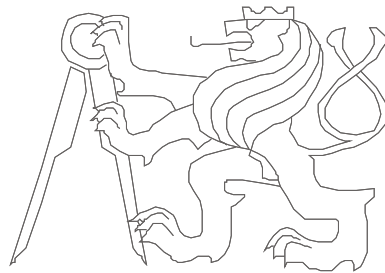


Architektura počítačů

Vývoj architektur mikroprocesorů - od 4 bitů k superskalárnímu RISC

Pavel Píša, Michal Štepanovský



České vysoké učení technické, Fakulta elektrotechnická

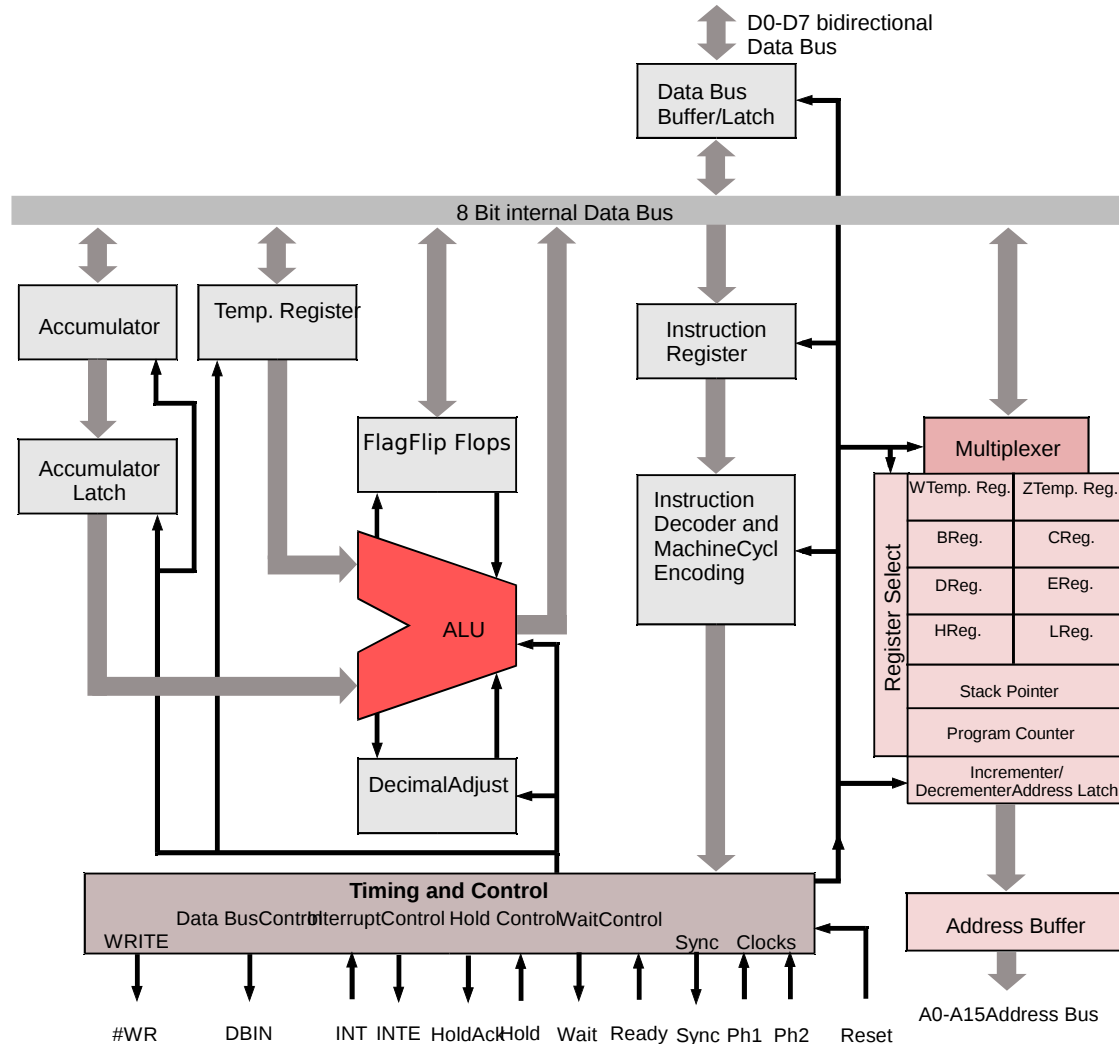
Technologické porovnání

CPU	Výrobce	Rok	Tranzistorů	Techno.	Reg/Bus	Data/prog+IO	Cache I/D+L2	Float	Frekv.	MIPS	Cena
4004	Intel	1971	2,300	10um - 3x4mm	4bit	1kB/4kB			750kHz	0.06	\$200
8008	Intel	1972	3,500	10um	8bit	16kB				0.06	
8080	Intel	1974	6,000	6um	8bit	64kB+256			2MHz	0.64	\$150
MC6501	NMOS T.	1975									\$20
8085	Intel	1976	6,500	3um	8bit	64kB+256			5MHz	0.37	
Z-80	Zilog	1976			8bit	64kB+256			2.5MHz		
MC6502	NMOS T.	1976									\$25
8086	Intel	1978	29,000	3um	16/16bit	1MB+64kB			4.77MHz	0.33	\$360
8088	Intel	1979		3um	16/8bit	1MB+64kB			4.77MHz	0.33	
MC68000	Motorola	1979	68,000		16-32/16bit	16MB					
80286	Intel	1982	134,000	1.5um	16/16bit	16MB/1GBvirt	256B/0B		6MHz	0.9	\$380
MC68020	Motorola	1984	190,000		32/32bit	16MB	Ano		16MHz		
80386DX	Intel	1985	275,000	1.5um	32/32bit	4GB/64TBvirt			16MHz		\$299
MC68030	Motorola	1987	273,000			4GB+MMU	256B/256B				
80486	Intel	1989	1.2mil	1um	32/32bit	4GB/64TBvirt	8kB	Ano	25MHz	20	\$900
MC68040	Motorola	1989	1.2mil			4GB+MMU	4kB/4kB	Ano			
PowerPC 601	Mot+IBM	1992	2.8mil		32/64bit	2 ⁵⁶	32kB	Ano	66MHz		
PA-RISC	HP	1992							50MHz		
Pentium	Intel	1993	3.1mil	0.8um - BiCMOS	32/64bit	4GB+MMU		Ano	66MHz	112	
Alpha	DEC	1994	9.3mil		64bit	4GB/64TBvir	8/8+96kB		300MHz	1000	
MC68060	Motorola	1994	2.5mil			4GB+MMU	8kB/8kB	Ano	50MHz	100	\$308
Pentium Pro	Intel	1995	5.5mil					Ano	200/60MHz	440	\$1682
Pentium II	Intel	1998	7.5mil		32/64bit			Ano+MMX	400/100MHz	832	
PowerPC G4MPC7400	Motorola	1999		0.15um - cooper6LM CMOS	64/128bit	4GB/2 ⁵²	32kB/32kB+2MB	Ano+AV	450MHz	825	

Akumulátorové architektury

- registr+akumulátor → akumulátor
 - 4bit Intel 4004 (1971)
 - 8bit Intel8080 (1974) - adresace s využitím dvojic registrů
 - pouze základní aritmetickologické operace – sčítání, odčítání, posuny pro 8-bit akumulátor
 - volání CALL a RET s ukládáním 16-bit PC na zásobník
 - několik 16-bit operací – inkrement/dekrement dvojic registrů, přičtení k HL a ukládání na zásobník
 - výkon instrukcí řízený mikrokódem – na jednu instrukci 2 až 11 cyklů při 2 MHz hodinách

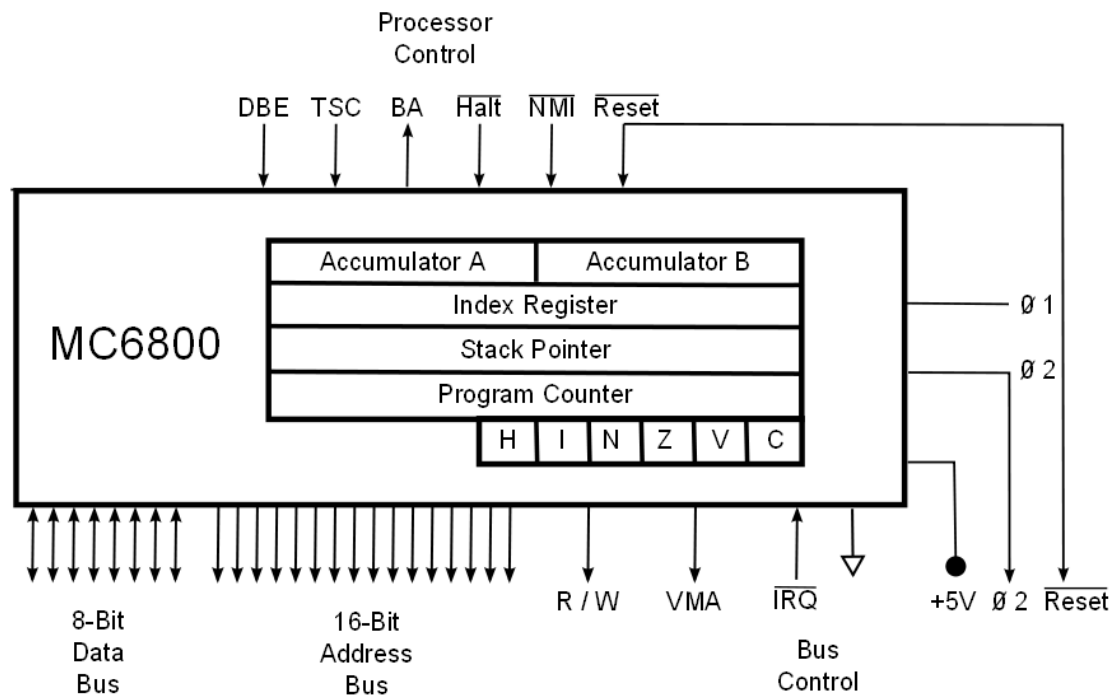
Intel 8080



http://en.wikipedia.org/wiki/Intel_8080

Rychlé paměti ⇒ málo registrů a více adresačních módů

- Motorola 6800, NMOS T. 6502 (1975) - jen akumulátor, index, SP a PC. - využití nulté stránky, HW dekodér
- Texas TMS990 pouze workspace pointer, PC, SP, registry v paměti, podobné u transputerů



Nedostatečná rychlost paměti \Rightarrow výkonný komplexní instrukční soubor, blízký C, CISC

- Motorola 68000 (1979) – 16/32bit
 - dvou-operandové instrukce
 - operace $\text{registr} += \text{registr}$, $\text{paměť} += \text{registr}$, $\text{registr} += \text{paměť}$, $\text{paměť} += \text{paměť}$
 - rozsáhlá instrukční sada, nutnost použít mikrokód
- Z-8000 16bit, Z-80000 32bit (1986) CISC
 - 6 fází zpracování, bez mikrokódu, 18000 tranzistorů

Nízká průchodnost dat a načítání instrukcí ⇒ cache paměti

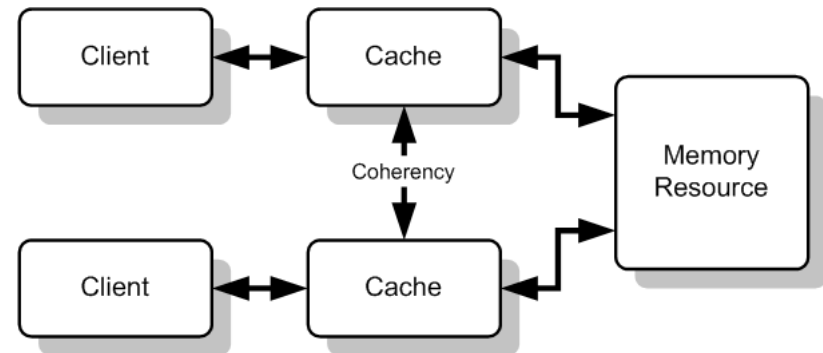
- Celkem dobře řešitelný problém
- Společná cache nebo harvardská I & D
- Další úrovně (rychlost limitovaná pro větší kapacity – dekodér, kapacita společných vodičů)
- Komplikuje se však koherence dat při přístupu přes DMA a pro SMP
 - speciální instrukce pro přístup k periferiím a synchronizaci např. `eieio` (PowerPC), `mcr p15` (ARM), ...
 - hardwarová podpora
 - protocol MSI , MESI (Pentium), MOSI
 - MOESI AMD64 (Modified, Owned, Exclusive, Shared, and Invalid)

Koherence dat a vyrovnávací paměti - cache

MOESI protokol

- Modified – řádka cache obsahuje aktuální a modifikovaná data, ostatní CPU s daty nepracují, v paměti jsou data stará
- Owned – řádka obsahuje aktuální data, ta mohou být sdílena i jinými CPU, ale pouze v S, v paměti může být stále předchozí hodnota
- Exclusive – pouze tento CPU a paměť obsahuje data
- Shared – řádka je sdílena i s jinými CPU, pokud některé z nich v O, tak mohou být data rozdílná od starého obsahu hlavní paměti
- Invalid – v řádce nejsou žádná platná data

	M	O	E	S	I
M	N	N	N	N	Y
O	N	N	N	Y	Y
E	N	N	N	N	Y
S	N	Y	N	Y	Y
I	Y	Y	Y	Y	Y

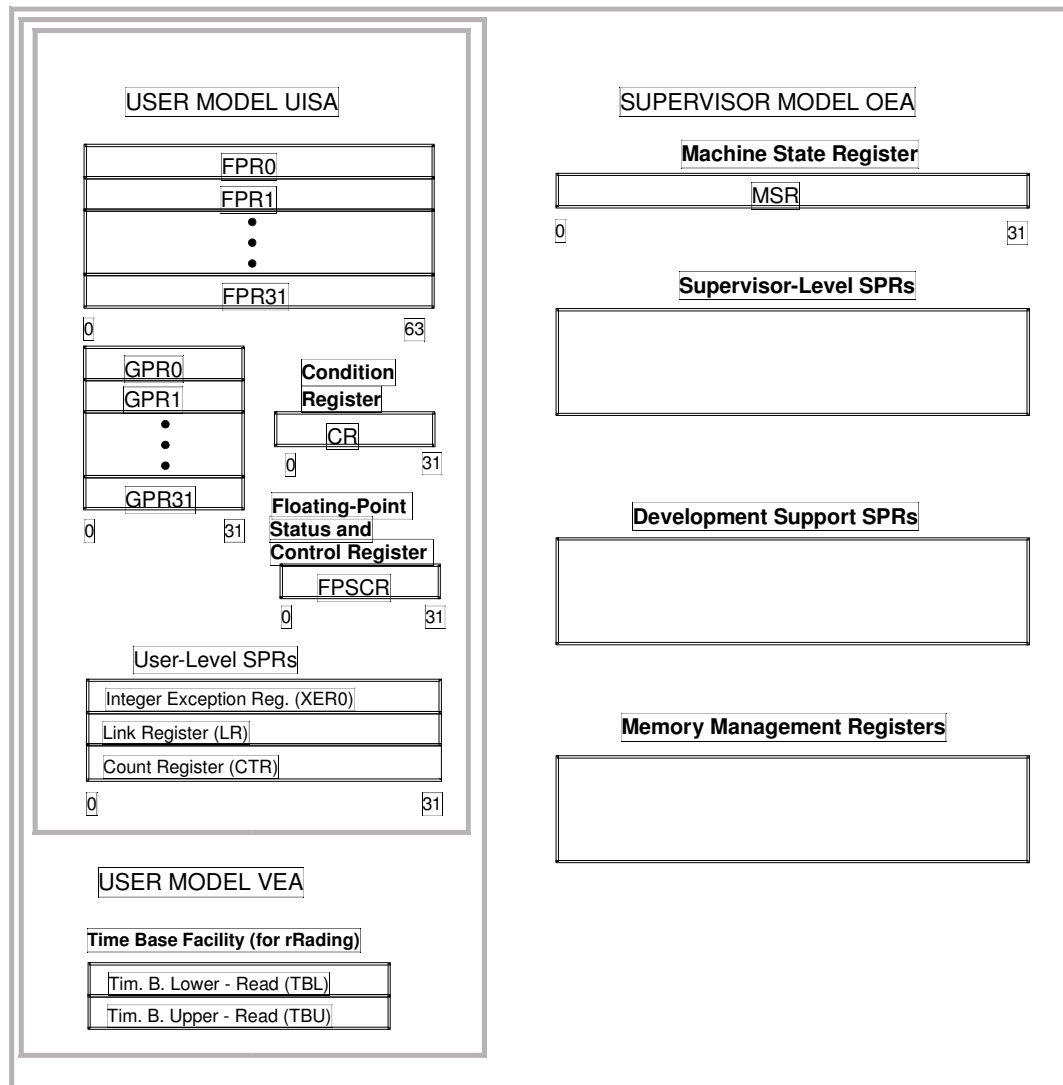


http://en.wikipedia.org/wiki/MOESI_protocol

Další eliminace přístupů do paměti ⇒ registrová okna, link/return registr

- SPARC - 8 globálních registrů, 8 z předchozího okna (parametry), 16 z aktuálního okna, více jak 100 registrů pro posuny okna. 8 registrů z aktuálního slouží k předání parametrů volané funkci
- PowerPC - zrychlení volání leaf-node funkcí s využitím specializovaného registru (link registr) pro návratovou hodnotu čítače instrukcí

Architektura PowerPC



SPARC – registrová okna

- CPU obsahuje 40 až 520 obecně použitelných 32-bitových registrů
- 8 z nich jsou globální registry ostatní jsou po 16 rozděleny do 2 až 32 registrových oken
- Každá instrukce může přistupovat k 8 globálním registrům a 24 registrům v právě přístupném okně
- 24 registrů v okně je rozděleno na 8 vstupních (in), 8 lokálních (local) a 8 registrů ze sousedního okna adresovatelných z aktuálního okna jako výstupní (out) registry
- Aktivní okno je dané hodnotou 5-bitového ukazatele Current Window Pointer (CWP).
- CWP je dekrementován při vstupu do podprogramu a další okno je nastaveno na aktivní
- Inkrementace CWP navrátí předchozí okno
- Bitový registr Window Invalid Mask (WIM) umožní označit libovolná okna za neplatná a vyžádat při jejich aktivaci výjimku (overflow or underflow)

SPARC - registry

R31	Return from actual window ...	%i7
R30	The frame pointer %fp ...	%i6
R29		%i5
R28		%i4
R27		%i3
R26		%i2
R25		%i1
R24		%i0
R23		%l7
R22		%l6
R21		%l5
R20		%l4
R19		%l3
R18		%l2
R17		%l1
R16		%l0
R15	CALL out return address ...	%o7
R14	The stack pointer %sp ...	%o6
R13		%o5
R12		%o4
R11		%o3
R10		%o2
R9		%o1
R8		%o0

I
(in)

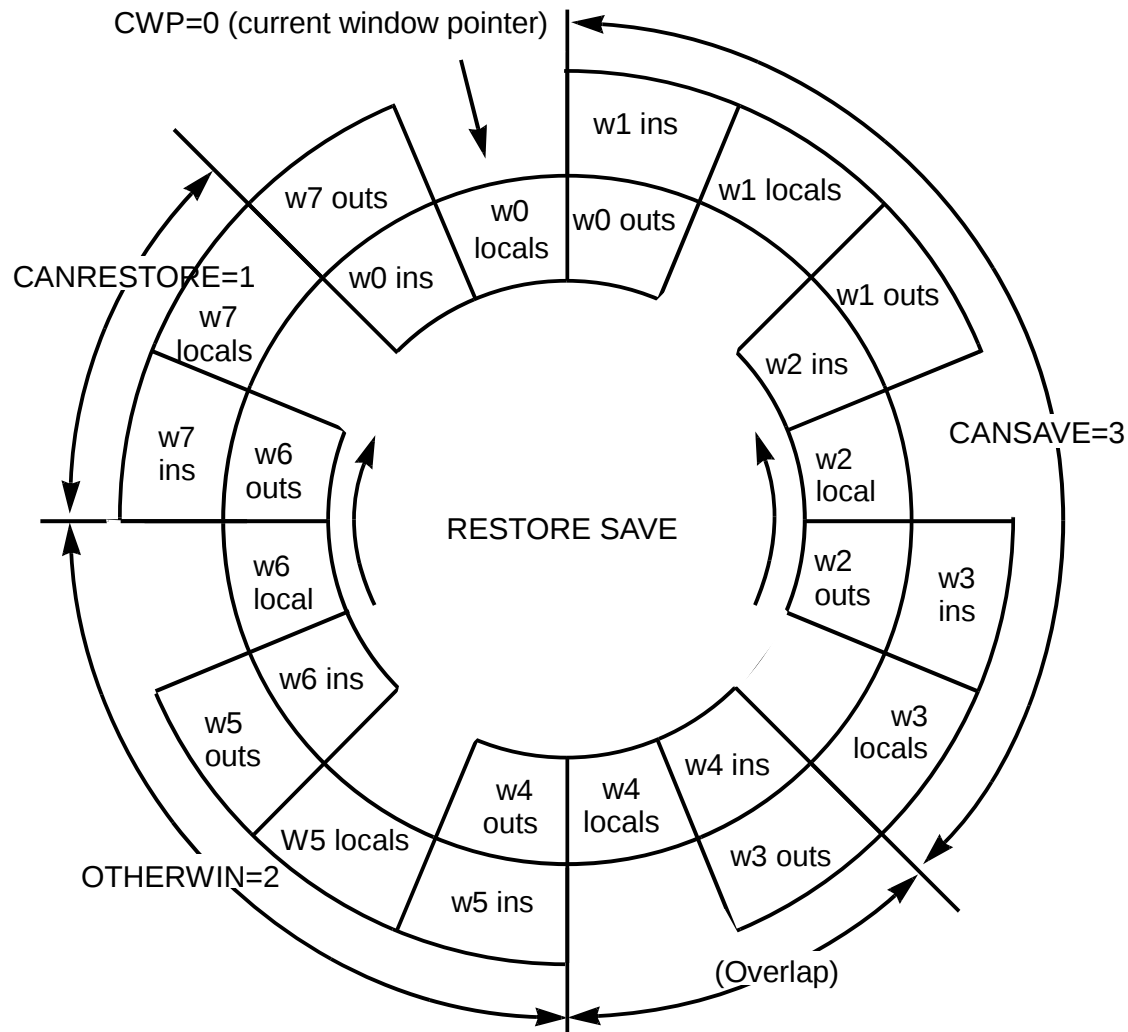
L
(local)

O
(out)

G (global)

R7	%g7
R6	%g6
R5	%g5
R4	%g4
R3	%g3
R2	%g2
R1	used by system %g1
R0	zero %g0

SPARC – práce s registrovými okny



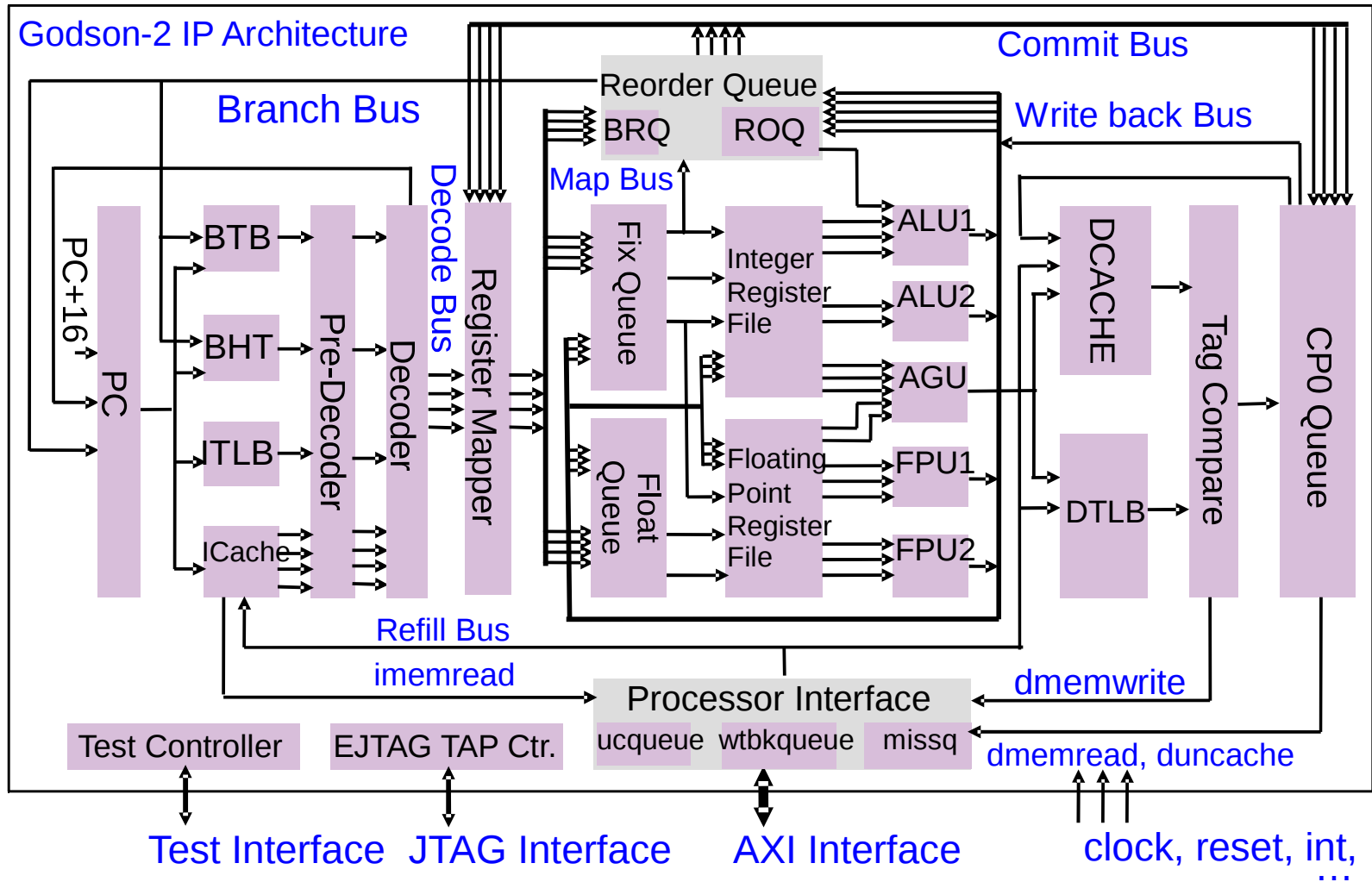
Vícefázové zpracování bez mikrokódu, stále potíže se skoky

- Včasné dekódování skoků
- Využití delay slotů MIPS, DSP
- Statická a dynamická predikce skoků, cache cílů skoků, spekulativní provádění instrukcí

Varianty na bázi MIPS

- Pravděpodobně jeden čas architektura s největším počtem užití na Zemi (různé AP, embedded systémy atd.)
- Stále ve vývoji i pro výkonné desktopy a superpočítače – Loongson3A
- Inspirováno mnoho SoftCore jader pro FPGA
 - Xilinx Microblaze
 - Altera Nios

Loongson3A



Zrychlení výkonu instrukcí \Rightarrow architektury RISC

- Menší vzájemná závislost, tříoperandové instrukce, spekulativní provádění instrukcí, přejmenovávání registrů, eliminace závislosti na příznakovém registru (DEC Alpha, několik př. registrů PowerPC, potlačení nastavování příznaků ARM)
- architektury load-store, operace pouze $\text{registr} += \text{registr}$ nebo $\text{registr} = \text{registr} + \text{registr}$ a separátní load-store instrukce.
- Pevná délka instrukčního kódu \Rightarrow dlouhé programy, ale velmi rychlé dekódování instrukcí, optimalizované pro zřetězené zpracování (pipeline)

Snaha opět zkrátit instrukce \Rightarrow aliasy, proměnná délka u RISC, VLIW

- ARM, 16bit aliasy nejčastějších 32bit instrukcí (režim Thumb)
- M-Core, 32bit CPU s 16bit instrukcemi
- ColdFire - RISC implementace na bázi 68000, 16, 32, 48bit instrukce

Architektura ARM - registry

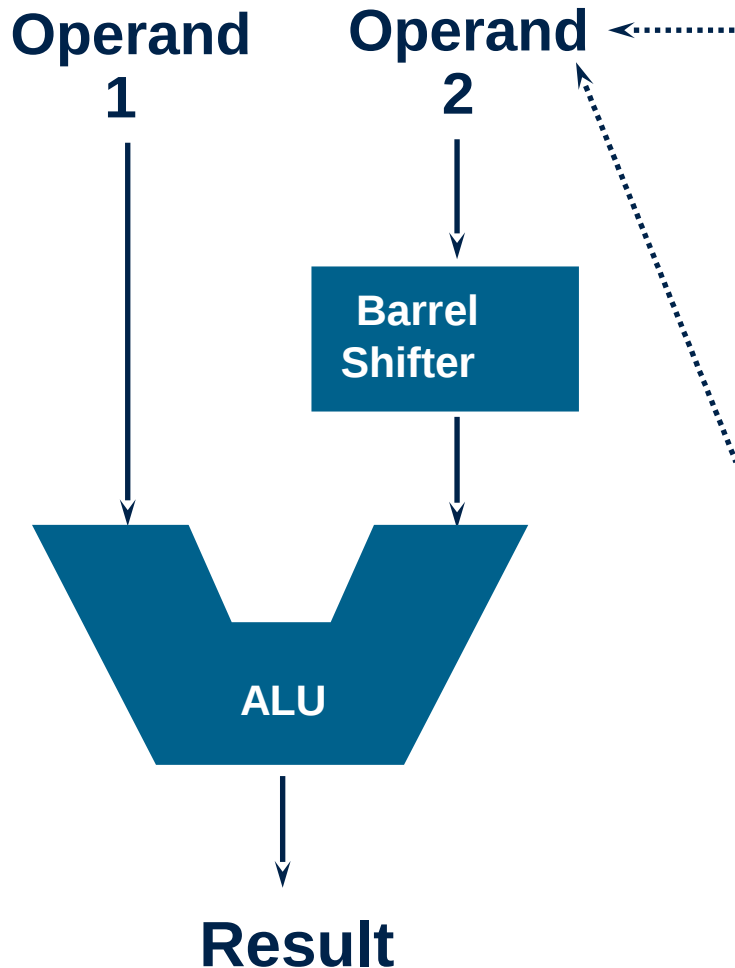
Current Visible Registers

Abort Mode	r0
	r1
	r2
	r3
	r4
	r5
	r6
	r7
	r8
	r9
	r10
	r11
	r12
	r13 (sp)
	r14 (lr)
	r15 (pc)
cpsr	
spsr	

Banked out Registers

User	FIQ	IRQ	SVC	Undef
	r8			
	r9			
	r10			
	r11			
	r12			
r13 (sp)	r13 (sp)	r13 (sp)	r13 (sp)	r13 (sp)
r14 (lr)	r14 (lr)	r14 (lr)	r14 (lr)	r14 (lr)
	spsr	spsr	spsr	spsr

Architektura ARM – ALU a operandy



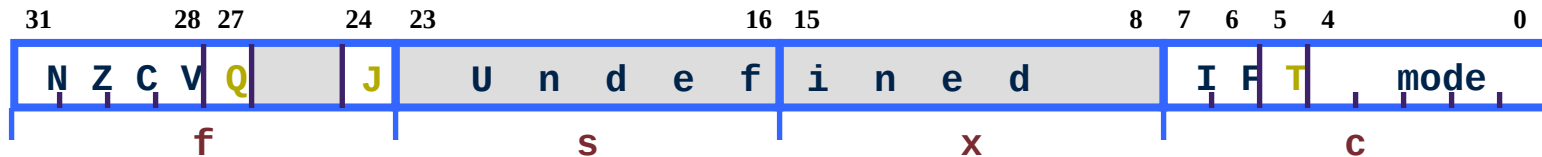
Register, optionally with shift operation

- Shift value can be either be:
 - 5 bit unsigned integer
 - Specified in bottom byte of another register.
- Used for multiplication by constant

Immediate value

- 8 bit number, with a range of 0-255.
 - Rotated right through even number of positions
- Allows increased range of 32-bit constants to be loaded directly into registers

Architektura ARM – stavové slovo



■ Condition code flags

- N = **N**egative result from ALU
- Z = **Z**ero result from ALU
- C = ALU operation **C**arried out
- V = ALU operation **o**Verflowed

■ Sticky Overflow flag - Q flag

- Architecture 5TE/J only
- Indicates if saturation has occurred

■ J bit

- Architecture 5TEJ only
- J = 1: Processor in Jazelle state

■ Interrupt Disable bits.

- I = 1: Disables the IRQ.
- F = 1: Disables the FIQ.

■ T Bit

- Architecture xT only
- T = 0: Processor in ARM state
- T = 1: Processor in Thumb state

■ Mode bits

- Specify the processor mode

Architektura ARM – operační režimy

- User : unprivileged mode under which most tasks run
- FIQ : entered when a high priority (fast) interrupt is raised
- IRQ : entered when a low priority (normal) interrupt is raised
- Supervisor : entered on reset and when a Software Interrupt instruction is executed
- Abort : used to handle memory access violations
- Undef : used to handle undefined instructions
- System : privileged mode using the same registers as user mode

Téměř závěr

- Neexistuje ideální řešení všech diskutovaných problémů vhodné pro všechna nasazení
- Nezbývá než všechny výše uvedené techniky kombinovat a optimalizovat podle oblasti použití procesoru (maximální výpočetní výkon/minimální spotřeba)
- heterogenní systémy pro výpočetně náročné aplikace – vektorové jednotky, GPU, FPGA akcelerátory

ARM 64-bit – AArch64

- K volání používá LR, ruší banky registrů, pro výjimky ELR
- PC vyčleněn zvlášť (není již běžný registr)
- 31 64-bitových registrů R0 až R30 (R30 = X30 \cong LR)
 - V kódu Wn (W0) pro 32-bit přístup, Xn (X0) pro 64-bit
 - Reg. code 31 stejně jako MIPS 0, v kódu WZR/XZR
 - V některých instrukcích má reg. code 31 význam WSP, SP
- Přímý operand 12-bit s volitelným LS 12 pro aritmetické operace, opakující se bitové masky pro logické
- 32-bit operace ignorují bity 32–63 na vstupu a nulují na výstupu

AArch64 – Větvení a podmíněné operace

- Zrušená možnost podmínění každé instrukce i Thumb IT
- Zůstává příznakový registr, přidané CBNZ, CBZ, TBNZ, TBZ
- Pouze několik podmíněných instrukcí
 - součet a rozdíl s přenosem, select (move C?A:B)
 - nastavení 0 a 1 (nebo -1) podle podmínky
 - podmíněná instrukce porovnání
- 32-bit a 64-bit násobení a dělení (3 registry), násobení se sčítáním $64 \times 64 + 64 \rightarrow 64$ (čtyři registry), bity 64 až 127 z výsledku násobení

AArch64 – Přístup do paměti

- Adresa 48+1 bit, znaménkové rozšíření na 64-bitů
- Přímě zakódovaný offset voliteně násoben šířkou přístupu
- Když je použitý registr jako index, může být násobený šířkou přístupu a také lze omezit na 32-bitů
- PC relative $\pm 4\text{GB}$ lze zakódovat do 2 instrukcí
- Pouze načtení dvou nezávislých registrů LDP a STP (zrušeno LDM, STM), přidané LDNP, STNP
- Podporuje nezarovnané přístupy
- LDX/STX(RBHP) pro exkluzivní přístup 1,2,4,8 a 16 bytů

AArch64 – Adresní režimy

- Simple register (exclusive)

[base{,#0}]

- Offset

[base{,#imm}]

– Immediate Offset

[base,Xm{,LSL #imm}]

– Register Offset

[base,Wm,(S|U)XTW {#imm}] – Extended Register Offset

- Pre-indexed

[base,#imm]!

- Post-indexed

[base],#imm

Bits	Sign	Scaling	WBctr	LD/ST type
0	-	-	-	LDX, STX, acquire, release
9	signed	scaled	option	reg. pair
10	signed	unscaled	option	single reg.
12	unsig.	scaled	no	single reg.

- PC-relative (literal) load

label

RISC-V – znovu zjednodušit/optimalizovat RISC

- Patterson, Berkeley RISC 1984 → počátek RISC éry, vyvinul se ve **SPARC** (Hennessy **MIPS**, Stanford University)
- Komercializací o rozvojem se architektury staly opět příliš složité, zároveň současné licence a patenty znemožňují původním tvůrcům vlastní reálnou implementaci ve formě křemíku pro výuku a výzkum
- MIPS je základem většiny základních kurzů a často je i implementace podobného procesoru součástí pokročilejších kurzů (A4M36PAP)
- Krste Asanovic a další studenti Dr. Pattersona pro začali vyvíjet vlastní novou architekturu (počátek 2010)
- BSD Licence, proto, aby byla přístupná i v budoucnu
- Podpora GCC, binutils., Linux, QEMU, atd.
- Proti SPAC jednodušší, více podobná MIPS ale optimalizovaná na úrovni zátěže hradel (fanout) a délky kritických cest v budoucích návrzích
- Existuje již několik otevřených implementací a projekt **lowRISC** má za cíl výzkum v oblasti bezpečnosti a implementaci SoC na křemíku

RISC-V – specifikace architektury

- Specifikace ISA dostupná na <http://riscv.org/>
 - The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.0
Andrew Waterman, Yunsup Lee, David Patterson, Krste Asanovic
 - Ne jen popis architektury, ale i důvodů proč bylo dané řešení vybrané a jaké problémy/cenu obnáší alternativní řešení
- klasický návrh 32 celočíselných registrů, jeden nula (zero), operandy `regsrc1`, `regsrc2`, `regdest`, zajímavost striktní dodržení i pro `SaveWord`, důsledkem nespojitost přímých/immediate operandů, PC mimo registry, PC-relativní adresování
- Varianty s 32, 64 a 128-bitovými registry a adresací
- Vysoká hustota kódu (plánovaná 16-bit varianta kódování instrukcí)
- V kódování systematicky vyhrazen prostor pro plovoucí řádovou čárku (single, double, quad) a multimediální SIMD instrukce atd.