

26. marec 2015

5. prednáška ČÍSLICOVÉ POČÍTAČE



Jana Milanová

Fakulta riadenia a informatiky,
Katedra technickej kybernetiky

PROCESSOR A OBSLUHA V/V ZARIADENÍ

- v prípade, že bude komunikáciu riadiť procesor počítača, môže to urobiť spôsobmi:
 - **priame riadenie** komunikácie procesorom,
 - **využitie prerušenia** procesora pri komunikácii,
 - **priamy prístup k pamäti (DMA)**, ktorý nevyužíva procesor k prenosu dát a používa sa hlavne na komunikáciu s rýchlymi perifériami,

TECHNICKÉ VYBAVENIE PRE PRIAME RIADENIE KOMUNIKÁCIE PROCESSOROM – INFORMÁCIE PRE PROGRAMÁTORA - PRÍKLAD

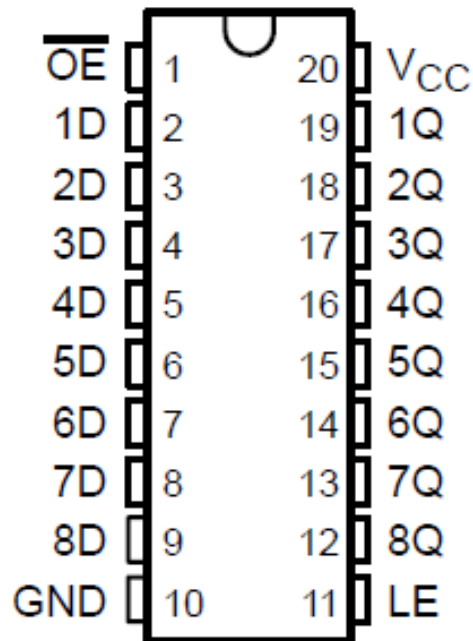
□ Výstup

Adresa	7	6	5	4	3	2	1	0
0xFFFFE	d(7)	d(6)	d(5)	d(4)	d(3)	d(2)	d(1)	d(0)
0xFFFFF	-	-	-	-	-	-	-	STB

□ Vstup

Adresa	7	6	5	4	3	2	1	0
0xFFFFF	-	-	-	-	-	-	-	BUSY

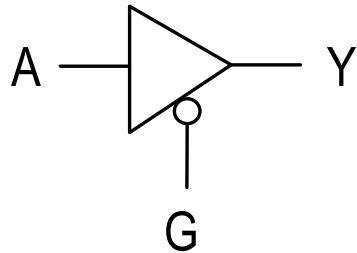
74573 – VYROVNÁVACIA PAMÄŤ



FUNCTION TABLE
(each latch)

INPUTS			OUTPUT Q
\overline{OE}	LE	D	
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

74125 - TROJSTAVOVÝ ODDEĽOVAČ SIGNÁLU



	A	
	0	1
G	Z	Z
	Y	

7474 – VYROVNÁVACIA PAMÄŤ

Table 1 See note 1

INPUT				OUTPUT	
$\overline{\text{SD}}$	$\overline{\text{RD}}$	CP	D	Q	$\overline{\text{Q}}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H	H

Table 2 See note 1

INPUT				OUTPUT	
$\overline{\text{SD}}$	$\overline{\text{RD}}$	CP	D	Q _{n+1}	$\overline{\text{Q}}_{n+1}$
H	H	↑	L	L	H
H	H	↑	H	H	L


Note

1. H = HIGH voltage level;
L = LOW voltage level;
X = don't care;
↑ = LOW-to-HIGH CP transition;
Q_{n+1} = state after the next LOW-to-HIGH CP transition.

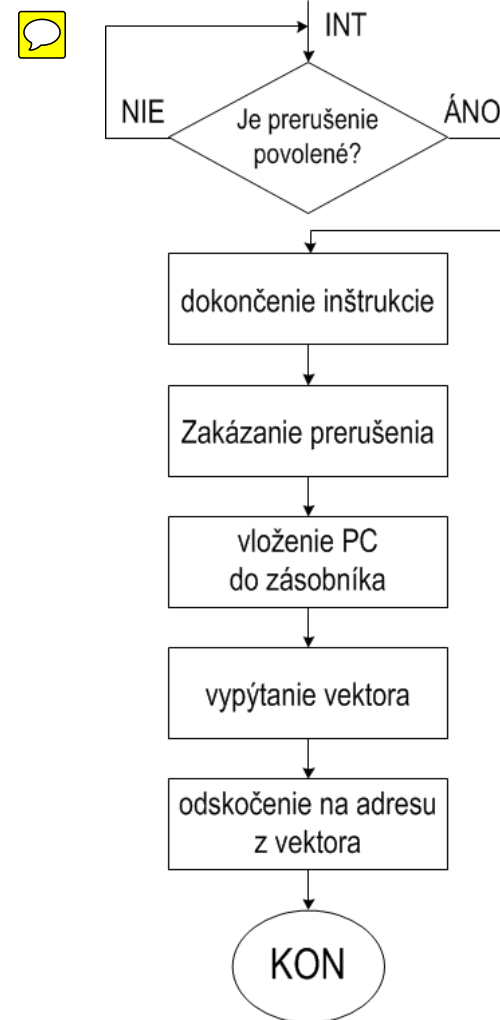
VYUŽITIE PRERUŠENIA PROCESORA PRI KOMUNIKÁCII S PERIFÉRIOU

- procesor sa venuje periférii len v tých okamihoch, keď to periféria potrebuje, z toho vyplýva výrazné šetrenie času procesora a možnosť návrhu počítača s vyšším výkonom,
- výkon takého počítača je však silne ovplyvnený vlastnosťami operačného systému a organizáciou vykonávania užívateľských programov,

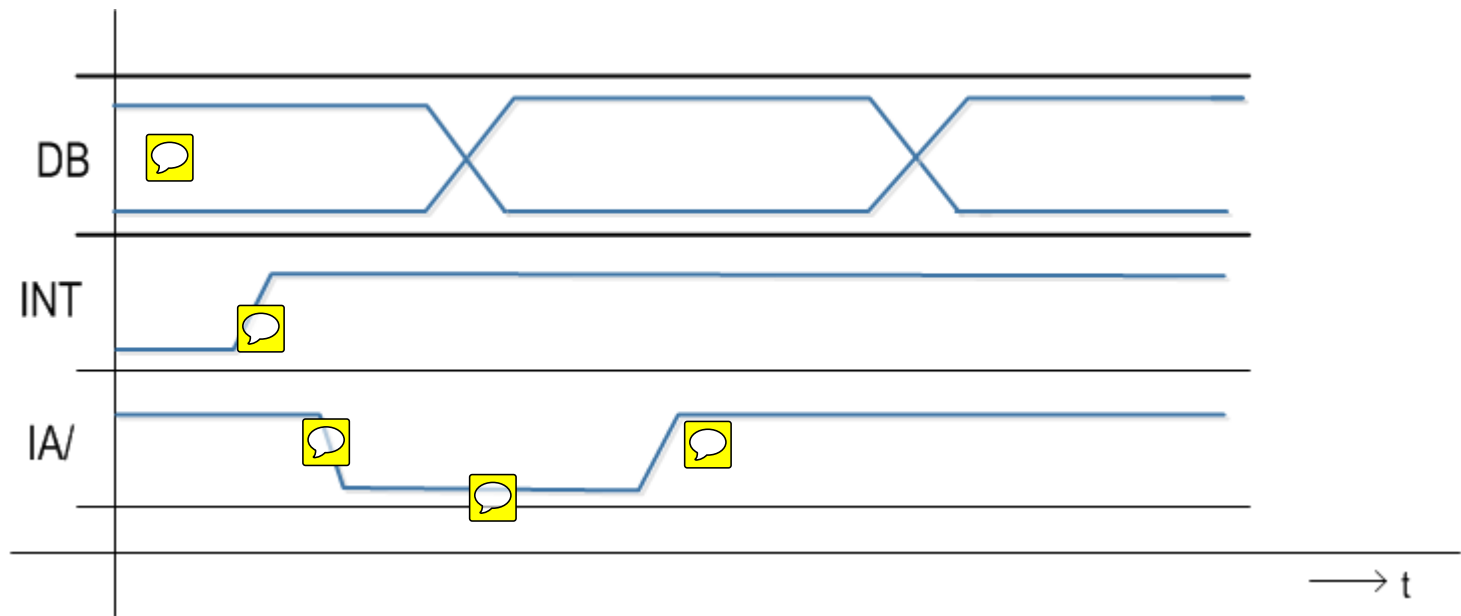
PRERUŠENIE

- prerušenie – mechanizmus, podľa ktorého preruší procesor vykonávanie jedného programu a začne vykonávanie iného programu, ktorý s daným prerušením súvisí – tento program nazveme **obslužný program prerušenia (obsluha prerušenia)**, 
- procesor je obvykle vybavený jedným vstupom často označeným “INT”, ktorým môže okolie **požiadat’ o prerušenie**; Ak sa **signál INT** nastaví na aktívnu úroveň, procesor reaguje na túto skutočnosť nasledovne: v prípade, **ak má zakázané prerušenie** (zákaz prerušenia je možné ovládať dvojicou inštrukcií Enable interrupt (EIT) a **Disable interrupt (DIT)**), **žiadosť o prerušenie ignoruje**,

- ak je prerušenie povolené, dokončí procesor vykonávanie aktuálnej inštrukcie, uloží do zásobníka obsah registra PC procesora (adresu inštrukcie, ktorú by mal vykonať, keby nebol prerušený), nastaví zákaz prerušenia (aby nebol nekontrolovateľne prerušovaný) a prostredníctvom špeciálneho signálu (najčastejšie IA/ (Interrupt Acknowledge non)) požiada, aby mu zariadenie, riadiace prerušenia počítača poslalo informáciu (takzvaný **vektor prerušenia**) o adrese, kde sa nachádza obsluha prerušenia; tento vektor obvykle nie je priamo adresa, ale procesor z neho adresu známym postupom vypočíta; **na túto adresu potom odskočí**,
- povoliť a zakázať prerušenie môže programátor,



SIGNÁLOVÝ SLED PRI PRERUŠENÍ

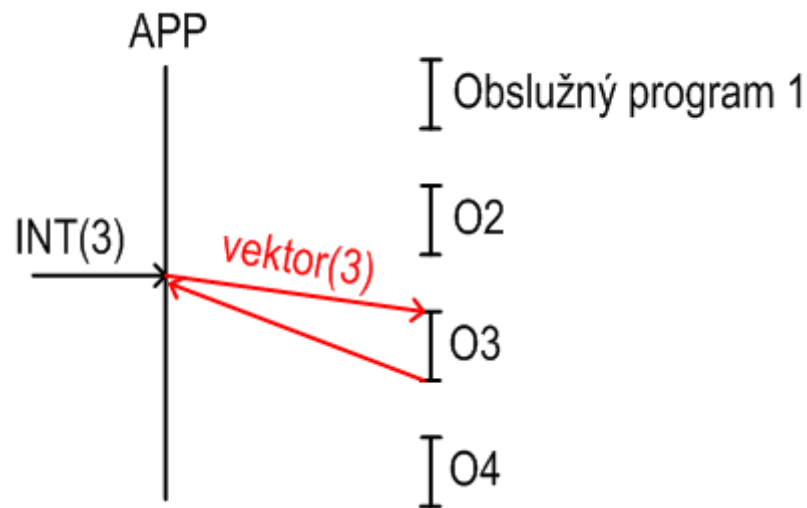


- zariadenie, ktoré **vektor prerušenia na dátovú zbernicu vysiela**, nazývame obvykle **radičom prerušenia**. Proces, ktorým sa procesoru oznamuje adresa obsluhy prerušenia, sa nazýva **identifikáciou zdroja prerušenia**,
- zdrojom prerušenia môže byť (a aj býva) viac ako jeden **prerušovací signál**. Radič prerušenia generuje žiadosť o prerušenie (**INT**) vždy, keď aspoň jeden z týchto prerušovacích signálov je aktívny,
- **v prípade, ak je naraz viac ako jeden aktívny, je treba rozhodnúť o ich dôležitosti (**priorite**) a odskočiť na obsluhu toho najdôležitejšieho (toho s najvyššou prioritou),**
- ak radič prerušenia umožní **selektívne blokovanie** jednotlivých prerušovacích signálov, býva vybavený takzvaným **maskovacím registrom**, ktorý dovoľuje zablokovat' účinok ktorejkoľvek žiadosti o prerušenie; **blokovaciu informáciu budeme nazývať maskou prerušenia**; **pri štarte počítača sú všetky prerušenia zamaskované (zakázané),**
- v závislosti na správaní sa radiča prerušenia, rozoznávame dve rôzne stratégie identifikácie zdroja prerušenia,

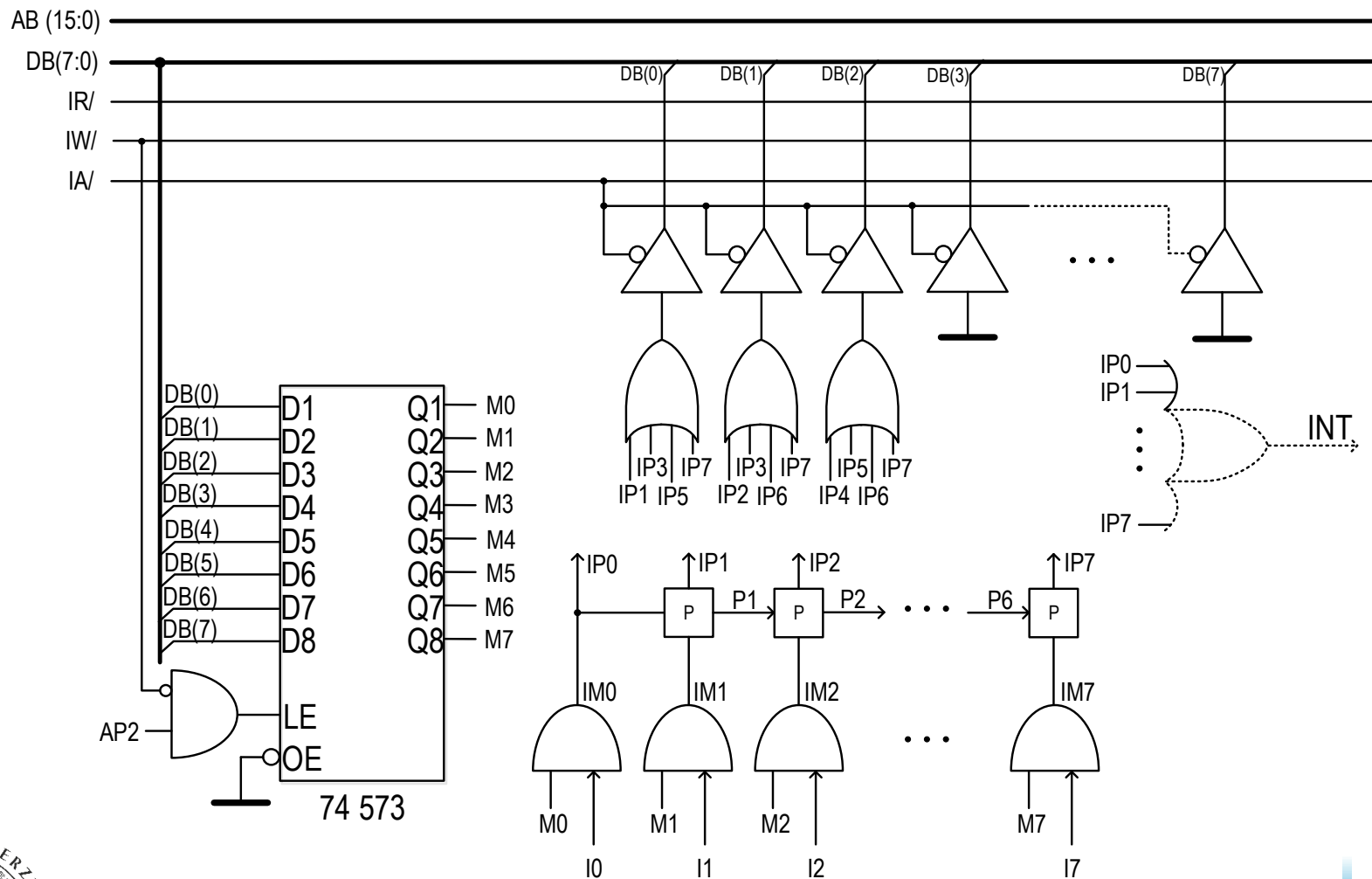
IDENTIFIKÁCIA ZDROJA PRERUŠENIA TECHNICKÝMI PROSTRIEDKAMI

- identifikácia zdroja prerušenia **technickými prostriedkami** (takzvané vektorované prerušenie) – v tomto prípade **radič prerušenia** vyšle ako odozvu na **signál IA/ vektor prerušenia**, ktorý ukazuje na adresu obsluhy aktívneho prerušenia s najvyššou prioritou,
- o **priorite** prerušovacích signálov **musí rozhodnúť už radič prerušenia** technickými prostriedkami,
- identifikácia zdroja prerušenia technickými prostriedkami vyžaduje **zložitejšie technické vybavenie** na strane radiča prerušenia, obvykle **nedovoľuje ľubovoľné nastavenie priority** prerušovacích signálov, avšak reakcia na žiadosť o prerušenie (tzv. **latencia prerušenia**) **je kratšia ako v prípade identifikácie programovými prostriedkami**,

IDENTIFIKÁCIA ZDROJA PRERUŠENIA TECHNICKÝMI PROSTRIEDKAMI



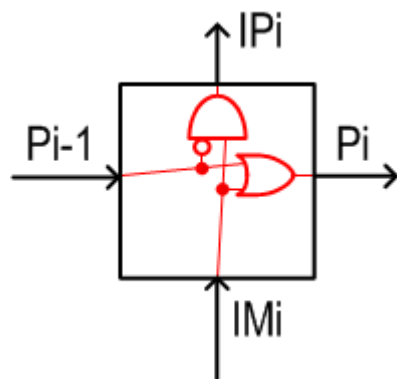
ŠTRUKTÚRA RADIČA PRERUŠENIA PRE IDENTIFIKÁCIU ZDROJA PRERUŠENIA TECHNICKÝMI PROSTRIEDKAMI



		P_{i-1}
	0	0
IM_i	1	0
	IP_i	

		P_{i-1}
	0	1
IM_i	1	1
	P_i	

$$\square IP_i = IM_i \cdot P_{i-1} \quad \square P_i = IM_i \vee P_{i-1}$$



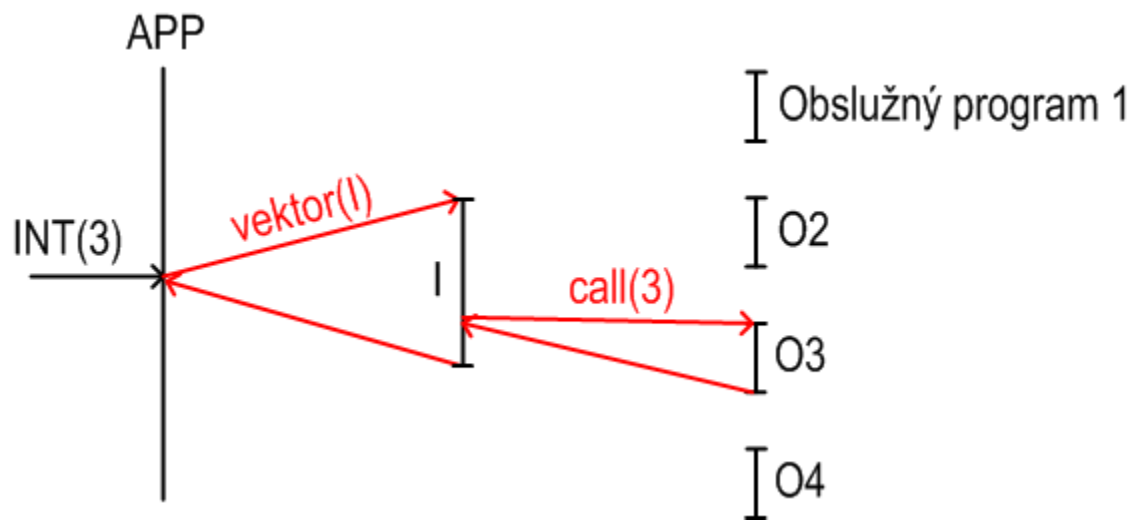
		D_2	D_1	D_0
IP_0	0	0	0	0
IP_1	0	0	0	1
IP_2	0	0	0	1
IP_3	0	0	0	1
IP_4	0	0	0	1
IP_5	0	0	0	1
IP_6	0	0	0	1
IP_7	0	0	0	1

- $\square D_0 - IP_1, IP_3, IP_5, IP_7$
- $\square D_1 - IP_2, IP_3, IP_6, IP_7$
- $\square D_2 - IP_4, IP_5, IP_6, IP_7$

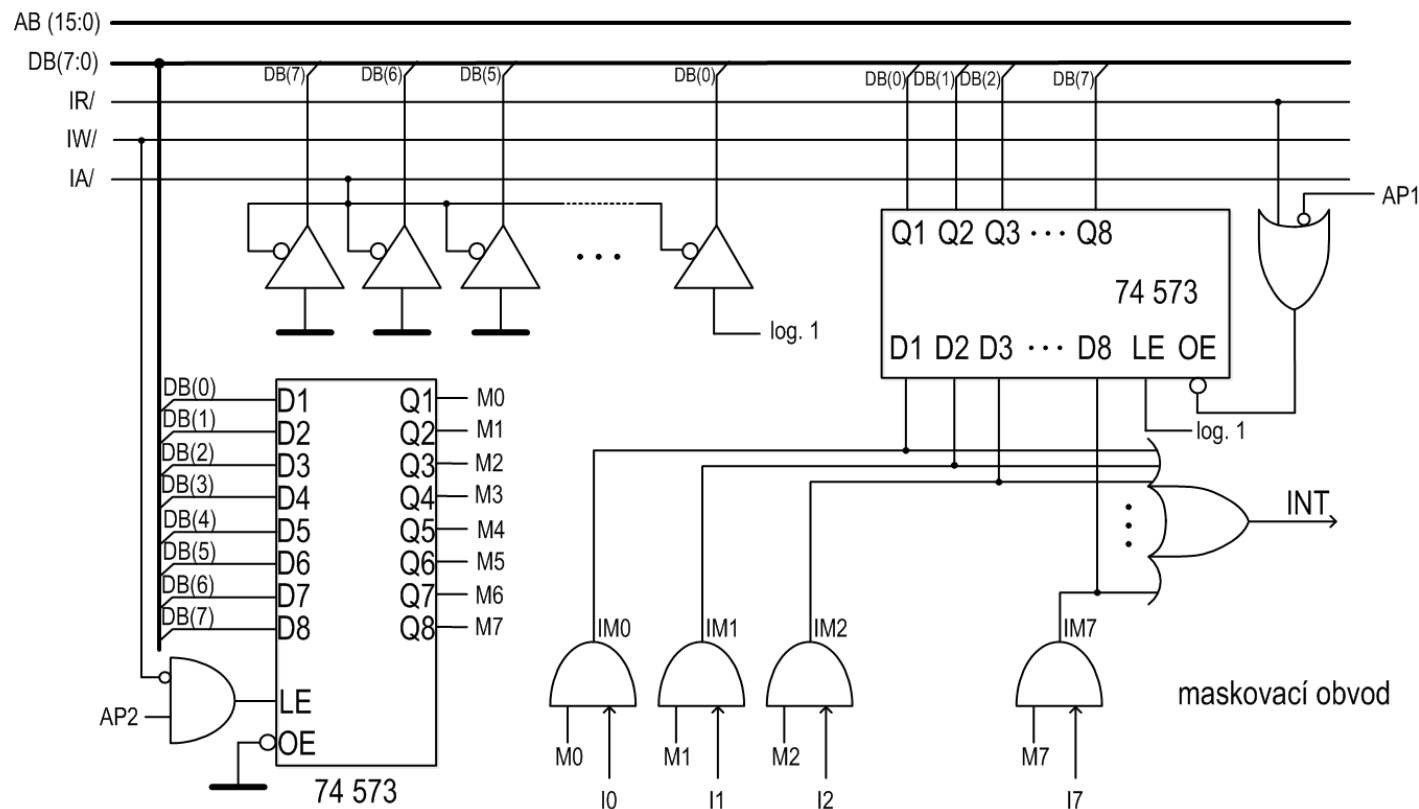
IDENTIFIKÁCIA ZDROJA PRERUŠENIA PROGRAMOVÝMI PROSTRIEDKAMI

- identifikácia zdroja programovými prostriedkami – radič vsúva vektor prerušenia, ktorý nie je od aktívnych prerušovacích signálov a ich priority závislý; vždy je to vektor ukazujúci na **identifikačný program zdroja prerušenia**; v rámci tohto programu potom **procesor prečíta stav všetkých vstupných prerušovacích signálov a môže podľa okamžite platných pravidiel rozhodnúť o ich prioritě a odskočiť na obsluhu toho s najvyššou,**
- **dlhšia latencia prerušenia** je jedinou nevýhodou identifikácie zdroja prerušenia programovými prostriedkami; tento spôsob identifikácie vedie na **jednoduchšie technické vybavenie radiča prerušenia a možnosť dynamickej zmeny priority prerušovacích signálov,**

IDENTIFIKÁCIA ZDROJA PRERUŠENIA PROGRAMOVÝMI PROSTRIEDKAMI



ŠTRUKTÚRA RADIČA PRERUŠENIA PRE IDENTIFIKÁCIU ZDROJA PRERUŠENIA PROGRAMOVÝMI PROSTRIEDKAMI




ak chceme zamaskovať I3, I5 a I7: mvi a, 0x57

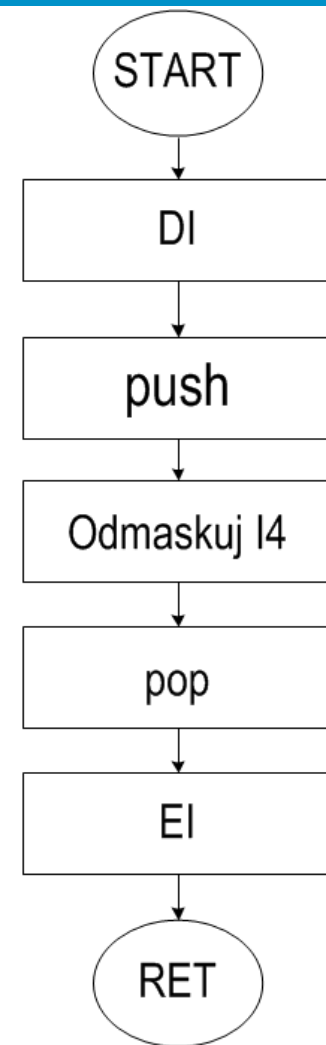
7	6	5	4	3	2	1	0
0	1	0	1	0	1	1	1



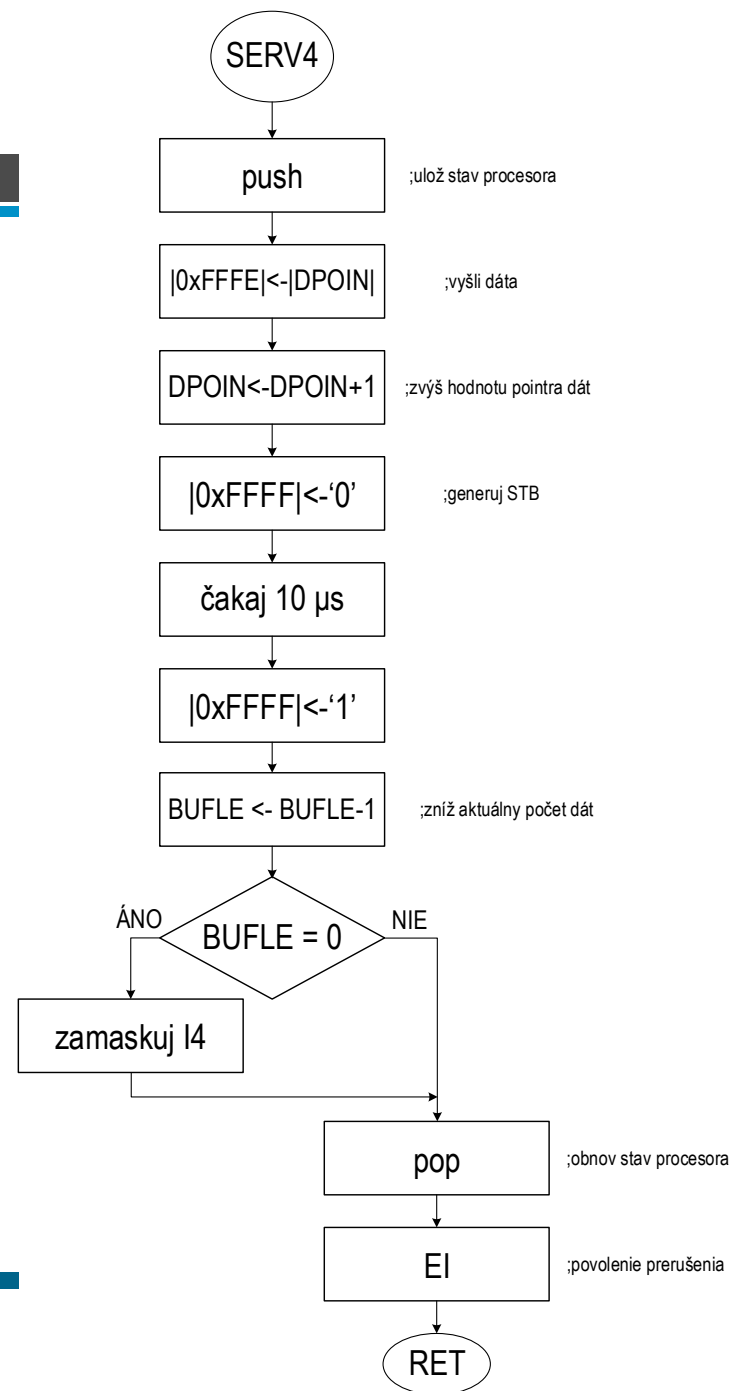
RADIČ PRERUŠENIA

- radič prerušenia:
 1. maskuje prerušovacie vstupy,
 2. priorita (nie vždy rozhoduje radič prerušenia),
- identifikácia zdroja prerušenia:
 - technickými prostriedkami:
 - maskuje,
 - rozhodne o prioritě (napr. podľa indexu),
 - generuje vektor závislý od prioritného prerušenia,
 - ❖ vlastnosti – nižšia latencia, systém tvrdší na modifikáciu, hardvérovo zložitejší,
 - programovými prostriedkami:
 - maskuje,
 - generuje vždy rovnaký vektor, 
 - dovolí procesoru vyčítať stav prerušovacích vstupov,
 - ❖ vlastnosti – pomalšie, ľahšie modifikovateľné, jednoduchší hardvér,

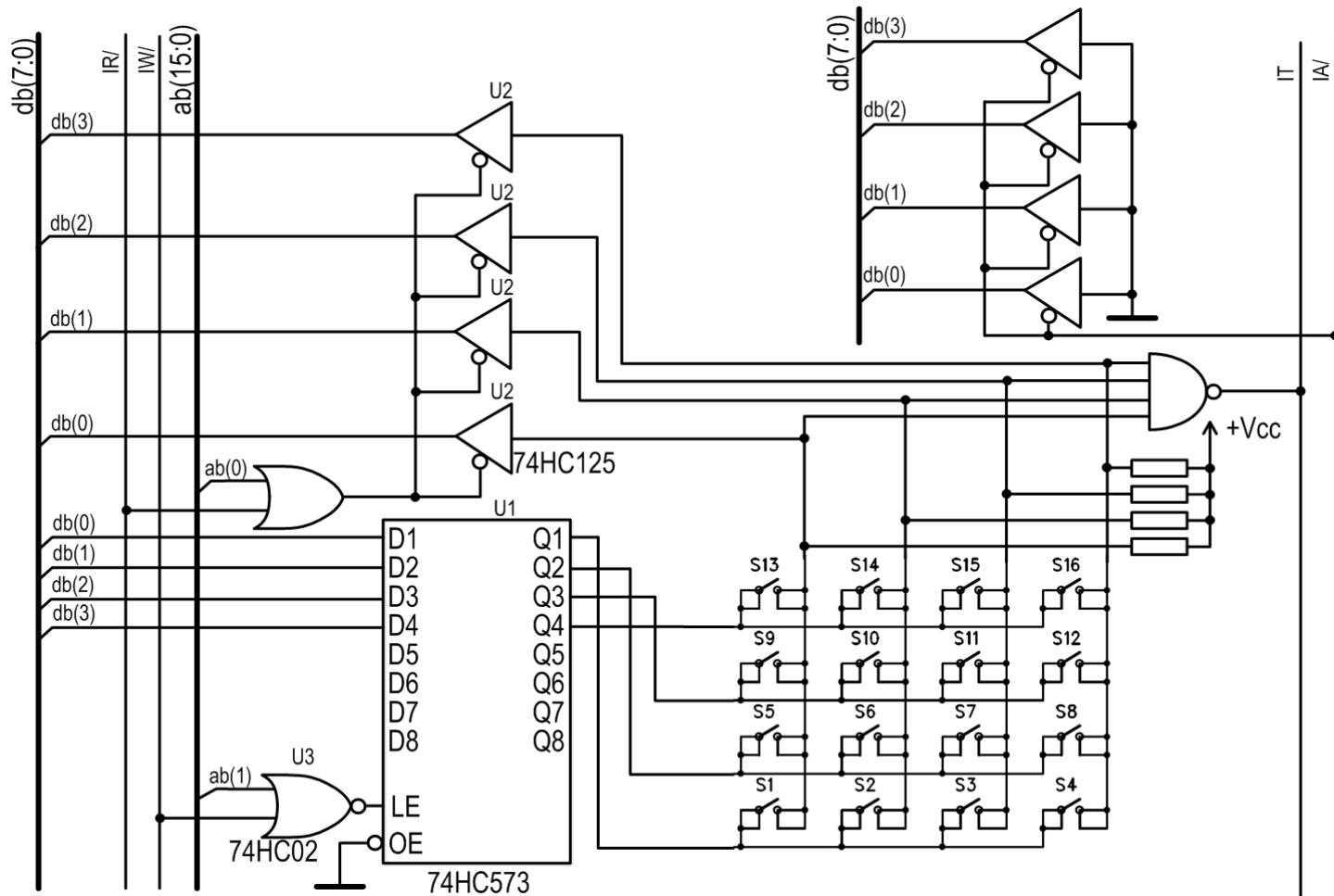
- prvý iba na štart tlačie; nie je to obsluha prerušenia, ale podprogram, ktorý zavolá aplikácia alebo operačný systém po tom, čo vytvoril vyrovnávaciu pamäť dát a zapísal príslušné systémové informácie DPOIN a BUFLE,
- činnosť spočíva hlavne v odmaskovaní úrovne I4 prerušenia, kam je pripojená inverzia signálu BUSY; predtým však odloží obsah tých častí, ktoré môže program štart modifikovať do zásobníka (PUSH), zakáže prerušenie v procesore (DI), po odmaskovaní prerušovacieho vstupu I4 obnoví pôvodný stav procesora a pred návratom z podprogramu povolí prerušenie; zakázanie a povolenie prerušenia plní účel zábrany pred prerušením vo vnútri štartovacieho podprogramu, a teda aj účel udržania kontroly nad dĺžkou zásobníka,



- ❑ Obslužný podprogram prerušovacieho signálu I4,
- ❑ zabezpečuje obsluhu prerušenia, ktoré vzniká pri činnosti periférie vždy vtedy, keď BUSY = 0,
- ❑ uložia sa obsahy tých častí procesora, ktoré sa v priebehu obsluhy môžu zmeniť,
- ❑ z aktuálneho miesta vyrovnávacej pamäte zapíše bajt dát do vyrovnávacej pamäte rozhrania,
- ❑ zvýši sa hodnota pointra vyrovnávacej pamäte tak, aby ukazovala na nasledujúci aktuálny bajt,
- ❑ generovanie impulzu na výstupe STB,
- ❑ aktualizácia zostávajúcej dĺžky vyrovnávacej pamäte znížením BUFLE o 1; ak BUFLE = 0, znamená to, že bol vyslaný posledný bajt dát a obsluha zamaskuje prerušovací vstup I4, čím zabráni ďalšiemu volaniu obsluhy,
- ❑ obsluha končí obvyklou obnovou stavu procesora a povolením prerušenia pred návratom (RET),



PRERUŠENIE A STAVEBNICA K EMULÁTORU PROCESORA



OPERAČNÝ SYSTÉM - MULTITASKING

- technika obsluhy periférií prostredníctvom prerušenia procesora môže významne zlepšiť využitie času procesora,
- podmienky pre efektívne využitie prerušenia –
 - aplikácia (program), ktorá potrebuje vyslať alebo prijať dáta z periférie, nemôže ďalej pokračovať, pokiaľ sa prenos neuskutoční; ak by to bola jediná užitočná činnosť, ktorú počítač vykonáva, musel by procesor čakať na ukončenie komunikácie s perifériou; to by však vzhľadom na využitie času procesora malo zhruba rovnaký efekt, ako keby bola periféria obsluhovaná pod priamym riadením procesora,
 - je dôležité, aby operačný systém počítača bol takzvaný **viacúlohový** (multitasking); v takomto systéme je súčasne niekoľko aplikácií; každá má k dispozícii istý procesorový čas, ktorý jej je cyklicky pridelovaný; zatiaľ čo jedna z aplikácií využíva čas procesora (procesor ju vykonáva, je aktívna), ostatné sú buď čakajúce na pridelenie času procesora (v poradí), alebo sú dočasne mimo poradia z dôvodu čakania na ukončenie udalosti, ktorú znemožňuje ich spustenie; takou udalosťou je typicky V/V komunikácia,
- pri štarte počítača sú všetky prerušenia zamaskované (zakázané),

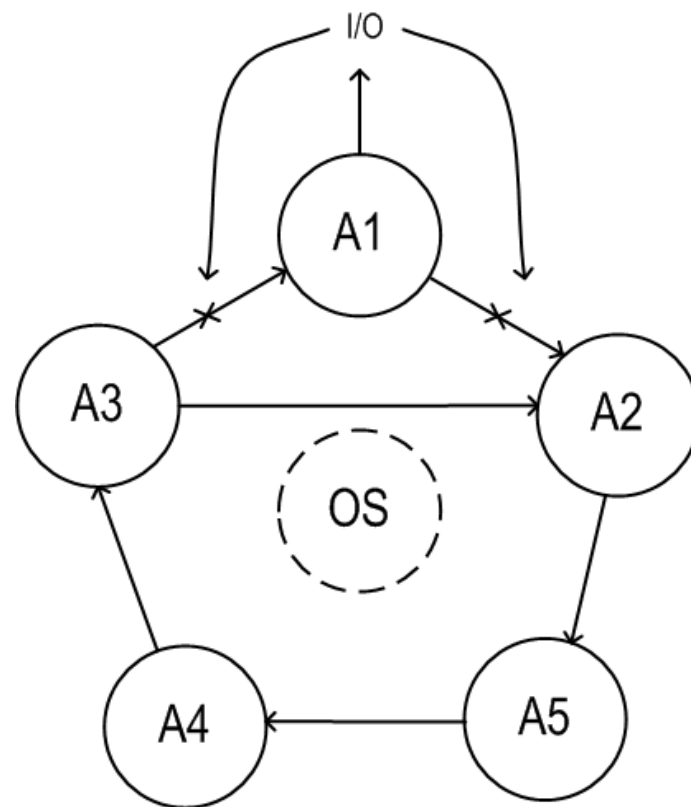
OPERAČNÝ SYSTÉM

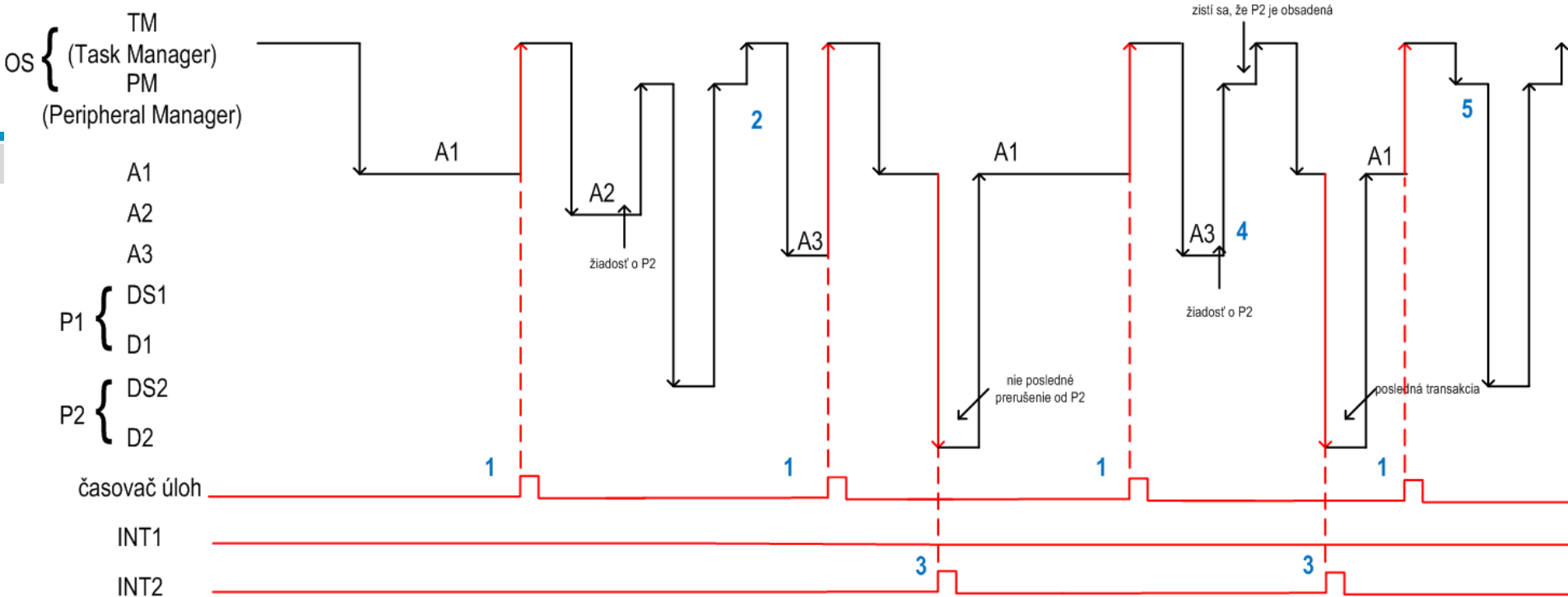
- operačný systém je prvok, ktorý riadi aktiváciu aplikácií a unifikuje prístup k perifériám z dvoch dôvodov:
 - zjednodušenie písania aplikácií, pretože programátor aplikácie nemusí byť oboznámený so spôsobom komunikácie s perifériou, ale stačí, ak využije jednoduché funkcie, ktoré operačný systém ponúka,
 - zásadný dôvod je ten, že komunikácia aplikácie s perifériou zásadne ovplyvní jej „spustiteľnosť“, a preto musí mať operačný systém informáciu o každej takejto činnosti

APLIKÁCIE V OPERAČNOM SYSTÉME

- **činnosť bežiackej aplikácie sa zastaví, ak:**
 - **uplynul jej čas** - procesor nezmení stav aplikácie (zostáva v poradí) a pri ďalšej príležitosti (keď znova na ňu príde rad) ju opäť spustí,
 - **obrátila sa na perifériu** - dočasne presunie mimo poradia a do poradia ju zaradí až po ukončení príslušnej V/V komunikácie,
 - **vykonala nedovolenú operáciu (napr. Runtime error)** - vyradí zo zoznamu a už ju neaktivuje. O tejto skutočnosti oboznámi obsluhu obvykle chybovým hlásením,
- **pravidelné striedanie úloh podľa časovača,**
- ak nemáme multitaskingový systém, nemá až taký význam trápiť sa s interruptami,

APLIKÁCIE V OPERAČNOM SYSTÉME





- OS – operačný systém; A1 , A2, A3 – aplikácie,
- P1, P2 – periférie; DS1, DS2 – štart, D1, D2 – obsluha,
- INT1 – prerušenie od periférie P1, INT 2 – prerušenie od periférie P2

- bežiacia aplikácia sa pozastaví , keď je vygenerovaný impulz od časovača úloh (1),
- ak sa aplikácia obrátila na perifériu, jej činnosť sa pozastaví a spustí sa ďalšia aplikácia (2),
- ak nastane prerušenie (3), činnosť aktuálnej aplikácie sa pozastaví, obslúži sa prerušenie a aplikácia pokračuje vo svojom behu,
- ak aplikácia žiada o perifériu, ktorá je obsadená (4), tá jej bude pridelená až po jej uvoľnení (5).

Ďakujem za pozornosť.

Použité materiály:

Peter Gubiš – Číslicové počítače (podporné učebné texty)

Ondrej Karpiš – Prednášky k predmetu Číslicové počítače