Strojový kód a data

5. Přednáška

Základní cyklus jednoduchého počítače s von Neumannovou architekturou 1

R. Lórencz

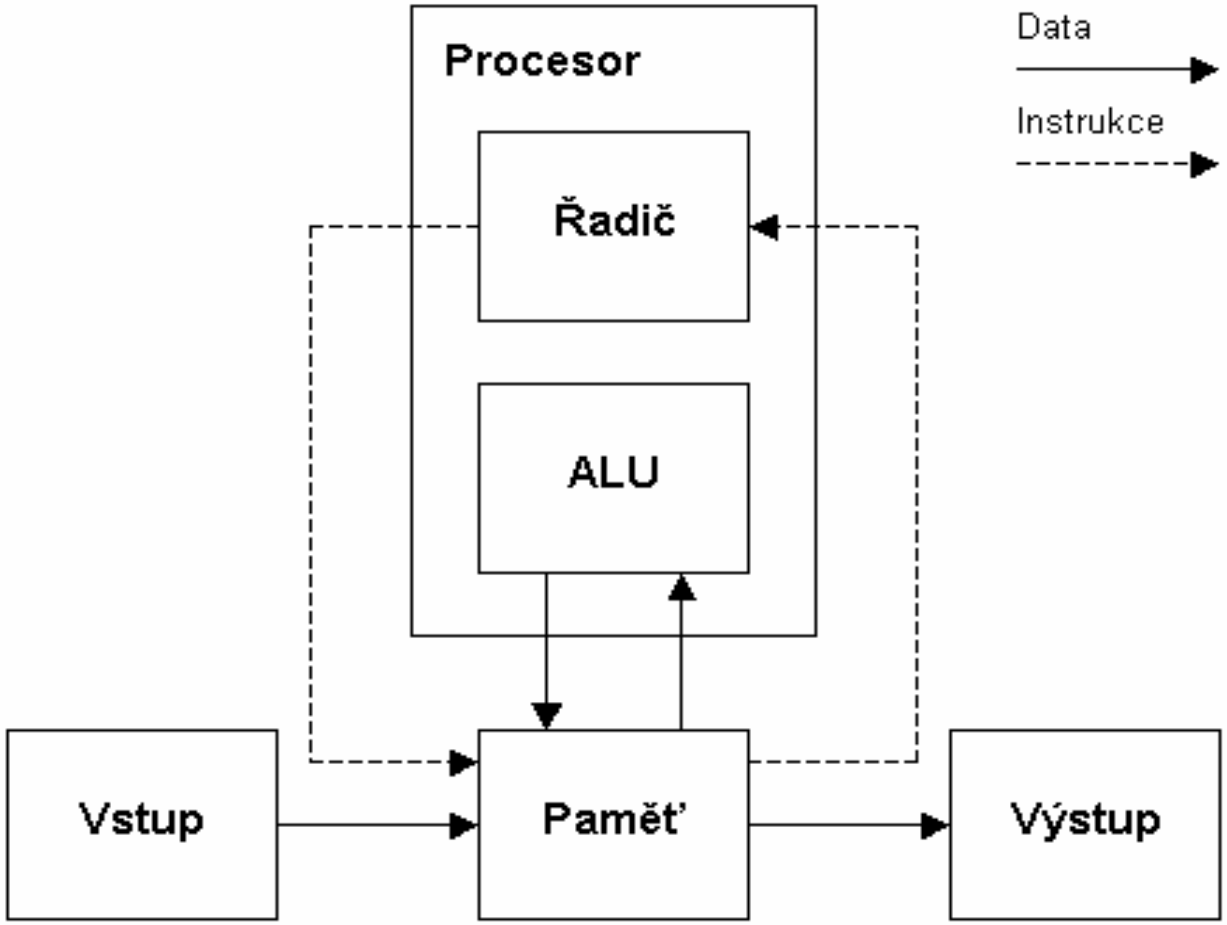
© 2004 R. Lórencz

Obsah přednášky

- popis architektury (paměť; CPU: ALU, CU, registry; sběrnice)
- paměť a její spolupráce s ostatními jednotkami počítače
- popis činnosti jednotek počítače při IF, ID, OF, IE a WB
- paměť a strojový kód (příklad na dané architektuře)
- činnost a význam PC, IR při základním cyklu počítače
- skoky (absolutní, podmíněné)

© 2004 R. Lórencz

Počítač von Neumannova typu, opakování



© 2004 R. Lórencz

Jednoduchý procesor von Neumannova typu (1)

CPU (Central Processing Unit) centrální procesorová jednotka

- ALU (Arithmetic/Logic Unit) aritmeticko-logická jednotka
- CU (Control Unit) řídící jednotka – řadič
- Registry

Paměť

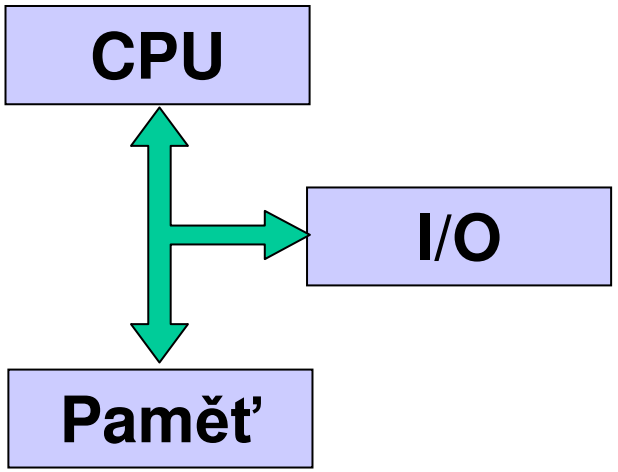
- Data
- Instrukce – program

I/O (Input/Output) Vstupně/Výstupní zařízení

- Diskové jednotky
- Grafické jednotky
- Porty, atd.

Mikroprocesorová sběrnice

- Komunikace mezi CPU, I/O a Pamětí

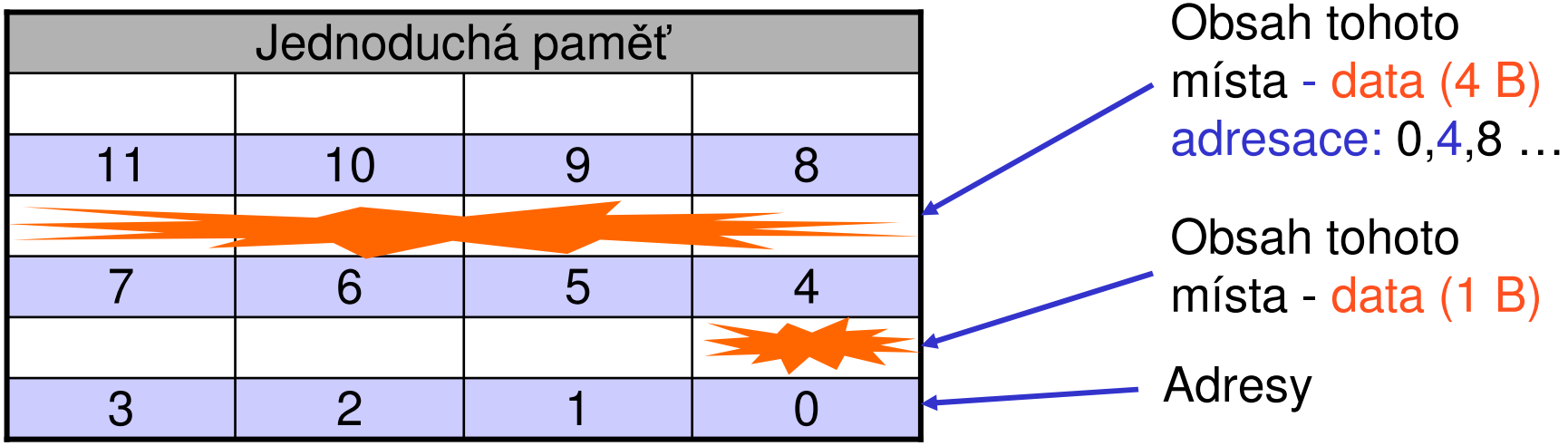


© 2004 R. Lórencz

Jednoduchý procesor von Neumannova typu (2)

Paměť

- Rozdělená na buňky - paměťová místa
- Buňky jsou adresovatelné celými nezápornými čísly
- Velikost obsahu paměťového místa (závisí na procesoru):
  - Slovo – word, např. 16b, 32b, 64b,…
  - Slabika – byte, zpravidla 1B = 8b
  - Označení obsahu paměťového místa na adrese *a:* [a] nebo ⟨a⟩.



© 2004 R. Lórencz

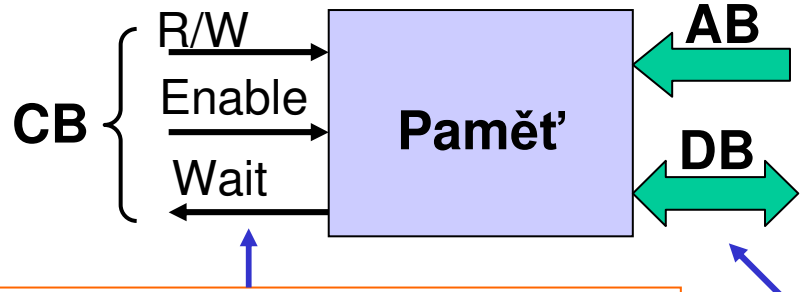
Jednoduchý procesor von Neumannova typu (3)

Paměť

CPU komunikuje s pamětí pomocí 3 typů signálů (buses) – sběrnicích

- Adresní sběrnice (AB – Address Bus), určuje místo v paměti, paměťovou buňku(y)
- Datová sběrnice (DB – Data Bus), slouží k přenosu obsahu místa paměti (dat)
- Řídící sběrnice (CB – Control Bus), řídí přenos obsahu paměti, atd.

R/W – určuje, jestli se bude z paměti číst nebo do ní zapisovat



Šířka AB určuje velikost paměti, pokud AB má 12 b □ 2^{12}=4096 adres

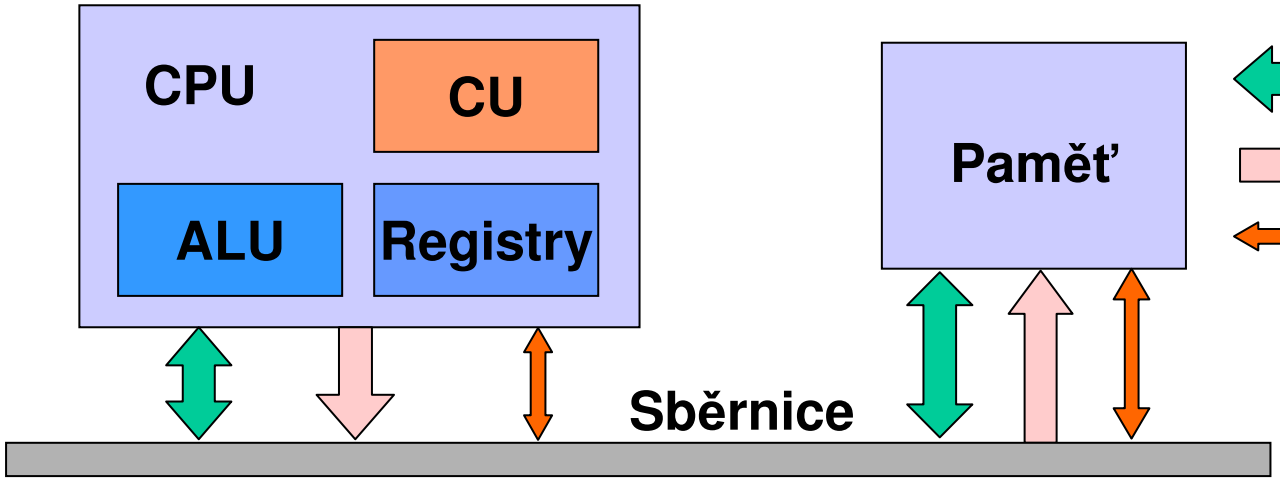
Enable – aktivuje paměť
Wait – informuje CPU o připravenosti z paměti číst nebo do ní zapsat data

Šířka DB určuje velikost obsahu jednoho adresovaného místa,
pokud DB má 16b □ 1 adresou určíme 16b obsahu paměti

© 2004 R. Lórencz

Jednoduchý procesor von Neumannova typu (4)

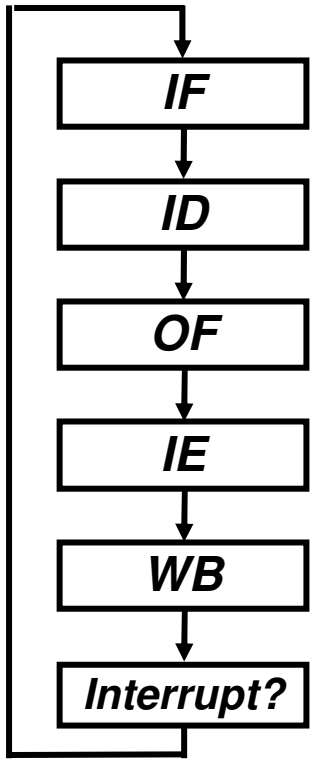
Paměť, CPU



CPU musí obsahovat ALU, CU a registry pro dočasné uchování dat, instrukcí a pomocných proměnných

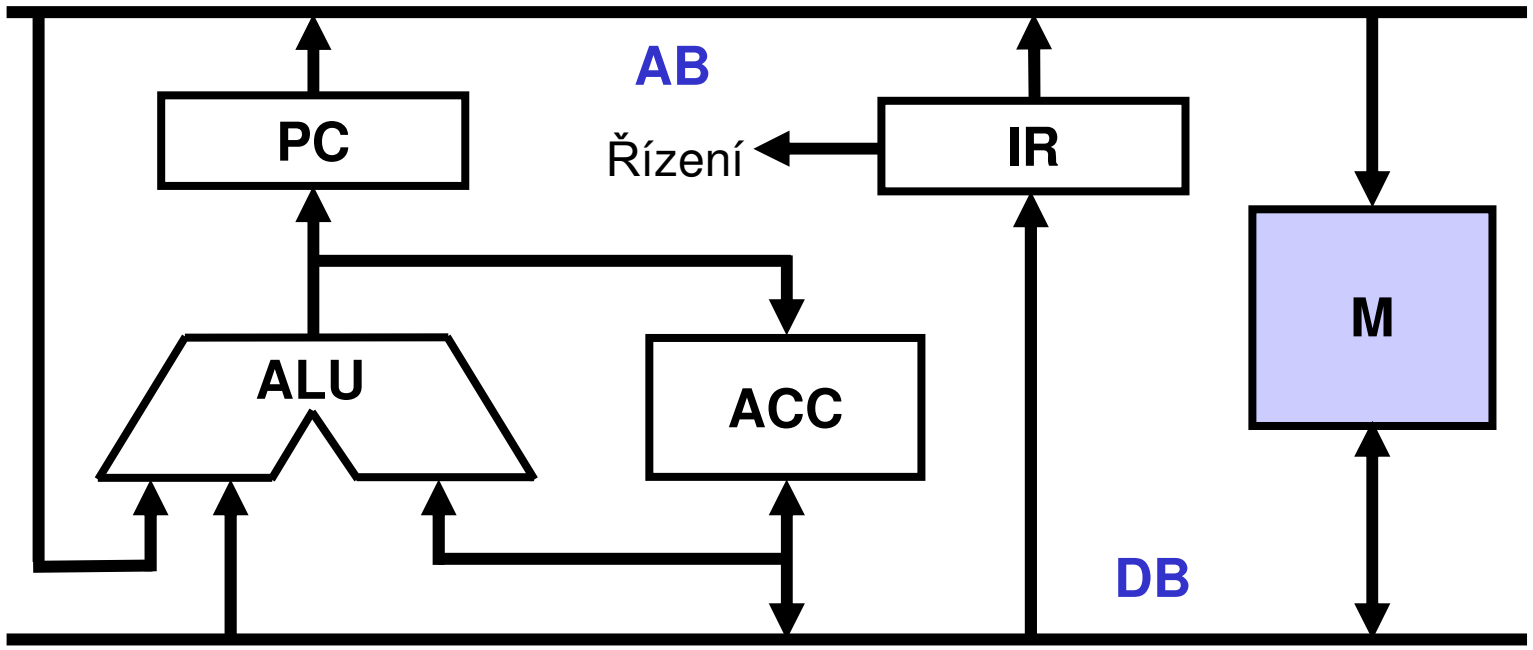
CPU pro běh programu prochází základním cyklem počítače:

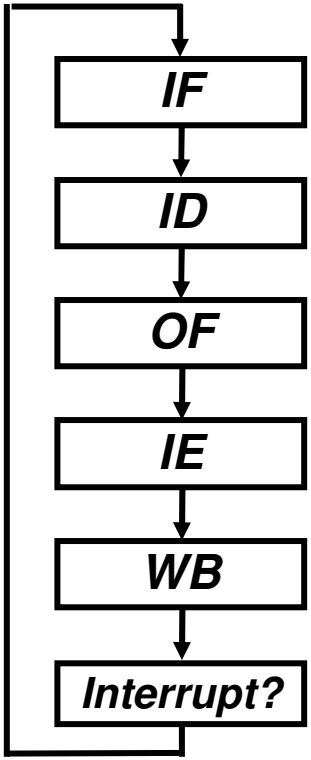
1. Instruction Fetch — IF – načtení instrukce
2. Instruction Decode — ID – dekódování instrukce
3. Operand Fetch — OF – načtení operandu(ů)
4. Instruction Execution — IE – vykonání instrukce
5. Write Back — WB – zapsání výsledku (také Result Store)



© 2004 R. Lórencz

Jednoduchý procesor – P0



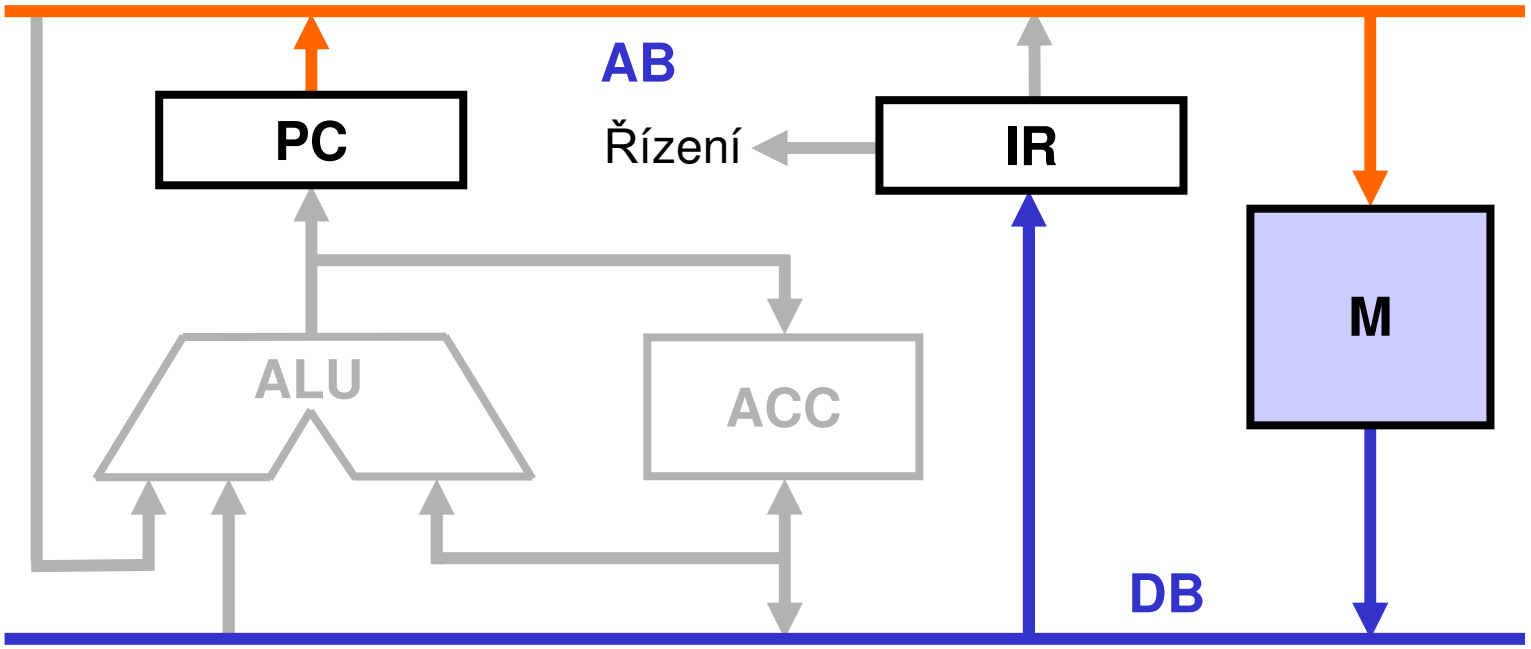


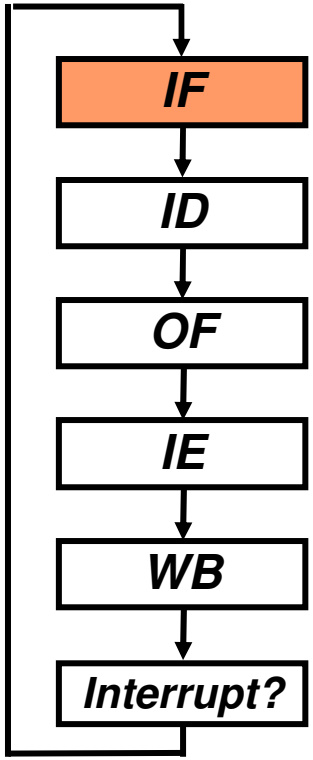
Nechť P0 je procesor s minimálním hardwarem

- M – paměť
- PC – Program Counter, prog. čítač, obsahuje adresu instrukce, která bude vykonána v dalším cyklu
- ACC – Accumulator, střádač, obsahuje zpracovávaná data
- ALU – Arithmetic Logic Unit, aritmeticko-logická jednotka, vykonává operace s daty
- IR – Instruction Register, instrukční registr, obsahuje kód aktuálně prováděné instrukce

© 2004 R. Lórencz

Jednoduchý procesor – P0, IF





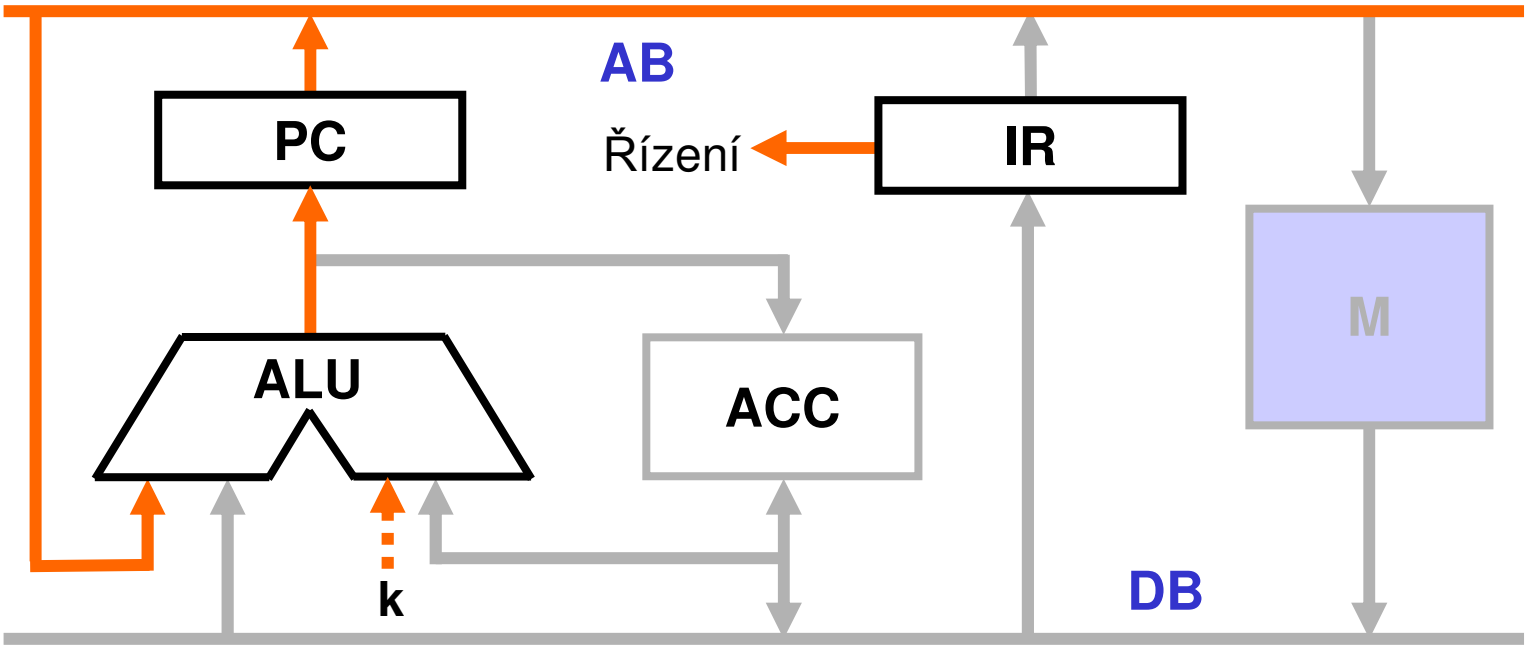
P0 – vystaví adresu instrukce obsazenou v PC na AB

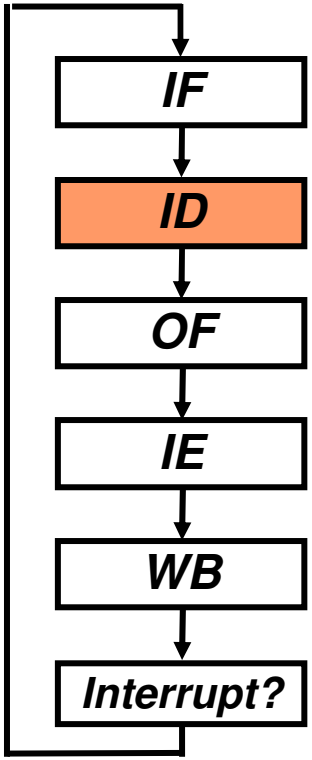
M – vystaví obsah paměťového místa instrukce na DB

IR – kód instrukce z DB je zapsán do IR

© 2004 R. Lórencz

Jednoduchý procesor – P0, ID



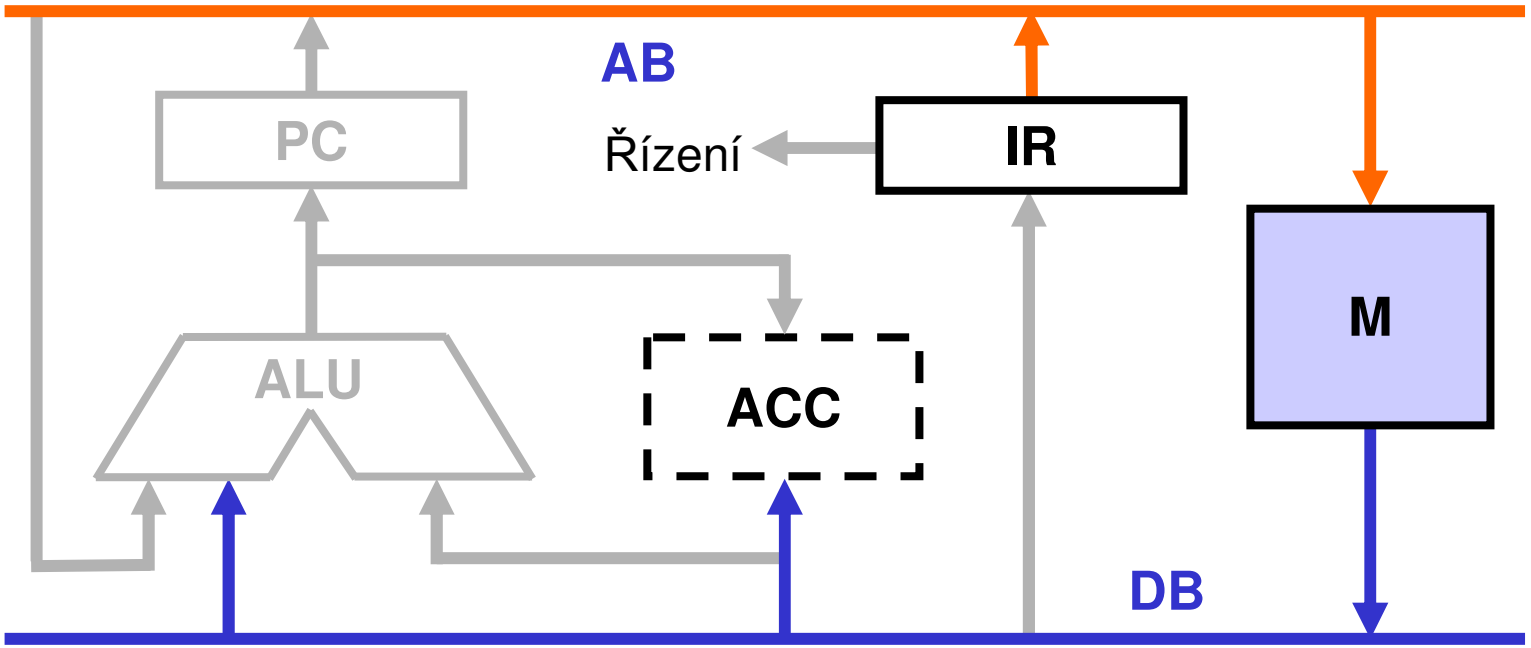


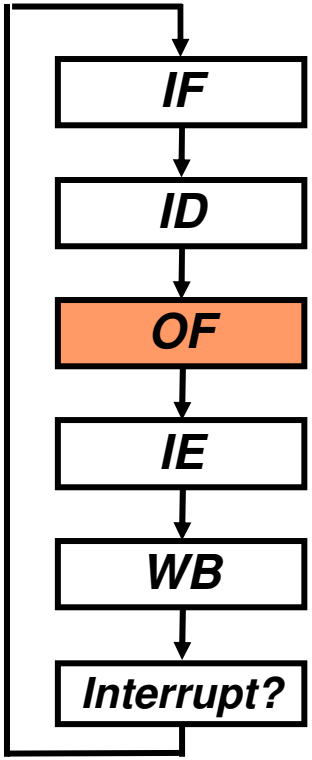
IR – instrukční kód z IR je dekódován interní logikou (dekódérem) a současně jsou generovány řídící signály pro ALU a další interní obvody P0.

PC – programový čítač vystaví hodnotu na AB, ALU zvětší tuto hodnotu o **k** a zapíše nazpět do PC (když **pc** původní obsah PC, ⇒ **pc** ← **pc+k**), hodnota **k** je určena dekódováním instrukce, sekvenční zpracování ⇒**k=1**, skok ⇒ **k** nějaké celé číslo s omezeného intervalu.

© 2004 R. Lórencz

Jednoduchý procesor – P0, OF



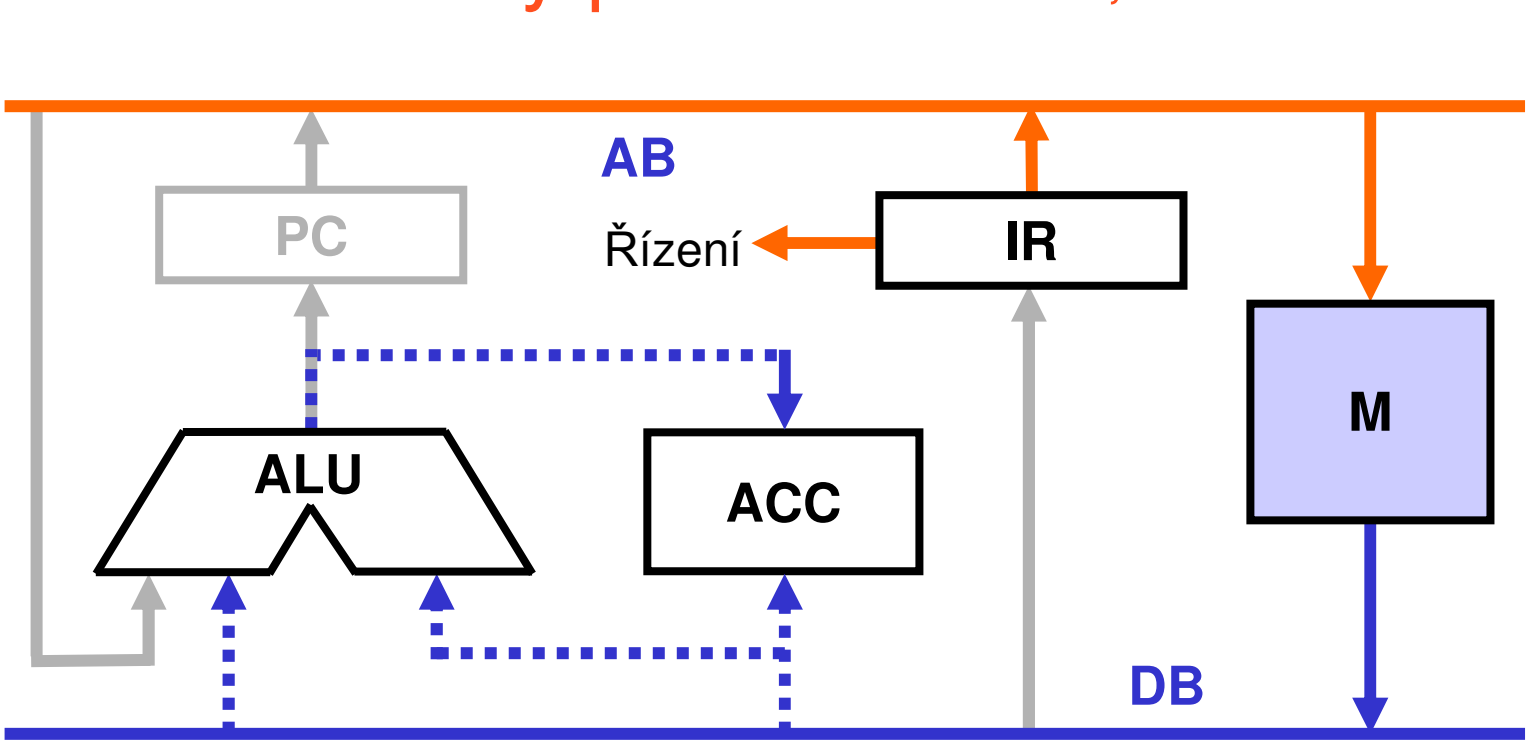


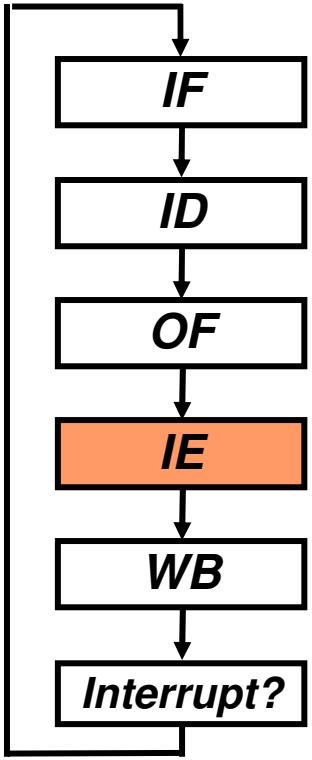
IR – vystavuje na AB adresu operandu, který má být při vykonávání instrukce použit

M – paměť vystaví hodnotu operandu na DB, takto je operand připraven pro zpracování buď v ALU nebo v ACC

© 2004 R. Lórencz

Jednoduchý procesor – P0, IE





IR – vystavuje na AB adresu hodnoty operandu, který má být při vykonání instrukce použit

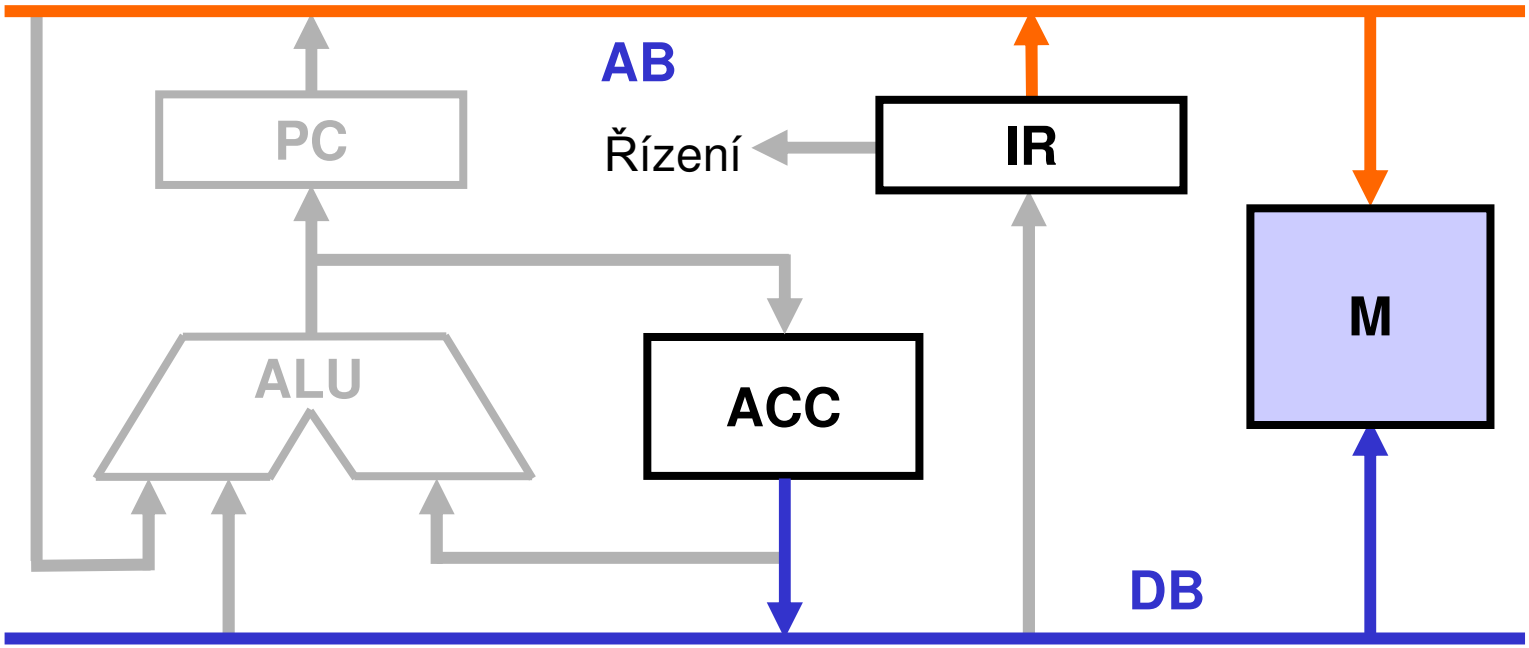
ALU – operace je vykonána v ALU podle instrukce, jejíž kód je v IR a generuje pomocí interní logiky řídící signály

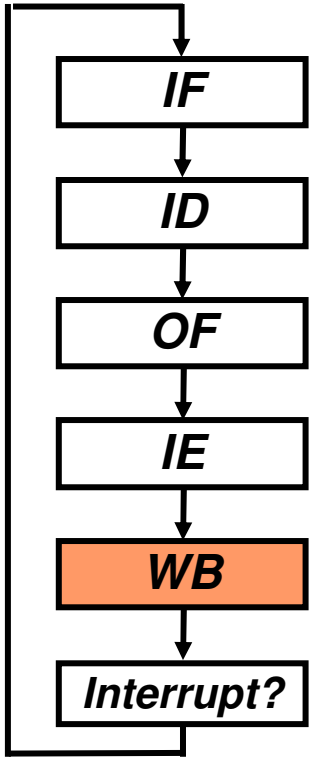
M – paměť může mít nadále vystavenou hodnotu operandu na DB

ACC – slouží jako cílový registr, může být ale také zdrojovým

© 2004 R. Lórencz

Jednoduchý procesor – P0, WB
ne vždy se provádí





IR – vystaví hodnotu adresy paměti, kam má být hodnota obsažená v ACC zapsána

ACC – vystaví hodnotu obsaženou v něm na DB

M – hodnota z DB je zapsána do paměti

Táto fáze základního cyklu může být, nebo nemusí být vykonána. V některých procesorech je táto fáze zákl. cyklu vykonávaná jako separátní instrukce.

© 2004 R. Lórencz

Jednoduchý procesor – P0, instrukce (1)

Předpokládejme že P0:

- Má jen 8 instrukcí
- Může adresovat jen 4 kB paměti (2^{12}) $\Rightarrow$ AB má 12b
- Délka všech instrukcí je 16b
- 16b instrukční kód, strojový kód, má formát:

OP = 4b	S = 12b

kde OP je operační kód, kód operace, také operační znak a S je buď:

- adresa operandu, kterého hodnota je v paměti nebo má být uložená do paměti
- nebo je to hodnota přímého operandu

© 2004 R. Lórencz

Jednoduchý procesor – P0, instrukce (2)

P0 instrukční soubor

Instrukce	OP	Význam
LDA S	0000	ACC ← M[S]
STO S	0001	M[S] ← ACC
ADD S	0010	ACC ← ACC + M[S]
SUB S	0011	ACC ← ACC - M[S]
JMP S	0100	PC ← S
JGE S	0101	If ACC ≥ 0, PC ← S
JNE S	0110	If ACC ≠ 0, PC ← S
STP	0111	Stop (PC ← FE0)

Typy instrukcí:

2 load/store: LDA, STO

2 ALU: ADD, SUB

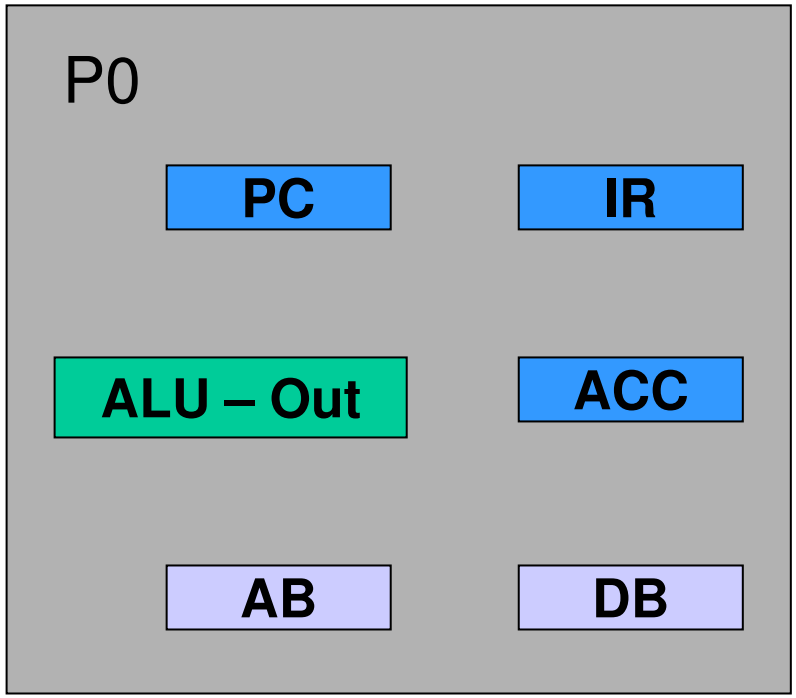
3 větvící: JMP, JGE, JNE

1 řídící: STP

© 2004 R. Lórencz

Jednoduchý procesor – P0, program (1)

Program:



Adresa	Mnemonik	Strojový kód
000	LDA 02E	**0**02E
001	ADD 02F	**2**02F
002	STO 030	**1**030
003	STP	**7**FE0
--	--	--
02E	ABCD	ABCD
02F	4321	4321
030	--	--

Jaké jsou hodnoty registrů PC, IR, ACC, hodnoty na výstupu ALU (ALU – Out) a hodnoty na sběrnicích AB a DB v průběhu vykonávání jednotlivých fází zákl. počítačového cyklu P0?

© 2004 R. Lórencz

Jednoduchý procesor – P0, program (2)

Po resetu jsou všechny hodnoty nastavené na 0.

Stav	PC	IR	ACC	AB	DB	ALU-Out
IF	000	0	0	000	**0**02E	X
ID	000	**0**02E	0	X	X	001
OF	001	**0**02E	0	02E	ABCD	X
IE	001	**0**02E	ABCD	X	X	X

Adresa	Mnemonik	Strojový kód
000	**LDA 02E**	**0**02E
001	ADD 02F	**2**02F
002	STO 030	**1**030
003	STP	**7**FE0
--	--	--
02E	ABCD	ABCD
02F	4321	4321
030	--	--

WB u instrukce LDA se samozřejmě neprovádí. Pokračuje se další instrukcí s adresou v PC, tj. instrukce na adrese 001: ADD 02F

Všimněte si že lze fázi IE sloučit s IF následující instrukce.

© 2004 R. Lórencz

Jednoduchý procesor – P0, program (3)

Pokračujeme prováděním instrukce

Na adrese 001: ADD 02F

Stav	PC	IR	ACC	AB	DB	ALU-Out
IF	001	**0**02E	ABCD	001	**2**02F	X
ID	001	**2**02F	ABCD	X	X	002
OF	002	**2**02F	ABCD	02F	4321	EEEE
IE	002	**2**02F	EEEE	X	X	X

Adresa	Mnemonik	Strojový kód
000	LDA 02E	**0**02E
001	**ADD 02F**	**2**02F
002	STO 030	**1**030
003	STP	**7**FE0
--	--	--
02E	ABCD	ABCD
02F	4321	4321
030	--	--

WB u instrukce ADD se opět neprovádí.

Pokračuje se další instrukcí s adresou v PC, tj. instrukce na adrese 002: STO 030

© 2004 R. Lórencz

Jednoduchý procesor – P0, program (4)

Pokračujeme prováděním instrukce

Na adrese 002: STO 030

Stav	PC	IR	ACC	AB	DB	ALU-Out
IF	002	**2**02F	EEEE	002	**1**030	X
ID	002	**1**030	EEEE	002	X	003
WB	003	**1**030	EEEE	030	EEEE	X

Adresa	Mnemonik	Strojový kód
000	LDA 02E	**0**02E
001	ADD 02F	**2**02F
002	**STO 030**	**1**030
003	STP	**7**FE0
--	--	--
02E	ABCD	ABCD
02F	4321	4321
030	EEEE	EEEE

Při zápisu obsahu ACC do paměti se fáze OF a IE neprovádí.
Končí se fází WB (zápis do paměti)

Pokračuje se další instrukcí s adresou v PC, tj. instrukce na adrese 003: STP

© 2004 R. Lórencz

Jednoduchý procesor – P0, program (5)

Pokračujeme prováděním instrukce

Na adrese 002: STP

Stav	PC	IR	ACC	AB	DB	ALU-Out
IF	003	**1**030	EEEE	003	**7**FE0	X
ID	003	**7**FE0	EEEE	X	X	X
IE	FE0	**7**FE0	EEEE	X	X	X

Adresa	Mnemonik	Strojový kód
000	LDA 02E	**0**02E
001	ADD 02F	**2**02F
002	STO 030	**1**030
003	**STP**	**7**FE0
--	--	--
02E	ABCD	ABCD
02F	4321	4321
030	EEEE	EEEE

Při této instrukci dojde jenom k načtení instrukce s paměti a přepisu PC ve fázi ID na hodnotu nové adresy FE0, na které může být uložený podprogram pro obsluhu přerušení.

To znamená, že se může čekat na znovu spuštění nějakého dalšího programu.

© 2004 R. Lórencz

Jednoduchý procesor – P0, program (6)

V našem programu Instrukce STP ukončí provádění programu tím, že provede skok na adresu FE0, na které je obslužní program například pro synchronní přerušení.

Instrukce JMP S provede stejnou akci, ale s tím rozdílem, že adresa, na které bude program pokračovat je daná přímým operandem S namísto implicitního operandu FE0.

Například pro instrukci 006: JMP 020 dostáváme:

Stav	PC	IR	ACC	AB	DB	ALU-Out
IF	006	**1**030	X	006	**4**020	X
ID	006	**4**020	X	X	X	X
IE	020	**4**020	X	X	X	X

© 2004 R. Lórencz

Jednoduchý procesor – P0, program (7)

Instrukce podmíněného skoku JGE S a JNE S provedou skok na adresu určenou hodnotou S jen když je splněna podmínka, pro JGE S to je: ACC ≥ 0 a pro JNE S: ACC ≠ 0.

Pro test ACC ≥ 0 to nebude problém, testuje se MSB ACC a v případě testu ACC ≠ 0 je možné provést logickou funkci „OR“ na všechny bity ACC (součást ALU).

Z předchozího plyne, že je výhodné mít ještě registr obsahující příznaky vztahující se k obsahu ACC nebo vztahující se k výsledku poslední provedené operace s ALU. Příznakové bity v tomto registru mohou být například:

- příznak nulového obsahu ACC
- příznak záporného obsahu ACC
- příznak přetečení, atd.

© 2004 R. Lórencz