به نام خداوند بخشنده مهربان

معماری کامپیوتر

دانشکده مهندسی کامپیوتر

دکتر اسدی
بهار ۱۴۰۳

امیرحسین صوری، ۴۰۱۱۰۶۱۸۲ و مهدی علی نژاد، ۴۰۱۱۰۶۲۶۶

تمرین پنجم عملی

سوال ۱

نوشتن استک مجازی برای پردازنده و تست آن با انجام یک محاسبه ی عددی

برای تست پردازنده ی ساخته شده، با استفاده از دستورات حافظه، یک استک در دیتا ممووری ساختیم و با انجام یک محاسبه ی پیچیده ی عددی، آن را تست کردیم. محاسبه ی مد نظر به فرمت زیر بوده:

$$(۲۹ \wedge ۵-) \oplus ۲۱) \vee (۲۳ - ۳۰)$$

		start		
000		ADDi	r1, r0, 50	r1 = sign_extend(50)
001		ANDi	r1,r1,-1	keeping only the 50
002		ADDi	r2, r0, 30	r2 = 30
003		JAL	push	push r2
004		ADDi	r2, r0, 23	r2 = 23
005		JAL	push	push r2
006		JAL	pop	pop r2
007		ADD	r5, r2, r0	r5 = r2
008		JAL	pop	pop r2
009		SUB	r2, r5, r2	r2 = r5 - r2
00A		JAL	push	push r2
00B		ADDi	r2, r0, 21	r2 = 21
00C		JAL	push	push r2
00D		ADDi	r2, r0, -5	r2 = -5
00E		JAL	push	push r2
00F		ADDi	r2, r0, 29	r2 = 29
010		JAL	push	push r2, stack complete
011		JAL	pop	pop r2
012		ADD	r5, r2, r0	r5 = r2
013		JAL	pop	pop r2
014		AND	r5, r2, r5	r5 = r2 & r5
015		JAL	pop	pop r2
016		XOR	r5, r2, r5	r5 = r2 ^ r5
017		JAL	pop	pop r2
018		OR	r5, r2, r5	r5 = r2 | r5, result ready
019		BEQ	r0, r0, -1	end
01A	push	SB	r1, r2, 0	stroes in memory
01B		ADDi	r1, r1, 1	incrementc stack pointer
01C		JR	r7	return
01D	pop	SUBi	r1, r1, 1	decrement stack pointer
01E		LB	r2, r1, 0	loads the data to reg
01F		JR	r7	return

ابتدا کد را توضیح می دهیم و سپس عملکرد آن را می بینیم. اول از همه در شروع، آدرس استک را در $r1$ قرار می دهیم. این کار را در دو مرحله انجام می دهیم، ابتدا عدد ۵۰ را با دستور ADDi در $r1$ قرار می دهیم. ولی از آنجایی که عدد ۵۰، در ۶ بیت imm ما به صورت یک عدد منفی ظاهر می شود و sign extend می شود، بیت های اضافی آن را با and کردن مقدارش با -1 دور می ریزیم. سپس $r2 = ۳۰$ را مقدار دهی می کنیم و با JAL به تابع push می رویم. کاری که این تابع می کند این است که داده ی موجود در $r2$ را در جایی از حافظه که $r1$ به آن اشاره می کند، می ریزد و سپس $r1$ را یک واحد زیاد کرده و JR می کند به آدرس بازگشت که در $r7$ ذخیره است. سپس $r2 = ۲۳$ و آن را push می کنیم. پس از آن، دو عدد اول عبارتمان را در حافظه داریم و باید آنها را از هم کم کنیم. ابتدا تابع pop را صدا می زنیم که به این صورت کار می کند که ابتدا آدرس در $r1$ را یک واحد کم می کند و به حافظه می رود و جایی که به آن اشاره می کند را می خواند و در $r2$ می ریزد. پس از pop کردن دو مقدار، آنها را تفریق کرده و دوباره در استک push می کنیم. (خط های $۰۰A \sim ۰۰۶$) سپس باقی اعداد را در استک قرار می دهیم، (خط های $۰۰B \sim ۰۱۰$) و پس از آن به همان ترتیبی که اعداد در استک قرار گرفته اند، آنها را خارج می کنیم، عملیات ها را انجام می دهیم و در $r5$ قرلار می دهیم. و برای نشان دادن پایان برنامه، به آن دستور خط ۰۱۹ را می دهیم. همچنین کد هکس هر کدام از دستورات را می توانید در زیر مشاهده کنید:

000		ADDi	r1, r0, 50	1232
001		ANDi	r1,r1,-1	527f
002		ADDi	r2, r0, 30	141e
003		JAL	push	de1a
004		ADDi	r2, r0, 23	1417
005		JAL	push	de1a
006		JAL	pop	de1d
007		ADD	r5, r2, r0	f0ac
008		JAL	pop	de1d
009		SUB	r2, r5, r2	f556
00A		JAL	push	de1a
00B		ADDi	r2, r0, 21	1415
00C		JAL	push	de1a
00D		ADDi	r2, r0, -5	143b
00E		JAL	push	de1a
00F		ADDi	r2, r0, 29	141d
010		JAL	push	de19
011		JAL	pop	de1d
012		ADD	r5, r2, r0	f0ac
013		JAL	pop	de1d
014		AND	r5, r2, r5	faa8
015		JAL	pop	de1d
016		XOR	r5, r2, r5	faaa
017		JAL	pop	de1d
018		OR	r5, r2, r5	faa9
019		BEQ	r0, r0, -1	003f
01A	push	SB	r1, r2, 0	8440
01B		ADDi	r1, r1, 1	1241
01C		JR	r7	efff
01D	pop	SUBi	r1, r1, 1	3241
01E		LB	r2, r1, 0	9440
01F		JR	r7	efff

و عملکرد کد را نیز در waveform های زیر مشاهده می کنید