

Architektura počítačů

Rudolf Marek

r.marek@sh.cvut.cz

ICQ: 111813813

Jabber: [ruik@jabber.sh.cvut.cz](jabber:ruik@jabber.sh.cvut.cz)

Cíle

- seznámit se s počítači po hardwarové stránce
- porozumět základním principům práce počítačů
- získat všeobecný přehled o současných počítačích
- seznámit se s periferiemi počítačů z hlediska architektury a programátora
- naučit se programovat v assembleru

Co je počítač?

- Oxfordský slovník
 - zařízení co provádí výpočty nebo řídí operace, které jdou popsat čísly nebo logickými výrazy

Opakování číselných soustav

- Číselné soustavy:
 - polyadické
 - soustava, kterou používáme v běžném životě
 - každé číslo lze zobrazit pomocí součtů součinů cifer čísla a mocnin základu
 - každá (obvyklá) soustava má číslice 0, 1, ... z-1
 - $a = a_n * z^n + a_{n-1} * z^{n-1} + \dots + a_1 * z^1 + a_0 * z^0$
 - 12345 = ?
 - $z = 10, n = 4$
 - $12345 = a_4 * 10^4 + a_3 * 10^3 + a_2 * 10^2 + a_1 * 10^1 + a_0 * 10^0$
 - $12345 = 1 * 10^4 + 2 * 10^3 + 3 * 10^2 + 4 * 10^1 + 5 * 1$
 - nepolyadické
 - soustava používaná např. v římě I, II, III, IV, V, VI, VII, VIII IX X XI ...
 - **MeDiCimbaL** (1000, 500, 100, 50)

Dvojková soustava

- někdy se jí říká binární
- výhodná pro počítače
 - snadná reprezentace signálu
 - počítání ve dvojkové soustavě zjednoduše aritmetické operace (přenosy)
- 000 0
- 001 1
- 010 2
- 011 3
- 100 4
- 101 5
- 110 6
- 111 7

Dvojková soustava II

- Do vzorce dosadíme základ $z=2$
 - $(110)_2 = 1*2^2 + 1*2^1 + 0*2 = (6)_{10}$
- Číslicím 0 a 1 se říká **bit** (binary digit)
- tří bitové číslo má nejvýše tři cifry (číslice)

Šestnáctková soustava

- Stejnou hodnotu čísla můžeme vyjádřit ve více soustavách
- Jaké číslice následují po 9?
 - A 10
 - B 11
 - C 12
 - D 13
 - E 14
 - F 15
- $(1DEAD)_{16} = (DEAD)_{16} = 1*16^4 + 13*16^3 + 14*16^2 + 10*16^1 + 13*16^0$
- Osmičková soustava se dnes již nevyužívá, princip je obdobný

Převody mezi soustavami

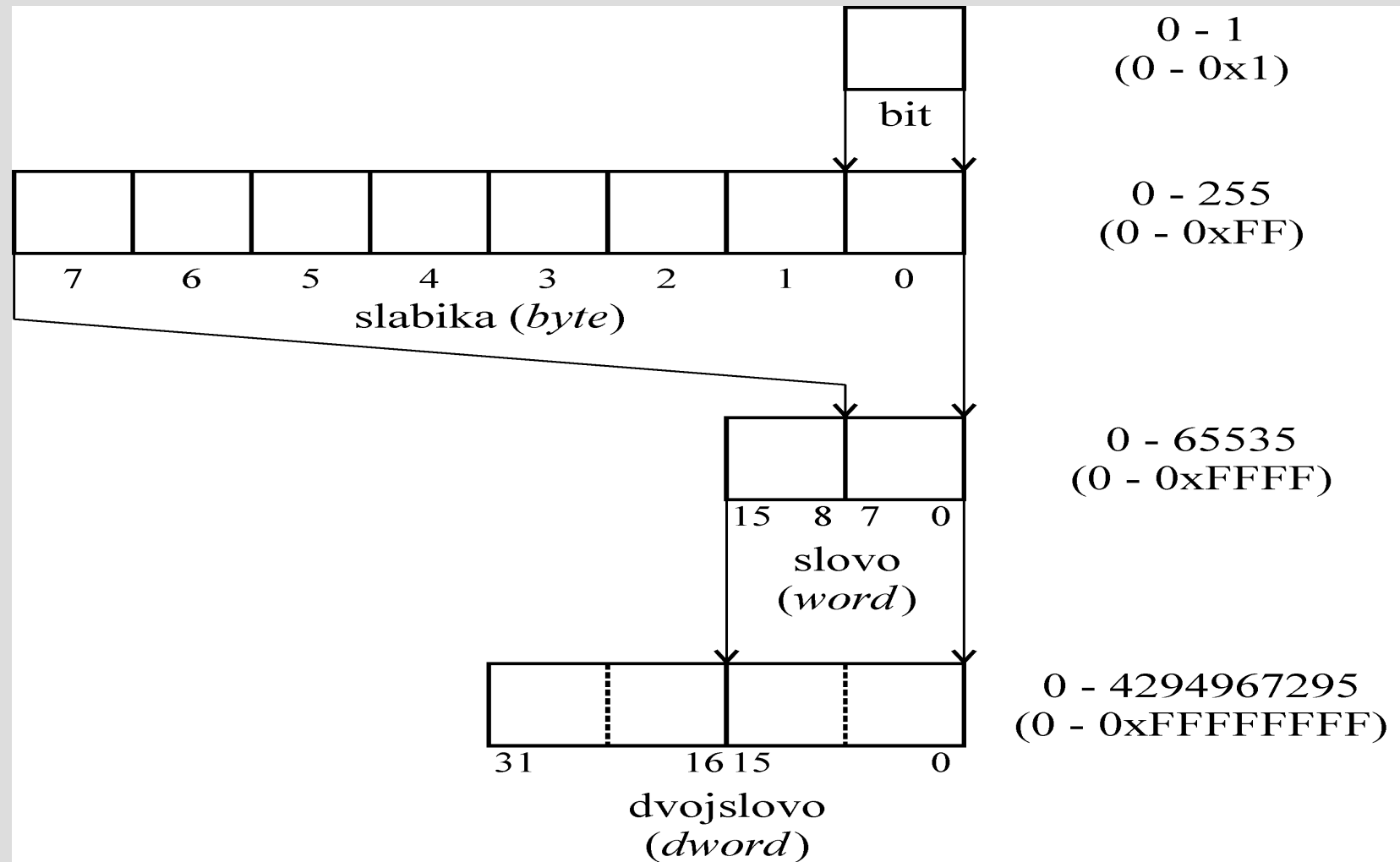
- Mezi soustavami, které jsou mocninami čísla 2
 - šestnáctková a dvojková
 - 00011001 -> 0001 1001 -> 1 9 -> 0x19
 - 0x22 -> 0010 0010 -> 00100010
 - 0xF -> 1111
- Pro převod mezi desítkovou a dvojkovou nebo desítkovou a šestnáctkovou (obecně jakoukoli jinou, např trojkovou :) musíme použít dělení se zbytkem
- Postup:
 - pro převod čísel z desítkové soustavy do soustavy o základu z, budeme převáděné číslo opakovaně dělit základem soustavy z a tyto zbytky budeme sepisovat dělíme tak dlouho až bude výsledek dělení nula. (NE zbytek)

Převody mezi soustavami

- Př: Převeďte číslo 10 do dvojkové soustavy
 - $10 / 2 = 5$ zbytek **0**
 - $5 / 2 = 2$ zbytek **1**
 - $2 / 2 = 1$ zbytek **0**
 - $1 / 2 = \mathbf{0}$ zbytek **1**
 - Zbytky sepíšeme “od konce” dostaneme $(1010)_2$
- Př: Převeďte číslo 23 do šestnáctkové soustavy
 - $23 / 16 = 1$ zbytek **7**
 - $1 / 16 = \mathbf{0}$ zbytek **1**
 - Výsledek je $0x17$ kontrola: $1 \cdot 16 + 7 = 23$
- Pozn: Počítače pracují se $z=2$ ale v šestnáctkové soustavě ($z=16$) zapíšeme číslo pomocí menšího počtu řádů

Základní jednotky

- bit, byte (bajt), slovo (word), dvojslovo (dword)
- Zdroj: Rudolf Marek, Učíme se programovat v jazyce Assembler pro PC, strana 19, první vydání



Pořadí v rámci Byte

- Čtyřbitové binární číslo: 1001
- nejpravější bit
 - nultý (pořadí číslujeme od nuly)
 - LSB – Least Significant Bit
 - nejméně významný bit
 - váha jedna
- nejlevější bit
 - MSB – Most Significant Bit
 - největší váha

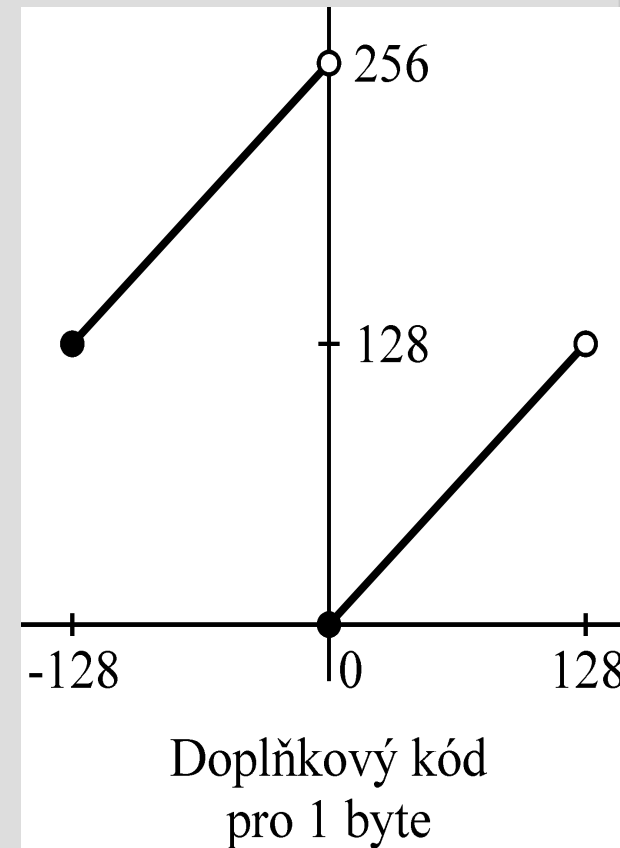
Pořadí v paměti

Indiáni - Endianness

- Proč píšeme zleva doprava? Proč ne opačně?
- Stejná otázka:
 - Jaké pořadí vícebajtových dat v paměti?
 - Je na nejnižší adrese uložen byte s nejnižším bitem (LSB) nebo naopak?
 - konvence zápisu se jmenuje „Endianness“
 - little endian
 - na nejnižší adresu zapíšeme bity 0 – 7
 - Intel x86 je malý indián
 - $0x12345678 \Rightarrow 0x78\ 0x56\ 0x34\ x12$
 - big endian
 - naopak
 - v sítích se používají
 - některé RISC procesory
 - $0x12345678 \Rightarrow 0x12\ 0x24\ 0x56\ x78$

Záporná čísla

- v běžných polyadických soustavách nejdou záporná čísla zobrazit
- Záporná čísla se v počítači zobrazují tak, že se jistý interval zobrazitelných čísel prohlásí za čísla záporná, pokud se zvolí dobře (kód zobrazení) není třeba modifikovat příliš HW ALU
- Doplnkový kód
 - nejběžnější, nejjednodušší na implementaci
 - dvojkový doplněk (bitová negace plus 1)
 - -1 je negace 1 $0xFE + 1 = 0xFF$
 - -2 je negace 2 $0xFD + 1 = 0xFE$
 - Fungují aritmetické operace?
 - Test: $-2 + 3 = 0xFE + 0x03 = 0x101$
 - Ano pokud zahodíme nejvyšší bit, který přetekl
 - Na PC se používá právě doplnkový kód
 - (unsigned int, signed int)



Základní logické operace

- logický součin
 - AND
 - $0 \text{ AND } 0 = 0$
 - $0 \text{ AND } 1 = 0$
 - $1 \text{ AND } 0 = 0$
 - $1 \text{ AND } 1 = 1$
- logický součet
 - OR
 - $0 \text{ OR } 0 = 0$
 - $0 \text{ OR } 1 = 1$
 - $1 \text{ OR } 0 = 1$
 - $1 \text{ OR } 1 = 1$
- negace
 - NOT $1 = 0$ a naopak

Základní logické operace II

- non-equivalence (výsledek je jedna pokud se arg nerovnají)
 - XOR
 - $0 \text{ XOR } 0 = 0$
 - $0 \text{ XOR } 1 = 1$
 - $1 \text{ XOR } 0 = 1$
 - $1 \text{ XOR } 1 = 0$

Základní logické operace III

- bitové posuvy
 - vlevo
 - vpravo
 - nejde náhodou násobit a dělit jistými hodnotami ?
 - jde např. mocninami čísla N, kde je N základ soustavy
- změny a zachování bitů
 - maska OPERACE hodnota
 - nastavení bitů
 - v masce nastavíme bity které budeme chtít mít 1 a provedeme OR
 - nulování bitů
 - v masce nastavíme všechny bity co chceme 0 na 0 zbytek na 1 a provedeme AND

Tabulka ASCII

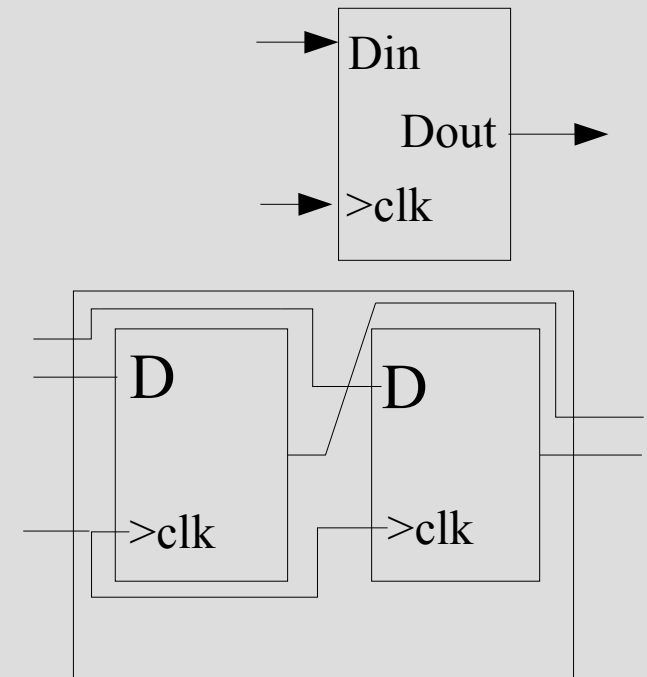
- Zobrazení čísel na znaky
- Množina 255 znaků (8b)
- Horních 128 pro národní abecedu
 - ISO 8859-2
 - CP1250
 - CP 852
 - ISO Latin2
 - mnoho dalších
- UTF
 - používá 16b (bit)
 - znaky všech národ. abeced

000	00		043	2B	+	086	56	U	129	81	ü	172	AC	¼	215	D7	
001	01	☉	044	2C	,	087	57	W	130	82	é	173	AD	½	216	D8	
002	02	☽	045	2D	-	088	58	X	131	83	â	174	AE	¾	217	D9	
003	03	♠	046	2E	.	089	59	Y	132	84	ä	175	AF	»	218	DA	
004	04	♣	047	2F	/	090	5A	Z	133	85	à	176	B0	»»	219	DB	
005	05	♠	048	30	0	091	5B	[134	86	á	177	B1	▨	220	DC	
006	06	♣	049	31	1	092	5C	\	135	87	ç	178	B2	▨	221	DD	
007	07	•	050	32	2	093	5D]	136	88	ê	179	B3		222	DE	
008	08	☐	051	33	3	094	5E	^	137	89	ë	180	B4		223	DF	
009	09		052	34	4	095	5F	~	138	8A	è	181	B5		224	E0	α
010	0A		053	35	5	096	60	`	139	8B	í	182	B6		225	E1	β
011	0B	δ	054	36	6	097	61	a	140	8C	î	183	B7		226	E2	Γ
012	0C	♀	055	37	7	098	62	b	141	8D	ì	184	B8		227	E3	Π
013	0D		056	38	8	099	63	c	142	8E	ñ	185	B9		228	E4	Σ
014	0E	♯	057	39	9	100	64	d	143	8F	ñ	186	BA		229	E5	σ
015	0F	*	058	3A	:	101	65	e	144	90	é	187	BB		230	E6	μ
016	10	▶	059	3B	;	102	66	f	145	91	æ	188	BC		231	E7	τ
017	11	◀	060	3C	<	103	67	g	146	92	œ	189	BD		232	E8	ϑ
018	12	‡	061	3D	=	104	68	h	147	93	ô	190	BE		233	E9	θ
019	13	!!	062	3E	>	105	69	i	148	94	ö	191	BF		234	EA	Ω
020	14	¶	063	3F	?	106	6A	j	149	95	ò	192	C0		235	EB	δ
021	15	§	064	40	@	107	6B	k	150	96	û	193	C1		236	EC	ω
022	16	_	065	41	A	108	6C	l	151	97	ù	194	C2		237	ED	ø
023	17	±	066	42	B	109	6D	m	152	98	ÿ	195	C3		238	EE	€
024	18	↑	067	43	C	110	6E	n	153	99	ö	196	C4		239	EF	∞
025	19	↓	068	44	D	111	6F	o	154	9A	ü	197	C5		240	F0	≡
026	1A		069	45	E	112	70	p	155	9B	ç	198	C6		241	F1	±
027	1B	←	070	46	F	113	71	q	156	9C	£	199	C7		242	F2	≥
028	1C	↳	071	47	G	114	72	r	157	9D	¥	200	C8		243	F3	≤
029	1D	↔	072	48	H	115	73	s	158	9E	℞	201	C9		244	F4	↑
030	1E	▲	073	49	I	116	74	t	159	9F	ƒ	202	CA		245	F5	↓
031	1F	▼	074	4A	J	117	75	u	160	A0	á	203	CB		246	F6	÷
032	20		075	4B	K	118	76	v	161	A1	í	204	CC		247	F7	≈
033	21	!	076	4C	L	119	77	w	162	A2	ó	205	CD		248	F8	°
034	22	"	077	4D	M	120	78	x	163	A3	ú	206	CE		249	F9	-
035	23	#	078	4E	N	121	79	y	164	A4	ñ	207	CF		250	FA	·
036	24	\$	079	4F	O	122	7A	z	165	A5	Ñ	208	D0		251	FB	√
037	25	%	080	50	P	123	7B	{	166	A6	æ	209	D1		252	FC	∞
038	26	&	081	51	Q	124	7C		167	A7	œ	210	D2		253	FD	∞
039	27	'	082	52	R	125	7D	}	168	A8	ç	211	D3		254	FE	∞
040	28	<	083	53	S	126	7E	~	169	A9	ı	212	D4		255	FF	∞
041	29	>	084	54	T	127	7F	Δ	170	AA	ı	213	D5				
042	2A	*	085	55	U	128	80	Ç	171	AB	½	214	D6				

- Zdroj: Rudolf Marek, Učíme se programovat

Součástky v konstrukci počítače

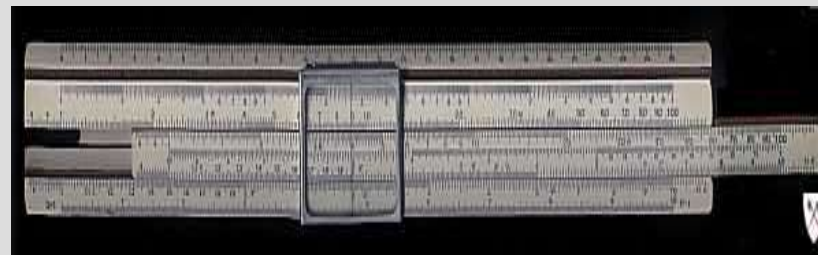
- Registr
 - skládá se z jednobitových registrů
 - jednobitový registr
 - má vstup, výstup, hodinový vstup
- Multiplexer
- Logické hradlo
- Paměť



Pre-historie počítačů

- Blíže v http://en.wikipedia.org/wiki/History_of_computing_hardware
- Úplně první počítač: váhy držené v ruce “slepé spravedlnosti”
- Počítače typu “kalkulačka”
 - 1623 Wilhelm Schickard, mechanická kalkulačka dokončená J. Kepllerem
 - Blaise Pascal (the Pascaline, 1640) i Gottfried Wilhelm von Leibniz (1670), který mimo jiné popsal prvně dvojkovou číselnou soustavu, které se konstruktéři vyhýbali celkem dlouho
- John Napier si všiml že násobení a dělení lze provádět postupným sčítáním nebo odčítáním logaritmů čísel.

Pre-historie počítačů II

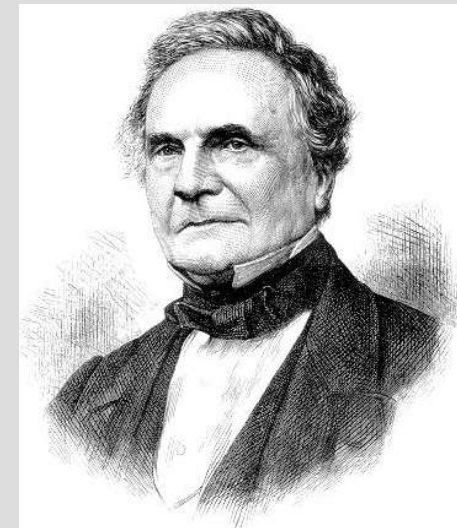


Příchod děrných štítků

- V roce 1801, Joseph-Marie Jacquard vynalezl tkalcovský stav, který uměl vyrábět různé vzory pouze výměnou děrných štítků
- V roce 1890 byly děrné štítky použity při sčítání lidu v přístrojích navrženém Hermanem Hollerithem, jehož společnost se stala jádrem společnosti IBM
- Děrné štítky se používali ve výpočetní technice až do konce 70 let 20tého století
- http://en.wikipedia.org/wiki/Punched_cards

První programovatelné počítače

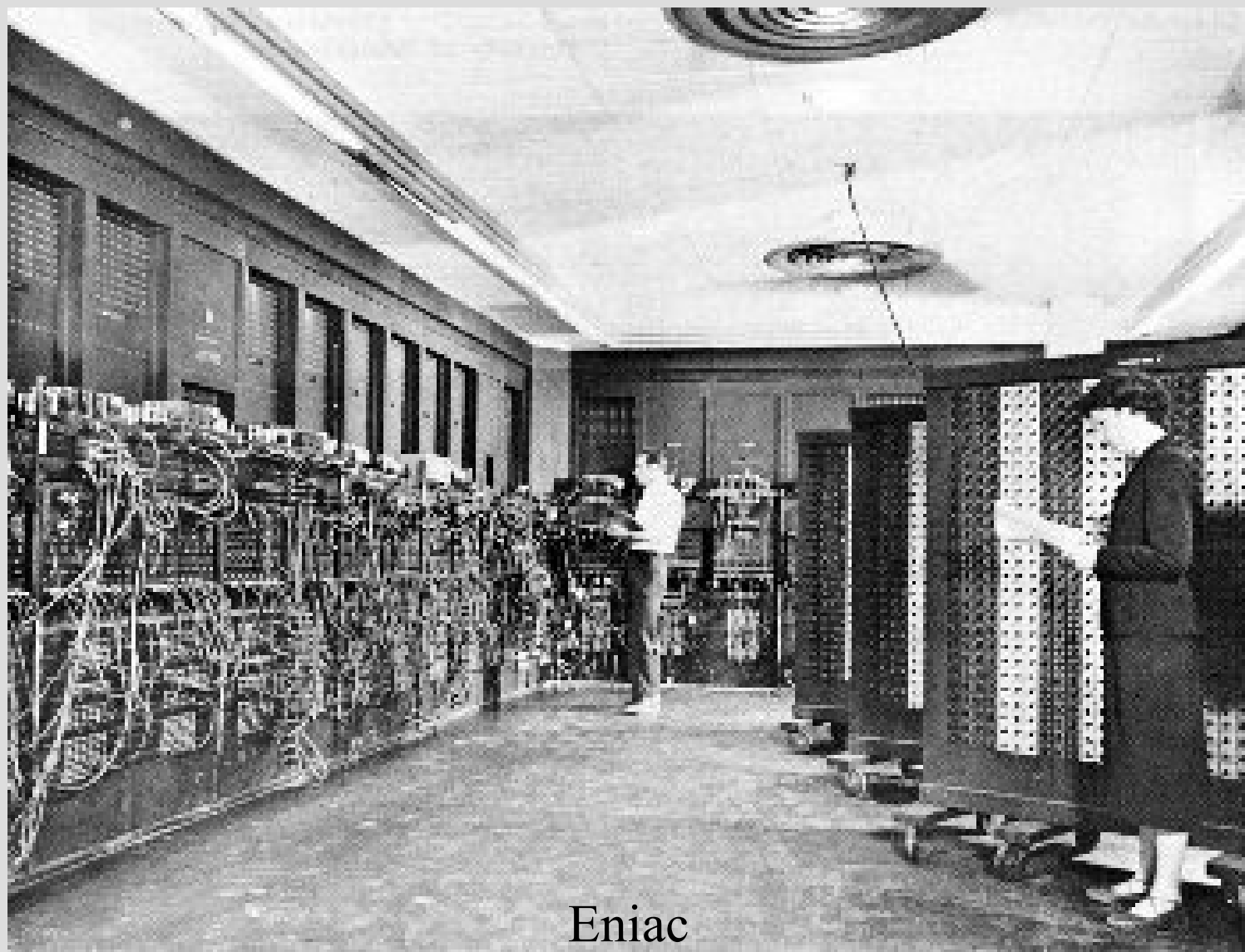
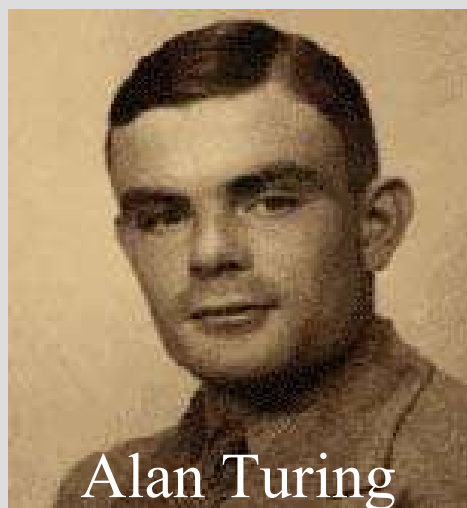
- Jsou univerzální
 - výměnou instrukcí mohou provádět jiné výpočty, konstrukce počítače se nemění
- 1835 Charles Babbage navrhl “analytical engine” nebyl dokončen
- pokračují různé další pákové počítače
- Analogové počítače
 - mohly řešit složité problémy (diferenciální rovnice)
 - špatně se ale rekonfigurovaly
 - http://en.wikipedia.org/wiki/Analog_computer



První generace počítačů

- vyvíjela se během druhé světové války
- používali se elektronky, relé které tak nahrazovali mechanické součástky. http://en.wikipedia.org/wiki/Vacuum_tube
- Paměť byla “papírová” (pásy, štítky) i akustické zpoždovací linky
- V roce 1937 se Claude Shannonovi podařilo implementovat Boolovu algebru pomocí relé a přepínačů
- IBM staví Mark I (Automatic Sequence Controlled Calculator)
- Spojené státy vytvoří ENIAC (Electronic Numerical Integrator and Computer spotřeba 160kW)
- Před i během války působí osobnost teoretické informatiky Alan Turing, který předkládá teoretické koncepty práce počítačů

První generace počítačů



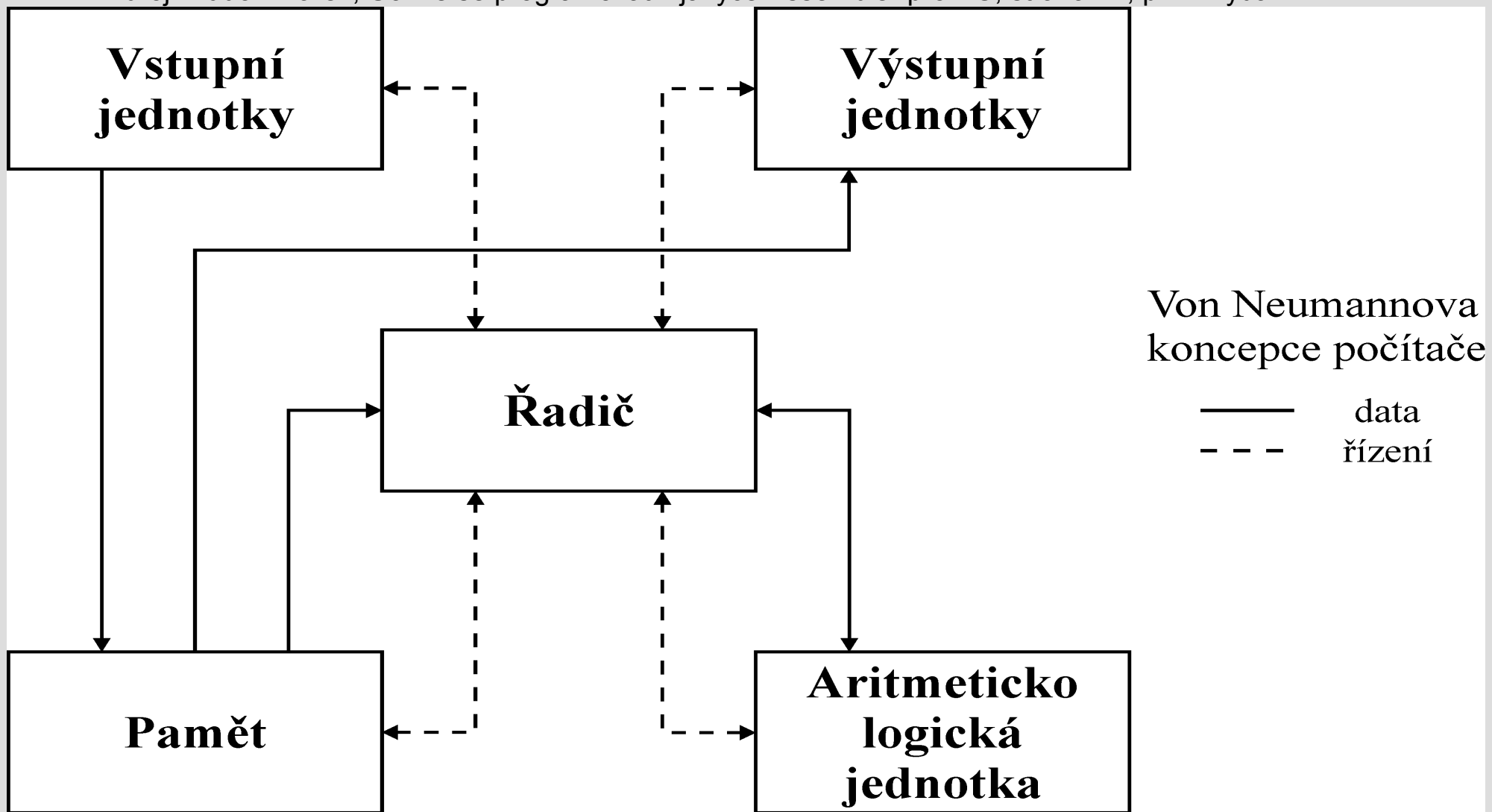
Eniac

Von Neumannova koncepce počítače

- Počítač tvoří tyto jednotky: **řadič, aritmetická jednotka, paměť, vstupní a výstupní jednotky**
- Struktura počítače je nezávislá na typu řešené úlohy, počítač se programuje obsahem paměti
- Instrukce a operandy jsou uloženy v téže paměti (Harvardská koncepce počítače, oddělila paměť pro program od paměti pro data)
- Paměť je rozdělena do buněk stejné velikosti, jejich pořadová čísla se používají jako adresy. (1 buňka je jeden byte)
- Program je tvořen posloupností elementárních příkazů (instrukcí), v nichž zpravidla není obsažena hodnota operandu (uvádí se pouze jeho adresa), takže program se při změně dat nemění. Instrukce se provádějí jednotlivě v pořadí, v němž jsou zapsány do paměti (dnešní mikroprocesory mimo jiné umožňují současné zpracování více instrukcí).
- Změna pořadí provádění instrukcí se vyvolá instrukcí podmíněného nebo nepodmíněného skoku.
- Pro reprezentaci instrukcí i čísel (operandů, výsledků, adres) se používají dvojkové signály a dvojková číselná soustava.
- **Zdroj:** Prof. Ing. Jan Hlavička, DrSc. strana 8-9 dotisk druhého vydání skript Architektura počítačů.

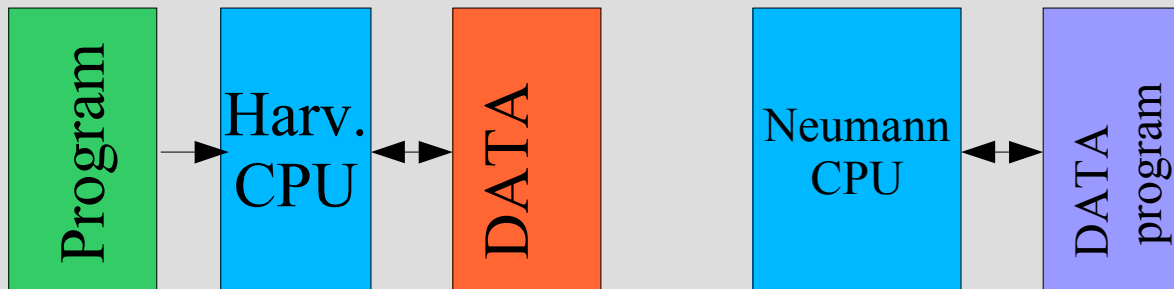
Von Neumannova koncepce počítače

- Zdroj: Rudolf Marek, Učíme se programovat v jazyce Assembler pro PC, strana 22, první vydání

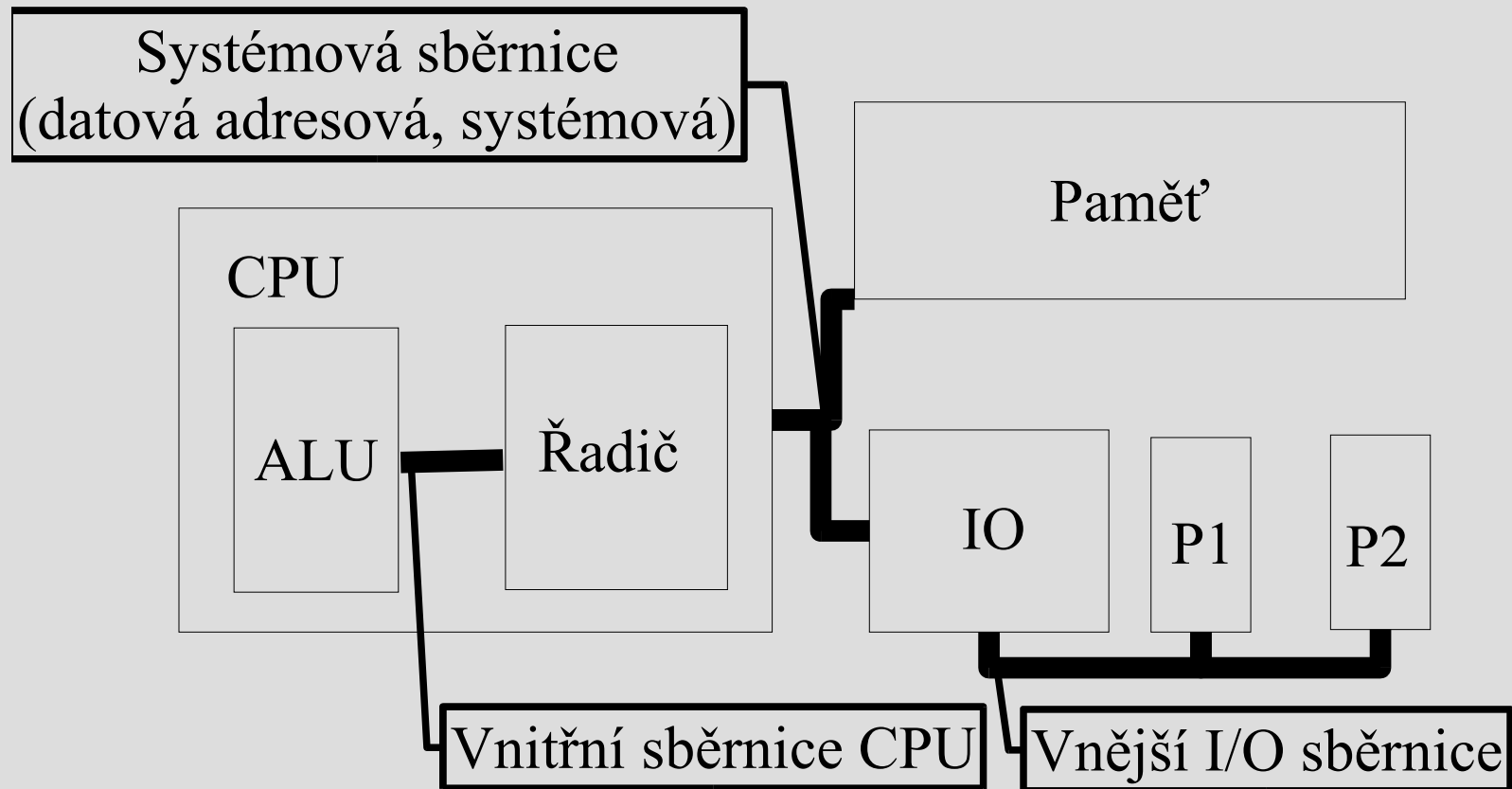


Rozdíly harvardské koncepce

- Harvardská koncepce
 - oddělená paměť pro data a pro program
 - jednodušší “pipeline”
 - žádné problémy se zarovnáním paměti
- Von Neumann
 - společná paměť pro data i program
 - více flexibilní
 - je možné vytvořit samomodifikující se kód



Generický počítač



Sběrnice

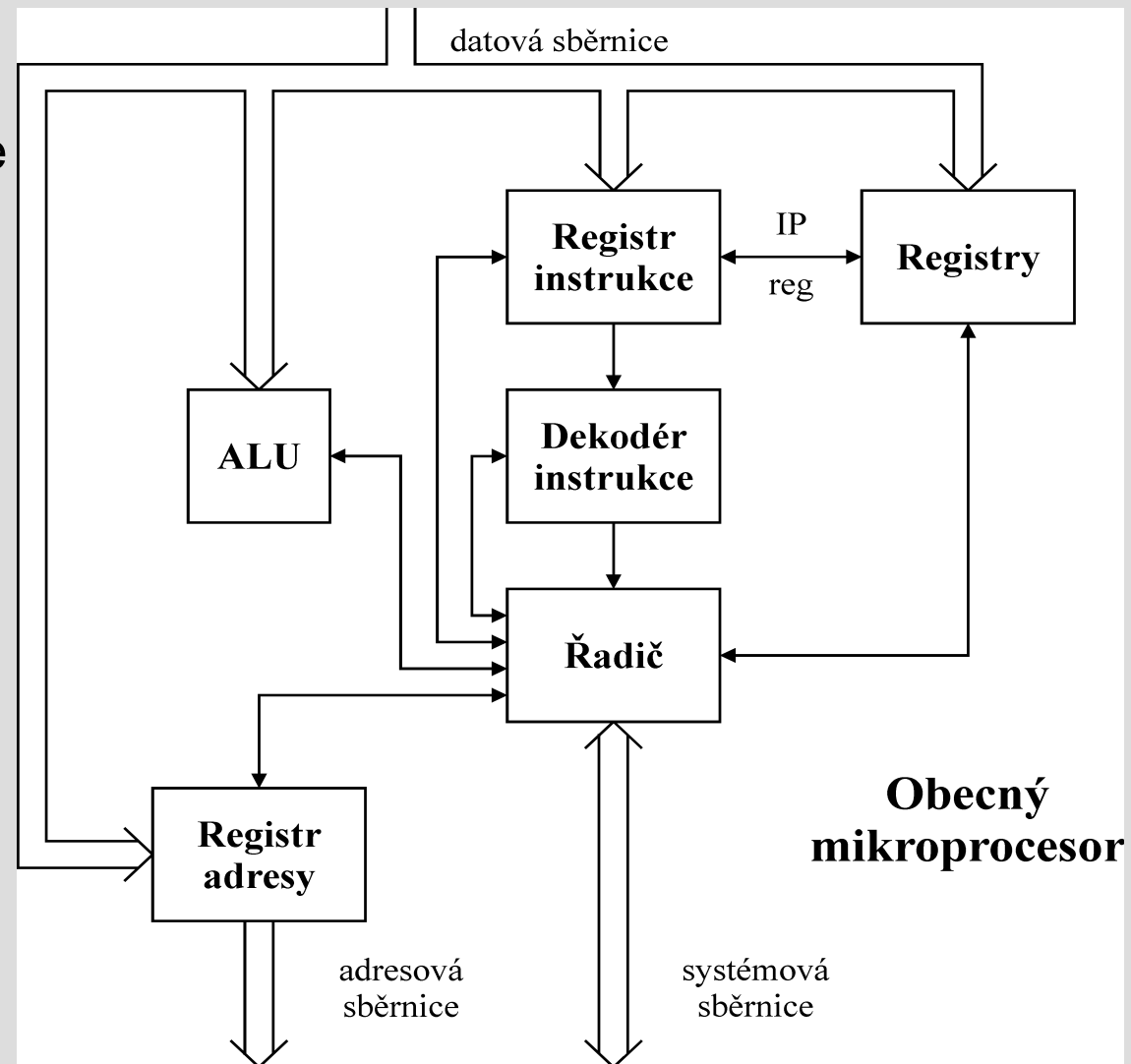
- Topologie
- Sériové
 - budeme o nich hovořit v souvislosti s periferními zařízeními
- Paralelní
 - počet bitů v názvu určuje jejich „šířku“
 - 16, 32, 64
 - procesor/paměť, interní sběrnice procesoru
 - vyskytuje se také v periferních zařízeních

Strojový cyklus a vnitřek mikroprocesoru

- Zdroj: Rudolf Marek, Učíme se programovat v jazyce Assembler pro PC, strana 25, první vydání

- **Strojový cyklus:**

- Načti instrukci z paměti
- Ulož ji v registru instrukce
- Dékoduj instrukci
- Proved' instrukci
- (ošetři přerušení)
- Načti další instrukci
- (adresa je uložena v programovém čítači registr IP)



Instrukční sada

- množina instrukcí (příkazů), které je schopen jistý konkrétní procesor vykonat
- http://en.wikipedia.org/wiki/Instruction_set_architecture
- programátorský pohled na procesor (AMD a Intel, jiný vnitřní návrh a přesto podpora stejných instrukcí)
- popis datových typů, se kterými umí procesor pracovat, registry, adresní režimi
- popis jednotlivých instrukcí
- “binární popis” - anatomie instrukce

Instrukce podrobněji

- Instrukce je příkaz k provedení operace zapsaný jako číslo
- Instrukce se skládá z:
 - operačního znaku – určuje o jakou instrukci se jedná (skok, přiřazení atd)
 - operandů
 - zdroj (nebo i zdroj 2)
 - cíl
 - operandů typu
 - registr
 - přímý
 - nepřímý (operand v paměti) paměť
 - s indexregistrem
 - jen adresou
 - nějak složitěji
- Počet bitů tvořící instrukci se nazývá délka instrukce
- Instrukční soubor – přiřazení operačních znaků různým operacím

OZ 8b	Source 4b	Dest 4b
-------	-----------	---------

CISC

- Complex Instruction Set Computers – počítače s komplexní instrukční sadou
- mnoho instrukcí, které provádějí složité operace
- byly přidány aby zaplnili propast mezi vyššími programovacími jazyky a assemblerem
- složité instrukce byly prováděny mikroprogramově (mikroprogramovatelný řadič)
- zjistlo se ovšem, že tyto instrukce jsou pomalé a dokonce pokud programátor použil jednodušší instrukce bylo zpracování rychlejší
- Procesory s CISC:
 - VAX, PDP-11, rodina Motorola 68000 a Intel x86
 - Dnešní procesory x86 převádějí uvnitř instrukce na jednodušší na tzv. mikroinstrukce a počítač pracuje jako RISC.

RISC

- Reduced Instruction Set Computers
- Motto: méně, a jednodušší instrukce se provedou rychleji
- SPARC, MIPS, and PowerPC.
- <http://en.wikipedia.org/wiki/RISC>
- ukázalo se že:
 - (1970) kompilátory nepoužívají všechny složité instrukce a adresační režimy
 - implementace vnitřních registrů procesoru už nestojí tolik a jsou mnohem rychlejší než už tedy pomalý přístup do paměti
 - stačí poskytnout programátorovi více jednodušších instrukcí než jednu složitou (zjednodušuje se tak návrh procesoru)
 - lze procesor významně zrychlit použitím HW prostředků (superskalarita, pipeling, cache viz další přednáška)
 - pro komunikaci s pamětí je dostačující jen dvojice instrukcí (load store)

Řadič podrobněji

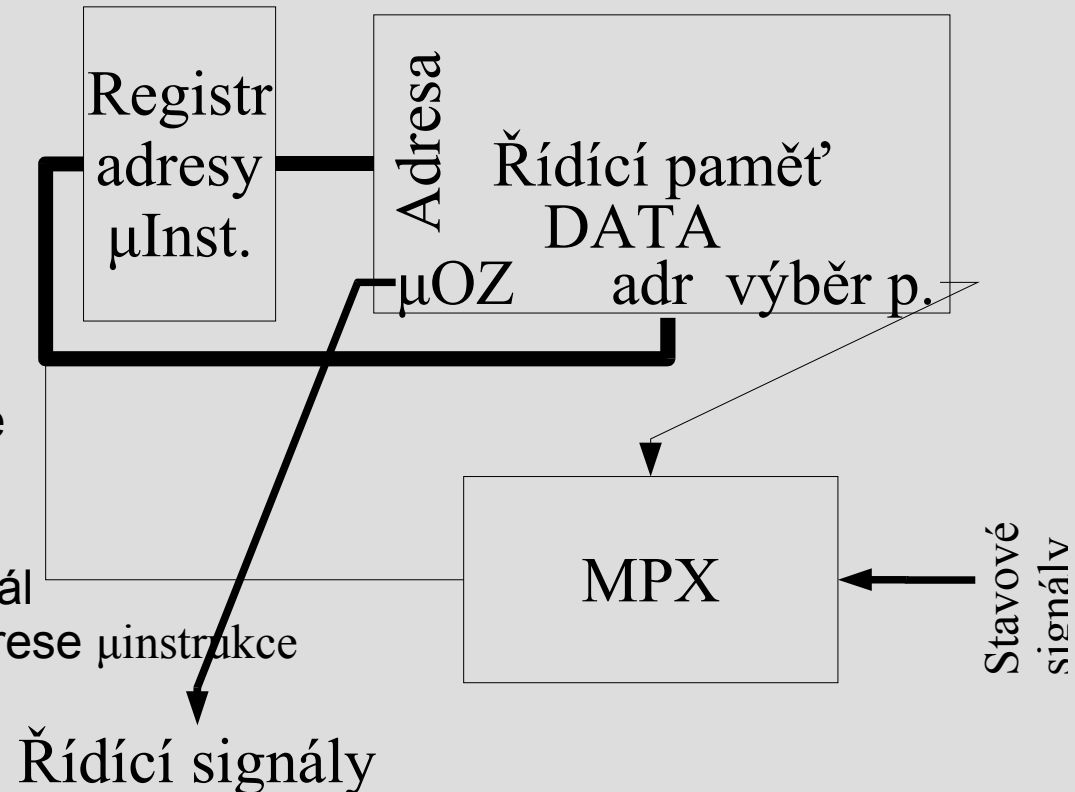
- je sekvenční obvod, který vytváří řídicí signály pro řízené obvody
- řadič vykonává instrukční cyklus. tj načítá instrukce z paměti a provádí je.
- řadič může být
 - obvodový
 - používá se u jednoduchých procesorů, je rychlý a levný
 - mikroprogramovatelný
 - více flexibilní, ale pomalejší a dražší
- bývá integrován na jednom chipu společně s ALU (aritmetickou jednotkou) a registry a tak tvoří mikroprocesor

Obvodový řadič

- Navržený konečný automat je možné přímo transformovat do obvodu který bude navrhovanou činnost vykonávat.
- Obvod se skládá se vzájemně propojených 1bitových registrů a logických členů, ze kterých vystupují řídicí signály nebo naopak do některých hradel venkovní stavové signály vstupují

Mikroprogramovatelný řadič

- Je tvořen pamětí ROM (EEPROM), ve které jsou uloženy mikroinstrukce
- mikroinstrukce se skládá z:
 - mikrooperační znak
 - jeho bity přímo tvoří hodnoty řídicích signálů
 - adresa další instr.
 - se nahraje do adr. registru a další mikrooperace se načte z této adresy
 - výběr podmínky
 - rozhoduje jaký stavový signál
 - bude rozhodovat o další adrese μ instrukce



Počítače s netradičním řízením

- nemusí používat instrukce
- neřadí instrukce do pevného pořadí
- řízení:
 - tokem dat (dataflow počítače)
 - např. výpočty výrazů
 - sestáváme graf závisostí
 - výpočet není synchronizován, každá operace probíhá nezávisle na ostatních, jsou-li k dispozici vstupní operandy operace
 - nutnost implementace skoků, cyklů
 - tokem požadavků
 - nevýhoda dataflow - výsledky jsou vypočteny když jsou k dispozici operandy a ne když je výsledek operace potřeba
 - výpočet probíhá postupnou redukcí požadavků 2×3 je 6
 - systolické systémy
 - pevně propojená síť funkčních jednotek, určená pro řešení jedné úlohy
 - data se pravidelně posouvají z jednotky do jednotky v hodinových pulsech
 - neuronové sítě a počítače

Trendy do budoucna

- počítače na bázi RISC (vnitřně)
- snad se zrychlí RAM
- počítače mají již nyní problém s fyzikálními omezeními
- kvantové počítače
- ještě větší paralelismus i na úrovni instrukcí, procesorů ...