Si522A ACD 功能初始化代码解析

前言：

本白皮书描述了如何快速的熟悉 Si522A 系列芯片的寄存器配置以及如何获得相关寄存器的最佳配置。本文以 Si522A 为例，说明的方法适用于南京中科微 13.56MHz 带 ACD 功能的 Reader 芯片。文中，涉及的低功耗 12/14 通道触摸芯片为我司的 Si12T,Si14T。

本白皮书不包括天线调谐，请参考《Si522A 13.56MHz 射频匹配调试建议手册》。

更新记录

创建日期	2021/08/01	原作者	Jack Ren	完成日期	2021/08/08
变更记录					
变更日期	变更人	变更摘要			
2021/08/08	Jack Ren	版本 V0.01，完成 Si522A ACD 功能初始化代码解析初步文档			

```
void ACD_init_Fun(void)
{
  PCD_SI522A_TypeA_Init();  //

  PCD_ACD_AutoCalc(); //自动获取阈值

  PCD_ACD_Init();
}
```

ACD 初始化部分，分为三部分。
第一部分的 PCD_SI522A_TypeA_Init 为刷卡部分的正常配置。
第二部分的 PCD_ACD_AutoCalc 为自动计算 ACD 模式下的 ADC 的设置值
第三部分的 PCD_ACD_Init 为填入 ACD 模式的寄存器值并且进入 ACD 轮询状态。

Tips1： ACD 模式实际为 APPL(Auto Low Power Polling Loop), Si522A 在此阶段寻找射频卡。Si522A 先发射载波然后检测 13.56MHz 载波幅度变化。若载波幅度变化大于设定阈值则判定为有卡并产生中断。 产生中断后，芯片自动退出 ACD 模式，进入 Reader 模式，MCU 操作相应指令完全后续读卡即可！

Tips2: ACD 模式下，建议客户用硬复位操作替代软复位操作！尽管上位机生成的程序大量使用软复位，还是建议用户使用硬复位，因为整机应用中，静电或者其他未知因素会影响软复位操作的稳定性。

Part1：

芯片 Reader 模式正常初始化

```
void PCD_SI522A_TypeA_Init(void)
{

  I_SI522A_ClearBitMask(Status2Reg, 0x08);
  // Reset baud rates
  I_SI522A_IO_Write(TxModeReg, 0x00);
  I_SI522A_IO_Write(RxModeReg, 0x00);
  // Reset ModWidthReg
  I_SI522A_IO_Write(ModWidthReg, 0x26);
  // RxGain:110,43dB by default;
  I_SI522A_IO_Write(RFCfgReg, RFCfgReg_Val);
  // When communicating with a PICC we need a tim
  // f_timer = 13.56 MHz / (2*TPreScaler+1) where
  // TPrescaler_Hi are the four low bits in TMode
  I_SI522A_IO_Write(TModeReg, 0x80);// TAuto=1; t
  I_SI522A_IO_Write(TPrescalerReg, 0xa9);// TPreS
  I_SI522A_IO_Write(TReloadRegH, 0x03); // Reload
  I_SI522A_IO_Write(TReloadRegL, 0xe8); // Reload
  I_SI522A_IO_Write(TxASKReg, 0x40);  // Default
  I_SI522A_IO_Write(ModeReg, 0x3D); // Default 0x
  I_SI522A_IO_Write(CommandReg, 0x00);  // Turn o

  PcdAntennaOn();
}
```

使用 NFC 芯片都会有这样一个初始化函数，Si522A 的这一部分的初始化可以与其他类似 NFC 芯片的初始化兼容。用户在移植程序时可以不作更改即可！（如客户使用的 Si523 读 B 卡部分，请将该部分替换为 B 卡的初始化部分）

Tips:

ACD 配置寄存器基本处于 PollReg 中，在地址 0x0F 下埋了 16 组寄存器，其访问方式与普通的寄存器地址存在一定的差异。

普通地址 SPI 接口寻址：

表 133　地址字节格式

地址（MOSI）	位 7，MSB	位 6—位 1	位 0
字节 0	1（读） 0（写）	地址	RFU（0）

7.4.1 PageSelReg

表 63　PageSelReg　地址：20h　复位值：00h

	7	6	5	4	3	2	1	0
	UsePage Select	Regbank Select	RegSelect				PageSelect	
访问权限	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

表 64　PageSelReg 位描述

位	符号	功能
7	UsePageSelect	设置为 1 时，PageSelect 的值被视为寄存器地址 A5 和 A4。寄存器地址的低位则分别由地址引脚和内部地址锁存决定； 设置为 0 时，寄存器地址完全由内部地址锁存所决定。地址引脚的描述见 9.1 节
6	RegbankSelect	设置为 1 时，可以读写 0Fh 寄存器组；设置为 0 时，可以访问 37h 寄存器
5-2	RegSelect	RegbankSelect 为 0 时，若 Regselect 为： 0000：读写 A 组寄存器； 0001：读写 B 组寄存器； … 1111：读写 P 组寄存器
1-0	PageSelect	PageSelect 的值只有在 UsePageSelect 为 1 时才有效，此时用于指定寄存器页（即寄存器地址 A5 和 A4）

```
void SI522A_SPI_LL_WriteRawRC(unsigned char RegAddr,unsigned char value)
{
    CSN = 1;
    RegAddr = (RegAddr & 0x3f) << 1;                    //code the first byte
    CSN = 0;
    SPI1_ReadWriteByte(RegAddr);    //write address first
    SPI1_ReadWriteByte(value);                  //write value then
    CSN = 1;
}
#define     ACDConfigSelReg         0x20
```

首先，通过配置 PageSelReg 寄存器，0x20。 SPI 操作函数中，地址选择为[6:1]，即寄存器的地址为 RegAddr = (RegAddr & 0x3f) << 1;由于 0F_G 属于下埋寄存器，其地址选择为[5:2]，并且 bit6 必须为 1，故需要对 0F_G 寄存器的地址进行移位操作：(ACDConfigG << 2) | 0x40)。选中 0F_G 后，可以对其正常操作，读和写数据即可。

以下以操作演示访问 0F_G 寄存器：

```
I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigG << 2) | 0x40);
temp_Compare = I_SI522A_IO_Read(ACDConfigReg);
```

注意：通过配置通过配置 PageSelReg 寄存器下的下埋寄存器后，你再读取 0x37 寄存器，你需要对 PageSelReg 寄存器写 00（对 bit6 写 0），否则读取 0x37 寄存器的值为 0xb2，并不是 0x92！！！

Part 2：

主要为了计算 0F_K 寄存器的配置值（ACDConfigK_Value）

0F_K 的寄存器表：

2.2.11 0F_K LPDConfig1

RFCC：RF CHECK CONTROL，检卡时使用。

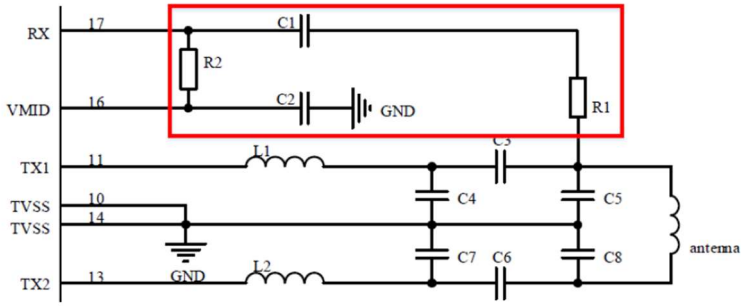
Addr	Bit	Name	R/W/D	Reset	Description
0x0a		LPDConfig1		0fh	
	7	LPDTestEn	R/W	0	1：强制打开检波器 0：此位无效
	6:5	TR	R/W	00	检波电路中减法器增益控制字。 TR<1:0> Gain 00 1 倍（0dB） 01 3 倍 (9.5dB) 10 7 倍 (16.9dB) 11 15 倍 (23.5dB)
	4:3	TI	R/W	01	检波电路中前段检波运放斜率控制字。 TI<1:0> K 00 0.5 01 1 10 1.5 11 2 (输入信号 VPP>1V 或者 VDD<2.8V 不建议使用)
	2:0	VCON	R/W	111	LDO 电阻分压输出给检波电路 8 组电压值可选 默认 111 1.8V T_CON<2:0> VCON 000 1.34V 001 1.307V 010 1.472V 011 1.537V 100 1.603V 101 1.66V 110 1.718V 111 1.9V

程序分为两部分：

第一部分，通过 RX 引脚电路上的电压 Vrx，调整 VCON 电压档位。档位为 8 档。TI 默认使用 01，故 VCON_TR 的 bit4 为 1！

```
unsigned char VCON_TR[8]={ 0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f};//acd灵敏度调节
```

Tips：由于 RX 电压通过分压后，要求 Vrx 的电压高于 1.4V，故要求天线场强的峰峰值 VPP 值至少大于 6V，RX 分压电路的电阻必须按照要求符合参考设计的 820Ω(R2) 与 5.1KΩ (R1) 电阻。在实际工作中，一般要求 VCON 的选值在第三档以上，即 0x0a 以上，如不符合需要先调整天线的匹配电路，让天线的场强的峰峰值 VPP 超过 6V。（调整方式参考 《Si522A 13.56MHz 射频匹配调试建议手册》）



隔直滤波电容 C1,C2 推荐使用 0.1uF，在示例 dome 板中 R1 为 5.1KΩ，R2 为 820Ω。不建议随便取值，只在推荐值上下微调阻值。其中，R1 接到

程序逻辑：

1. **首先，打开天线场强**

```
I SI522A IO Write(TxControlReg, 0x83);  //打开天线
```

通过对 0x14 寄存器 TxControlReg [1:0] 配置 11，打开天线场强。

7.3.5 TxCtrlReg

控制天线驱动管脚 TX1 和 TX2 的逻辑特性。

表 45 TxCtrlReg 地址：14h 复位值：80h

	7	6	5	4	3	2	1	0
	InvTx2RFOn	InvTx1RFOn	InvTx2RFOff	InvTx1RFOff	Tx2CW	RFU	Tx2RFEn	Tx1RFEn
访问权限	r/w	r/w	r/w	r/w	r/w	-	r/w	r/w

表 46 TxCtrlReg 位描述

位	符号	功能
7	InvTx2RFOn	设置为 1 时，如果 TX2 驱动开启，则 TX2 管脚的输出信号反相
6	InvTx1RFOn	设置为 1 时，如果 TX1 驱动开启，则 TX1 管脚的输出信号反相
5	InvTx2RFOff	设置为 1 时，如果 TX2 驱动关闭，则 TX2 管脚的输出信号反相
4	InvTx1RFOff	设置为 1 时，如果 TX1 驱动关闭，则 TX1 管脚的输出信号反相
3	Tx2CW	设置为 1 时，管脚 TX2 持续输出未调制的 13.56MHz 载波； 设置为 0 时，Tx2CW 使能调制载波信号
2	RFU	保留为将来使用
1	Tx2RFEn	设置为 1 时，管脚 TX2 输出由待传输数据调制的 13.56MHz 载波
0	Tx1RFEn	设置为 1 时，管脚 TX1 输出由待传输数据调制的 13.56MHz 载波

2. **通过 ADC_EXCUTE 命令，打开轮询 ADC**（即检波模块，是 ACD 模块中的一部分）只在进入 ACD 模式后自动打开或通过写 ADC_EXCUTE 命令强制打开。使能轮询 ADC 打开后，等待 100us 以上，轮询 ADC 约每 1.5us 采集一次，记录数据并存入 0F_G[6:0]。

0F_G		ADCVal		xx	轮询ADC采样值
	7	RFU	-	0b	保留为将来使用
	6:0	VAL_ADC	R	x	ADC 采样值

```
I_SI522A_SetBitMask(CommandReg, 0x06);  //开启ADC_EXCUTE
```

通过对 0x01 寄存器 CommandReg，写命令码 0110.

a. 程序在 VCON_TR 数组中从 0x0f 开始遍历，将数组的值逐一写 0F_K，然后通过查看 0F_G 的采样值，找到 Vrx-VCON 后的最小非 0 值。

```
for(int i=7; i>0; i--)
{
  I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigK << 2) | 0x40);
  I_SI522A_IO_Write(ACDConfigReg, VCON_TR[i]);

  I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigG << 2) | 0x40);
  temp_Compare = I_SI522A_IO_Read(ACDConfigReg);
  for(int m=0;m<100;m++)
  {
    I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigG << 2) | 0x40);
    temp = I_SI522A_IO_Read(ACDConfigReg);
    if(temp ==  0)
    {
        temp_Compare  = 0;
        break;
    }
    temp_Compare=(temp_Compare+temp)/2;
    delay_us(100);
  }

  if(temp_Compare == 0 || temp_Compare == 0x7f) //比较当前值和所存值
  {

  }
  else
  {
    if(temp_Compare < ACDConfigRegC_Val)
    {
      ACDConfigRegC_Val = temp_Compare;
      ACDConfigRegK_Val = VCON_TR[i];
    }
  }
}//取得最接近的参考电压VCON
ACDConfigRegK_RealVal = ACDConfigRegK_Val;
```

Tips: 在遍历过程中，每个 VCON 值被写入 0F_K 后，会读取 100 次采样值，（如果 100 次中有一次出现采样值为 0 时，那么也需要判断该 VCON 值下，Vrx-VCON 已经出现采样为 0 值，需要舍弃，VCON 的值需要取上一个值。如果最终计算下来，ACDConfigRegK_Val 的值的 bit[3:0]非 8 组 VCON 数组中的值，例如值为 60，那么就意味着计算失败，失败的原因可能时 MCU 与 Si522A 通讯异常或者天线场强没有被打开导致，请在打开天线场强命令后加延时或者调整数字接口代码后再次尝试。

b. 选中 VCON 后，ADC 采样值=（Vrx-VCON）*TR。其中 TR 就是增益放大倍数，ADC 采样值的最大阈值为 7F，在选择 TR 时，在 TR 和选中的 VOCN 填入 0F_K 寄存器后，需要判断 0F_G 的采样值不超过 7F 或者在 7F 附近。选中 VCON 的档位同时，也配置了 0F_C 寄存器的无卡场强采样值。

```
  for(int j=0; j<4; j++)
  {
    I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigK << 2) | 0x40);
    I_SI522A_IO_Write(ACDConfigReg, j*32+ACDConfigRegK_Val);

    I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigG << 2) | 0x40);
    temp_Compare = I_SI522A_IO_Read(ACDConfigReg);
    for(int n=0;n<100;n++)
    {
      I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigG << 2) | 0x40);
      temp = I_SI522A_IO_Read(ACDConfigReg);
      temp_Compare=(temp_Compare+temp)/2;
      delay_us(100);
    }
    TR_Compare[j] = temp_Compare;
  }//再调TR的档位，将采集值填入TR_Compare[]

  for(int z=0; z<3; z++)
  {
    if(TR_Compare[z] == 0x7f)
    {

    }
    else
    {
      ACDConfigRegC_Val = TR_Compare[z];//最终选择的配置
      ACDConfigRegK_Val = ACDConfigRegK_RealVal + z*32;
    }
  }//再选出一个非7f大值
  I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigK << 2) | 0x40);
  printf("\r\n ACDConfigRegK_Val:%02x ",ACDConfigRegK_Val);

  I_SI522A_SetBitMask(CommandReg, 0x06);     //关闭ADC_EXCUTE
}
```

TR 档位为四档增益可调，在实际应用中发现，在使用第四档的时候，容易出现 0F_G 的采样值过高，接近于 7F 且波动较大，导致容易误触发，故一般只建议使用前三档。

基于此，通过 VCON 和 TR 的选择，0F_K（ACDConfigK 寄存器）的值就全部计算出来了。关于 TI 的选择，TI 会改变 ADC 的量程，默认的 TI 量程选择为第二档，ADC 的量程加大会降低 ADC 的精度，在使用时，如果天线的场强峰峰值 VPP 超过 12V 时，可以尝试使用第三档 TI，其他场强区间不建议用户进行调整。

在成功计算获取 0F_K 寄存器的值后，需要关闭轮询 ADC 功能，即再次调用 ADC_EXCUTE 命令即可。

Part 3：

配置基本工作的 ACD 寄存器的值，以下 ACDConfig x 寄存器，均以 0F_x 来描述：

```
void PCD_ACD_Init(void)
{
   I_SI522A_IO_Write(DivIrqReg, 0x60); //清中断
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigJ << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, 0x55);  //Clear ACC_IRQ

   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigA << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegA_Val );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigB << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegB_Val );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigC << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegC_Val );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigD << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegD_Val );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigH << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegH_Val );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigI << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegI_Val );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigK << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegK_Val );

//  I_SI522A_IO_Write(ACDConfigReg, 0x4c );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigM << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegM_Val );
   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigO << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegO_Val );

   I_SI522A_IO_Write(ComIEnReg, ComIEnReg_Val);  //IRqInv
   I_SI522A_IO_Write(DivIEnReg, DivIEnReg_Val);

   I_SI522A_IO_Write(ACDConfigSelReg, (ACDConfigJ << 2) | 0x40);
   I_SI522A_IO_Write(ACDConfigReg, ACDConfigRegJ_Val );

   I_SI522A_IO_Write(CommandReg, 0xb0);  //进入ACD
}
```

a. 需要对 DivIraqReg，进行清中断操作。（注意：此操作不能少，否则进入 ACD 模式后，会产生寻卡中断）bit[6:5]，为场强中断使能，定时器中断。

b. 对 0F_J 寄存器写 0x55，清除 ACCEn。0F_J 寄存器是用来监测重要寄存器的值，防止其丢失后，ACD 功能异常，参与校验的轮询配置为：

{1'b0,DivIEnReg[6:5],5'b00000
}+

{TxControlReg[7],5'b0 ,TxContr
olReg [1:0]}+
0F_B+0F_C+0F_D+0F_E
+{1'b0,0F_I[6:4],1'b0,0F_I[2:1],
1'b0}+0F_J+0F_K

寄存器有：DivIEnReg，TxControlReg，0F_B，0F_C，0F_D，0F_E，0F_I，0F_J，0F_K；校验和每次 ACD 轮询时，都会判断校验和，即可判断寄存器丢失，产生中断通知用户刷新配置，重新初始化寄存器！！！

（注意：在写相关 ACD 寄存器时，必须要先清除 ACCEn 使能，否则后面写完 ACD 寄存器后，立马就会报 ACC_IRQ。清除 ACC_IRQ 中断，对 0F_J 写 0x55 即可）

0F_J ACC 寄存器表：

2.2.10 0F_J ACC

ACC: ACD CONFIGURE CHECK，轮询配置数据监测

Addr	Bit	Name	R/W/D	Reset	Description
0x09		ACC			
	7	ACCErr	R	0	0：轮询配置数据没有丢失 1：轮询配置数据丢失 仅在 ACCEn 为 1 的情况下有效 1. 参与校验的轮询配置： {1'b0,DivIEnReg[6:5],5'b00000}+ {TxControlReg[7],5'b0 ,TxControlReg [1:0]}+ 0F_B+0F_C+0F_D+0F_E +{1'b0,0F_I[6:4],1'b0,0F_I[2:1],1'b0}+0F_J+0F_K
	6	ACCEn	R/W	x	校验原理见附图 1 非轮询模式写 0 轮询模式写 1 写 0：只有 0F_J 写 55h 才能将 ACCEn 清 0 写 1：写非 55h 的任意值置 1
	5:0	-	R	0	

c. 0F_A 寄存器，设置轮询间隔时间，用户只需配置[5:0] mdelay:例如，寻卡间隔为 200ms，写 0x02 即可。

Addr	Bit	Name	R/W/D	Reset	Description
0x00		RCConfig1		05h	
	7	Trimsel	R/W	0	1：选择手动校正 0：选择自动校正
	6	Max	R/W	0	1：使能精校正
	5:0	mdelay	R/W	00101b	自动轮询间隔 mdelay*100ms，最小 100ms，最大 6400ms，也即休眠时间

d. 0F_B 寄存器，ACD 模式配置

Addr	Bit	Name	R/W/D	Reset	Description
0x01		ACDConfig		02h	
	7:6	ACDEdge	R/W	00	10：下降沿 01：上升沿 00/11：双沿
	5	ACDMode	R/W	0	0：绝对值比较 1：相对值比较
	4:3	ACDRFEn	R/W	00	01：使能低功耗卡检测（ADC 采样 4 次的平均值） 10：使能低功耗 RF 检测（ADC 采样采样 7 次，取后 4 次平均值） 00/11：同时使能低功耗卡和 RF 检测
	2:1	MaskAcd	R/W	01b	自动轮询模式下 00：前 2 次检卡不产生中断（SI522 设计） 01：前 3 次检卡不产生中断 10：前 4 次检卡不产生中断 11：前 5 次检卡不产生中断
	0	Reserve	R/W	0	

Bit[7:6] 可以选择检测的场强值变化状态，由于天线匹配设计，Tag 标签和 Reader 天线互感时（卡片接近读卡器天线时），天线的场强可能会变大或者变小，从而 ADC 采样的值可能会变大或者变小，一般情况基本上采样的值会变小，采用下降沿。(不建议使用上升沿，由于卡片远离读卡器天线时，触发中断，但是可能会由于挪开的距离超过最大读卡距离，造成读卡失败的现象。不过兼顾需要保证读卡稳定性的客户，可以尝试使用双边沿，但是软件上要做好中断屏蔽工作—：靠近和离开天线都会产生 ACD 中断)

5 7：6	0：绝对值比较	1：相对值比较
00/11(双沿)	ValDelta<ValSet 或 ValDelta>ValSet	(CSample – Lsample)<ValDelta 或 (CSample – Lsample)>ValDelta
10(下降沿)	ValDelta<ValSet	(CSample – Lsample)<ValDelta
01(上升沿)	ValDelta>ValSet	(CSample – Lsample)>ValDelta

简语定义：**LSample**(Last Sample):上一次轮询 ADC 采样值，**CSample**(Current Sample):本次轮询 ADC 采样值。轮询 ADC 约每 1.5us 采集一次，记录数据并存入 0F_G。绝对值比较即**电平触发**，相对值比较即**边沿触发**。

ValSet 在 0F_C 中手动设置（初始值 0x3f）；ValDelta 在 0F_D 中手动设置（初始值 0x0f）。

Bit[4:3] ACDREFn，我们一般设置为 01，检测卡，由于是 ADC 采样 4 次的平均值，相对应的稳定性更高。

Bit[2:1] MaskAcd， 该位置可以设置，可以设定在 ACD 中断产生后，在接下来的几次唤醒都不产生中断标志。在产生 ACD 中断后，防止读卡后的电源不稳定造成天线场强会改变不稳定的情况，从而产生异常的 ACD 唤醒中断增加功耗。

Tips: 使用绝对值比较的优势：只需要一次有效 ADC 采样值与设定的 0F_C 中的 ValSet 的值比较即可判断是否产生有卡中断。劣势：无法对器件老化进行自动的调整设定值。

相对值比较的优势：环境的变化因素对唤醒是否产生的影响小，他需要两次采样的值进行比较。劣势：判断中断的间隔需要两次 mdelay，例如 200ms 的寻卡间隔，需要 400ms 来判断，即产生中断的时间会略长。

e. 0F_C ManRefVal 手动设置无卡场强参考值

Addr	Bit	Name	R/W/D	Reset	Description
0x02		ValSet		70h	
	7	Reserve	R	0	
	6:0	VAL_SET	R/W	1110000 b	手动设置无卡场强参考值

Tips：无卡场强值默认值为 0x70，但是 ADC 采样值一般在 0x60 附近，故没有改变设置时，很难产生唤醒中断，用户可以在实际安装场景中进行模拟使用取采集 ADC 的值作为 0F_C 的参考值。

f. 0F_D ValDelta 场强变化范围

Addr	Bit	Name	R/W/D	Reset	Description
0x03		ValDelta		0fh	
	7	Reserve	R	0	
	6:0	VAL_DELTA	R	0001111 b	场强变化范围设置

0F_D 寄存器我们一般将其称呼为灵敏度，该值用来作为比较值的结果判断。相对比较模式下：第一次采样的值 – 第二次采样的值 > ValDelta，即产生唤醒中断退出轮询，自动进入 Reader 模式，否则不产生中断继续轮询。 绝对比较模式： 第一次采样的值 – 0F_C 的无卡场强值 > ValDelta ，即产生唤醒中断退出轮询，自动进入 Reader 模式，否则不产生中断继续轮询。

Tips：通过设置 0F_D 寄存器的值，我们可以让唤醒的成功率增加，天线尺寸小或者匹配一般的用户可以将该值设置小一点例如 2 来测试，但是比较好的匹配情况下，建议 ValDelta>=4;否则会产生误触发！匹配部分调试请参考《Si522A 13.56MHz 射频匹配调试建议手册》。

g. 0F_F RCConfig1 用于使能 OSC 27.012MHz 晶振监测功能。

Addr	Bit	Name	R/W/D	Reset	Description
0x05		RCConfig1		C0h	
	7	OMEN	R/W	1	1：使能 OSC 监测功能 0：关闭 OSC 监测功能
	6:0	TRIMSET	R/W	1000000b	手动设置 RCOSC 校正值

开启 OSC 监测功能的原因：因为 OSC 在频繁工作时，有概率无法正确起振，Si522A 在驱动晶振时内部会逐渐加大驱动能力，直到最大值也无法将晶振起振时，便触发 OSC 起振失败中断。Bit7 为开启使能监测操作。Bit[6:0]使用默认值即可。

h. 0F_G ADCValue 轮询 ADC 采样值，该寄存器为只读寄存器，可以读出当前的 RX 采样值。

Addr	Bit	Name	R/W/D	Reset	Description
0x06		ADCVal		xx	
	7	-	R	0	
	6:0	VAL_ADC	R	x	ADC 采样值

读取 0F_G 寄存器的值，需要轮询 ADC 功能开启，开启有两种办法，一种是进入 ACD 模式后，一种是通过写 ADC_EXCUTE 命令强制打开。使能轮询 ADC 打开后，等待 100us 以上，轮询 ADC 约每 1.5us 采集一次，记录数据并存入 0F_G[6:0].

i. 0F_H 看门狗间隔设置

Addr	Bit	Name	R/W/D	Reset	Description
0x07		WDTCounter		26h	
	7:0	WDT_CNT	R/W	00100110b	轮询模式下，设置唤醒间隔（唤醒失败 N 次就唤醒芯片）

OF_H 寄存器复位值为 0x26，也即 26 个轮询周期（time=26*mdelay*100 ms）内没有触发 ACD 轮询中断，则自动退出 ACD 轮询，产生看门狗中断。看门狗会自动清零，但是看门狗中断需手动清零。清零为 0x05 寄存器，IRq2Reg bit5--WdtIRq, 写 1 清除标志。
看门狗中断的使能是在 0x03 IRq2EnReg bit5—WdtIRqEn 写 1 使能中断。

j. 0F_I 下埋地址 0x08 ARI :ACD RF Indicator ACD RF 场强提示，该功能复用使用 Si522A PIN25 D1 引脚。（只有在置 0F_I bit1 ARIEn 为 1 后，D1 脚才作为 ARI 使用。）

0F_I		ARI		00h	
	7:3	RFU	-	-	保留为将来使用
	2	ARIPol	r/w	0b	ARI 极性控制 1：ARI 低电平有效 0：ARI 高电平有效
	1	ARIEn	r/w	0b	ARI 使能 1：使能，即 D1 输出 ARI 0：不使能，即不影响 D1 引脚状态
	0	ARI	R	x	ACD 模式下 RF 状态指示 轮询模式下比天线早开 1us，比天线使能晚关 1us

ARI 功能，开启后，可以在轮询 ACD 模式下，每次探测卡打开场强之前提前 1us 在 D1 引脚输出高或低电平，在探测结束后 1us，在 D1 脚停止输出电平。

Tips：该功能可以很好的解决，ACD 轮询场强打开时，触摸芯片会有概率误触发中断，例如配合我司的 Si12T 或 Si14T 芯片（12 键或 14 键触摸芯片）的 PIN8 SCT 硬件屏蔽功能，ARI 会提前 1us 输出高电平，将 SCT 拉高，触摸芯片会硬件进入休眠，在 ACD 探测结束后，ARI 不再输出电平，触摸芯片会进入扫描键盘状态，两者互不干扰！

k. 0F_M RFLowDetect 自动轮询期间低 RF 监测配置

Addr	Bit	Name	R/W/D	Reset	Description
0x0c		RFLowDetect		08h	
	7	RFLowDetectEn	R/W	0	1：使能 Reader 所发 RF 异常检测 0：关闭 Reader 所发 RF 异常检测
	6:0	RFLowThreshold	R/W	0001000	检卡时判断 RF 是否过低 阈值可选范围 0~128

设置[6:0]检卡阈值为 X，置 1bit7 使能 RF 低场强监测。若 ACD 过程中 ADC 采集到的场强值小于阈值 X，则触发中断。由于异常情况可能导致芯片配置丢失，场强不开启，这个时候可以提示 MCU 进入复位重新配置 Si522A.

Tips: 通常这个功能更适合整机设备调试，将设备安装好了后，通过开启 ADC_EXCUTE 命令，打开 ADC 功能，读取 0F_G 寄存器的值，该时的值为无卡场强值 ADCValue1，将 Tag 标签大程度贴近已经安装亚克力板的中间，读取该状态下的 0F_G 寄存器的值 ADCValue2。该 ADCValue2 就可以设置为 X。例如，将 X 设置为 1 在天线裸漏时，将 Tag 标签过去贴近 Reader 天线时，极有可能 ADCValue2 的值为 0，本该产生 ACD 中断的，确因为 ADC 采集的值为 0，产生 RF 低场强中断，该点需要用户去注意的。（Vrx-VCON）*A 为采样计算公式，当标签过于贴近 Reader 天线后，LC 谐振的场强会变小，有概率 Vrx<VCON，从而 ADC 超出量程，显示为 0！！！

l. 0F_O ACRDIrqEn 自动轮询其他相关中断使能（配置同 0x02,0x03（非 0F 复用）寄存器中关于 IRQ 引脚的描述）

Addr	Bit	Name	R/W/D	Reset	Description
0x0e		ACRDIrqEn		00h	
	7:4	-		0	
	3	OSCMonIrqEn	R/W	0	1:使能 OSCMonIrqEn 中断
	2				
	1	RFLowIrqEn	R/W	0	1：使能 RFLowIrq 中断
	0	RFExIrqEn	R/W	0	1：使能 RFExIrq 中断

注意： 0F_F 中可以使能 0MEN 功能（默认开），0F_O 中 bit3 使能该中断。0F_M 中可以使能 RFLowDetect 功能（默认关）。切记，必须打开对应的功能，然后才能使用该中断。(简单点，就是先开启功能，再使能中断，再将中断标志传到 IRQ 引脚)

m. 0F_P ACRDIrq 自动轮询其他相关中断标志位（自动传到 IRQ 引脚，配置同 0x04，0x05（非 0F 复用） 寄存器中关于 IRQ 的描述）

Addr	Bit	Name	R/W/D	Reset	Description
0x0f		ACRDIrq		00h	
	7	set3	W	0	1：中断位置 1 0：中断位清 0
	6:4	-	-	0	
	3	OSCMonIrq	R/D/W	0	1：OSC 连续四次唤醒失败
	2				
	1	RFLowIrq	R/D/W	0	1：检卡时 RF 值过低
	0	RFExIrq	R/D/W	0	1：检测到外部 RF

标志位，通过写 1 清除，bit7 默认使用 0.

n. IRQ 配置 IRQ 引脚支持上升沿（0 – 1） 下降沿 (1 –0)!

IRQ 引脚在使能中断后，将中断信息输出。与其配置相关的寄存器有 ComIEnReg[7]和 DivIEnReg[7]:

Addr	Bit	Name	R/W/D	Reset	Description
0x02		IRq1EnReg		80h	也即 **ComIEnReg**
	7	IRqInv	R/W	1	设置为 1 表示 IRQ 管脚上的信号与 Status1Reg 中的 IRq 位相反； 设置为 0 则相等。与 IRq2EnReg 中的 IRqPushPull 位一起使用，默认值为 1 时 IRQ 管脚的输出是三态的。

Addr	Bit	Name	R/W/D	Reset	Description
0x03		IRq2EnReg		00h	也即 **DivIEnReg**
	7	IRQPushPull	R/W	0	设置为 1 表示 IRQ 管脚用作标准 CMOS 输出管脚； 设置为 0 表示 IRQ 管脚。

通常（上位机平台使用），为了保证 IRQ 引脚的稳定可靠，推荐 IRQ 引脚使用 MCU 内部上拉输入或者外接上拉电阻。此时应设置 ComIEnReg[7]为 1，DivIEnReg[7]配置无论选用标准 CMOS 输出管脚还是 IRQ 引脚都是高电平，产生中断时有一个由高到低的跳变，清除中断标志，则恢复高电平。如下：

```
I_SI522A_IO_Write(ComIEnReg, 0x80); //IRqInv 为 1
I_SI522A_IO_Write(DivIEnReg, 0xe0);        // IRQPushPull 为 1
```

另外，如果不使用上拉，想使用上升沿触发，此时设置 ComIEnReg[7]为 0，DivIEnReg[7]为 1，IRQ 引脚将输出低电平，产生中断时有一个由低到高的跳变，清除中断标志，则恢复低电平。