文章编号:1671-4598(2024)11-0308-07 DOI:10.16526/j.cnki.11-4762/tp.2024.11.043 中图分类号:TP274.2 文献标识码:A

基于 FPGA 的多串口中断管理器设计

熊荐辕[1,2], 王保成[1]
(1. 中国科学院 空天信息创新研究院, 北京 100094; 2. 中国科学院大学 航空宇航学院, 北京 100049)

摘要: 为了解决飞控计算机与多个不同周期和数据帧大小的外部设备串口合理传输数据的问题, 在 FPGA 中设计了一款多串口中断管理器, 其包含有寄存器控制模块、中断控制模块和串口控制模块; 为每一个外部设备设计了发送和接收的 FIFO 模块来缓存串口数据, 同时根据不同外部设备的数据收发需求, 设计了 5 种合理的中断类型来触发飞控计算机的中断, 包括接收数据就绪中断、发送 FIFO 空中断、数据状态错误中断接收超时中断和发送超时中断; 通过总线与该多串口中断管理器通信, 进一步实现与不同类型外部设备的合理交互; 通过对系统进行仿真和实际测试, 表明该系统能实现 16 路串口的 5 种中断管理功能。

关键词: 飞控计算机; FPGA; 中断管理器; 中断类型; 多路串口

Design of Multichannel Serial Port Interrupt Manager Based on FPGA

XIONG Jianyuan[1,2], WANG Baocheng[1]
(1. Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China;
2. University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: To address the challenge of efficient data transmission between the flight control computer and multiple external devices with varying data frame sizes and time cycles, a multi-serial port interrupt manager is designed using a Field Programmable Gate Array (FPGA). This manager comprises a register control module, an interrupt control module, and a serial port control module. For each external device, First in, First out (FIFO) modules are designed for both transmission and reception to buffer serial port data. Additionally, according to the data transmission and reception requirements of different external devices, five types of reasonable interrupts are designed to trigger the interrupts in the flight control computer, including the interrupt for data reception readiness, FIFO empty interrupt for transmission, data status error interrupt, reception timeout interrupt, and transmission timeout interrupt. Through bus communication with multi-serial interrupt manager, it further realizes reasonable interaction with different types of external devices. Simulation and practical testing show that the system can achieve five types of interrupt management with sixteen serial ports.

Keywords: flight control computer; FPGA; interrupt manager; multichannel serial port

0 引言

飞控计算机作为飞行器的重要组成部分, 主要起着姿态稳定与控制、导航与制导控制、自主飞行控制、起飞与着陆控制等作用[1]。

对飞控计算机而言, 需要能够与飞行器上的多类外部设备通信, 以高空飞艇的飞控计算机为例, 需要同时采集包括惯性测量单元、温度传感器、压差传感器、地面遥控指令在内的多种传感器数据, 同时需要向地面发送遥测指令和向电机发送控制指令等[2]。因此, 对飞控计算机而言, 如何实现对多个外部设备数据收发的合理管理, 成为了目前的一个热门研究方向。在目前的飞控计算机中, 为了发挥不同处理器的优势, 常采用多种处理器联合工作的方式, 通过合理的任务分配来弥补单一处理器的不足[3]。目前应用较广的多处理器组合有 ARM (Advanced RISC Machines) + DSP[4-5]、ARM + FPGA (Field Programmable Gate Array)[6-7]、DSP+FPGA[8-9]等。ARM 一般作为飞控计算机的主处理器, 需要完成飞行控制、姿态解算等任务[10]。由于 ARM 接口资源不足、并行计算能力较差, 为了使得飞控计算机能够与大量的外部设备进行数据交互, 通常使用具有良好扩展性的 FPGA 作为协处理器来实现接口扩展和并行计算[11-13]。为了实现 ARM 与 FPGA 间的数据传输, Peng Liu 等[14]介绍了一种 ARM 通过 FSMC (Flexible static memory controller) 总线协议片选连接到 FPGA 上的方法, 并在 FPGA 内部设计了 FIFO (First in, First out) 模块用来缓存交互的数据, ARM 接收到外部数据后通过 FSMC 总线将数据储存到 FIFO 里, 需要使用时再读取出来, 经过仿真和验证能有效实现数据的传输。Chunxiang Zhu 等[15]设计的电机主控单元, 结合了 ARM 优秀的控制性能和 FPGA 数据处理速度方面的优势, 通过

收稿日期:2023-11-18; **修回日期**:2024-01-11。
基金项目:国家自然基金重大仪器项目(52227811)。
作者简介:熊荐辕(2000-),男,硕士研究生。
通讯作者:王保成(1974-),男,副研究员,硕士生导师。
引用格式:熊荐辕,王保成.基于 FPGA 的多串口中断管理器设计[J].计算机测量与控制,2024,32(11):308-314.

FSMC 总线通信和数据乒乓算法，较好的满足了电机瞬态测试和动态加载的要求。

为了能与较多的外设进行交互，通常使用 FPGA 来扩展串口的数量。HU zhe 等[16] 基于 FPGA 实现了一款 UART（Universal Asynchronous Receiver/Transmitter）基本收发模块，将串行的数据收发经过 FIFO 转成总线发送的形式，提升了数据的传输效率。Sardi Irfansyah[17] 在实现 UART 基本收发模块的基础上，新增了 FIFO 来缓存数据，避免数据的丢失。Bibin M C 等[18] 在 FPGA 实现串口正常收发的基础上，增加了状态寄存器来检测数据传输过程中产生的错误。Ying Wang 等[19] 基于 FPGA 设计了一款串口控制器，能对串口的波特率进行设置并阅读当前的状态。Shouqian Yu 等[20] 基于异步 FIFO 设计了一款多通道 UART 收发器，能在波特率不同的情况下进行串口数据的收发。

可以看出，以上设计均使用 FPGA 来扩展可使用的串口的数量，同时使用了 FIFO 来暂缓数据或完成同步等，还包含一些对串口的设置，如波特率的测试，对串口状态的检测等，但均没有为整个系统设置合理的管理器，因此系统只能周期性收发数据，难以对各串口收发的数据实现高效的管理。

为了方便管理，M. DE. ALBA 等[21] 基于 PXA270 设计了一款中断控制器，能够识别来自不同设备的中断源，提高了中断控制的效率。但该设计也仅能判断某设备产生了中断，难以对设备数据进行进一步的判断与后续处理。

由于不同外部设备传输数据的周期和大小不一致，同时会有一些突发性的数据量（地面人员的遥测指令、故障报警等），因此，为了提高飞控计算机的执行效率，需要在能与大量外设实现基本数据交互的基础上，针对不同的外设采取合适的数据收发方式。本文在 FPGA 中设计了一款多路串口中断管理器，可在实现串口数据收发的基础上，将不同外部设备的串口数据缓存到 FIFO 里，再通过不同类型的中断来触发，实现对多路外部设备数据收发的合理调配。

1　系统设计

系统的总体框图如图 1 所示，在 FPGA 内部实现了寄存器控制模块（REG _ CONTROL）、多路中断控制模块（INT _ CONTROL）和各串口各自的控制模块（UART _ CONTROL）。ARM 连接到寄存器控制模块来进一步完成与 FPGA 内部其他模块的交互。各串口控制模块连接到外部设备来完成数据的合理传输。同时使用多路中断控制模块加一个连接到 ARM 的中断引脚来实现对 ARM 的外部中断触发。

1.1　寄存器控制模块

寄存器控制模块直接通过 FSMC 总线与 ARM 相连，将 FPGA 以 NOR FLASH 的形式挂载到 ARM 上，引脚包括片选引脚 CS _ N，读数据使能引脚 OE _ N，写数据使能引脚 WE _ N，地址有效引脚 NADV，16 位地址数据复用总线 AD［15：0］。ARM 可通过 FSMC 协议生成指定地址

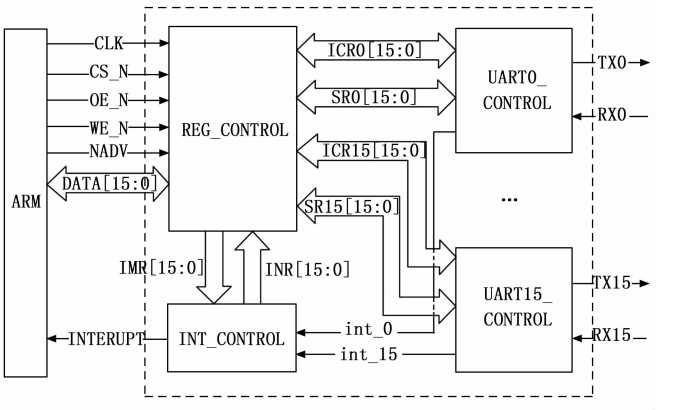


图 1　系统总体框图

的读写信号，复用模式下读写的时序如图 2 和图 3 所示，开始传输数据后 CS _ N 信号拉低，读数据的时候 OE _ N 信号拉低，写数据的时候 WE _ N 拉低，操作结束后拉高，NADV 作为地址有效位，当总线传输地址时拉低。

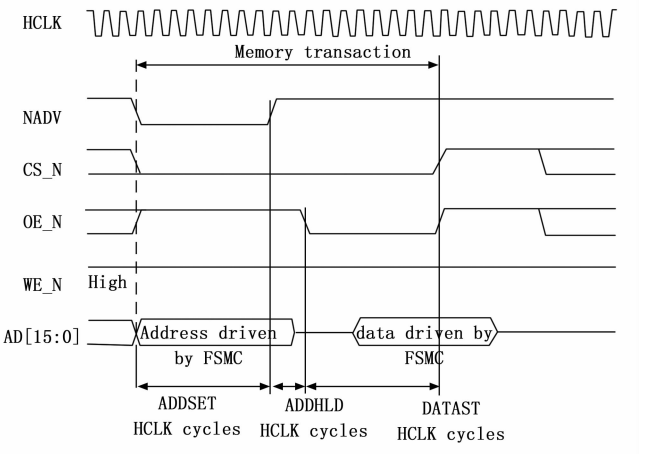


图 2　FSMC 读取时序图

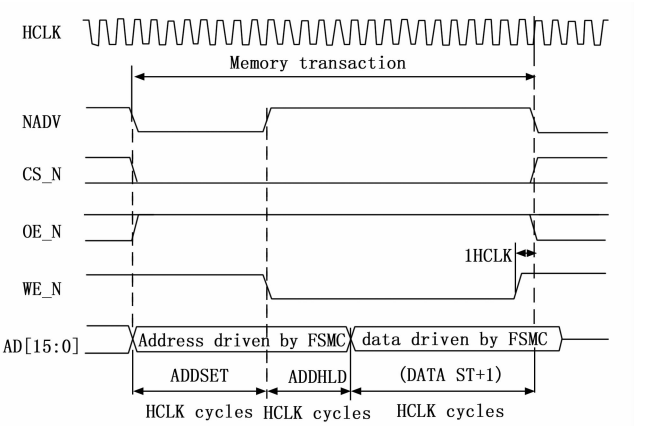


图 3　FSMC 写入时序图

寄存器控制模块中保存了系统中所有需要使用到的寄存器，由于 FSMC 复用数据宽度为 16 位，因此将所有寄存器宽度也配置为 16 位。该模块接收到 FSMC 读写信号后将

其转化地址和数据总线，通过地址总线寻址到需要进行操作的寄存器，并通过使能信号来决定是读还是写寄存器，通过对寄存器的配置进一步完成对其他模块的控制。

1.2 多路中断控制模块

在 FPGA 中利用中断控制模块（INT CONTROL）来实现对多路串口中断的管理，系统框图如图 4 所示（这里用两路串口做示例）。将 16 路串口产生的中断转化为一路 INTERRUPT，连接到 ARM 用来触发 ARM 的外部中断，INTn _ O（n 为序号）为各序号串口产生的中断信号。

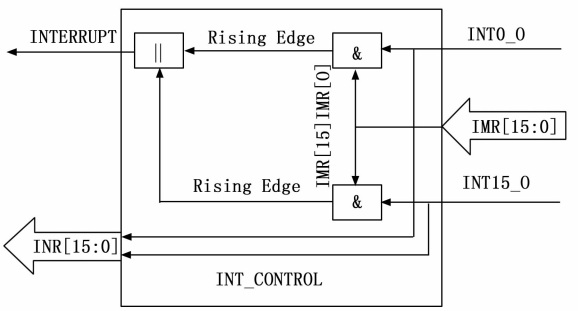


图 4 中断控制模块框图

IMR 为各串口模块中断使能寄存器，可通过配置 IMR 来使能各串口模块的中断。INR 为各串口模块中断触发寄存器，用来保存串口中断信号的电平状态。

当有一路串口模块产生中断后，其对应的 INTn _ O 会被拉高，同时将 INR 寄存器相应序号位置 1。若该串口中断已使能，则 INT _ CONTROL 模块会将该中断信号传输到 ARM，ARM 触发中断后通读取 INR 来判断产生中断的串口序号。由于 IMR 和 INR 均为 16 位，因此支持最多 16 位中断串口的使能与检测，若需对更多路串口进行管理，可进一步增加 IMR 和 INR 的数量。

ARM 通过检测 INTERRUPT 的上升沿来触发中断。当任意被使能的串口模块产生中断后，中断控制模块检测到其上升沿将 INTERRUPT 拉高，持续一个时钟周期后拉低，等待后续其他中断的触发，当有另一个串口产生中断后 INTERRUPT 仍会触发一次，有效避免了单串口中断电平一直拉高影响后续中断触发的问题。

1.3 单串口控制模块

确定触发中断的具体接口后，ARM 继续通过对相应寄存器的读取来确定该串口产生中断的具体类型，在 FPGA 中为每一个串口设置了一个串口控制模块（UART _ CONTROL），如图 5 所示，其中 DataIn、DataOut、ICR 和 SR 均通过寄存器模块与 ARM 进行交互。DataIn 和 DataOut 分别为接收 FIFO 最终传向 ARM 的数据和 ARM 写入发送 FIFO 的数据，ICR（Interrupt Control Register）为串口控制寄存器，用来控制 UART 通信过程中各类中断的开启，SR（Status Register）为状态寄存器，用来记录产生中断的具体类型和该串口控制器当前的一些控制状态。

为了提高通信效率，避免频繁的串口数据读写降低

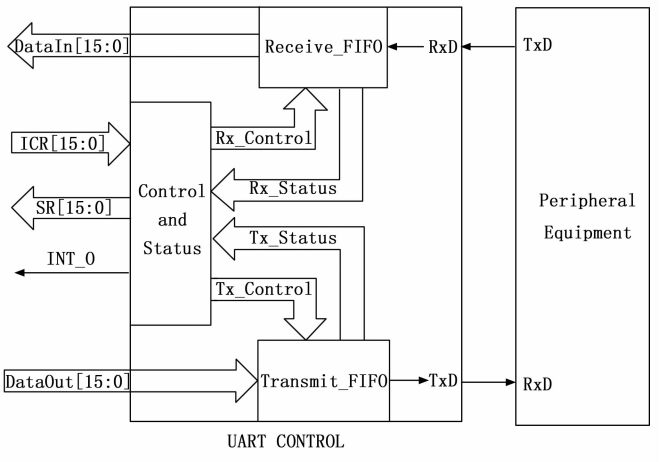


图 5 串口控制模块框图

ARM 的工作效率引起数据拥堵，在串口控制模块中实现读写的 FIFO 模块，FIFO 模块与外部设备连接并可将数据缓存在 FIFO 中，方便合适的时候批量的与 ARM 进行传输。

由于不同外部设备传输的数据时钟周期和数据帧大小均不相同，为了适用于不同的应用场景，在串口控制模块中实现了 5 种常用的中断触发方式，通过将 ICR 指定位置 1 来开启不同的中断，本设计中能触发中断的类型如表 1 所示。

表 1 中断控制寄存器

ICR 置 1 的位数	中断类型	触发方式
0	接收数据就绪中断	接收 FIFO 中数据深度大于设定的阈值后触发
1	发送 FIFO 空中断	发送 FIFO 无数据帧后触发
2	数据状态错误中断	多种导致数据状态错误的情况均会引起该中断，包括奇偶校验错误、帧错误、读空 FIFO 错误、发送接收 FIFO 溢出中断，触发后可通过读取 SR 来确定
3	接收超时中断	接收 FIFO 中有数据帧且超过一定时间没被 ARM 读取触发
4	发送超时中断	发送 FIFO 超过特定时间没有数据写入则产生该中断

（该寄存器的 5～15 位作为保留位默认置 0，可用以后续扩展其他的中断类型）

ARM 可随时读取相应串口模块的 SR 来判断触发串口中断的类型和该串口模块当前的状态，并根据不同的需求进行相应的处理。SR 各位数表示的状态如表 2 所示。

2 系统仿真测试

2.1 仿真环境搭建

基于 Vivado 平台，编写 Testbench 文件对 FPGA 中的各类中断进行测试，仿真过程中的各外部信号定义如下：

表 2　状态寄存器

位数	值 1	清零方法
0	有中断产生	无中断自动清零
1	产生接收数据就绪中断	读接收 FIFO 里的数据到深度少于阈值
2	产生发送 FIFO 空中断	向发送 FIFO 里写入数据
3	产生数据状态错误中断	读取 SR
4	产生接收超时中断	读接收 FIFO 中的数据
5	产生发送超时中断	向发送 FIFO 里写入数据
6	最新接收到的数据有奇偶校验错误	读取 SR
7	最新接收到的数据帧格式有错误	读取 SR
8	发送 FIFO 为空	向发送 FIFO 里写入数据
9	接收 FIFO 为空	向接收 FIFO 里写入数据
10	发送 FIFO 溢出	读取 SR
11	接收 FIFO 溢出	读取 SR
12	保留位	保留位
13	保留位	保留位
14	保留位	保留位
15	保留位	保留位

（注：12～15 作为保留位，可用以扩展其他状态）

1）FSMC 读写的控制信号：CS _ N、OE _ N、WE _ N、NADV。

2）FSMC 读写的地址和数据复用信号：AD［15：0］。

3）各路串口的输入输出信号：uart _ rx、uart _ tx。

在仿真文件中编写 ARM 通过 FSMC 配置 FPGA 数据的时序，开始通信后 CS _ N 拉低，读数据时 OE _ N 拉低，写数据时 WE _ N 拉低，AD 总线实现了 16 位地址与数据信号的复用，通过 NADV 来对地址和数据进行区分，NADV 为低电平时表示 AD 总线传输的为地址，为高电平时表示为数据。

为了遵循传统的编码习惯以及将仿真中的寄存器与实际的概念区分开，仿真中各寄存器统一使用小写命名。ARM 通过配置中断控制模块的 IMR 和每一路串口模块的 ICR 完成系统初始化的设置。在中断控制模块中使能串口 0～4 的中断，屏蔽掉其余的中断，即将 IMR 设置为 0x001f。对各串口模块开启特定的中断，串口 0 开启接收数据就绪中断，串口 1 开启发送 FIFO 空中断，串口 2 开启数据状态错误中断，串口 3 开启接收超时中断，串口 4 开启发送超时中断，其余串口的中断默认关闭。

2.2　中断控制模块仿真测试

对中断控制模块进行仿真，仿真波形如图 6 所示，配置 imr 为 0x000f，即开启序号 0～3 的串口中断，int0 _ o～int4 _ o 分别表示各个串口模块的中断信号，用 inr 来保存其电平状态，interrupt 为连接到 ARM 的中断信号，由图 7 可见，在 int0 _ o，int1 _ o，int2 _ o 的中断上升沿均将 interrupt 拉高了一个时钟周期触，且当 int1 _ o 未拉低时 int2 _ o 仍能将其拉高，避免了单个串口模块中断持续时间过长导致系统错过其他模块中断的问题。而 int4 _ o 由于未使能，其上升沿并不会导致 interrupt 被拉高（图中竖线处），但 inr 仍会记录 int4 _ o 的电平状态，方便观察其串口模块的真实状态。

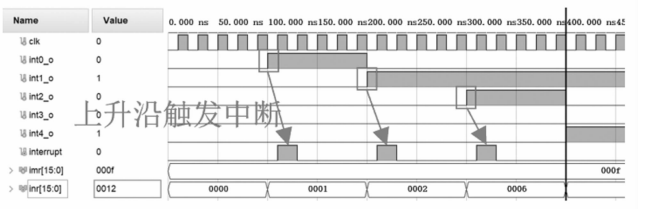


图 6　中断控制模块仿真波形

2.3　各串口仿真测试

串口 0 开启接收数据就绪中断，当接收 FIFO 中存储的数据深度超过设定的阈值后触发该中断，该中断常用于接收固定大小的数据包（惯导的数据等）。将阈值设定为一个完整数据帧的大小，接收到一个完整的数据帧后触发中断，ARM 触发中断后批量读取一个完整的数据帧。

将串口 0 的触发深度设置为 4，当串口 0 的接收 FIFO 存储了 4 个及以上字节后，串口 0 的中断电平 int _ 0 会被拉高。

该串口仿真波形图如图 7 所示，为了方便测试，在仿真中将串口 0 的发送和接收引脚连接到一起，串口 0 随机发送 4 个字节的数据，由于 data _ out 和 data _ in 宽度被配置为 16 位，而这里的串口发送数据长度为 8 位，因此 data _ out 和 data _ in 的前 8 位数据为无效位，默认置 0，实际传输数据为（ab，34，a7，77），串口 0 接收这 4 个数据，用 rf _ count 表示接收 FIFO 中储存的数据深度，可看到当第 4 个数据被存入接收 FIFO 后，rf _ count 变为 4，int0 _ o 被拉高。inr 第 0 位置 1 表示串口 0 触发了中断，sr0 第 0 位和第 1 位置 1 表示为接收数据就绪中断。此时中断控制模块检测到 int0 _ o 的上升沿，产生一个时钟的高电平信号触发一次 ARM 外部中断。外部中断触发后读取接收 FIFO 里的值到 ARM，可看到当读取一个数据后，rf _ count 变成了 3，此时接收 FIFO 中的数据低于触发深度阈值，因此中断信号被拉低。inr 恢复到 0x0000，sr0 的第 0 位和第 1 位也被置 0。一直到读取完全部 4 个字节后（图中竖线处），sr0 的第 7 位和第 8 位被拉高，表示此时接收 FIFO 和发送 FIFO 均为空，一次中断触发结束。

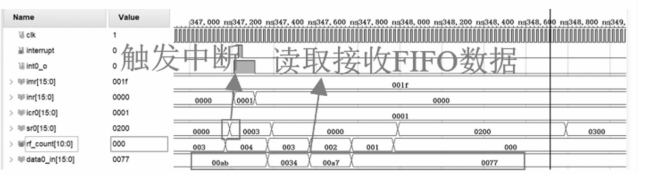


图 7　接收就绪中断仿真波形

串口 1 开启发送 FIFO 空中断，当发送 FIFO 中没有数据后触发该中断，该中断常用于发送大批量数据时提示 ARM 数据全部发送完成。

仿真图如图 8 所示，图中 tf _ count 表示发送 FIFO 中储存的数据深度，data1 _ out1 表示传入发送 FIFO 的数据。系统开启后发送 FIFO 默认为空，可看到系统开启后 sr1 第 8 位默认为 1，表示发送 FIFO 为空，随后 int1 _ o 中断被触发，信号拉高，中断控制模块检测到 int1 _ o 上升沿后将 interrupt 拉高一个时钟周期触发系统中断，此时 inr 第 1 位置 1 表示串口 1 产生了中断，sr1 第 0 位置 1 表示有中断被触发，第 2 位置 1 表示产生的为发送 FIFO 空中断。

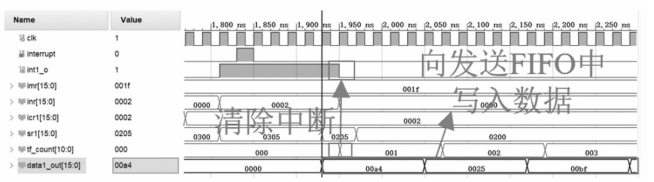


图 8 清除发送 FIFO 空中断仿真波形

中断触发后向发送 FIFO 中写入数据，图中竖线处表示系统开始向发送 FIFO 中写入数据，此时 sr1 第 8 位置 0，一个时钟周期后数据写入到发送 FIFO 里，中断信号随之拉低，一次触发结束。

经过一段时间后，写入发送 FIFO 的数据全部传输完毕（图中细线处），此时 tf _ count 变为 0，为了防止此时有数据正在写入，因此延迟一定的时间再对 FIFO 的状态标志进行改变，随后可再次触发发送 FIFO 空中断，仿真图如图 9 所示。

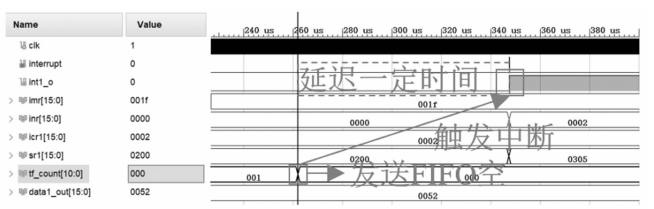


图 9 发送 FIFO 空中断仿真波形

串口 2 开启数据状态中断，当检测到数据帧传输错误后触发该中断，常用于需要较强可靠性数据的场合，如识别地面站发送的控制指令等。

开启串口 2 模块中通信数据的奇偶校验位并向串口的接收 FIFO 传入数据，仿真波形图如图 10 所示，在传入数据时给定错误的奇偶校验位，可看到在竖线处 sr2 的第 9 位置 0，第 6 位置 1，表示串口 2 控制模块的接收 FIFO 中收到了数据且有奇偶校验错误，随后触发数据状态错误中断，int2 _ o 被拉高，icr9 第 2 位被置 1 表示是串口 2 产生中断，sr2 第 0 位和第 3 位置 1 表示产生了数据状态错误中断，检测到 interrupt 触发后读取一次 sr2，可清除中断标志位，等待下一次中断产生。

串口 3 开启接收超时中断，接收 FIFO 中有数据帧且超过一定时间没被 ARM 读取触发该中断，常用于一些数据量不大且不需要立刻处理的串口数据，如湿度数据等。

仿真波形如图 11 所示，向串口 3 的接收 FIFO 中写入

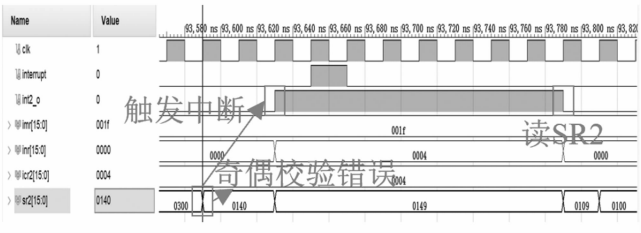
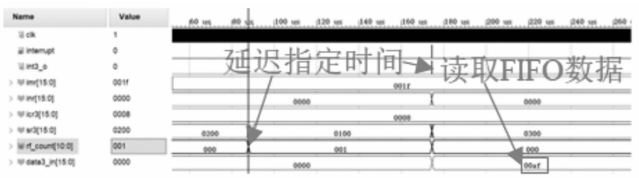
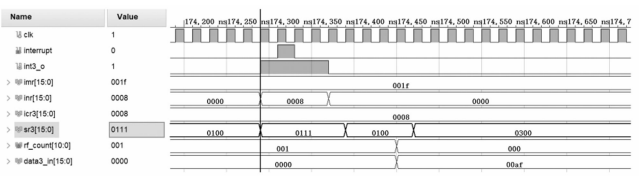


图 10 数据状态错误中断仿真波形

一个字节数据，rf _ count 表示此时接收 FIFO 中的数据深度，data3 _ in 为读取接收 FIFO 时输出的值，此时 sr3 第 9 位置 0 表示接收 FIFO 不为空，串口控制模块检测到接收 FIFO 中有数据且一直未被读取，等待设置时间到达后自动触发中断，此时 inr 第 3 位置 1 表示串口 3 产生了中断，sr3 第 0 位和第 4 位置 1 表示产生了接收超时中断，interrupt 拉高后开始读取接收 FIFO 中的值，随后 rf _ count 开始减少，中断信号被拉低。



(a)



(b)

图 11 接收超时中断仿真波形

串口 4 开启发送超时中断，当经过特定的时间发送 FIFO 没有数据写入后则会触发此中断，常用于需要周期性发送数据的串口，如心跳信号的发送，电机的状态查询指令等。

仿真波形图如图 12 所示，中断触发前 sr4 第 8 位置 1 表示发送 FIFO 一直为空，中断触发后，可见 inr 第 4 位置 1，表示串口模块 4 产生中断，sr4 第 0 位和第 5 位置 1 表示产生发送超时中断。与发送 FIFO 空中断类似，中断触发后向串口 4 的发送 FIFO 中写入数据（图中竖线处），data _ out 为写入发送 FIFO 的值，可见此时 sr4 第 8 位置 0，一个周期后 tf _ count 的值变成 1，表示发送 FIFO 中已经写入了数据，随后中断信号拉低，等待下次触发。

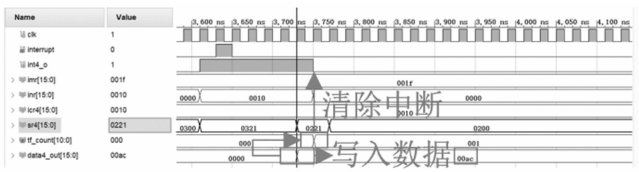


图 12 发送超时中断仿真波形

3　板级测试

通过自行设计的 PCB 板来进行测试，如图 13 所示，选用 STM32F407 作为主控芯片，FPGA 芯片则选用了 Xilinx 旗下的 xc7a35t 芯片，使用 FPGA 扩展出多路串口，经过电平转换芯片转化为 RS-422 接口与上位机连接。stm32 通过 FSMC 总线完成对 FPGA 寄存器的读写来实现对各串口中断的管理。因为 FSMC 总线的 data 常见宽度为 16 位，为了方便配置将寄存器宽度设计为 16 位，因此 IMR 和 INR 刚好可对 16 路串口的中断进行管理，若需要对更多的串口实现中断管理，仅需增加 IMR 和 INR 的数量即可。

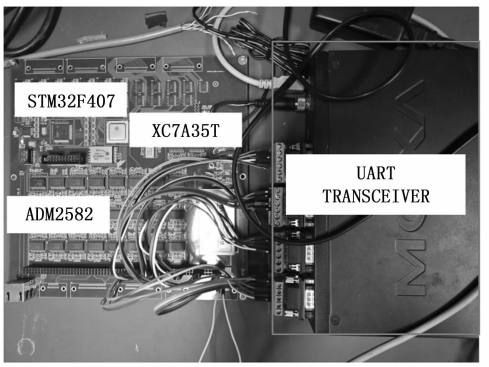


图 13　自行设计的测试板

将 FPGA 视作 NOR FLASH 挂载到 STM32 上，STM32 的 FSMC 分配给 NORFLASH 的可用地址范围为 60000000h－6FFFFFFFh，由于 STM32 内部每一个地址对应 8 位数据，当外部存储器宽度为 16 位时，向外部存储器发出的地址需要右移一位。因此每两个地址可表示 FPGA 中一个 16 位寄存器的读写地址，只需要地址范围不超过 6FFFFFFFh，则可继续增加寄存器管理更多的串口。

在 STM32 中配置好 FSMC 协议，利用指针直接对指定地址的数据进行读写，STM32 检测到是 FSMC 的地址区域时会自动生成读写的时序协议，未配置的情况下寄存器的值默认为 0x00。开始工作前在 STM32 中配置寄存器，配置 IMR 为 0x001f，ICR0 为 0x0001，ICR1 为 0x0002，ICR2 为 0x0004，ICR3 为 0x0008，ICR4 为 0x0010。将 interrupt 引脚连接到 STM32 的外部中断引脚上，STM32 检测到 interrupt 的上升沿后触发中断。

测试多路串口外设的中断管理，对整个系统上电，同时开启五路串口连接的上位机（串口 1 与串口 4 结果较为相似），使用上位机对串口 0 和串口 3 发送数据，STM32 检测到中断触发后首先读取 INR 判断产生中断的串口，若检测到为串口 0 或串口 3，ARM 读取相应接收 FIFO 的数值并将其写入自身发送 FIFO。上位机与串口 0 和串口 3 的数据收发分别如图 14 和 15 所示，对上位机的数据进行对比，可看出串口 0 仅当接收数据超过触发深度（这里设置为 4）后触发中断，而串口 3 当接收 FIFO 中有数据且超过特定时间（这里设置为 4 个时钟周期）后即可产生中断。

图 14　上位机与串口 0 数据收发界面

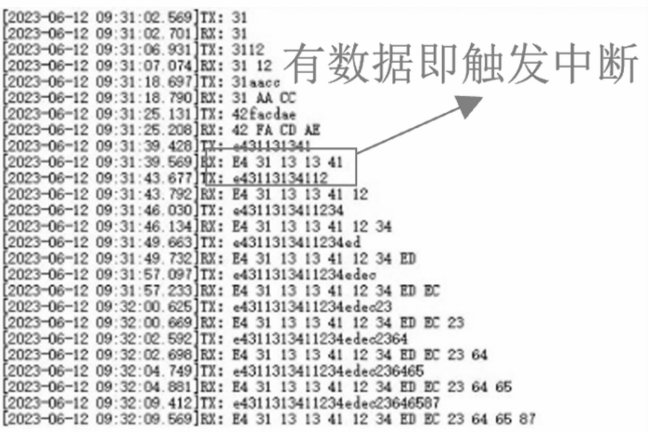


图 15　上位机与串口 3 数据收发界面

若检测为串口 1 和串口 4 产生的中断后，向其发送 FIFO 中写入数据，以串口 1 为例，检测到其产生发送 FIFO 空中断后向发送 FIFO 中写入数据（0x10，0x24，0xa4，0xce），通过上位机可观测到串口 1 不停的向外发送数据，如图 16 所示。

图 16　上位机不停接收到串口 1 发送的数据

串口 2 开启数据状态错误中断，以奇偶校验错误为例，我们在串口 2 中使用数据的奇数校验，但用上位机发送经

过偶数校验的数据。STM32 检测到串口 2 发生数据状态错误中断后读取 sr2，若 sr2 第 6 位被置 1（即产生了奇偶校验错误），则向外输出奇偶校验错误标志 0xaa，如图 17 所示。

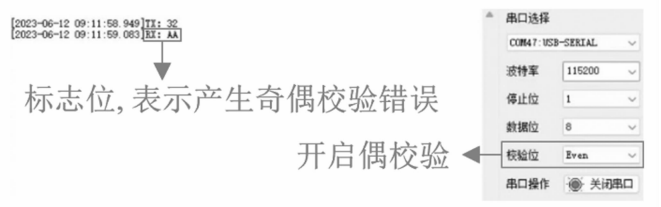


图 17 上位机与串口 2 奇偶校验错误测试

4 结束语

本文设计实现了基于 FPGA 的多串口中断管理器。在 ARM＋FPGA 的架构下在 FPGA 中扩展带有 FIFO 模块的多路串口模块，同时设计了五类较为实用的中断类型来触发 ARM 的外部中断并可对多路串口的中断进行统一管理。通过配置不同类型的中断触发方式使飞控计算机有效与各种类型的设备进行交互，合理的收发不同周期和大小的数据包。

与外部设备的数据传输、各中断类型的管理和串口数据的缓存均在 FPGA 中实现，仅使用一个 IO 口来触发 ARM 的外部中断，仅当某串口设备完成一个完整的操作后（如收到一包完整的数据包）才触发一次中断，有效提升了 ARM 的接口资源利用率和运行效率。同时本设计具有较强的可扩展性，在 FPGA 资源足够的情况下可进一步扩展更多的串口或者其他协议的模块，具有较强的工程应用性。

参考文献：

[1] 徐冬梅，王文峰，余雪梅. 无人机系统标准化概述及应用展望 [J]. 信息技术与标准化，2020 (7)：64－67.

[2] 张建鹏. 小型低空飞艇测控系统中的飞行控制计算机设计 [D]. 上海：上海交通大学，2010.

[3] WANG Z，HE Z S. Flight control system design for a small size unmanned helicopter [C] //2011International Conference on Mechatronic Science，Electric Engineering and Computer (MEC). Jilin，China：IEEE，2011，543－546.

[4] 刘福周. 基于 ARM 和 DSP 超小型无人直升机飞控系统设计与实现 [D]. 哈尔滨：黑龙江大学，2017.

[5] LIU H T，DU H B，WANG J H. Construction of hydraulic chuck reliability test platform by ARM DSP dual-core system [J]. Journal of Physics：Conference Series，2020，1601：062030.

[6] 范君健，吴国东，王志军. 基于 FPGA 和 ARM 的飞行器控制系统设计 [J]. 自动化与仪表，2018，33 (1)：23－28.

[7] YANG H X，GENG Q B. The design of flight control system for small UAV with static stability [C] //2011 Second International Conference on Mechanic Automation and Control Engineering. Inner Mongolia，China：IEEE，2011，799－803.

[8] 黄 浩. 无人机飞行控制计算机硬件平台研制 [D]. 哈尔滨：哈尔滨工业大学，2021.

[9] YANG X F，QIU C C. Design and implementation of miniaturized flight control computer based on FPGA＋DSP [C] //Proceedings of the 2022 5th International Conference on Artificial Intelligence and Pattern Recognition. Xiamen，China：Association for Computing Machinery，2023：1141－1145.

[10] LIU X C，LIU L B，BAI X B，et al. A low-cost solution for leader-follower formation control of multi-UAV system based on Pixhawk [J]. Journal of Physics：Conference Series，2021，1754：012081.

[11] MA Y，DUAN P，HE P C，et al. FPGA implementation of extended Kalman filter for SOC estimation of lithium-ion battery in electric vehicle [J]. Asian Journal of Control，2019，21：2126－2136.

[12] MUHSEN O，DAVID S，KAAN K，et al. Centaur：A Framework for Hybrid CPU-FPGA Databases [C] //2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). Napa，CA，USA：IEEE，2017：211－218.

[13] 罗 霄，薛亚洲，张 乐. 一种无人机飞控计算机硬件平台的设计实现 [J]. 电子测量技术，2021，44 (1)：50－54.

[14] LIU P，ZHANG S Z，PAN X B，et al. Design and implementation of heterogeneous dual core data transmission system based on FSMC bus [C] //2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT. Weihai，China 2020，434－437.

[15] ZHU C X，BAO L X，ZHENG B W，et al. Motor dynamic loading and comprehensive test system based on FPGA and MCU [J]. Electronics，2022，11：1317－1328.

[16] HU Z，ZHANG J，LUO X L. A Novel Design of Efficient Multi-channel UART Controller Based on FPGA [J]. Chinese Journal of Aeronautics，2007，20：66－74.

[17] SARDI I. Design and implementation of UART with FIFO buffer using VHDL on FPGA [J]. ICTACT J. Microelectron，2019，5：7.

[18] BIBIN M C，PREMANANDA B S. Implementation of UART with BIST technique in FPGA [J]. International Journal of Inventive Engineering and Sciences (IJIES)，ISSN，2013：2319－9598.

[19] WANG Y，SHEN Z. The design of UART controller based on FPGA [C] Berlin，Heidelberg：Springer Berlin Heidelberg，2011：221－225.

[20] YU S Q，YI L L，CHEN W H，et al. Implementation of a Multi-channel UART Controller Based on FIFO Technique and FPGA [C] //2007 2nd IEEE Conference on Industrial Electronics and Applications. Harbin，China：IEEE，2007：2633－2638.

[21] ALBA M D，ANDRADE A，GOMEZ T J，et al. FPGA design of an efficient and low-cost smart phone interrupt controller [J]. Latin American Applied Research，2007，37：59－63.