文章编号:1671-4598(2016)05-0190-05 DOI:10.16526/j.cnki.11—4762/tp.2016.05.054 中图分类号:TP274.2 文献标识码:A

基于 OMAP3530 嵌入式最小系统的开发

李 攀[1],马云彤[1],邵云峰[2],彭 涛[2]
(1.哈尔滨工业大学 电气工程及自动化学院 自动化测试与控制系,哈尔滨 150080;
2.北京电子工程研究所,北京 100854)

摘要:随着嵌入式系统日趋复杂,类似机载状态监测系统,对嵌入式处理器在高性能计算和复杂控制方面的性能提出了更高的要求;针对复杂嵌入式系统的高性能需求,介绍了 TI 公司 ARM+DSP 架构的双核异构处理器 OMAP3530,并进行了 OMAP3530 嵌入式最小系统的开发,包括硬件、软件设计两个方面;硬件设计包括电源、时钟、存储器以及外围接口等模块,实现了 OMAP3530 启动以及和外围通信的最基本硬件组成;软件设计包括 ARM 端操作系统移植、ARM 和 DSP 双核通信,在此基础上设计了双核的应用程序;最后采用 LabVIEW 在上位机上设计了软件部分的测试程序,测试了系统的完整性;OMAP3530 作为一款高性能的嵌入式处理器,为复杂的嵌入式系统应用提供了解决思路。

关键词:高性能计算;复杂控制;OMAP3530;嵌入式最小系统;双核通信

Development of Embedded Minimum System With OMAP3530

Li Pan[1], Ma Yuntong[1], Shao Yunfeng[2], Peng Tao[2]
(1. Department of Automatic Test and Control, Harbin Institute of Technology, Harbin 150080, China;
2. Beijing Institute of Electrical Engineering, Beijing 100854, China)

Abstract: With the increasing complexity of embedded systems, like airborne monitoring system, require high-performance computing and complex control performance of embedded processors. For high-performance requirements of complex embedded systems, introduce the TI's ARM + DSP dual-core heterogeneous processor, OMAP3530. Develop OMAP3530 embedded minimum system, including hardware and software designs. In hardware, design power, clock, memory and peripheral interface module, which are the basic hardware components of start and communications. In software, migrate operating system for ARM, and achieve ARM and DSP dual-core communication, and design the ARM's and DSP's application based above. Finally, design test program using LabVIEW on the host computer, testing the integrity of the system. As a high-performance embedded processor, OMAP3530 provides solution ideas for complex embedded systems applications.

Keywords: high-performance computing; OMAP3530; embedded minimum system; dual-core communication

0 引言

随着科技的发展,嵌入式系统面向的应用越来越复杂,所需要的功能也更多样。例如在机载状态监测系统中[1],系统首先需要采集飞机的振动信号,温度信号、应力信号等,然后将信号传给数据处理中心通过复杂的算法进行实时分析处理,最后将分析的结果通过人机交互接口呈现给技术人员[2],以此来对飞机的状态进行分析和预测,为飞机安全飞行和维护提供依据[3]。在整个系统中,需要采集大量的数据,利用复杂的处理算法,要求系统同时具备高性能的实时计算和外围设备控制能力,对系统功耗的要求也非常严格。目前,在嵌入式系统领域,类似机载状态监测系统,对兼具高性能实时计算和复杂控制需求的系统越来越多。

针对上述的需求,对系统的处理核心嵌入式处理器提出了较高的要求。在嵌入式处理器领域中,DSP 由于其独特的硬件结构,在数字信号处理方面具有很大的优势;ARM 具有强大的控制能力,可以运行嵌入式操作系统,为用户提供了强大的软硬件开发平台;但二者主要针对单一的应用,对于复杂的嵌入式系统的需求难以胜任。

随着集成技术的发展,嵌入式多核处理器已经成为发展趋势,针对需要同时处理高性能计算和复杂控制的应用,各大公司推出了相应的嵌入式处理器方案,主要是异构多核处理器[4]。典型代表是 TI 公司的 OMAP 系列处理器,采用 ARM+DSP 的双核架构,使得数据处理能力得到大大的增强[5]。

本文针对复杂嵌入式系统的需求,基于 TI 公司的 OMAP3530 处理器,介绍 ARM+DSP 架构的嵌入式处理器的软硬件设计方法。

1 OMAP3530 最小系统硬件设计

1.1 OMAP3530 简介

如图 1 所示为 OMAP3530 框架图,它在一片硅片上集成了 720 MHz 的 ARM Cortex-A8 核和 520 MHz 的 DSPT-MS320C64x+核,处理性能优越,而且支持丰富的外设资源。在功耗方面,OMAP3530 集成时钟门控与睡眠模式,可在不降低性能的情况下降低功耗。

收稿日期:2015-11-02; **修回日期**:2015-12-24。

作者简介:李 攀(1993-),男,四川阆中人,硕士研究生,主要从事嵌入式系统开发、异构 DSP 的开发。

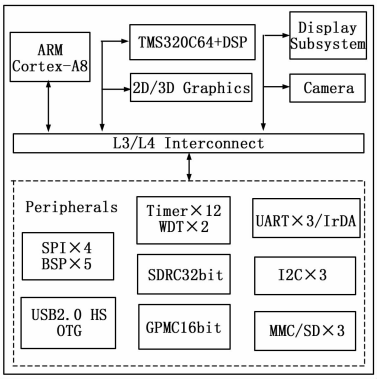

图 1　OMAP3530 硬件结构

1.2　硬件设计

OMAP3530 最小系统是 OMAP3530 正常工作的最基本硬件构成，满足基本的数据处理和系统控制功能，主要包括 OMAP3530 启动必需的电源模块、时钟模块、存储器模块。外围接口方面，需具备调试所需的接口如 JTAG 接口、串行通信接口、SD 卡模块，考虑到后期应用开发过程涉及到人机交互，还需考虑设计 USB 接口、视频显示接口等。如图 2 所示为 OMAP3530 最小系统的硬件设计框图。

OMAP3530 嵌入式最小系统的核心部分是电源、时钟和存储器部分。

OMAP3530 需要多种电压值供电，采用 TI 为集成电源管理芯片 TPS65930 为系统供电，其集成了多路电源管理转换通道，同时还支持音频编解码器、USB2. 0 收发器、键盘等接口资源，简化了系统设计，同时在功耗也进行了优化[6]。

在时钟设计方面，针对 OMAP3530 的低功耗特性，本文为 OMAP3530 提供了两个外部时钟，一个是 26 MHz 的高频时钟，作为系统主时钟；一个是 32 kHz 的低频时钟，作为 OMAP3530 低功耗模式下的时钟。

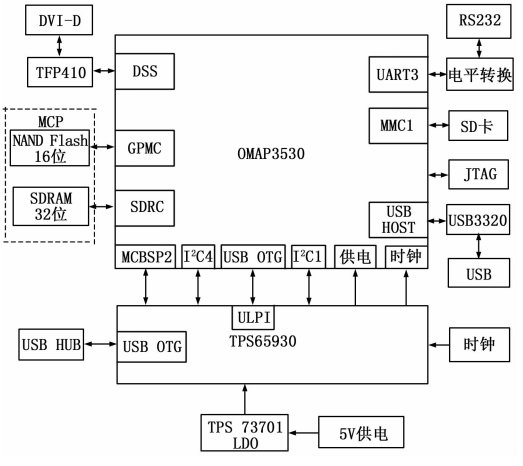

图 2　OMAP3530 最小系统硬件设计框图

OMAP3530 处理器内部集成了 80 KB 的 ROM 和 64 KB 的 RAM，容量太小，本文采用镁光 MT29C4G48M 存储芯片扩展存储，该芯片将 512 MB 的 16 位 NAND Flash 和 256 MB 的 32 位 LPDDR SDRAM 集成在封装在一起，但二者的地址线、数据线、控制线、电源引脚相互独立，有效的节省了 PCB 面积。

2　OMAP3530 软件设计

2.1　软件总体设计框架

OMAP3530 软件设计主要分为系统软件和应用程序两个部分，图 3 所示为软件设计框图。

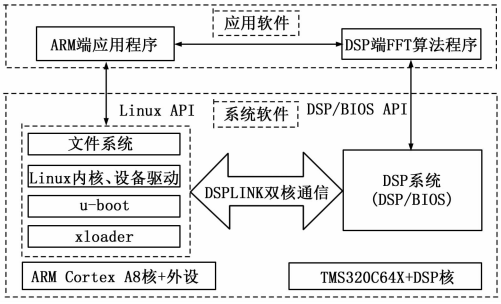

图 3　软件设计整体框图

系统软件方面，本文在 OMAP3530 系统 ARM 端运行嵌入式 Linux 系统，用于实现系统的启动，外围设备资源的管理以及应用程序的运行；在 DSP 端运行 DSP/BIOS 系统，主要工作是算法的实现；为了发挥 OMAP3530 双核架构的优势，使 ARM 核和 DSP 核协同工作，需要实现双核通信，本文采用 TI 开发的 DSP/BIOS Link（简称 DSPLINK）驱动模块实现双核通信[7]。

应用程序方面，在 ARM 端，设计了应用程序通过串口和上位机通信，同时利用 DSPLINK 模块实现和 DSP 通信；相应地，在 DSP 端，除了实现和 ARM 的通信外，以 FFT 算法为例，实现在 DSP 端进行算法处理。

2.2　系统软件实现

2.2.1　嵌入式 Linux 系统移植

ARM 端嵌入式 Linux 系统的移植，主要分为 4 个层面：xloader、u-boot、kernel 和文件系统的移植。

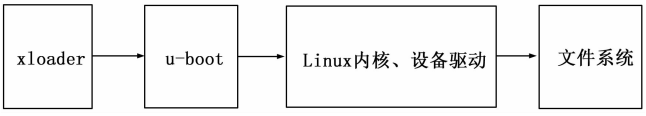

图 4　嵌入式 Linux 系统移植流程

嵌入式操作系统在硬件上电后利用引导程序将 Linux 内核加载到目标硬件上，这就是 xloader 和 u-boot 需要完成的工作。xloader 是一级引导程序，程序短小、功能简单，用作简单初始化，加载并执行 u-boot；u-boot 是二级引导程序，完成初始化系统时钟和 SDRAM，关闭看门狗等工作，从 Flash 中读出内核写到 SDRAM 中。

嵌入式 Linux 内核是操作系统的核心，其主要功能包括进程管理、内存管理、文件管理、设备管理、网络管理等。

根文件系统的功能主要是将操作系统中与管理相关的所有软件和数据进行目录结构式的统一管理[8]。

2.2.2　OMAP 双核通信机制

本文采用 TI 公司提供的 DVSDK 软件进行应用程序的开

发，利用 DVSDK 的 DSPLINK 模块实现 ARM 和 DSP 之间底层通信。DSPLINK 的核心是 ARM 和 DSP 共享内存，其具体机制是 ARM 或 DSP 将数据写到共享内存，然后通过邮箱（Mailbox）发送消息并产生中断，通知对方可以读写共享内存中的数据，从而实现了有效的通信机制。当小数据量的通信时，一般使用片内的共享内存，速度最快，当需要大数据量的通信时，可以分配片外的 DDR 作为共享内存[9]。

DSPLINK 以设备驱动的形式加载在 Linux 系统中，将 ARM 与 DSP 的物理连接特性抽象出来，为 ARM 端应用程序提供了一套 API，即 DSPLINK API。在 DSP 端与其连接的是 DSP/BIOS 实时操作系统，作为驱动形式存在，为 DSP 端应用程序提供 API，即 DSP/BIOS API，其结构和作用和 DSPLINK API 相似。在应用层实现了 ARM 和 DSP 之间的通信，降低了用户开发程序的复杂度[7]，DSPLINK 模块的结构如图 5 所示。

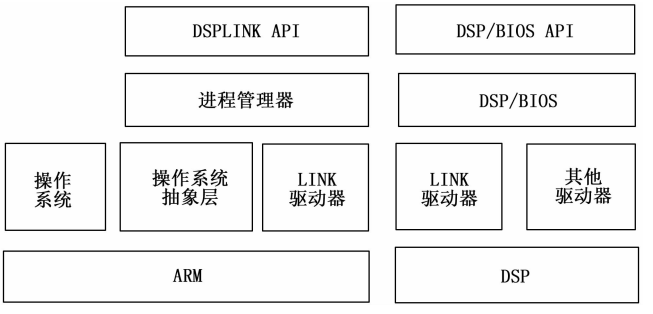

图 5 DSPLINK 模块结构

基于 ARM 端 DSPLINK 包括以下几个部分：

操作系统抽象层，包含 DSPLINK 需要的一些通用的操作系统服务组件，提供一套通用的 API 与操作系统的其它组件隔离，使 DSPLINK 可以方便地移植到不同的操作系统中[10]。

LINK 驱动器，包含基于 ARM 与 DSP 的物理连接的底层控制操作，负责 ARM 与 DSP 之间的数据传输和 DSP 的运行等操作[10]。

进程管理器，维护针对所有模块的 Book-Keeping 信息，通过 API 给用户提供通过 LINK 驱动器的控制操作[10]。

在 DSP 端，LINK 驱动器是 DSP/BIOS 驱动中的一部分，只负责基于物理连接之上与 ARM 之间的交互。

DSPLINK 为 ARM 应用程序提供的常用 API 组件有 PROC、CHNL、MSGQ、POOL 等[10]。

PROC 组件，其功能是基本的处理器控制，在 OMAP3530 中主要是 ARM 对 DSP 的控制。包括初始化 DSP、加载 DSP 的程序、从 DSP 代码指定的地址运行 DSP 程序等。

CHNL 组件，功能是在应用空间提供一个逻辑数据传输通道，是一种 ARM 和 DSP 数据传输的方式。

MSGQ 组件，负责 ARM 与 DSP 之间可变长度的短消息交互，消息的发送接收都通过队列实现，发送者将数据写入到消息队列中，接收者从消息队列接收信息。一个消息队列只可以有一个接收者，但可以有多个发送者，DSP/BIOS 系统为 DSP 也提供相应的 API 组件。

POOL 组件，用于控制存储器池的创建和关闭，POOL 创建的存储器池可供 CHNL 和 MSGQ 组件使用，用于创建数据和消息传递的缓存区，DSP/BIOS 系统为 DSP 也提供相应的 API 组件。

2.3 应用程序设计

通过在 ARM 端和 DSP 端应用程序分别调用 DSPLINK 和 DSP/BIOS 提供的 API 可以实现双核通信，在此基础上实现应用程序。

2.3.1 应用程序总体结构

如图 6 所示为应用程序的总体结构，整个程序的核心思想是从上位机串口获取波形数据并在 DSP 中进行 FFT 处理，在实际应用中数据还可以来自传感器采集通过 AD 转换而来，也可以来自 USB 接口等。串口的收发控制在 ARM 中完成，通过调用嵌入式 Linux 系统的串口驱动实现；ARM 将波形数据通过消息队列发送给 DSP 进行 FFT 处理；DSP 处理完成后，通过消息队列将结果发送给 ARM；ARM 再将处理结果通过串口发送给上位机，进行显示或进一步分析处理。

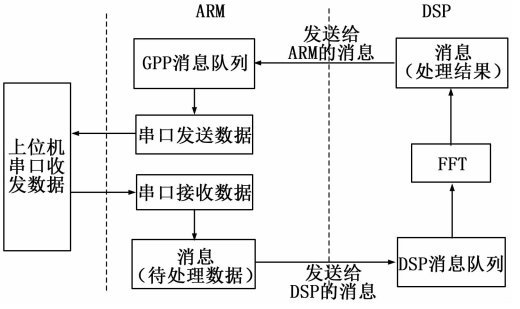

图 6 应用程序总体结构

2.3.2 ARM 端应用程序

如图 7 所示为 ARM 端应用程序流程图，主要分为两部分：消息队列的创建、发送和接收；串口数据的收发。

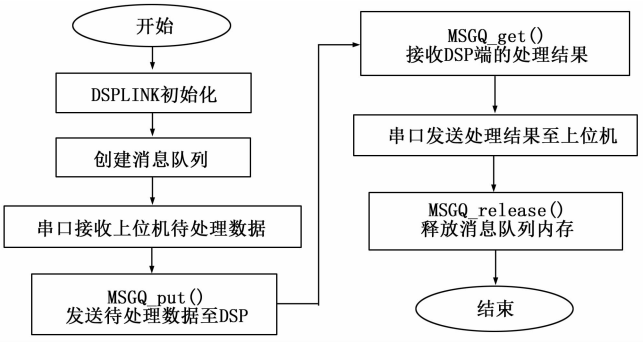

图 7 ARM 端应用程序流程图

1）消息队列：

如图 8 所示为消息队列创建流程图，消息队列主要是利用 DSPLINK 的 PROC、MSGQ、POOL 等 API 组件实现 ARM 和 DSP 的通信，通过调用这些 API 实现对消息队列的创建和消息的收发等。

由图 8 所示，ARM 端通过以下顺序调用 API 打开消息队列[11]：

(a) PROC_setup ()，对 PROC 组件进行创建和初始化；

(b) PROC_attach ()，建立与 ARM 端通信的 DSP 的连

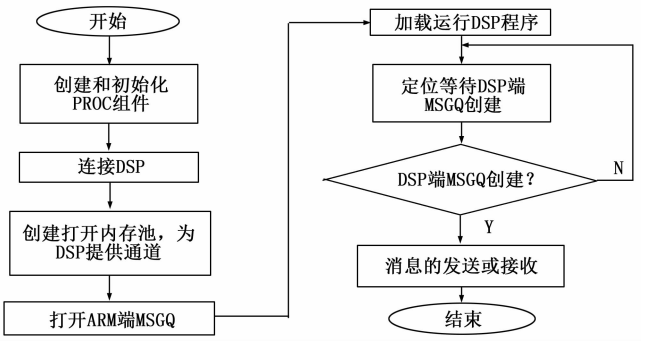

图 8　ARM 端消息队列创建流程

接，调用时通过 ID 号 ARM 就能连接到指定的 DSP；

（c）POOL _ open（），打开共享内存池；

（d）MSGQ _ open（），打开消息队列，使用消息队列通信前，通信双方需要打开一个消息队列，每个消息队列有独立的 name，当通信双方消息队列的 name 相吻合时，消息队列才能成功创建；

（e）PROC _ load（），将编译好的 DSP 程序加载到指定 ID 的 DSP 中；

（f）PROC _ start（），开始运行指定 ID 的 DSP 的程序；

（g）MSGQ _ locate（），等待已经连接的 DSP 方打开消息队列，才能进行通信；

通过调用上述的 API，ARM 端的消息队列创建完成，当 DSP 端消息队列创建完成后，就可以调用 MSGQ 组件的 MSGQ _ put（）和 MSGQ _ get（）两个 API，进行消息的发送和接收，在传输完成后，需要调用 MSGQ _ release（）释放消息队列内存。

2）串口：

Linux 系统下驱动以文件的方式放在文件系统/dev 目录下，串口的驱动也是如此，如串口 1 是/dev/ttyS0。在 Linux 下对串口的操作相当于对文件的操作，采用文件操作函数的“open”、“close”、“read”、“write”等可以实现对串口打开关闭和数据读写，如图 9 所示为串口应用程序流程图。

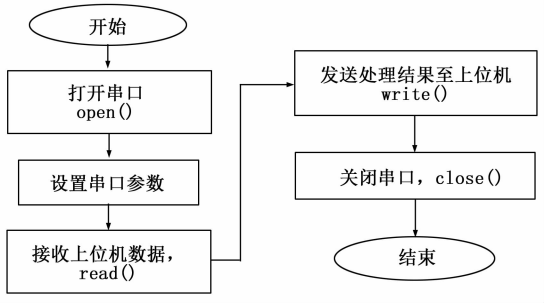

图 9　串口应用程序流程图

（a）打开串口，利用 open（）函数以读写方式打开串口；

（b）设置串口参数，打开串口成功后需要对串口参数进行设置，主要对串口波特率、校验位、停止位等的设置；

（c）读串口，使用 read（）函数，将串口缓冲区的数据读出来；

（d）写串口，使用 write（）函数，给串口缓冲区写数据；

（e）关闭串口，使用 close（）函数，关闭之前打开的串口。

通过以上步骤，ARM 就可以利用串口和上位机进行通信。

2.3.3　DSP 端算法程序

如图 10 所示为 DSP 端程序流程图，该程序主要是基于 DSP/BIOS 提供的 API 实现消息队列的创建以及和 ARM 的通信，在实现通信的基础上实现 FFT 算法。

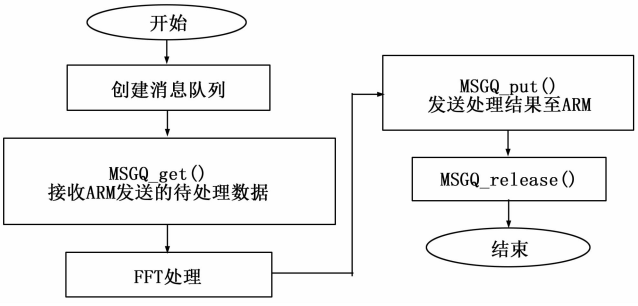

图 10　DSP 端程序流程图

1）消息队列

如图 11 所示为 DSP 端消息队列创建流程图，DSP 端消息队列的创建主要是利用 DSP/BIOS 提供的 MSGQ 组件实现的，和 ARM 端类似，MSGQ 为 DSP 端提供了相应的 API，通过调用这些 API 实现对消息队列的创建、数据的收发等。

DSP 端按如下顺序打开消息队列[11]：

（a）建立 TASK 任务，由于双核通信是基于两端操作系统进行的连接，因此，在 DSP 端同样必须采用操作系统作为通信的媒介，DSP 端采用 DSP/BIOS 操作系统，以任务的形式运行程序；

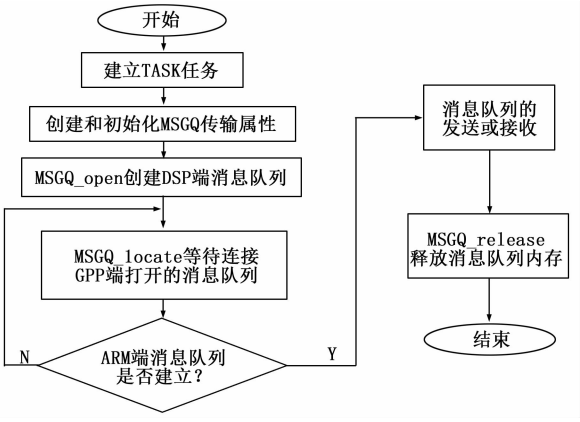

图 11　DSP 端消息队列创建流程图

（b）创建和初始化 MSGQ 传输属性，在进行 MSGQ 的创建打开之前，要先指定 MSGQ 的相关属性；

（c）MSGQ _ open（），打开 DSP 端消息队列；

（d）MSGQ _ locate（），等待连接 ARM 创建的消息队列；

（e）当 ARM 和 DSP 端消息队列都建立完成并连接，就可以开始通信，DSP 通过调用 MSGQ _ put（）和 MSGQ _ get（）可以实现消息的发送和接收。

2）算法设计

DSP 具有高性能的数字信号处理能力，本文为了说明开发流程，以频域抽取的 FFT 算法为例进行说明，在实际应用可以根据实际情况设计更为复杂的算法。

3 测试程序设计

本文利用 LabVIEW 软件设计了上位机测试程序，程序结构图如图 12 所示。测试程序是为了验证系统功能的正确性和完整性，本文主要验证 OMAP3530 最小系统串口通信和算法程序的正确性。

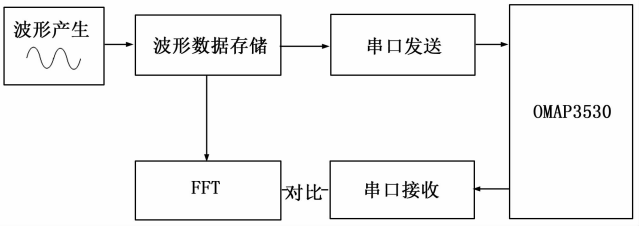

图 12 测试程序结构

如图 12 所示，测试程序主要包括波形数据产生、串口收发、FFT 计算：

1）波形数据产生，本文设计三路正弦波叠加，采样率为 1000Hz，采样点数为 16 点，这里的波形数据可以根据应用需要进行产生；

2）串口收发，调用 NI 提供的 VISA 库实现串口的数据收发功能，主要包括串口参数的设置，串口数据的读写，和 OMAP3530 最小系统的通信；

3）FFT 处理，本文采用 LabVIEW 的 FFT 模块对产生的波形数据实现 FFT 处理，作为 DSP 的 FFT 处理结果的验证参考标准，如果需要验证其他算法可以在 LabVIEW 上实现其他算法，也可以利用 Matlab 和 LabVIEW 混合编程实现。

通过多次测试，OMAP3530 和上位机的串口通信是正确的，同时 DSP FFT 计算所得的结果和上位机 LabVIEW 中 FFT 模块所得的结果在误差范围内是一致的，符合要求，因此验证了整个软件设计的正确性和完整性。

4 总结

本文根据机载状态监测系统等复杂系统的数据处理需求，提出了采用 TI 公司的 ARM＋DSP 架构的处理器 OMAP3530 作为数据处理核心的方案，并介绍了基于 OMAP3530 最小系统的硬件设计。软件方面，在 ARM 端移植了嵌入式 Linux 操作系统，利用 DSPLINK 实现了 ARM 和 DSP 的双核通信，介绍了 DSPLINK API 的调用方法，在此基础上设计应用程序，实现了 ARM 和 DSP 的协同工作。最后，针对应用程序的功能验证，利用 LabVIEW 设计了上位机测试程序，验证了应用程序功能的完整性和正确性。以上的开发流程针对 TI 的 ARM＋DSP 架构的处理器皆可使用。

OMAP3530 作为一款双核异构的处理器，在 ARM 端运行的是 Linux 系统，用于控制外设接口，如串口、USB 接口、HDMI 接口等，串口的功能主要用于打印调试信息和通信；USB 接口用于连接键盘、鼠标、U 盘等设备，或者用于通信；HDMI 接口主要用于连接显示器进行界面显示。DSP 端，为 TMS320C64x＋核，能胜任复杂的数字信号处理应用，本文设计了算法程序，用于处理 ARM 通过消息队列发送给 DSP 的数据，处理完后通过消息队列发送给 ARM。在系统运行的过程中，DSP 端可以看作是 ARM 的一个外设，ARM 负责 DSP 程序的启动、执行、结束等。

由以上的分析可以看出，ARM 和 DSP 是并行运行的，ARM 负责界面的运行和外围设备的控制，DSP 负责算法的执行，实现了在复杂嵌入式系统应用中 ARM 和 DSP 的协同工作。

随着嵌入式系统应用越来越复杂，目前单一的处理器已经不能满足复杂的嵌入式系统应用，根据本文针对 OMAP3530 的分析和软硬件设计，可以看出，异构的嵌入式处理器能够很好地应用于复杂的嵌入式系统中。目前，异构的嵌入式处理器已经成为一种发展趋势，为复杂的嵌入式系统应用提供了新的解决方案，因此，开发异构的嵌入式处理器具有重要的价值。

参考文献：

[1] 孟宏伟，马建仓，张国强，等. 飞机振动及应变状态监测的无线传感器网络系统研制 [J]. 计算机测量与控制，2014，22 (3)：860－862.

[2] 李文明，张 涛，陈俊江. 无人机载设备状态监测系统的设计与实现 [J]. 长春理工大学学报，2008，31 (2)：17－20.

[3] 曹 霞，黄圣国. ACMS 的飞机状态监控新概念 [J]. 江苏航空，2000，(3)：36－37.

[4] Gepner P，Kowalik M F. Multi-core processors：New way to achieve high system performance [A] . Parallel Computing in Electrical Engineering，2006. PAR ELEC 2006. International Symposium on [C] . IEEE，2006：9－13.

[5] Morozov S，Tergino C，Schaumont P. System integration of Elliptic Curve Cryptography on an OMAP platform [A] . Application Specific Processors (SASP)，2011 IEEE 9th Symposium on [C] . IEEE，2011：52－57.

[6] 刘立哲. 基于双核处理器 (OMAP3530) 的嵌入式开发平台研究与实现 [D]. 北京：北京工业大学，2012.

[7] Cui Y，Li B. A Palm-Print Recognition SystemBased on OMAP3530 [A]. Proceeding of IEEE 2010 sixth International Conference on Wireless Communications Networking and Mobile Computing [C]. Chengdu，Sichuan，China，2010：1－4.

[8] 李文婷，林 岩. 基于 OMAP3530 的多媒体信息处理及通信系统设计 [J]. 信息与电子工程，2010，8 (2)：149－154.

[9] 鲁 琴，胡 冰，罗武胜. 基于 OMAP 的无线多媒体传感网图像节点设计 [J]. 计算机测量与控制，2009，17 (9)：1831－1833.

[10] Texas Instruments Incorporated. DSP/BIOS Link User Guide [EB/OL]. //http：www.ti.com，USA：Texas Instruments Incorporated，April 2006.

[11] 栾小飞. OMAPL138 双核系统的调试方案设计 [J]. 单片机与嵌入式系统应用，2012，12 (1)：16－19.