被 FPGA 驱动成高电平[22]。当 CPLD 检测到 DONE 拉高之后，它控制 Flash RAM 的控制信号，让 Flash RAM 输出高阻态，同时将外部晶振时钟直接输出作为 FPGA 的全局工作时钟。如果在配置的过程中出错，当PROGRAM被拉低时，CPLD 将使 Flash RAM 的地址计数器恢复为零，从而重新开始一轮配置过程。

## 5.3.2 FPGA 片内数据接收和处理模块的设计

FPGA 片内程序是此系统数据处理的主要模块，不仅实现接收外部的数据并将它们送入外部 SDRAM，而且可以提供一个接口给用户的算法处理模块连接，同时可以实现在数据处理完之后，再将数据从 SDRAM 中读取出来送给外部连接的电路。片内程序包括了三个主要电路模块，FIFO 模块、SDRAM 控制器和主控制模块，还有一个数据与信号选择模块。由于其中的数据与信号选择模块比较简单，它只实现数据和信号的选择，所以下面就对前三个主要模块分别进行详细介绍，然后再对整体的连接框架进行介绍[23]。

### 5.3.2.1 FIFO 模块的实现

在本系统设计中，数据信息的传递和交互是非常频繁的，而且存在着不同工作速率的器件，因此，在外部数据与 FPGA 通信之间都加入 FIFO，这样可以使数据进行一定的缓冲[24]。在 ISE 软件里面集成有许多 FIFO 的 IP 核资源，因此我们就直接采用里面的 IP 核来设计。由于系统中的每一个 USB 模块的数据包的大小都是 64 字节，而且为了数据传输的正确性，我们选择了一个具有 64 位深度和 8 位数据宽度的同步 IP 核进行设计，这样可以保证在每个数据包被取走之后另一个数据包才可以写入。

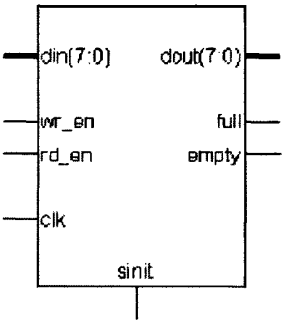


图 5-11 FIFO 的综合的电路框图

此FIFO的综合的电路框图如图5-11所示：

可以看出，它带有 wr_en 写使能引脚和 rd_en 读使能引脚，但只有一个 clk 时钟引脚。因此在写有效的时候，clk 就做为数据写入控制信号，在每个时钟上升沿写入在 din 引脚上面的数据。当读有效的时候，clk 作为数据读出控制信号，同样在每个时钟上升沿从 dout 引脚送出数据。当数据写满 FIFO 的时候，full 引脚就输出高电平。当 FIFO 里面的数据为空是，empty 引脚就输出高电平[25]。

### 5.3.2.2 SDRAM 控制器的设计

SDRAM 的控制逻辑比较麻烦，对时序要求也十分严格，使用很不方便，这就要求有一个专门的控制器，使得在电路中可以对 SDRAM 操作变得简单。因此，在

此系统中利用 FPGA 设计了一个 SDRAM 控制器。其综合出来的电路图及其与外部 SDRAM 连接的框图如图 5-12 所示：

此控制器左边显示的管脚就是与外部通信的引脚，右侧是与 SDRAM 通信的引脚。外部电路只要对此控制器输入地址、数据以及读/写信号就可以通过控制器对 SDRAM 进行读/写操作，这样就使得对 SDRAM 的操作变的很容易[26]。同时，在控制器里面还输出了一个 done 引脚，该引脚在每次数据读/写完成后就输出高电平，可以提供给外部电路进行更好的控制。

根据控制信号和地址输入，SDRAM 包括多种输入命令：模式寄存器设置命令、激活命令、预充电命令、写命令、读命令、自动刷新命令与空闲命令。根据输入命令，SDRAM 状态在内部状态间转移。内部状态包括：模式寄存器设置状态、激活状态、预充电状态、写状态、读状态、自动刷新状态和空闲状态[27]。本系统只考虑了以下几点：器件仅在上电后进行一次初始化，在初始化中预充电所有的内存区、进行 8 个周期的自动刷新，之后设置工作模式，处于正常工作状态后就不

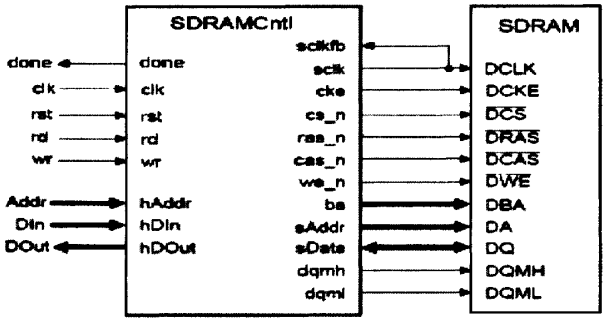


图 5-12 SDRAM 控制器与 SDRAM 连接的框图

能再改变工作模式；器件在空闲内处于连续的自动刷新状态；在有读/写要求时，先使用激活命令进入激活状态，然后再使用读/写命令进入读/写操作；在读/写操作要求完成或读/写满一行后，立即使用预充电命令进入预充电状态，以便进入下一次的读/写操作。

5.3.2.3 主控制器的设计

为了对 FPGA 内部的电路模块进行协调处理，本文编写了一个主控制器模块。主控制器实现了两个功能：当有外部数据输入的时候，可以将外部存入到 FIFO 的数据包读取出来，然后写入 SDRAM 中，同时根据写入的数据量通过地址发生器自动增加 SDRAM 的地址；当外部数据输入完成的时候，将 SDRAM 中的数据读取出来并存入 FIFO 中提供给外部电路进行读取数据包[28]。因此，它综合的电路如图 5-13 所示：

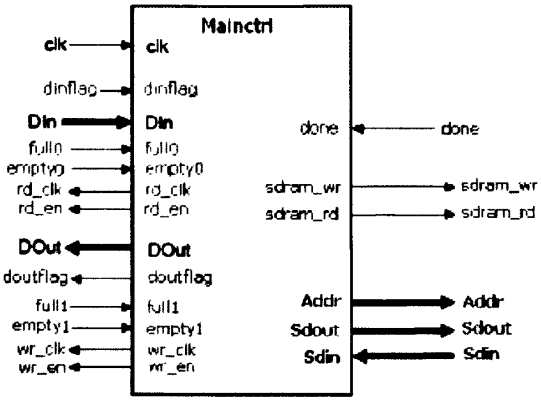


图 5-13 主控制器框图

它通过数据输入标志 dinflag 来判断外部是否有数据输入。如果该信号为高电平，则表示数据输入，此时主控制器通过对输入 FIFO 的 full0 信号进行检测，判断数据包是否已满。如果满就使能 rd_en 并发送读信号 rd_clk 把数据读出发送给 SDRAM 控制器。当检测到 FIFO 为空的信号即 empty0 为高时拉低 rd_en 信号并停止输出 rd_clk，使其处于高电平状态。当外部数据输入完毕，即 dinflag 为低电平的时候，主控制器就把数据通过 SDRAM 控制器把数据从 SDRAM 里面读出来，通过使能 wr_clk 以及发出 wr_clk 将数据发给 FIFO，同时使能 doutflag 信号来通知外部电路有数据输出[29]。当此 FIFO 的 full 输出高电平的时候主控制器停止写入数据，如果外部电路把数据读完即此 FIFO 的 empty 输出高电平的时候主控制器就重新开始写入数据。当全部数据发送完之后，控制器拉低 doutflag 信号以通知外部电路数据已经发送完毕。

5.3.2.4 FPGA 内部数据接收和处理模块的实现

结合上面介绍的几个模块和数据选择模块，我们设计的 FPGA 的内部模块如图 5-14 所示，图中虚线部分就是数据接收和处理的主要模块，它由主控制器、SDRAM 控制器、FIFO 和数据选择器组成。左边的框图即为它与外部电路的接口和用户算法处理模块的连接方式[30]。

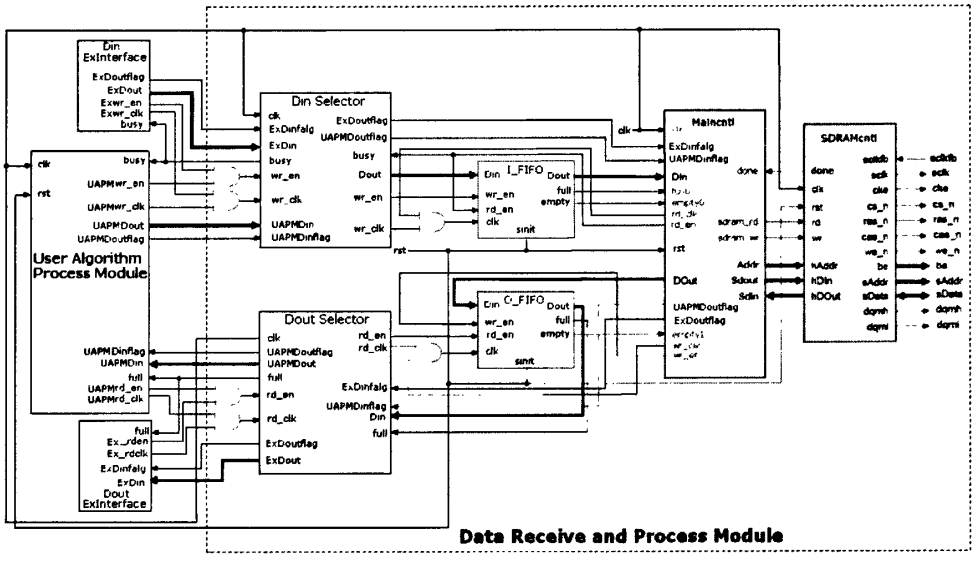


图 5-14 FPGA 的内部模块

在内部处理模块对数据进行处理之前，需要先将外部数据存入 SDRAM 中。当外部有数据输入的时候，它先通过拉高 ExDinflag 通知数据选择器外部数据输入，则数据输入选择器将外部电路的数据信号和控制信号与主控制器的输入数据信号的输入控制信号连接。然后外部电路先把 64 位的数据包写入 I_FIFO 中，在 I_FIFO 写满的时候通过 full 信号通知主控制器来读取数据。在主控制器读取 I_FIFO 的时候同时把 rd_en 信号输出给外部接口，通知外部接口电路主控制器在读取数据。当读取完数据时，rd_en 就拉低，此时外部接口电路就可以继续写入数据。当外部数据全部写完的时候，ExDinflag 信号被拉低以通知主控制器数据已经写完，此时数据选择器就选择用户算法处理模块的输出数据信号和输出控制信号与主控制器的输入数据信号的输入控制信号连接。然后主控制器接下来将数据发送给用户算法处理模块，它就将数据从 SDRAM 中读取出来送入 O_FIFO 中，同时拉高 UAPMDoutflag 信号[26]。当数据输出选择器查询到该信号时，就选择将主控制器的数据和其它信号与用户算法处理模块相连，则该模块通过 O_FIFO 写满一个数据包时产生的 full 信号来读取数据并进行处理。而 O_FIFO 会在数据包被读取完之后通过 empty 信号来通知主控制器继续发送数据。

当数据经过算法处理之后，用户算法处理模块拉高 UAPMDoutflag 信号通过数据选择器来通知主控制器开始有数据输入并把数据写入 I_FIFO 中。当数据满的时候，主控制器就开始来读取处理之后的数据并存入 SDRAM 中，而此时存入的地址将从零开始，即会覆盖相对应的未处理前的数据。当 SDRAM 中的数据被全部读出后，主控制器就主控的 UAPMDoutflag 信号拉低，以通知算法处理器数据已经处理完毕。则算法处理模块处理完最后一组数据之后就将模块的UAPMDoutflag 信号拉低，从而通知主控制器处理完的数据已经发送完毕，主控制器就从 SDRAM 中将数据读出，发送给 O_FIFO，同时拉高 ExDoutflag 信号[31]。当数据输出选择

器检测到该信号后，就将主控制器的数据输出信号和其他控制信号连接给外部电路接口。外部电路在查询到 O_FIFO 的 full 信号时，将数据取走直到数据发送完毕。当所有数据发完之后，主控制器拉低 ExDoutflag 通知外部电路数据已发送完成。

## 5.4 驱动程序设计与应用程序设计简要介绍

### 5.4.1 驱动程序软件设计简介

USB 设备驱动程序响应的是系统软件层的请求，而不能与设备硬件直接通信。当数据采集系统处于联机工作状态，和上位机进行通信时，D12 是作为 USB 从机设备，如果将其描述符配置为 HID 设备(人机接口设备)。当它与上位机相连时，上位机操作系统检测到 HID 设备时，会自动安装驱动程序。但作为 HID 设备，上位机不能和系统进行交互的数据传输，若要实现交互的数据传输，就必须编写驱动程序，使 PC 上位机能识别 D12 从设备，并与之进行通信。上位机上的驱动程序实际上操作系统与硬件层之间的纽带，将操作系统的要求翻译给硬件抽象层，与底层 USB 从设备的硬件基本上无太大关系[32]。关于驱动程序的开发这里不再详述。

### 5.4.2 应用软件设计简介

应用软件由两部分组成:动态链接库和应用程序。动态链接库负责与内核态的 USB 功能驱动程序通信并接收应用程序的各种操作请求；应用程序提供给用户一个与系统进行交互的界面，用户通过这个界面可以对输出用电源设备和数据采集进行控制，并且负责对所采集的数据进行实时显示、分析和存盘等操作。软件设计采用 Visual C++进行编写。