电子科技大学

电子科学与工程学院（示范性微电子学院）

实 验 报 告

(2019 -2020 - 1)

课程名称 IC 综合实验三

实验名称 IC 测试实验

指导老师 XXX ，XX

学生姓名 XXX XXX XXX

学生学号 20173401020XX 20173401010XX

20173401010XX

电子科技大学电子科学与工程学院

电 子 科 技 大 学

实 验 报 告

实验地点： **基础实验大楼**

实验时间： **2021/01/11**

报告目录：

一、实验室名称：基础实验室

二、实验项目名称：**IC** 测试实验

三、实验学时：**8**

四、实验目的： 另附页

五、实验原理： 另附页

六、实验器材（设备、元器件）：另附页

七、实验内容： 另附页

八、实验步骤： 另附页

九、实验数据及结果分析：另附页

十、实验结论： 另附页

十一、总结及心得体会：另附页

十二、对本实验过程及方法、手段的改进建议：另附页

指导教师签字：

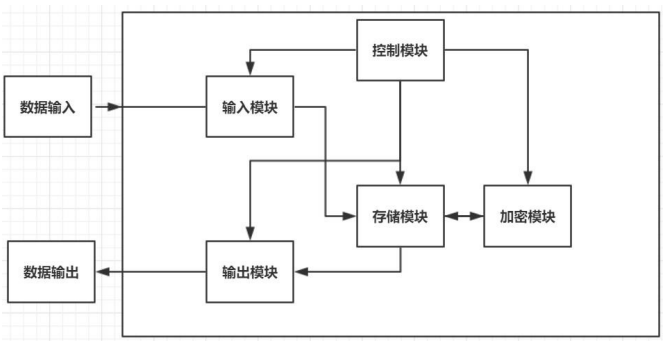
四、实验目的

基于 Zynq 7020 SoPC，搭建 AES 加密芯片测试平台，实现对 AES 加密芯片的自动、大数据量随机测试，并通过 UART 向上位机的串口调试助手发送结果，以验证 AES 加密芯片的功能正确性与可靠性。

五、实验原理

1、128 位 AES 加密芯片原理框架

本小组 AES 加密芯片共有四个模块：接口模块、存储模块、加密模块、控制模块。下图为 AES 加密芯片的原理框图。

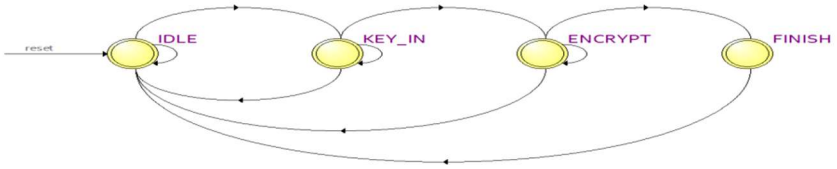


（1）接口模块：采用串行输入的方式，选择改进版 UART 串行通信接口，在原有的基础 UART 接口上加入了奇偶校验位并引出可测试性接口 UART_ERROR，以便代码调试及流片后检测借口模块是否正常工作。

（2）存储模块：采用寄存器方式存储秘钥、待加密明文、以及待输出密文。

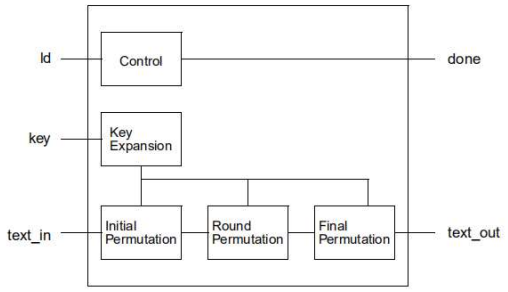
（3）加密模块：使用 AES 核实现加密功能，因面积限制删减掉了该 IP 核的加密模块，并对 S-BOX 模块进行优化以减少芯片面积。

（4）控制模块：采用状态机的状态跳转实现对芯片的控制，无需接入多余的控制信号接口，端口定义比较简单，用户使用起来十分容易。状态机一共设置了四个状态：①IDLE（复位后空闲状态）②KEY_IN（秘钥输入状态）③ENCRYPT（明文输入和加密状态）④FINISH（输出密文状态）。下图为加密芯片的状态跳转图

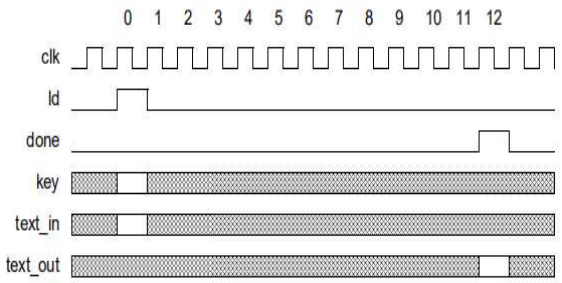


2、128 位 AES 加密芯片加密时序

128 位 AES 加密模块的实现采用了经过流片验证的 IP 核。其基本架构以及时序如下图



IP 核内部架构图



AES 加密时序图

从图中可以看到，AES 加密模块共有 5 个信号:

①使能信号 ld：该信号拉高时表示向加密模块传输秘钥和加密文本。

②秘钥信号 key：128 位的秘钥输入端口。

③加密完成信号 done：该信号拉高时，表示加密已完成，密文 text_out 易经准备好了。需要注意的是，从时序图中我们可以看到，加密完成后，done 只拉高一个周期，代表密文 text_out 只在一个周期内有效。因此我们需要及时将密文取出，以免失效。

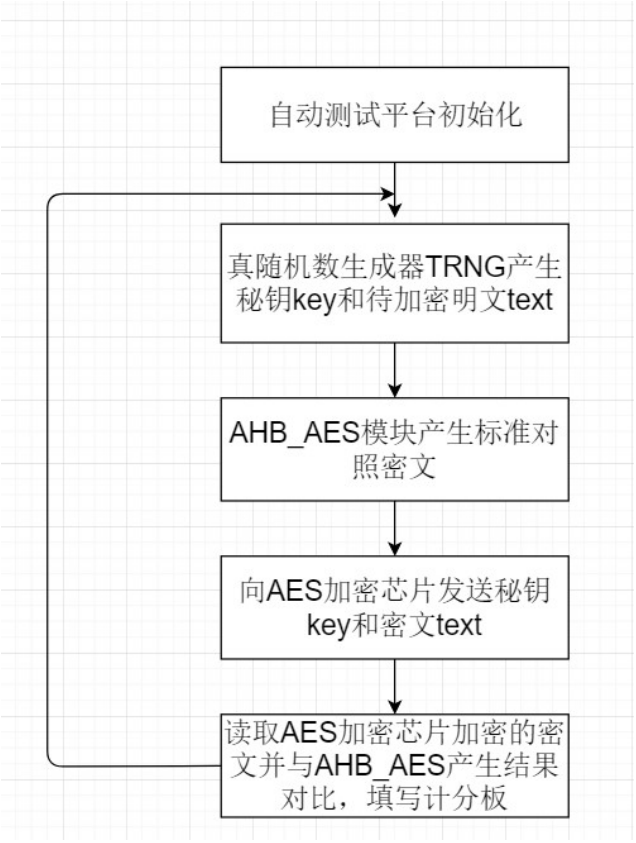
④明文信号 text_in：用于输入待加密的 128 位明文。

⑤密文信号 text_out：用于输出加密完成后的 128 位密文。

3、自动测试平台工作原理

自动测试平台的基本要求是能自发产生大量的、随机的待加密明文和秘钥，将明文和秘钥送入 AES 加密芯片后测试平台要能对加密芯片产生的密文进行自动校验，并把校验结果输出至上位机以验证芯片功能的正确性和稳定性。下图为自动测试平台的工作原理：在开始测试时，自动测试平台进行初始化，然后真随机数生成器 TRNG 模块随机生成秘钥 key 和 128 位待加密明文 text。AHB_AES 模

块是经过验证的 AES 加密软核，用于产生标准的密文和加密芯片产生的密文进行对照，验证其正确性。在完成结果对比后填写计分板并将结果通过 UART 发送至上位机进行显示。下图为自动测试平台工作原理：



六、实验器材（设备、元器件）

PC 机一台

FPGA 开发板一套

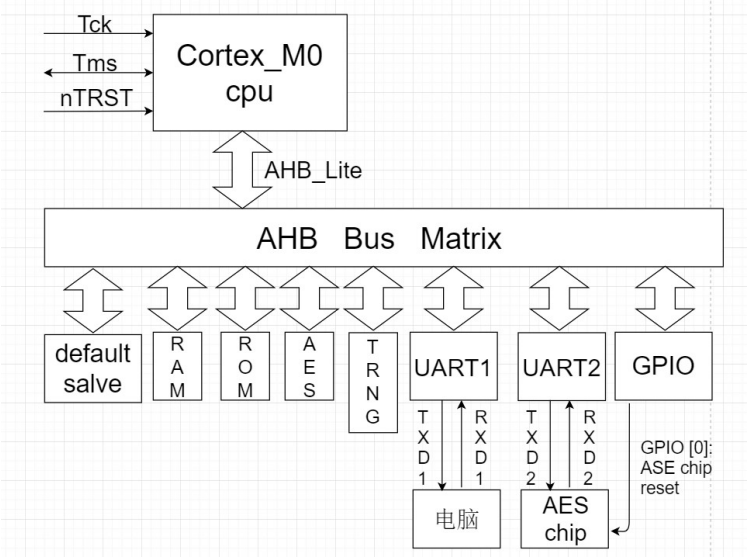
CMSIS DAP 调试器一套

JTAG 下载线一根

USB 供电线一根

七、实验内容

自动测试平台的原理框图如下图，如图所示测试平台部分包括 CortexM0 CPU，ROM，RAM，基于 AHB 总线的 AES128 位加密模块，真随机数生成器 TRNG，通用输入输出接口（GPIO）模块以及用于和电脑、AES 芯片通信的 2 个 UART 模块，配合软件使用可以对 AES 芯片进行自动测试。



八、实验步骤

1、硬件模块设计

硬件部分包括 CortexM0 CPU，ROM，RAM，基于 AHB 总线的 AES128 加密模块，真随机数生成器，GPIO 模块以及用于和电脑、AES 芯片通信的 2 个 UART 模块。配合软件使用可以对 AES 芯片进行自动测试。

测试平台的硬件外设地址分配如下：

```
parameter DW=32,
parameter PORT0_ENABLE=1, //Default Slave
parameter PORT1_ENABLE=1, //ROM
parameter PORT2_ENABLE=1, //RAM
parameter PORT3_ENABLE=1, //UART1
parameter PORT4_ENABLE=1, //UART2
parameter PORT5_ENABLE=0, //MC
parameter PORT6_ENABLE=1, //GPIO
parameter PORT7_ENABLE=1, //TRNG
parameter PORT8_ENABLE=1, //AES
parameter PORT9_ENABLE=0)(
```

```
assign HSEL1 = (HADDR[31:17] == 15'b00000_00000_00000)? PORT1_ENABLE : 1'b0;
assign HSEL2 = (HADDR[31:16] == 16'h2000)? PORT2_ENABLE : 1'b0;
assign HSEL3 = (HADDR[31:4] == 28'h4000001)? PORT3_ENABLE : 1'b0;
assign HSEL4 = (HADDR[31:4] == 28'h4000002)? PORT4_ENABLE : 1'b0;
assign HSEL5 = ((HADDR[31:12] == 20'h40002) | (HADDR[31:12] == 20'h40003))? PORT5_ENABLE : 1'b0;
assign HSEL6 = (HADDR[31:4] == 28'h4000000)? PORT6_ENABLE : 1'b0;
assign HSEL7 = (HADDR[31:4] == 28'h4000003)? PORT7_ENABLE : 1'b0;
assign HSEL8 = (HADDR[31:8] == 24'h400001)? PORT8_ENABLE : 1'b0;
assign HSEL9 = (HADDR[31:0] == 32'h0000_0000)? PORT9_ENABLE : 1'b0;
```

测试平台的顶层模块端口包括系统时钟 OSC_CLK 输入，系统复位 RSTn 输入，加

密芯片时钟 AES_CLK 输出，调试端口，GPIO 端口（其中 GPIO[0]用于控制 AES 加密芯片的复位），两个 UART 端口（一个连接到计算机，另一个连接到 AES 加密芯片）：

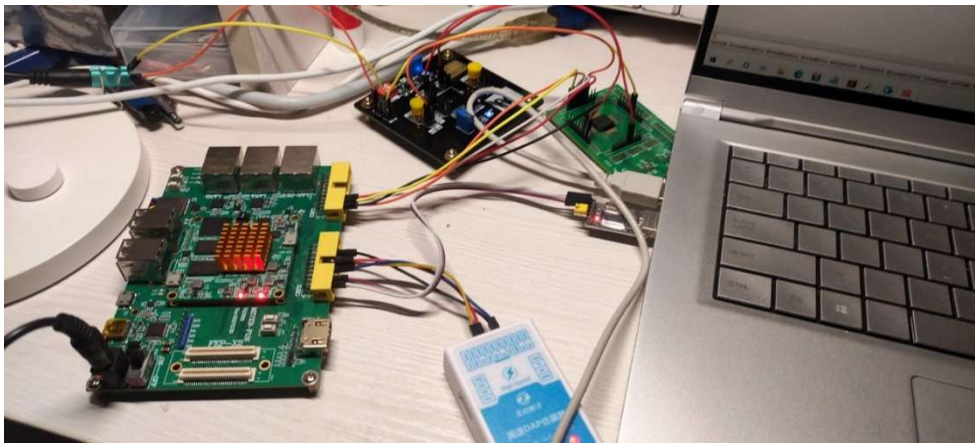
```
module CortexM0_AT(
    input OSC_CLK, //100M oscillator
    input RSTn,    //reset
    output AES_CLK,//AES chip clock

    input nTRST,   //debugger nTRST
    input TCK,     //debugger TCK
    inout TMS,     //debugger TMS

    inout [7:0] GPIO,  //General Purpose Input and Output

    output TXD1, //UART1(connected to computer)
    input RXD1,  //UART1
    output TXD2, //UART2(connected to AES chip)
    input RXD2   //UART2
    );
```

测试平台的 AHB_AES 模块定义了 14 个寄存器，包括 4 个 32 位的 key 寄存器，4 个 32 位的 text_in 寄存器，4 个 32 位的 text_out 寄存器，1 个使能(ld)寄存器以及一个状态(done)寄存器。可以快速的实现 AES 加解密。AHB_AES 是片上模块，抗干扰能力极强，可以认为该模块的输出结果万全正确，若 AES 加密芯片的输出与 AHB_AES 模块的结果不一致，则认为是加密芯片错误。下图为 AHB_AES 模块的寄存器映射（4‘dx 代表地址偏移量）：

```
assign HRDATA = (read_en_reg)?
                (addr_reg == 4'd0)? key[31:0] :
                (addr_reg == 4'd1)? key[63:32] :
                (addr_reg == 4'd2)? key[95:64] :
                     (addr_reg == 4'd3)? key[127:96] :
                     (addr_reg == 4'd4)? text_in[31:0] :
                     (addr_reg == 4'd5)? text_in[63:32] :
                     (addr_reg == 4'd6)? text_in[95:64] :
                     (addr_reg == 4'd7)? text_in[127:96] :
                     (addr_reg == 4'd8)? cipher_text[31:0] :
                     (addr_reg == 4'd9)? cipher_text[63:32] :
                     (addr_reg == 4'd10)? cipher_text[95:64] :
                     (addr_reg == 4'd11)? cipher_text[127:96] :
                     (addr_reg == 4'd12)? {31'b0,ld_reg} :
                     (addr_reg == 4'd13)? {31'b0,done_reg} :
                32'b0
                : 32'b0;
```

测试平台的板级连接如下图所示，黑色的电路板上板载了 LDO，用于产生芯片的电源。连接到 FPGA 自动测试平台的杜邦线包括 AES_CLK，AES 芯片的 reset 端口以及

AES 芯片的 TX 和 RX 端口。我们组在设计芯片时使用了异步通信，因而只需要 2 根数据线就能够实现通信：



2、软件部分设计

软件部分主要使用 C 语言和汇编语言写成。

首先编写 CortexM0 的启动程序，该程序由汇编语言写成。测试平台使用了中断的方式接收 AES 芯片回传的数据，非常适用于 UART 这种低速通信。中断入口向量的设置如下图所示，由于平台包含了 2 个 UART 外设，因此分配了 2 个中断入口向量，本次实验不需要接收电脑发来的数据，因而只用到了 UART2_Handler：

```
; Interrupts
DCD      UART1_Handler              ;   0 Interrupt 0
DCD      UART2_Handler              ;   1 Interrupt 1
DCD      Interrupt2_Handler          ;   2 Interrupt 2
DCD      Interrupt3_Handler          ;   3 Interrupt 3
DCD      Interrupt4_Handler          ;   4 Interrupt 4
DCD      Interrupt5_Handler          ;   5 Interrupt 5
DCD      Interrupt6_Handler          ;   6 Interrupt 6
DCD      Interrupt7_Handler          ;   7 Interrupt 7
DCD      Interrupt8_Handler          ;   8 Interrupt 8
DCD      Interrupt9_Handler          ;   9 Interrupt 9
```

中断对应的硬件连线如下图所示：

```
assign IRQ  = {30'b0, interrupt_UART2, interrupt_UART1};
assign NMI  = 1'b0;
assign RXEV = 1'b0;
```

进入主程序(main.c)以后，首先进行测试平台的初始化，初始化过程包括定义循环变量，定义状态变量（compare_flag 和 communication_flag），定义计分板变量，定义中间变量（AHB_AES,chip_AES,key,text_in），对 UART 和 Sys tick 计时器初始化，配置 GPIO 端口模式（此处讲 GPIO[0]配置为推挽输出，硬件上连接至 AES 加密芯片的复位端），设置随机数生成器的 seed。初始化完成后，会通过 UART1 模块向电脑传输初始化完成

的信息：

```
int main()
{
  uint8_t i,j,k;
  uint8_t compare_flag;
  uint8_t communicate_flag;
  uint32_t correct = 0;
  uint32_t wrong = 0;
  uint32_t communication_err = 0;
  char temp;
  uint32_t AHB_AES[4];
  uint32_t chip_AES[4];
  uint32_t key[4];
  uint32_t text_in[4];
  delay_init();
  delay_ms(100);
  UART_Init(UART1,57600);
  UART_Init(UART2,115200);
  WriteGPIO_Mode(GPIOA,GPIO_Pin_0,GPIO_Mode_Output_PP);
  WriteGPIO_BSRR(GPIOA,GPIO_Pin_0,Reset);
  k=1;//防止显示过快
  TRNG->wrseed=0x19990121;
  printf("platform initialization done !\r\n");
```

初始化完成后进入主循环，自动测试启动。测试平台首先用随机数生成器产生符合 AES 芯片要求的密钥和明文，由于我们使用了 0d0a 作为芯片内部状态机的控制字段，因此需要从随机数发生器的输出中剔除控制字段，防止误触发状态机的跳转：

```
//key generation 加入逻辑判断防止产生0D0A控制字符
key[0]=TRNG->rddata;
key[1]=TRNG->rddata;
key[2]=TRNG->rddata;
key[3]=TRNG->rddata;
for(i=0;i<4;i++)
{
  for(j=0;j<3;j++)
  {
    if(((key[i]>>(8*j))&0xffff)==0x0d0a)
    {
      key[i]=0;
    }
  }
}
//text_in generation 加入逻辑判断防止产生0D0A控制字符（一旦产生控制字符，会触发状态机误切换）
text_in[0]=TRNG->rddata;
text_in[1]=TRNG->rddata;
text_in[2]=TRNG->rddata;
text_in[3]=TRNG->rddata;
for(i=0;i<4;i++)
{
  for(j=0;j<3;j++)
  {
    if(((text_in[i]>>(8*j))&0xffff)==0x0d0a)
    {
      text_in[i]=0;
    }
  }
}
```

随后向 AHB_AES 模块发送明文和密钥，进行加密操作得到密文：

```
//get AHB_AES results, which is totally true
AES->key0=key[0];
AES->key1=key[1];
AES->key2=key[2];
AES->key3=key[3];
AES->text_in0=text_in[0];
AES->text_in1=text_in[1];
AES->text_in2=text_in[2];
AES->text_in3=text_in[3];
while(AES->status==0);
AES->en=1;
while(AES->status==0);
AHB_AES[3]=AES->cipher_text3;
AHB_AES[2]=AES->cipher_text2;
AHB_AES[1]=AES->cipher_text1;
AHB_AES[0]=AES->cipher_text0;
```

得到 AHB_AES 模块输出的无条件正确密文后，按照设计时规定的时序向 AES 加密芯片发送同样的数据，并读出 AES 加密芯片的输出结果。读出结果的操作考虑了通信失败的情况，一旦通信超时，就放弃读取 AES 芯片的输出，并将 communicate_flag 置一以表示通信错误：

```
//send key and text_in to chip
WriteGPIO_BSRR(GPIOA,GPIO_Pin_0,Reset);
delay_us(10);
WriteGPIO_BSRR(GPIOA,GPIO_Pin_0,Set);
delay_us(10);
for(i=0;i<4;i++)
{
  temp = (key[3-i]>>24)&0xff;
  Write_TX(UART2,temp);
  temp = (key[3-i]>>16)&0xff;
  Write_TX(UART2,temp);
  temp = (key[3-i]>>8)&0xff;
  Write_TX(UART2,temp);
  temp = key[3-i]&0xff;
  Write_TX(UART2,temp);
}
UART_String(UART2,"\r\n");
delay_us(10);
for(i=0;i<4;i++)
{
  temp = (text_in[3-i]>>24)&0xff;
  Write_TX(UART2,temp);
  temp = (text_in[3-i]>>16)&0xff;
  Write_TX(UART2,temp);
  temp = (text_in[3-i]>>8)&0xff;
  Write_TX(UART2,temp);
  temp = text_in[3-i]&0xff;
  Write_TX(UART2,temp);
}
UART_String(UART2,"\r\n");
//get chip output
rx_counter = 0;
NVIC->ISER[0] = 0x2;
communicate_flag = 0;
delay_ms(7);
if(rx_counter != 16)
{
  communicate_flag = 1;
  printf("communication error !\r\n");
}
NVIC->ICER[0] = 0x2;
rx_counter = 0;
for(i=0;i<4;i++)
{
  chip_AES[i]=(chip_output_temp[i*4+3]<<24)+(chip_output_temp[i*4+2]<<16)+(chip_output_temp[i*4+1]<<8)+chip_output_temp[i*4];
}
```

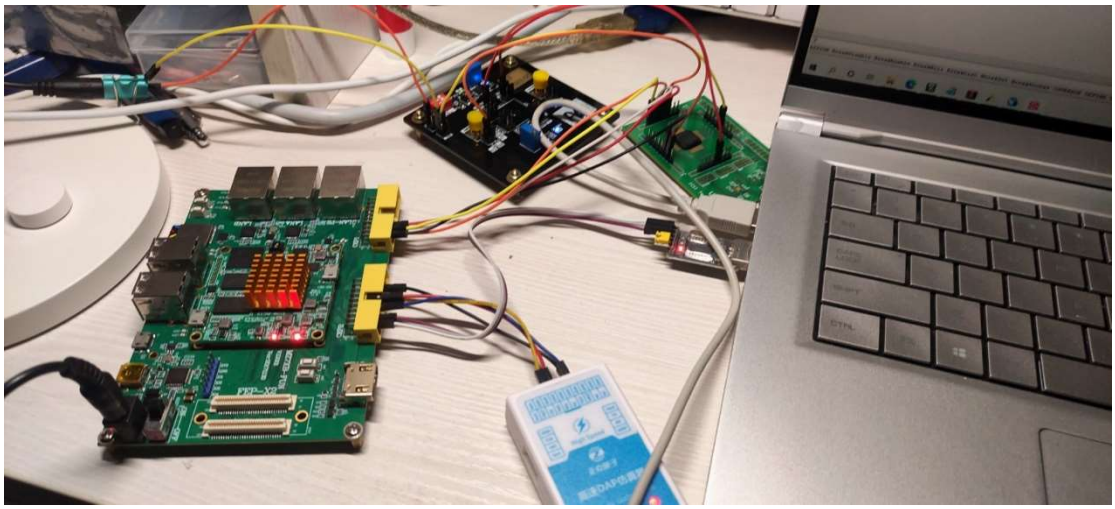
将读出的 chip_AES 与 AHB_AES 进行比对，如果比对正确，就在计分板上给 correct 变量加一，否则给 wrong 变量加一。计分板上的 communication_err 则只在 UART 通信超时的时候加一，表示 AES 芯片和 FPGA 间的通信有错误，这种错误可能来源于外部干扰，也可能来源于芯片内部的错误。自动测试平台的测试速度非常快，因此每测试 100

组才向计算机发送一次测试结果：

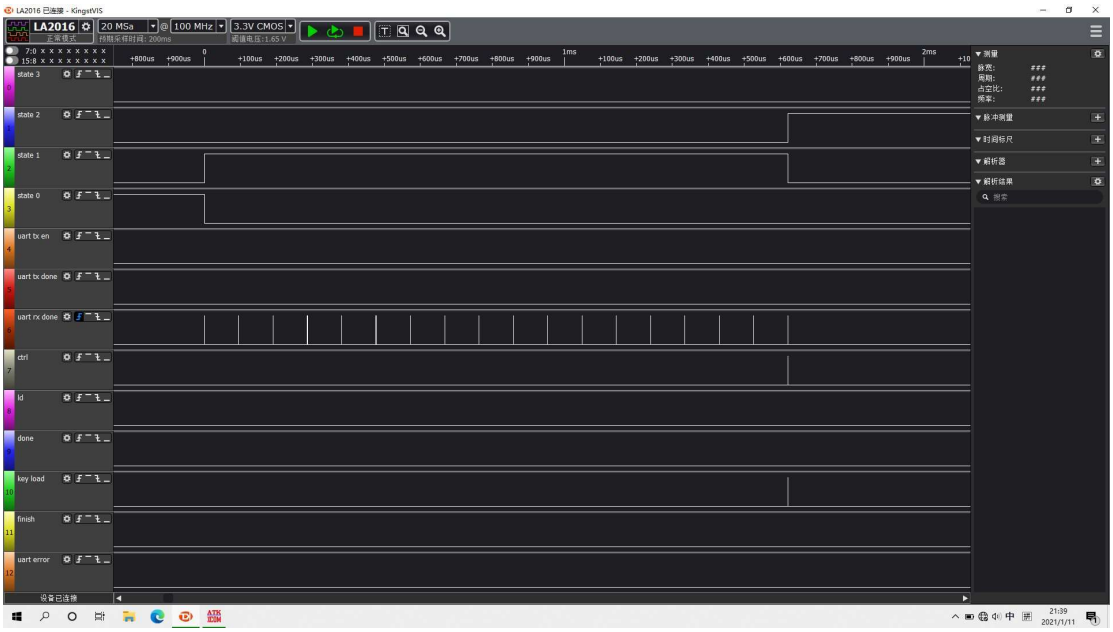
```
//compare chip output "chip_AES" and "AHB_AES"
compare_flag=0;
for(i=0;i<4;i++)
{
  if(chip_AES[i] == AHB_AES[i])
    compare_flag = compare_flag + 1;
}
if(compare_flag == 4)
  correct = correct + 1;
else
  wrong = wrong + 1;
if(communicate_flag == 1)
  communication_err = communication_err + 1;
compare_flag = 0;
if(k>=100)
{
  printf("correct=%d,wrong=%d,com_err=%d\r\n",correct,wrong,communication_err);
  k = 1;
}
else
{
  k = k + 1;
}
```

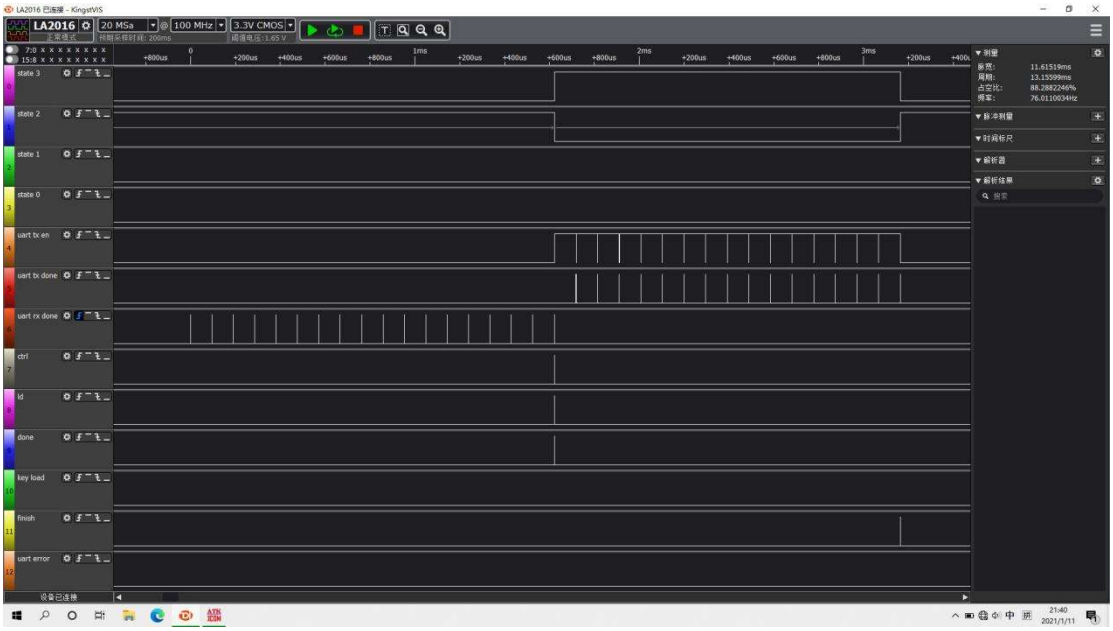
九、实验数据及结果分析

1、将自动测试平台进行板级连接：

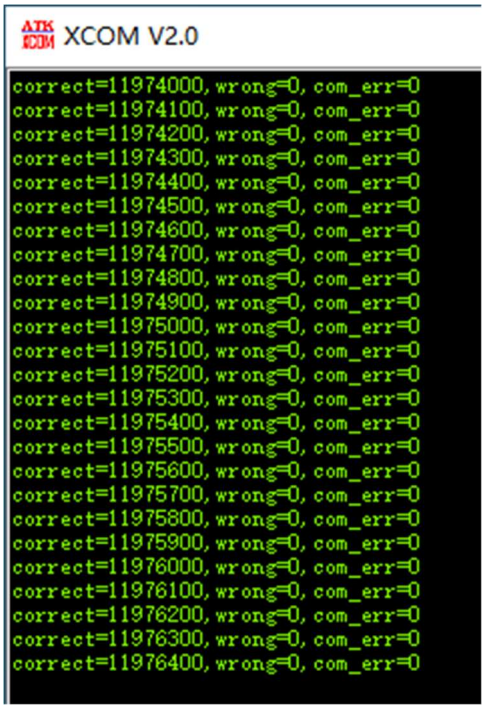


2、使用逻辑分析仪对内部信号进行抓取和分析：





3、验证数据信息输出：（验证数据条数逾千万且无一出错）



十、实验结论

成功基于 Zynq 7020 SoPC，搭建 AES 加密芯片测试平台，并实现对 AES 加密芯片的自动、大数据量随机测试，最后通过 UART 向上位机的串口调试助手发送结果。在验证过程中对 AES 芯片进行了 11976400 次加密测试（超过 1 千万组数据），测试环境温度 15.4℃到 23.1℃变化，时间跨度 1 天以上，传输数据量超过 593.6MB。测试结果全部正确，证明了我们组的芯片工作稳定，结果可靠。

十一、总结及心得体会

在搭建测试平台时遇到了不少问题，但经过和王忆文老师、助教以及小组成员沟通后解决了问题。此次测试实验让同学们有机会对自己 IC2 设计的芯片进行测试，对于集成电路设计与集成系统的学生是一个锻炼自身实践水平的机会，感谢学院和老师为我们精心准备的课程和教学。

十二、对本实验过程及方法、手段的改进建议

建议将测试实验大幅提前，临近期末有许多选修课和本实验冲突。