APRIL 2023, 4 BY ACKYE

Smart ZYNQ (SP&SL Edition) Project XI FPGA Hardware Debug ILA Demonstration Feature

WHEN USING FPGA TO DO PROJECTS, MOST OF THE TIME IS SPENT ON DEBUGGING, DEBUGGING IN ADDITION TO THE BOARD ACTUALLY RUNNING RESULTS AND USING VIVADO'S SIMULATION FUNCTION, YOU CAN ALSO USE XILINX'S BUILT-IN ILA FUNCTION TO DEBUG, THIS ARTICLE WILL BRIEFLY DEMONSTRATE XILINX'S ILA FUNCTION.

This article is demonstrated on vivado2018.3, please research for other versions

(Note: The content of this section applies to the boards of Smart ZYNQ SP and SL Edition, if it is Smart ZYNQ Standard Edition, please refer to the corresponding board directory)

The way ILA works, equivalent to putting a small logic analyzer inside the chip, and using the probe of this logic analyzer to grab different signals inside the chip, and the waveform of the signal is collected and uploaded and finally displayed, according to these waveforms, we can judge whether the work of various modules meets expectations, or where the signal is not consistent with the design, so as to achieve the purpose of debugging.

During the working process, ILA automatically temporarily stores the user's signal to BRAM, and then reads it to the PC through JTAG for waveform display.

The way ILA works is a little similar to simulation, but the waveform captured by ILA is the result of the real operation after the code compilation is comprehensively downloaded to the FPGA, and the simulation is the result of the ideal output, and the two will be different

when the program is more complex, or the timing constraints are not done, so when the program does not meet expectations and cannot find the direction, you can use ILA to grab the waveform of the key signal to see, because ILA is the waveform captured in the process of real running, so the result is more convincing.

1. Add ILA module

This article uses a simple project of counters to demonstrate the use of ILA function modules.

This article is demonstrated on vivado2018.3, please research for other versions

1.1 Create a project: (This article only introduces the ILA part, if you are not familiar with the project creation process, you can see Project <>)

Smart ZYNQ (SP&SL version) Project 1 Light up an LED with ZYNQ's PL resources (full graphic)

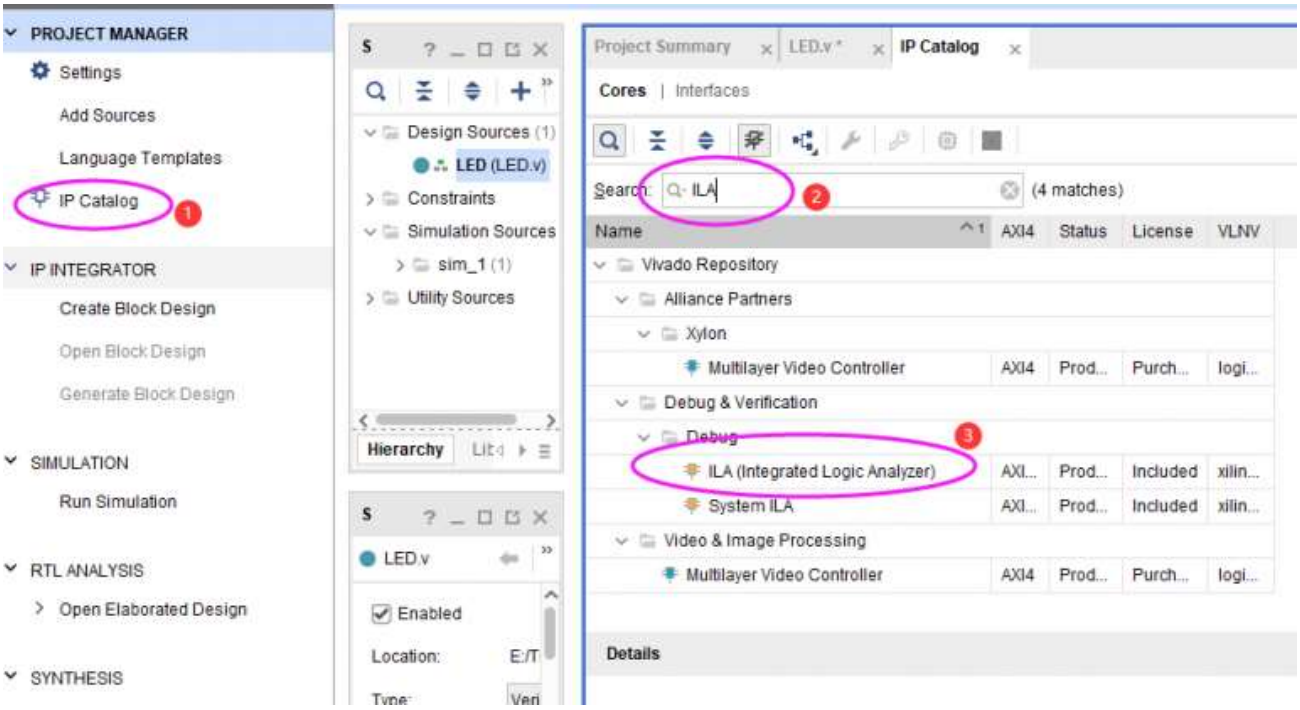
1.2 Code: Here we do a simple function on the code, the FPGA chip internal counter time_count every 100 times the count (0-99) result register flips once (from 1 to 0, or from 0 to 1), and use the ILA module to detect these two signals inside the chip, counter variables (time_count) and result registers (result) **The program is very similar to engineering one, but for demonstration, So the counter time is shortened**

```
`timescale 1ns / 1ps
module ILA_TEST(
     input clk
 );
 parameter T1MS = 7'd99; //50M晶振时钟
 reg [7:0]time_count;//时钟计数器
 reg result;
 always@(posedge clk)
     if(time_count>=T1MS)begin
         time_count<=26'd0;
         result<=~result;
     end
```

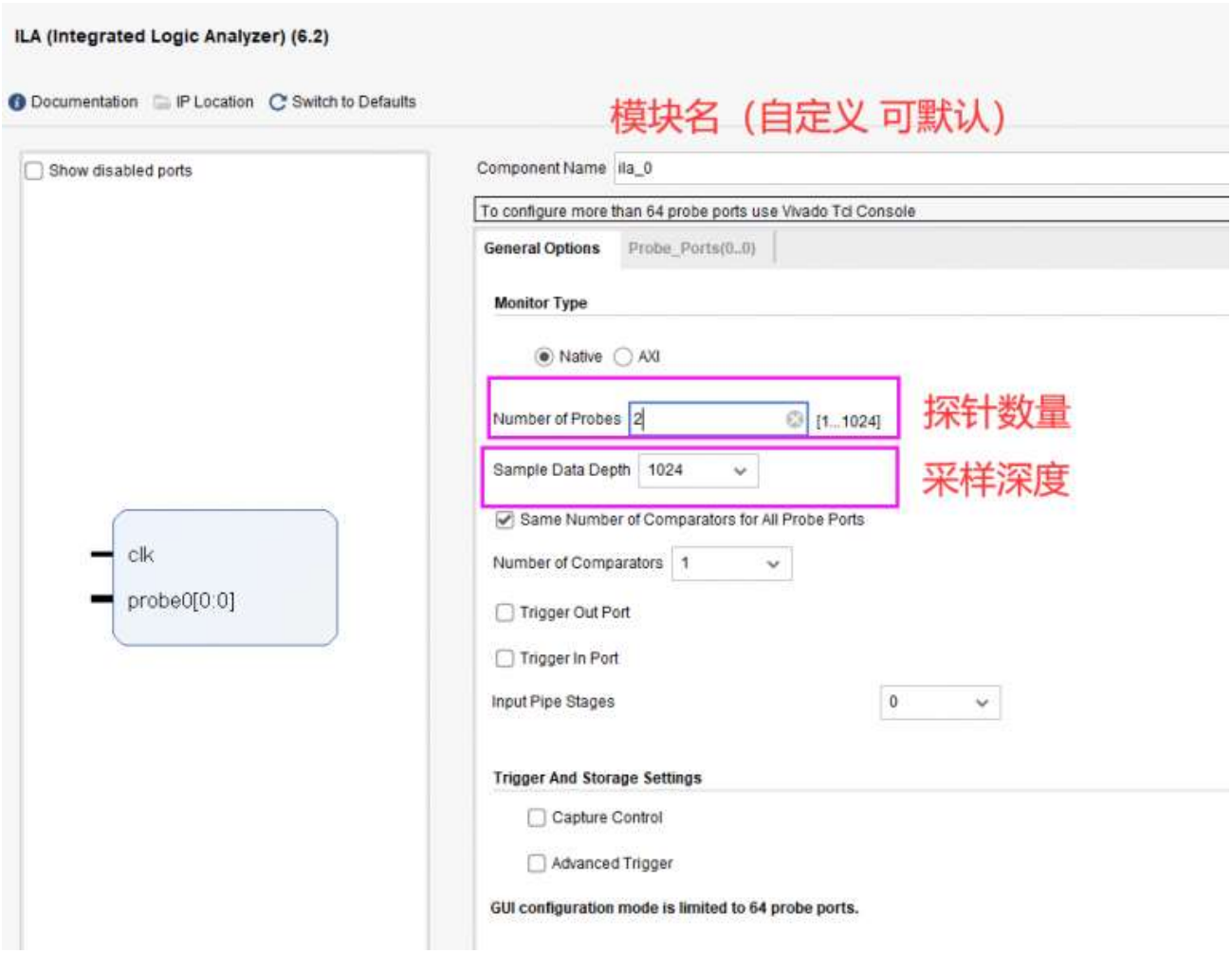
```
        else time_count<=time_count+1'b1;
endmodule
```

1.3 Call the ILA module

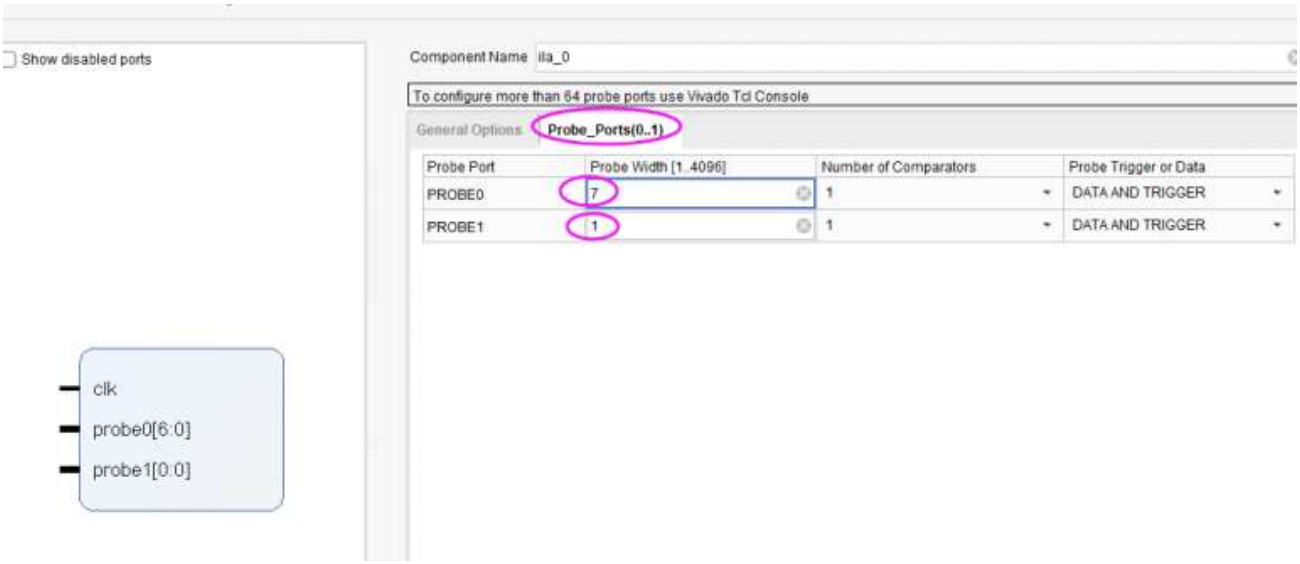
In the vivado software, open the IP Core Directory and search for ILA, as shown in the following figure



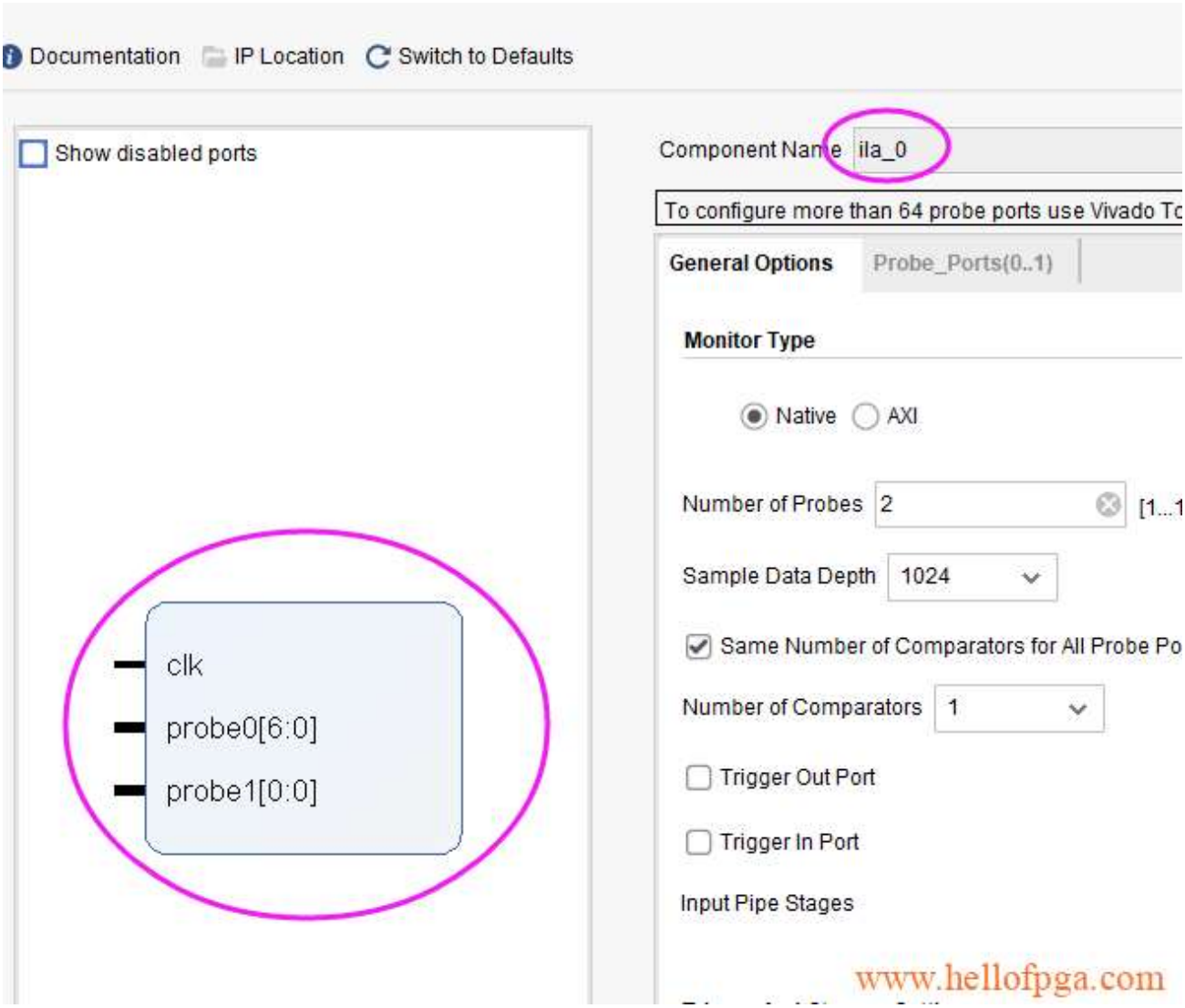
1.4 In the pop-up ILA configuration interface modify the parameters (because we want to collect 2 signals, that is, two probes, so the number of probes is changed to 2, since our count counts from 0-99, the depth of 1024 is enough to sample for several cycles, so the sampling depth here remains unchanged)



1.5 Modify the bit width of the probe, because our time_count is 7 bits, and the result variable is 1 bit, so modify it here to 7 and 1, then click OK, and then select generate in the pop-up window to generate ILA module (it takes a little time)



1.6 Remember the module name shown in the figure below, and the signal composition



1.7 Add the instantiation code of the ILA module to the code we wrote earlier (instantiation is to generate an entity)

```
ila_0 u_0(
.clk(CLK),
.probe0(time_count),
.probe1(result)
) ;
```

The complete code is as follows:

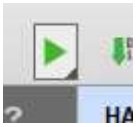
```
`timescale 1ns / 1ps
module ILA_TEST(
     input CLK
 );
 parameter T1MS = 7'd99; //50M晶振时钟
```

```
    reg [7:0]time_count;//时钟计数器
    reg result;
    always@(posedge CLK)
        if(time_count>=T1MS)begin
            time_count<=26'd0;
            result<=~result;
        end
        else time_count<=time_count+1'b1;


ila_0 u_0(
    .clk(CLK),
    .probe0(time_count),
    .probe1(result)
) ;

endmodule
```

To add a constraint file:

```
set_property IOSTANDARD LVCMOS33 [get_ports CLK]
set_property PACKAGE_PIN M19 [get_ports CLK]
```

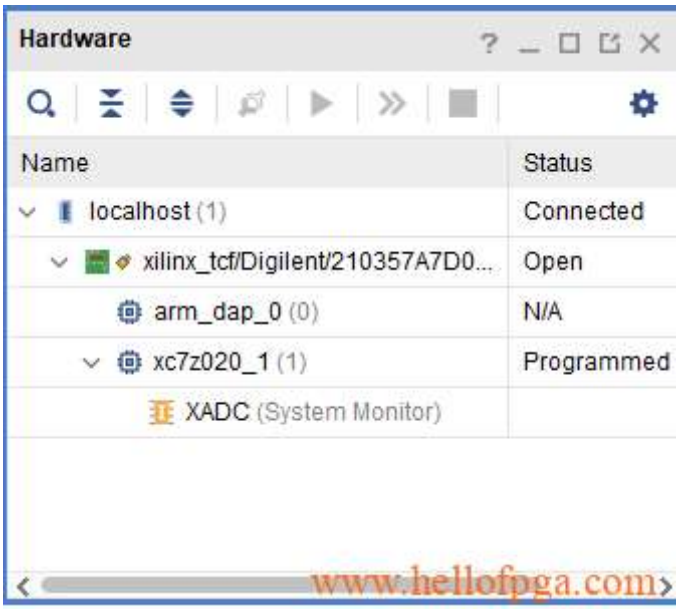
After that, click Run and Generate Bitstream to synthesize the cabling and generate the binary



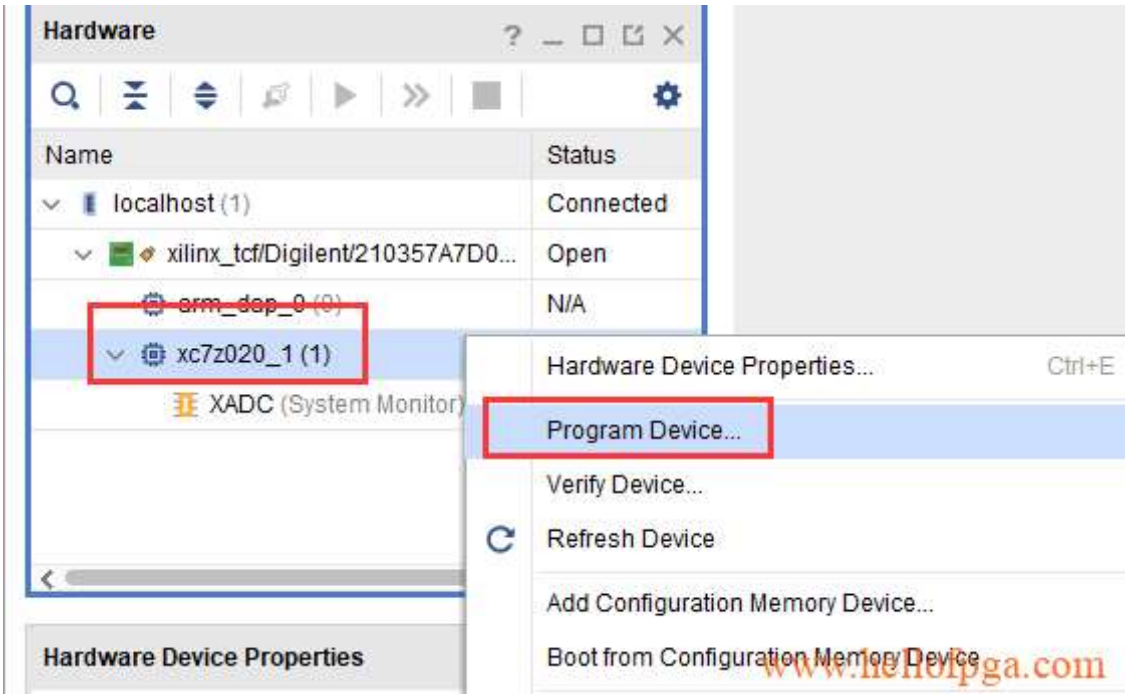
2. Download the code

2.1 Turn on JTAG and the motherboard, and power the motherboard, click open target to connect the target board

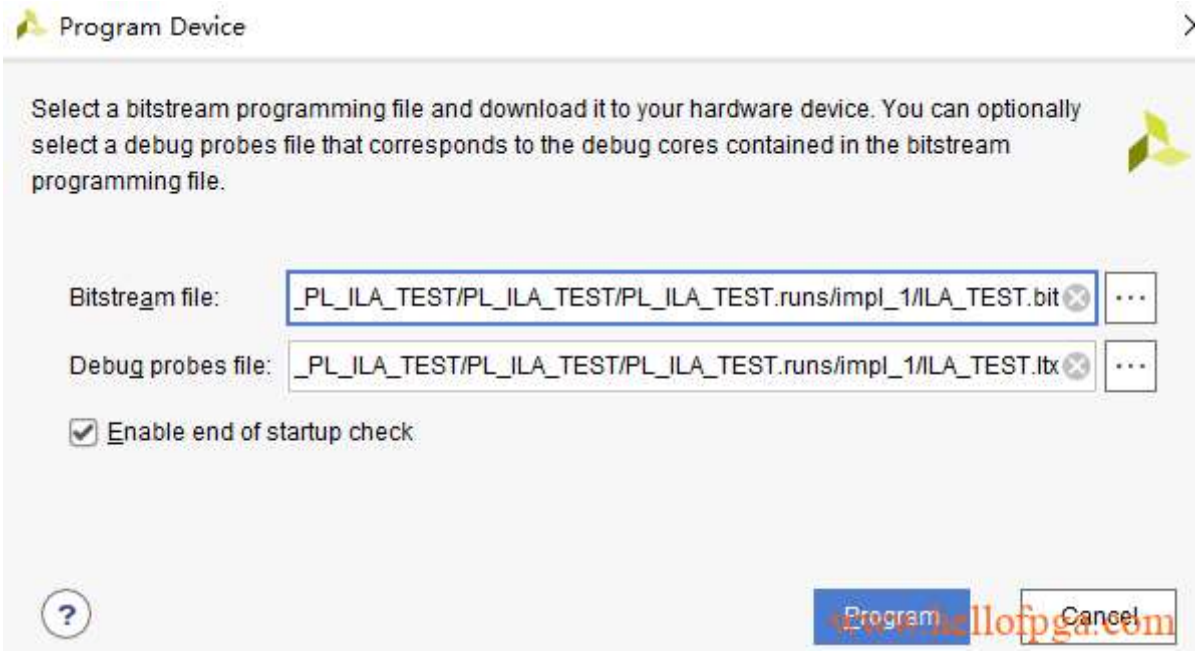
2.2 The following figure will appear after successful connection



2.3 Right-click to select the chip model, and click program device to download the FPGA code (these operations are consistent with normal FPGA engineering)

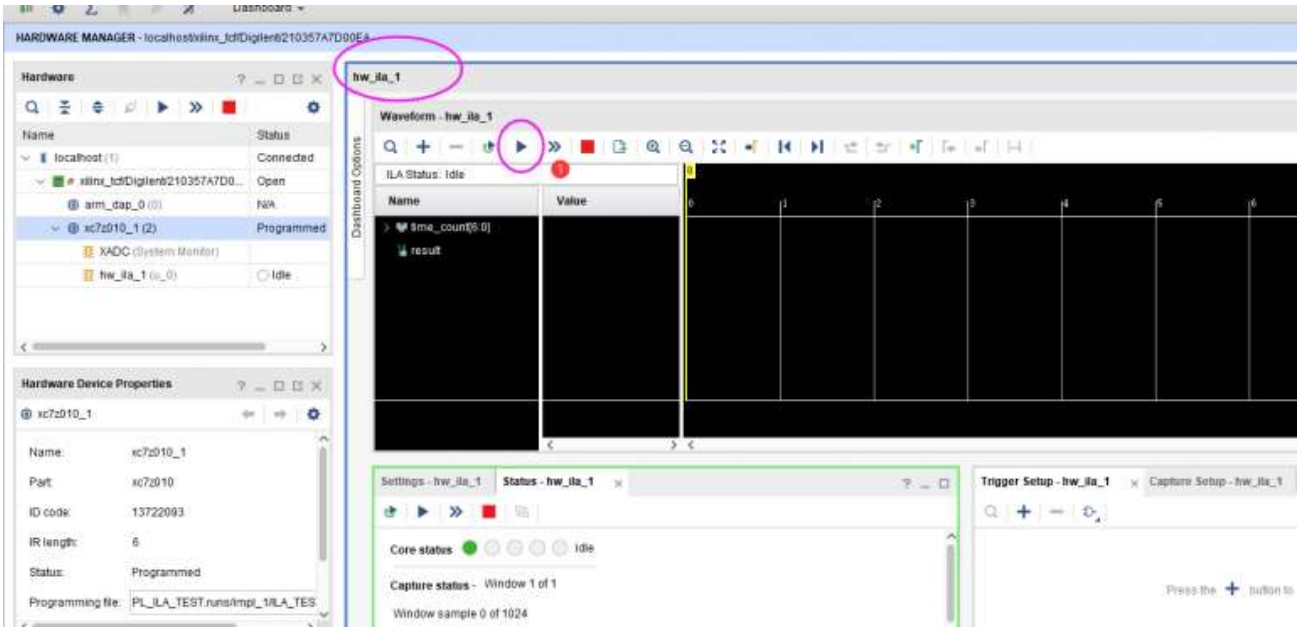


Click Program to download

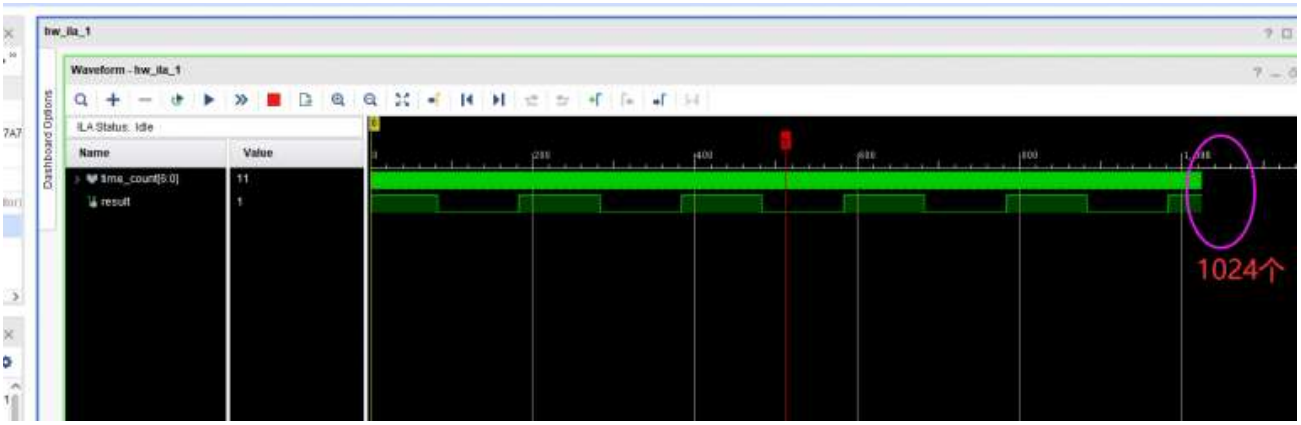


3 ILA debugging

3.1 After performing the previous download operation, the system will automatically pop up an ila debugging window if it is running fine, as shown in the figure below Click the arrow (Run trigger for this ILA core) to start sampling

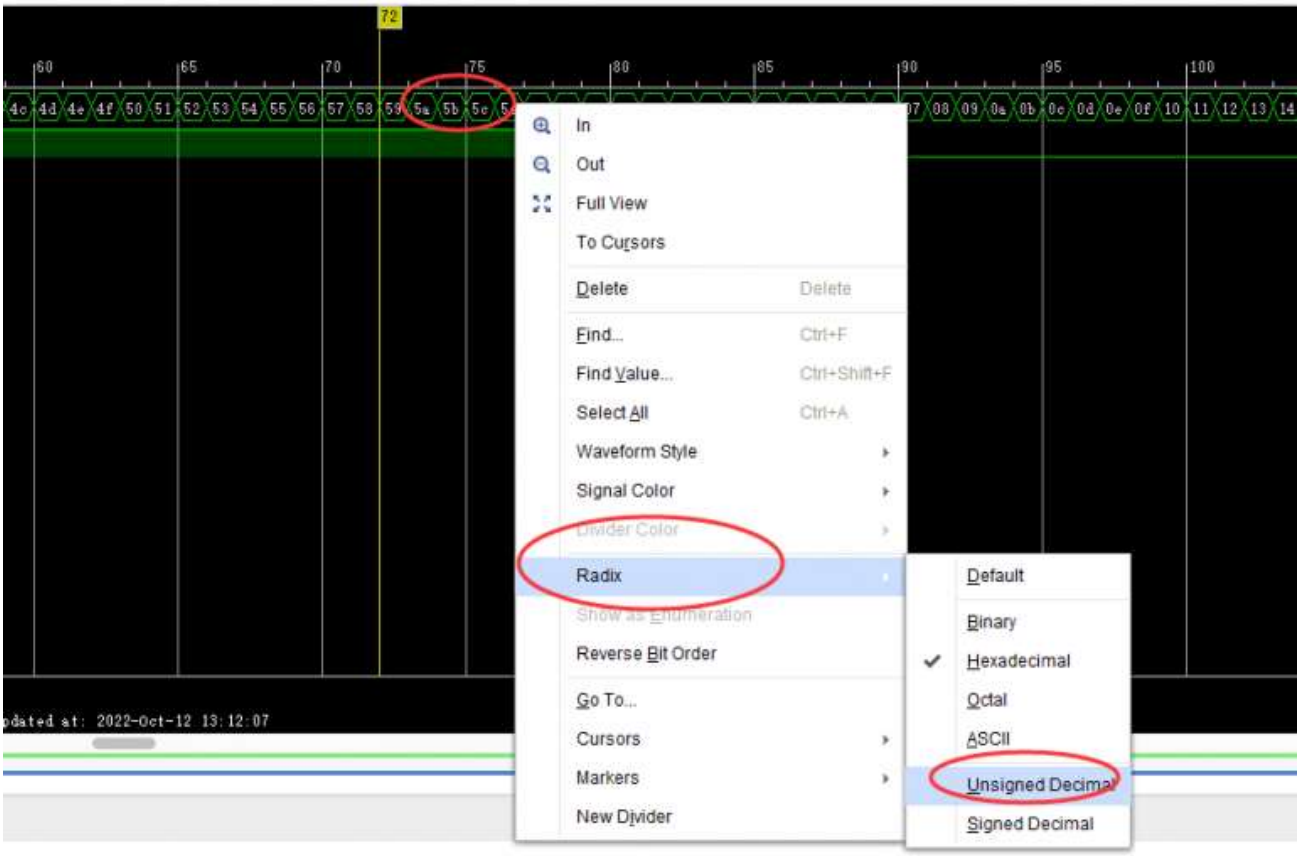


3.2 After that, we will read the probe logic waveform data for 1024 (depth) clocks

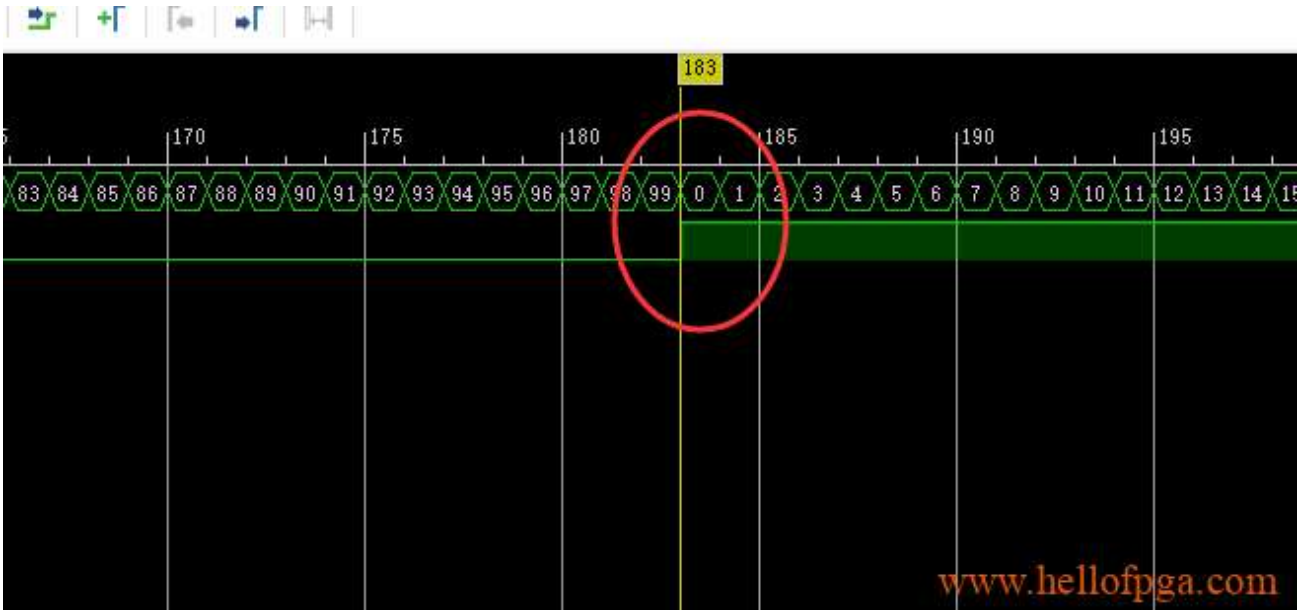


3.3 Through the CTRL + mouse wheel, we can zoom in and out to see the waveform details

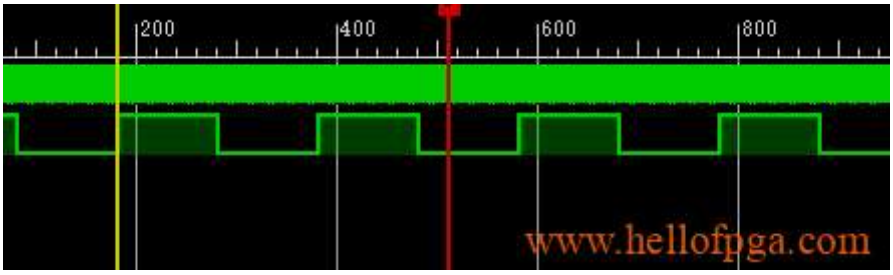
3.4 The system defaults to 16 decimal data display, by right-clicking the waveform to select Radix –> Unsigned Decimal (unsigned, you can also choose signed, this signed), you can view the decimal value



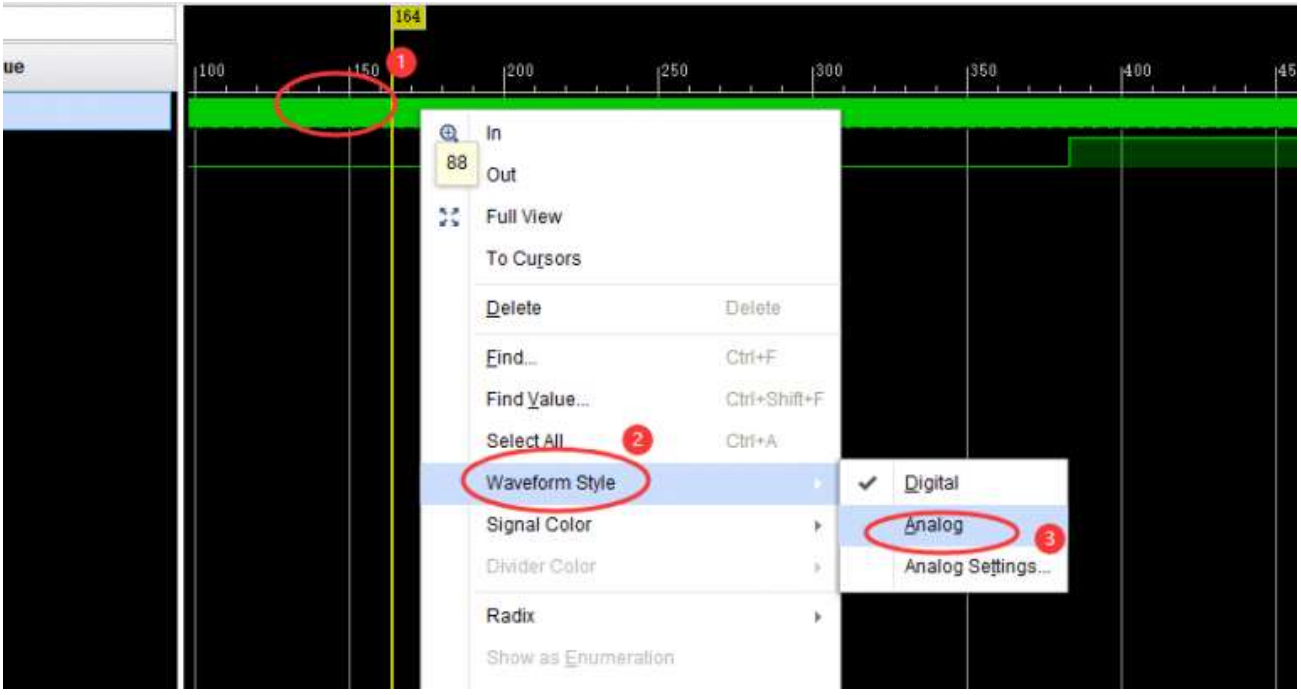
3.5 Through the waveform, we can see that every time the system counts to 99, the result changes the state



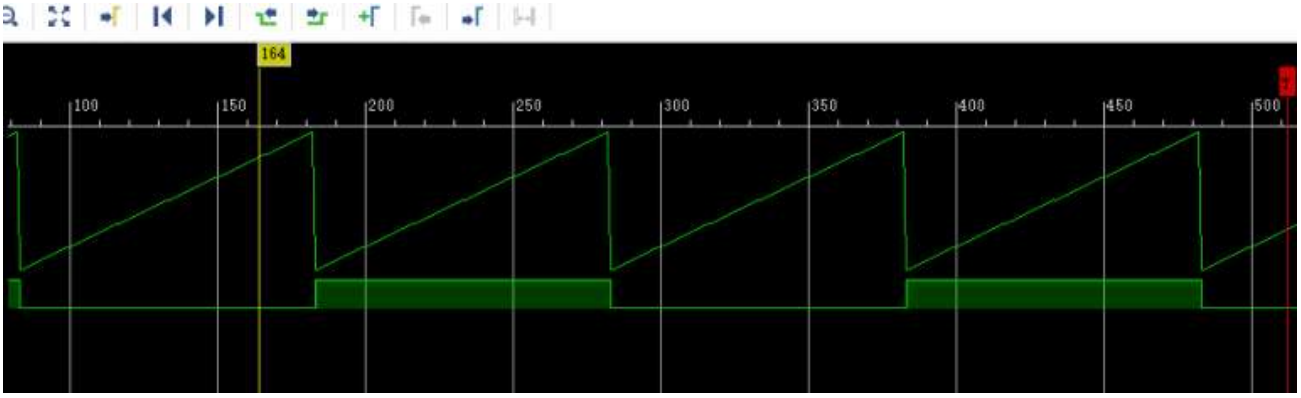
And the result is in a periodic state of change



3.6 Alternatively, click Waveform Style –> Analog to display the analog waveform corresponding to the value (such as sine wave, triangle wave, etc.)



Here, because count is from 0-99 to 0, the shape looks a bit like a triangle wave, as shown in the figure below



Here is the complete project:

11_PL_ILA_TEST_XC7Z020 **Download**

The above is a simple call and introduction of ILA function, of course, the function of ILA is far more than what I introduced, including trigger functions, such as the target value reaches some specific values to trigger waveform sampling, these functions need to be dug by yourself, do not expand

SMART ZYNQ SP & SL