在LonWorks中X24C45的掉电保护

范来良[1]，白洪光[1]，赵 刚[2]，袁宪强[2]

（1山东省冶金科学研究院，山东 济南 250014；2济南钢铁集团总公司，山东 济南 250101）

摘　要：为了避免数据保存过程中因系统掉电或不稳定可能导致的保存位不正确，采用备份数据和置标志位相结的方法，消除了保存数据时的错误，保证了保存数据的准确性；同时给出了在LonWorks节点中，保存器在保存数据时掉电或电压不稳情况下保护和恢复数据的应用实例。

关键词：NOVRAM；LonWorks；X24C45；掉电保护；恢复数据

中图分类号：TP309.2　　文献标识码：B　　文章编号：1004-4620（2004）03-0030-03

Protection Data of X24C45 when Power off in LonWorks

FAN Lai-Liang[1], BAI Hong-guang[1],ZHAO Gang[2],YUAN Xian-qiang[2]

(1 Shangdong Metallurgical Research Institute, Jinan 250014,China;
2 Jinan Iron and Steel Group,Jinan 250101,China)

Abstract: In order to avoid the unprecise stored data in storing while power off or the electric voltage is unsteady, the backup data is combined with placing a sign to dissolve the mistake of storing data, then the veracity of data storing is ensured. An application example about protecting and storing data during memorizer store data while power off or the electric voltage is unsteady in the LonWorks is also given.

Keywords: NOVRAM; LonWorks; X24C45; data protection when drop the electricity; data instauration

1 问题的提出

采用LonWorks技术设计开发智能计数器时，为了满足计数器计数频率高、计数准确、掉电数据保持等要求，采用了Xicor公司的X24C45存储器。该存储器是串行256位（16x16）NOVRAM（非易失性RAM）；其NOVRAM与EEPROM阵列位对位重叠，可通过软件命令实现两个存储器阵列之间传送数据；NOVRAM可以不受次数限制写入数据，掉电后，其数据将丢失，而EEPROM可满足掉电保持的要求。但EEPROM却有写次数限制，其最小的存储操作次数为百万次，然而计数器的计数次数却远远高于EEPROM的次数限制。为了满足上述要求，可利用X24C45的自动存储功能，设置了该功能，则X24C45的VCC端电压下降到预置的门限时，NOVRAM数据将自动存储到EEPROM中。

在实际应用过程中，用LonWorks开发的节点，采用X24C45来存储数据，节点在停电或电压不稳定时，X24C45的某些存储单元经常出现一些随机数或存储数据过少现象。其原因是：（1）节点在执行存储操作过程中，电源突然停电或电压不稳定，导致CPU在执行写数据时（存储数据到X24C45的NOVRAM），写入了不正确的数据位，使存储的数据位出现了随机数。（2）为了保证X24C45写次数，对存储数据的操作应在NOVRAM

中完成，只有掉电时，数据才自动保存到EEPROM中。在系统RESET事件（系统启动时执行）中，应把X24C45的EEPROM数据调入NOVRAM。由于X24C45的自动存储电压（4.0～4.3V）和CPU的复位电压（4.7V）不一致，当CPU出现复位时，X24C45中的EEPROM和NOVRAM两个存储区域的数据可能会不相同，若在RESET事件中，直接将EEPROM数据调入NOVRAM，导致NOVRAM正确数据被覆盖，使计数器出现漏计。

2 掉电保护方法

2.1 存储数据

为了实现数据的准确性，在对X24C45执行写数据操作过程中，应考虑到掉电和电压波动情况。在存储数据前，必须对要存储的数据进行额外的保护。数据存储流程见图1。

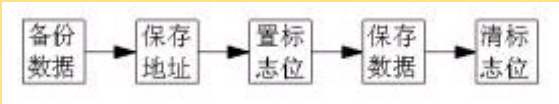

图1 数据存储流程

存储数据时，1片X24C45被额外的占用了3个地址（保存地址位、标志位、备份数据位），剩下的13个地址可存储计数数据。当一个电路中使用了多片X24C45时，每片都应留有标志位，保存地址位和备份数据位可在1片X24C45中共享。

存储过程中的“标志位”标识了该X24C45芯片在执行存储数据过程中是否发生了掉电现象，当系统再次上电时，只须判断该芯片的“标志位”：“标志位”非0，表示该芯片在执行数据存储过程中，发生了掉电现象，当系统再次上电后，系统可根据保存地址把备份的数据写入该地址单元，完成数据的恢复；“标志位”为0，表示正常，无须作处理。

上述的处理过程并不会因为突然掉电而使存储器出现随机数，其原因是：（1）“备份数据”和“保存地址”过程发生掉电。这时标志位为0，原有存储的数据单元数据没有改变。（2）"置标志位"过程发生掉电。这时，存储标志位的存储单元可能产生随机数，但备份数据和地址位已成功保存，若存储标志位的随机数非0，系统再上电后，即可恢复数据；若随机数等于0，原有的数据没有改变，结果只是丢失一位计数。（3）“保存数据”过程发生掉电。这时的标志位、备份数据、地址位都保存完好，再次上电后，因为标志位非0，系统可根据保存的地址位和备份数据，实现数据的恢复。（4）“清标志位”过程发生掉电。这时，存储标志位的存储单元可能产生随机数，但备份数据、地址位和计数数据都成功地保存，若存储标志位的随机数非0，系统再上电后，即可恢复数据；若随机数等于0，表明计数数据已经成功保存。

分析看出，在执行保存数据的过程中，掉电或电压波动都能克服存储器的数据位产生的随机数。采用上述存储过程，最多只能丢失一次计数，在掉电时允许误差范围之内，基本上保证了数据的准确性。

2.2 EEPROM数据保护和数据恢复

对于X24C45自动存储电压和CPU复位电压不一致，所导致的存储在NOVRAM中正确数据没有及时保存到EEPROM中的情况，需在复位事件中作特殊处理。即在CPU的RESET事件中，执行EEPROM数据调入NOVRAM命令前，必须能够判断出EEPROM和NOVRAM哪一个存储区内的数据是正确的。而EEPROM数据只有在停电或初上电状态下，其数据才是正确的，所以，在CPU的RESET事件中，只需要判断X24C45是否是初上电状态。

CPU（NEURON CHIP）复位事件不单是在初上电时才产生，在CPU出现异常，如"看门狗"时间到、运算数据出现异常或网络数据出现错误时，也可产生复位。NEURON C内置函数POWER_UP可以判断CPU是否为初上电状态，但是当系统的供电电压在4.7V（CPU的复位电压）以下、却高于X24C45的自动存储电压间波动时，待系统工作正常后，POWER_UP函数只标识了CPU为初上电状态，却没有正确地反映X24C45为初上电状态，所以应用POWER_UP函数不能正确地判断X24C45是否为初上电状态。

为克服上述情况，可分析X24C45的工作状态图。从X24C45的状态图上看，X24C45初上电时，芯片没

有"写功能"，即存储功能，这时即使执行了存储命令（把NOVRAM数据调入到EEPROM），X24C45也不会完成该功能，即EEPROM中的数据不会产生改变。根据这一特性，可以判断X24C45是否为初上电状态，从而确定EEPROM和NOVRAM两个存储区域内哪一个为正确数据。在实际应用中，可在CPU的RESET事件中，先执行X24C45的存储命令，若系统处于初上电状态时，NOVRAM数据为错误的，但X24C45却没有"写功能"，EEPROM数据仍能保留其正确性；若系统为非初上电状态，这时的NOVRAM数据是正确的，执行了存储命令后，把NOVRAM正确数据保存到EEPROM中，然后再置X24C45的"使能读写"命令，最后把EEPROM数据调入到NOVRAM中。

数据的恢复可通过判断"标志位"来实现。如"标志位"非0，可将备份的数据保存到相应的地址单元，实现数据的恢复。处理流程见图2。

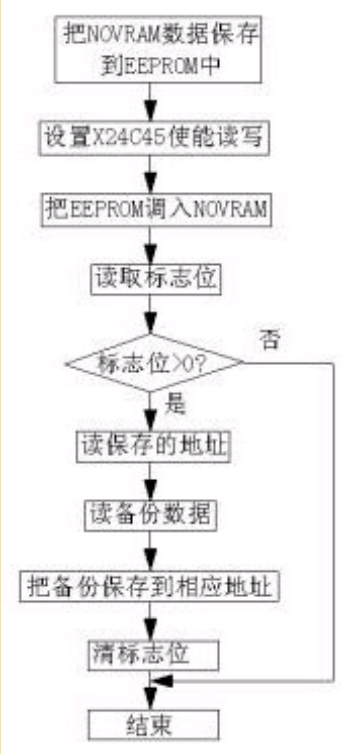

图2 初始化数据处理流程

3 系统实现

在LonWorks节点中，CPU是Neuron Chip(MC143150)，X24C45硬件原理图见图3。

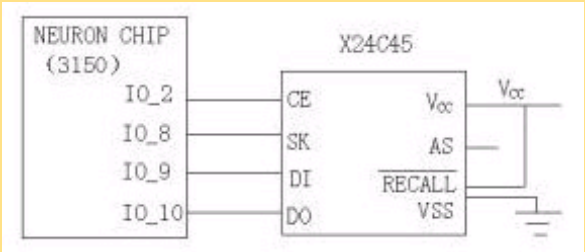

图3 硬件接线图

以下是实际的编程代码：

```
IO_8 neurowire master select (IO_2) io_ram;
IO_2 output bit io_2=1;
unsigned int address,status;
unsigned long data_back,data_sign;
```

```
void write(unsigned long data,unsigned int address);
unsigned long read(unsigned int address);
when(reset)//RESET事件处理过程
{ status=0x81;
io_out(io_ram,&status,8);  //NOVRAM存储到EEPROM
delay(80); //延迟2s
status=0x85;
io_out(io_ram,&status,8);  //EEPROM 数据到RAM
status=0x84;
io_out(io_ram,&status,8);  //使能读写
status=0x82;
io_out(io_ram,&status,8);  //使自动存储功能
address=read(0xc); //读地址
data_back=read(0xd):/读备份数据
data_sign=read(0xe); //读标志
//以下是数据恢复
if (data_sign>0)//标志位大于0
{ write(data_back,address); //把备份的数据写入相应地址单元
write(0,0xe); //清标志 }}
when(写)//存储操作
{ write(data,0xd); //备份数据
write(address,0xc); //保存地址
write(1,0xe); //置标志
write(data,address); 保存数据
write(0,0xe); //清标志}
```

返回上页