CQU Dual Issue Machine

第六届“龙芯杯”全国大学生计算机系统能力培养大赛

重庆大学 所以延迟槽会消失对不队

陈泱宇 李燕琴 王梓宇 张翀

重庆大学
CHONGQING UNIVERSITY

重庆大学
CHONGQING UNIVERSITY

目录
CONTENTS

重庆大学
CHONGQING UNIVERSITY

04 CPU架构设计

CQU Dual Issue Machine

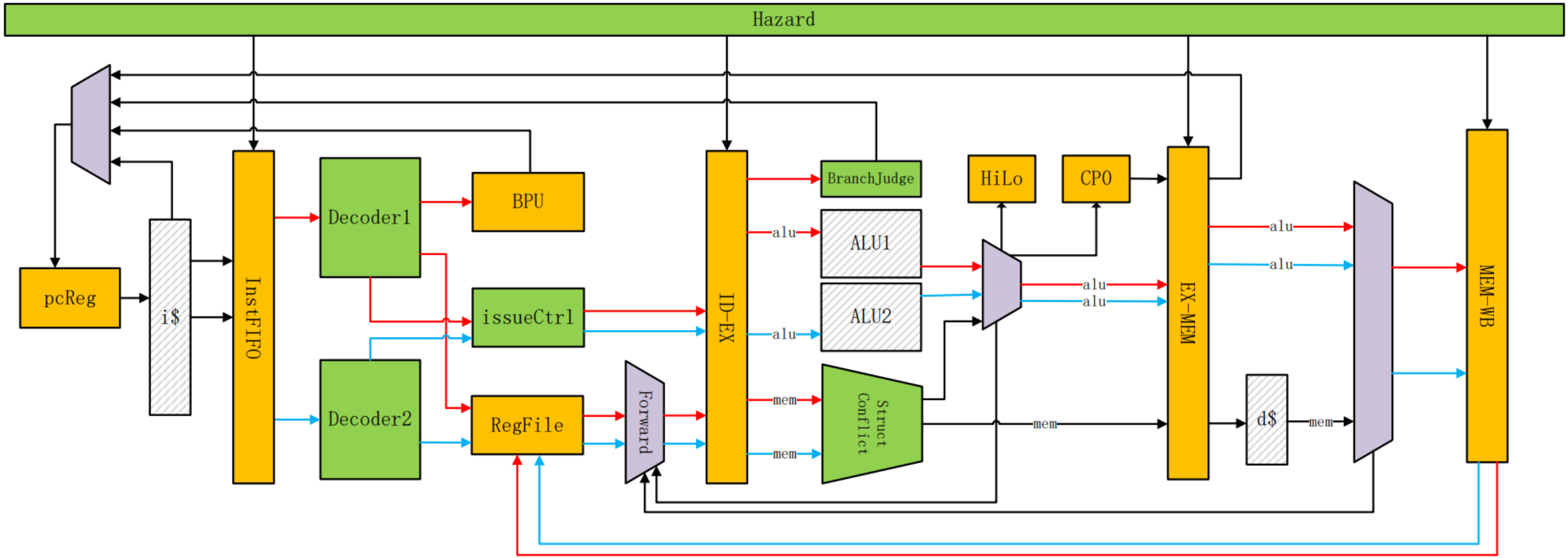
- 对称双发射五级顺序流水线架构（基于System Verilog语言）
  - 双发控制
  - 指令FIFO
  - 2比特分支预测单元
  - 二路组相联和64B缓存行的Cache
  - MMIO Store Buffer
  - 两级TLB
- ISA
  - 除Branch-Likely、浮点指令的 MIPS Release 1 指令

性能结果

最高主频100MHz

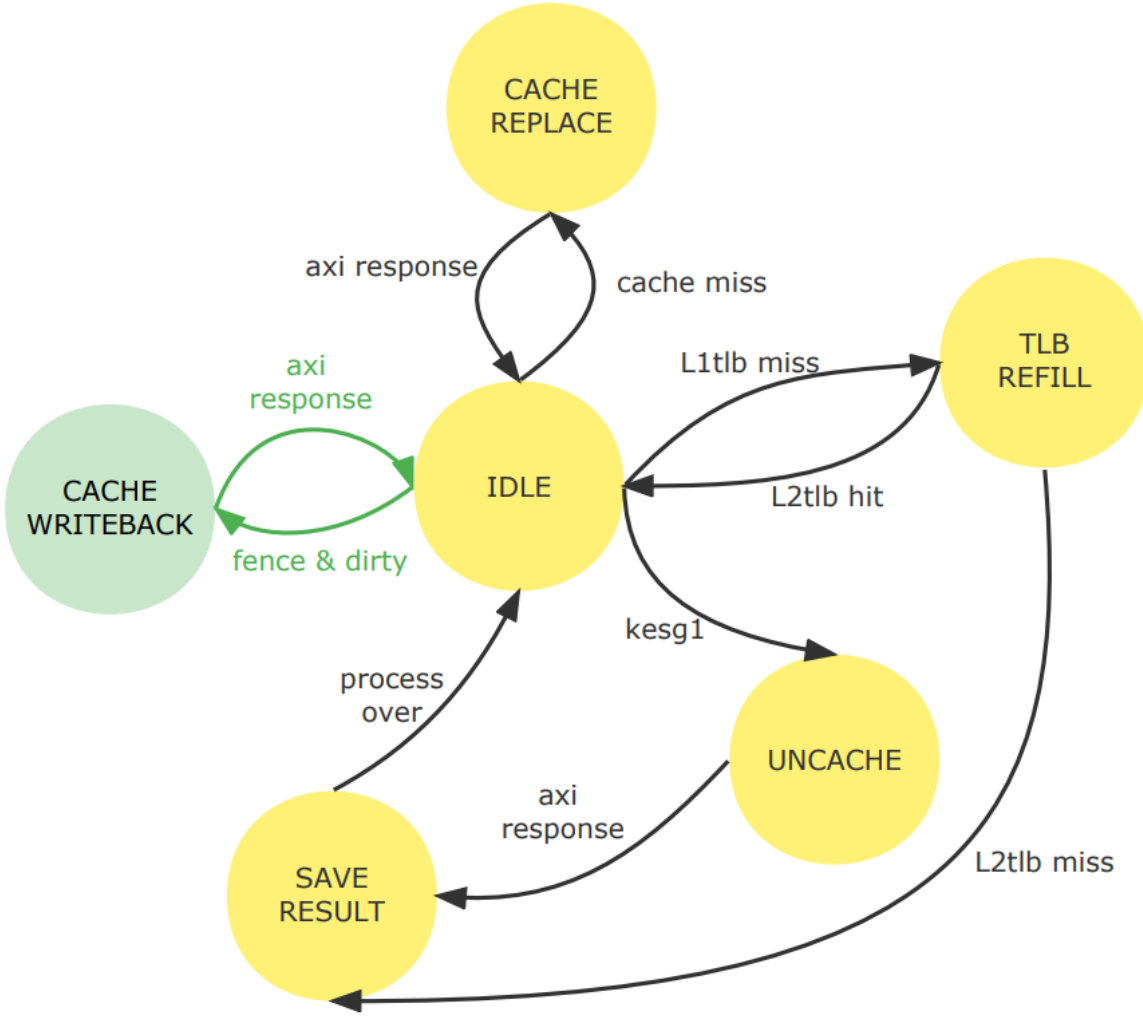
测试程序	性能分	IPC 绝对值	IPC 比值	双发率(仿真)
bitcount	92.503	1.454	45.688	0.665
bubble_sort	82.938	1.089	41.471	0.532
coremark	69.938	1.015	34.970	0.623
crc32	64.572	0.927	32.287	0.597
dhrystone	60.974	0.889	30.496	0.439
quick_sort	69.558	0.999	34.781	0.455
select_sort	65.911	1.016	32.957	0.490
sha	78.993	1.129	39.499	0.605
stream_copy	76.660	1.005	38.375	0.592
stringsearch	68.691	0.964	34.348	0.387
几何平均值	**72.530**	**1.039**	**36.227**	**0.531**

Pipeline结构


Hazard
pcReg
i$
InstFIFO
Decoder1
Decoder2
BPU
issueCtrl
RegFile
Forward
ID-EX
BranchJudge
alu
ALU1
ALU2
mem
Struct Conflict
HiLo
CP0
EX-MEM
d$
MEM-WB

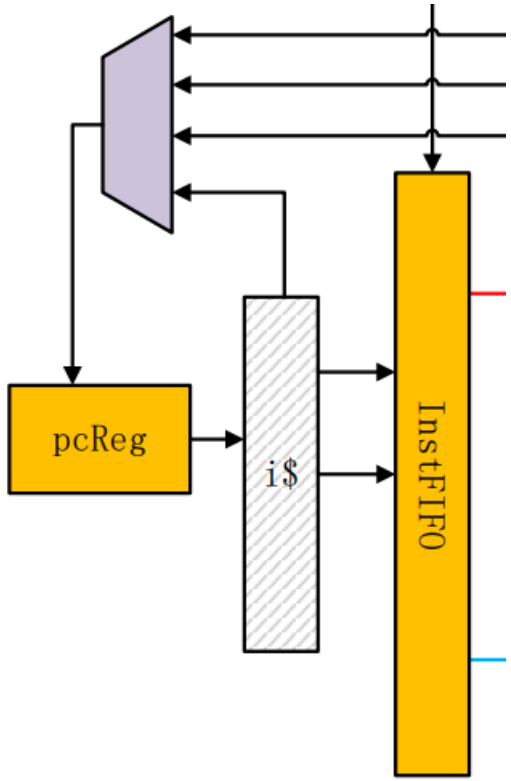
Cache结构

- 指令Cache
  - 2-way, 8KB,VIPT, 命中率99.95%
  - 双端口BRAM, DataLine-2字
- 数据Cache
  - 2-way, 8KB, VIPT, 命中率98.74%
  - 双端口BRAM, DataLine-1字
  - Write Back, Write Allocate
  - 4项MMIO Store Buffer
- 支持16字突发传输和Cache指令



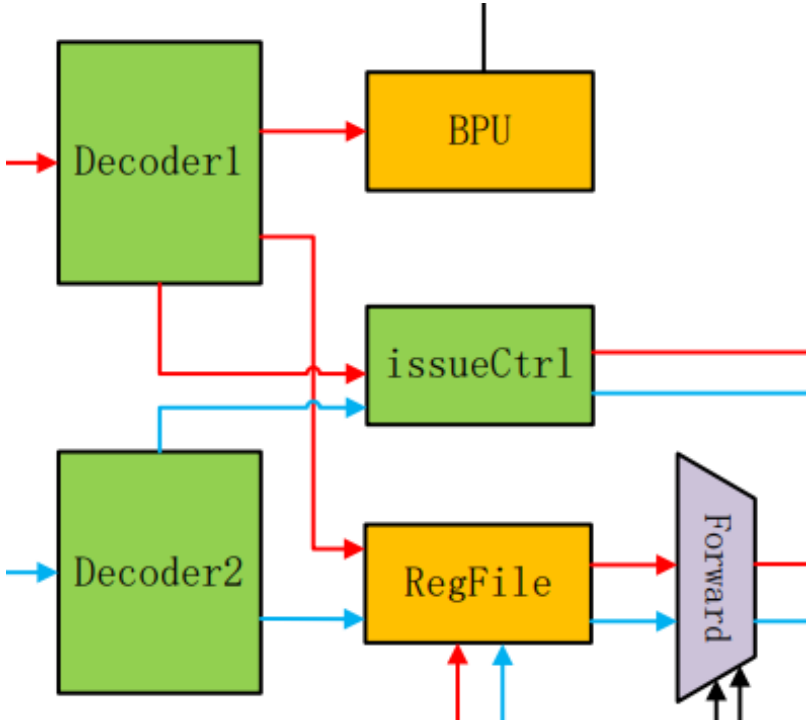
取指结构

- 取指
  - Next PC提前访存，保证PC访存可同周期返回指令
  - 一次取指最多返回2条指令
- **指令FIFO**
  - 可缓存16条指令
  - 取指阶段和后续阶段分离
  - 设置delayslot寄存器，用于缓存未能和分支指令一起发射的延迟槽数据



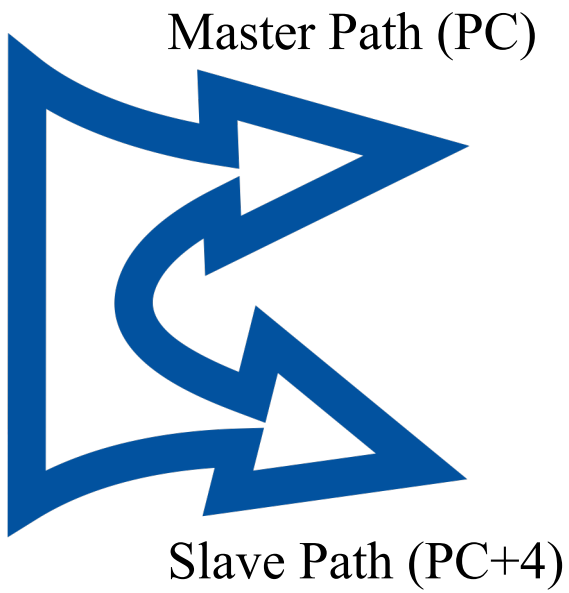
译码阶段

- **分支预测单元**
  - 局部历史预测，2比特饱和更新
  - ID阶段，预测并跳转
  - EX阶段，判断预测结果、解决误判、更新BPU
- **双发控制**
- 数据准备
  - RegFile：4读2写
  - Forward：在进入触发器前完成前推



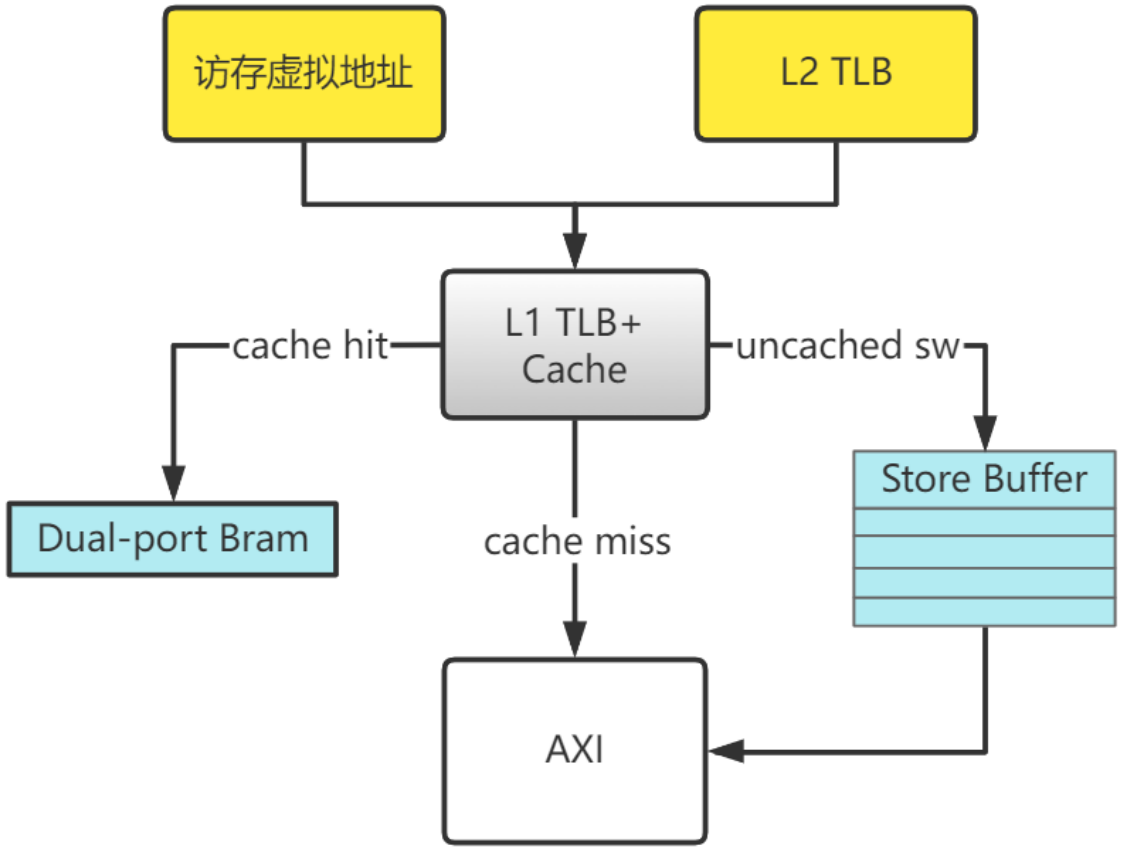
双发控制

- 特殊情况
  - 自陷、例外等指令：放在Master发射，Slave不发射
  - 分支指令：放在Master发射，Slave按照策略发射。
- 数据冲突
  - RAW：Master和Slave的HiLo、CP0、GPR
  - Load To Use：读GPR
- 结构冲突
  - 乘法单元、除法单元、访存单元 → 只发射第一条
- 数据有效性
  - Master发射且第二条指令有效 → 才可发射第二条



访存单元

- 访存逻辑
  - EX阶段，提前访存和 L1 TLB 映射地址翻译
  - MEM阶段，处理访存结果、访存缺失、L1 TLB 缺失并访问 L2 TLB
- **MMIO Store Buffer**
  - 避免Uncached写阻塞流水线
  - Store Buffer非空不允许访问AXI以保证顺序一致的Memory Ordering
- **两级TLB**
  - L1-inst: 1项，L1-data: 1项，L2: 8项(全相联结构)
  - L2提供3个查找端口（TLBP，Inst，Data）
  - TLB不影响频率



重庆大学
CHONGQING UNIVERSITY

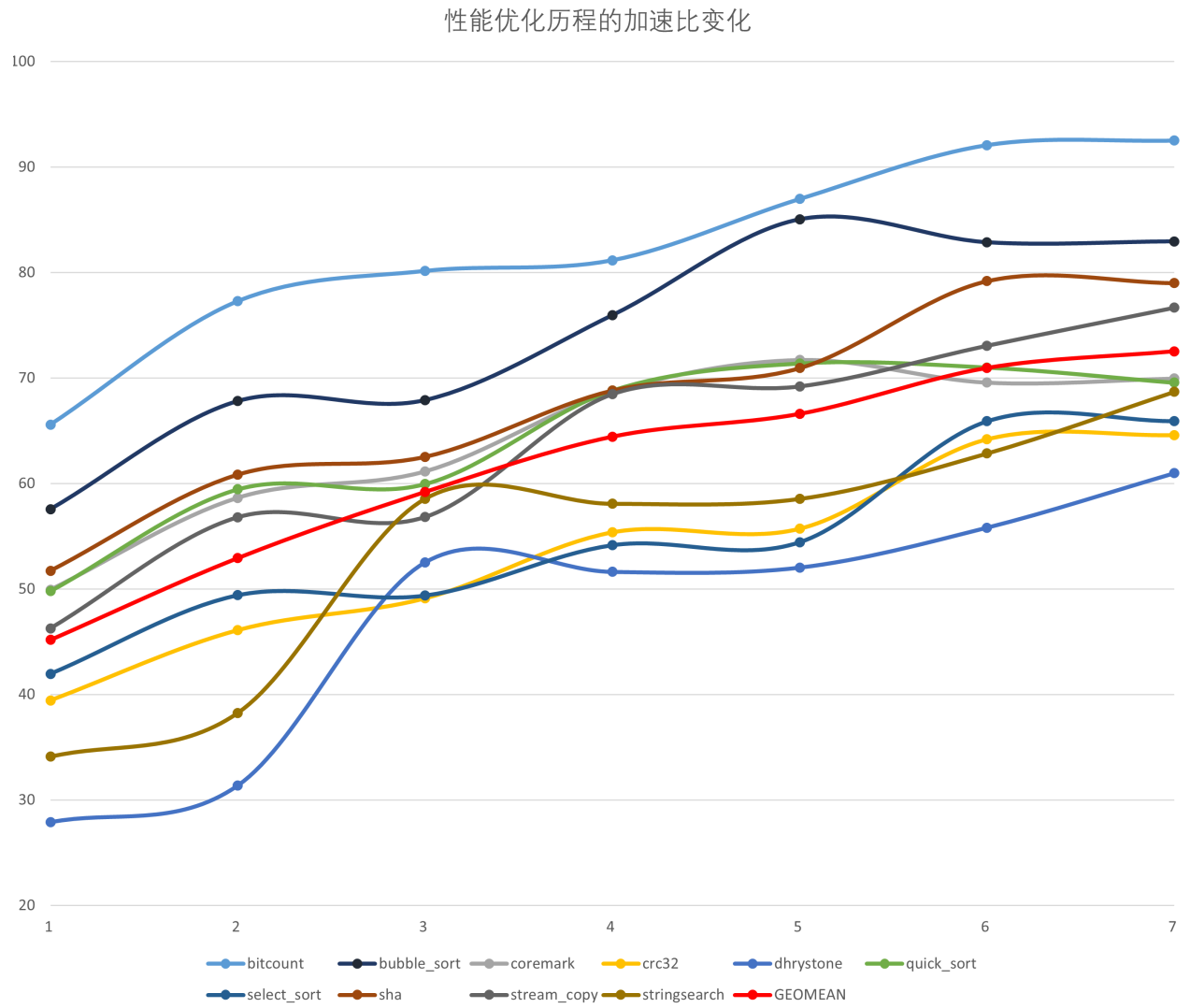
02

优化与调试

优化历程

- 性能优化

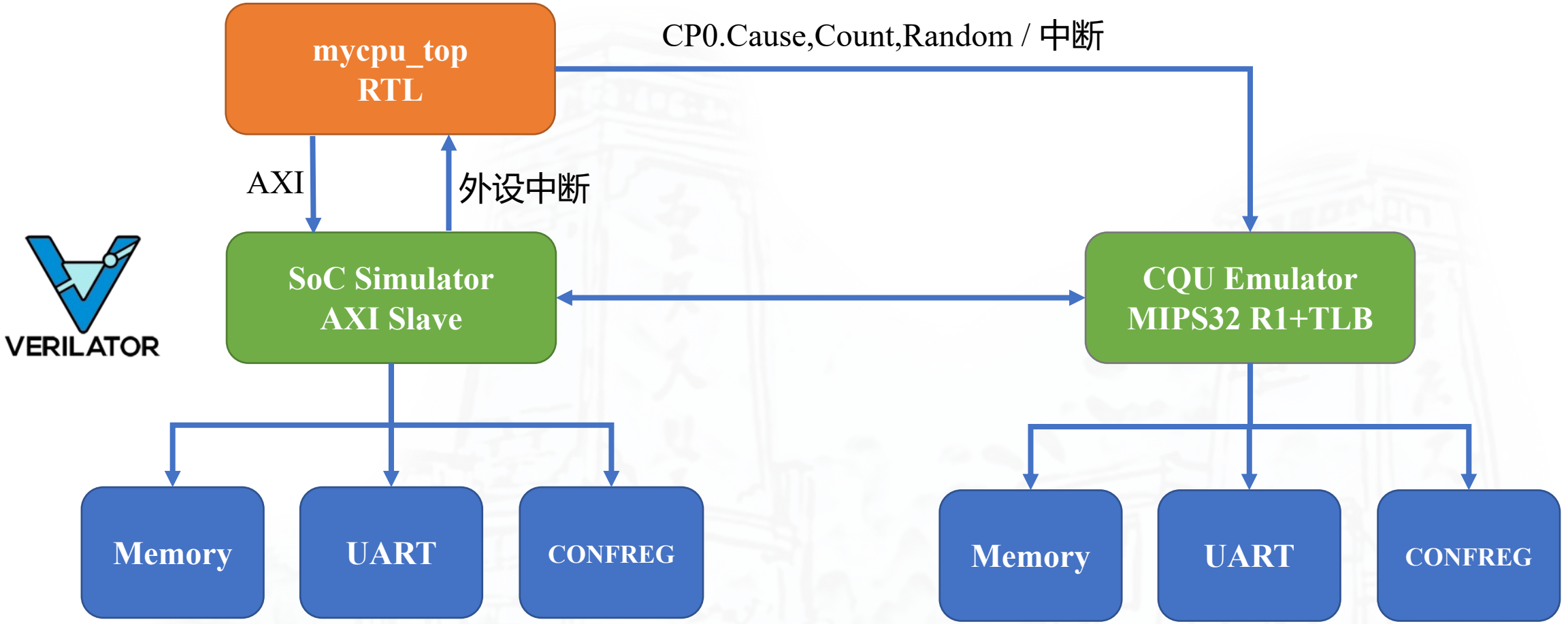
说明	性能分	频率(MHz)
非对称双发射五级流水线	45.164	85
乘法拆分两周期	52.925	100
添加StoreBuffer	59.194	100
减少 load to use 停顿	64.434	100
非对称双发→对称双发	66.596	100
添加分支预测	70.952	100
用尽AXI3突发传输数量	72.530	100



优化历程

- 时序优化
  - Logic Delay
    - ✓ 组合逻辑语句简化
    - ✓ CPU参数调整
    - ✓ 重定时（如增加流水级或计算周期数）
  - Net Delay
    - ✓ 减少扇出
    - ✓ 简化部件连接关系

差分测试框架



https://github.com/cyyself/soc-simulator https://github.com/cyyself/cemu

SoC Simulator 介绍

- 功能（比对）测试
- 性能（差分）测试
- 系统测试
- uCore（差分）运行
- Linux（差分）运行

支持 difftest 的 debug 信号集：

```
      output wire[31:0] debug_wb_pc,
      output wire[3:0] debug_wb_rf_wen,
      output wire[4:0] debug_wb_rf_wnum,
+     output wire[31:0] debug_wb_rf_wdata,
+     // soc-simulator + cemu debug interface
+     output wire [31:0] debug_cp0_count,
+     output wire [31:0] debug_cp0_random,
+     output wire [31:0] debug_cp0_cause,
+     output wire debug_int,
+     output wire debug_commit
```

调试 Linux 期间新增功能：

- 转储CEMU Memory
- AXI写回数据比对
- 条件输出VCD波形图
- PC历史记录

SoC Simulator 介绍

```
757 void linux_run(Vmycpu_top *top, axi4_ref <32,32,4> &mmio_ref) {
758     axi4     <32,32,4> mmio_sigs;
759     axi4_ref <32,32,4> mmio_sigs_ref(mmio_sigs);
760     axi4_xbar<32,32,4> mmio;
761
762     // uart8250 at 0x1fe40000 (APB)
763     uart8250 uart;
764     mmio.add_dev(0x1fe40000,0x10000,&uart);
765
766     // DRAM(128M) at 0x00000000
767     mmio_mem dram(0x8000000);
768     dram.load_binary(0x100000, "../linux/vmlinux.bin");
769     mmio.add_dev(0x0,0x8000000,&dram);
```

```
154 void func_run(Vmycpu_top *top, axi4_ref <32,32,4> &mmio_ref) {
155     axi4     <32,32,4> mmio_sigs;
156     axi4_ref <32,32,4> mmio_sigs_ref(mmio_sigs);
157     axi4_xbar<32,32,4> mmio(23);
158
159     // func mem at 0x1fc00000 and 0x0
160     mmio_mem func_mem(262144*4, "./func_test.bin");
161     func_mem.set_allow_warp(true);
162     mmio.add_dev(0x1fc00000,0x100000,&func_mem);
163     mmio.add_dev(0x00000000,0x10000000,&func_mem);
164     mmio.add_dev(0x20000000,0x20000000,&func_mem);
165     mmio.add_dev(0x40000000,0x40000000,&func_mem);
166     mmio.add_dev(0x80000000,0x80000000,&func_mem);
167
168     // confreg at 0x1faf0000
169     nscscc_confreg confreg(true);
170     confreg.set_trace_file("./golden_trace.txt");
171     mmio.add_dev(0x1faf0000,0x10000,&confreg);
```

从Vivado
到仿真

Diagram | Address Editor

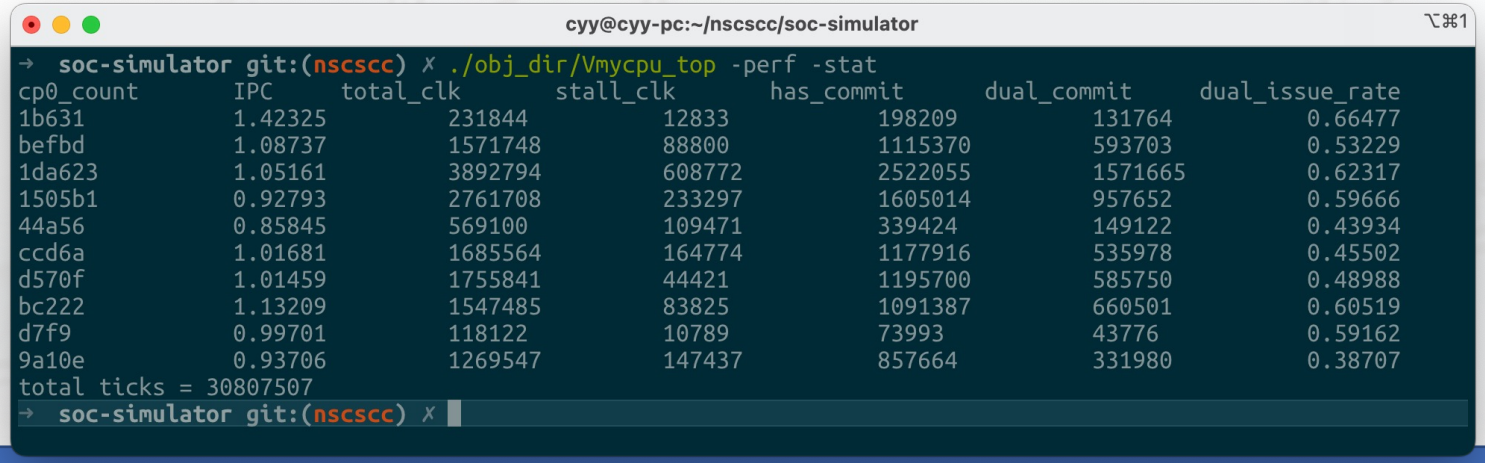
Cell	Slave Interface	Slave Segment	Offset Address	Range	High Address
mycpu_wrapper_0					
MAXI (32 address bits : 4G)					
apb_uart_wrapper_0	axi_s	reg0	0x1FE4_0000	64K	0x1FE4_FFFF
mig_7series_0	S_AXI	memaddr	0x0000_0000	128M	0x07FF_FFFF

用我们的SoC Simulator搭一个仿真SoC就像Vivado Block Design一样简单！

好用的工具是第一生产力

项目	SoC Simulator运行时间	Vivado仿真运行时间
编译CQU Dual Issue Machine	3.881s	~15s
功能测试	1.657s	~1min
性能测试	6.447s （可同时得到统计数据）	>10min
uCore进入shell	~6s	（仿真较困难）
Linux开始执行/sbin/init	~4min	（仿真较困难）

备注：该成绩基于AMD Ryzen 7 5800X，CPU RTL使用CDIM决赛提交版本。



好用的工具是第一生产力

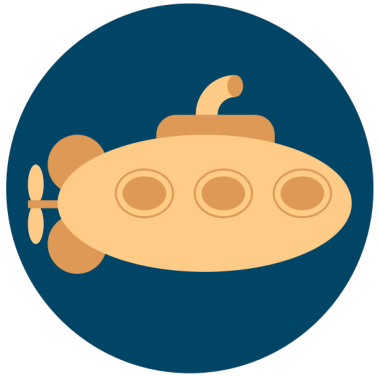
- Dhrystone与string search IPC优化 **+6.27**
  - 通过SoC-Simulator发现这两个测试向confreg串口写了数十KB的数据，每次AXI访问时间20+周期。
  - 设计了MMIO Store Buffer，通过仿真优化Store Buffer项数保证时序与IPC的平衡。
- Load to use Latency优化 **+5.24**
  - 使用SoC Simulator与CEMU的统计结果发现我们的核Load to use停顿严重。
  - 直接转发访存结果、前推逻辑放在EX前以减少停顿周期。
- Linux启动仅用 **2天**
  - 一个白天用printk调到自闭，一个晚上写Linux启动的difftest，第二天调通并进入Busybox。
  - 新增功能可迅速定位错误点、输出错误点附近波形图并保留写入指令后的内存状态用于反汇编。
- 参数优化
  - 不需要等待综合实现即可评估IPC，有助于我们快速trade off各模块参数配置以平衡IPC和时序。

重庆大学
CHONGQING UNIVERSITY

系统软件

引导程序

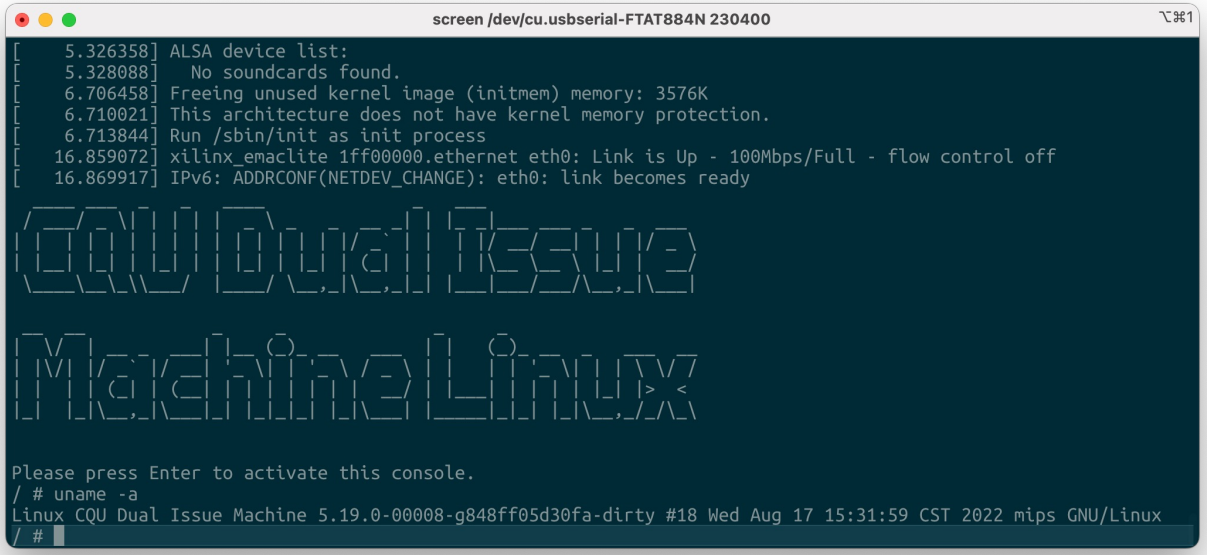
- U-Boot
- 移植处理
  - 去除Branch-Likely指令
  - 编写设备树并配置相关参数
  - 配置Cache Line大小参数用于正确刷新Cache
- SoC-Simulator支持仿真运行U-Boot




```
./obj_dir/Vmycpu_top -uboot                                    ⌥⌘2
➜  soc-simulator git:(nscscc) ✗ ./obj_dir/Vmycpu_top -uboot
<debug_uart>

U-Boot 2019.07-g22f54cca24-dirty (Aug 16 2022 - 13:23:13 +0800)

DRAM:  128 MiB
In:    serial@bfe40000
Out:   serial@bfe40000
Err:   serial@bfe40000
Net:   EMACLITE: bff00000, phyaddr 1, 1/1

Warning: ethernet@bff00000 using MAC address from ROM
eth0: ethernet@bff00000
u-boot@CQU Dual Issue Machine #
```

Linux移植

- Linux v5.19（于2022年7月31日发布）
- 移植处理
  - 编写设备树并配置相关参数
  - 去除Branch-Likely指令
  - 关闭浮点支持
- 指令集处理
  - PERF、SYNC、WAIT等指令实现为NOP
- SoC-Simulator支持仿真运行Linux
  - 仿真到执行/sbin/init仅需4分钟。



在自己的CPU上运行自己CPU的Verilator仿真

- Verilator with CDIM RTL + SoC-Simulator + CQU Emulator
  - 修改Verilator的Makefile，使用mipsel交叉编译工具链

home > linuxbrew > .linuxbrew > Cellar > verilator > 4.216 > share > verilator > include > verilated.mk

```
1   # -*- Makefile -*-
2   ##############################################################################
3   # DESCRIPTION: Makefile commands for all verilated target files
4   #
5   # Copyright 2003-2021 by Wilson Snyder. This program is free software; you
6   # can redistribute it and/or modify it under the terms of either the GNU
7   # Lesser General Public License Version 3 or the Perl Artistic License
8   # Version 2.0.
9   # SPDX-License-Identifier: LGPL-3.0-only OR Artistic-2.0
10  ##############################################################################
11
12  AR = mipsel-buildroot-linux-musl-ar
13  CXX = mipsel-buildroot-linux-musl-g++
14  LINK = mipsel-buildroot-linux-musl-g++
15  OBJCACHE ?=
16  PERL = /home/linuxbrew/.linuxbrew/opt/perl/bin/perl
17  PYTHON3 = /home/linuxbrew/.linuxbrew/opt/python@3.10/bin/python3
```

screen /dev/cu.usbserial-FTAT884N 230400 ⌥⌘7

```
/ # cat /proc/cpuinfo
system type             : MegaSoC,CQU_Dual_Issue_Machine
machine                 : MegaSoC,CQU_Dual_Issue_Machine
processor               : 0
cpu model               : MIPS 4Kc V0.3
BogoMIPS                : 65.79
wait instruction        : yes
microsecond timers      : yes
tlb_entries             : 8
extra interrupt vector  : no
hardware watchpoint     : no
isa                     : mips1 mips2 mips32r1
ASEs implemented        :
Options implemented     : tlb 4kex 4k_cache 32fpr llsc nan_legacy nan_2008
shadow register sets    : 1
kscratch registers      : 0
package                 : 0
core                    : 0
VCED exceptions         : not available
VCEI exceptions         : not available

/ # time ./Vmycpu_top -func
Number 1 Functional Test Point PASS!
Number 2 Functional Test Point PASS!
```

Mainline Linux Kernel v5.19

支持最新主线Linux Kernel

在自己的CPU上跑自己CPU的仿真

“禁止”套娃自举

CEMU

差分测试模拟

SoC-Simulator
仿真4分钟进init

Difftest与Profiler将自主进行到底

性能提11分，2天跑通Linux

64B 超大缓存行

AXI3真的尽力了

多级TLB

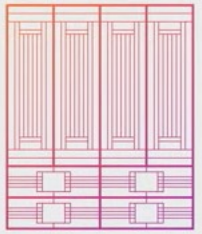
跑系统不降频

CDIM

NSCSCC 2022

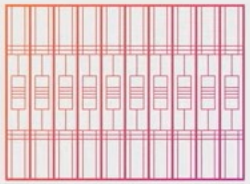
对称双发射五级顺序流水线

几何平均双发率53.1%



Store Buffer

拯救串口写性能



指令FIFO

分离取指单元

1.039

IPC绝对值

36.227

IPC比值

2-bit BPU

精准预测

72.53

GS132加速比

100MHz

提交主频

附录 - IPC计算方式

测试程序	CEMU模拟IPC=1 (cp0_count)	CDIM (cp0_count)	IPC绝对值
bitcount	27d6f	1b668	1.454
bubble_sort	d033f	bf26a	1.089
coremark	1f3529	1ec1fc	1.015
crc32	138751	1512ea	0.927
dhrystone	3b3b2	42a1b	0.889
quick_sort	d0ccc	d101e	0.999
select_sort	d9093	d59c5	1.016
sha	d5821	bd1f6	1.129
stream_copy	df51	de34	1.005
stringsearch	90c58	963d3	0.964
几何平均值			1.039

附录 – NonTrivial MIPS 最终性能结果

- 最高 CPU 主频 123MHz

测试程序	加速比	IPC 绝对值	IPC比值
bitcount	97.886	1.1545	39.308
bubble_sort	85.152	0.9037	34.600
coremark	74.182	0.8278	30.139
crc32	93.048	0.9747	37.805
dhrystone	86.365	0.4983	35.107
quick_sort	75.984	0.8981	30.873
select_sort	101.278	1.2584	41.153
sha	102.928	1.1429	41.823
stream_copy	87.286	0.8598	35.544
stringsearch	94.136	0.5898	38.253
（几何）平均值	**89.326**	**0.8788**	**36.361**

项目展示——TrivialMIPS

备注：引用自2019年NonTrivial MIPS的答辩PPT

附录 - CDIM上运行Linux用户程序的IPC对比分析

对比项目	MT7621 （ newifi-d2 ）	CQU Dual Issue Machine
Core	MIPS 1004Kc	CQU Dual Issue Machine (CDIM)
Frequency	880 MHz (ASIC)	66.666 MHz (FPGA SoC)
TLB	32 entries	L1-I: 1, L1-D: 1, L2: 8 entries
Cache	L1I 32KB 4-way line 32Bytes L1D 32KB 4-way line 32Bytes L2 256KB 8-way line 32Bytes	L1I 8KB 2-way line 64Bytes L1D 8KB 2-way line 64Bytes
Linux Kernel	5.10 (OpenWRT)	5.19
运行md5sum Vmycpu_top时间	0.09s	0.88s
运行Verilator仿真CDIM功能测试时间	10.06s	328.43s
md5sum相对IPC	1	1.350
Verilator相对IPC	1	0.404

备注：使用的Vmycpu_top采用静态编译完全相同的二进制文件，所涉及文件IO均存放于内存文件系统中。