# 基于 TTA 的异步微处理器设计及其 VLSI 实现

石 伟,陈芳园,王志英,任洪广,苏 博,王友瑞,陆洪毅
(国防科学技术大学计算机学院,湖南长沙 410073)

**摘 要:** 本文针对传输触发体系结构设计了一款异步微处理器.由于异步 TTA 采用分布式的控制方式,数据相关会导致程序执行错误,因此提出了一种数据源选择技术来保证程序执行的正确性,并给出了异步 TTA 的微体系结构与电路实现.最后,在 0.18$\mu$m 工艺下采用基于宏单元的异步集成电路设计方法实现了该异步微处理器.实验结果表明提出的数据源选择技术能够有效保证异步 TTA 微处理器正确执行,同时异步 TTA 计算内核功耗仅为相应同步计算内核功耗的 40%左右.

**关键词:** 传输触发结构; 异步电路; 低功耗; VLSI 设计
**中图分类号:** TP332.2 **文献标识码:** A **文章编号:** 0372-2112 (2011) 02-0395-07

# Design and VLSI Implementation of an Asynchronous Microprocessor Based on Transport Triggered Architecture

SHI Wei, CHEN Fang-yuan, WANG Zhi-ying, REN Hong-guang, SU Bo, WANG You-rui, LU Hong-yi
(*School of Computer, National University of Defense Technology, Changsha, Hunan* 410073, *China*)

**Abstract:** The design of an asynchronous pipelined microprocessor based on transport triggered architecture is introduced. The distributed control in the asynchronous TTA microprocessor, the data dependence might give rise to a wrong result of the executed program. A data source select scheme is proposed to guarantee instructions running correctly on the asynchronous TTA microprocessor. The micro-architecture of the proposed asynchronous processor is presented, and an asynchronous microprocessor is implemented in an 180nm technology. The experimental results show that the implemented asynchronous microprocessor can run correctly, and the asynchronous circuit style can offer a low power solution for future microprocessor design. The power dissipation of asynchronous TTA core takes only about 40% of the synchronous equivalent.

**Key words:** transport triggered architecture; asynchronous circuit; low power; VLSI design

## 1 引言

随着半导体工艺进入深亚微米阶段,同步微处理器设计遇到了前所未有的挑战.复杂时钟网络带来的功耗增加、面积增大、时钟扭曲等问题日益突出.由于同步电路设计遇到越来越多的问题,异步电路设计方法开始得到了更广泛的关注.首先,异步电路采用局部通信模式,以局部握手协议取代全局时钟,不需要设计庞大的时钟分布网络,很好地解决了时钟扭曲等问题.其次,异步电路中空闲电路模块几乎没有动态功耗,整个系统的功耗得到有效控制.最后,异步电路还具有模块化程度高与可重用性好等优点.

国内外大学和研究机构对异步数字电路展开了深入的研究,取得了许多创新成果,设计实现了一系列异步微处理器.英国曼彻斯特大学从 90 年代初期就开始进行异步电路设计研究,并设计实现了与 ARM 系列兼容的 AMULET[1] 系列异步微处理器.其它具有一定影响力的异步微处理器还包括 NSR (Non-synchronous RISC)[2], CAP (Caltech Asynchronous Processor)[3], FAM (Fully Asynchronous Microprocessor)[4], STRiP (Self-Timed RISC Processor)[5]等.

上述异步微处理器基本上都是基于 RISC 结构实现,本文将针对传输触发体系结构(Transport Triggered Architecture, TTA)[6]进行异步微处理器实现.随着 TTA 的异步化,分布式的控制方式(异步握手产生局部控制信号)不能像集中式的控制方式(全局时钟信号作为控制信号)那样保证指令的正确执行时序,从而可能导致错误的执行结果.这也是异步 TTA 微处理器设计过程

收稿日期:2009-09-25;修回日期:2010-01-14
基金项目:国家 863 高技术研究发展计划基金项目(No.2007AA01Z101);国家自然科学基金(No.60873015, No.60773024, No.60903039);国家 973 重点基础研究发展规划(No.2007CB310901);国防科技大学校预研项目(No.JC-08-06-02);新教师博士点基金(No.20094307120012)

中必须解决的一个重要问题.本文首先分析了异步 TTA 微处理器中由指令间的相关所引起的问题,并提出了一种数据源选择策略(Data Source Select,DSS)来消除指令间相关带来的影响;然后给出了数据源选择策略及异步 TTA 微处理器的具体电路实现;最后在 0.18$\mu$m 工艺下设计实现了一款基于异步 TTA 内核的嵌入式微处理器,验证了提出的数据源选择策略能够保证异步 TTA 微处理器高效正确运行.

## 2 相关工作

由 Corporaal[6]提出的 TTA 可以看成传统 VLIW 体系结构的一个超集,它采用更加灵活、更加细粒度的传输触发模式.TTA 结构特别适应于面向密码运算、多媒体运算等特定嵌入式应用的专用指令集处理器(Application Specific Instruction-set Processor,ASIP)设计.鉴于 TTA 的嵌入式应用环境,其功耗问题已经成为一个重要的研究课题.文献[7]研究了 TTA 不同互连网络结构对功耗与性能的影响.文献[8]将编码技术引入到 TTA 的数据传输中,以降低互连网络功耗.文献[9]提出了一种高效的指令压缩技术,使 TTA 计算内核及指令存储器的功耗下降了约 23%.由于异步电路技术能够通过消除电路中的时钟信号及无效的信号翻转来降低功耗,文献[10]将异步电路技术引入到 TTA 的功能单元设计中,从而使得 TTA 计算内核的总体功耗降低了 20%左右.文献[11]基于 TTA 结构设计实现了一款异步原型芯片 FLEETzero,但该原型芯片旨在研究高速的异步交换结构.在 FLEETzero 中,为了保证异步结构正确运行,需要为每一流水段指定统一的逻辑延迟,而相同的段延迟为异步 TTA 设计带来了设计难度增大及性能优势减小等问题.针对文献[10,11]工作的不足,本文设计实现了一款全异步的 TTA 处理器内核,该微处理器内核能够从逻辑上保证其运算的正确性,设计难度较小,具有较强的鲁棒性,并能够体现异步逻辑的性能优势.

## 3 异步 TTA 体系结构

### 3.1 同步 TTA 结构

与 VLIW 、超标量等体系结构中操作触发的原理不同,TTA 结构的核心思想是利用数据传输触发具体操作,即所有功能单元上的任何操作均以数据传输作为唯一的触发激励.TTA 计算内核由取指单元、译码单元、互连网络、寄存器文件及各种功能单元组成.寄存器文件及各种功能单元通过 socket 接口连接到互连网络上,而译码逻辑通过控制 socket 接口的闭合来进行数据交换.TTA 结构中的每个功能单元均包含一个或者多个操作数寄存器、唯一的触发寄存器和若干个结果寄存器.当数据被写入触发寄存器时,它会触发功能单元将操作数寄存器和触发寄存器中的值作为源操作数来完成相应的操作,并将结果写入结果寄存器.TTA 结构微处理器的流水线可以划分为取指(IF)、译码(ID)、传输(MV)和执行(EXE)四段,其中执行段中的功能单元可以进一步采用多级流水方式实现.

TTA 指令将若干 mov 传输指令打包为长指令,分派到多条传输总线,开发多个独立功能部件上的操作并行.TTA 的指令格式如图 1 所示,其包括若干个传输槽及一个长立即数槽,分别用于编码传输操作与长立即数.每个传输槽分为三个子域:条件域(Guard)、源域(Source)及目标域(Destination).条件域描述传输有效的条件,源域、目标域分别描述数据源与传输目标地址.

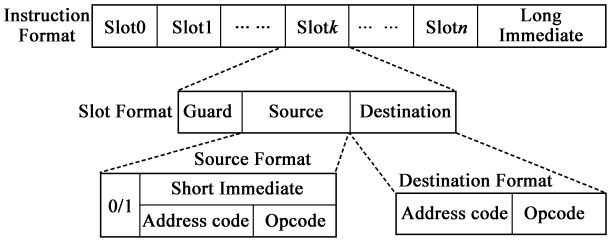


图1 TTA长指令格式

### 3.2 TTA 结构异步设计问题分析

具有流水能力的微处理器中,流水执行的指令之间存在一定的依赖关系,这些相关性不仅会引起流水线冲突的产生,阻塞流水线中的指令流,甚至会导致运算出错.无论同步微处理器还是异步微处理器,相关检测和冲突消除都是重要的研究问题.同步微处理器采用动态调度、动态硬件分支预测、循环展开、静态分支预测等软硬件技术来克服各种冲突,保证程序正确运算的前提下提高流水线的效率[12].文献[13]总结了异步 RISC 处理器中数据冲突及控制冲突的消除方案.

与传统的 RISC 结构不同,TTA 的相关检测及冲突消除均由编译器负责.首先,编译器通过优化调度将必须执行的指令插入到分支指令延迟槽中,从而避免硬件处理控制冲突.其次,在同一条长指令中的不同传输指令不会具有相同的目标地址,避免结构冲突的出现.最后,编译器根据功能单元的运算延迟来安排传输指令,避免数据冲突.

图 2($a$)是同步 TTA 流水结构示意图,如果指令 $i$ 所触发操作的结果将作为指令 $i+j$ 触发操作的操作数,那么在指令 $i+j$ 进入 MV 段时,指令 $i$ 触发的操作已经运算完毕且结果没有被后续指令的结果覆盖.由软件全权负责相关检测及冲突消除的特性简化了 TTA 结构的硬件实现,提高了执行效率,但也为 TTA 的异步化带来了一定的困难.编译负责的相关检测及冲突消除工作是以全局时钟作为基准,而随着同步时钟的消除,异步 TTA 的逻辑行为不再可控,无法按照编译器指

定的方式进行工作,从而导致了程序的错误执行.简单的异步 TTA 流水结构如图 2(*b*)所示,其中指令 *i* 与指令 *i* + 1 触发相同的功能单元进行运算,编译器指定指令 *i* 的结果作为指令 *i* + *j* 的输入.由于缺乏统一的全局时钟进行同步,可能出现指令 *i* + 1 提前完成的情况,即在指令 *i* + *j* 进入 MV 段之前指令 *i* + 1 已经执行完毕.此时,指令 *i* + 1 的结果将替代指令 *i* 的结果作为指令 *i* + *j* 的输入,从而导致了程序的错误执行.

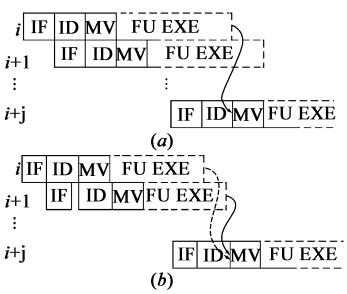

图2　同步及异步TTA流水结构

### 3.3　异步 TTA 结构

采用异步电路实现的 TTA 结构中,由于无法用全局的时钟周期数来衡量异步功能单元的运算延迟,功能单元实际运算时间可能影响后续 mov 指令的运算,即当一条指令译码结束并打开互连网络 socket 时,源 socket 处的数据可能并不是预期的数据,此时会导致错误的数据传输.因此,上述由数据相关及分布式的控制策略引起的执行错误问题是异步 TTA 必须解决的首要问题,本文提出一种数据源选择技术来解决这个问题.

异步 TTA 微处理器的结构如图 3 所示.在 ID 段,指令的条件域译码需要根据比较单元的运算结果来判断相应的传输操作能否执行.在 MV 段,需要将功能单元的执行结果作为数据源传输到目的寄存器,以供新的操作使用.由于 EXE 段的功能单元采用异步流水方式实现,功能单元的运行时间相对于同步电路存在两种情况,(1)功能单元运算较慢;(2)功能单元运算较快.功能单元运算较慢的情况会导致 ID 段译码错误或者 MV 段数据源选择错误,我们可以通过增加 ID 段及 MV 段延迟匹配单元的延迟来解决问题.实际上,同步 TTA 也是采用类似的手段来保证微处理器的正确性,即通过增大全局时钟周期保证功能单元的运算在预期时间内完成.对于功能单元运算快的情况,本文提出两种技术来保证指令正确有序执行.首先,为每条指令增加一个指令序号(Instructions Index);其次,在功能单元的结果寄存器后面再增加一级结果缓冲,防止结果在使用之前被新的运算结果覆盖.图 3 的 ID 段和 MV 段增加了数据源选择逻辑,用于从功能单元结果缓冲中选择正确的数据供译码或数据传输使用.文章 4.2 节与 4.3 节将对数据源选择策略及具体电路实现详细介绍.

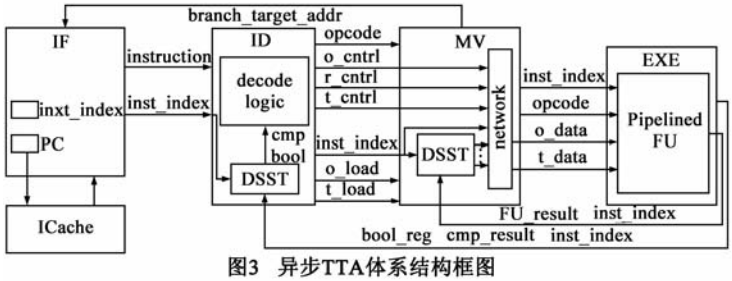

图3　异步TTA体系结构框图

## 4　异步 TTA 微体系结构及电路实现

### 4.1　异步电路设计方法

基于语法驱动转换(Syntax-Directed Translation)的设计方法与控制通路/数据通路划分的设计方法是两种常见的异步电路设计方法.但上述两种设计方法仍存在较多的问题,如(1)需要对现有电路进行重新设计,工作量较大;(2)缺少成熟的异步 EDA 工具;(3)设计人员对设计方法不了解,需要重新学习.我们课题小组结合现有的同步及异步工具提出了一种切实可行的基于宏单元的异步设计流程[14],该方法从已有的同电路描述着手,将其全局时钟网络替换为本地握手电路,然后使用与同步电路设计相同的 EDA 工具和流程来实现相应的异步电路.基于宏单元的异步设计流程采用四段握手数据打包协议实现控制通路.简单四段握手协议(Four-Phase Latch Control, FLC)是异步电路握手通信的最直接实现,但是该协议会使异步流水线出现一段满一段空的现象.为了解决简单四段握手协议性能低的问题,我们课题小组提出了冗余四段握手协议(Redundant FLC, RFLC)[15].

### 4.2　异步 TTA 流水线结构

采用 4.1 节介绍的异步电路设计方法对 TTA 微处理器进行异步设计,图 4(*a*)给出了该异步微处理器的控制通路结构.由于 IF、ID 及 MV 段采用线性流水方式实现,所以其控制通路由三个 RFLC 串联组成.功能单元的控制通路同样由多个 RFLC 控制器串联实现,其级数与相应功能单元的流水线级数相同.MV 段的指令会触发多个功能单元进行运算,因此 MV 段的 RFLC 输出请求信号 Rout 连接到多个功能单元的 RFLC 输入请求信号 Rin 上;此外,只有当所有功能单元都已接受新数据后,才会反馈应答信号给 MV 段,因此所有功能单元的输入应答信号 Ain 通过 C 门运算后再连接到 MV 段 RFLC 的输出应答信号 Aout 上.

在图 4(*a*)所示的控制通路结构中,每条 MV 段的指令将触发所有功能单元进行一次运算,从而可能导致某些功能单元进行了无效运算,浪费了功耗.本文采用图 4(*b*)所示的结构来进一步优化异步 TTA 处理器的流水线结构.在优化后的异步流水线结构中,只有当功能单元的触发信号 tload 有效时,MV 段的请求才会传送到相应功能单元.因此,只有被 mov 指令触发的功能单元才会

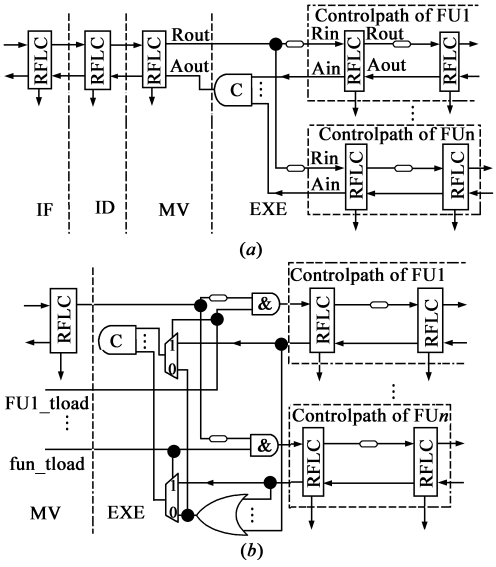


图4 异步TTA控制通路结构图

进行数据运算,消除了无效操作,降低了功耗.

异步 TTA 中的异步功能单元流水线结构如图 5($a$)所示.本文对功能单元数据通路进行了修改,以配合数据源选择策略来解决由数据相关引起的问题.首先,在每个中间结果寄存器中加入了指令序号 $i$ 和数据可用标识位 $v$;其次,在每个功能单元流水线的末端增加了一级结果缓冲.指令序号表明该运算结果是由哪一条指令产生的,而数据可用标识表示该运算结果能否作为数据源传输到互连网络上去.当功能单元计算出结果后,结果首先被锁存到结果寄存器 R1 中去,如果结果寄存器 R2 为空,结果将继续传递到结果寄存器 R2 中去.在这种情况下,只有当结果在 R2 中稳定后才能被使用,从而增加了功能单元的实际运算时间.为了解决上述问题,本文提出了图 5($b$)所示的异步功能单元结构.在优化后的流水线结构中,RFLC 控制器的输出请

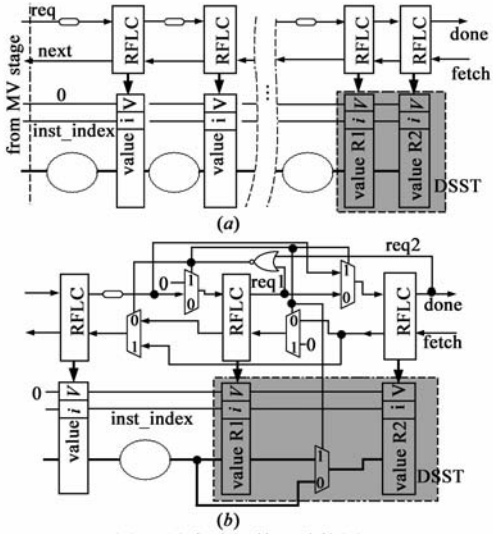


图5 异步功能单元结构图

求信号可以用来表征相应寄存器中的值是否有效,称作数据有效标识位.如果信号 req1 与 req2 都为低,表示 R1 与 R2 中都未锁存有效结果,因此新产生的结果可以直接传输到结果寄存器 R2 中去,而不需要经过 R1 中转,从而降低了时间开销.

为了解决数据相关在异步 TTA 中所引起的问题,本文将各个功能单元的运算结果及其相关信息组成一个数据源选择表(Data Source Select Table,DSST)以供 ID 和 MV 段的数据源选择使用.DSST 的每一项对应一个功能单元,包括该功能单元的运算延迟 $n$、运算结果值 value、指令序号 $i$、结果有效标志 req 及结果可用标志 $v$.每条新的指令进入流水线后都将得到一个指令序号,指令序号由 IF 段中指令序号计数器产生.本文根据 TTA 微处理器中最大流水线长度来确定指令序号编码,如果流水线最大长度为 $N$,则可以采用$\lceil \log_2 N \rceil$位的二进制编码来表示指令序号.

### 4.3 数据源选择策略

为保证 TTA 程序中指令能够按正确的顺序执行,且 MV 段的数据传输能够选择正确的数据源,本文在 4.2 节描述的硬件结构基础上提出了一种数据源选择策略.图 6 为异步 TTA 的数据源选择策略示意图.当功能单元完成一次运算后,结果被锁存在结果寄存器缓冲中,如图中 DSST 所示.MV 段中正在执行指令的指令序号 $I$、结果对应的指令标号 $i$ 和功能单元延迟 $n$ 经过数据源选择逻辑(Data Source Select Logic,DSSL)计算后,得到该结果能否被当前指令使用.如果结果寄存器中的数据能被使用,且该结果对应控制通路的请求信号 req 为高,则将结果可用标识位寄存器 $v$ 复位为高电平.如果功能单元的最新计算结果可用,则选择该结果连接到互连网络上;同时也意味着老的计算结果将不会再被使用,需要将其移出结果缓冲序列.图 6 中使用信号 $s$ 决定将哪个运算结果作为数据源连接到互连网络上.当 $s$ 等于 1 时,结果寄存器 R1 中存储最新运算结果且该结果可用,结果寄存器 R1 的值也将作为数据源连接到互连网络上;否则选择结果寄存器 R2 的值作为数据源.图中 fetch_gen 信号用于为功能单元的 fetch 端

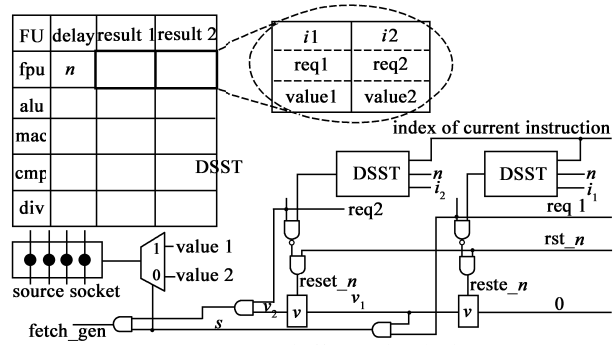


图6 MV段的数据源选择策略

口生成一个取数据脉冲信号,从而将老的运算结果移出结果缓冲,为新的运算结果提供空间.

DSSL用于计算当前指令的序号 $I$ 与结果对应的指令序号 $i$ 之差与功能单元延迟 $n$ 的关系.如果差值大于等于功能单元延迟,则表明该运算结果可以被后续的 mov 指令使用,否则不可用.由于功能单元流水线不会堵塞,且在可用标识位寄存器 $v$ 复位之前 $I$ 与 $i$ 的差距不会超过 $N$,因此指令序号差可以采用公式 $(I + N\text{-}i) \bmod N$ 计算得到.

## 5　异步微处理器实现及评测

### 5.1　异步微处理器实现

为验证提出的异步 TTA 体系结构的正确性,本文设计实现了一款面向多媒体应用的"腾越Ⅱ"嵌入式异步微处理器."腾越Ⅱ"的结构如图 7 所示,该微处理器集成了一个同步 TTA 内核与一个异步 TTA 内核.主处理器可以通过主-从处理器通信接口 Adapter 对计算内核进行控制或查询状态.为了便于验证比较,两个内核采用相同的配置,即具有相同的功能单元及指令集.每个计算内核主要包括 4 个 ALU 单元、4 个乘累加单元、2 个浮点单元、1 个整数除法单元、1 个浮点除法单元、2 个 load/store 单元、2 个比较单元、1 个 I/O 单元以及寄存器文件等,所有的功能单元与寄存器文件通过 8 条互连总线相连.

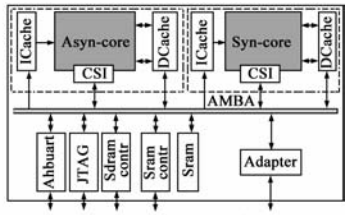

图7　微处理器结构框图

采用基于标准单元的半定制同步设计流程及基于宏单元的异步设计流程对"腾越Ⅱ"异步微处理器进行了物理实现.由于芯片集成了同步及异步两种电路风格,在综合、布局布线时需要对芯片时序进行详细约束."腾越Ⅱ"采用 UMC 0.18$\mu$m 工艺实现,芯片面积为 4.8mm×4.8mm,管脚数为 197.

### 5.2　性能分析

本节首先以浮点功能单元为例分析异步电路对性能的影响情况,该浮点单元采用四段流水方式实现.实现同步浮点单元时,使用 Design Complier 在工艺典型条件(typical case)下对浮点单元进行综合得到其时钟周期为 3.14ns.实现异步浮点单元时,我们需要对其各流水段分别进行综合,得到其流水段延迟分别为 0.5ns、2.89ns、3.14ns 及 1.69ns,然后按照图 5 所示的功能单元结构进行设计实现.由于四段握手协议的固有信号回零特性,异步流水段完成一次完整运算的时间开销要大于流水线实际逻辑延迟.异步流水线的工作性能与流水线的数据输入密集程度相关,而同步流水线的性能则独立于输入数据.当浮点单元进行稀疏数据运算时,其完成一次浮点运算的总时间可达到 8.22ns,远小于同步浮点单元的 12.56ns;而当异步浮点单元完全流水执行时,其一次运算延迟为 14.44ns,大于同步浮点单元操作延迟.

在异步 TTA 处理器中,为了保证操作的正确性,IF、ID 及 MV 段延迟需要根据 TTA 各流水段的最大延迟来确定.在该微处理器的设计实现过程中,我们发现延迟最大的流水段为浮点单元的第三段.因此,在工艺典型工艺条件下,同步 TTA 的时钟周期为 3.14ns;而异步 TTA 的 IF、ID 及 MV 段的等效周期均为 3.61ns.异步 TTA 等效周期的增加主要是由异步握手开销引起的.由上述分析知,典型条件下的完全流水异步 TTA 的性能要差于相同条件下的同步版本.但在一般情况下,同步电路时钟周期需要根据流水段在工艺最差条件(worst case)下的延迟来确定,而异步电路的性能则可以根据环境在电路最好与最差性能之间调节.文献[16]假定异步流水线的实际性能在最好性能与最差性能之间服从正态分布.在考虑了电路实际运行环境以后,同步 TTA 与异步 TTA 的性能比较如图 8($a$)所示.由于同步 TTA 在工艺最好条件(best case)与最差条件下的最大流水段延迟分别为 2.15ns 与 4.64ns,因此其时钟周期设为 4.64ns.异步 TTA 在最好及最差条件下的最大流水段延迟分别为 2.49ns 与 5.34ns,因此异步 TTA 的实际等效时钟周期在 2.49ns 与 5.34ns 之间服从正态分布.从图中可以看出,异步 TTA 在大部分情况下性能要优于同步 TTA.为了定量比较同步与异步 TTA 的性能,我们分别在正态分布的标准方差 $\sigma$ 为 0.4、0.5、0.6 情况下,求出异步性能优于同步的概率,如图 8($b$)所示.从图中可以看出,异步内核在 90% 以上的情况下,其等效周期要小于同步内核时钟周期.

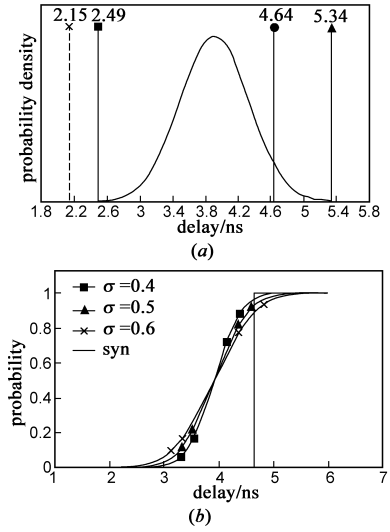

图8　同步与异步TTA内核的性能比较

### 5.3　面积比较

本文分别从逻辑综合及版图实现两方面对同步异步 TTA 内核进行面积比较,比较结果如表 1 所示.逻辑

综合后的同步 TTA 内核面积为 3.42mm$^2$,异步 TTA 内核的面积为 3.57mm$^2$,异步内核面积较同步内核增加了约 4.44%,增加的面积主要是由异步控制通路产生的.另外,异步设计对内核的每一流水段都进行了充分延迟优化,这也使得数据通路的面积有所增加.在对综合后网表进行功能仿真以后,我们使用 Encounter 工具对同步及异步 TTA 内核分别进行了物理实现.由于在布局布线过程中,需要在电路中插入缓冲进行延迟优化,因此布局布线后的内核面积及单元数目较综合后都有所增加.同步内核需要建立一个全局时钟树网络,开销较大;而异步内核则只需要对局部时钟网络进行优化,开销相对较小.最终实现的异步 TTA 的版图面积较同步 TTA 版图面积增加了约 2.01%.

**表 1 同步及异步 TTA 内核面积比较($\mu$m$^2$)**

| | 属性 | 同步 TTA | 异步 TTA | 比例 |
|---|---|---|---|---|
| 逻辑综合 | 单元数目 | 151713 | 163225 | +7.59% |
| | 组合逻辑 | 2651649.03 | 2799977.04 | +5.59% |
| | 时序逻辑 | 765388.51 | 768772.67 | +0.44% |
| | 单元总面积 | 3417037.54 | 3568749.71 | +4.44% |
| 版图实现 | 单元面积 | 3661355.72 | 3780376.56 | +3.25% |
| | 内核面积 | 3869319.18 | 3946997.38 | +2.01% |
| | 内核利用率 | 94.6% | 95.8% | |

## 5.4 功耗分析

本节首先对同步及异步电路的功耗组成进行分析,然后对同步及异步 TTA 内核进行详细功耗比较.同步电路的功耗主要由三部分组成,包括时钟功耗、漏流功耗及数据运算的动态功耗.对于某个同步模块 $i$,其总功耗可以表示为:

$$P_{fu(i)\_syn} = (\Sigma E_{fu(i),j}(f_j)(t_e + P_{fu(i)\_s} + P_{clk}$$

其中,$E_{fu(i),j}$表示模块 $i$ 单次执行第 $j$ 类操作所消耗的动态能量,$f_j$ 表示第 $j$ 类操作执行次数,$t_e$ 表示程序运行的时间,$P_{fu(i)_s}$表示模块 $i$ 的漏流功耗,$P_{clk}$表示由时钟翻转而引起的功耗开销.异步电路没有全局时钟,因此消除了时钟功耗,但是增加了控制通路功耗.模块 $i$ 的异步版本的功耗可以表示为:

$$P_{fu(i)\_asyn} = (\Sigma E_{fu(i),j} \times f_j)/t_e + P_{fu(i)\_s} + P_{cntrl\_path}$$

$$= (\Sigma E_{fu(i),j} \times f_j)/t_e + P_{fu(i)\_s} + (\Sigma E_{cntrl\_path(i),j} \times f_j)/t_e$$

从上述公式可以看出,同步及异步电路的功耗区别主要在于时钟功耗.在同步电路中,很大一部分功耗是由时钟产生的,无论电路是否进行计算行为,都要消耗大量的时钟功耗;而在异步电路中,控制通路的功耗$P_{cntr_path}$只有在数据通路进行数据运算时才会产生.由于 TTA 结构中存在大量的计算功能单元,而只有少数几条传输总线,因此每个工作周期中必然有较多的功能单元处于空闲状态,而异步电路技术能够有效降低这些功能单元的功耗.此外,当整个计算内核处于空闲状态时,异步 TTA 更能体现出其功耗优越性.

为了对同步与异步 TTA 计算内核的功耗进行定量比较分析,本文从微处理器版图中提取出门级网表及延迟信息文件(Standard Delay Format,SDF)进行后仿,然后将门级网表、SDF 文件、后仿产生的 VCD(Value Change Dump)文件及相关库文件输入门级功耗评估工具 PrimePower 进行功耗仿真.为了使得同步及异步内核具有相同的仿真环境,仿真在工艺的典型条件下进行,且同步内核的时钟周期及异步内核的等效时钟周期均设为 4$ns$.图 9 给出了几种目标应用程序在两种内核上的功耗仿真结果.从图中可以看出,当两种内核均处于空闲状态时,同步内核的功耗达到 146mW,而异步内核的功耗只有 7mW.这主要是因为在空闲状态下,异步内核中几乎没有信号翻转活动,只消耗了较小的漏流功耗;而同步内核中时钟信号的翻转产生较大的功耗.当计算内核运行程序时,同步及异步内核的功耗都有所增加,但异步内核功耗仍远小于同步内核功耗,只有同步内核功耗的 40% 左右.

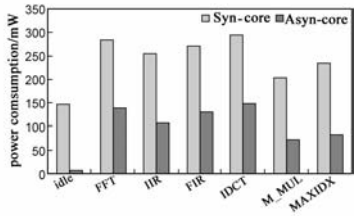

**图9 同步及异步TTA内核功耗比较**

上述功耗分析是在同步及异步电路具有相同性能的前提下进行的,没有综合考虑功耗及性能.Martin 等提出的 ED$^2$P(Energy Delay Squared Product)[17]能够对 VLSI 的性能及能耗进行更有效的度量,且不受电压变化的影响.图 10 给出了本文目标应用程序在同步与异步内核上的归一化 ED$^2$P(normalized ED$^2$P).在本文的实验中,E 表示内核运行某一应用程序时所消耗的能量,而 D 表示内核完成相应程序运行所需要的总时间.图中给出了同步内核在工艺最差条件与典型条件下的归一化 ED$^2$P,还给出了异步内核在工艺最差条件与最好条件下的归一化 ED$^2$P.由于异步电路性能能够根据实际工作环境在工艺最好及最差条件下动态变化,因此异步内核的 ED$^2$P 也将在一定范围内波动;而同步内核的时钟周期一般是由工艺最差情况下的最大段延迟确定,因此同步内核的 ED$^2$P 由工艺最差条件下的能耗及性能确定.从图 10 可以看出,同步内核运行各种程序的 ED$^2$P 最大,其归一化 ED$^2$P 均为 1;异步内核的归一化 ED$^2$P 较同步内核则小得多.此外,异步内核的最大归一化 ED$^2$P 仅稍大于同步内核在典型条件下的归一化 ED$^2$P.

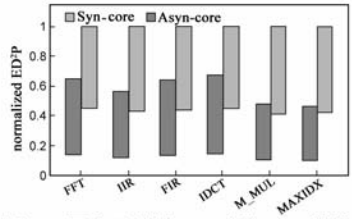

**图10 同步及异步TTA内核ED$^2$P比较**

## 6 结论

随着工艺的不断进步及片上计算资源的不断增加,同步集成电路设计遇到了时钟及功耗等一系列问题,而异步电路设计能够有效地解决这些问题.本文针对传输触发体系结构设计了一款异步微处理器,并提出了一种数据源选择技术来保证程序在异步 TTA 上的正确执行.最后,在 0.18$\mu$m 工艺下采用基于宏单元的异步集成电路设计方法实现了一款集成了同步及异步 TTA 计算内核的微处理器,实验结果表明提出的数据源选择技术能够保证异步 TTA 计算内核正确执行.实验表明异步处理器在基本不增加面积开销的情况下,能够获得较高的性能以及较低的功耗,异步内核的功耗仅为同步内核功耗的 40%左右.

**参考文献:**


[1] Garside J D, et al. AMULET3i-an asynchronous system-on-chip [A]. Proceedings of Async2000 [C]. Washington, USA: IEEE Computer Society, 2000. 162 – 175.

[2] Brunvand E. The NSR processor [A]. Proceedings of the 26th Annual Hawaii International Conference on System Sciences [C]. Washington, DC, USA: IEEE Computer Society, 1993. 428 – 435.

[3] Martin A J, et al. The design of an asynchronous microprocessor [A]. Advanced Research in VLSI: Proceedings of the Decennial Caltech Conference on VLSI [C]. Cambridge, MA, USA: MIT Press, 1989. 351 – 373.

[4] Cho K R, et al. Design of a 32-bit fully asynchronous microprocessor (FAM) [A]. Proceedings of the 35th Midwest Symposium on Circuits and Systems [C]. Washington, DC, USA: IEEE Computer Society, 1992. 1500 – 1503.

[5] Dean M E. STRiP: A Self-Timed RISC Processor [D]. Stanford, CA, USA: Stanford University, 1992.

[6] Corporaal H. Microprocessor Architecture: From VLIW to TTA [M]. West Sussex, England: John Wiley & Sons Ltd, 1999.

[7] Mäkelä R, et al. Analysis of different bus structures for transport triggered architecture [A]. Proceedings of the 21st Norchip Conference [C]. Washington, DC, USA: IEEE Computer Society, 2003. 56 – 59.

[8] Pionteck T, et al. Hardware evaluation of low power communication mechanisms for transport-triggered architectures [A]. Proceedings of the 14th IEEE International Workshop on Rapid Systems Prototyping (RSP'03) [C]. Washington, DC, USA: IEEE Computer Society, 2003. 141 – 147.

[9] 赖明澈,王志英,等.基于代码特征分析的 TTA 指令压缩技术与解压部件实现[J].电子学报,2008,36(11):2234 – 2238.
Lai Mingche, Wang Zhiying, et al. Characteristics analysis—based code compression and one-cycle decompression engine for transfer triggered architecture [J]. Acta Electronica Sinica, 2008, 36(11): 2234 – 2238. (in Chinese)

[10] Li Y, et al. Design of a low-power embedded processor architecture using asynchronous function units [A]. Asia-Pacific Conference on Advances in Computer Systems Architecture (ACSAC 2007) [C]. London: Springer Berlin, 2007. 354 – 363.

[11] Coates W S, et al. FLEETzero: An asynchronous switching experiment [A]. Proceedings of Async2001 [C]. Washington, DC, USA: IEEE Computer Society, 2001. 173 – 182.

[12] Hennessy J L, et al. Computer Architecture: A Quantitative Approach, Third Edition [M]. San Francisco, CA: Morgan Kaufmann Publishers, 2003.

[13] Chang M C, et al. Design of an asynchronous pipelined processor [A]. 2008 International Conference on Communications, Circuits and System (ICCCAS 2008) [C]. Washington, DC, USA: IEEE Computer Society, 2008. 1226 – 1229.

[14] 李勇,王蕾,等.一种 32 位异步乘法器的研究与实现[J].计算机研究与发展,2006,43(10):2152 – 2157.
Li Yong, Wang Lei, et al. Research and implementation of a 32-bit asynchronous multiplier [J]. Journal of Computer Research and Development, 2006, 43(10): 2152-2157. (in Chinese)

[15] 龚锐.异步乘法器设计与实现关键技术研究[D].长沙:国防科学技术大学,2005.

[16] Andrikos N, et al. A fully-automated desynchronization flow for synchronous circuits [A]. Proceedings of the 44th ACM/IEEE Design Automation Conference [C]. New York, USA: ACM Press, 2007. 982 – 985.

[17] A Martin, et al. $ET^2$: A Metric for Time and Energy Efficiency of Computation [M]. Norwell, MA, USA: Kluwer Academic Publishers, 2002.


**作者简介:**


**石　伟**　男,1982 年生于江苏涟水,国防科学技术大学计算机学院博士研究生,研究方向为低功耗微处理器设计及异步电路设计.
E-mail: shiwei@nudt.edu.cn

**陈芳园**　女,1982 年生于湖北钟祥,国防科学技术大学计算机学院博士研究生,研究方向为高性能计算机体系结构设计.

**王志英**　男,1956 年生于山西长治,国防科学技术大学计算机学院,博士生导师,研究方向为高性能计算机体系结构设计、计算机虚拟化和信息安全等.