国网单相智能电表嵌入式软件设计

金蜂通信有限公司研发中心　　**吴述梗**

摘要: 本文以金蜂单相电表飞思卡尔 MZ60CPU 方案为实例，结合国网智能电表系列标准(宣贯材料)的具体功能要求，详细分析了其嵌入式软件设计需求，阐明了为达到产品较高性价比和可靠性对 CPU 片上资源合理优化分配方案，并以实例详尽说明软件中部分关键模块设计思路及编程实现技巧。

一：单相智能电表软件设计需求分析

根据国家电网行业标准的要求，结合本方案的硬件，单相智能电表内的嵌入式软件需完成以下功能：

(一)：实现以下与硬件相关的底层接口，包括:

1：通过 SPI 口驱动专用计量芯片 RN8209, 完成与其命令的交互和数据读写。
2：通过模拟 I2C 总线，驱动液晶控制器 BU9792, 完成与其命令的交互和数据读写。
3：通过模拟 I2C 总线，驱动接 RTC 芯片 PCF2129, 完成与其命令的交互和数据读写。
4：通过模拟 I2C 总线，驱动接 EEPROM AT24L512, 完成与其命令的交互和数据读写。
5：通过软件模拟的 ISO7816 协议时序，完成 ESAM 芯片的热复位、读/写操作。
6：通过软件模拟的 ISO7816 协议时序，完成 IC 卡的插入检测、上电控制、复位、数据读/写 和下电操作。
7：通过软件模拟的带 38KHz 调制的异步红外串口，完成红外数据的接收和发送。
8：利用片内 A/D,配合相关外围电路，完成电表交流供电状态、后备电池电压、跳闸断电检测等 A/D 数据的采集和滤波。
9：利用片内 I/O 腿，完成跳闸控制继电器的开关双向驱动、报警、跳闸、电量脉冲 LED 灯和液晶背光、报警蜂鸣器、载波控制等的输出控制信号的实时控制。
10：利用片内 I/O 腿，完成巡显、编程按键和开盖检测、载波状态等开关输入信号的检测。
11：利用片内两个异步串口，完成电表 485 口和载波接口的物理层及链路层数据收/发。

（二）：在实现上述底层驱动的基础上，实现以下国网电表标准要求的应用功能：

1：电能量和实时电参数的采集、计算，以及由此触发的数据存储。
2：电能和多费率的电费计量，以及由此触发的数据存储和对外交互。
3：按标准要求进行本地液晶的按键显示和自动巡显。
4：按标准要求完成本地 IC 卡与电表的各种已定义的交互操作，以及由此触发的数据交互和更新、由此触发的与 ESAM 的数据交互。
5：按标准要求完成来自 485 口、载波接口、红外接口的 DL/T645-2007 通信协议数据解析，并根据指令完成标准既定的对应功能，回复相应数据，并完成由此触发的本地数据检索和存储，由此触发的与 ESAM 的数据交互。

（三）：除上述功能外，还要实现电表生产中的出厂调校、参数配置等功能。

（第二、三项只是概要的叙述，其全部功能需求已由国家电网企业标准及宣贯材料非常详尽的定义，在这不再赘述）

二：MZ60 方案的硬件特性和片上可用资源优化分配

为了合理有效的利用 CPU 片上资源设计出高性价比驱动软件，先对 MZ60 CPU 的硬件特性做一简短回顾：

MZ60 完整型号为 SC116009,是飞思卡尔 HC08 指令集的高速 8 位 CPU； 内置 1%精度可利用 NVM 中存储的信息调整频率的高频 RC 震荡器，省掉外部晶振而节省成本和降低功耗。片上 60K 字节 FLASH 程序存储器正适合于单相电表的代码量而不浪费。2K 字节的 RAM 不充裕，但在采取本文后述的一些编程技巧后还基本够用。有两个硬件异步串口，正合适用来做电表的 485 接口和载波接口。有 16 路 8 位 A/D,可宽裕的任选几路经分压电阻网络配合完成电池电压检测和交流供电检测，并能在内部选择采集内置基准电压源和芯片供电电压而对自身电源自检。有独立时钟源的看门狗和上电检测(POR)及低电压检测(LVD)多个复位源，以及非法指令中断，能可靠的省去外部复位及看门狗硬件以降低成本。它有两个低功耗省电模式可方便的用于后备电池工作。最重要的是其内置的三组功能强大的定时器组，可配置成三组不同计数频率和重装模数的捕捉、匹配或 PWM，用来可高效的实现 38KHz 调制的红外收发和两组 ISO7816 接口时序。一组硬件 SPI 接口方便的连接计量采集芯片。每个 I/O 都可内部配置为上拉输出或高阻输入的特性可用来方便的仿真 I2C 接口时序，用来连接液晶、EEPROM 和 R T C 等芯片。另还有未公开的 IO 口按键中断等片上资源，可更有效的进行按键检测和唤醒及开盖等检测。

根据 MZ60CPU 的上述资源特点，本方案在硬件设计时各管脚的连接按下述原则进行，以便充分利用片内集成的外设，达到软件最优性价比设计：

1：ESAM_CLK(ESAM 时钟) 和 Card_CLK(IC 卡时钟) 分别选择利用两组双路的定时器 T2 或 T3 的同一组的两路，即：

方案 A：34 腿(T3.0) 或 35 腿(T3.1)

方案 B： 8 腿(T2,0) 或 11 腿(T2.1)

注：在两组方案中，ESAM_CLK 或 Card_CLK 分别使用定时器的 1 或 2 路都可以。这样，软件编程时，就可以将这组定时器设为 PWM 方式运行产生约 4MHz 的高频时钟且不需中断干预，极大提高 CPU 的利用率，且两路时钟的开、停分别可控。

2：38kHz(红外发送调制时钟) 和 IR_RXD(红外接收线) 分别选择利用两组双路的定时器 T2 或 T3 的同一组的两路，且和 1 中互斥选择，即：

方案 A： 8 腿(T2,0) 或 11 腿(T2.1)

方案 B：34 腿(T3.0) 或 35 腿(T3.1)

注：1：在两组方案中，时钟或接收连定时器的 1 或 2 路都可以。

2：若一选方案 A，则二选方案 B, A/B 分组间不能交叉，比如， 不能 34 腿做 ESAM_CLK 而 8 腿做 Card_CLK。

这样，软件编程时，可充分利用红外接口半双工的特点，在平时等待接收状态时，这组定时器编程为 IR_RXD 腿捕捉中断，而发送时 38kHz 腿编程为 PWM 方式自动发送调制时钟，此时红外发送(IR_TXD)可选任意输出腿。

3：ESAN_I/O(ESAM 模块数据收发线)和 Card_I/O(IC 卡数据收发线)

可分别选择第 4，5，6，7，15，16 腿即定时器 T1 的任一路连接。由于定时器 1 有六路使用同一计数源的定时通道，可分别选为匹配定时或输入捕捉，将其共同的计数源选为总线时钟即 11.0592Mhz 的 8 分频，即可满足系统中全部定时和捕捉的精度和时间长度的两方面要求。这里，ESAN_I/O 和 Card_I/O 在收发中不同状态下被分别灵活编程为下降沿捕捉中断或定时匹配输出，详见后边关于 ISO7816 通讯协议时序软件模拟实现的编程技巧。

4：CF_IN(电能量脉冲输入线) 和 Relay_CHK(跳闸控制后交流电检测线)

可选择定时器 T1 在 3 中选择后剩余的 4 个腿中的任一个。为了准确采集两个电量脉

冲间的时间间隔，用来同步驱动由 CPU 另一 I/O 控制的脉冲灯，CF_IN 需要一个捕捉中断。把 Relay_CHK 也连接上捕捉中断资源，是为了更快更可靠的检测到交流脉冲：不但进行电平检测，还能在几个交流电周期(几十毫秒)内通过检测捕捉到的周期核对是 50Hz 交流电还是干扰脉冲，从而提高系统可靠性。

5：DISP(巡显键)

一定要接 CPU 2 腿(IRQ), 同时接 20k 电阻上拉。因为国网标准对电表交流断电后后备电池供电的苛刻要求，必须在交流断电后使 CPU 进入掉电休眠状态，而是用巡显键唤醒。但 IRQ 是 MZ60 手册上唯一说明的除各定时器通道外的外部中断触发腿。

6：PROG(编程按键)和 Open_CHK(开盖检测按钮)：

分别选择 23，24，25，48，49 这 5 个管腿之一，要接 20k 上拉电阻。

这五个管脚即 MZ60 的 PG 口，是手册上没说但实际可用的“键盘中断”腿，他们都是可以触发外部中断的。因此，在程序设计中可利用其中断功能更快速有效的检测到这两按钮的触发。当然，它们也是可作为普通输入腿用的。

7：BAT_CHK(电池检测)和 Power_CHK(交流供电检测)

分别选择 CPU 第 36，37，38，39，40，41，42，43 等 A/D 输入腿。这两个检测线分别通过各自的分压电阻和保护电路分压到 0-3.3V 内，连接到 CPU 的 A/D 输入腿，软件则定时采集其电压值并数字滤波处理取得所要求数据。

8：其他 I/O 管腿选择优先原则：

专用功能腿选有对应硬件功能的（如 BKGD、串口或 SPI 等)。

各输入信号优先选择具有定时器通道、A/D 输入或中断触发等功能的管腿，以便编程中可灵活高效的对输入数据进行处理。

三：软件总体设计思路：基于状态机和事件中断驱动的编程

根据上述软件设计需求和硬件相关资源特性，为提高 CPU 利用率降低成本，软件设计决定不使用 RTOS 实时操作系统。但其纷杂众多的实时/异步数据接口需求，特别是要软件模拟与 ESAM 和 IC 卡两路 7816 接口时序和一路红外数据收发，同时处理两路(485 和载波)较复杂的 DL/T645-2007 通信协议解析，还要同时处理随时可能发生的计量数据采集、解算，以及插卡、按键、掉/上电等外部干预，和由此触发的计量、冻结、事件记录等数据存储操作。由此可见，传统的简易前后台+中断处理的编程模式已不相适应。为此，提出在传统前后台编成模式下扩展成“基于状态机和事件中断驱动”的新编程框架。具体实现思路是：

1：main（）主循环中顺序按就绪标志处理以下实时性相对要求不高的“慢速事件”，诸如：液晶显示刷新，645 协议解析和命令回复的发送，EEPROM 数据的批量写入和读出，冻结和事件记录的生成等。它们这些任务在主循环中执行时间是不定的，但由系统硬件看门狗界定每次循环时间不超过 5 秒。

2：以两个物理串口收发中断、三个(ESAM_IO、Card_IO 和 IR_RXD)捕捉中断，和 IRQ 及 PG4 口的几个按键中断为主线，建立“事件中断驱动”机制。在 RAM 中建立数据先进先出缓冲队列和相应读写指针，为每个中断建立信息处理状态机，在相应所需数据或管腿状态改变而触发的中断到来驱动下，状态机的逐步步进而完成数据的后台接收、缓冲和处理，并根据结果改变（触发）主循环中各相应处理程序的就绪标志。

3：以定时器的一路 32mS 定时中断为主线，协同调度没有明显物理事件驱动的高实时性任务，如计量信息采集和计算，交流断电和电池电压检测，RTC 时钟读出，定时冻结和费率切换等定时事件处理，主循环中精确定时标志更新等。

4：以上三条准并行操作线程的数据交换主要靠全局变量进行。由于 CPU 不允许中断嵌

套，所以后两线程之间不会引起资源冲突。中断和主循环中的数据交换由数据缓冲队列和关闭中断的临界代码完成。

(具体编程细节请参见程序清单和详细中文注释，为节约篇幅这里从略)

四：软件设计中一些关键功能实现的算法和编程技巧

(一)：电参数和计量值的采集及计算

在软件设计中，由于 CPU 中 RAM 和运算速度的限制，根据实际需求，不采用浮点运算库，所有相关运算均用 32 位(4 字节)有符号整数定点计算。根据计量芯片 RN8209 的计算公式，在编程中有以下关键量值的计算：

在本方案中，电量采集芯片选用了瑞能微的 RN8209, 下面就以此为实例对软件编程中的算法和技巧详细说明：

1：功率值的计算

手册给出的计算公式是：Px=Pt/r 其中，折算系数 r=((2^32) * K * Xh)/3222000000。

其中，K 为电表脉冲常数，在程序中定义为一个 2 字节 16 位(范围 1-65535)的无符号整数，Xh 为 RN8209 中 HFCONST 的值，它也是一个 2 字节无符号整数。而 Pt 是从 RN8209 的 PowerPA 寄存器读出的值，是 4 字节 32 位的有符号整数，最高位为符号位。

Xh 是和电表的硬件设计参数(包括电表脉冲常数 K)相关的一个常量，其计算公式是：

Xh = (23.2075 * Vu * Vi * (10^11))/(K * Un * Ib)　其中：

Un = 额定输入电压，　Ib = 额定输入电流

Vu: 额定电压输入时，电压 AD 通道输入值（引脚上电压*通道增益)

Vi: 额定电流输入时，电压 AD 通道输入值（引脚上电压*通道增益)

对于一个给定设计参数的电表，Un, Ib 为一确定值，对我们设计的电表，Un=220V，Ib=10A。

Vu 与硬件设计的电压采样分压电阻有关，这里取电压通道增益为 1，分压比 1/1081，故 Vu=0.203515V。

Vi 与锰铜电阻值和电流通道增益相关，我们的表用 250 微欧锰铜，取电流通道增益为 16，故 Vi=10*0.00025*16=0.04V。

因此 Xh=(23.2075 * 0.203515 * 0.04 *(10^11))/(K * 220 * 10) =18892297450/(K * 220 * 10) = 8587407.9318/K。

我们的电表设计 K=1200， 由上式代入计算可得， Xh = 7156.

由此，r = (4294967296 * 7156 * 1200)/ 3222000000 = 11446847.6611

在电表中实际要求功率以 4 位小数显示，因此，采用定点计算算出 PX = Px*10000 的值。

PX = (Pt * 10000)/11446847.6611

考虑编程中使用定点整数运算，为了计算中不溢出又保证精度，在编程中采取如下办法：

根据表的额定设计，最大功率值 PxMax 约 270*60 =16.2 千瓦，Pt 的最大读出值<16.2*r =185438937。而 4 字节有符号数最大可表示的值 2147483647，由此可见使用 32 位有符号计算时，保证不溢出可以乘的最大因子仅为 11，按上述公式在程序中直接计算在功率值较大时是会溢出的。 但由于实际中精度要求 4 位小数，在计算时，采用以下技巧：读出 Pt 后，先除以 256，再乘以 997，再除以 4458，即：

PX = (Pt * 10000)/11446847.6611 = (Pt/256)*997)/4458

具体程序实现是： 设 PX 为一个 32 位有符号数变量。从 PowerPA 读回的 4 字节值由高到低顺序存入 bufs[0]，bufs[1]，bufs[2]，bufs[3]

PX= (vs32)((((vu32)bufs[0])<<16)|(((vu32)bufs[1])<<8)|((vu32)bufs[2]));

// 注：这里丢弃最低 1 字节 bufs[3], 即数据整体右移 8 位，相当于除以 256

PX *= 997;

PX = PX/4458;

2：电流有效值计算

RN8209 数据手册给出的理论计算公式为：

Ix = It / (Ris * Ki * 1.25 * (2^23))

其中，It 为从 RN8209 芯片 IARMS 寄存器读出的值，它是 24 位的一个有符号整数。 Ris 为锰铜的阻值，Ki 为电流通道的增益。

为了校表方便，设 Ix = k * It,　则

k = 1/(Ris * Ki * 1.25 * (2^23)), 代入本设计中的参数值，则 k 的理论值为：

k = 1/(0.00025 * 16 * 1.25 * 8388608) = 1/41943。

也就是说，理论上，从 IARMS 寄存器读出的值除以 41943，就是当前电流值。

但由于锰铜电阻与理论是有误差的，需要在实际校表时校准。因此，在这引入电流有效值系数 KiA= k * 0x19000000

很巧合，这时的理论值 KiA=419430400/41943 =10000

这时的计算公式为 Ix = （KiA/419430400) * It

因实际电流精度要求三位小数，计算时要算 IX = （1000 * Ix) 的值。为了防止溢出，采用以下编程技巧：

IX = (1000 * KiA /419430400) * It = ((It/64) * KiA *5)/32768

具体程序实现是： 设 IX 为一个 32 位有符号数变量,从 IARMS 读回的 3 字节值由高到低顺序存入 bufs[0]，bufs[1]，bufs[2]

IX = (vs32)((((vu32)bufs[0])<<24)|(((vu32)bufs[1])<<16)|(((vu32)bufs[2])<<8));

IX /=16384;

IX = ((lsvs32 * (vs32)KiA) * 5)/32768;

//注：第一行中，数据整体左移了 8 位 即*256，故第二行是/16384 而不是 64。

3：电压有效值计算

RN8209 数据手册给出的理论计算公式为：

Ux = (Ut * Rt)/ (Rus * Ku * 1.25 * (2^23))

其中，Ut 为从 RN8209 芯片 URMS 寄存器读出的值，它是 24 位的一个有符号整数。 Rt 为电压通道分压电阻串的总阻值，Kv 为电压通道的增益，Rus 为电压取样电阻的值。

为了校表方便，设 Ux = k * Ut,　则

k = Rt/ (Rus * Kv * 1.25 * (2^23)), 代入本设计中的参数值，则 k 的理论值为：

k = 1081/(1 * 1 * 1.25 * 8388608) = 1/9700。

也就是说，理论上，从 URMS 寄存器读出的值除以 9700，就是当前电压值。

但由于分压电阻与理论是有误差的，需要在实际校表时校准。因此，在这引入电压有效值系数 Ku= k * 0x6400000

这时的理论值 Ku =104857600/9700 =10810

这时的计算公式为 Ux = （Ku/104857600) * Ut

因实际电压精度要求一位小数，计算时要算 UX = （10 * Ux) 的值。为了防止溢出，采用以下编程技巧：

UX = (10 * Ku /104857600) * Ut = ((Ut/256) * Ku)/40960

具体程序实现是： 设 UX 为一个 32 位有符号数变量。从 URMS 读回的 3 字节值由高

到低顺序存入 bufs[0]，bufs[1]，bufs[2]

```
UX= (vu32)bufs[0]*256+(vu32)bufs[1];      //3 字节，丢掉最低字节，相当/256
UX= (UX * (vu32)K_U)/40960;
```

(二)：通用的国网 LCD 显示程序模块设计

在国网企业标准中，对液晶显示的内容和格式都给出了具体统一的要求。若能对于不同的硬件实现方案在软件模块级编写统一的接口函数，而把底层硬件的驱动和应用层的功能分离开，把硬件区别封装在底层函数内部，而在应用层给出一个可简易且灵活配置的机制，则可极大提高软件的可移植性和对硬件改变的适应性。针对这个目标，为液晶显示编写了如下函数：：

void　GW_LED_SHOW(u8 BJSY, u16 XSKZ, u16 BAOJ, u8 DUKA, u8 JTDJ, u8 FLSD, s32 V1)

这个函数在应用层完成了：在电表液晶屏上从可配置的多个模板中选择显示一屏，显示内容由既定汉字和图标的背景、8 位笔段式实时数据和一组由实时标志控制的报警图标，共三层分别可控的内容叠加而成，并有与硬件无关的闪烁控制机制。

从下面的入口参数入手，介绍此模块的设计思想和编程技巧:

参数 u8 BJSY：背景索引号，8 位无符号数，表示要显示的背景字符(格式)和小数点位置的索引。其意义是： 在国网标准中，严格定义了 29 屏液晶显示内容的汉字和图标组合格式，然而在不同厂家的液晶模块中这些图标的 COM/SEG 位置是可以不同的，另外由于 PCB 布线的不同，液晶驱动芯片和液晶屏的各 SEG 对应关系也可灵活改变。现在把这些可变的因素剥离出来，定义成一个固化在 FLASH ROM 代码段的常量数组，而在应用中由这个索引挑选上述已预先配置好的一屏背景。

按国网标准，预标定义如下背景如下表：：

BJS 值	显示项目	数据显示格式	背景中显示的字段
00	液晶清屏	全 0 或空白	全清，并且各报警图标也全清
255	液晶全显	全 8	全置，并且各报警图标也全置
01	当前剩余金额	XXXXXX.XX	(S4),S12,S26,S35,P7
02	当前组合有功总电量	XXXXXX.XX	S4,组合,S7,S20,S21,S37,S36,P7
03	当前组合有功尖电量	XXXXXX.XX	S4,组合,S8,S20,S21,S37,S36,P7
04	当前组合有功峰电量	XXXXXX.XX	S4,组合,S9,S20,S21,S37,S36,P7
05	当前组合有功平电量	XXXXXX.XX	S4,组合,S10,S20,S21,S37,S36,P7
06	当前组合有功谷电量	XXXXXX.XX	S4,组合,S11,S20,S21,S37,S36,P7
07	上 1 月组合有功总电量	XXXXXX.XX	S5,1B,1C,组合,S7,S20,S21,S37,S36,P7
08	上 1 月组合有功尖电量	XXXXXX.XX	S5,1B,1C,组合,S8,S20,S21,S37,S36,P7
09	上 1 月组合有功峰电量	XXXXXX.XX	S5,1B,1C,组合,S9,S20,S21,S37,S36,P7
10	上 1 月组合有功平电量	XXXXXX.XX	S5,1B,1C,组合,S10,S20,S21,S37,S36,P7
11	上 1 月组合有功谷电量	XXXXXX.XX	S5,1B,1C,组合,S11,S20,S21,S37,S36,P7
12	上 2 月组合有功总电量	XXXXXX.XX	S5,1A,1B,1G,1E,1D,组合 S7,S20,S21,S37,S36,P7
13	上 2 月组合有功尖电量	XXXXXX.XX	S5,1A,1B,1G,1E,1D,组合, S8,S20,S21,S37,S36,P7
14	上 2 月组合有功峰电量	XXXXXX.XX	S5,1A,1B,1G,1E,1D,组合, S9,S20,S21,S37,S36,P7
15	上 2 月组合有功平电量	XXXXXX.XX	S5,1A,1B,1G,1E,1D,组合,

			S10, S20, S21, S37, S36, P7
16	上 2 月组合有功谷电量	XXXXXX. XX	S5, 1A, 1B, 1G, 1E, 1D, 组合, S11, S20, S21, S37, S36, P7
17	当前 阶梯 电价	XXXX. XXXX	S4, S20, S22, S35, P5
18	用户户号 低 8 位	XXXXXXXX	户，S6
19	用户户号 高 4 位	XXXX	户，S6
20	表号 低 8 位	XXXXXXXX	S27，S6
21	表号 高 4 位	XXXX	S27，S6
22	当前日期	XX. XX. XX	S23, S24, P5, P7
23	当前时间	XX. XX. XX	S23, S24, P5, P7, P9, P10
24	故障代码	Err -XX	2A, 2F, 2G, 2E, 2D, 3F, 3B, 3G, 3E, 4F, 4B, 4G, 4E,
25	电压	U- XXX. X	2F, 2E, 2D, 2C, 2B, 3G, P8, V
26	电流	XXX. XXX	P6, L, A
27	零线电流	XXX. XXX	P6, N, A
28	功率	P-XX. XXXX	2A, 2F, 2B, 2G, 2E, 3G, S37, P5
29	功率因数	X. XXX	COSf, P6

参数 u16 XSKZ ：显示方式控制字，16 位无符号数，表示要显示的变量和图标的补充特性，各位定义如下。

D15:　0=背景显示不闪烁，　1=背景显示闪烁

D14:　0=变量 V1 显示不闪烁，　1=变量 V1 显示闪烁

D13 D12:　变量 V1 的显示格式，00=不显示，01=带前导 0 显示，10=不带前导 0，11=保留

D11..D0:　对应的 BAOJ 变量各位图标的闪烁控制，0=不闪，1=闪烁。

参数 u16 BAOJ ：屏幕上能显示的 16 个报警图标，对应位=1 表示显示。各位对应的图标定义如下：

D15,D14,D13,D12 :　保留=0

D11　:　1= 点亮 "请购电"汉字提示(S48)

D10　:　1= 点亮 "拉闸" 汉字提示(S49)

D9　:　1= 点亮 "透支" 汉字提示(S50)

D8　:　1= 点亮 "囤积" 汉字提示(S51)

D7,D6　:　保留=0

D5　:　1= 点亮 "黑左箭头"(S38)，功率反向指示

D4　:　1= 点亮 "电池图标"(S41), 电池欠压指

D3　:　1= 点亮 "电话图标"(S16), 红外、485 通信中

D2　:　1= 点亮 "电波图标"(S3), 载波通信中

D1　:　1= 点亮 "编程图标"(S42), 按下编程键允许编程

D0　:　1= 点亮 "铁锁图标”(S43)，三次密码错指示

参数 u8 DUKA : 读卡汉字状态图标提示，

0x00=不显示，0x01=读卡中，0x02=读卡成功， 0x03=读卡失败，

最高位=1（即上述值+0x80）表示此状态的图标闪烁。其他保留。

根据以上数据组合，驱动 "读卡"(S44), "中"(S45), "成功"(S46), "失败"(S47)

参数 u8 JTDJ：阶梯电价图标指示。

高 4 位=1 或 2，表示当前使用哪套阶梯电价，驱动三角框内"1"，"2"图标(S30,S34). 高位=其他不显这组图标

低位 =1,2,3,4 ,表示当前使用的电价梯度号，驱动方框内"1"，"2"，"3"，"4"图标(S1,S2,S14,S15)，

其中高/低位的分别第 7，第 3 位，=1 控制本组图标的闪烁。

低位=除 1,2,3,4 以外的其他值，则不显示这组图标。

参数 u8 FLSD：费率时段图标指示

高 4 位=1，2，3，4，表示所使用的尖峰平谷费率状态，分别驱动圆圈内"尖", "峰", "平","谷" 图标；

低 4 位=1，2， 表示当前计费的时段号，分别驱动圆圈内"1", "2" 图标；

其中高/低位的分别第 7，第 3 位，=1 控制本组图标的闪烁。

不论高低位，如果输入非上述其他数字则不显示这组图标。

参数 s32 V1 ：屏幕上 8 位的大数码管实时显示内容

有符号数，范围为-99999999 到 99999999.

数值的显示前都设为整数，电量为 1/100, 电价和功率为 1/1000, 电压为 1/10, 电流和功率因数为 1/1000。

实现此函数的编程思路及技巧：

1： 闪烁是基于累计调用此函数次数计数的纯软件实现，和液晶驱动器无关的，实现办法是：在偶数次调用本函数时，需要闪烁的显示字段代替为清 0 不显。这需要设一个专用的静态变量记忆调用此函数的次数。

2：在代码空间定义 液晶显示用背景配置表

u8 const LCD_BJSY[30][19]；

它的第一下标 0..29，用来区分不同背景，在函数调用时用参数 **BJSY** 来索引，而其数组中的前 17 字节，与/液晶驱动缓冲区 4 位*33 段=132 位=17 字节相对应。映射为 BU9792 的显示数据缓冲区前 17 字节(因只使用 33 段，故后 3 字节个填 0)。最后 2 字节分别为主数码管右起显示位数和小数点位置

3：在代码段空间，定义 8 字型数码管 0-9 字模常量数组 u8 const LCD_8[10]；

和 主 8 字数码显示添加负号用常数表 u8 const LCD_FHSY[9][2]；

以及一些各零碎图标点亮（置位）的宏。

这样，在调用此函数显示**参数 s32 V1** 的值时，就能依照此表索引既定硬件对应关系的笔段位置数据和消前导 0 及添加负号的笔段。

4：函数在堆栈中开 17 字节的临时显示缓冲区 Xbuf[17]，每次函数调用时，先清 0，再按索引从 LCD_BJSY[]中拷入背景，再根据各参数调用相应宏在 Xbuf[]中置位相应字节的位而“点亮”各图标，然后再根据 V1 的值实时显示 8 位主数码管的值，并完成消零和负号的合成。这些处理都完成后，才调用一次底层的写液晶函数 LcdDatSend(XSbuf, 0, 17); 把这 17 个字节一次写入液晶控制器。函数调用后临时显示缓冲区随堆栈释放。这样，就可用最小的 RAM 和时间开销完成可硬件可灵活配置的显示驱动了。