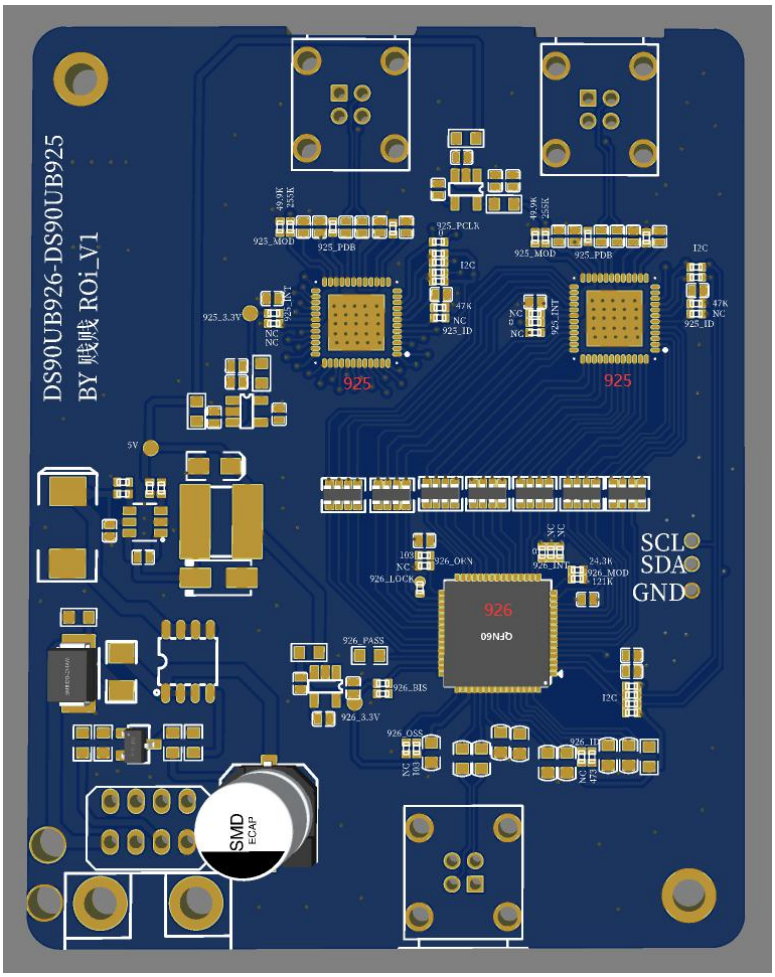
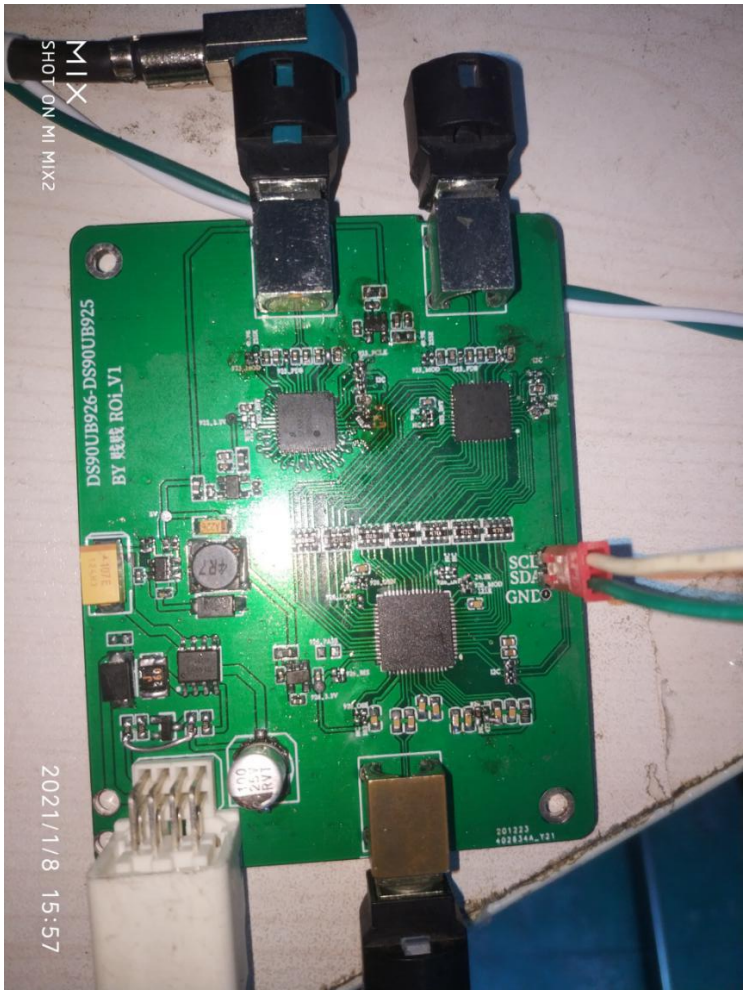
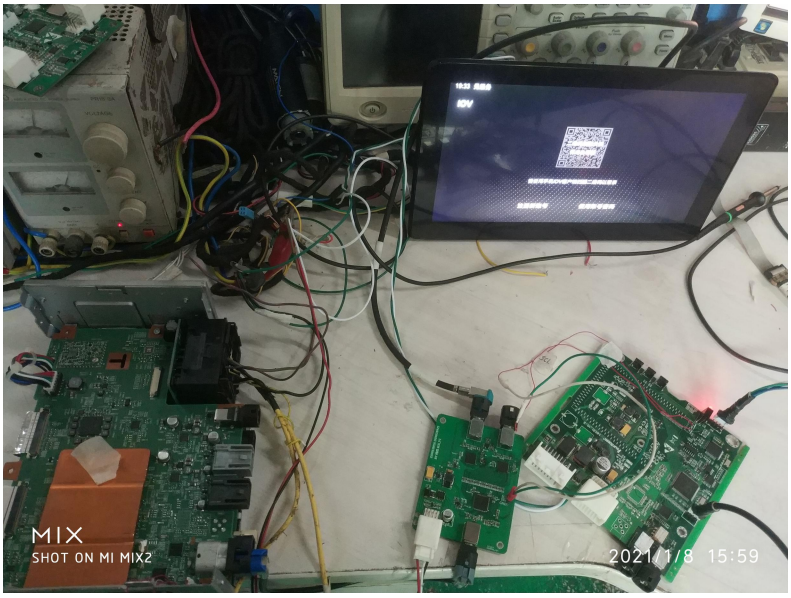
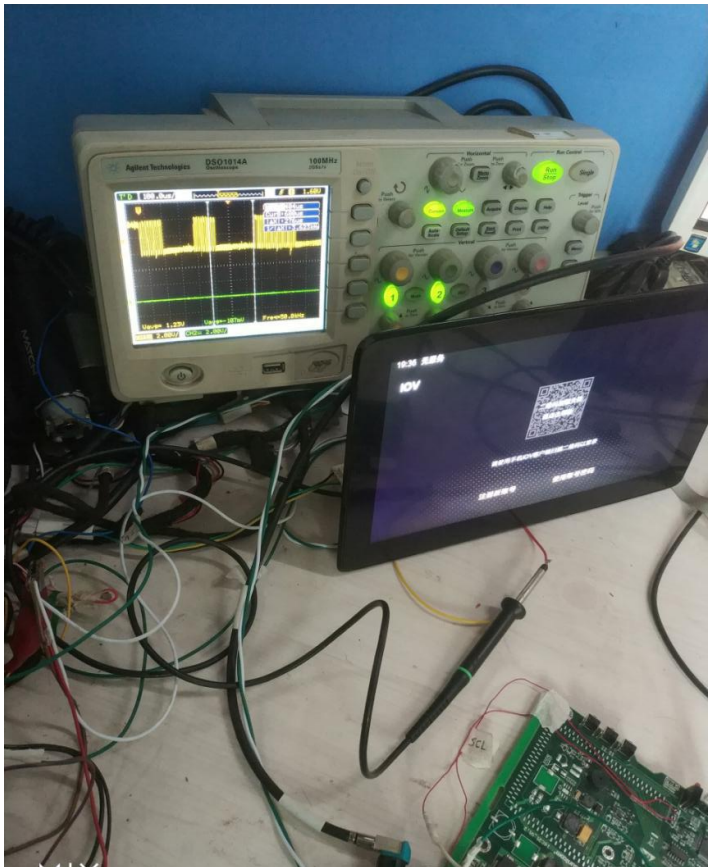

DS90UB926-DS90UB925
BY 贱贱 ROi_V1
925
925
926
QFN60
SCL
SDA
GND
SMD
ECAP


MIX
SHOT ON MI MIX2
DS90UB926-DS90UB925
BY 贱贱 ROi_V1
SCL
SDA
GND
2021/1/8 15:57


MIX
SHOT ON MI MIX2
2021/1/8 15:59


Agilent Technologies
DSO1014A
100MHz
10V
SCL

项目简介：

我们设计了一个模块使用 DS90UB926 和 DS90UB925,用来作为中继模式使用，用来增加一路视频输出。安装到原本的导航主机（使用 DS90UB921）和触摸显示模组（使用 DS90UB924）之间。

问题点：

我们将 926 和 925 的 i2c 寄存器都设置为 pass_all，在屏幕那端的 i2c 能测到波形，且两路图像输出显示正常，但是触摸无法使用，我们把中断打开，发现主板上的中断脚电平一直被拉低。

1# DS90UB925 ID 0x28
2#DS90UB925 ID 0x26
DS90UB926 ID 0x66

Project introduction:

We designed a module using ds90ub926 and ds90ub925, which is used as repeater mode to increase one channel video output. It is installed between the original navigation host (using ds90ub921) and the touch display module (using ds90ub924).

Problem points:

We set the I2C register of 926 and 925 to pass_ All, the I2C at the other end of the screen can detect the waveform, and the two-way image output display is normal, but the touch can't be used. We turn on the interrupt and find that the level of the interrupt pin on our board has been always pulled down.

1# DS90UB925 ID 0x28

2#DS90UB925 ID 0x26

DS90UB926 ID 0x66

```
669
670
671 #if 1//test
672
673     type = keycode&KEY_TYPE_MASK;
674     key  = keycode&(~KEY_TYPE_MASK);
675     if(key == BSP_PB_ID_KEY1){
676         if(type == KEY_RELEASED){
677
678
679 #if 0//948,949
744 #if 1
745             if(pwr_en_ctrl)//接收端 948:  0x58  ,  ///remote GPIO2,Pin10
746             {
747                 pwr_en_ctrl = 0;
748
749 //              WriteI2C(0x66,0x02,(0x00 | 0x40 | 0x10 | 0x04));//接收端 926: 0x05 OEN,OSS_SEL
750                 WriteI2C(0x66,0x02,0xF0);//接收端 926: 0x05 OEN,OSS_SEL
751                 WriteI2C(0x66,0x03,0xF8);//接收端 926: 0x03 //0xFC
752                 WriteI2C(0x66,0x05,0xAE);//接收端 926: 0x05 I2C Pass //0xAE
753 //              WriteI2C(0x66,0x06,0x04);//接收端 926: 0x05
754 //              WriteI2C(0x66,0x06,0x33);//接收端 926: 0x05
755 //              WriteI2C(0x66,0x07,0x1C);//接收端 926: 0x05
756
757
758 //              WriteI2C(0x58,0x03,0xFC);//接收端 926: 0x03
759 //              WriteI2C(0x58,0x02,(0x00 | 0x40 | 0x10 | 0x04));//接收端 926: 0x05 OEN,OSS_SEL
760 //              WriteI2C(0x58,0x02,0xF0);//接收端 926: 0x05 OEN,OSS_SEL
761 //              WriteI2C(0x58,0x05,0x9E);//接收端 926: 0x05 I2C Pass //0xAE
762 //              WriteI2C(0x58,0x06,0x04);//接收端 926: 0x05
763 //              WriteI2C(0x58,0x06,0x33);//接收端 926: 0x05
764
765
766                 printf("\r\nDS926 on");
767
768
769             }
770             else{
771                 pwr_en_ctrl = 1;
772                 WriteI2C(0x66,0x02,0x00);//接收端 926: 0x05 OEN,OSS_SEL
773                 WriteI2C(0x66,0x03,0xF0);//接收端 926: 0x03
774                 WriteI2C(0x66,0x05,0x2E);//接收端 926: 0x05 I2C Pass
775
776 //              WriteI2C(0x58,0x02,0x00);//接收端 926: 0x05 OEN,OSS_SEL
777 //              WriteI2C(0x58,0x05,0x1E);//接收端 926: 0x05 I2C Pass //0xAE
778
779
780                 printf("\r\nDS926 off");
781             }
782 #endif
783
784
785
786             #if 0
810 #if 0
828         }
829     }
830     else if(key == BSP_PB_ID_KEY2){
831         if(type == KEY_RELEASED){
832             //Driven_ChangeModeTo(MODE_OFFSET);
833
834             if(pwm_ctrl)//接收端 948:     0x58  ,  //remote GPIO5_REG,Pin11
835             {
836                 pwm_ctrl = 0;
837
838
```

DS90UB925.c | driven.c

```
836                 pwm_ctrl = 0;
837
838
839 //              WriteI2C(0x18,0x03,(0xD2 | 0x20 | 0x08));//0xFA//DA     //发送端 925: 0x03
840 //              WriteI2C(0x18,0x04,(0x80 | 0x10 | 0x08 | 0x04));//发送端 925: 0x05 //Backward compatible
841 //              WriteI2C(0x18,0x04,0x8E);//发送端 925: 0x05 Local Write
842 //              WriteI2C(0x18,0x05,(0x00 | 0x04));//发送端 925: 0x05 Local Write
843 //              WriteI2C(0x18,0x06,0x01);//发送端 925: 0x05
844 //              WriteI2C(0x18,0x17,0xDE);//发送端 925: 0x05
845 //              WriteI2C(0x18,0xC6,0x21);//发送端 925: 0x05 INTB
846
847
848 //              WriteI2C(0x28,0x02,0xF0);//发送端 925: 0x02
849 //              WriteI2C(0x28,0x03,(0xD2 | 0x20 | 0x08));//0xFA//DA        //发送端 925: 0x03
850 //              WriteI2C(0x28,0x03,0xDA);//0xFA//DA     //发送端 925: 0x03
851                 WriteI2C(0x28,0x03,0xDA);//0xFA//DA     //发送端 925: 0x03
852 //              WriteI2C(0x28,0x04,(0x80 | 0x10 | 0x08 | 0x04));//发送端 925: 0x05 //Backward compatible
853 //              WriteI2C(0x28,0x04,0x8E);//发送端 925: 0x05 Local Write
854 //              WriteI2C(0x28,0x05,(0x00 | 0x04));//发送端 925: 0x05 Local Write
855 //              WriteI2C(0x28,0x06,0x01);//发送端 925: 0x05
856                 WriteI2C(0x28,0x17,0xDE);//发送端 925: 0x05
857                 WriteI2C(0x28,0xC6,0x21);//发送端 925: 0x05 INTB
858
859 //              WriteI2C(0x26,0x03,(0xD2 | 0x20 | 0x08));//0xFA     //发送端 925: 0x03
860                 WriteI2C(0x26,0x03,0xDA);//0xFA     //发送端 925: 0x03
861 //              WriteI2C(0x26,0x04,(0x80 | 0x10 | 0x08 | 0x04));//发送端 925: 0x05 //Backward compatible
862 //              WriteI2C(0x26,0x04,0x8E);//发送端 925: 0x05 Local Write
863 //              WriteI2C(0x26,0x05,(0x00 | 0x04));//发送端 925: 0x03
864 //              WriteI2C(0x26,0x06,0x01);//发送端 925: 0x05
865                 WriteI2C(0x26,0x17,0xDE);//发送端 925: 0x05
866                 WriteI2C(0x26,0xC6,0x21);//发送端 925: 0x05 INTB
867
868
869 //              WriteI2C(0x1C,0x03,0xDA);//0xFA//DA     //发送端 925: 0x03
870 //              WriteI2C(0x1C,0x17,0xDE);//发送端 925: 0x05
871 //              WriteI2C(0x1C,0xC6,0x21);//发送端 925: 0x05 INTB
872
873
874                 printf("\r\nDS925 0x17=0xDE,I2C Pass ON");
875             }
876             else{
877                 pwm_ctrl = 1;
878
879
880 //              WriteI2C(0x28,0x02,0x00);//发送端 925: 0x02
881                 WriteI2C(0x28,0x03,0xD2);//发送端 925: 0x03
882                 WriteI2C(0x28,0x04,0x80);//发送端 925: 0x05
883 //              WriteI2C(0x28,0x05,0x00);//发送端 925: 0x05
884 //              WriteI2C(0x28,0x06,0x00);//发送端 925: 0x05
885                 WriteI2C(0x28,0x17,0x5E);//发送端 925: 0x05
886                 WriteI2C(0x28,0xC6,0x00);//发送端 925: 0x05 INTB
887
888                 WriteI2C(0x26,0x03,0xD2);//发送端 925: 0x03
889 //              WriteI2C(0x26,0x04,0x80);//发送端 925: 0x05
890 //              WriteI2C(0x26,0x05,0x00);//发送端 925: 0x03
891 //              WriteI2C(0x26,0x06,0x00);//发送端 925: 0x05
892                 WriteI2C(0x26,0x17,0x5E);//发送端 925: 0x03
893                 WriteI2C(0x26,0xC6,0x00);//发送端 925: 0x05 INTB
894
895
896 //              WriteI2C(0x1C,0x03,0xD2);//0xFA//DA     //发送端 925: 0x03
897 //              WriteI2C(0x1C,0x17,0x5E);//发送端 925: 0x05
898 //              WriteI2C(0x1C,0xC6,0x00);//发送端 925: 0x05 INTB
899
900
901                 printf("\r\nDS925 0x17=0x5E,I2C Pass OFF");
```