# **2相关技术介绍**

本论文研发的基于LS1024A的海底油气勘探数据采集传输系统软件， 使用PCIe和DMA技术实现CPU和FPGA之间的数据传输，使用多核并行编程技术实现处理器性能的效率最大化，使用事件驱动编程的思想进行业务逻辑设计。本章将介绍本系统在整个研发过程当中所设计到的相关技术，包括嵌入式CPU，PCIe和DMA，多核并行编程技术和事件驱动编程技术。

## 2.1嵌入式处理器

嵌入式处理器是嵌入式系统的核心部件，掘不完全统计，目前全世界的嵌入式处理器品种已经有成千上万种之多。随着人工智能、物联网等行业的快速发展，人们对于嵌入式处理器的需求也不断增高，同时也对嵌入式处理器的运算能力、可扩充能力、系统稳定性、功耗和集成度等各方面提出了更高的要求，为了不断适应各方面需求，嵌入式处理器的体系结构也经历了指令集从复杂指令集运算（Complex Instruction Set Computing，CISC）、精简指令集运算（Reduced Instruction Set Computing，RISC）、显式并行指令集运算（Explicitly Parallel Instruction Computing，EPIC）、到超长指令字指令集运算（Very Long Instruction Word，VLIW）,位宽从4位、8位、16位、32位到64位；寻址空间从64kB到256MB甚至更大；运算速度从0.l MIPS到5000 MIPS甚至更快；常用封装从8个引脚到725个引脚甚至更多。另外，处理器的集成度进一步提高，并且功耗也有了明显的降低；。

### 2.1.1 嵌入式处理器概述

嵌入式处理器可以分为以下几类：嵌入式微处理器（Embedded Microprocessor Unit，EMPU），嵌入式微控制器（Microcontroller Unit，MCU），DSP处理器（Embedded Digital Signal Processor，EDSP）以及嵌入式片上系统（System on Chip）。

嵌入式处理器的基础是通用计算机中的CPU，专门设计在指定装配的电路板上，只保留与嵌入式应用有关的母板功能，但是电路板上必须包括总线接口、ROM、RAM、以及各种外接设备等。目前主要的嵌入式微处理器有X86、DSP、PowerPC、MIPS、ARM系列等。

一般情况嵌入式微处理器具备以下4个特点：

1、实时的多任务处理能力。能够完成实时多任务，并且具有较短的中断响应时间，从而使内部的代码和实时内核的执行时间减少到最低限度；

2、强大的存储区保护能力。由于嵌入式系统的软件结构已模块化，为了避免各软件模块之间出现非预期的交叉作用，必须设计功能强大的存储区保护机制，同时也有利于软件诊断；

3、可扩展的处理器架构。能够最迅速地开发出满足应用的最高性能的嵌入式微处理器；

4、较低的功耗。嵌入式微处理器大都用在便携式场景，比如无线设备，移动通信设备等需要靠电池供电的嵌入式系统，其需要的功耗只有毫瓦甚至微瓦级别。

### 2.1.2 Layerscape系列处理器

Layerscape架构是NXP公司推出的下一代QorIQ LS系列片上系统（SoC）的底层系统架构。从一开始便旨在充分利用新的开发、提取和效率现实条件，Layerscape架构的创建是为了让程序员找到极为轻松的方式“释放”每一块芯片的性能。该架构延伸了当前向多核芯片设计发展的趋势（包含同构和异构），以获得性能最大化，同时也可以提取足够的复杂硬件，以便让软件开发变得高效、可维护、灵巧、快速和相对简洁。简而言之，Layerscape架构可以实现手写汇编语言代码的性能和效率与高等级语言易用性和现代代码可维护性之间的平衡。

Layerscape架构方框图如下所示：

图X：Layerscape系列处理器架构方框图

在QorIQ LS系列中，每一个通信处理器按照逻辑方式划分为三层，如图1所示。通用处理层（GPPL）、加速包处理层（APPL）和快递包（express packet）输入/输出层（EPIL）分别粗略地代表了标准ISO模型的高、中和低层。无论芯片是否采用物理方式进行如此划分无关紧要；程序员认同这种方式，这与单个芯片如何进行分配无关。

在最底层，快递包I/O层提供支持L2+转换功能的所有网络接口之间真正具有决定性的线速性能，并且包含芯片的网络数据报接口（例如Ethernet、Interlaken、Serial RapidIO、HiGig和PCI Express）。重要但不相关的接口（例如USB或者SATA）将不会属于这个接口层的组成部分，但会成为芯片“系统接口”模块的一部分，如图左侧所示（图1）。尽管严格来说，PCI Express并不是网络接口，它通常用作为堆栈中刀片之间的接口，因此也包括在此处。

中间层包含芯片的包处理元素，它们或者是硬接线加速器、可编程引擎或者二者的组合。APPL可以通过传统顺序、同步、完整运行的模型提供客户定义的、自主和附加值功能，通过嵌入式C语言结构化编程实现完全可编程。再次重申，这些元素可以通过定义明确的接口与通用处理器进行通信，采用保留极具价值的开发人员代码的方式，提取它们（和处理器的）执行的详细信息。

最顶层通用式处理器层属于通用性质层，面向用户/开发人员免费提供，用于他们的操作系统、应用、高水平代码和其他附加值、功能。与Layerscape架构的提取、效率和硬件独立性的价值保持一致，这一层可以同时支持Power Architecture和ARM内核。一个人人皆知的事实是，Power Architecture技术通常采用二进位字节顺序，而ARM技术通常属于小端字节顺序，然而Layerscape架构可以轻松地支持这两种技术。

Layerscape架构将如今性能最强的通信处理器与全世界都在采用的相似的模块化、高水平编程模型相互结合。这无需硬件工程设计的高等级别，便可轻松获得高级通信引擎。更加重要的是，它不需要重新学习每个芯片实现的详细细节，可以作为一代QorIQ LS系列器件由其后继产品直接取代。界限分明和定义清晰的编程模型可以在芯片之间、代代之间保存下来，这是基于开发人员的工作构建形成，而非将其视为硬件实现变更任务弃之不用。简而言之，Layerscape架构将开发团队最为重要和最具价值的方面保留了下来：即它独具特色的软件。

### 2.1.3 LS1024A处理器

本论文研究的数据采集传输系统以LS1024A处理器为核心进行设计，LS1024A处理器以LS102MA为基础进行改进，大大提高了处理能力，其I/O接口与飞思卡尔创新的多层总线架构相结合，可跨所有数据接口实现无阻塞的并发事务处理，从而最大限度地减少片上数据包处理延迟，优化了数据包的处理速度。另外，LS1024A利用ARM的高能效核心技术和飞思卡尔的低功耗设计流程，使其实现同类产品中最低的功耗。符合本论文研究系统的低功耗，小型化指标。

LS1024A处理器的架构图如图2.x所示。在LS1024A内部集成两个ARM Cortex A9内核器件，其主要性能特征如下：

* 精简指令集（Reduced Instruction Set Computing，RISC）体系结构；
* 单内核主频达到1.2GHz，共可提供高达6000 DMIPS的处理性能;
* 每个内核都有自己的高速缓存单元，包括32KB的高速指令缓存，32KB的数据缓存以及总共256KB的L2缓存;；
* 一个带宽高达533MHz的16/32位DDR3 SDRAM控制器；
* 一个高速串行外设接口（HS-SPI），带宽高达50MHz，2个从机选择；
* 一个低速串行外设接口（LS-SPI），带宽为16MHz，3个从机选择；
* 两个支持Gen1（2.5Gbps）和Gen2（5Gbps的）的PCIe接口；
* 三个可编程SerDes接口，最高带宽可达5Gbps；
* 64个通用输入输出接口（GPIO）；

图2.x：ARM LS1024A处理器结构框图

## 2.2 PCIe和DMA

PCI-Express（Peripheral Component Interconnect Express），官方简写为PCIe，是一个高速串行计算机总线标准，其原名为“3GIO”，是由Intel公司在2001年提出的，旨在替代老的PCI，PCI-X和AGP标准[]。PCIe是高速串行点对点双通道高带宽传输，连接PCIe两端接口设备分别独享通道带宽，不占用总线带宽，可支持端对端的可靠性传输。在2002年，PCIe交由PCI-SIG（PCI特殊兴趣组织）并由其拟定并发布了PCI-Express1.0标准，才正式改名为“PCI-Express”。在2003年，PCI-SIG发布PCI-Express 1.1标准，进一步对一些规范重新进行声明和定义。在2007年，PC1-Express 2.0标准发布，其中，X1模式的数据传输速率达到了500MB/s，比PCI-Express1.1标准数据传输熟虑提升了一倍。另外，X4模式的数据传输速率达到2GB/s，而最高的X32模式的数据传輪速率甚至能够达到16GB/s。目前最新的PCIe标准PC1-Express 3.0，其比特率为8Gbps，约为上一代标准带宽的两倍。并且增加了数据发射接收均衡、锁相环改善以及时钟数据恢复等一系列十分重要的功能，极大的改善了数据传输和保护性能。

既然PCIe总线标准支持如此高的传输带宽，如果直接使用CPU来负责整个PCIe的数据传输过程，CPU需要从来源把数据的每一个片段拷贝到暂存器，再写到新的地方，这必然导致整个系统的业务处理能力下降。再加上一般的嵌入式CPU主频普遍较低，这时如果再进行大数据传输，CPU的绝大部分资源将会浪费在数据传输上，以至于CPU没有空闲去执行系统的其它业务。通常在这种情况下，我们会选择PCIe专用的内存直接访问机制（Direct Memory Access）来进行实际的数据传输。

DMA（Direct Memory Access）是一种允许外接设备在不使用系统CPU的情况下直接访问系统内存的硬件机制。它允许不同速度的外设进行沟通，不需要依赖CPU的中断负载。CPU只需要对DMA控制器进行相应的配置，后续的传输动作本身由DMA控制器来执行和完成，这样的操作不仅没有增加CPU的工作拖延，反而可以将CPU腾出去执行其它任务，大大提高了系统的处理能力。

DMA的使用在带来高效数据传输的同时也导致了缓存一致性的问題。由于当前的CPU都带有缓存（cache），CPU直接读写cache的数据，而DMA访问的是外部内存中的数据。因此就会出现以下两种情况：

1．CPU读取超前：即当DMA更新完外部内存中的数据后，cache没有进行该数据的更新，导致CPU访问到cache中的旧数据；

2．DMA读取超前：即当CPU更新完cache的数据后，没有将该数据更新至外部内存，导致DMA访问到外部内存中的旧数据。

对于这个问題，一般有两种解决方法：

1．使用一致性内存（Coherent Memory）：由硬件来确保cache数据和外部内存数据的一致性。当DMA向外部内存写数据后，通知缓存控制器更新cache数据，防止CPU读取超前；当DMA向外部内存读数据前，通知缓存控制器将cache数据全部清空到外部内存中，防止DMA读取超前。

2．对于使用非一致性内存（non-coherent memory）的系统，通常使用软件控制的方式来解决。当CPU向cache读数据时，由软件确保cache中的数据是最新的，防止CPU读取超前；当DMA读外部内存中的数据时，由软件来确保外部内存中的数据是最新的，防止DMA读取超前。

## 2.3多核并行编程技术

多核处理器发展的同时对软件行业带来巨大的影响。在过去单核时代，程序员基本上不用考虑并行编程的问题，所以传统的软件大多都是基于串行单线程模式编写，将传统的应用程序直接运行在多核处理器上时，程序只能运行在单个核上，多核的运算能力无法被充分利用，不能发挥出多核处理器高性能的优势。在多核处理器的要件架构中，只有采用软件多进程、多线程技术方能充分发挥硬件的性能，线程分配到每一个核上，使得每一个线程都能得到一个核运行，多个线程并行运行。另外还要注意的是单核的多线程技术和多核多线程的区别，多核多线程技术是在多个物理核上的并行操作，真正意义上的并行执行，而单核上面的并行操作，实际上只是宏观上的一种并发运行，尽管核上有多个线程，但是同一时刻仅有一个线程在运行，多个线程按照时间先后依次运行，并非真正的并行运算。针对这样的问题，人们开始引入并行编程技术，从多个方面对并行编程进行设计和优化。首先针对多核处理器操作系统进行改进优化，比如多核间进程调度算法，核间负载均衡算法等，以更好地支持多核处理器。其次在应用程序设计架构上，除了多线程编程技术得到更加广泛地应用，人们设计实现了一些并行编模型比如OpenMP和MPI。下面将分别介绍多线程编程技术和并行编程模

在Linux编程环境下，Linux系统提供了一套多线程编程接口，即Pthread线程库。该库符合POSIX标准，被各种UNIX操作系统所支持。相比于多进程编程，多线程编程有如下优点：创建线程的效率高，线程可使用其所属进程中的资源，因而创建线程的速度要比创建进程快；线程间IPC通信方式更加简单和高效，由于进程间的地址空间是相互独立的，进程间通信需要操作系统提供特定方式的支持，比如共享内存和域套接字（Domain Socket），但是线程之问是共享全局数据的，可以直接通过全局变量通信；线程间调度切換的代价也小于进程问的切換。而且在一些网络服务器上通常会频繁地响应请求，比如本文设计的视频监控服务器，需要频繁地响应客户端和前端设备的命令或是回应，所以其通常会在短时间内处理大量的务请求。在此环境下我们就可以利用多线程技术在初始化的时候创建一定数量线程的线程池，通常情况下池中线程处于阻塞状态，当服务器收到请求的时候，就从线程地中取出空闲的线程执行该请求，执行完后该线程重新变成阻塞状态，等待下一次请求的到来。当系统比较空闲时，大部分线程都一直处于阻塞状态，线程池可以自动销毁一部分线程以回收资源。相比于传统简单的多线程编程，即“即时创建，即时销毁”，也就是一旦收到一个请求后，创建一个新的线程，然后由该线程执行任务，执行完毕任务后，线程退出。尽管创建线程的时间与创建进程相比已经大大的缩短，但是如果每次线程执行任务的时间都非常短，而且服务器接收到请求的次数非常频繁，那么服务器将处于不断地创建和销毁线程的状态。如果我们将线程执行过程分为三个过程：T1、T2、T3。

T1：线程创建时间；

T2：线程执行时间，包括线程的同步等待的时间；

T3：线程销毁时间；

那么我们可以看出，线程产生和退出的开销所占的比例为(T1+T3)/(T1+T2+T3)。如果线程执行的时间T2很短的话，这比开销可能占到50%左右[]。如果任务执行次数非常频繁的话，这笔开销将是不可忽略的。因此线程地的出现正是着眼于减少线程创建和销毁带来的开销[]。线程地采用预创建的技术[]，基于这种预创建技术，线程池将创建和销毁线程所带来的开销分推到了各个具的任务上，执行次数越多，每个任务所分担到的线程本身开销则越小，提高了多核处理器的利用率，结短了系统的响应时间。

除了多线程编程技术外，还有一些并行编程模型，比如OpenMP。OpenMP是一种面向共享内存以及分布式共享内存的多处理器多线程并行编程语言。它以线程为基础，通过编译指导语句来显示地指导并行化。OpenMP的：执行模型采用Fork-Join的形式，Fork创建新线程或者唤醒已有线程，Join则是多线程汇合。其典型的编程模式如下：

Fork-Join在开始执行的时候，只有一个主线程在运行，当主线程在运行过程遇到需要进行并行计算的时候，这时候一般由OpenMP编译指导语言指示，会派生出子线程来执行并行的任务。在并行执行过程，主线程和派生线程共同工作。在并行代码结東后，派生线程退出或者挂起，不再工作，控制流程重新回到单独的主线程中[]。其工作流程如图2。3所示；主线程在适行过程中遇到并行指导语言，然后派生出4个子线程共同完成任务，其中第四个子线程在运行过中又嵌套地派生出4个子线程完成其任务，新产生的线程组并不会影响原进程的执行，最后子线程完成任务后退出，所有的线程汇合到主线程处[]。

## 2.4 虚拟局域网

事件驱动编程是。。。

## 2.5本章小结

本章重点针对。。。