

# PCI-E 设备模块设计及其 WDF 驱动开发

张琳琳 张 涌

(中国科学院上海技术物理研究所, 上海 200083)

**摘要** 针对红外图像采集系统中的 PCI-E 设备模块开发, 重点介绍了其作为主设备时的工作情况及其优势。并介绍了基于 WDF 模型的 PCI-E 主设备模式的驱动开发过程。列举了测试结果, 结果表明主设备方式 CPU 占用率更低, 数据传输速率更高, 并验证了驱动程序的稳定工作。

**关键词** PCI-E 总线 主设备方式 WDF 驱动 DMA

**中图分类号** TN219; **文献标志码** A

红外探测技术的飞速发展使得红外图像采集系统需要更高速的传输方式。PCI-E 总线是新一代的总线技术, 它采取的是点到点的串行连接方式, 每个设备都有自己的专用连接, 不需要向总线请求带宽, 可以达到很高的传输频率, 正在逐步取代 PCI 总线。PCI-E 总线以其优异的性能被广泛应用到各领域, 如数据采集系统。故在本设计中采用了 PCI-E 总线, 并且, 随着实时性需求的增加, 采用传统的从设备方式工作容易引起数据丢失, 故主设备工作方式的重要性日渐凸显。

WDF (Windows Driver Foundation) 是微软提出的下一代全新的驱动程序模型, 是在 WDM 的基础上发展而来的。提供了更高层次抽象的高度灵活、可扩展、可诊断的驱动程序框架, 提供面向对象和事件驱动的驱动开发框架, 将驱动程序与系统内核操作隔离开来, 大大降低了驱动开发的复杂度和风险性<sup>[1]</sup>。

本文介绍的是基于 WDF 模型的 PCIE 主设备模式驱动程序的开发过程。

## 1 PCI-E 设备框架

本文的驱动程序是基于自行研发的 PCI-E 设备, 该设备为在 Xilinx 公司的 ml507 平台 (采用 Virtex-5 FXT 系列芯片) 上用硬件描述语言实现。通

过使用 Xilinx 公司提供的 PCI-E LogiCORE IP Endpoint Block Plus 核, 可方便地实现符合 PCI-E 总线协议的数据传输。

本文的 PCI-E 设备可于两种模式下工作: 从设备模式及主设备模式。

### 1.1 从设备模式

从设备模式下, 由主机 (PC 机) 发起通讯。主机通过应用程序发起命令, 执行对 PCI-E 设备的控制、读、写操作。

PCI-E 设备只有在收到主机发送的命令时才执行相应的操作, 无法主动向主机发起通信。

从模式下, 设备和计算机之间的数据传输, 可以通过程序查询方式和中断方式进行。这两种方式都是在 CPU 控制下, 通过 CPU 执行指令来完成的。数据传送方向为外设——CPU——内存。

程序查询方式中, 当 CPU 需要数据时, 需要反复测试外设状态, 当设备准备完成之后才能进行数据传输, 否则处于等待状态。

中断方式中, CPU 在每次接收到外设发来的中断后进行数据传输, 每一次的数据传输, CPU 都需要进入中断服务子程序, 断点保护、现场保护、现场恢复、返回主程序等系列操作, 程序开销很大, 并且当数据传输速率要求很高时, 中断的频率也相应提高, 对 CPU 的中断响应时间要求很高, 如果来不及响应, 就会丢失数据, 引起传输错误。

故从模式一般只用于简单的设备控制, 而无法满足高频率、大数据量的数据传输, 更无法满足日益发展的实时数据传输系统。

2011年10月18日收到

第一作者简介: 张琳琳 (1988—), 女, 福建莆田人, 硕士研究生, 研究方向: 信号与信息处理。E-mail: linlinzy@mail.ustc.edu.cn.

## 1.2 主设备模式

为了解决从模式的问题,PCI-E 设备可工作于主设备模式。主设备模式下,PCI-E 设备可主动发起通信,并且整个传输过程无需 CPU 的干预,直接在设备和计算机的内存之间进行数据传输。以自行设计的图像采集系统为例,主设备模式的具体工作流程如下:

- (1) PCI-E 设备收到图像数据,存入缓存区;
- (2) 当缓存区的数据到达一定量,以一帧为例,设备就向计算机发送数据;
- (3) 当传送完一帧时,设备发送一个中断信号通知计算机;
- (4) 计算机可以对接收到的数据进行相应的处理。

主设备模式可以通过两种方式实现,一种是通过计算机内的 DMA 控制器 DMAC 进行 DMA 传输。这种方式需要 DMAC 向 CPU 发送总线请求信号并等待 CPU 同意总线请求信号。DMA 传输期间 CPU 让出 PCI-E 总线的控制权,直到传输结束后,DMAC 撤销总线请求信号,CPU 收回总线控制权。

本文在研究中使用的是另一种方式,即在设计 PCI-E 设备时,直接在 FPGA 内部实现主设备控制模块。这种方式依靠硬件描述语言实现整个主设备工作过程,不需要计算机内部的 DMA 控制器的参与。

采取 FPGA 内部集成主设备模块的主设备工作方式如图 1 所示。

采用这种主设备方式工作可以提高整个系统的性能,最直接体现在以下几点。

### 1.2.1 节省设备存储空间

数据采集系统中,采集到的数据是先存放在设备的缓冲区内等待传输的。这个缓冲区可以是 FPGA 内部的 Block RAM 或者是 SDRAM。FPGA 内部的 Block RAM 资源非常珍贵,不适合存放太大的数据量,较大的数据存储就需要用到 SDRAM,但是比起计算机内的存储空间依然显得较为紧张。从设备方式由于采集到数据后需要等待计算机发起传输,故需要较大的存储空间,而主设备模式下,可以在计算机内开辟相对大得多的存储空间用于存放数据,等待计算机空闲时再进行数据处理,这样可以节省 PCI-E 设备上的存储空间,甚至可以不需要

SDRAM,可以降低系统的复杂度,降低设计成本。

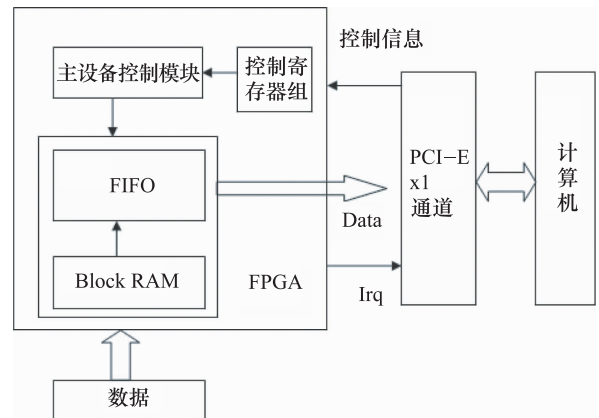


图 1 集成主设备模块的 PCI-E 设备结构框图

### 1.2.2 提高数据传输速度,防止丢帧

从设备模式下,需要等待计算机发起数据传输,在这段过程,CPU 就无法顾及其它工作,即使采取从模式下的中断等待方式,在高速数据传输过程中 CPU 需要不停地进行中断处理,不仅开销很大,也会因为来不及响应部分中断而丢失数据,这对数据采集系统来说是非常致命的问题。

采取主设备模式,设备向计算机发起数据传输将不再需要 CPU 的介入,CPU 可以继续处理其它任务,等待空闲时再进行数据处理即可。并且杜绝了数据丢失现象。

### 1.2.3 方便设备间通信

主设备模式下,总线上的多个 PCI-E 设备间的通信变得更加方便。各个设备间的通信不再需要通过 CPU 控制,而可以实现设备——设备的通信。

1.2.4 在 FPGA 内部实现主设备控制模块不需要等待计算机使能 DMA 控制器。在传输帧间,计算机可以随时向 PCI-E 设备发送控制信息或数据,而不需要等待整个数据传输结束并回收总线控制权之后。使得整个传输过程更加简单和方便。

## 2 基于 WDF 的驱动编写

### 2.1 WDF 驱动程序框架

WDF 是微软对 WDM 驱动架构改进后提出的一种驱动框架结构。

WDF 驱动程序包括两个类型,内核级(KMDF)和用户级(UMDF)。下文的介绍都是基于 KMDF 的。WDF 的框架由对象和事件回调例程构成。框

架中所有的事物都由对象表示,各种事件的处理都由事件回调例程完成。这样使得整个驱动的层次更加清晰<sup>[1]</sup>。

驱动程序所要做的包括:对需要环境变量结构的对象定义其环境变量结构并初始化对象的属性,创建对象,定义对应的回调例程。

## 2.2 PCI-E 设备驱动开发

PCI-E 设备驱动结构框架如图 2 所示。

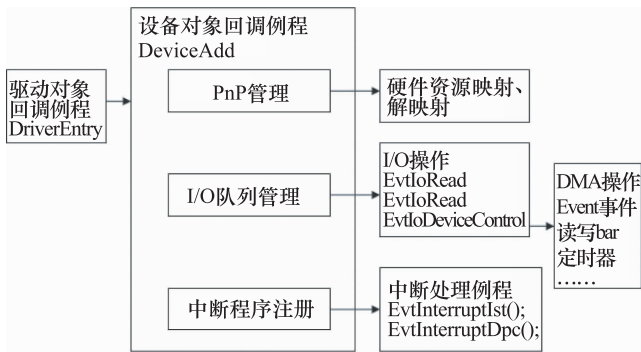


图2 基于WDF的PCI-E设备驱动结构框架

主设备模式下,PCI-E 设备驱动的开发主要处理四类操作:硬件空间访问、中断处理、DMA 方式、Event 事件。

### 2.2.1 硬件空间的访问

PCI-E 的驱动程序中需要对该设备内存地址空间进行内存映射才可访问。

故需在即插即用基本例程中定义内存映射及解映射<sup>[1]</sup>:

EvtDevicePrepareHardware 中调用 MmMapIoSpace 函数将物理地址转换成系统内核模式地址。

EvtDeviceReleaseHardware 中调用对应的 MmUnmapIoSpace 解除物理地址与系统内核模式地址的关联。

内存映射之后就可以在需要时调用存储器访问语句来读写基址空间内设置的寄存器: WRITE\_REGISTER\_XXX(); //往寄存器内写;

READ\_REGISTER\_XXX(); //读取寄存器的值。

### 2.2.2 中断处理

设备向计算机发送完一帧数据后向计算机发送一个中断,以通知计算机可以处理这帧数据。

中断服务例程分成两部分,上部分中断处理例

程 EvtInterruptIsr 主要工作为在收到中断后通过读取 PCIE 的中断状态寄存器来判断是否来自该设备的中断,如是,对中断控制寄存器写值以撤销中断,并启动下部分的延迟中断处理例程 EvtInterruptDpc, EvtInterruptDpc 例程可以在空闲时实现更多更复杂的操作,这样可以保证一次中断不会占用太长时间。因为中断处理例程的运行在 DIRQL 级别上,所以需要尽量短的执行时间,而延迟中断处理例程运行在 DISPATCH\_LEVEL 级别上,可以处理大部分的中断处理工作。

本文的延迟中断处理例程 EvtInterruptDpc 主要工作为唤起一个事件,通知应用程序对已收到的数据进行处理。

### 2.2.3 DMA 方式

由于本文采取的是 FPGA 内部集成主设备控制模块,故驱动无需对计算机内的 DMA 控制器进行操作。驱动内只需要申请一个内存空间用于存放数据,并将其物理地址告诉设备。因为应用程序需要读取这部分空间内的数据以进行下一步的数据处理,故需要对该空间进行地址映射,否则,虽不影响整个数据传输,但是无法进行后续处理<sup>[2,3]</sup>。

申请 DMA 空间并进行地址映射的过程如下:

```
//申请一个 length 字节大小的空间,并返回其内核模式虚拟地址
```

```
Virtual_Address = MmAllocateContiguousMemory ( length, maxAddress );
```

```
//将虚拟地址转换为物理地址,用于写入 PCIE 设备的目标地址寄存器,使得设备获得数据传输的目标地址。
```

```
Physical_Address = MmGetPhysicalAddress ( Virtual_Address );
```

```
//分配一个 MDL 来描述这段空间。
```

```
mdl = IoAllocateMdl ( Virtual_Address, length, FALSE, FALSE, NULL );
```

```
//将 MDL 描述的物理页面集合映射到系统地址空间
```

```
MmBuildMdlForNonPagedPool ( mdl );
```

```
//在内存中锁定 MDL 描述的页
```

```
MmProbeAndLockPages ( mdl, KernelMode, ( LOCK_OPERATION ) ( IoWriteAccess | IoReadAccess | IoModifyAccess ) );
```

```
//完成映射,并获取用户模式虚拟地址
```

```
User_Address = MmMapLockedPagesSpecifyCache ( mdl, UserMode, MmNonCached, NULL, FALSE, NormalPagePriority );
```

```
//将物理地址写入 PCIE 设备寄存器。
```

```
WRITE_REGISTER_ULONGLONG ( ( PULONG ) ( pREG + 0x04 ), Physical_Address );
```

相对应地,在退出 DMA 模式时需要对申请到

的地址进行解映射和释放内存操作:

```
MmUnmapLockedPages( pDeviceContext-> m_memaddress. useraddress, pDeviceContext-> mdl );
MmUnlockPages( pDeviceContext-> mdl );
IoFreeMdl( pDeviceContext-> mdl );
MmFreeContiguousMemory( pDeviceContext-> m_memaddress. systemaddress );
```

#### 2.2.4 事件 Event

事件是驱动用以唤起应用程序进行数据处理等操作的方法。本文中在延迟中断处理例程中设置事件的发生<sup>[4]</sup>。

应用程序创建一个通知事件,并通过 DeviceIoControl 的输入缓冲区传递给驱动,驱动内创建内核事件,当事件发生时,驱动会通知应用程序,发起相应的操作。当退出程序时需要撤销内核事件。具体如下:

应用程序创建事件:

```
m_hEvent = CreateEvent(NULL, FALSE, FALSE, NULL); //创建自动重置事件
```

应用程序等待事件发生:

```
WaitForSingleObject(m_hEvent, 0) == WAIT_OBJECT_0
```

驱动创建内核事件:

```
ObReferenceObjectByHandle(...);
```

本文中在中断产生时设置一个事件:

```
KeSetEvent(pDeviceContext-> Event, 0, FALSE); //设置事件,
```

将触发应用程序读取数据

驱动内撤销内核事件:

```
ObDereferenceObject(...);
```

### 3 实验结果与分析

硬件平台: xilinx 公司的 ML507 开发板(FPGA: XC5VFX70T-1FFG1136), PC 机(WinXP)

在 ML507 内实现 PCI-E 总线主设备模式数据传输以及从设备模式数据传输,并在 PC 机上采用 WDF 框架编写相应的驱动及应用程序。实验结果

如表 1。

表 1 实验结果

	主设备方式	从设备方式
CPU 占用率	16% ~ 18%	43% ~ 48%
数据传输速率	135MB/s	63MB/s

实验结果鲜明地表现了主设备方式相对从设备方式在传输速率和 CPU 占用率上的优势。

### 4 结束语

PCI-E 设备工作在主设备模式下时可以大大提高整个系统的工作性能,降低 CPU 的占用率,提高数据传输速度,并防止数据丢失现象的产生。同时采用 FPGA 内部集成主设备控制模块的方式可以简化系统,进一步提高传输速度。综上,这样的设计使得整个系统在红外图像采集系统中的性能优于传统方式。

本文基于 WDF 框架编写了 PCI-E 设备驱动,特别是主设备模式下的相应操作。经测试,该驱动工作良好,对整个系统的工作及研究起到非常重要的作用。并且该驱动架构清晰,可移植性和扩展型也较强。

#### 参 考 文 献

- 1 武安河. Windows 设备驱动程序 WDF 开发. 北京: 电子工业出版社, 2009
- 2 杨阿锋. 基于 WDF 的 PCIe 接口高速数据传输卡的驱动程序开发. 硕士论文. 长沙: 国防科学技术大学, 2008
- 3 黎绍秀, 卫红, 兰春嘉. PCI-E 图像采集系统的 WDF 驱动程序设计. 科学技术与工程, 2011; 11(16): 2834—2837
- 4 Orwick P, Smith G. Developing drivers with the microsoft windows driver foundation. Microsoft Press, 2007

## PCI-E Module Design and its WDF-based Driver Development

ZHANG Lin-lin, ZHANG Yong

(Shanghai Institute of Technical Physics, Chinese Academy of Sciences, Shanghai 200083, P. R. China)

**[Abstract]** The development of the PCI-E module for infrared image acquisition system is described. Advantages of PCI-E device is described when working as master device, and introduced the PCI-E device driver development based on WDF. The testing results show that the master mode has lower CPU utilization and more efficient data transfer, and prove the driver works stably.

**[Key words]** PCI-E bus master-mode WDF driver DMA