

# AzureMIPS——顺序双发射 11 级流水处理器

## NSCSCC2022 决赛答辩

李睿潇、朱元依、张政镒、李少群

复旦大学

2022 年 8 月 20 日

## ① 工作介绍

## ② 处理器介绍

## ③ 外设

## ④ 系统设计



## ① 工作介绍

## ② 处理器介绍

## ③ 外设

## ④ 系统设计



# 开发语言



(a) Scala



(b) SpinalHDL

图 1: Scala 与 SpinalHDL

参数化、灵活、可读性高。

SpinalHDL 相比于 Chisel 更有独特的优势

## ① 工作介绍

## ② 处理器介绍

处理器整体架构  
流水线设计  
分支预测设计  
高速缓存设计

## ③ 外设

## ④ 系统设计

## ① 工作介绍

## ② 处理器介绍

处理器整体架构

流水线设计

分支预测设计

高速缓存设计

## ③ 外设

## ④ 系统设计

# 前端架构

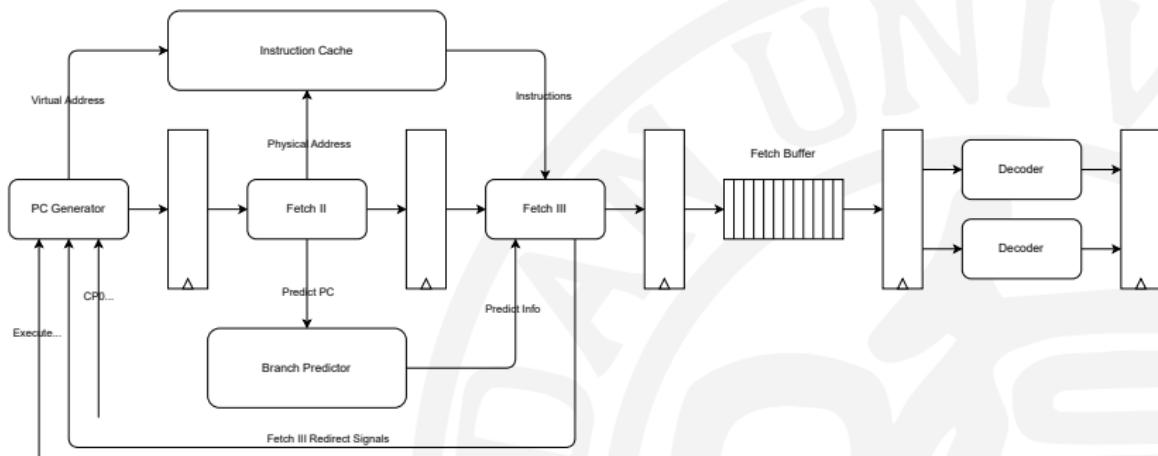


图 2: 前端架构设计简图

# 后端架构

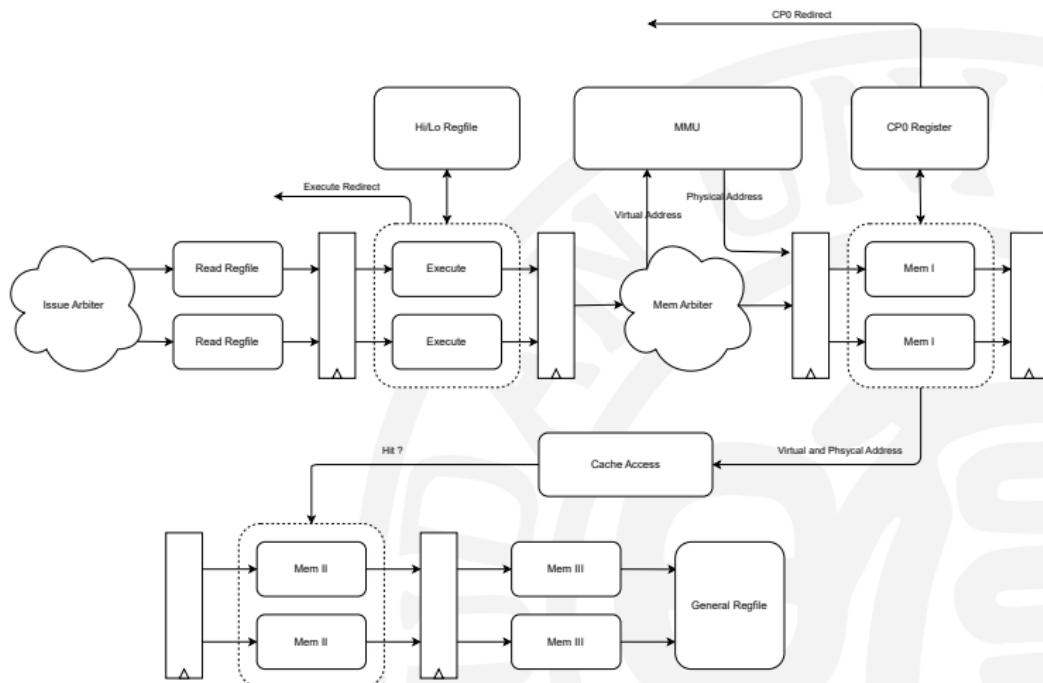


图 3: 后端架构设计简图

## ① 工作介绍

## ② 处理器介绍

处理器整体架构

流水线设计

分支预测设计

高速缓存设计

## ③ 外设

## ④ 系统设计

# 取指

- 取指流水线
  - Fetch 1: 仲裁 PC, 发送虚拟地址至 ICache
  - Fetch 2: 发送通过 MMU 获得的物理地址, 若有异常, 则作标记
  - Fetch 3: 对 ICache 返回的指令作**分支快速译码**, 判断出分支指令, 必要时进行 PC 重定向
- 取指缓冲
  - 16-Entries 指针 FIFO
  - 每周期最多 “4 进 2 出”
  - 跳过非分支延迟槽中的空指令
- 分支预测
  - RAS 分支预测
  - Bi-Mode 分支预测器
  - 256-Entries BTB

# 取指缓冲对双访存的优化

```
1 sw t8, 0(a3)
2 lw t7, 8(t0)
3 nop
4 sw t7, 8(a3)
5 lw t6, 12(a0)
6 nop
7 sw t6, 12(a3)
8 lw t5, 16(a0)
9 nop
10 sw t4, 20(a3)
11 lw t3, 24(t0)
12 nop
13 ...
```

```
1 sw t8, 0(a3)
2 lw t7, 8(t0)
3
4 sw t7, 8(a3)
5 lw t6, 12(a0)
6
7 sw t6, 12(a3)
8 lw t5, 16(a0)
9
10 sw t4, 20(a3)
11 lw t3, 24(t0)
12
13 ...
```

# 译码、发射与读寄存器

- **译码**
  - 译码模块单独占一个流水级
  - 将指令译成对应的 uOp，并译出必要信息
  - 判断是否为保留指令或会触发协处理器不可用的指令，并作标记
- **发射**
  - 组合逻辑，与读寄存器共占一级流水级
  - 进行单发射仲裁
- **读寄存器**
  - 后续流水级旁路和通用寄存器堆间的数据仲裁
  - 当所需数据在流水线中还没被计算出来时，暂停操作
  - **提前计算跳转地址**供执行阶段进行分支预测正确性检验

# 执行

- 执行

- 对于普通运算指令，执行阶段直接计算结果并放入流水线中
- 对于访存指令，执行阶段计算其访存地址
  - 在初赛提交中，本周期还会进行访存地址冲突检查
  - 在决赛提交中，访存地址冲突检查移到单独一级流水
- 对于多周期指令，执行单元请求流水线暂停，直到其执行完毕
- 执行单元直接和 Hi/Lo 寄存器交互，故不需要对 Hi/Lo 寄存器作旁路转发
- 如果指令需要跳转，执行单元会发出跳转信号

# 执行

通过参数化标记,上下两个执行单元有着不一样的功能,但用着同一份代码。

只有上执行单元拥有处理分支指令的能力,去除了无用的冗余,也不引来额外的麻烦。

```
1 class SingleExecute(  
2     advanced : Boolean = true  
3 ) extends Component {  
4     ...  
5 }  
  
6 class Execute extends Component {  
7     ...  
8     val units = Seq(  
9         new SingleExecute(true),  
10        new SingleExecute(false)  
11    )  
12    ...  
13 }  
14 }
```

## 访存

**预访存流水级：**该级为决赛提交中拥有的一级流水级，其主要功能如下

- 地址冲突检查
- 虚实地址转换

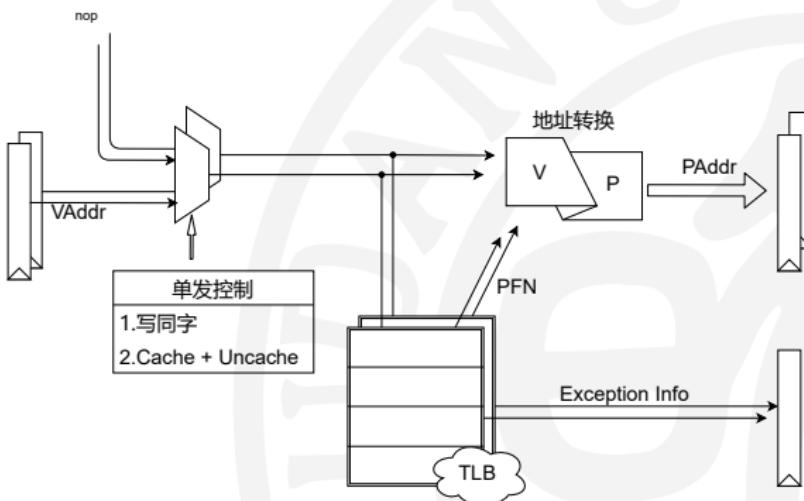


图 4: TLB 转换示意图

# 访存

- 访存第一阶段
  - 向 CacheAccess 发送虚拟地址与物理地址，拉起访存请求
  - 将指令的异常信息交付异常处理单元
- 访存第二阶段
  - 等待 CacheAccess 返回 Hit 信号
- 访存第三阶段
  - 收到返回的数据，写回通用寄存器堆

## ① 工作介绍

## ② 处理器介绍

处理器整体架构  
流水线设计  
分支预测设计  
高速缓存设计

## ③ 外设

## ④ 系统设计

# Bi-Mode 分支预测器

- 借鉴于玄铁 910 的分支预测设计
- 融全局与局部为一体
- 分表存储，降低混叠

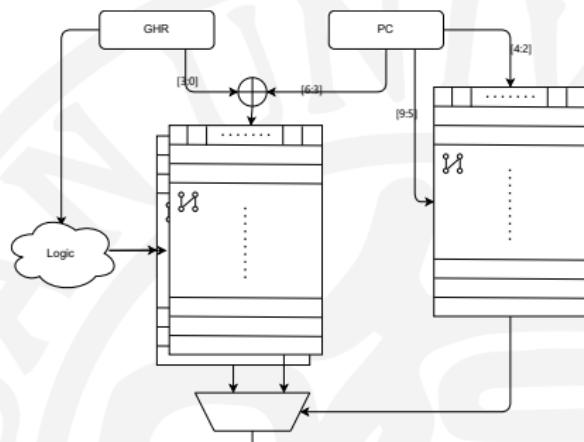


图 5: 分支预测图示

# 分支预测准确率

表 1: 性能测试中的分支预测表现

程序	准确率	程序	准确率
bubble sort	70.1%	coremark	77.3%
crc32	95.3%	dhrystone	96.0%
quick sort	75.5%	select sort	92.1%
sha	97.6%	stream copy	97.1%
stringsearch	88.4%		

## ① 工作介绍

## ② 处理器介绍

处理器整体架构

流水线设计

分支预测设计

**高速缓存设计**

指令高速缓存

数据高速缓存

## ③ 外设

## ④ 系统设计

# 指令高速缓存

- 4 路组相联, 4 Banks, 基于 xpm\_memory
- 每周期可返回 128 bit 数据 (4 条指令)
- 取指合并
  - 所需的 4 条指令不在同一个缓存行中 → 1 次请求访问两条 Cache Line, 将访问结果合并后输出
- 支持 Cache 指令

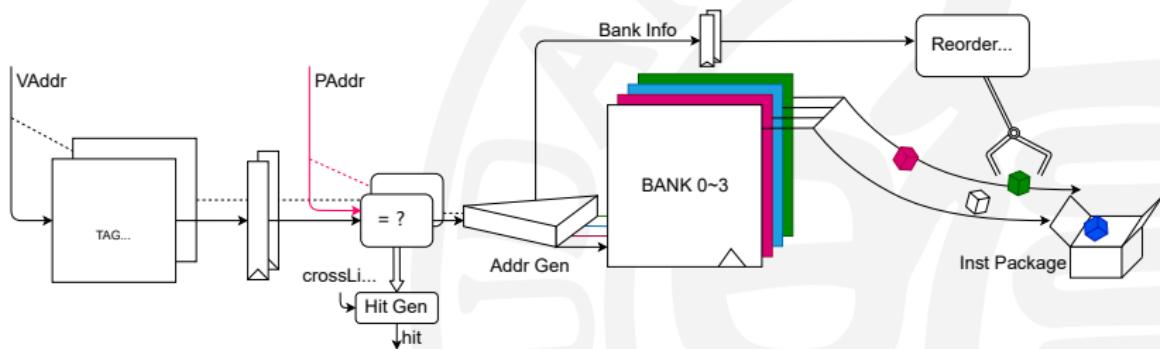


图 6: 指令高速缓存结构示意图

# 数据高速缓存

- 4 路组相联
- 基于 xpm\_memory
- 真双端口，一周期两条访存指令，收益  $\gg$  成本！
- 支持 Cache 指令

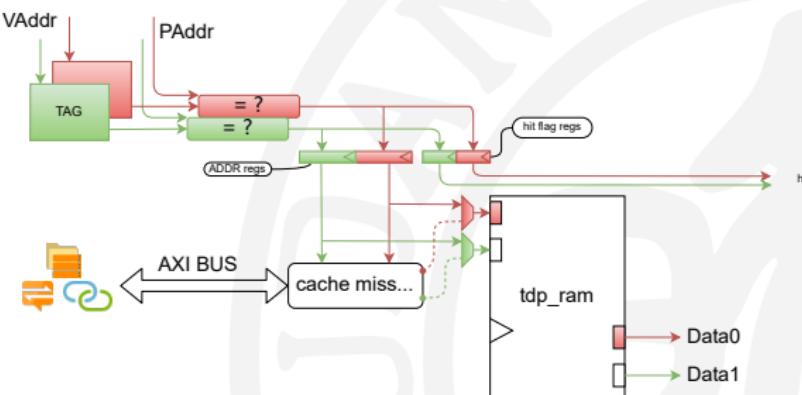


图 7: 数据高速缓存结构示意图

## ① 工作介绍

## ② 处理器介绍

## ③ 外设

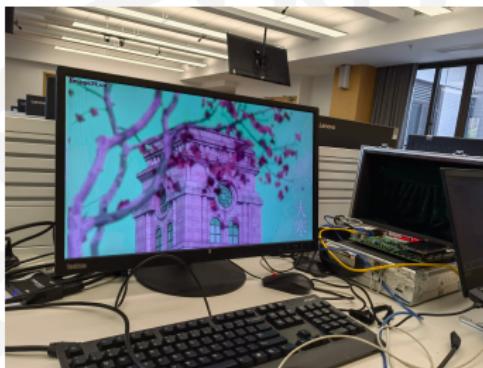
## ④ 系统设计



# VGA



(a) VGA：显示条形图



(b) VGA：显示图片

## 飞机大战游戏演示

## ① 工作介绍

## ② 处理器介绍

## ③ 外设

## ④ 系统设计

PMON

uCore

Linux

## ① 工作介绍

## ② 处理器介绍

## ③ 外设

## ④ 系统设计

PMON

uCore

Linux



# PMON

- 成功启动 PMON
- 正确运行 PMON 的全部指令

```
PMON2000 MIPS Initializing, Standby...
Processor: 100MHz, Cache: 16KB, RAM: 512MB, ROM: 128MB
PREGD=000018000
Processor cache: 16KB, RAM cache: 16KB
postboot caches Found
Init caches done, cfg = 80000082
Copy ROM to execute location...
ROM = 0x80790000
SD = 0x18300000
ADR = 0x8079100010
...and = 0x807910208
ARM processor
copy text section done.
Copy text section execute location done.
SDP=0x80c000
Uncompressing file...
...existing files...
RTC time invalid, reset to epoch.
RTC time: 0
DONE
RTC
ENVX
PMON
On envinit
envvar=0x80000000
value is Invalid!
newvalue=0x80000000
80000000 bytes memory between 82fff800-83000000 is already been allocated, heap is at
83000000
SD000000
...and: gpio status 0
RTC
ATCL
before configure
on configure
localhost status
localhost status
defined at localhost: address 00:98:76:64:32:19
loopbackd at localhost: configure
loopbackd at localhost: ps/2 kb
deconfig done.
deconfig done.
domaininit done.
domainproc...
NETT
SVID
SVID
SVID

PMON2000 Professional
Configuration [/FCR.EL.NET]
Build Date: 2022-08-13 07:54:26 UTC Commit: 43e6479ed
8ff7fda43bdec0514a002bd407d617 Author: zewyyzy <1162551439@qq.com> Date: Sat
Aug 13 07:54:26 2022
Supported loaders [src, elf, bin]
Supported file formats [elf, fat, fx, disk, socket, tty, ram]
This software may be redistributed under the MIT copyright.
Copyright 2000-2002, Aspmic AB, Sweden.
CPU unidentified @ 99.99 mhz / bus @ 99.99 mhz
Primary instruction cache size 16kb (84 Time, 4 way)
Primary data cache size 16kb (64 Time, 4 way)
SEV
SEV
SEV
SEV
SEV in SR set to zero.
Page 13
main: _start .text extext start
PRIME
```

Ready Serial: COM3, 57600 78, 7 78 Rows, 80 Cols

## ① 工作介绍

## ② 处理器介绍

## ③ 外设

## ④ 系统设计

PMON

uCore

Linux



## ① 工作介绍

## ② 处理器介绍

## ③ 外设

## ④ 系统设计

PMON

uCore

Linux

## Linux

- 完成 Linux 的内核启动流程，进入 kernel\_init 函数

```

serial-console - SecondCRT
File Edit View Options Transfer Script Tools Window Help
File Edit View Options Transfer Script Tools Window Help
serial-console 2
Linux version 2.6.32.13 (cgruixiao@sixstars) (gcc version 4.4.0 (GCC) ) #50 Sat Aug 20 02:20:37 UTC 2022
busclock=99999999, cpuclock=33333333, memsize=128, highmemsize=0
bootconsole [early0] enabled
cpu: Intel(R) Dual Band Wireless-AC 7265 4kc
Determined physical RAM map:
    memory: 08000000 0 00000000 (usable)
initrd not found or empty - disabling initrd
20MB RAM available
    Normal 0x00000000 -> 0x00008000
Movable zone start Pfn for each node
early1: 0x00000000 -> 0x00000000
    0: 0x00000000 -> 0x00000000
Built 1 zonelists in ZONE order, mobility grouping on. Total pages: 32512
Kernel command line:
console=ttyS0,115200 rtdns/init root=/dev/hd1a rdparts=nand-flash:10M00(kernel)ro,-(rootfs)

PID hash table entries: 512 (order: -1, 2048 bytes)
Dentry cache hash table entries: 16384 (order: 4, 65536 bytes)
Inode-cache hash table entries: 1812 (order: 3, 32768 bytes)
Primary data cache 16kB, 4-way, VIP, no aliases, linesize= 64 bytes
Primary data cache 16kB, 4-way, VIP, no aliases, linesize= 64 bytes
Micro-assembled Field overflow
Memory: 128MB RAM at 0x00000000 (131072k available) (3116k kernel code, 9576k reserved, 842k data, 1316k init, 0k highmem)
Hierarchical RCU implementation.
NR_IRQS:256
early1: irq init pass!
init_IRQ done.
prior tree done.
init_IPI done.
init_timers done
hrtimer init...
softirq init done!!!
timekeeping will be done
done.
TIME INIT...
-----
time init pass!
profile init...
early boot IRQ's on...
-----
kmem cache init late...
kmem cache init late done.
- CONSOLE INIT -
Console: colour dummy device 80x25

```