

AY-LDC1000_用户手册

杭州艾研信息技术有限公司

2014 年 11 月

申明

杭州艾研信息技术有限公司保留随时对其产品进行修正、改进和完善的权利，同时也保留在不作任何通告的情况下，终止其任何一款产品的供应的权利。用户在下订单前应及时获取相关信息的最新版本，并验证这些信息是当前的和完整的。

可通过如下方式获取最新信息、技术资料和技术支持：

技术支持电话：0571-86134572

技术支持邮箱：support@hpati.com

产品&资料下载中心：<http://www.hpati.com/products/>

互动论坛：<http://www.hpati.com/bbs/forum.php>

公司地址：浙江省杭州市西湖区留和路16号新峰商务楼B306

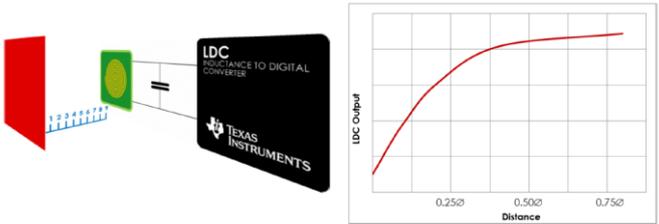
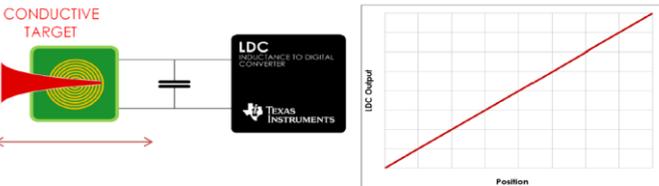
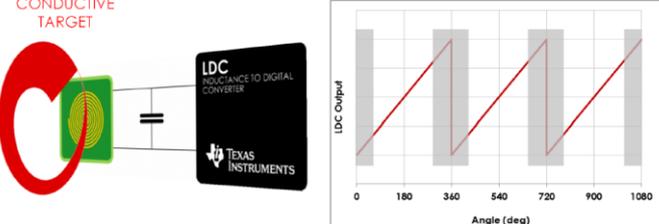
目录

1	LDC1000 简介.....	2
2	单片机连接 LDC1000.....	3
2.1	硬件连接.....	3
2.2	使用 F5529LP 控制 LDC1000.....	4
2.2.1	程序下载.....	4
2.2.2	SPI 通讯.....	6
	SPI 初始化.....	6
	SPI 读写.....	7
2.2.3	SPI 扩展通信模式.....	8
2.2.4	数据处理.....	9
2.3	使用 TivaM4 控制 LDC-1000.....	9
2.3.1	程序下载.....	10
2.3.2	函数说明.....	13
3	寄存器设置及数据处理.....	16
3.1	RpMIN 和 RpMAX 值设定.....	16
3.2	Rp 值计算公式.....	16
3.3	电感计算公式.....	16
3.4	输出数据速率.....	17
4	电路设计注意点.....	17
4.1	滤波电容选择（CFA 和 CFB 管脚管脚间）.....	17
4.2	EVM 板掰开的方式.....	18
5	附录.....	18
5.1	LDC1000 电感检测原理.....	18
5.2	使用 LDC1000 测量并联谐振阻抗和电感.....	20
5.3	参数计算.....	22
5.3.1	Rp 的 Min 和 Max 值计算.....	22
5.3.2	Rp 的 Min 和 Max 值电路意义分析.....	23
6	原理图.....	24

1 LDC1000 简介

LDC1000 是世界首款电感到数字转换器。提供低功耗，小封装，低成本的解决方案。它的 SPI 接口可以很方便连接 MCU。LDC1000 只需要外接一个 PCB 线圈或者自制线圈就可以实现非接触式电感检测。LDC1000 的电感检测并不是指像 Q 表那样测试线圈的电感量，而是可以测试外部金属物体和 LDC1000 相连的测试线圈的空间位置关系。

利用 LDC1000 这个特性配以外部设计的金属物体即可很方便实现：水平或垂直距离检测；角度检测；位移监测；运动检测；振动检测；金属成分检测（合金检测）。可以广泛应用于汽车、消费电子、计算机、工业、通信和医疗领域。

典型应用	说明
 <p>Figure 21. Axial Distance Measurement</p>	<p>纵向距离测量。红色是金属片。</p>
 <p>Figure 22. Linear Position Sensing</p>	<p>横向距离测量。红色是金属片。</p>
 <p>Figure 23. Angular Position Sensing</p>	<p>转动角度测量。红色是金属片。</p>

2 单片机连接 LDC1000

2.1 硬件连接

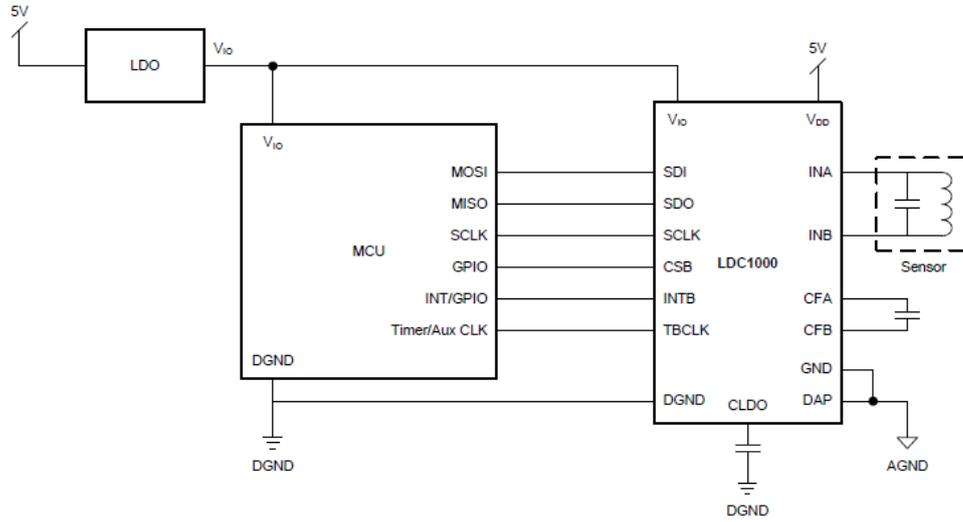


图 1 LDC1000 与 MCU 的连接原理图

LDC-1000 与 MCU 的连接原理图如图 1 所示。采用了四线制 SPI 连接方式，信号线的具体定义如表 1 所示。MCU 通过 SPI 连接（SDI、SDO、SCLK、CSB）实现对 LDC-1000 的控制，以及数据读取。在 SPI 通信过程中，LDC-1000 扮演从机（Slave）的角色。

2.2 使用 F5529LP 控制 LDC1000

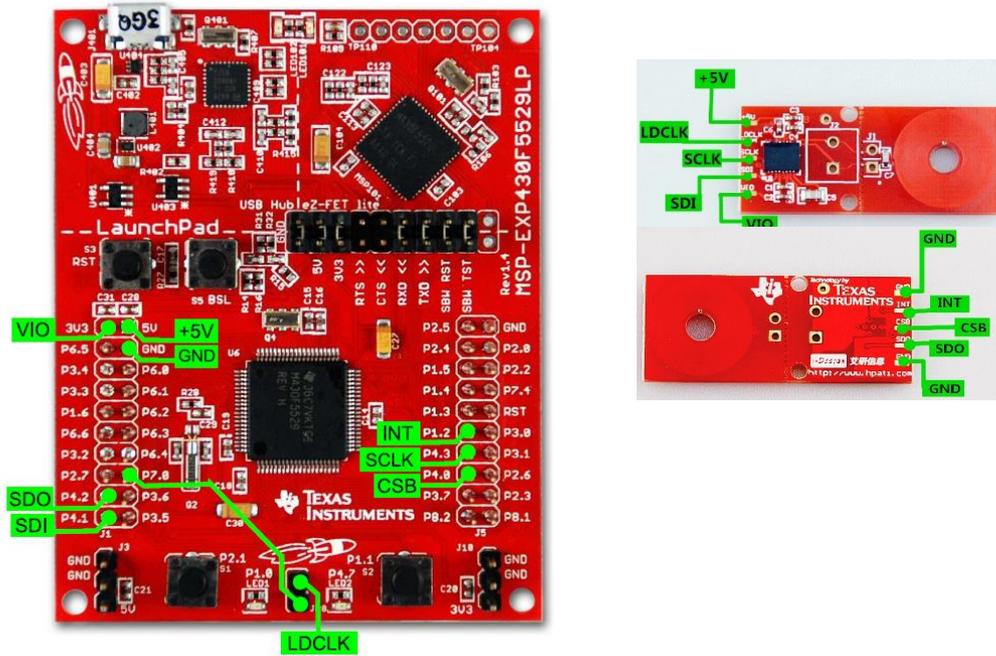


图 2 MSP430F5529 LaunchPad 控制 LDC1000

表 1 与 5529 相连的 数据管脚定义表接口

LDC-1000 接口	F5529LP 接口	图 1 中的 MCU 接口	说明
SDO	P4.2/UCB1SOMI	MISO	SPI 数据输出
SDI	P4.1/UCB1SIMO	MOSI	SPI 数据输入
SCLK	P4.3/UCB1CLK	SCLK	SPI 时钟信号
CSB	P4.0/UCB1STE	GPIO	从设备使能信号
INT	P1.2	INT/GPIO	中断接口
TBCLK	P1.0/ACLK	Timer/Aux CLK	频率计数时钟频率
VIO	3V3		供电接口
+5V	5V		
GND	GND		
NA	P7.0		红色 LED
NA	P1.1		绿色 LED

2.2.1 程序下载

按上述描述进行硬件连接，连接时可先焊接 2*5 的双排排母（EVM 包装袋中自带）到 EVM 板上，再用杜邦线将排母连接到 F5529LP 的相应管脚上，连接完成后，用万用表测试连通性。确保连通性后再通过 USB 线将 F5529LP 与电脑连接。如果是第一次连接 F5529LP 到电脑，将会出现驱动安装的提示，安装驱动（若后续 CCS 下载程序时提示找不到设备，检查驱动是否安装完成）。如正确安装驱动，在 Windows 系统的设备管理器中会看到如图 3 设备管理器标识：

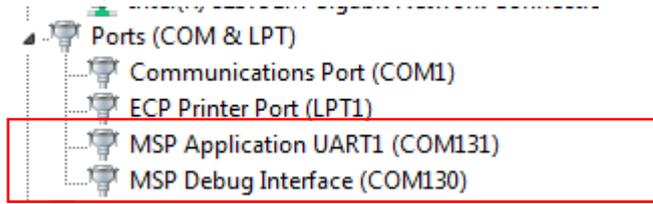


图 3 设备管理器

2、打开 CCS (V5 版本), 单击 “Project->Import existing CCS eclipse project ”, 如图 4 所示。选择实验例程所在的文件夹, 将示例工程导入 (图 5)。注意: 实验例程所在的文件夹的目录不能有中文字符, 因为 CCS 只接受 8 位 ASCII 的文件名及文件路径。

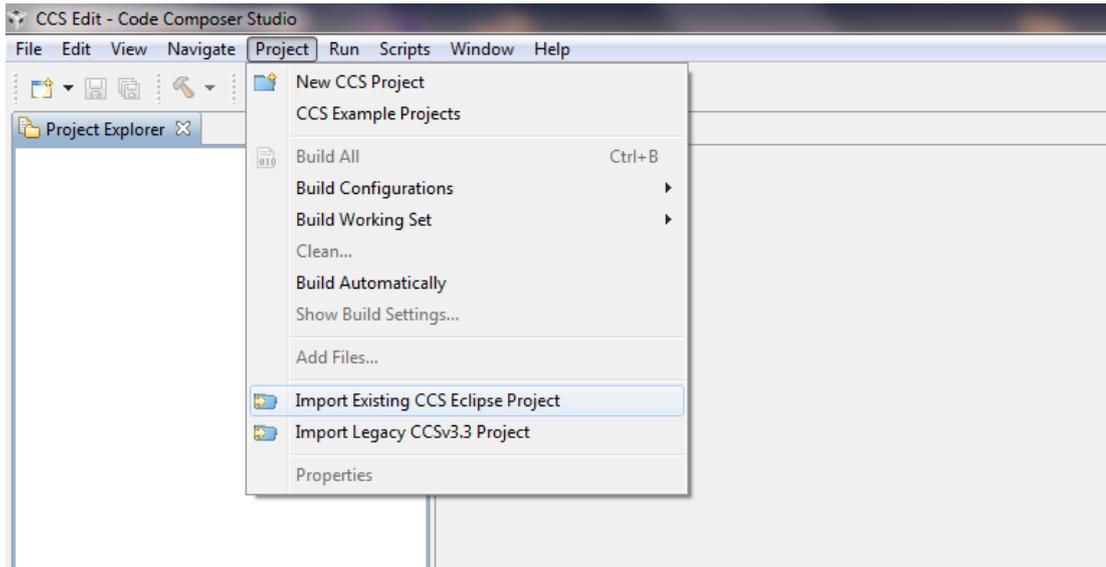


图 4 导入工程的菜单选项

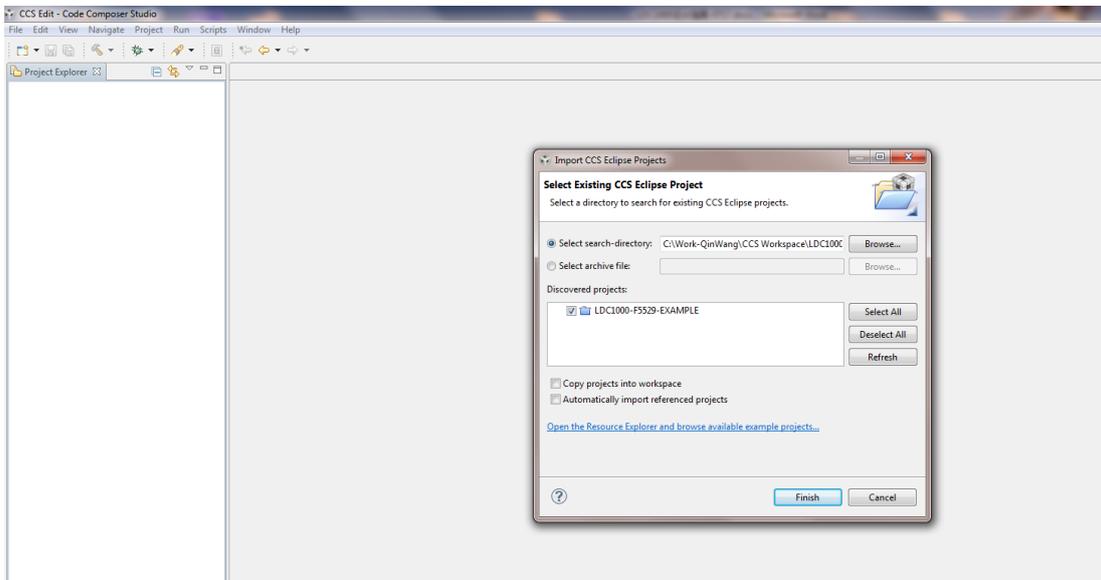


图 5 导入 LDC1000 例程

3、在工程项目上, 右击, 点击 “Build Project” 编译 (图 6)

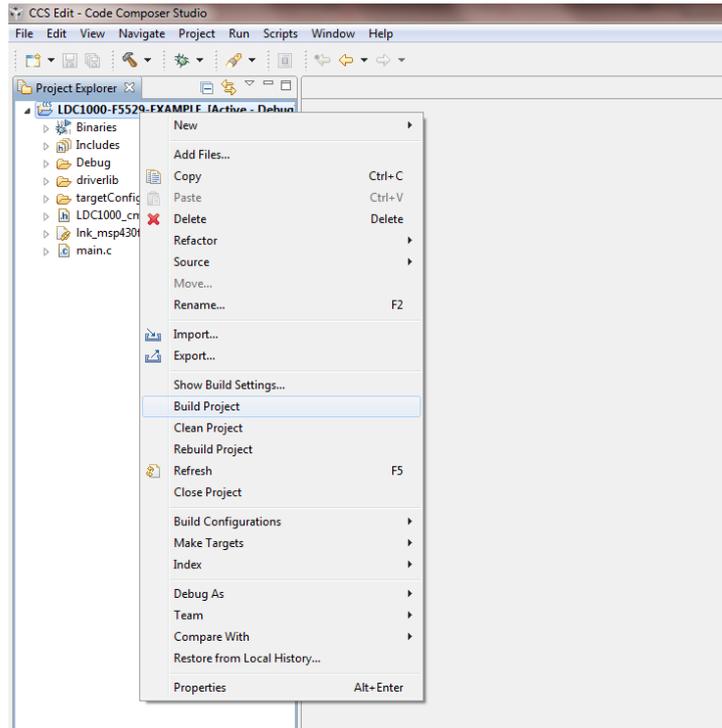


图 6 Build Project

4、单击“Debug”将程序下载至板卡，单击“Run”运行代码(图 7)。

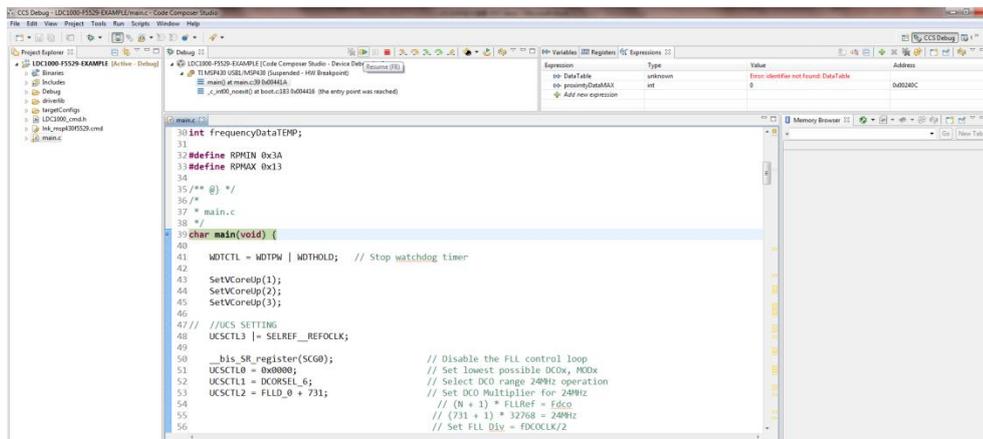


图 7 Debug

2.2.2 SPI 通讯

LDC1000 与 F5529 的通讯接口为 SPI，F5529LP 有多组 SPI 接口，我们选用了其中的 UCBI1 的那一组，参考以下步骤进行配置。如果选用 F5529 中的其它组的 SPI 接口可参考下面步骤进行相似配置。

SPI 初始化

使用 F5529 硬件 SPI 模块，按照所选择模块引脚对 SPI 进行配置，基本步骤如下所示，更多可参考”MSP430x5xx/MSP430x6xx Family User’s Guide”(SLAU208J)-Chapter 33:

```

// initialize SPI
P4DIR |= BIT0; // CS引脚功能和方向选择，为IO口
P4SEL &= ~BIT0;

//SPI SETUP

```

```
P4SEL |=BIT1 + BIT2 + BIT3;           //SOMI,SIMO,CLK引脚功能选择
UCB1CTL1 |=UCSWRST;
UCB1CTL0 |= UCMST+UCMSB+UCSYNC+UCCKPL;
//SPI工作模式配置,3线SPI,8位数据,主机,MSB优先
UCB1CTL1 |= UCSSEL_1;                 //选择SPI模块的时钟
UCB1BR0 = 0x06;
UCB1BR1 = 0;                          //对时钟进行分频
UCB1CTL1 &= ~UCSWRST;
```

SPI 读写

在参考程序中定义了如下所示的几个函数，可直接进行 SPI 的读写。

SPI读，读取指定地址寄存器的值。其中传递参数addr为从机读数据的地址，8位；data为读取到的寄存器值保存的地址指针；len为读取的字节数/次数

```
char spi_readByte( char addr, char * data); //单字节读取
char spi_readWord(char addr, unsigned int * data); //双字节读取
char spi_readBytes( char addr, char * buffer, char len); //多字节读取
```

SPI写，向指定寄存器写入控制值。其中传递参数addr为从机写入数据的地址，8位；data为写入到的寄存器值保存的地址指针；len为写入的字节数/次数

```
char spi_writeByte(char addr, char data); //单字节写入
char spi_writeWord(char addr, unsigned int data); //双字节写入
char spi_writeBytes( char addr, char * buffer, char len); //多字节写入
```

对上述函数进行调用时，函数参数值的选定应根据 LDC1000 的 SPI 通信协议（即 SPI 通信的时序，时序图如图 8 所示）。在主机(MSP430)与从机(LDC1000)通讯的时候，遵循以下步骤：

1. 片选信号置零；
2. MSP430 通过 SDI 线向 LDC1000 写入访问寄存器地址，其中最高位 0 表示写入，1 表示读出，剩余 7 位为寄存器的地址；
3. 步骤 2 过程占据 8 个时钟周期，该时间内 SDO 线处于高阻状态；
4. 如果命令为读，即步骤 1 中传输的数据最高位为 1，SDO 线上发送来自其地址寄存器的 8 位字节；
5. 如果命令为写，SDI 线接收来自 MSP430 的 8 位字节数据写入相应的寄存器中；
6. 片选信号置高，释放对该从机的控制。

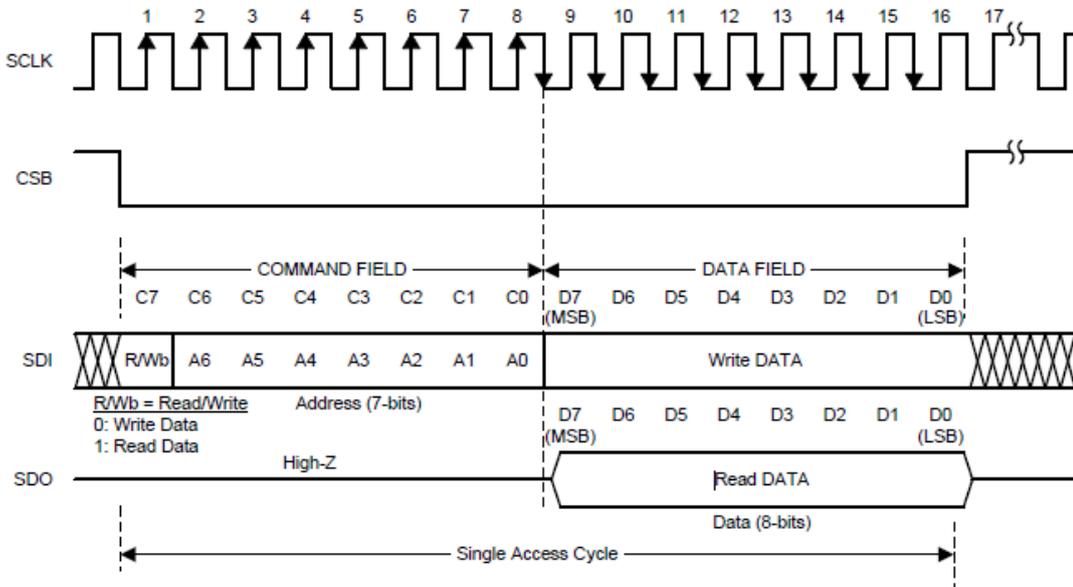


图 8 LDC1000 SPI 读写时序

片选信号 CSB 的置位意味着新的寄存器访问。数据输出在时钟的下降沿发生，数据的写入在时钟的上升沿发生。需要注意，片选信号的复位必须保证在数据读取/写入完全完成后，即第 16 个时钟的上升沿，否则该次数据的读写无效。

LDC1000_cmd.h 内对 LDC1000 的寄存器地址及对应的位进行了定义。在编写程序时可以进行参考。

```
#define LDC1000_CMD_REVID      0x00
#define LDC1000_CMD_RPMAX     0x01
#define LDC1000_CMD_RPMIN     0x02
#define LDC1000_CMD_SENSORFREQ 0x03
#define LDC1000_CMD_LDCCONFIG 0x04
#define LDC1000_CMD_CLKCONFIG 0x05
#define LDC1000_CMD_THRESHILSB 0x06
#define LDC1000_CMD_THRESHIMSB 0x07
#define LDC1000_CMD_THRESLOLSB 0x08
#define LDC1000_CMD_THRESLOMSB 0x09
#define LDC1000_CMD_INTCONFIG 0x0A
#define LDC1000_CMD_PWRCONFIG 0x0B
#define LDC1000_CMD_STATUS     0x20
#define LDC1000_CMD_PROXLSB    0x21
#define LDC1000_CMD_PROXMSB    0x22
#define LDC1000_CMD_FREQCTRLSB 0x23
#define LDC1000_CMD_FREQCTRMID 0x24
#define LDC1000_CMD_FREQCTRMSB 0x25
```

2.2.3 SPI 扩展通信模式

LDC1000 支持 SPI 的扩展通信模式，可实现对多个寄存器连续访问。具体方法为保持片选信号有效，进行连续的读/写，此时寄存器的地址会自动增加。在 $8*(1+N)$ 个时钟周期后将片选信号复位即可。例程中的多字节写入/读取即实现了这样的功能。

```
spi_readBytes(LDC1000_CMD_FREQCTRLSB,&frequencyData[0],3);
```

2.2.4 数据处理

```

//read all REG value using default setting
char orgVal[20];

//write to register
spi_writeByte(LDC1000_CMD_RPMAX,      RPMAX);
spi_writeByte(LDC1000_CMD_RPMIN,      RPMIN);
spi_writeByte(LDC1000_CMD_SENSORFREQ, 0x94);
spi_writeByte(LDC1000_CMD_LDCCONFIG,  0x17);
spi_writeByte(LDC1000_CMD_CLKCONFIG,   0x02);
spi_writeByte(LDC1000_CMD_INTCONFIG,   0x02);

spi_writeByte(LDC1000_CMD_THRESHILSB,  0x50);
spi_writeByte(LDC1000_CMD_THRESHIMSB,  0x14);
spi_writeByte(LDC1000_CMD_THRESLOLSB,  0xC0);
spi_writeByte(LDC1000_CMD_THRESLOMSB,  0x12);

spi_writeByte(LDC1000_CMD_PWRCONFIG,    0x01);

//read all registers using extended SPI

spi_readBytes(LDC1000_CMD_REVID, &orgVal[0],12);
    
```

在例程中对 LDC1000 的读写如上图所示。首先依次按照设定对 LDC1000 的寄存器写入对应的控制字(可参考 LDC1000datasheet 中关于寄存器的描述)。利用例程提供的 spi_readBytes 函数可以一次性将所有寄存器内的值全部读出。

其中用户关心两个值, Rp 和 Frequency, 通过前者我们可以推算出金属的距离, 而后者可以计算得到电感值。

```

spi_readBytes(LDC1000_CMD_PROXLSB,&proximtyData[0],2);
spi_readBytes(LDC1000_CMD_FREQCTRLSB,&frequencyData[0],3);

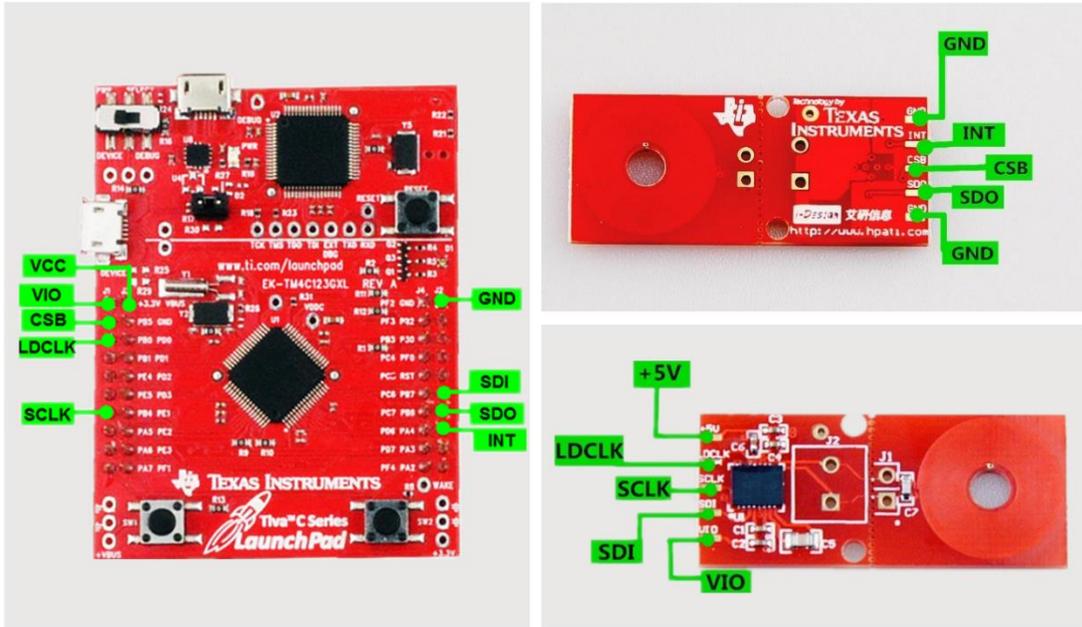
proximtyDataMAX = ((unsigned char) proximtyData[1]<<8) +
                  proximtyData
[0];
frequencyDataMAX = ((unsigned char) frequencyData[1]<<8) +
                   frequencyData[0];
    
```

而 Rp 和 frequency 分别占用了 2 个和 3 个寄存器, 也就是说 Rp 为 16 位, Frequency 为 24 位。用户可以参考例程的方法分别读取完整的等效电阻和频率值。注意, 例程中仅读取了频率的低 16 位。

具体 Rp 与电感的计算方法及公式参考本文档 3 [寄存器设置及数据处理](#) 章节。

2.3 使用 TivaM4 控制 LDC-1000

TIVA 控制 LDC1000 的硬件连接如下图所示。



LDC-1000 接口	TIVA LP 接口	说明
SCLK	PB4/SPICLK	SPI 时钟信号
CSB	PB5/SPICS	从设备使能信号
SDI	PB7/SPIMOSI	SPI 数据输入
SDO	PB6/SPIMISO	SPI 数据输出
INT	PA4	中断接口
LDCLK	PB0	频率计数时钟频率
VIO	3V3	供电接口
+5V	VBUS	
GND	GND	

2.3.1 程序下载

按上述方法连接 M4Launchpad 与 LDC1000，将 Launchpad 上的 power select switch 切到右侧。通过 USB 连接电脑。打开 CCS，单击“Project->Import existing CCS eclipse project”，选择实验例程所在的文件夹，将示例工程导入。注意：路径中不能有中文字符。

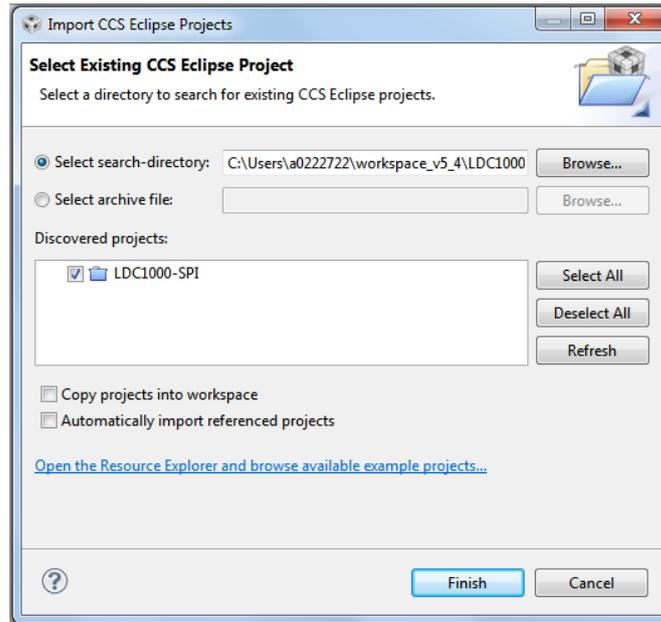


图 9 导入工程

单击“build”编译，完成后单击“Debug”进行程序下载。

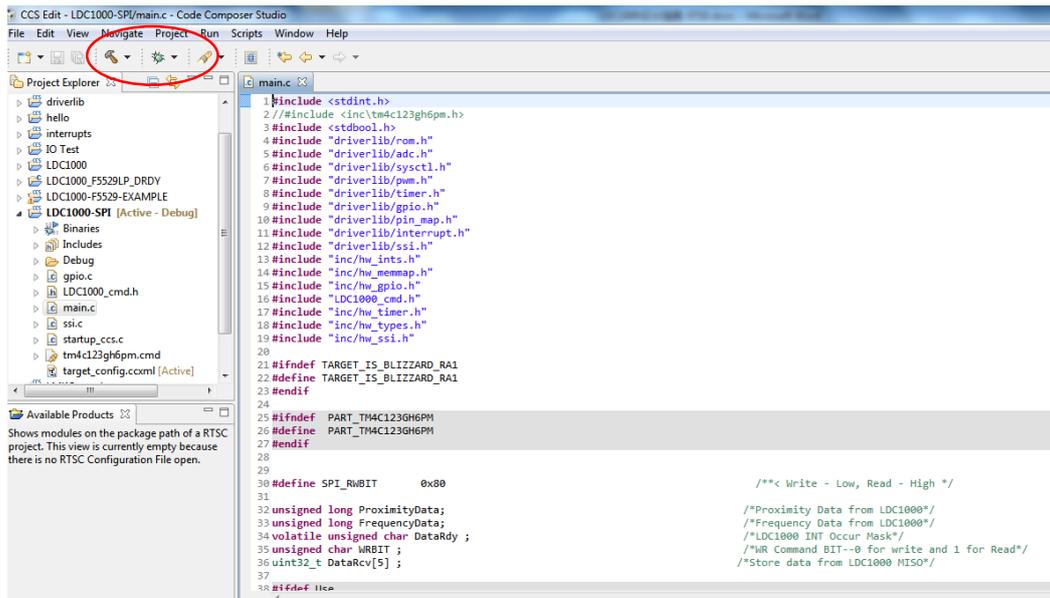


图 10 编译工程

注意：在该例程中使用了 TivaWare 软件，所以在使用前先确保电脑上已安装 TivaWare。如果编译无法通过，请检查以下步骤：

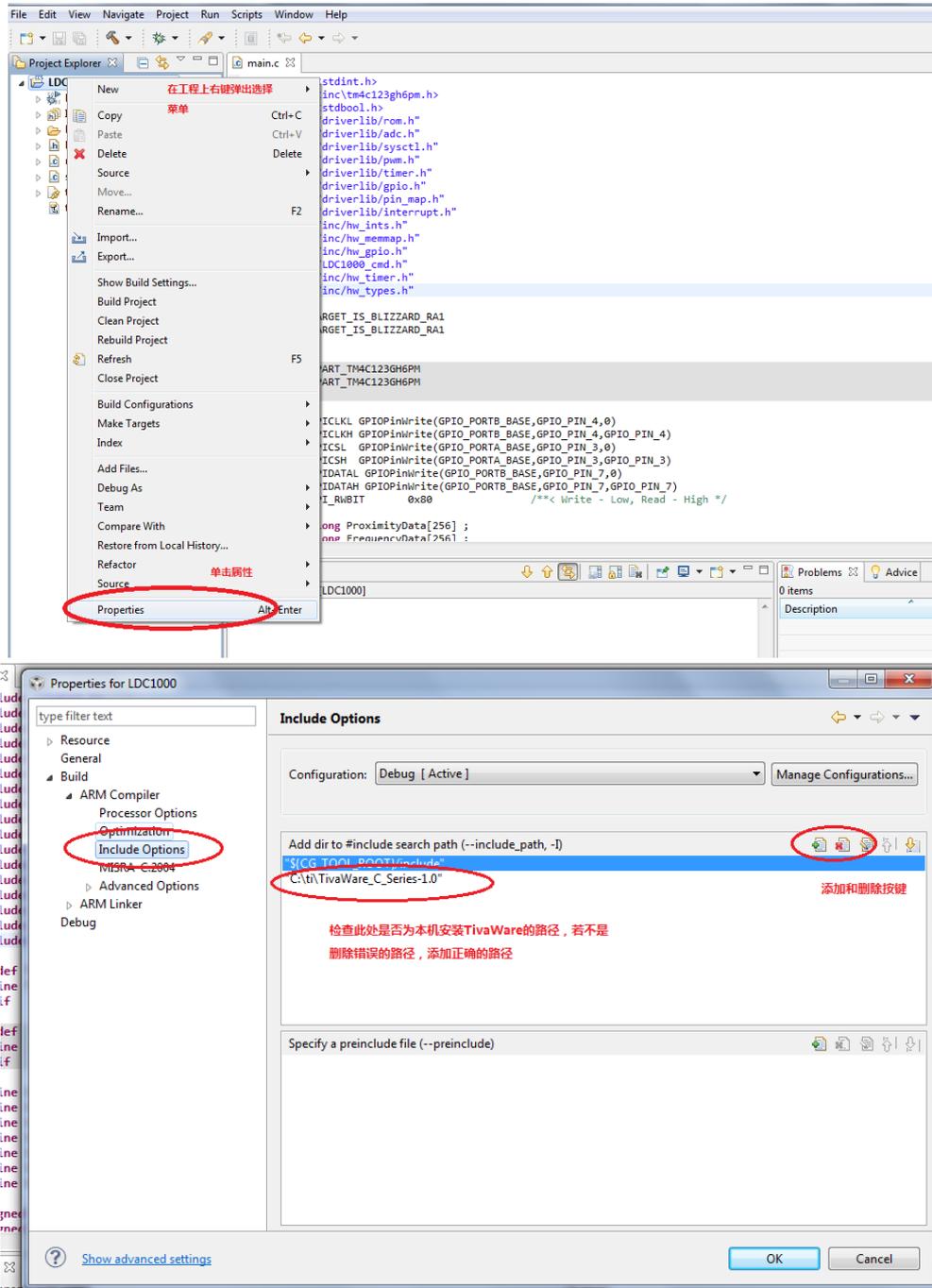


图 11 包含路径设置

同上，进行 link 路径库文件路径的检查

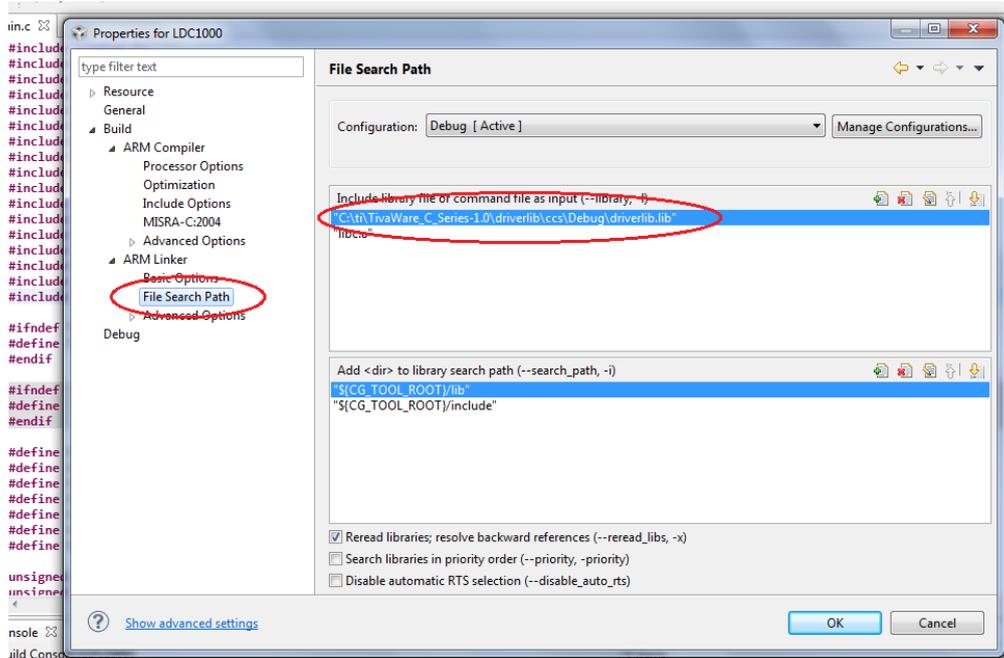


图 12 链接路径设置

2.3.2 函数说明

LDC1000 与 TIVA 的通讯接口为 SPI，在前文的连接方式下（UCB1），参考以下步骤进行配置。选用 TIVA 中 SPI 接口进行数据处理。SPI 初始化过程：

```

/*****
 * @brief:      SPI 通信初始化
 * @param:      none
 * @return:     none
 *
 * _____|PB4 (SSI2CLK)  ---->      SCLK |
 *            |PB5 (SSI2FSS)  ---->      CSB  |
 *            |PB6 (SSI2RX)   <----      SDO  |
 * Tiva M4    |PB7 (SSI2TX)   ---->      SDI  |  LDC1000
 *            |              |
 * _____|              |_____
 *****/
/

void SPIInit()
{
    //配置 PB6 为 SSI2RX，即对 Tiva M4 而言的 SPI 数据接收线
    GPIOPinTypeSSI (GPIO_PORTB_BASE,GPIO_PIN_6) ;
    GPIOPinConfigure (GPIO_PB6_SSI2RX) ;

    //配置 PB6 为 SSI2TX，即对 Tiva M4 而言的 SPI 数据发送线
    GPIOPinTypeSSI (GPIO_PORTB_BASE,GPIO_PIN_7) ;
    GPIOPinConfigure (GPIO_PB7_SSI2TX) ;

    //配置 PB4 为 SSI2CLK 线，作为时钟线
    GPIOPinTypeSSI (GPIO_PORTB_BASE,GPIO_PIN_4) ;
    GPIOPinConfigure (GPIO_PB4_SSI2CLK) ;

    //配置 PB5 为 SSI2FFS 线，作为片选线
    GPIOPinTypeSSI (GPIO_PORTB_BASE,GPIO_PIN_5) ;
    GPIOPinConfigure (GPIO_PB5_SSI2FSS) ;
}

```

```

SSIDisable (SSI2_BASE); //禁能 SSI2

//配置 SSI2 为 SSI_FRF_MOTO_MODE_0 协议格式, SPI 主模式, 时钟源为 5K, 16 位数据长度

SSISetExpClk (SSI2_BASE, SysCtlClockGet (), SSI_FRF_MOTO_MODE_0, SSI_MODE_MASTER, 5000, 16);

SSIEnable (SSI2_BASE); //使能 SSI2
}

```

SPI 的读写时序同上 MSP430F5529, LDC-SPI 协议。

通信时遵循以下步骤:

1. 片选信号置低;
2. TIVA 通过 SDI 线向 LDC1000 写入访问寄存器地址, 其中最高位 0 表示写入, 1 表示读出, 剩余 7 位为寄存器的地址;
3. 如果命令为读, 即步骤 1 中传输的数据最高位为 1, SDO 线上发送来自其地址寄存器的 8 位字节; 如果命令为写, SDI 线接收来自 TIVA 的 8 位字节数据写入相应的寄存器中。

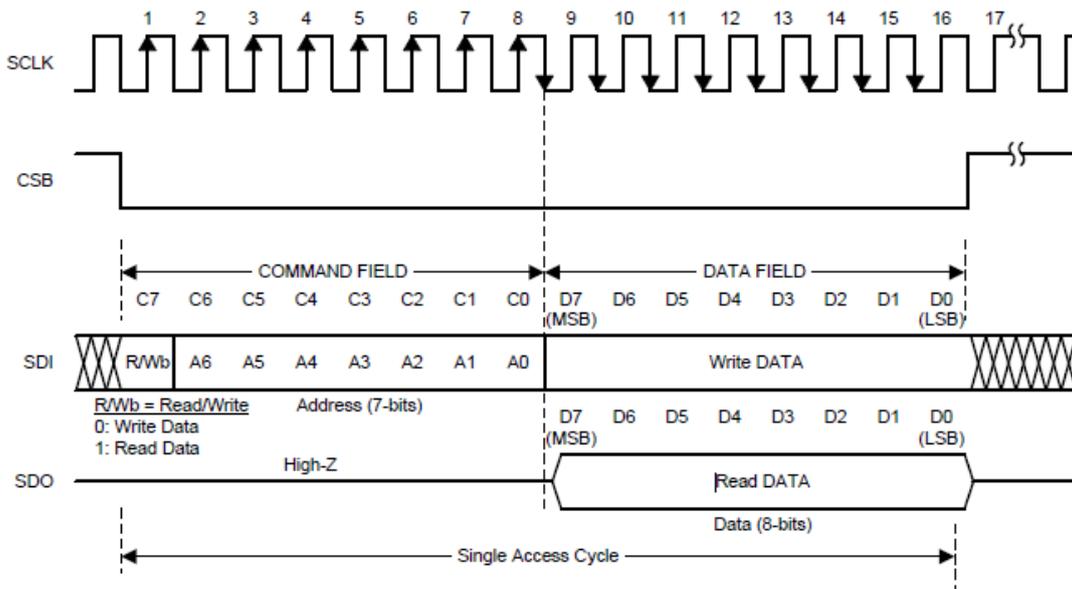


图 13 时序示意

LDC1000 写寄存器操作

```

/*****
 * @brief:      SPI 写数据
 * @param:      unsigned int, SPIdata:待写的的数据
 * @return:     none
 *****/

/
void SPIDataSend(unsigned int SPIdata)
{

```

```

SSIDataPut (SSI2_BASE,SPIData);           //SPI 发送（写）数据
while (SSIBusy (SSI2_BASE)) ;           //等待 SPI 发送（写）完成
}

```

LDC1000 数据读操作

```

/*****
* @brief:    SPI 读取数据
* @param:    uint8_t,ui8Add:寄存器地址
*            uint32_t,*p_ui32Data:读取值存放缓存
* @return:   none
*****/
void SPIDataRead(uint8_t ui8Add,uint32_t *p_ui32Data)
{
    SPIDataSend((ui8Add | SPI_RWBIT)<<8); //写入将要读取的寄存器地址
    SSIDataGet (SSI2_BASE,p_ui32Data);   //读取上述寄存器中的值
    while (SSIBusy (SSI2_BASE));        //等待读取完成
}

```

LDC1000 需要外部加时钟方可工作，采用 TIVA 的定时器输出作为时钟。定时器的配置如下。

```

/*****
* @brief:    Timer 初始化
* @param:    none
* @return:   none
*
* _____|_____ |_____
* Tiva M4    |PB0 (Timer CLK) ---->  TBCLK|  LDC1000
* _____|_____ |_____
*****/
/
void TimerInit ()
{
    TimerDisable (TIMER2_BASE,TIMER_A); //禁能 timer
    GPIOPinTypeTimer (GPIO_PORTB_BASE,GPIO_PIN_0);
    GPIOPinConfigure (GPIO_PB0_T2CCP0); //配置 PB0 为 CCP 模式
    HWREG (TIMER2_BASE + TIMER_O_CFG) = 0x04; //选择 16-bit timer

    //配置 TimerA 周期计数 (Periodic Timer mode)
    HWREG (TIMER2_BASE +TIMER_O_TAMR) |=
        (TIMER_TAMR_TAAMS|TIMER_TAMR_TAMR_PERIOD) ;

    //加载 Timer 计数值:40, 并且设置 Match 值:20 (Timer 默认为减计数)
    HWREG (TIMER2_BASE + TIMER_O_TAMATCHR) = 20;
    TimerLoadSet (TIMER2_BASE,TIMER_A,40);

    TimerEnable (TIMER2_BASE,TIMER_A); //使能 Timer
}

```

3 寄存器设置及数据处理

3.1 RpMIN 和 RpMAX 值设定

如前文所讲，为保证 Rp 的实际值落在采样区间内，同时又保证足够的精度，需要用户合理地设置 RpMAX 和 RpMIN 寄存器的值。可以通过实际测量的方法在两个极限条件下测出 Rp 等效的最大值和最小值。在测试仪器有限的情况下，可以简单地通过软件算法比较得到两个范围的限定值。

1. 首先通过表格选取两个合适的 RpMAX 和 RpMIN 的值写入寄存器中；
2. 将金属物体放在距离线圈最近的位置（最近位置是指用户设备结构设计的最近位置），此时涡流损耗最大，将 Rp_Min 的值逐渐增大当 code 值接近 25000 时选择此时的 Rp_Min。（选择 25000 是为了给 32768 最大值留有余量）；
3. 将金属物体放在距离线圈最远的位置（最远位置是指用户设备结构设计的最远位置），此时涡流损耗最小，将 Rp_Max 的值逐渐减小，code 值接近 3000 时选择此时的 Rp_Max。（继续减小 Rp_Max 可以看到 code 被钳位到 0 值）。

3.2 Rp 值计算公式

计算公式如下：

$$R_p = (R_{pMAX} \times R_{pMIN}) / (R_{pMIN} \times (1-Y) + R_{pMAX} \times Y), \text{ in } \Omega.$$

Where:

- $Y = \text{Proximity Data} / 2^{15}$
- Proximity data is the LDC output, register address 0x21 and 0x22.

例如，0x21 和 0x22 寄存器（两个寄存器组成 16bit 数据）的读值为 5000.

Proximity Data LSB	0x21	RO		Proximity Data[7:0]
Proximity Data MSB	0x22	RO		Proximity Data[15:8]

对照 RpMAX 和 RpMIN 的表格可以得到两者的具体电阻值，假设 **Rp_MIN is 2.394 kΩ, and Rp_MAX is 38.785 kΩ**

代入公式中得到：

$$Y = 5000 / 2^{15} = 0.1526$$

$$R_p = (38785 \times 2394) / (2394 \times (1 - 0.1526) + 38785 \times 0.1526) = (92851290) / (2028.675 + 5918.591)$$

$$R_p = 11.683 \text{ k}\Omega$$

3.3 电感计算公式

LDC1000 测量电感频率是用测试 LC 谐振频率的方法。LDC1000 有外部的基准时钟，也是使用计数方法来做频率计。

$$\text{Sensor frequency, } f_{\text{sensor}} = (1/3) \times (F_{\text{ext}} / F_{\text{count}}) \times (\text{Response Time})$$

f_{sensor} 是 LC 谐振频率，F_{ext} 是外部基准时钟频率，F_{count} 是 LDC1000 内部计数器值

Frequency Counter Data LSB	0x23	RO		ODR LSB
Frequency Counter Data Mid-Byte	0x24	RO		ODR Mid Byte
Frequency Counter Data MSB	0x25	RO		ODR MSB

Response time 是寄存器设定的一个值 (0x04)

LDC Configuration	0x04	R/W	0x1B	Reserved(000)	Amplitude	Response Time
-------------------	------	-----	------	---------------	-----------	---------------

将上面公式左右分别求倒数:

$$1/f_{\text{sensor}} = 3 * F_{\text{count}} / F_{\text{ext}} * (1/\text{Response time})$$

将 response time 移动到等式左边:

$$\text{Response time} * 1/f_{\text{sensor}} = 3 * F_{\text{count}} * 1/F_{\text{ext}}$$

这样看就很清晰了, $1/f_{\text{sensor}}$ 是 LC 谐振周期, $1/F_{\text{ext}}$ 是基准时钟周期, 也就是说在 response time 个 LC 谐振周期内, 使用 LDC1000 的 Fcount 计数器记录基准时钟的个数用来推算 LC 的谐振频率。

根据 $f_{\text{sensor}} = 1/\sqrt{2 * \pi LC}$ 计算出 L。(C 是电路设计中的已知量)

$$L = 1/[C * (2 * \pi * f_{\text{sensor}})^2],$$

3.4 输出数据速率

输出数据速率跟 LC 的谐振频率有关, 手册中给出的计算方法如下:

$$\text{Output data rate} = (f_{\text{sensor}}) / (\text{Response Time} / 3), \text{ Sample per second (SPS)}$$

回顾一下电感测量里的分析, 我们可以更清楚的看出数据速率的物理意义, 电感测量中给出的谐振频率公式如下:

$$f_{\text{sensor}} = (1/3) * (F_{\text{ext}} / F_{\text{count}}) * (\text{Response Time})$$

将 1/3 和 response time 移动到左边

$$F_{\text{sensor}} / (\text{Response time} / 3) = F_{\text{ext}} / F_{\text{count}}$$

及 output data rate = $F_{\text{ext}} / F_{\text{count}}$

所以输出数据速率就是外部基准时钟的 Fcount 分频。

4 电路设计注意点

4.1 滤波电容选择 (CFA 和 CFB 管脚管脚间)

LDC1000 对滤波电容的选择十分苛刻, 需要选择低泄露, 温度稳定性好, 压电噪声低的电容。最优的电容值在 20pF 到 100nF 之间。电容值由 LC 的谐振时常数决定。如果使用陶瓷电容的话, class I 类的陶瓷电容可以使用, 这类电容有着很好的温度特性。电容耐压值要大于 10V。为了减小寄生参数, 滤波电容应该尽量靠近芯片, 走线要短。

这个滤波电容是 LDC1000 内部有源滤波器的一部分, 选取时应尽量小, 但是要确保有源滤波器不饱和。时常数越大, 电容值越大。也就是电感越大, 滤波电容值越大。

以下步骤可以较为方便的找到最优滤波电容。

- 1、先焊接一个电容, 带磁芯的电感一般使用 10nF, 空心电感一般使用 100pF。
- 2、上电后, 配置好 LDC1000, 将金属物远离电感线圈。
- 3、用示波器观察 CFB 管脚的波形。由于无源探头的输入电容较大, 所以推荐使用有源探头, 或者无源探头前加 1K 电阻串联。
- 4、改变滤波电容值, 直到示波器出现 1V 峰峰值的信号。信号大小跟滤波电容的倒数成正比, 所以例如 100pF 时测试信号的峰峰值为 200mV, 为了达到 1V 需要使用 $0.2V/1V * 100pF = 20pF$ 的电容。

4.2 EVM 板掰开的方式

采用非板载 PCB 线圈的时候可以掰开 AY-LDC1000 EVM，注意：因为 PCB 板较厚，直接沿邮票孔掰开时需要较大力，且容易弄坏板，建议用刻刀或者美工到沿邮票孔上先划上一定深度的槽，再掰开。邮票孔位置如图 14 所示

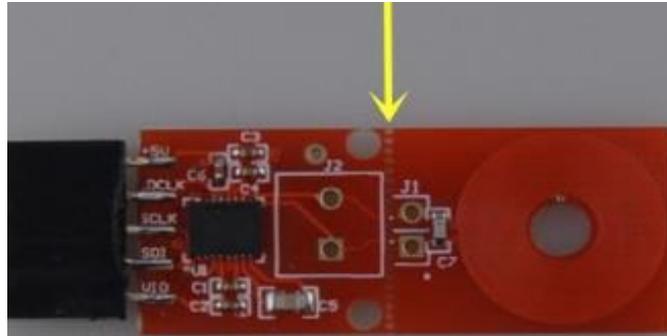


图 14 邮票孔位置

将非板载的线圈的引线连接到已经掰开的 EVM 板（掰开后焊接图 15 所示的接线柱）上即可使用。

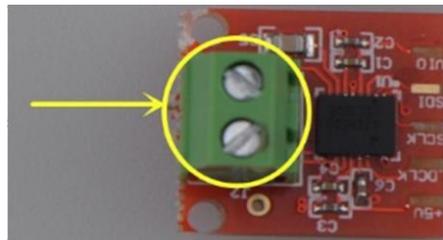


图 15 自制线圈的使用

5 附录

5.1 LDC1000 电感检测原理

LDC1000 的电感检测原理是利用大学物理中讲到的电磁感应原理。在 PCB 线圈或者自制线圈（如图 16 左边）中加上一个交变电流，线圈周围就会产生交变电磁场，这时如果有金属物体（如图 16 右边）进入这个电磁场则会在金属物体表面产生涡流（感应电流）。涡流电流跟线圈电流方向相反，涡流产生的感应电磁场跟线圈的电磁场方向相反。涡流是金属物体的距离，大小，成分的函数。

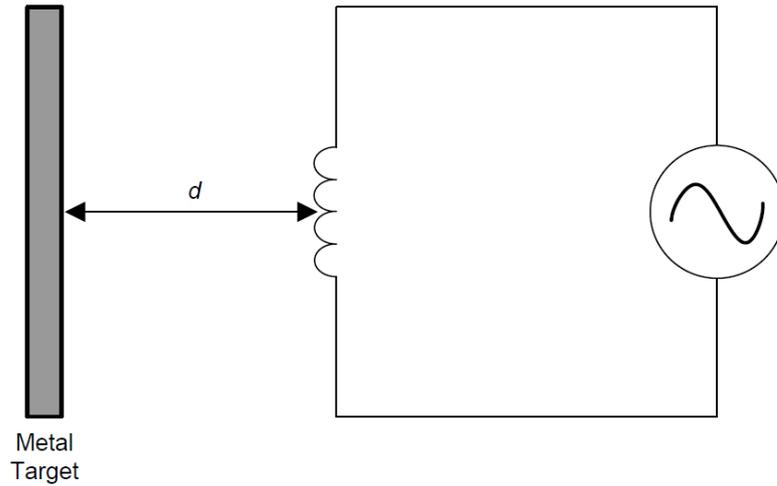


图 16 电感的感应

涡流产生的反向磁场跟线圈耦合在一起，就像是有另一个次级线圈存在一样。这样 LDC1000 的线圈作为初级线圈，涡流效应作为次级线圈，就形成了一个变压器。如图 17 所示，由于变压器的互感作用，在初级线圈这一侧就可以检测到次级线圈的参数。

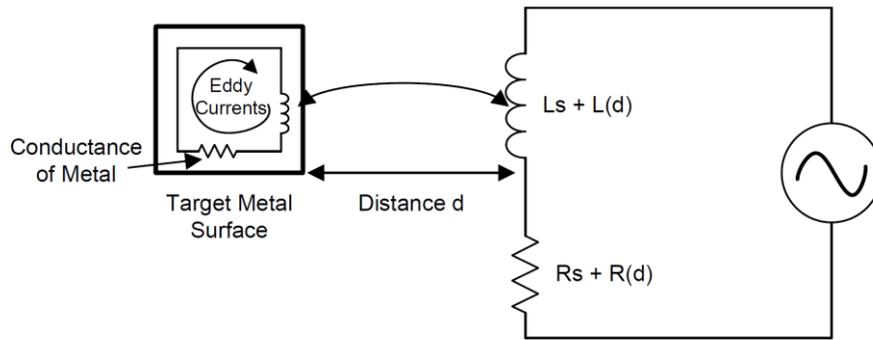


图 17 互感

上图中 L_s 是初级线圈电感值， R_s 是初级线圈的寄生电阻。 $L(d)$ 是互感值， $R(d)$ 是互感的寄生电阻，括号中用 d 是因为它们是距离的函数。

交变电流如果只加在电感上（初级线圈），则在产生交变磁场的同时也会消耗大量的能量。这时将一个电容并联在电感上，由于 LC 的并联谐振作用能量损耗大大减小，只会损耗在 R_s 和 $R(d)$ 上。如下图所示。可以看出检测到 $R(d)$ 的损耗就可以间接的检测到 d 。

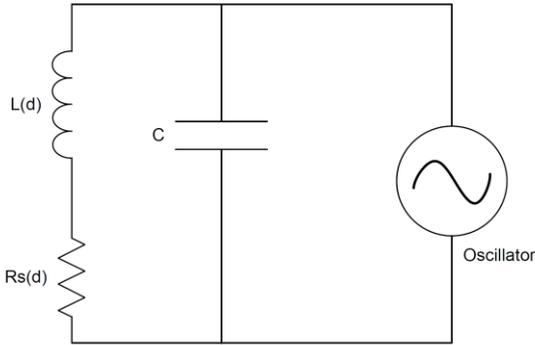
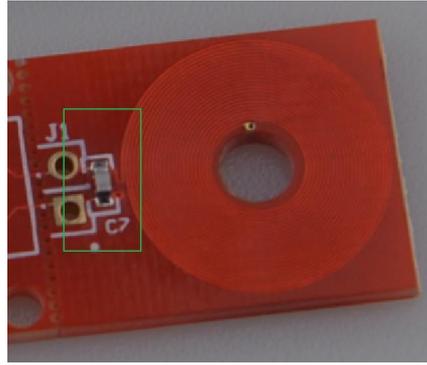


Figure 9. LC Tank Connected to Oscillator



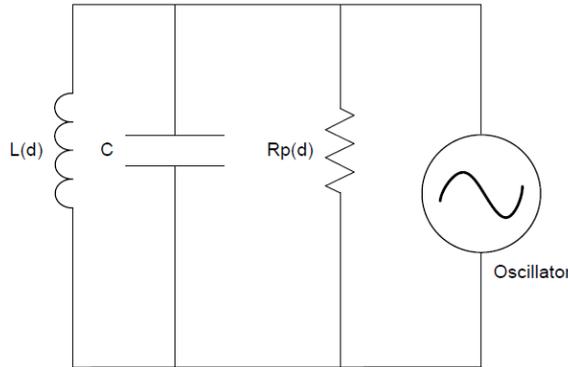
LDC1000 并不是直接检测串联的电阻，而是检测等效并联电阻。等效并联模型如下图所示，根据教科书可以得到等效并联电阻的计算公式，这里不再推导过程。

$$R_p(d) = (1 / [R_s + R(d)]) * ([L_s + L(d)] * C)$$

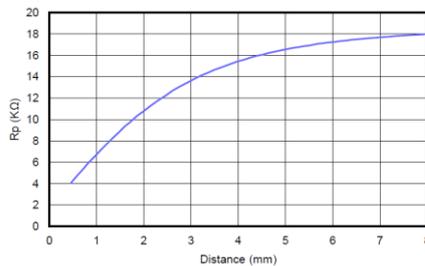
等效并联电阻

$$R_p = (1 / R_s) * (L / C)$$

外部没有金属物体时，也就是去掉跟 d 相关变量



做一个简单的测量，看一下 $R_p(d)$ 和 d 的关系。用 2mm 厚的不锈钢金属片作为待测物体，LDC1000 的线圈使用 PCB 线圈，14mm 直径，23 圈，4mil 线宽，4mil 间距，1 盎司铜皮。变化 d 进行测量，可以得到如下图所示的变化规律。



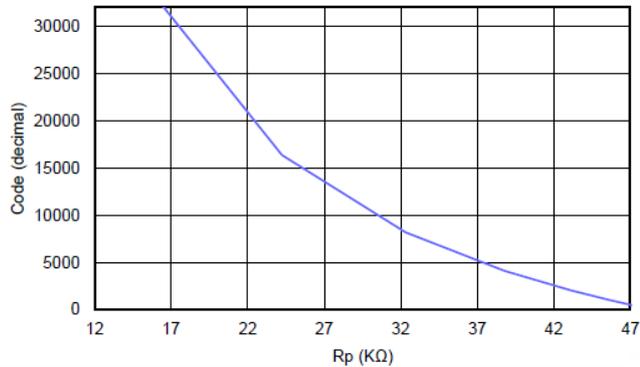
5.2 使用 LDC1000 测量并联谐振阻抗和电感

LDC1000 可以同时测量阻抗和谐振频率。LDC1000 是通过调节振荡器的幅度同时检测 LC 的谐振损耗来实现这个测量的。通过检测注入到 LC 谐振单元的能量，可以计算出 R_p 。在 LDC1000 中 R_p 值被转化为数字量，数值跟 R_p 的值成反比。LDC1000 可以检测到 LC 的谐振频率，谐振频率用于计算 LC 中的 L 值。频率值也由 LDC1000 转换为数字量。

LDC1000 支持很宽范围的 LC 组合，也就是说支持 5KHz 到 5MHz 的谐振频率， R_p （等

效并联电阻)的范围支持 798 欧到 3.93M 欧。 R_p 的范围可以看作是 LDC1000 内部 ADC 的信号输入范围。通过编程可以改变这个范围,注意这个范围影响 ADC 的分辨率。注:这跟用电阻分压调节 ADC 的量程一样,分压比越大 ADC 量程越宽,但是对小信号的精度越差。

下图是 R_p 对应的码值图。



回顾 R_p 的计算公式,可以看到 R_p 跟 R_s 成反比, R_p 又跟 ADC 码值成反比,所以 ADC 码值跟 R_s 成正比。 R_s 跟 LC 谐振损耗成正比,所以损耗越大,ADC 输出码值越大。上图最左边当 R_p 小于最小设定值时,达到最大码值。这种情况是当金属物体离线圈最近的时候发生,此时涡流最大,损耗也就最大了。

下面看一下实测结果,帮助理解 LDC1000 检测 R_p 和 L 的过程。测试点在线圈的两级,如下图红色框。使用单端探头时要用双路测试并在示波器内求差,使用差分探头则可以直接看到波形。

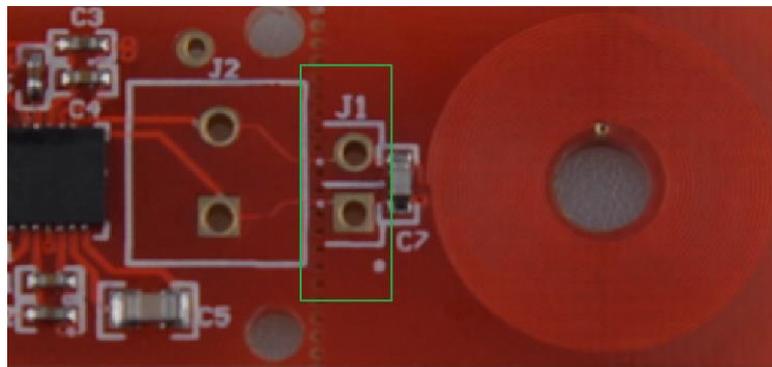
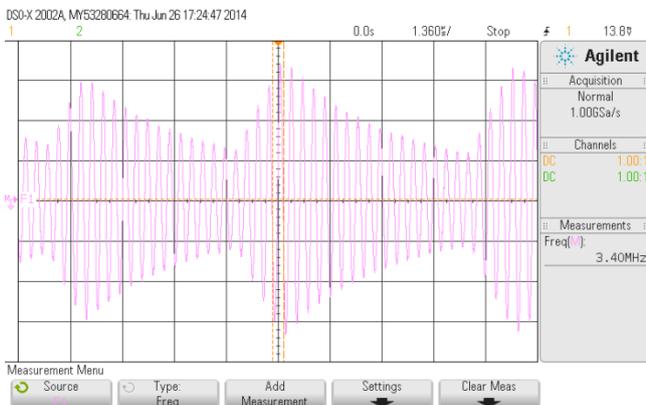
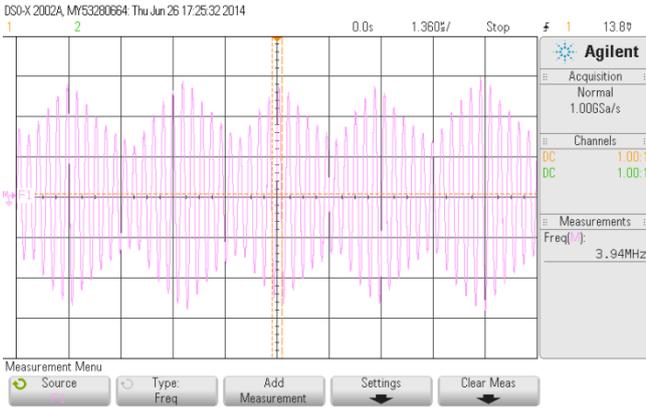


图 18 测试点





金属物远离线圈

金属物靠近线圈

可以看出实测波形是幅度有变化而正弦波，正弦波的频率是谐振频率（左图 3.4MHz，右图 3.94MHz）。正弦波的最高点是 LDC1000 向 LC 中注入能量的起点，然后 LC 开始衰减振荡，再注入能量，如此往复。金属物远离线圈时，由于没有涡流的反向磁场，线圈的电感最大，谐振频率最低。当金属物靠近时，由于涡流产生的反向磁场，使线圈的等效电感下降，谐振频率就会提高，例如图中从 3.4MHz 提高到 3.94MHz。

谐振 LC 中的 C 是已知的（电路板上焊接），所以根据谐振频率就能计算出 L 值。根据衰减振荡的曲线可以计算出 R_p 。

5.3 参数计算

5.3.1 R_p 的 Min 和 Max 值计算

不同的测试对象和距离会产生不同的损耗，也就是 R_p 的范围不同。所以应用中需要配置合适的 R_p 范围。LDC1000 中有两个寄存器 R_p_Min 和 R_p_Max 用于配置。

R_p_MAX	0x01	R/W	0x0E	R_p Maximum
R_p_MIN	0x02	R/W	0x14	R_p Minimum

应用中 R_p 的值如果超出这个范围就会被钳位。如果设置的 R_p 范围过大，真实的 R_p 只占这个范围的一小部分，这样就会浪费 LDC1000 内部 ADC 的精度。

手册中给出 R_p_Max 的选择方法：

- 将 LDC1000 的外部线圈设置为涡流损耗最小，例如将金属物体远离线圈。
- 测试此时线圈的等效并联谐振阻抗 R_p ，需要使用**阻抗分析仪**。LC 谐振组件与 LDC1000 断开测试 R_p 。
- 将 R_p 的值乘以 2，在手册的 table 7 中找最接近的值。注意 table 7 中只有 32 个值，也就是说 R_p_Max 寄存器虽然有 8bit 但只能使用 0x00 到 0x1F 的范围。

例如用阻抗分析仪测试出 R_p 是 18k，则 $18k * 2 = 36k$ ，table 7 中 38.785k 是最接近的。

手册中给出的 R_p_Min 的选择方法

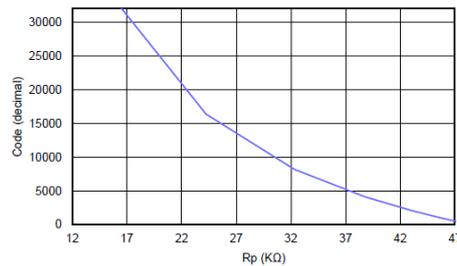
- 将 LDC1000 的外部线圈设置为涡流损耗最大，例如将金属物体靠近线圈。
- 测试此时线圈的等效并联谐振阻抗 R_p ，需要使用**阻抗分析仪**。LC 谐振组件与 LDC1000 断开测试 R_p 。

- 将 R_p 的值除以 2，在手册的 table 9 中找最接近的值。注意 table 9 中只有 32 个值，也就是说 R_p_Min 寄存器虽然有 8bit 但只能使用 0x20 到 0x2F 的范围。

上述方法中，金属物体与 LC 的距离是应用场景中的最大和最小距离，也就是说设备的结构设计好后，这个最大和最小距离由设备的结构决定。

5.3.2 R_p 的 Min 和 Max 值电路意义分析

等效并联阻抗 R_p 表示的就是涡流损耗， R_s 跟涡流损耗成正比，所以 R_p 跟涡流损耗成反比。 R_p 越小，涡流损耗越大。寄存器 R_p_Max 和 R_p_Min 是设定 LDC1000 内部 ADC 的信号调理电路的增益。 R_p_Max 决定 ADC 的下限，也就是能检测的最小信号； R_p_Min 决定 ADC 的上限，也就是能检测的最大信号。回顾一下 R_p 跟 code 之间的关系图， R_p 决定上下限的作用更清晰。



也可以从实验中看出这个检查范围对检测精度的影响。实验说明：金属物跟 LDC1000 的线圈固定一个距离，用于比照。设定一个 R_p_Min 值，将金属物远离，读 code 的值（0x21 和 0x22 寄存器）并平均，记为 min，然后将金属物放在固定位置，读 code 的值并平均，记为 max。然后更换一个 R_p_Min 值，继续一次实验。可以得到下表数据。

R_p_Min Kohm	1.796	2.394	3.078	4.309	5.387	由寄存器转化的 R_p 值，控制检测的上限
min	3660	4738	6229	9127	11940	金属物远离线圈
max	3701	4825	6380	9315	12227	金属物靠近线圈，保证各次实验距离相等
差值	41	87	151	188	287	对相同的涡流损耗有不同的差值

将 R_p_Min 的电阻值变小，可以将 ADC 量程加大，但是这样对于相同的涡流损耗量化后的 code 值位数减小，也就是精度降低。结合 R_p 和 code 的图，左边曲线变化陡峭，右边曲线变化平缓，也就是说 R_p_Min 值对 code 的影响快， R_p_Max 对 code 值影响慢。

