input and output speciﬁcations are defined below, a block diagram of the unit follows in Figure 11, and the simulation results for the unit follows in Figure 12. According to simulation, when the data remains stable for 40 ns, the data is sent to the last flip flop and the result signal is updated.

• Inputs: button

• Outputs: result

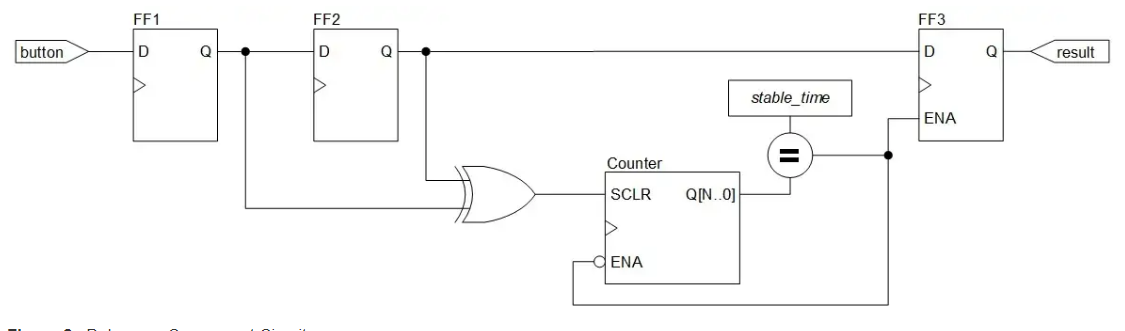


Figure 11: The logic design of the debounce block

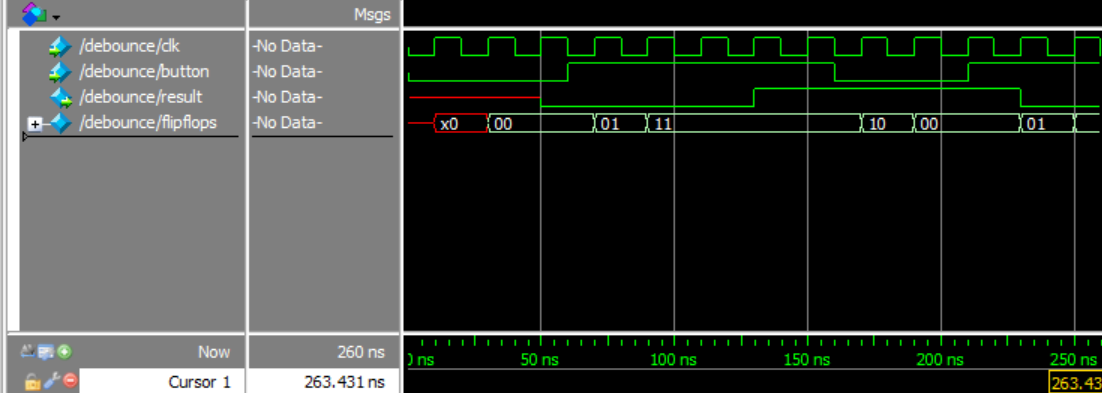


Figure 12: The simulation results of the debounce block

2.6 Square Wave Generator

Square Wave Generator generates square wave and it is used to generate tones in the piano in this project. When ps2\_code[7..0] is read from the PS/2 Keyboard functional unit, a decoder decodes the data into three-digit binary data. The SquareWaveGenerator block receives the data and converts it into the square wave with specific Hz that indicates notes in the piano. The input and output speciﬁcations are defined below, and a block diagram of the unit follows in Figure 12.

• Inputs: clk\_50MHz, ps2\_code[7..0], en

• Outputs: sqWave

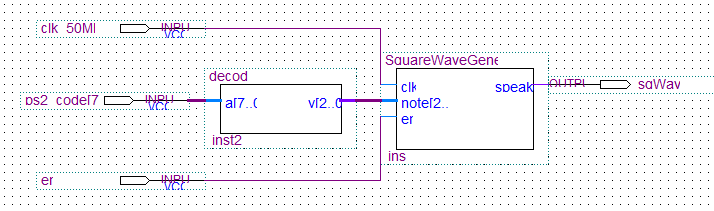


Figure 12: The logic design of the Square Wave Generator functional unit used in the ﬁnal design

2.6.1 Square Wave Generator

The square wave generator block converts eight types of 3-bit data from 0b000 to 0b111 into a square wave. The internal counter div\_cnt is compared to max\_count which indicates the period taken to reach specific hz. If div\_cnt reaches max\_count, tmp\_clk changes to make a square wave. The en signal determines whether the tmp\_clk is sent to speaker or not. To make it clear, max\_count for middle C is 95419 because the period for 262Hz is 3816973ns. Considering that the clock has the frequency of 50MHz, the signal is read in every 20ns and the square wave should be changed after it passes half of the period. Therefore, after calculating 3816973/20/2=95419. The max\_count for other notes is also derived from this calculation.

The input and output speciﬁcations are defined below, a block symbol of the unit follows in Figure 13, and the simulation results for the unit follows in Figure 14.

• Inputs: clk, note[2..0], en, reset

• Outputs: speak

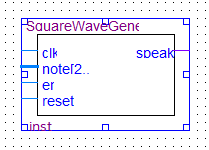


Figure 13. The block symbol of the SquareWaveGenerator block used to further simulate the square wave generator functional unit

The notes from the middle C to the C with next octave have the frequency of 262 Hz, 294 Hz, 330 Hz, 350 Hz, 392 Hz, 440 Hz, 494 Hz, and 523 Hz in sequence. This means that each note has the period of 381673 ns, 3401360 ns, 3030303 ns, 2857142 ns, 2551020 ns, 2272727 ns, 2024291 ns, and 1912045 ns respectively. When they are simulated in the period, the square wave changes only once at the half of the period as shown in the figure below.

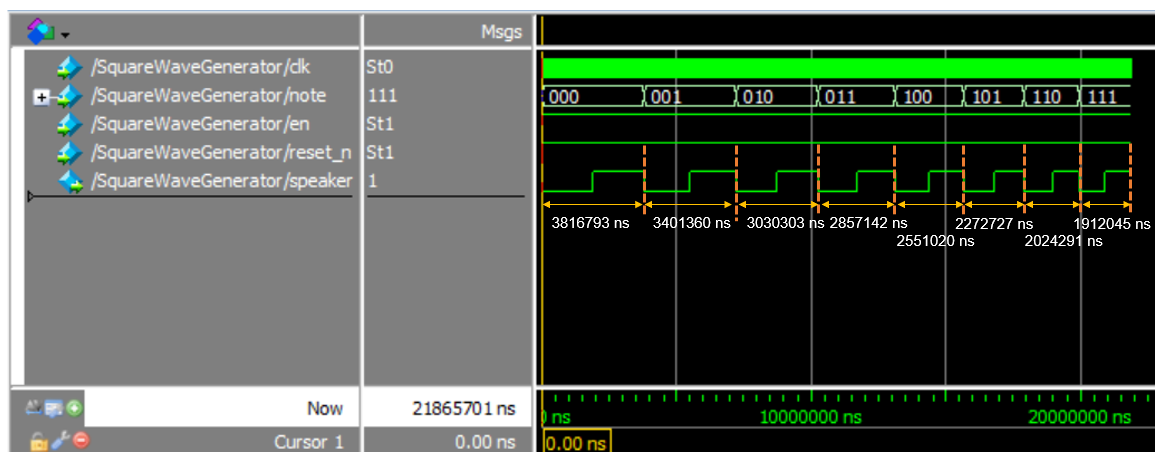


Figure 14: The simulation results of square wave generator block

2.6.2 decoder

The decoder block decodes the data from the PS/2 keyboard to the three-bit data. The input and output speciﬁcations are defined below, a block diagram of the unit follows in Figure 15, and the simulation results for the unit follows in Figure 16. To match the keys in keyboard from ‘a’ to ‘t’ with the piano notes from middle C to the upper octave C, the hexadecimal code of the keyboard is converted to 3-bit data as shown in the simulation.

• Inputs: a[7..0]

• Outputs: y[2..0]

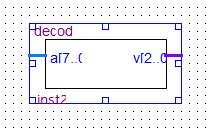


Figure 15. The block symbol of the decoder block used to further simulate the square wave generator functional unit

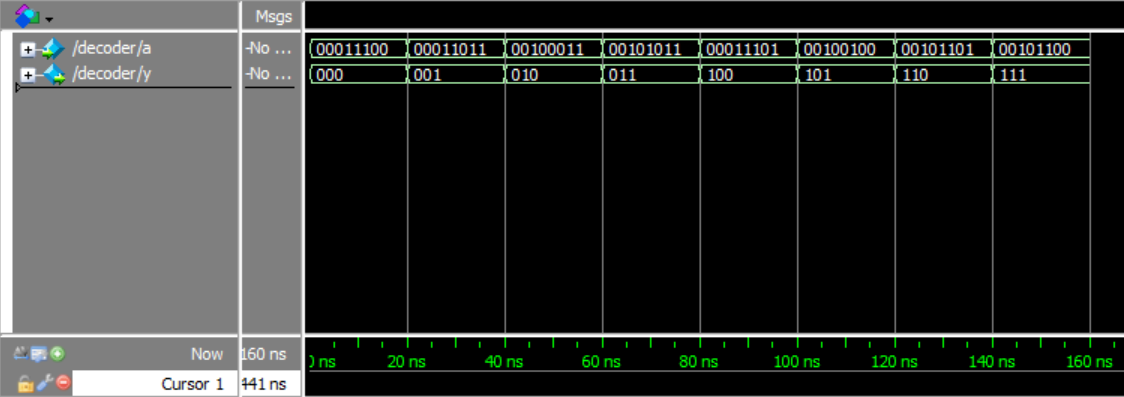


Figure 16: The simulation results of the decoder block