# 嵌入式主板 EM9160 精简 ISA 总线硬件中断的使用方法

英创公司

2008 年 11 月

英创公司新近推出的嵌入式主板 EM9160 是一款专门面向工业智能设备的高性价比 ARM9+WinCE 平台，该工控主板不仅配置了常规的标准通讯接口，如以太、串口、USB，而且还配置了当前智能控制设备中常用的 SPI、$I^2C$ 以及脉宽调制输出 PWM，以及精简 ISA 总线和外部中断，以方便客户做专用电路的扩展。本文主要介绍 EM9160 的外部中断使用上的特色，以帮助客户能快速完成自己的产品开发。

EM9160 的精简 ISA 扩展总线，包括了 2 路独立的外部硬件中断输入 ISA_IRQ1 和 ISA_IRQ2，平时输入电平应为低，当外部设备产生中断时，ISA_IRQ1 或 ISA_IRQ2 变高，其上升沿将触发中断，在系统对中断进行响应前，中断请求信号需保持为高。当应用程序对外设进行相应的中断响应处理后，ISA_IRQ1 或 ISA_IRQ2 应返回常规的低电平状态。

在 WinCE 下对于中断的处理是一个相对比较复杂的过程，当一个硬件中断发生时，首先是系统内核对中断进行诊别，再启动相应中断服务例程来处理这个中断，在具体实现中断处理的过程中，需要调用系统提供的中断相关的函数，就涉及到内核函数的调用。而用户在使用 eVC 等工具软件进行应用程序开发时，是不能直接调用这些内核函数的。针对这一情况，为了方便客户对于外部中断的使用，英创公司设计完成了在内核中直接加载外部中断的驱动程序，一旦中断事件发生，驱动程序的中断线程将响应硬件中断同时产生一个事件，以通知上层的用户模式线程。按照这种方式导出一个定制的用户接口（共 4 个函数），用户只需要在应用软件中直接调用这些定制的接口函数即可方便地实现对于中断的应用处理，下面就介绍这些相关的接口函数。

（1）HANDLE InstallExternIRQ( DWORD dwIRQNo );

功能描述：安装启动 ISA 总线外部中断。

输入参数 dwIRQNo： 需要启动安装的外部中断号，输入值为 1 或 2，分别对应 ISA 总线上的 IRQ1 和 IRQ2。

返回值 = NULL： 安装 ISA 总线外部中断失败。

!= NULL： 启动 ISA 总线外部中断返回的句柄。

（2）HANDLE GetExternIRQEvent( HANDLE hIRQ );

功能描述：获取 ISA 总线外部中断的中断事件。

输入参数 hIRQ： 调用函数 InstallExternIRQ( )所返回的外部中断句柄。

返回值 = NULL： 操作失败。

!= NULL：ISA 总线外部中断的中断事件。

（3）BOOL EnableExternIRQ( HANDLE hIRQ );

功能描述：使能 ISA 总线外部中断，允许下一次的中断。

输入参数 hIRQ：函数 InstallExternIRQ( )所返回的外部中断句柄。

返回值 = TRUE：操作成功。

= FASLE：操作失败。

（4）BOOL UninstallExternIRQ( HANDLE hIRQ );

功能描述：卸载关闭 ISA 总线外部中断。

输入参数 hIRQ：函数 InstallExternIRQ( )所返回的外部中断句柄。

返回值 = TRUE：操作成功。

= FASLE：操作失败。

这四个函数定义在 IRQ_API.h 文件下，相应的 IRQ_API.LIB 已经直接打包在 SDK 中，用户直接安装嵌入式工控主板 EM9160 的 SDK 文件即可。

作为应用程序来说，可以通过函数 InstallExternIRQ( … )来启动外部中断，并通过 GetExternIRQEvent( … )获取中断事件的句柄。应用程序应创建一个处理对应的外部中断处理线程，该线程等待中断事件来触发，一般用 WaitForSingleObject( )来等待事件被触发，当硬件中断发生时，用户处理线程就可以完成必要的 I/O 操作来采集数据或处理数据了，再调用函数 EnableExternIRQ( )再次开启硬件中断。

典型的应用程序中断处理线程如下：

```
DWORD WINAPI CEM9160_IRQ::IRQThreadFunc(LPVOID lparam)
{
    CEM9160_IRQ *lpIRQ = (CEM9160_IRQ*)lparam;
    for( ; ; )
    {
        WaitForSingleObject( lpIRQ->hIRQEvent, INFINITE ); // 等待硬件中断事件
```

```
            ExIRQHandler( );                    // 硬件中断事件处理，用户可在此函数
                                                // 中添加中断处理代码。
            EnableExternIRQ( lpIRQ->hIRQ );     // 再次开启硬件中断
        }
    }
```

需要注意的是，当具体的中断处理函数 ExIRQHandler( )执行完毕时，硬件中断请求输入 ISA_IRQ1 或 ISA_IRQ2 的电平已回到低电平状态，并确保在系统再次使能外部中断前一直保持为低，即函数 EnableExternIRQ( lpIRQ->hIRQ )完成前一直保持为低，以防止中断嵌套。