**1186普通LCD人体秤模板**

**使用说明与注意事项**

**REV1.0**

--------------------------------------------------------

深圳市西城微科电子有限公司

地址：深圳市福田区新闻路深茂商业中心26B

电话：0755-83063040  传真：0755-83065035

网站：<http://www.sictech.com.cn>

**版本历史**

|  |  |  |
| --- | --- | --- |
| 历史版本 | 修改内容 | 版本日期 |
| REV 1.0 | 初始版本 | 2019年1月5日 |
|  |  |  |

目录

[**第一章** **初识模板** 5](#_Toc534389140)

[**1.1** **模板程序介绍** 5](#_Toc534389141)

[**1.2** **程序规范相关** 5](#_Toc534389142)

[1.2.1 子程序命名： 5](#_Toc534389143)

[1.2.2 变量命名： 5](#_Toc534389144)

[**1.3 工程文件组织** 6](#_Toc534389145)

[**1.3** **流程图** 6](#_Toc534389146)

[**第二章** **配置文件** 6](#_Toc534389147)

[**2.1 配置文件介绍** 6](#_Toc534389148)

[**2.2 如何修改配置文件** 6](#_Toc534389149)

[**2.3 注意事项** 6](#_Toc534389150)

[**第三章** **库文件** 7](#_Toc534389151)

[**3.1 引入库情况** 7](#_Toc534389152)

[**3.2 库文件说明及使用方法** 7](#_Toc534389153)

[ Fun\_Math\_Add2\_2 7](#_Toc534389154)

[ Fun\_Math\_Sub2\_2 7](#_Toc534389155)

[ Fun\_Math\_Sub2\_2\_Neg 7](#_Toc534389156)

[ Fun\_Math\_Sub3\_3 8](#_Toc534389157)

[ Fun\_Math\_Sub3\_3\_Neg 8](#_Toc534389158)

[ Fun\_Math\_Div6\_3 8](#_Toc534389159)

[ Fun\_Math\_Div6\_3\_Rounded 9](#_Toc534389160)

[ Fun\_Math\_Mul3\_3 9](#_Toc534389161)

[ Fun\_Math\_Hex3\_Bcd 10](#_Toc534389162)

[ Fun\_ProcAdc 10](#_Toc534389163)

[ Fun\_ProcAdcStart 10](#_Toc534389164)

[ Fun\_SetZeroPoint 10](#_Toc534389165)

[ Fun\_SetCountZero 11](#_Toc534389166)

[ Fun\_CurAD\_Sub\_ZeroAD 11](#_Toc534389167)

[ Fun\_ScanWeihgt 11](#_Toc534389168)

[ Fun\_GetAutoOnADC 11](#_Toc534389169)

[ Fun\_GetCount 11](#_Toc534389170)

[ Fun\_key\_Scan 11](#_Toc534389171)

[ Fun\_3W\_CAL\_Init 11](#_Toc534389172)

[ Fun\_3W\_CAL\_Disable 11](#_Toc534389173)

[ Fun\_3W\_CAL\_ResetLock 11](#_Toc534389174)

[ Fun\_3W\_CAL 12](#_Toc534389175)

[ Fun\_Delay\_100MS 12](#_Toc534389176)

[ Fun\_Delay\_40MS 12](#_Toc534389177)

[ Fun\_Delay\_20MS 12](#_Toc534389178)

[ Fun\_Delay\_10MS 12](#_Toc534389179)

[ Fun\_Delay\_1MS 12](#_Toc534389180)

[ Fun\_power\_Init 12](#_Toc534389181)

[ Fun\_power\_Close 12](#_Toc534389182)

[ Fun\_GPIO\_Init 13](#_Toc534389183)

[ Fun\_ADC\_Init 13](#_Toc534389184)

[ Fun\_ADC\_Close 13](#_Toc534389185)

[ Fun\_TIMER\_init 13](#_Toc534389186)

[ Fun\_TIMER\_close 13](#_Toc534389187)

[ Fun\_RAM\_Zero 13](#_Toc534389188)

[ Table\_Lcd\_Num 13](#_Toc534389189)

[ Fun\_LCD\_Init 13](#_Toc534389190)

[ Fun\_LCD\_Close 13](#_Toc534389191)

[ Fun\_LCD\_Load 14](#_Toc534389192)

[ Fun\_OTP\_READ\_CAL 14](#_Toc534389193)

[ Fun\_OTP\_WRITE\_CAL 14](#_Toc534389194)

[**第四章** **应用层** 14](#_Toc534389195)

[**4.1 GPIO** 14](#_Toc534389196)

[**4.2 系统TIMER** 14](#_Toc534389197)

[**4.3 系统时钟** 14](#_Toc534389198)

[**4.4 看门狗** 15](#_Toc534389199)

[**4.5 LCD显示** 15](#_Toc534389200)

[**4.5.1 LCD显示驱动原理** 15](#_Toc534389201)

[**4.5.2 如何根据LCD逻辑图来定义段码位图** 15](#_Toc534389202)

[**4.6 按键** 17](#_Toc534389203)

[**4.7 重量开机** 17](#_Toc534389204)

[**4.8 标定** 17](#_Toc534389205)

[**4.9 ADC** 17](#_Toc534389206)

[**4.10 重量点数** 18](#_Toc534389207)

[**4.11 默认单位** 18](#_Toc534389208)

[**4.12 外部中断** 18](#_Toc534389209)

[**4.13 低电压检测** 18](#_Toc534389210)

[**第五章** **扩展与兼容性** 18](#_Toc534389211)

1. **初识模板**
   1. **模板程序介绍**

该模板工程以芯海科技8 BIT RISC SOC（CSU8RP1186）为载体实现了一个普通LCD人体秤功能，同时模板工程将部分代码封装成库的形式供应用逻辑层调用，工程中相近功能的代码文件以模块化的形式组织，方便后期的开发与维护工作。

* 1. **程序规范相关**

考虑到后期开发的不可能由专人来维护指定架构风格的代码，现对代码做一定程序的规范。

* + 1. 子程序命名：

以Fun\_xx\_yy形式

Fun 表示要实现某一功能的方法，

xx 表示这个方法隶属于哪个类别,例如 LCD，ADC,OTP等

yy 表示这类别要执行的具体功能概述，例如init,close,write,read等

例：LCD初始化子程序名字的写法，Fun\_LCD\_Init。

* + 1. 变量命名：

变量的定义没有子程序那样的严格，只要遵循一个原则：看到变量名就可明白这个变量目的，例如零点，根据其中长度一般为3字节，我们用汇编表示为ZeroH、ZeroM、ZeroL。

另外要特别说明一下位定义的要求，由于随着程序功能的增加，标志位的使用也会成倍增加，标志位使用错乱的可能就提高了，A变量中标志位xx，可能在使用时却把B变量弄成操作数1，例如 BSF B,xx，这样程序编译不会报错一运行就不正常了。

于是我们寻找了一种方法，当然还有更好的方法。我们拿主流程变量SysFlow来说，在它下面有两个标志位，一个用来表示上电处理流程命名为B\_SysFlow\_PWR，一个用来表示秤体流程命名为B\_SysFlow\_SCALE。如图1

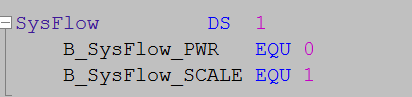


图1 SysFlow及标志位定义

拿B\_SysFlow\_PWR来说，其中

B：表示这是一个位定义

SysFlow: 表示它隶属于哪个变量

PWR：表示上电

写成这样的形式主要是为了后期查找BUG时的方便，比如我们由于其它原因把本意

要修改的标志位写在了其它的变量名下，如 BSF ScaleFlow,B\_SysFlow\_PWR,当我们在运行调试时发现程序没有按我们预期设定的那样运行，于是当我们看到这行语句时，我们会发现标志位与隶属变量不是同一个，这样就解决了一个BUG。

**1.3 工程文件组织**

图2列出了IDE工程文件的分布组织，

主要分4个部分：头文件，主流程，用户区，秤模块

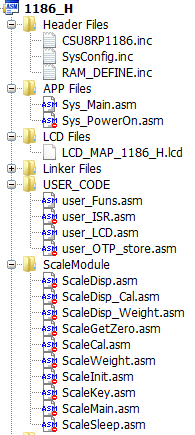


图 2 工程文件分布组织

**1.4流程图**



****

上面三个文档为整个工程模板的流程图

1. **配置文件**

**2.1 配置文件介绍**

文件”SysConfig.inc “里面是一些宏定义，用于初始化对应的片内外设与其它功能模块的参数设定

**2.2 如何修改配置文件**

后面的章节会对各部的宏定义作详细介绍

**2.3 注意事项**

后面的章节会对各部的宏定义作详细介绍

1. **库文件**

**3.1 引入库情况**

目前引入的库主要封装了不常改动但是必须的代码段。

**3.2 库文件说明及使用方法**

* Fun\_Math\_Add2\_2

功能描述：两字节长度的变量加法，TempRam3、4 += TempRam5、6

输入：操作数1： TempRam3(高8位)、TempRam4（低8 位）

操作数2： TempRam5(高8位)、TempRam6（低8 位）

输出：操作数1： TempRam3(高8位)、TempRam4（低8 位）

例：1000 = 112 + 888

MVL LOW 112

MOVWF TempRam4

MVL HIGH 112

MOVWF TempRam3

MVL LOW 888

MOVWF TempRam6

MVL HIGH 888

MOVWF TempRam5

CALL Fun\_Math\_Add2\_2

上面的代码段执行完后，TempRam3中值为03H，TempRam4中值为E8H，十六进制03E8H的十进制为1000。

* Fun\_Math\_Sub2\_2

功能描述：两字节长度的变量减法，TempRam3、4 -= TempRam5、6

输入：操作数1： TempRam3(高8位)、TempRam4（低8 位）

操作数2： TempRam5(高8位)、TempRam6（低8 位）

输出：操作数1： TempRam3(高8位)、TempRam4（低8 位）

例：776 = 888 - 112

MVL LOW 888

MOVWF TempRam4

MVL HIGH 888

MOVWF TempRam3

MVL LOW 122

MOVWF TempRam6

MVL HIGH 122

MOVWF TempRam5

CALL Fun\_Math\_Sub2\_2

上面的代码段执行完后，TempRam3中值为03H，TempRam4中值为08H，十六进制0308H的十进制为776。

* Fun\_Math\_Sub2\_2\_Neg

功能描述：两字节长度的变量减法后，如果需要取这两个差的绝对值，调用这个子程序，必须在调用Fun\_Math\_Sub2\_2后再调用Fun\_Math\_Sub2\_2\_Neg

例：776 = | 112 – 888 |

MVL LOW 112

MOVWF TempRam4

MVL HIGH 112

MOVWF TempRam3

MVL LOW 888

MOVWF TempRam6

MVL HIGH 888

MOVWF TempRam5

CALL Fun\_Math\_Sub2\_2

CALL Fun\_Math\_Sub2\_2\_Neg

上面的代码段执行完后，TempRam3中值为03H，TempRam4中值为08H，十六进制0308H的十进制为776

* Fun\_Math\_Sub3\_3

同Fun\_Math\_Sub2\_2，只不过是操作数为3字节长度

* Fun\_Math\_Sub3\_3\_Neg

同Fun\_Math\_Sub2\_2\_Neg，只不过是操作数为3字节长度

* Fun\_Math\_Div6\_3

功能描述：六字节长度 / 三字节长度变量除法,不带四舍五入

输入：

操作数1： TempRam1~ TempRam6

操作数2： TempRam11 (高8位)、TempRam12（中8 位）、TempRam13（低8 位）

输出：

商 ：TempRam4(高)~TempRam6（低 ）

余数：TempRam1(高)~TempRam3（低 ）

例：1000/256 = 3 ….. 232

MVL LOW 1000

MOVWF TempRam6

MVL HIGH 1000

MOVWF TempRam5

CLRF TempRam4

CLRF TempRam3

CLRF TempRam2

CLRF TempRam1

MVL LOW 256

MOVWF TempRam13

MVL HIGH 256

MOVWF TempRam12

CLRF TempRam11

CALL Fun\_Math\_Mul3\_3

上面的代码段执行完后，

商： TempRam4中值为00H, TempRam5中值为00H, TempRam6中值为03H

十六进制000003H的十进制为3

余数：TempRam1中值为00H，TempRam2中值为00H, TempRam3中值为E8H

十六进制0000E8H的十进制为232

* Fun\_Math\_Div6\_3\_Rounded

功能描述：调用六字节长度 / 三字节长度变量除法后，如果需要四舍五入时调用这个子程序

例：1000/256 = 3 ….. 232

MVL LOW 1000

MOVWF TempRam6

MVL HIGH 1000

MOVWF TempRam5

CLRF TempRam4

CLRF TempRam3

CLRF TempRam2

CLRF TempRam1

MVL LOW 256

MOVWF TempRam13

MVL HIGH 256

MOVWF TempRam12

CLRF TempRam11

CALL Fun\_Math\_Mul3\_3

CALL Fun\_Math\_Div6\_3\_Rounded

上面的代码段执行完后，

商： TempRam4中值为00H, TempRam5中值为00H, TempRam6中值为04H

十六进制000004H的十进制为4

余数：TempRam1中值为00H，TempRam2中值为00H, TempRam3中值为E8H

十六进制0000E8H的十进制为232

* Fun\_Math\_Mul3\_3

功能描述：三字节长度的变量乘法，

输入：

操作数1： TempRam4(高8位)、TempRam5（中8 位）、TempRam6（低8 位）

操作数2： TempRam11(高8位)、TempRam12（中8 位）、TempRam13（低8 位）

输出：

TempRam1(高)~TempRam6（低 ）

例：99456 = 888 \* 112

MVL LOW 888

MOVWF TempRam6

MVL HIGH 888

MOVWF TempRam5

MVL 00H

MOVWF TempRam4

MVL LOW 122

MOVWF TempRam13

MVL HIGH 122

MOVWF TempRam12

MVL 00H

MOVWF TempRam11

CALL Fun\_Math\_Mul3\_3

上面的代码段执行完后，

TempRam1中值为00H，TempRam2中值为00H, TempRam3中值为00H,TempRam4中值为01H, TempRam5中值为84H, TempRam6中值为80H

十六进制018480H的十进制为99456

* Fun\_Math\_Hex3\_Bcd

功能描述：三字节的变量转换成6位的BCD码 ，主要用于显示查表工作

输入：

操作数1： TempRam11 (高8位)、TempRam12（中8 位）、TempRam13（低8 位）

输出：

TempRam1(高)~TempRam6（低 ）

例：01E240H 转换在6位的BCD码为123456

MVL 040H

MOVWF TempRam13

MVL 0E2H

MOVWF TempRam12

MVL 001H

MOVWF TempRam11

CALL Fun\_Math\_Hex3\_Bcd

上面的代码段执行完后，

TempRam1中值为01H, TempRam2中值为02H, TempRam3中值为03H

TempRam4中值为04H, TempRam5中值为05H, TempRam6中值为06H

* Fun\_ProcAdc

功能描述：ADC滤波器

输入：无

输出：无

这个子程序主要是将ADC输出的信号量进行处理，用户不用关心内部操作

* Fun\_ProcAdcStart

功能描述：复位ADC滤波器并开始运行

* Fun\_SetZeroPoint

功能描述：刷新零点，同时调用Fun\_SetCountZero

输入：无

输出：无

* Fun\_SetCountZero

功能描述：秤台重量低于最小锁定重量时调用

输入：无

输出：无

* Fun\_CurAD\_Sub\_ZeroAD

功能描述：当前滤波器输出ADC 减去零点ADC

输入：无

输出：无

* Fun\_ScanWeihgt

功能描述：重量开机检测，秤处于低功耗模式时1秒调用一次。

输入：无

输出：标志位【SysFlag1,B\_SysFlag1\_WakeUp】，如果为1表明重量开机。

* Fun\_GetAutoOnADC

功能描述： 计算开机重量

输入：无

输出：无

* Fun\_GetCount

功能描述： 将ADC内码转换成重量

输入：无

输出：CountH, CountL

* Fun\_key\_Scan

功能描述： 按键长短按处理,输出按键长短按状态

输入：Key\_IO\_PRESS

输出：Key\_TRG,Key\_HOLD

* Fun\_3W\_CAL\_Init

功能描述： 使能并初始化重量标定

输入：无

输出：无

* Fun\_3W\_CAL\_Disable

功能描述： 关闭重量标定

输入：无

输出：无

* Fun\_3W\_CAL\_ResetLock

功能描述： 解除锁定调用处理对应重量标定的标志位

输入：无

输出：无

* Fun\_3W\_CAL

功能描述： 解除锁定调用处理对应重量标定的标志位

输入：无

输出：标志位【ScaleFlag3,B\_ScaleFlag3\_3wCalOk】，1为三点标定成功

* Fun\_Delay\_100MS

功能描述： 延时100MS

输入：无

输出：无

* Fun\_Delay\_40MS

功能描述： 延时40MS

输入：无

输出：无

* Fun\_Delay\_20MS

功能描述： 延时20MS

输入：无

输出：无

* Fun\_Delay\_10MS

功能描述： 延时10MS

输入：无

输出：无

* Fun\_Delay\_1MS

功能描述： 延时1MS

输入：无

输出：无

* Fun\_power\_Init

功能描述：设定芯片模拟电源部分

输入：无

输出：无

* Fun\_power\_Close

功能描述：关闭芯片模拟电源部分

输入：无

输出：无

* Fun\_GPIO\_Init

功能描述：初始化芯片GPIO

输入：无

输出：无

* Fun\_ADC\_Init

功能描述：初始化芯片ADC

输入：无

输出：无

* Fun\_ADC\_Close

功能描述：关闭芯片ADC

输入：无

输出：无

* Fun\_TIMER\_init

功能描述：初始化芯片TIMER

输入：无

输出：无

* Fun\_TIMER\_close

功能描述：关闭芯片TIMER

输入：无

输出：无

* Fun\_RAM\_Zero

功能描述：清除连续多个字节的RAM

输入：[BSR,IRP0] ,[WORK]

输出：无

* Table\_Lcd\_Num

功能描述：LCD数字查表

输入：[WORK]

输出：LCD数字段码

* Fun\_LCD\_Init

功能描述：初始化芯片LCD

输入：无

输出：无

* Fun\_LCD\_Close

功能描述：关闭芯片LCD

输入：无

输出：无

* Fun\_LCD\_Load

功能描述：加载显示缓冲区数据到LCD显示寄存器

输入：无

输出：无

* Fun\_OTP\_READ\_CAL

功能描述：读取标定数据到RAM中

输入：无

输出：无

* Fun\_OTP\_WRITE\_CAL

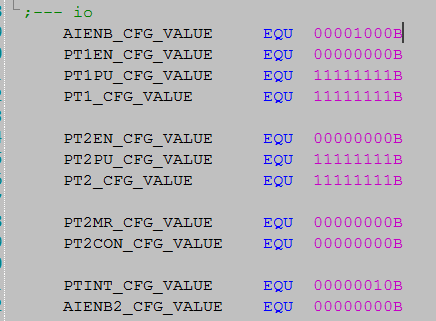
功能描述：写标定数据到OTP ROM中

输入：无

输出：无

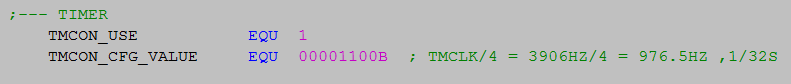
1. **应用层**

**4.1 GPIO**



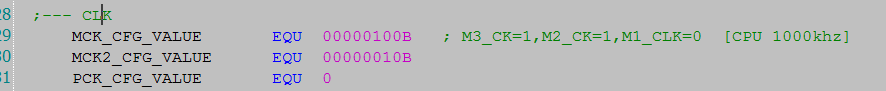
SysConfig.inc 中GPIO相关的宏定义

**4.2 系统TIMER**



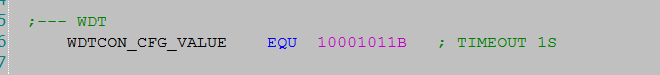
SysConfig.inc 中定时器TIMER的宏定义

**4.3 系统时钟**



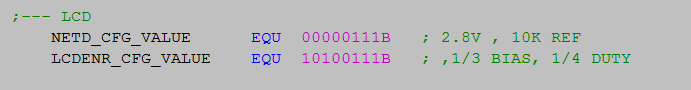
SysConfig.inc 中系统时钟的宏定义

**4.4 看门狗**



SysConfig.inc 中看门狗的宏定义,这个立即数将赋值给寄存器 WDTCON

**4.5 LCD显示**



SysConfig.inc 中宏定义，用户按实际项目来进行NETD与LCDENR寄存器的配置

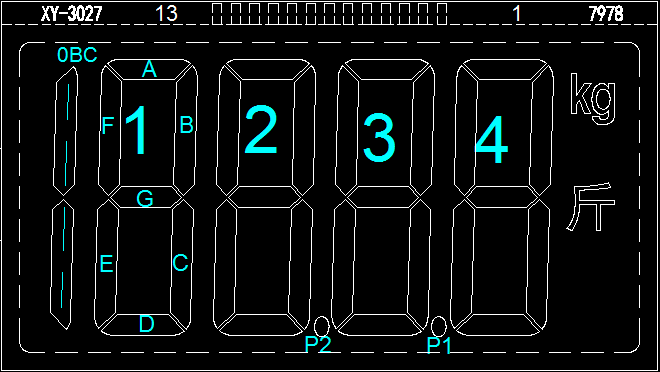
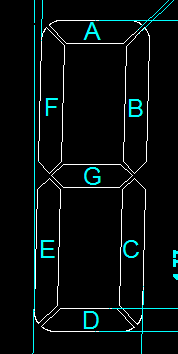
**4.5.1 LCD显示驱动原理**

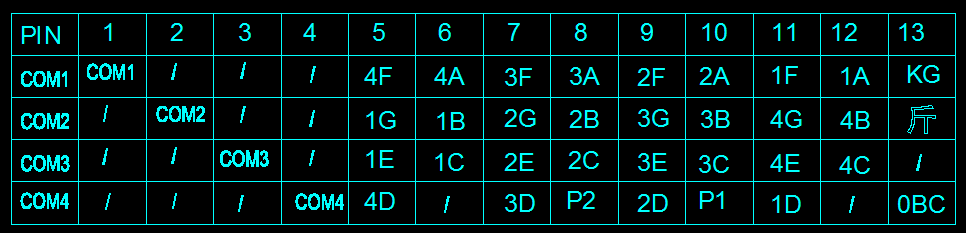
模板中的LCD驱动采用了描点原理，驱动的最小单元为一个位。程序员需要将LCD寄存器中各个位与LCD图中的显示内容一一映射，不再需要关心LCD的SEG连线顺序，降低了开发中LCD显示的难度。

**4.5.2 如何根据LCD逻辑图来定义段码位图**

关于如何使用模板程序的LCD驱动，下面以例子来说明：

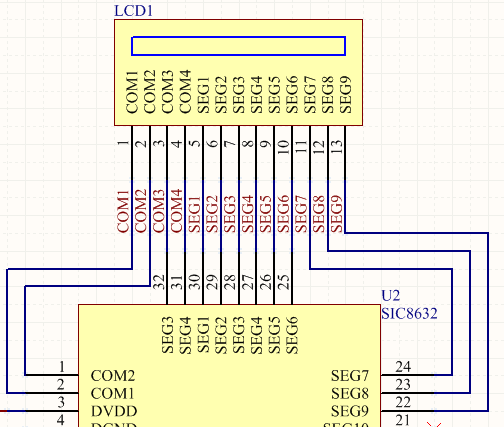
LCD逻辑图如下：



从LCD逻辑图上看，逻辑确实有些凌乱

原理图如下：



确认LCD逻辑图与原理图的接法后，我们就可以开始进行描点工作了。

模板程序的user\_LCD.asm文件中有7个变量LCD1\_map，LCD2\_map，LCD3\_map，LCD4\_map，LCD5\_map，LCD6\_map，LCD7\_map，与LCD寄存的对应关系如下表

|  |  |
| --- | --- |
| LCD1\_map[7~4] | LCD1\_map[3~0] |
| LCD2[7~4] | LCD1[3~0] |

|  |  |
| --- | --- |
| LCD2\_map[7~4] | LCD2\_map[3~0] |
| LCD4[7~4] | LCD3[3~0] |

|  |  |
| --- | --- |
| LCD3\_map[7~4] | LCD3\_map[3~0] |
| LCD6[7~4] | LCD5[3~0] |

|  |  |
| --- | --- |
| LCD4\_map[7~4] | LCD4\_map[3~0] |
| LCD8[7~4] | LCD7[3~0] |

|  |  |
| --- | --- |
| LCD5\_map[7~4] | LCD5\_map[3~0] |
| LCD10[7~4] | LCD9[3~0] |

|  |  |
| --- | --- |
| LCD6\_map[7~4] | LCD6\_map[3~0] |
| LCD12[7~4] | LCD11[3~0] |

|  |  |
| --- | --- |
| LCD7\_map[7~4] | LCD7\_map[3~0] |
| LCD14[7~4] | LCD13[3~0] |

然后我们开始给需要显示的段码位进行描点

；数字1的段码位描点

#DEFINE Set\_Num1\_F BSF LCD4\_map,0

#DEFINE Set\_Num1\_G BSF LCD1\_map,1

#DEFINE Set\_Num1\_E BSF LCD1\_map,2

#DEFINE Set\_Num1\_D BSF LCD4\_map,3

#DEFINE Set\_Num1\_A BSF LCD4\_map,4

#DEFINE Set\_Num1\_B BSF LCD1\_map,5

#DEFINE Set\_Num1\_C BSF LCD1\_map,6

；数字2的段码位描点

#DEFINE Set\_Num2\_F BSF LCD3\_map,0

#DEFINE Set\_Num2\_G BSF LCD2\_map,1

#DEFINE Set\_Num2\_E BSF LCD2\_map,2

#DEFINE Set\_Num2\_D BSF LCD3\_map,3

#DEFINE Set\_Num2\_A BSF LCD4\_map,4

#DEFINE Set\_Num2\_B BSF LCD2\_map,5

#DEFINE Set\_Num2\_C BSF LCD2\_map,6

#DEFINE Set\_Num2\_P2 BSF LCD2\_map,7

；数字3的段码位描点

#DEFINE Set\_Num3\_F BSF LCD2\_map,0

#DEFINE Set\_Num3\_G BSF LCD3\_map,1

#DEFINE Set\_Num3\_E BSF LCD3\_map,2

#DEFINE Set\_Num3\_D BSF LCD2\_map,3

#DEFINE Set\_Num3\_A BSF LCD2\_map,4

#DEFINE Set\_Num3\_B BSF LCD3\_map,5

#DEFINE Set\_Num3\_C BSF LCD3\_map,6

#DEFINE Set\_Num3\_P1 BSF LCD3\_map,7

；数字4的段码位描点

#DEFINE Set\_Num4\_F BSF LCD1\_map,0

#DEFINE Set\_Num4\_G BSF LCD4\_map,1

#DEFINE Set\_Num4\_E BSF LCD4\_map,2

#DEFINE Set\_Num4\_D BSF LCD1\_map,3

#DEFINE Set\_Num4\_A BSF LCD1\_map,4

#DEFINE Set\_Num4\_B BSF LCD4\_map,5

#DEFINE Set\_Num4\_C BSF LCD4\_map,6

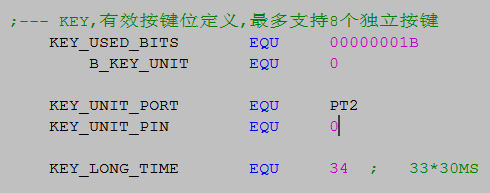
；其它符号的段码位描点

#DEFINE Set\_DOT\_KG BSF LCD5\_map,0

#DEFINE Set\_DOT\_SJ BSF LCD5\_map,1

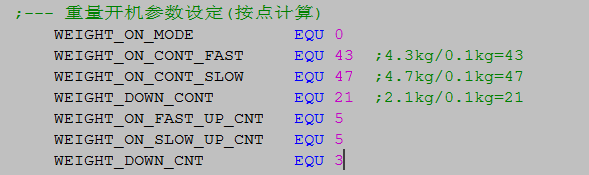
#DEFINE Set\_DOT\_OBC BSF LCD5\_map,3

**4.6 按键**



SysConfig.inc 中按键宏定义,用户需要按照实际电路来进行对应IO的映射。

**4.7 重量开机**



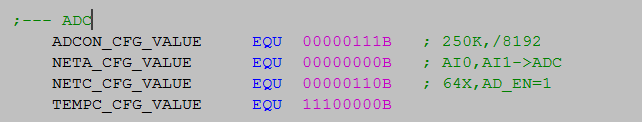
SysConfig.inc 中重量参数的宏定义，用户需要按照实际需求来进行开机重量值的设定。

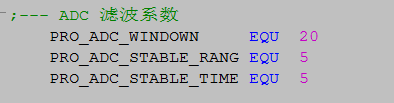
**4.8 标定**



SysConfig.inc 中重量标定与手动标定时第一标定点通过的最小阀值的宏定义

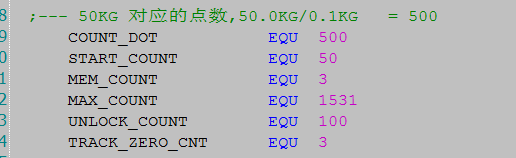
**4.9 ADC**





SysConfig.inc 中ADC配置与滤波器参数的宏定义

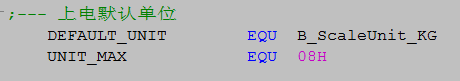
**4.10 重量点数**



SysConfig.inc 中宏定义

模板程序以0.1KG分度计算处理重量

**4.11 默认单位**



SysConfig.inc 中宏定义

上电默认KG单位

**4.12 外部中断**

模板中采用PT20的外部中断0程序，在开机时PT20做单位键使用，低功耗时PT20下降沿空唤醒开机，详情请见模板程序。

**4.13 低电压检测**

这部分代码由于灵活性较强，暂不放入库中

1. **扩展与兼容性**

用户可在此基础上进行其它功能的扩展，比如脂肪秤，蓝牙秤,WIFI秤等