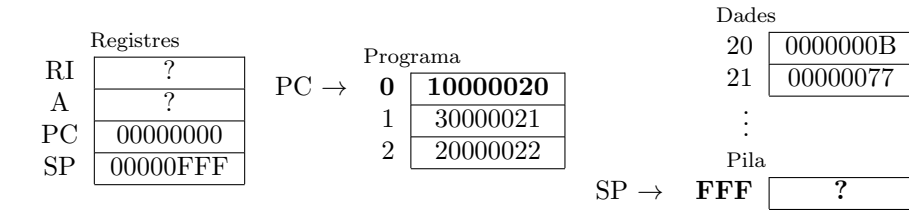


- a) Tenint acumulador, registre d'instrucció de 32 bits li convé tenir un bus de dades de 32 bits. D'altra banda, el fet que el PC (comptador de programa) i el SP (apuntador de pila) tinguin només 24 bits, suggereix que només pot fer adreces a memòria de 24 bits. Bus d'adreces de 24 bits.

Per tant, pot adreçar 2^{24} paraules de 2^{32} bits. És a dir 16M ($2^4 2^{20}$) de paraules de 32 bits. (equivalent a 64 Mbytes d'informació).

[b) Estat Inicial:

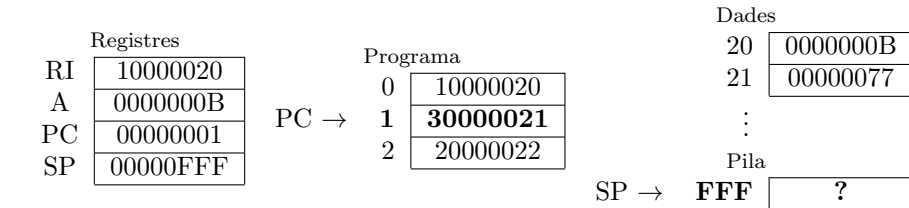


S'executa

Captació Instrucció → $RI := Memoria[PC]$;

Descodificació → $A := Memoria[20]$;

Següent Instrucció → $PC := PC + 1$;

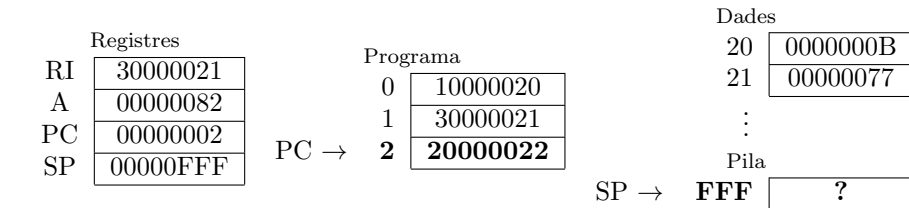


S'executa

Captació Instrucció → $RI := Memoria[PC]$;

Descodificació → $A := A + Memoria[21]$;

Següent Instrucció → $PC := PC + 1$;



S'executa

Captació Instrucció $\rightarrow RI := Memoria[PC];$

Descodificació $\rightarrow Memoria[22] := A;$

Següent Instrucció $\rightarrow PC := PC + 1;$

Registres		Programa		Dades	
RI	20000022	0	10000020	20	0000000B
A	00000082	1	30000021	21	00000077
PC	00000003	2	20000022	22	00000082
SP	00000FFF	PC $\rightarrow$ 3	...	:	
				Pila	
				SP $\rightarrow$ FFF	?

- c La interrupció s'executara quan la instrucció 30000021 hagi acabat deixant els registres en el següent estat:

Registres		Programa		Server Int.		Dades	
RI	30000021	0	10000020	50	A0000001	20	0000000B
A	00000082	1	30000021	51	A0000000	21	00000077
PC	00000002	PC → 2	20000022	52	10000021	:	
SP	00000FFF			53	30000021		
				54	20000022	Pila	
				55	B0000000	SP → FFF	?
				56	B0000001		

Abans de saltar a l'adreça 50, cal salvar el PC actual. Així doncs,

$SP := SP - 1;$

$Memoria[SP] := A;$

$PC := 50$

Registres		Programa		Server Int.		Dades	
RI	30000021	0	10000020	PC → 50	A0000001	20	0000000B
A	00000082	1	30000021	51	A0000000	21	00000077
PC	00000050	2	20000022	52	10000021	:	
SP	00000FFE			53	30000021	Pila	
				54	20000022	SP → FFE	00000002
				55	B0000000	FFF	?
				56	B0000001		

En el servei d'interrupció, hi ha dos empilaments a l'inici:

$A0000001 \rightarrow SP := SP - 1; Memoria[SP] := PC; PC := PC + 1$
 $\{PC = 51, SP = FFD\}$

$A0000000 \rightarrow SP := SP - 1; Memoria[SP] := A; PC := PC + 1$
 $\{PC = 52, SP = FFC\}$

Registres		Programa		Server Int.		Dades	
RI	A0000000	0	10000020	50	A0000001	20	0000000B
A	00000082	1	30000021	51	A0000000	21	00000077
PC	00000052	2	20000022	52	10000021	⋮	⋮
SP	00000FFC			53	30000021	Pila	
				54	20000022	SP →	FFC 00000082
				55	B0000000	FFD	00000050
				56	B0000001	FFE	00000002
						FFF	?

... Després hi ha unes instruccions semblants a la del programa anterior.

$10000021 \rightarrow A := \text{Memoria}[21]; PC := PC + 1 \{A = 77, PC = 53\}$
 $30000021 \rightarrow A := A + \text{Memoria}[21]; PC := PC + 1 \{A = EE, PC = 54\}$
 $20000022 \rightarrow \text{Memoria}[22] := A; PC := PC + 1 \{A = EE, PC = 55\}$

Registres		Programa		Server Int.		Dades	
RI	A0000000	0	10000020	50	A0000001	20	0000000B
A	000000EE	1	30000021	51	A0000000	21	00000077
PC	00000055	2	20000022	52	10000021	22	000000EE
SP	00000FFC			53	30000021	⋮	⋮
				54	20000022	Pila	
				55	B0000000	SP →	FFC 00000082
				56	B0000001	FFD	00000050
						FFE	00000002
						FFF	?

...I els desempilaments

$B0000000 \rightarrow A := \text{Memoria}[SP]; SP := SP + 1; PC := PC + 1$
 $\{PC = 56, SP = FFD\}$
 $B0000001 \rightarrow PC := \text{Memoria}[SP]; SP := SP + 1; \{PC = 50, SP = FFE\}$

Registres		Programa		Server Int.		Dades	
RI	B0000001	0	10000020	PC → 50	A0000001	20	0000000B
A	00000082	1	30000021	51	A0000000	21	00000077
PC	00000050	2	20000022	52	10000021	22	000000EE
SP	00000FFE			53	30000021	⋮	⋮
				54	20000022	Pila	
				55	B0000000	SP →	FFE 00000002
				56	B0000001	FFF	?

... Essent un desastre, ja que es torna a activar el servei d'interrupció, no tornant mai al programa interromput.

- d) En el bus de dades de 8 bits caldrà enviar 32 bits de dada en quatre peces de 8 bits en el temps. Per tant l'accés a dades es fa quatre cops més lent.

En el bus d'adreces de 16 bits, caldrà donar l'adreça de 24 bits en dos cops i per tant es farà el doble de temps que si tinguéssim un bus de 24 bits.