第十六讲、异步串口通信和逻辑分析仪

实验概况

通过 PC 机和 FPGA 进行板级间信号的通信

实验目的

学会使用如何从上位机向 FPGA 发送数据，并如何从 FPGA 接收数据到上位机

软件

ISE 14.7、modelsim、Sublime、友善串口助手

硬件

FPGA 初级开发版一套、电脑

UART 和 RS232

1、UART 的全称叫做通用异步收发传输器。将数据在串行通信和并行通信间的传输转换。通俗的讲就是把多比特的数据转化为单比特的数据，或者把单比特的数据转化为多比特的数据。工作原理是将数据的每个 bit 一位接一位地传输。

2、UART 是一种通用串行数据总线，用于异步通信。该总线双向通信，可以实现全双工传输和接收。

3、RS232 是 UART 的一种，是目前最常用的一种串行通讯接口，用于 pc 机跟外部板级通信。**RS232 通信协议**

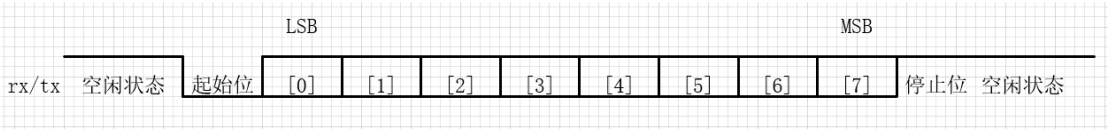
1、rs232 是 uart 的一种，有两根线，分别是 rx 和 tx，这两根线都是 1 比特位宽的。其中 rx 是接收线，tx 是发送线。

2、rx，位宽为 1 比特，pc 机通过串口往 FPGA 发 8 比特数据时，FPGA 通过串口线 rx 一位一位地接收，从最低位到最高位依次接收，最后在 FPGA 里面位拼接成 8 比特数据。

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

3、tx，位宽为 1 比特，FPGA 通过串口往 pc 机发 8 比特数据时，FPGA 把 8 比特数据通过 tx 线一位一位的传给 pc 机，从最低位到最高位依次发送，最后上位机通过串口助手把这一位一位的数据位拼接成 8 比特数据。

4、在不发送或者不接收数据的情况下，rx 和 tx 处于空闲状态，此时 rx 和 tx 线都保持高电平(1)，如果有数据传递，首先会有一个起始位(0)，然后是 8 比特的数据位，接着有 1 比特的停止位(1)，如果停止位以后不再发数据，将进入空闲状态，否则又将数据线拉低(进入起始位状态)。



5、波特率：在串口通信时的速率，单位时间内载波变化的次数，这里选用的是 9600Bd，即发送一比特数据需要的时间为 1/9600 秒。

6、用串口发送或者接收数据(起始位、数据位、停止位)时，每发送或者接收一位数据的时间都需要 1 个波特，即 1/9600 秒。

7、串口发送或者接收一比特数据的时间为一个波特(1/9600)，因此如果用 50M 的系统时钟来计数，就需要记数 cnt=(1/9600s)/20ns≈5208 个系统时钟，才再次发送或者接收下一个数据。

8、上位机通过串口发 8 比特数据时，会自动在发 8 位有效数据前发一个波特时间的起始位，也会自动在发完 8 位有效数据后发一个停止位。同理，串口助手接收 fpga 发送的数据前，必须检测到一波特的起始位才会接收数据，接收完数据后，再接收一个停止位，所以 FPGA 通过串口除了发数据以外，还要发起始位和停止位。

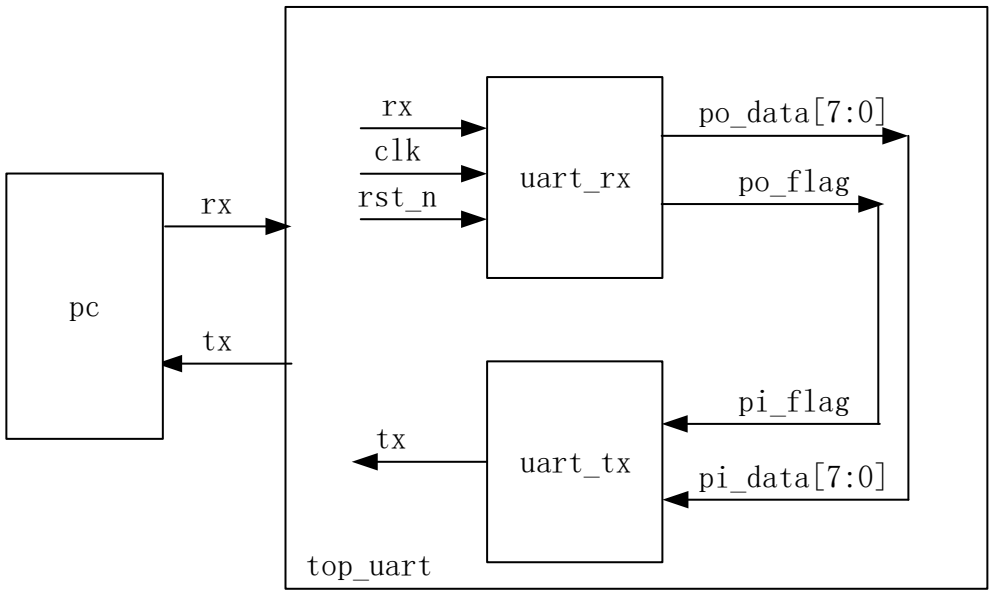
RS232 测试回环

写一个测试回环：上位机由串口助手通过 rx 线往 FPGA 发 8 比特数据，当 FPGA 接收到 8 比特数据后，再通过 tx 线把接收到的 8 比特数据给上位机发回去，要求上位机接收到的数据和上位机发送的数据一样，并且保证连续发送也没问题。rx 和 tx 写在两个单独的模块里面。顶层端口描述如下：

接口传输方向	接口名称	位宽	备注
输入	sclk	1	系统时钟 50M
输入	s_rst_n	1	系统复位
输入	rx	1	FPGA 接收到上位机发送的数据，由低到高接收
输出	tx	1	FPGA 向上位机发送数据，由低位到高位发送

模块结构

1、模块总体架构



2、独立模块端口描述

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

rx

接口传输方向	接口名称	位宽	备注
输入	sclk	1	系统时钟 50M
输入	s_rst_n	1	系统复位
输入	rx	1	FPGA 接收到上位机发送的数据，由低到高接收
输出	po_data	8	FPGA 通过 rx 线接收到的 8 个数据后位拼接成 8 位的数据
输出	po_flag	1	当 po_data 接收完 8 位数据的同时，拉高一个时钟的标志

tx

接口传输方向	接口名称	位宽	备注
输入	sclk	1	系统时钟 50M
输入	s_rst_n	1	系统复位
输入	pi_data	8	rx 模块输入的 8 比特数据，准备通过 tx 线传给上位机
输入	pi_flag	1	rx 模块输入的标志，当标志为高的时候，pi_data 有效
输出	tx	1	pi_data 通过 tx 线由低到高一位一位传给上位机

注意事项：

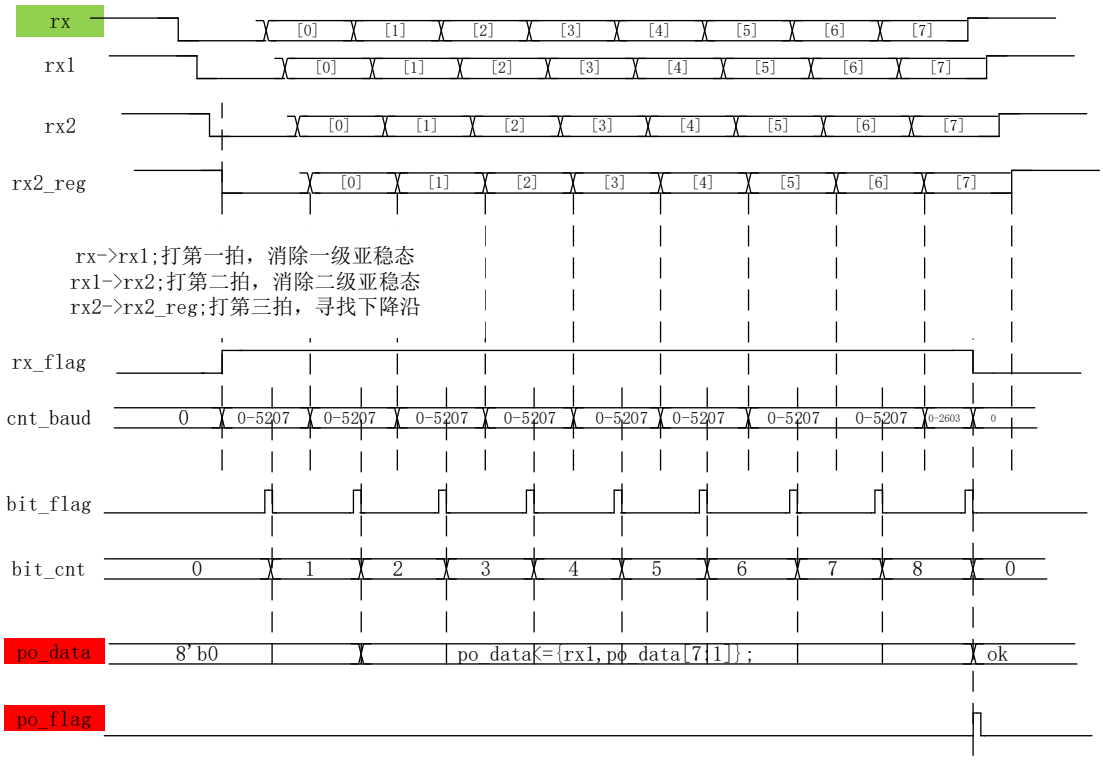
1、rx 传到 FPGA 里面之后会用寄存器打两拍，目的是为了尽量降低亚稳态，如果不做这样的处理，就可能导致采集的数据不稳定，只对做了亚稳态处理后的数据才在后面使用。

2、外部的数据通过 rx 线传到 FPGA，FPGA 采数据时，在中间时刻采数可以保证采的数据最稳定。

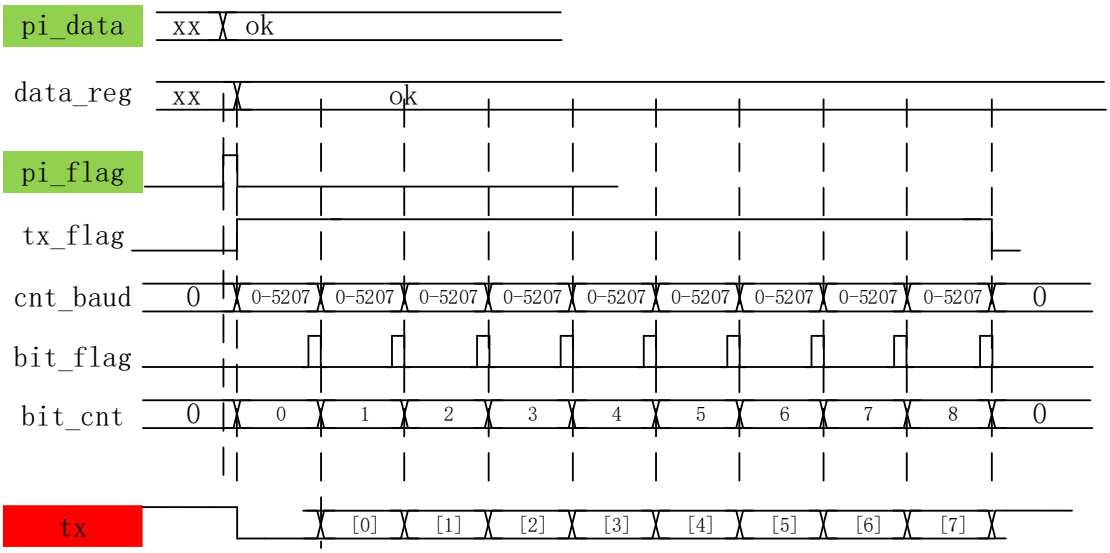
简易波形

uart_rx:

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992



uart_tx:



关键代码

uart_rx:

```
0   module  uart_rx(
```

```
1   input   wire        clk,
2   input   wire        rst_n,
3   input   wire        rx,//上位机传过来的 1bit 数据
4
5   output  reg[7:0]    po_data,//位拼接 rx 的 8bit 数据
6   output  reg     po_flag//标志 po_data 数据有效
7   );
8
9   reg     rx1;
10  reg     rx2;
11  reg     rx2_reg;
12  reg     rx_flag;
13  reg [12:0]  cnt_baud;
14  reg     bit_flag;
15  reg [3:0]   bit_cnt;
16
17  parameter   CNT_BAUD_MAX    =   5207;
18  parameter   CNT_HALF_BAUD_MAX=  2603;
19
20  //rx1,打第一拍，消除一级亚稳态
21  always@(posedge clk or  negedge rst_n)
22      if(rst_n==0)
23          rx1 <=  1;
24      else
25          rx1 <=  rx;
26  //rx2,打第二拍，消除二级亚稳态
27  always@(posedge clk or  negedge rst_n)
28      if(rst_n==0)
29          rx2 <=  1;
30      else
31          rx2 <=  rx1;
32  //rx2_reg,打第三拍，寻找第一个下降沿
33  always@(posedge clk or  negedge rst_n)
34      if(rst_n==0)
35          rx2_reg <=  1;
36      else
37          rx2_reg <=  rx2;
38  //rx_flag,控制 cnt_baud 循环计数
39  always@(posedge clk or  negedge rst_n)
40      if(rst_n==0)
41          rx_flag <=  0;
42      else if(rx2_reg==1&&rx2==0)
43          rx_flag <=  1;
44      else if(bit_cnt==8&&bit_flag==1)
```


码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

```
45          rx_flag <=  0;
46  //cnt_baud,当 rx_flag 为 1 时,0-5207 循环计数
47  always@(posedge clk or  negedge rst_n)
48      if(rst_n==0)
49          cnt_baud    <=  0;
50      else if(cnt_baud==CNT_BAUD_MAX)
51          cnt_baud    <=  0;
52      else if(bit_cnt==8&&bit_flag==1)
53          cnt_baud    <=  0;
54      else if(rx_flag==1)
55          cnt_baud    <=  cnt_baud+1;
56  //bit_flag,代替了 cnt_baud==2603 条件,控制 po_data 取数
57  always@(posedge clk or  negedge rst_n)
58      if(rst_n==0)
59          bit_flag    <=  0;
60      else if(cnt_baud==CNT_HALF_BAUD_MAX)
61          bit_flag    <=  1;
62      else
63          bit_flag    <=  0;
64  //bit_cnt,对 bit_flag 计数,0-8 循环计数
65  always@(posedge clk or  negedge rst_n)
66      if(rst_n==0)
67          bit_cnt <=  0;
68      else if(bit_cnt==8&&bit_flag==1)
69          bit_cnt <=  0;
70      else if(bit_flag==1)
71          bit_cnt <=  bit_cnt+1;
72  //po_data,对 rx2 进行位拼接的变量,在后 8 个 bit_flag 为 1 时取数
73  always@(posedge clk or  negedge rst_n)
74      if(rst_n==0)
75          po_data <=  0;
76      else if(bit_cnt>=1&&bit_flag==1)
77          po_data<={rx2,po_data[7:1]};
78  //po_flag,当拼接完 8bit 数据时,拉高一个 clk 的高电平
79  always@(posedge clk or  negedge rst_n)
80      if(rst_n==0)
81          po_flag <=  0;
82      else if(bit_cnt==8&&bit_flag==1)
83          po_flag <=  1;
84      else
85          po_flag <=  0;
86
87  endmodule
```

uart_tx:

```
module  uart_tx(
input   wire        clk,
input   wire        rst_n,
input   wire    [7:0]   pi_data,//需要通过 tx 发送给上位机的
input   wire        pi_flag,//标志 pi_data 有效

output  reg     tx
);

reg [7:0]   data_reg;
reg     tx_flag;
reg [12:0]  cnt_baud;
reg     bit_flag;
reg [3:0]   bit_cnt;

parameter   CNT_BAUD_MAX    =   5207;

//data_reg,数据缓存,避免上个模块输出变量发生变化而影响
always@(posedge clk or  negedge rst_n)
    if(rst_n==0)
        data_reg    <=  8'b0;
    else if(pi_flag==1)
        data_reg    <=  pi_data;
//tx_flag,控制 cnt_baud 计数
always@(posedge clk or  negedge rst_n)
    if(rst_n==0)
        tx_flag <=  0;
    else if(pi_flag==1)
        tx_flag <=  1;
    else if(bit_cnt==8&&bit_flag==1)
        tx_flag <=  0;
//cnt_baud,当 tx_flag 为 1 时,cnt_baud 计数,0-5207 循环计数
always@(posedge clk or  negedge rst_n)
    if(rst_n==0)
        cnt_baud    <=  0;
    else if(cnt_baud==CNT_BAUD_MAX)
        cnt_baud    <=  0;
    else if(tx_flag==1)
        cnt_baud    <=  cnt_baud+1;
//bit_flag,控制给 tx 赋值标志
always@(posedge clk or  negedge rst_n)
    if(rst_n==0)
        bit_flag    <=  0;
    else if(cnt_baud==CNT_BAUD_MAX-1)
```


码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

```
44          bit_flag    <=  1;
45      else
46          bit_flag    <=  0;
47  //bit_cnt,对 bit_flag 计数，0-8 循环计数
48  always@(posedge clk or  negedge rst_n)
49      if(rst_n==0)
50          bit_cnt <=  0;
51      else if(bit_cnt==8&&bit_flag==1)
52          bit_cnt <=  0;
53      else if(bit_flag==1)
54          bit_cnt <=  bit_cnt+1;
55  //tx
56  always@(posedge clk or  negedge rst_n)
57      if(rst_n==0)
58          tx  <=  1;//空闲状态
59      else if(pi_flag==1)
60          tx  <=  0;//起始位
61      else if(bit_cnt<=7&&bit_flag==1)
62          tx  <=  data_reg[bit_cnt];//8 个数据位
63      else if(bit_cnt==8&&bit_flag==1)
64          tx  <=  1;//停止位
65
66  endmodule
```

top_uart 例化：

```
0   module  top_uart(
1   input   wire        clk,
2   input   wire        rst_n,
3   input   wire        rx,
4
5   output  wire        tx
6   );
7
8   wire    [7:0]   data;
9   wire        flag;
10
11  uart_rx uart_rx_inst(
12  .clk    (clk),
13  .rst_n  (rst_n),
14  .rx (rx),
15
16  .po_data(data),
17  .po_flag(flag)
18  );
```


码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

```
19
20  uart_tx uart_tx_inst(
21  .clk    (clk),
22  .rst_n  (rst_n),
23  .pi_data(data),
24  .pi_flag(flag),
25
26  .tx     (tx)
27  );
28  endmodule
```

测试文件写法

1、关键是每隔一个波特时间给 rx 赋一次值，由于 5027 时间太长，所以仿真的时候把它改小一些，方便仿真。

2、用 task 函数产生 rx 比较方便，需要在 sim 文件夹下面新建一个文本文件，如 data.txt，这个文件里面存放的是从 00 到 ff 的 8 比特数据，一共 256 个，上电之后让测试文件读取 data.txt 里面的数据，再把读出的数据一位一位传给 rx。

3、测试文件关键代码

```
0   `timescale  1ns/1ns
1   module  tb_top();
2   reg clk;
3   reg rst_n;
4   reg rx;
5
6   wire    tx;
7
8   reg [7:0]   a_mem[15:0];//a_mem 是一个存储器，相当于一个 ram
9
10  initial
11      $readmemh("./data.txt",a_mem);
12
13  initial
14      begin
15          clk =   0;
16          rst_n   <=  0;
17          #30
18          rst_n   <=  1;
```

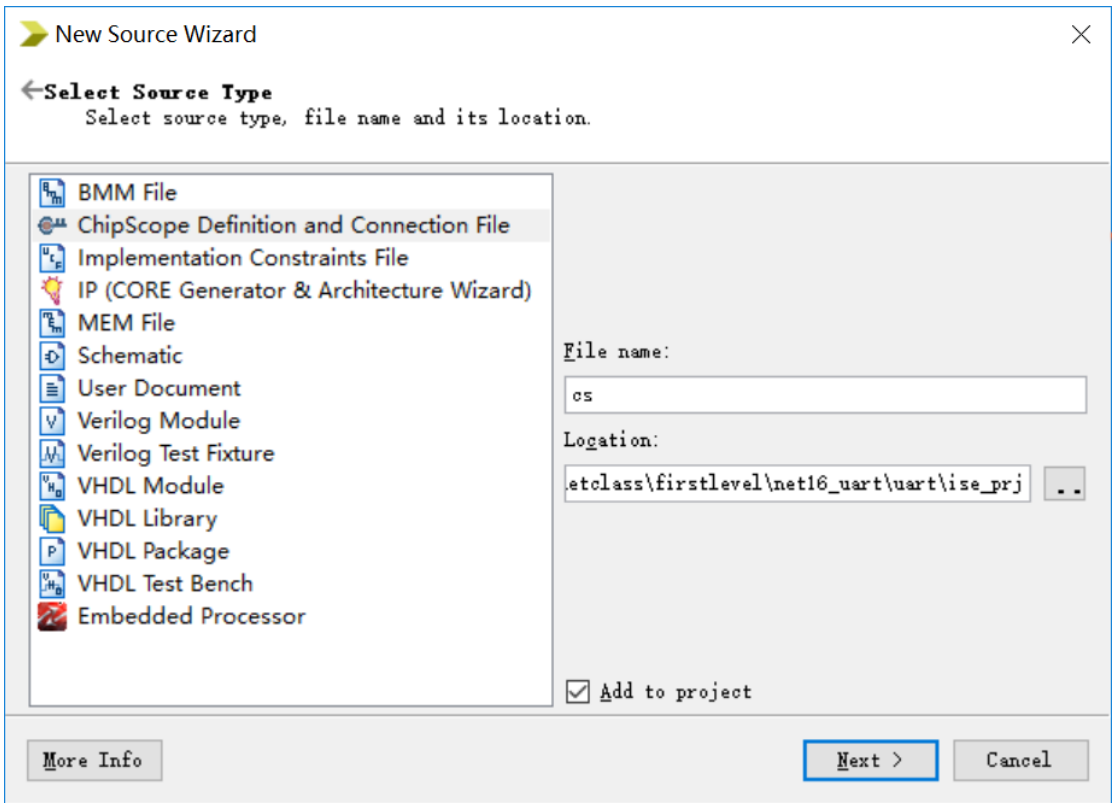

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

```
19      end
20  initial
21      begin
22          rx  <=  1;
23          #200
24          rx_byte();
25      end
26
27  always  #10 clk =   ~clk;
28
29  task    rx_byte();
30      integer j;//定义了一个整形变量
31      for(j=0;j<16;j=j+1)
32          rx_bit(a_mem[j]);//a_mem[j]是 data.txt 文件里面第 j 个 8bit 的数
据
33  endtask
34
35  task    rx_bit(input[7:0]   data);//data 是 a_memd[j]的值
36      integer i;
37      for(i=0;i<10;i=i+1)
38          begin
39          case(i)
40          0:  rx  <=  1'b0;//起始位
41          1:  rx  <=  data[0];
42          2:  rx  <=  data[1];
43          3:  rx  <=  data[2];
44          4:  rx  <=  data[3];
45          5:  rx  <=  data[4];
46          6:  rx  <=  data[5];
47          7:  rx  <=  data[6];
48          8:  rx  <=  data[7];
49          9:  rx  <=  1'b1;//停止位
50          endcase
51          #104160;   //104160*10*16  =18ms
52          end
53  endtask
54  top_uart    top_uart_inst(
55  .clk    (clk    ),
56  .rst_n  (rst_n  ),
57  .rx (rx ),
58
59  .tx     (tx     )
60  );
61  endmodule
```

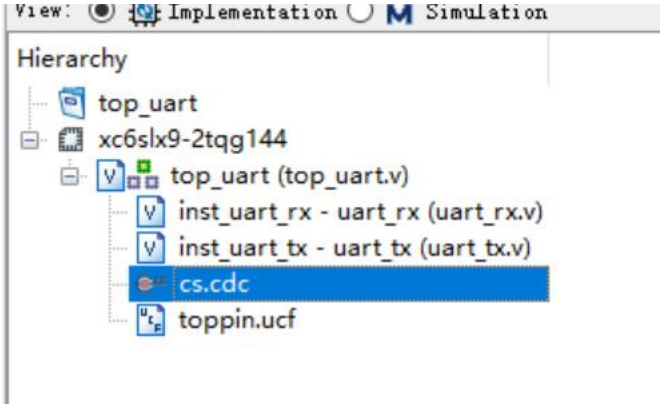

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

逻辑分析仪使用：

点击工程，右键新建 sources，选中 ChipScope Definition and Connection

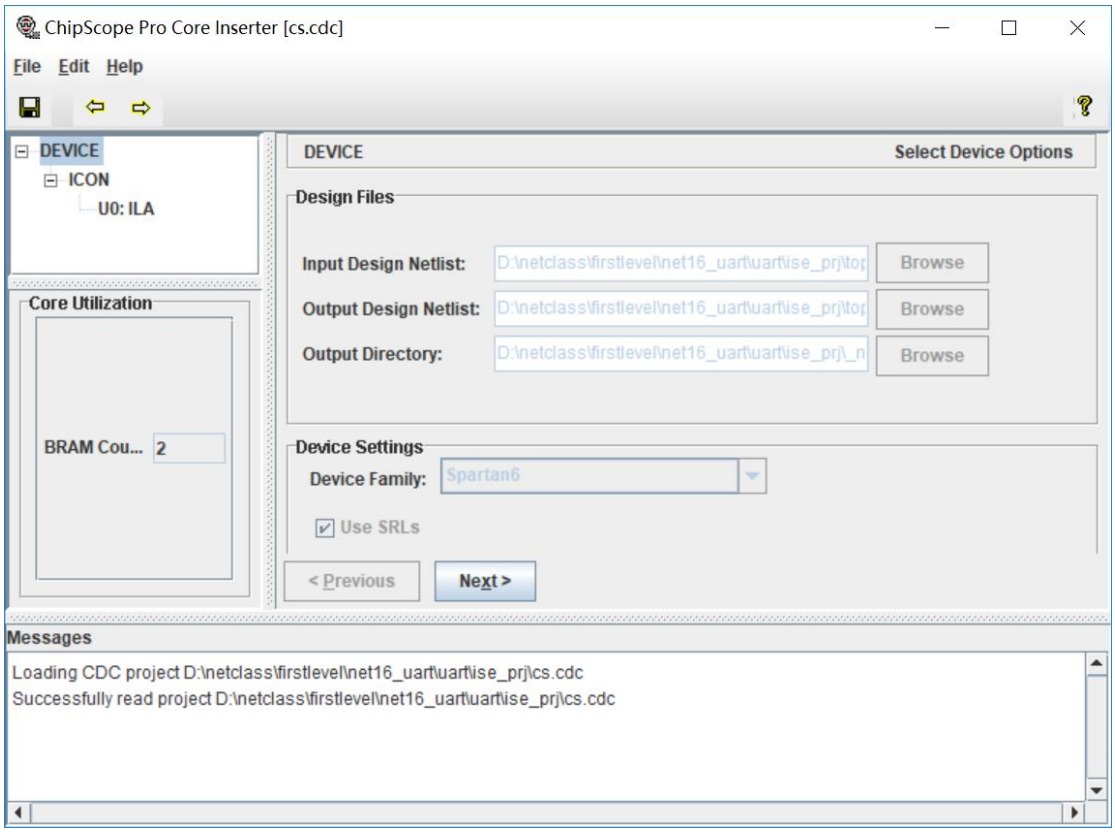


双击 cs.cdc 文件打开逻辑分析仪配置界面

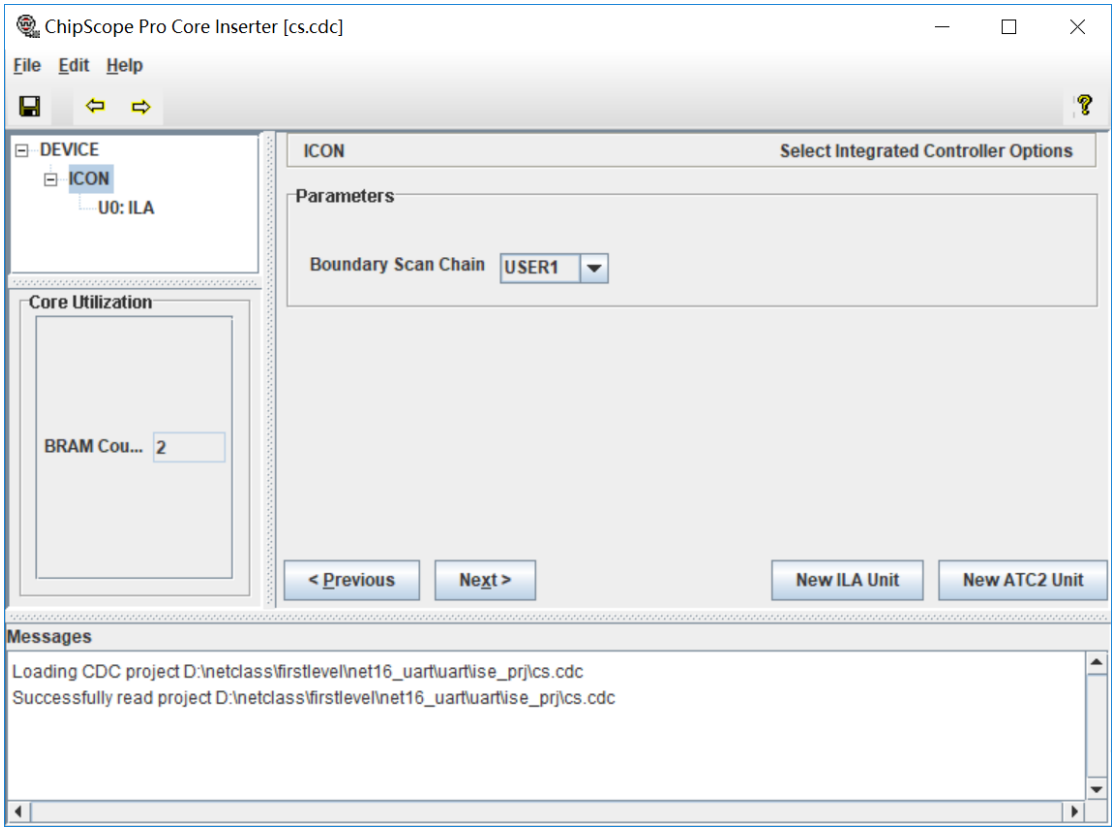


打开后第一个页面无需配置直接点击 NEXT

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992



第二个界面也无需设置直接 NEXT



第三个界面中有不少配置了

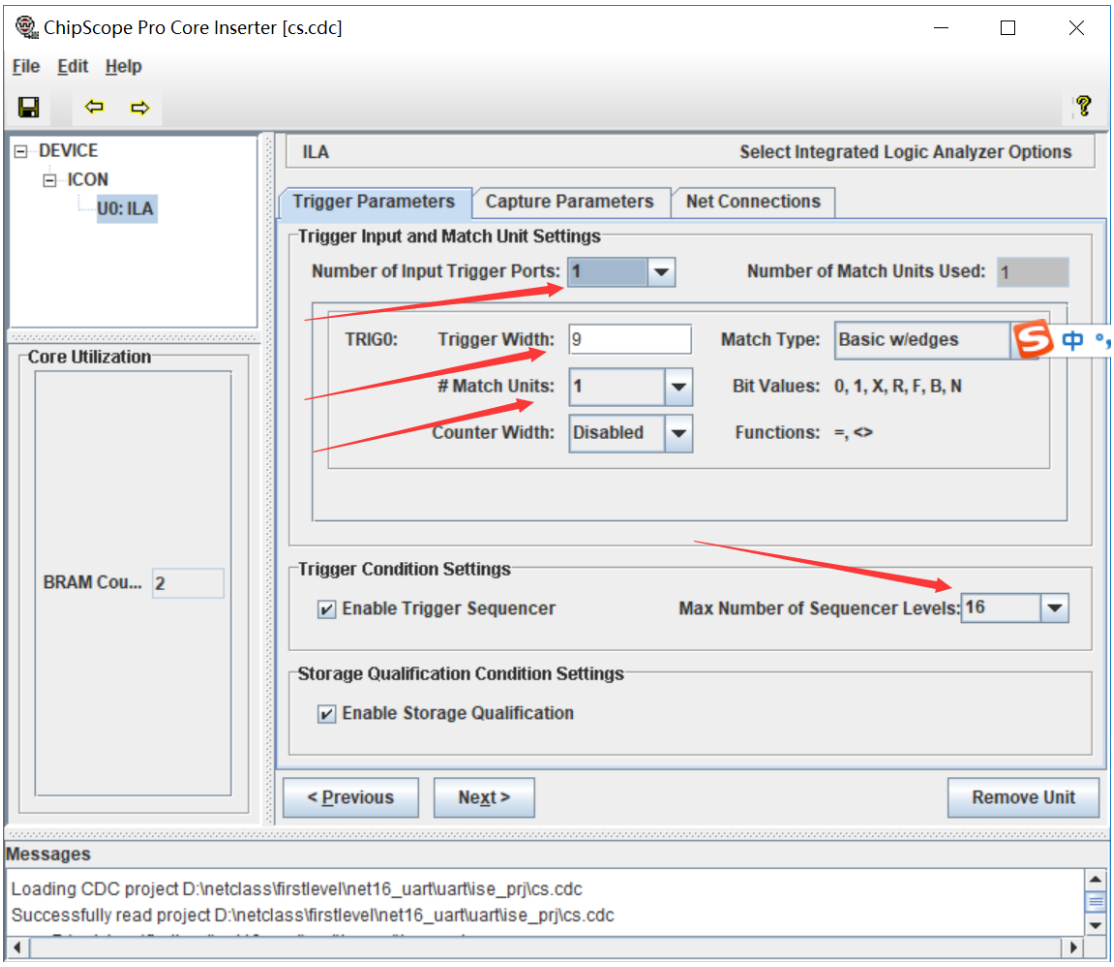
Number of input Trigger port 设置为 1，这里触发接口可以多个，每个可以设置不同的位宽，当前用用我们配置为 1 个，把接口位宽配置更多即可。

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

TRIG0 就是我们设置的触发接口，位宽设置为 9，因为我们后续要观察的信号 po_flag 和 po_data 一起是 9bit 位宽。

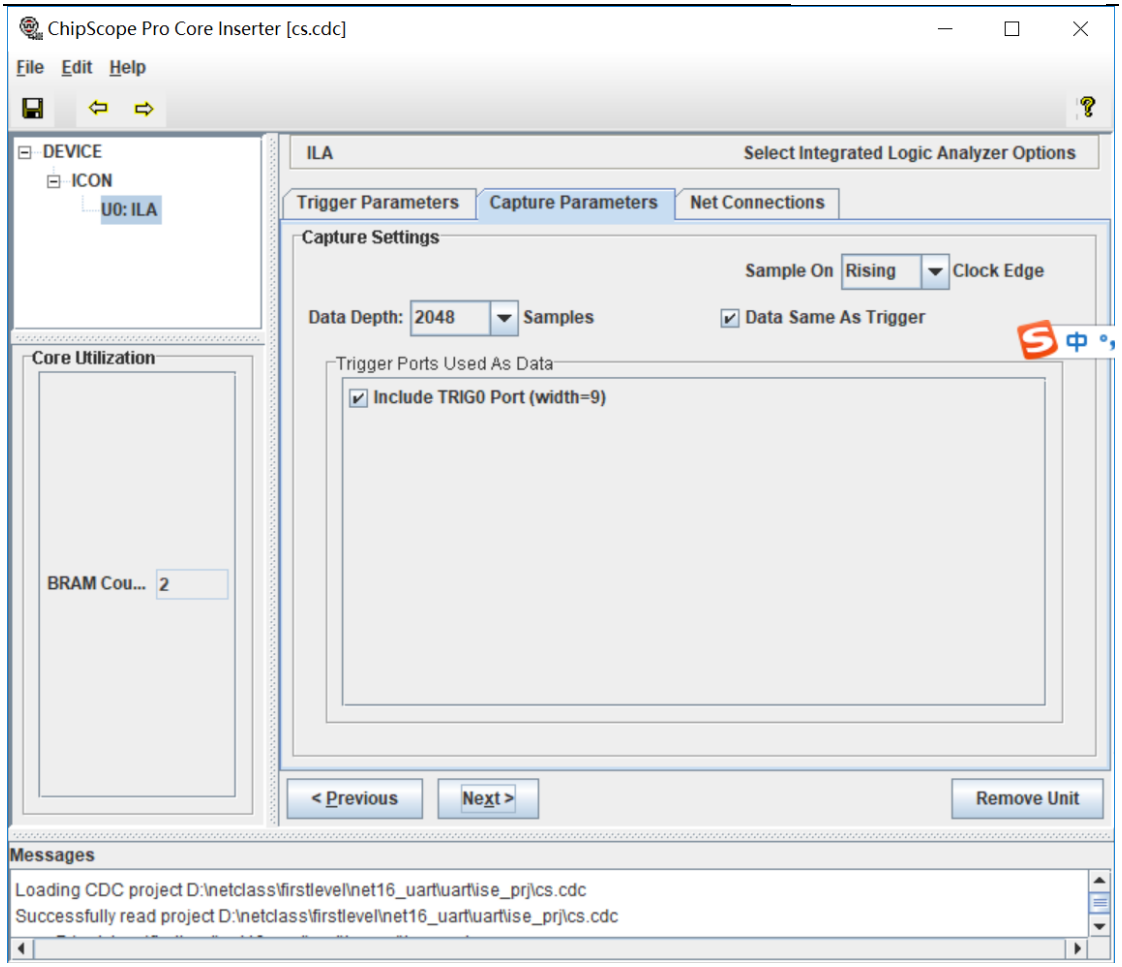
Match units 这里是指比较器数量，一般我们只需要一个就可以了，也就是判断等于，0 底电平、1 高电平、X 任意电平、R 上升沿、F 下降沿、B 双边沿交变

Max number of sequencer levels 我们默认 16 就可以了，这个指的是一个信号可以做一个最大 16 个序列的匹配，这样可以使用此序列作为触发条件，比如一个信号出现 1011 时触发，就可以使用此设置，只要小于等于 16 即可



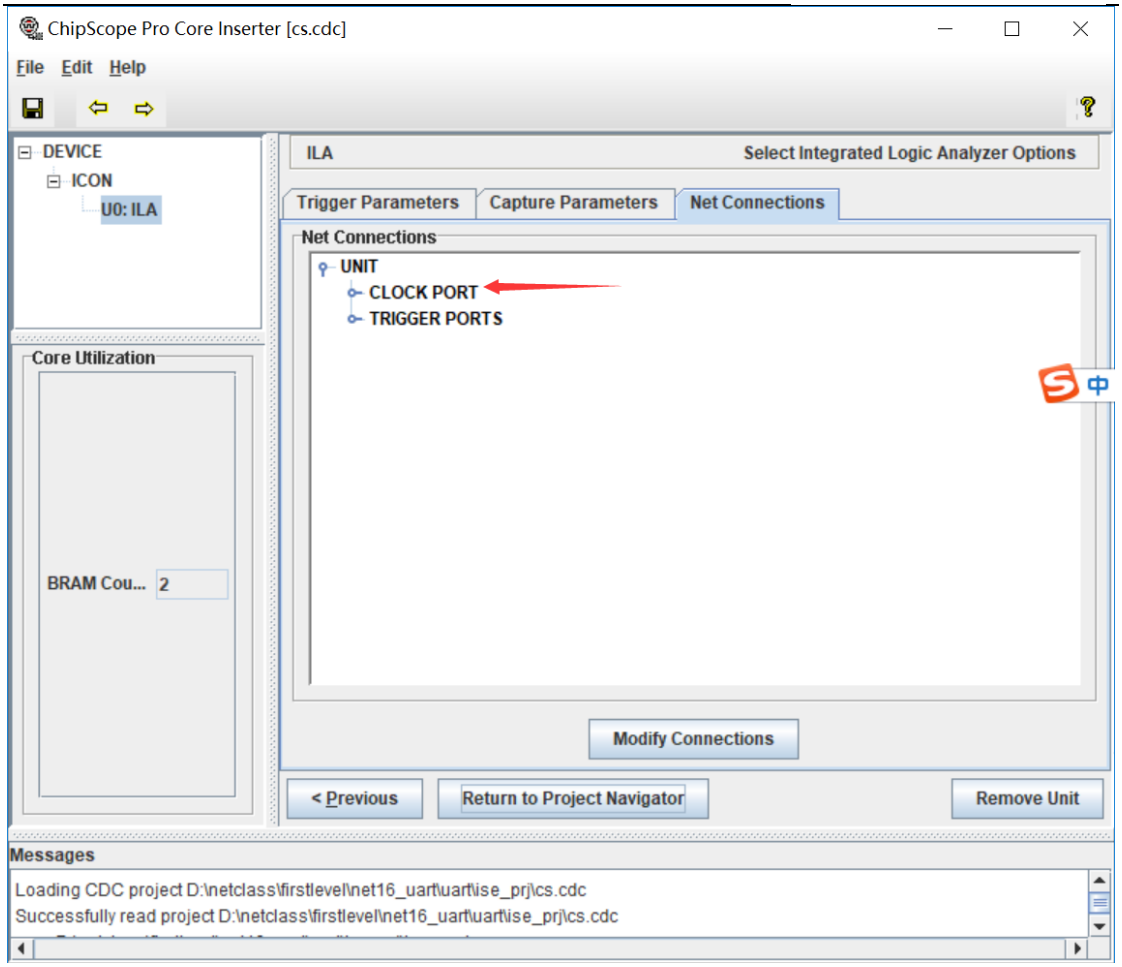
设置采集信号存储深度，其实逻辑分析仪就是在设计内部插入 RAM 把我们观察的信号按照设置条件存储到了 RAM 中，然后从 JTAG 线读会上位机界面显示，这里的采样存储深度指的是 RAM 深度，所以设置要合理，不然综合时会报资源过载。

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992



设置采样时钟和数据信号，双击信号即可

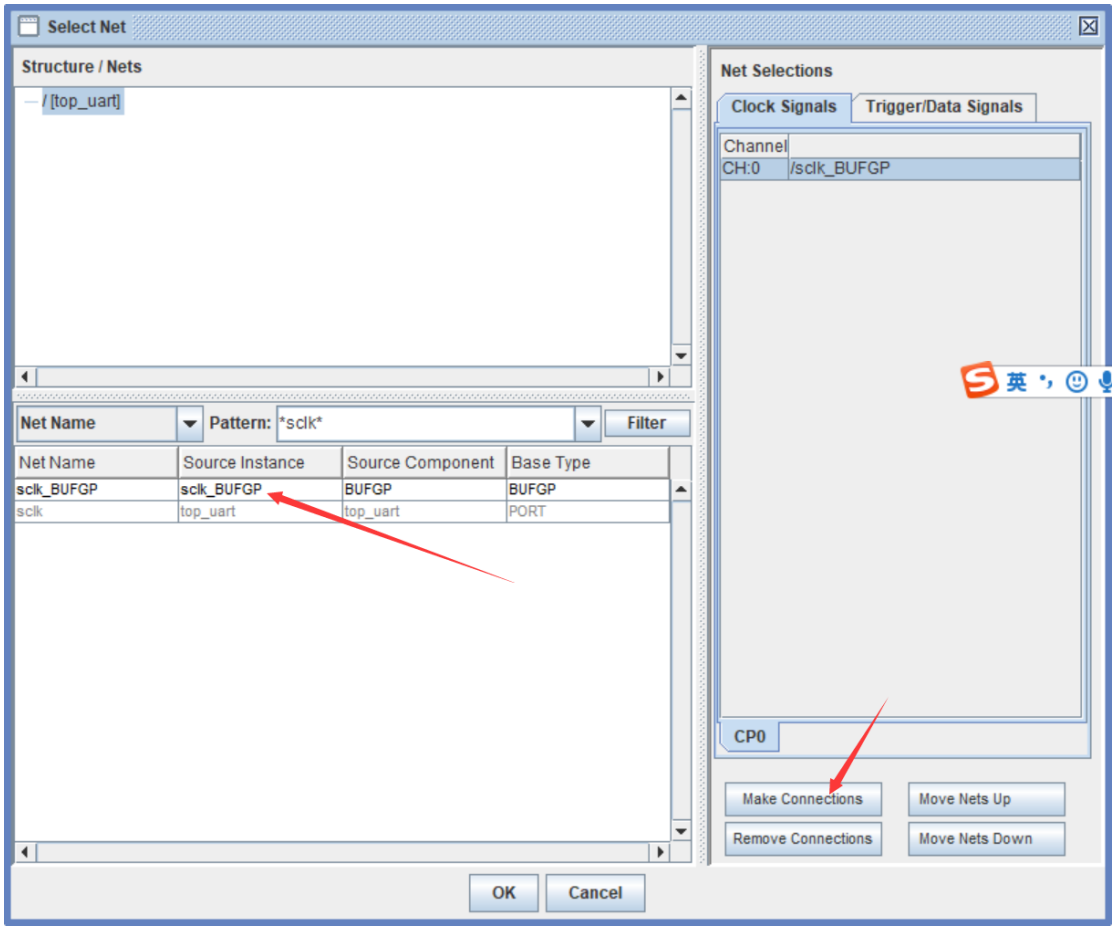
码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992



输入通配符*sclk*来匹配

选中 sclk_bufgp 然后点击 make connections，记住是对 clock 的选项卡设置哦。

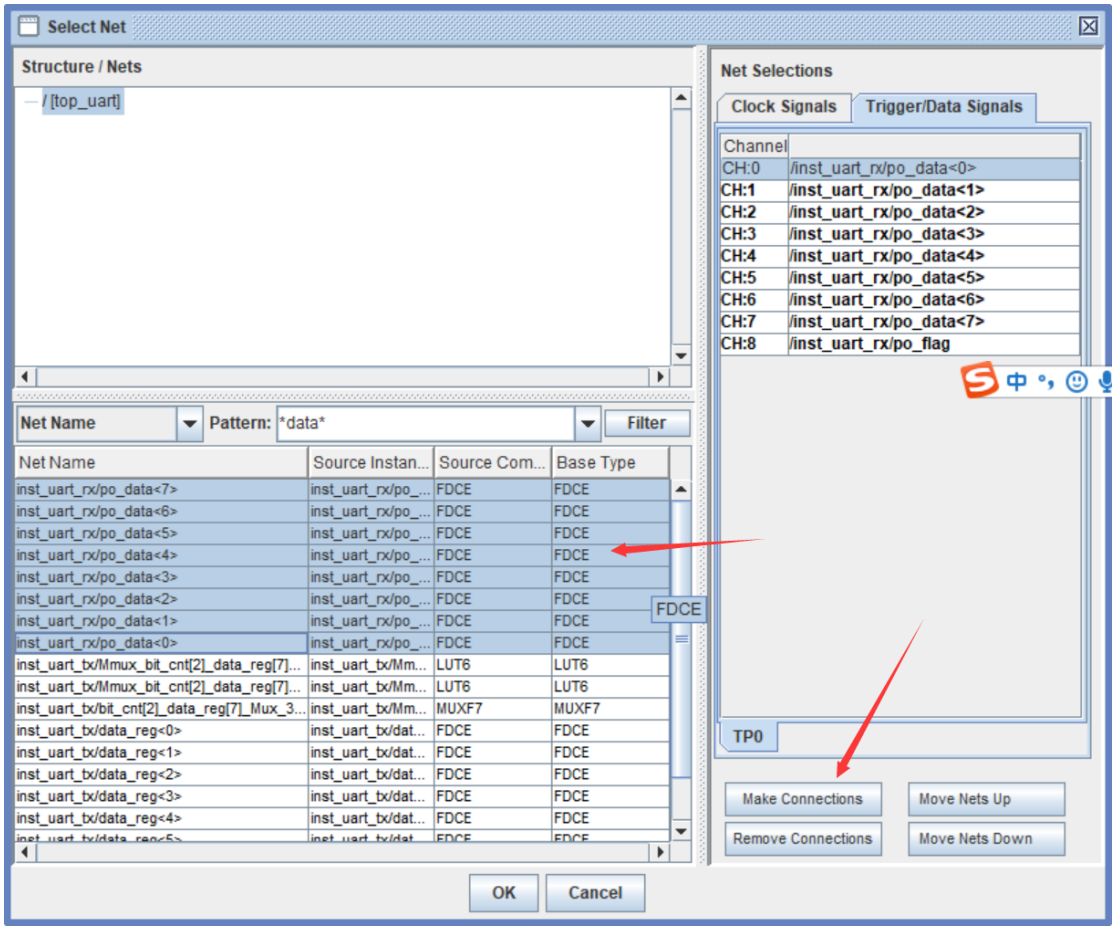
码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992



设置数据通道

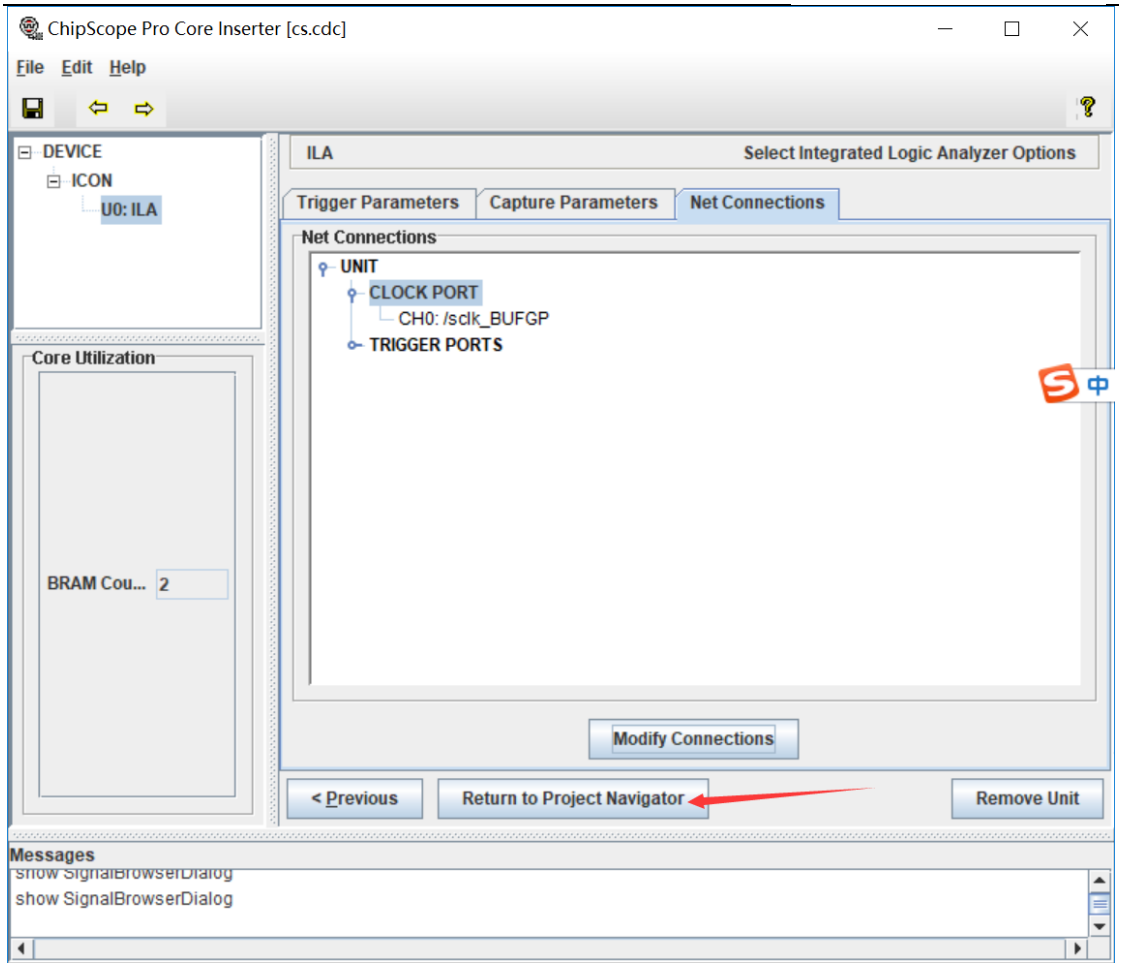
搜索*data*和 *flag*依次进行设置连接如图所示，注意信号顺序哦要和接口信号对应 bit 顺序

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

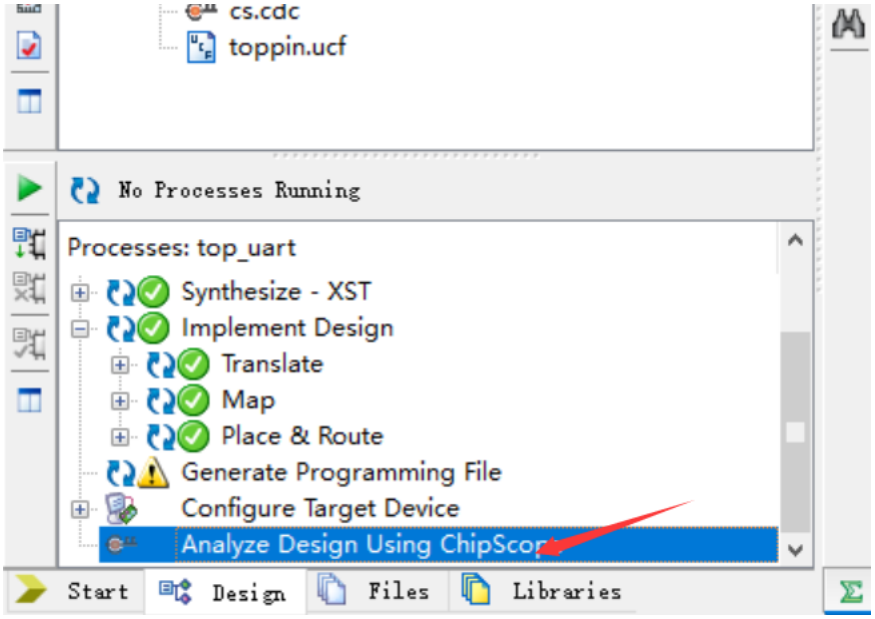


点击 OK 后回到 ISE 界面进行综合

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

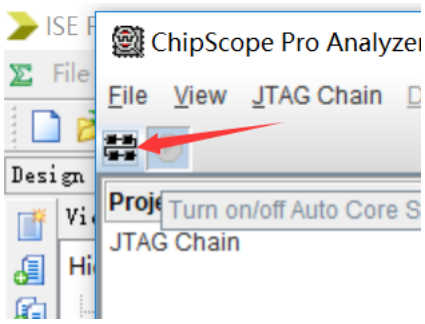


生成 BIT 文件后，按照如下界面开始 Chipscope

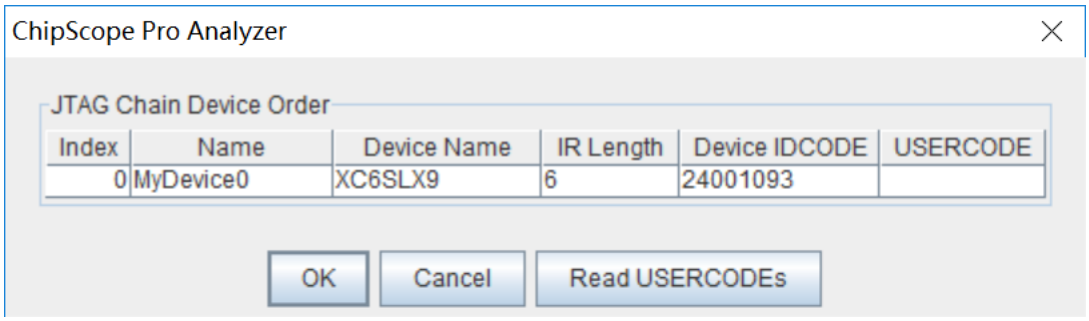


连接板卡

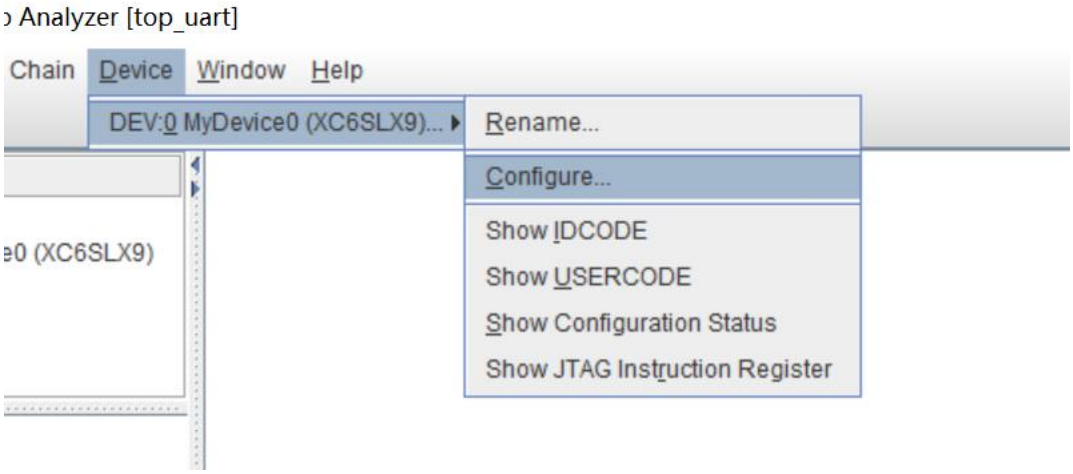
码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992



选择 OK

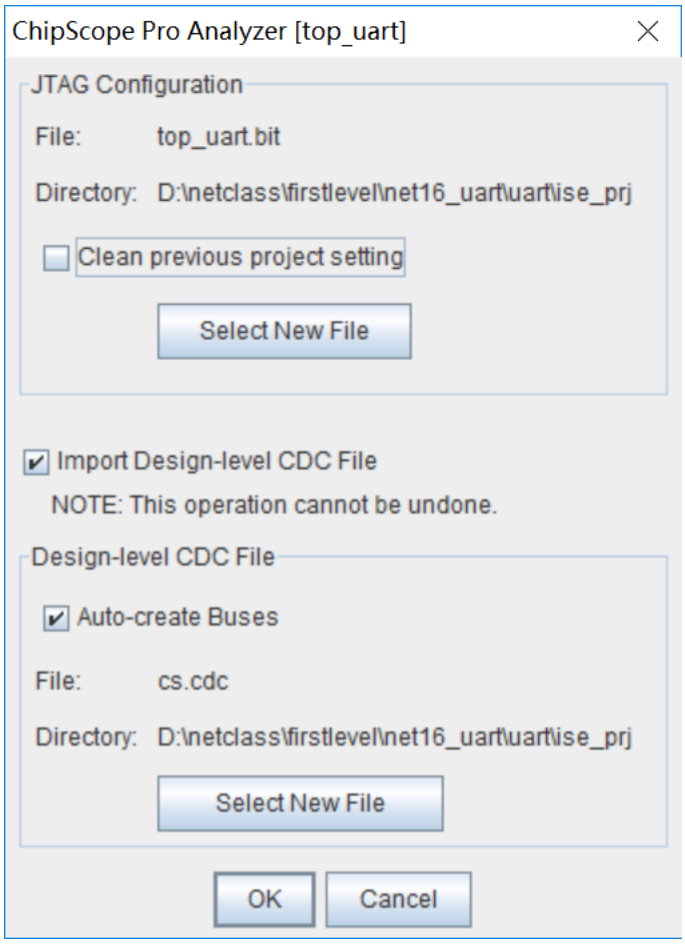


选择 config

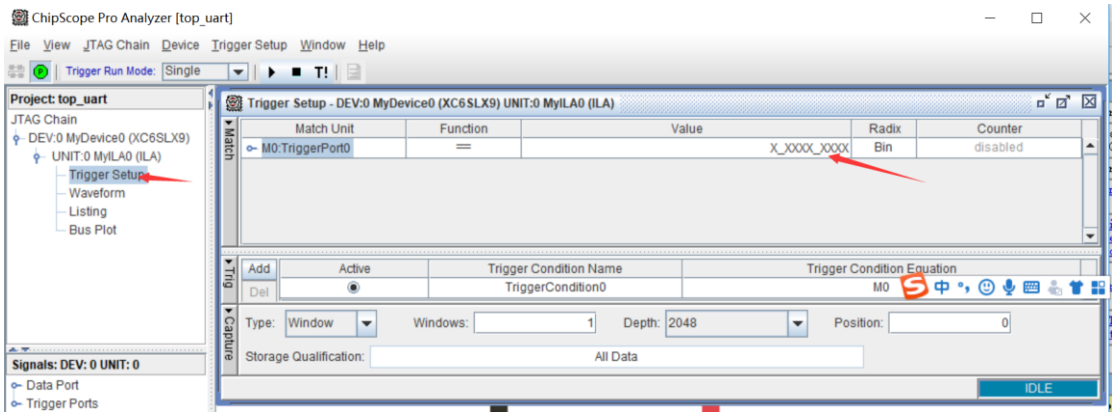


点击 OK

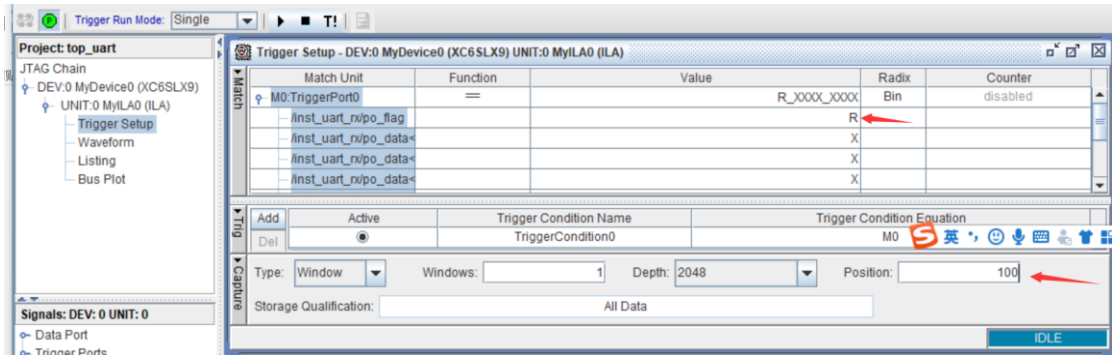
码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992



打开触发设置界面



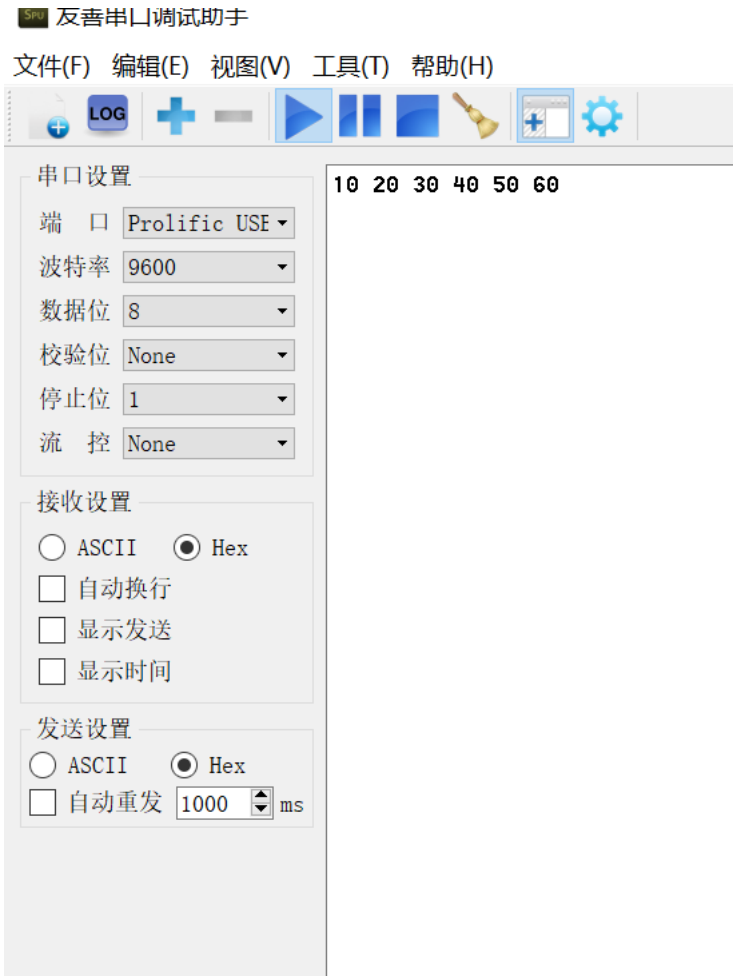
设置触发值，将 po_flag 设置为上升沿触发，position 设置为 100



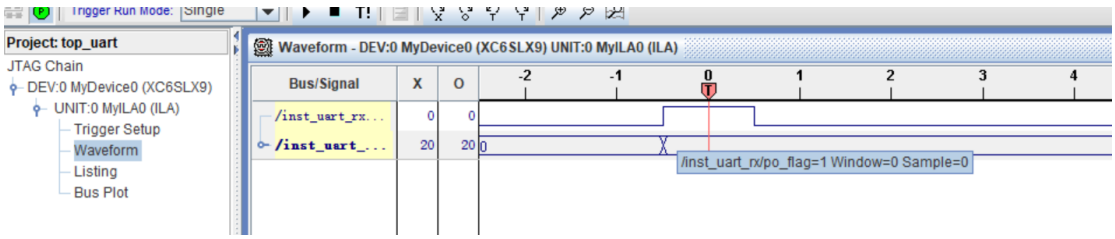
码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992

启动触发

启动串口发送数据



抓取到第一个数据 0x10



可以放大触发信号，可以看见细节

码课网&RTL 库&威三学院
北京市延庆区妫水北街 5 号院辰龙国际 6 号楼 406
010-60168992