



دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر
ساختمارو زبان کامپیوتر دکتر اسدی

عنوان پژوهه امتیازی:

پیاده سازی ساده MIPS

اعضای گروه

نادر احمدی ۰۳۱۰۵۶۷۴

محمد پارسا محمودآبادی آرانی ۰۳۱۰۶۶۷۹

سید امیرحسین هاشمی شاهروdi ۰۳۱۷۰۲۷۱
پارسا پاک ۰۳۱۰۵۸۲۲

۱۴۰۴ بهمن

چکیده

در این پژوهه، یک شبیه‌ساز نرم‌افزاری برای زیرمجموعه‌ای از دستورات معماری MIPS32 طراحی و پیاده‌سازی شده است. هدف اصلی، مدل‌سازی دقیق اجزای سخت‌افزاری پردازنده شامل حافظه اصلی (RAM)، مجموعه ثبات‌ها (Register File) و شمارنده برنامه (PC) در یک محیط نرم‌افزاری است. این شبیه‌ساز قادر است دستورات ماشین را در قالب هگزادسیمال دریافت کرده و با طی کردن چرخه کامل Fetch-Decode-Execute، تغییرات متناظر را در وضعیت ثبات‌ها و حافظه اعمال کند.

۱ مقدمه

هدف این پژوهه، درک عمیق‌تر از نحوه عملکرد پردازنده در سطح کد ماشین است. شبیه‌ساز طراحی شده با زبان پایتون، دستورات ماشین را خوانده، آنها را رمزگشایی کرده و اجرا می‌کند. معماری پشتیبانی شده زیرمجموعه‌ای از MIPS32 است.

۲ اجزای سیستم

- حافظه اصلی: آرایه‌ای ۱۰۲۴ عضوی از نوع ۳۲ بیتی (۴ کیلوبایت)، به صورت بایت‌آدرس‌پذیر.
- مجموعه ثبات‌ها: شامل ۳۲ ثبات ۳۲ بیتی که ثبات صفر همواره مقدار صفر دارد.
- شمارنده برنامه: نگهدارنده آدرس دستور جاری (شروع از صفر).

۳ قالب دستورات

۱. نوع R: عملیات بین ثبات‌ها مانند ADD, SUB, SLT
۲. نوع I: شامل مقدار فوری و دسترسی حافظه مانند ADDI, LW, SW, BEQ
۳. نوع J: پرش غیرشرطی J

۴ دستورات پیاده‌سازی شده

جدول ۱: جدول دستورات پشتیبانی شده

Operation	Funct	Opcode	Type	Instruction
$R[rt] + R[rs] = R[rd]$	۲۰x۰	۰۰x۰	R	ADD
$R[rt] - R[rs] = R[rd]$	۲۲x۰	۰۰x۰	R	SUB
$R[rt] > (R[rs] = R[rd])$	A۲x۰	۰۰x۰	R	SLT
$Imm + R[rs] = R[rt]$	-	۰۸x۰	I	ADDI
$Imm + Mem[R[rs]] = R[rt]$	-	۲۳x۰	I	LW
$R[rt] = Imm + Mem[R[rs]]$	-	B۲x۰	I	SW
$\text{if } R[rs] == R[rt] \text{ then } PC = PC + Imm$	-	۰۴x۰	I	BEQ
$(\text{if } R[rt] == R[rs] \text{ then } PC = Addr)$	-	۰۲x۰	J	

۵ چرخه اجرا

واکشی: خواندن دستور از حافظه با توجه به PC

رمزگشایی: استخراج فیلد های مختلف با عملگرهای بیتی

اجرا: انجام عملیات یا تغییر PC

توقف: در صورت واکشی دستور صفر

۶ نمونه آزمایش اولیه

کد نمونه زیر برای بررسی صحت عملکرد دستورات اجرا شد:

20080005 20090003 01095022 0109582A AC0A0000 00000000

توضیح عملیات:

ADDI \$t0, \$zero, 5 •

ADDI \$t1, \$zero, 3 •

```
SUB $t2, $t0, $t1 •
```

```
SLT $t3, $t0, $t1 •
```

```
SW $t2, 0($zero) •
```

```
> python3 ./mips_sim.py ./sample_input.txt
Final register values:
R00 = 0
R01 = 0
R02 = 0
R03 = 0
R04 = 0
R05 = 0
R06 = 0
R07 = 0
R08 = 5
R09 = 3
R10 = 2
R11 = 0
R12 = 0
R13 = 0
R14 = 0
R15 = 0
R16 = 0
R17 = 0
R18 = 0
R19 = 0
R20 = 0
R21 = 0
R22 = 0
R23 = 0
R24 = 0
R25 = 0
R26 = 0
R27 = 0
R28 = 0
R29 = 0
R30 = 0
R31 = 0
```

شکل ۱: خروجی نهایی ثبات‌ها در کنسول

۷ راهنمای اجرا

اجرای خط فرمان

```
python3 mips_sim.py sample_input.txt
```

اجرای رابط گرافیکی

```
python3 mips_gui.py
```

۸ نمونه تکمیلی: مجموع ۱ تا ۳۰۰

20030000 20010001 2002012D 00611820 20210001 10220001 08000003 00000000

نتیجه نهایی:

R3 = 45150 •

۹ نتیجه گیری

پیاده‌سازی این شبیه‌ساز باعث درک بهتر ساختار پردازنده، چرخه اجرا و مدیریت حافظه شد. چالش اصلی پروژه، پیاده‌سازی صحیح پرس‌ها و مدیریت دقیق آدرس دهی حافظه بود.