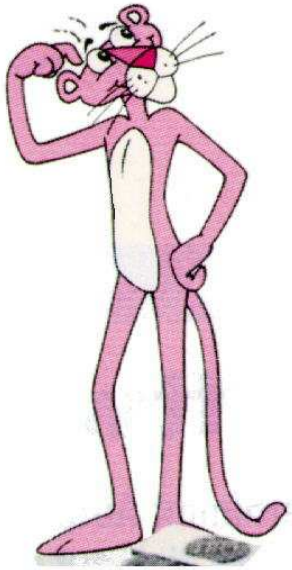
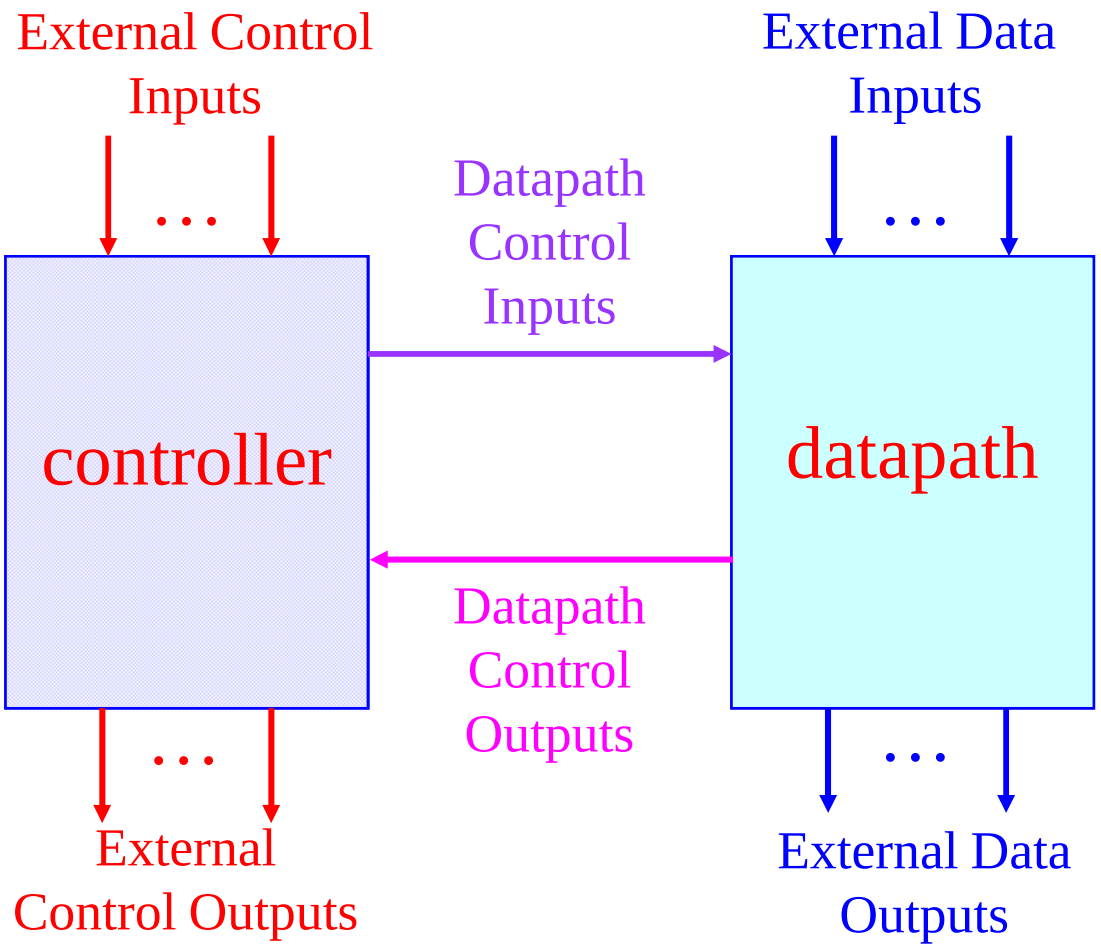
Design of Control Units

Repeat

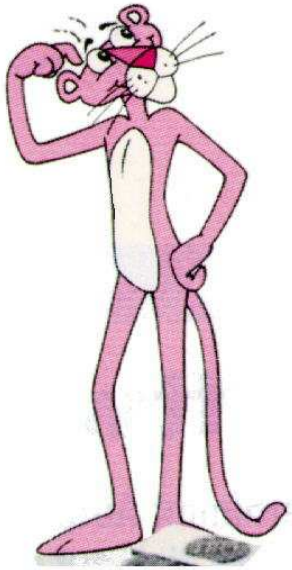
1. Single-Purpose Processor Basic Model

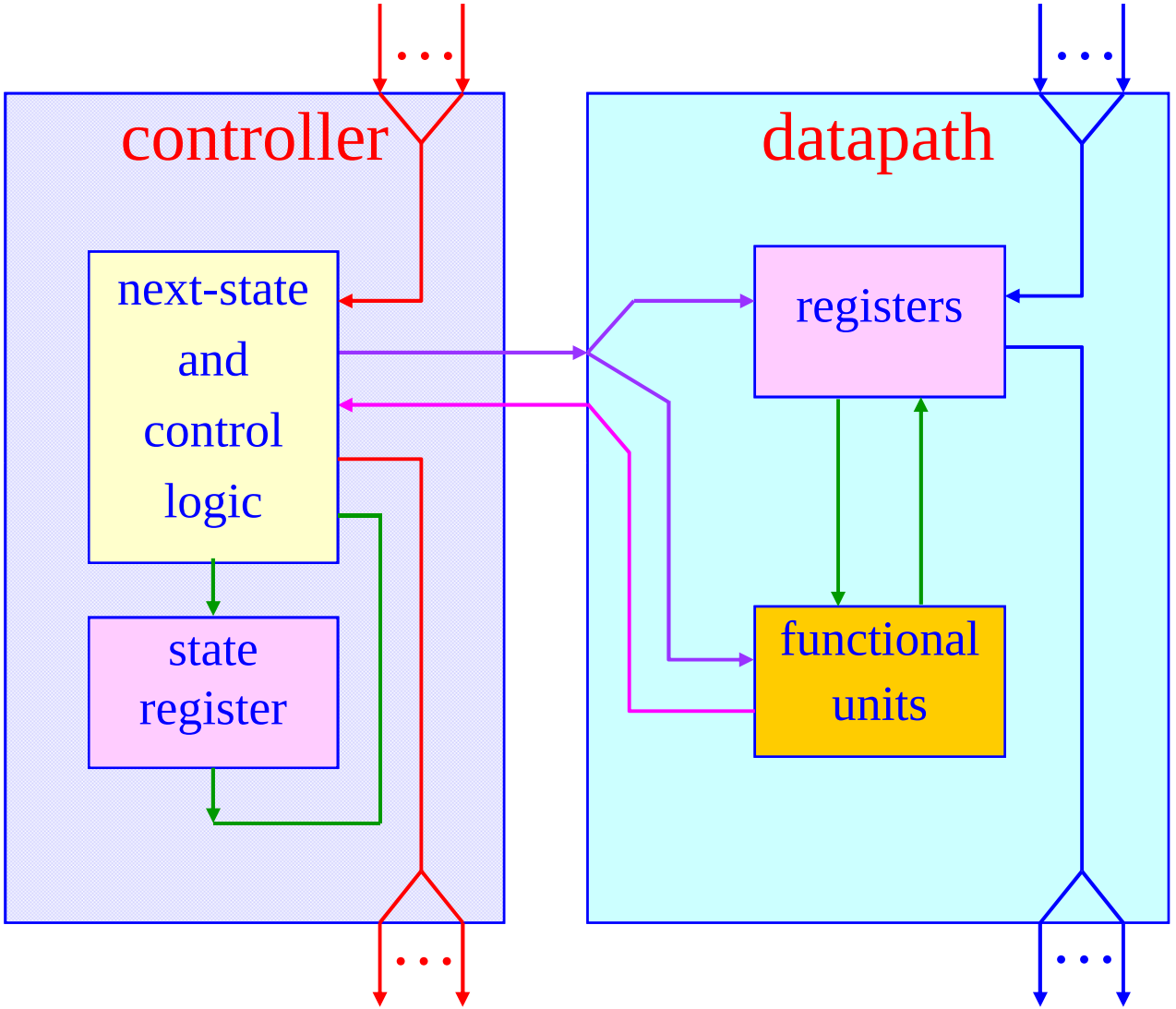




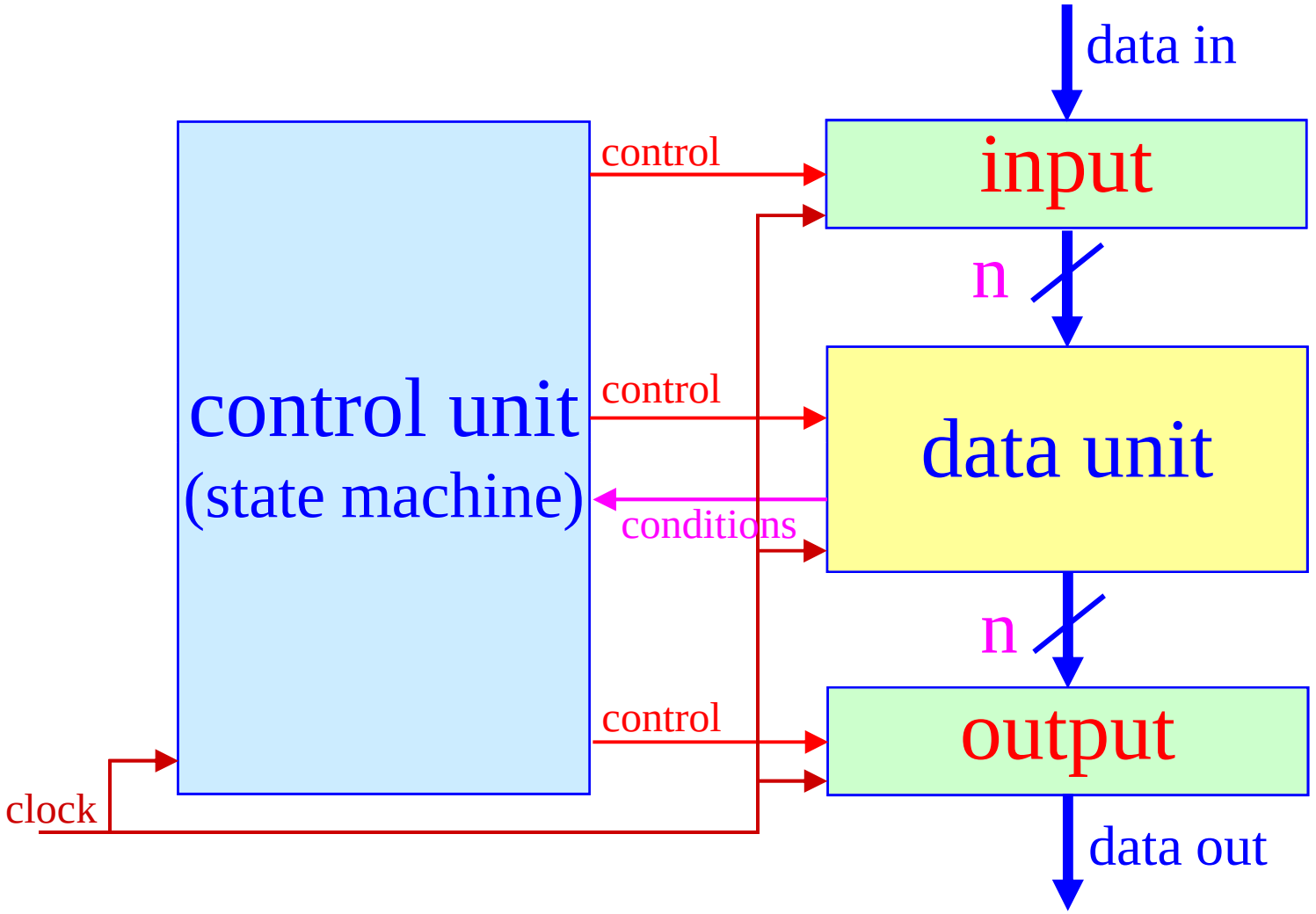
Controller and Datapath

1. View inside: controller and datapath

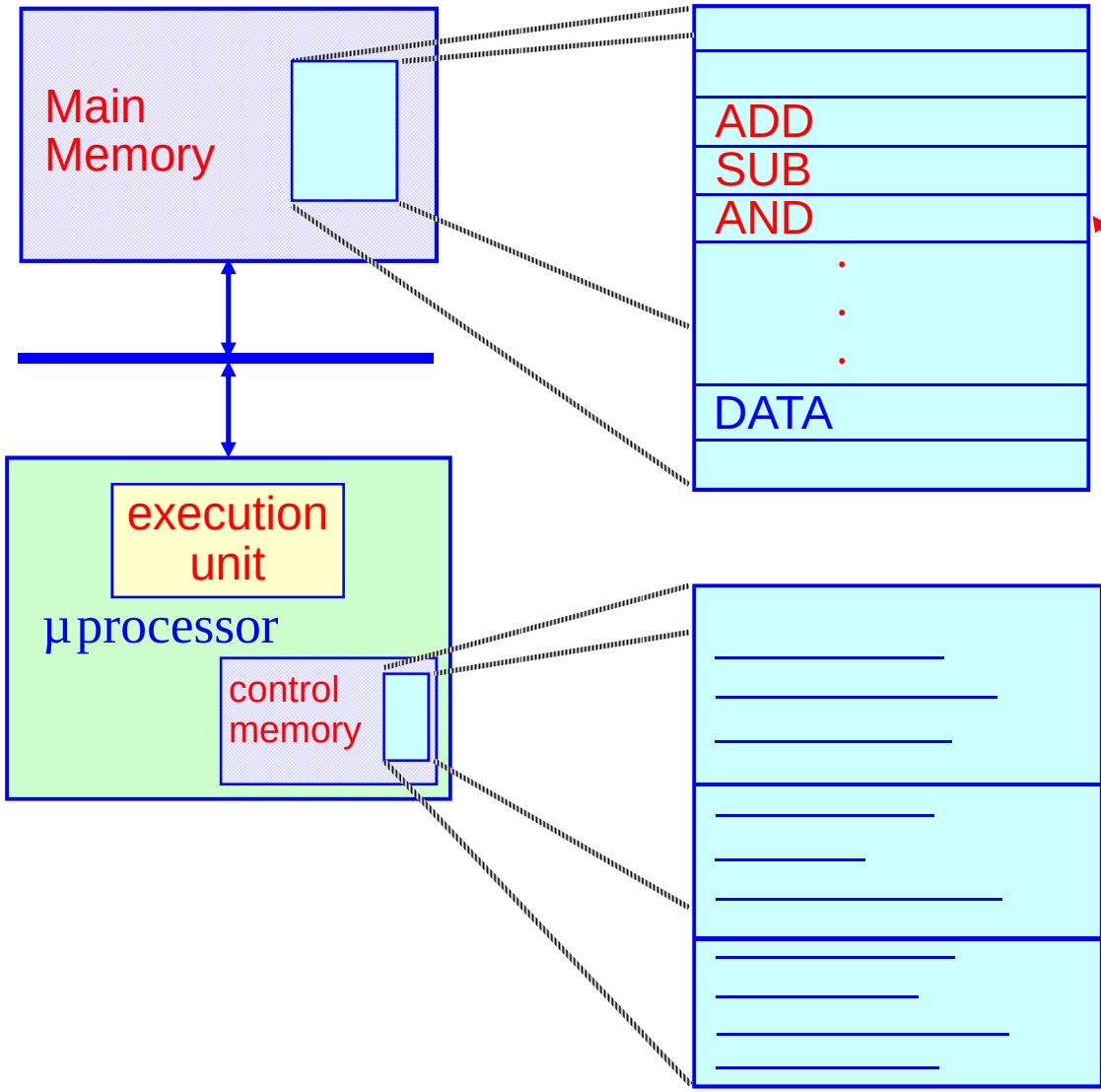




1. Synchronous System Structure



1. “Macro and micro-instruction” Interpretation



User program
plus Data

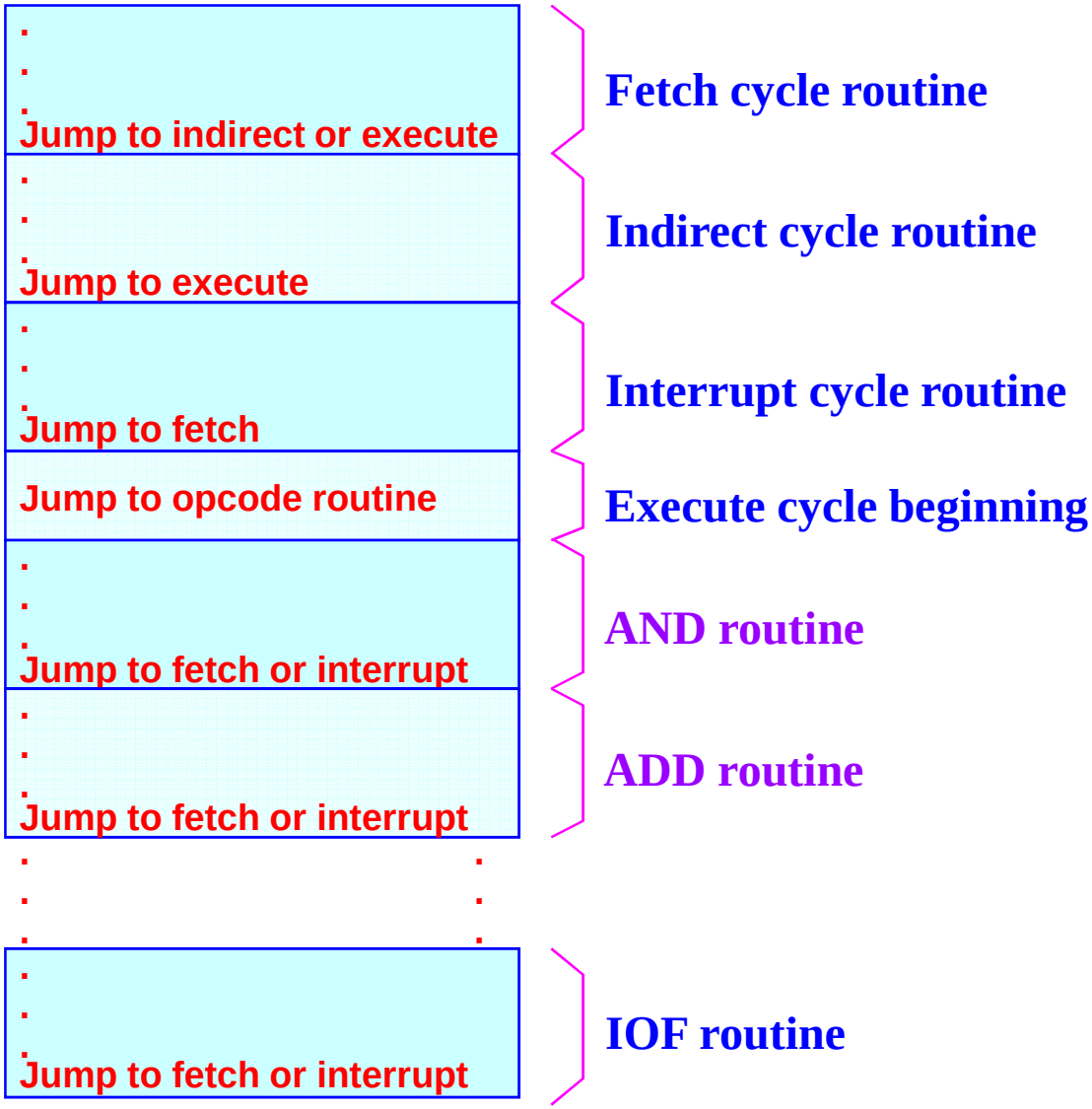
this can change!

one of these is
mapped into one
of these

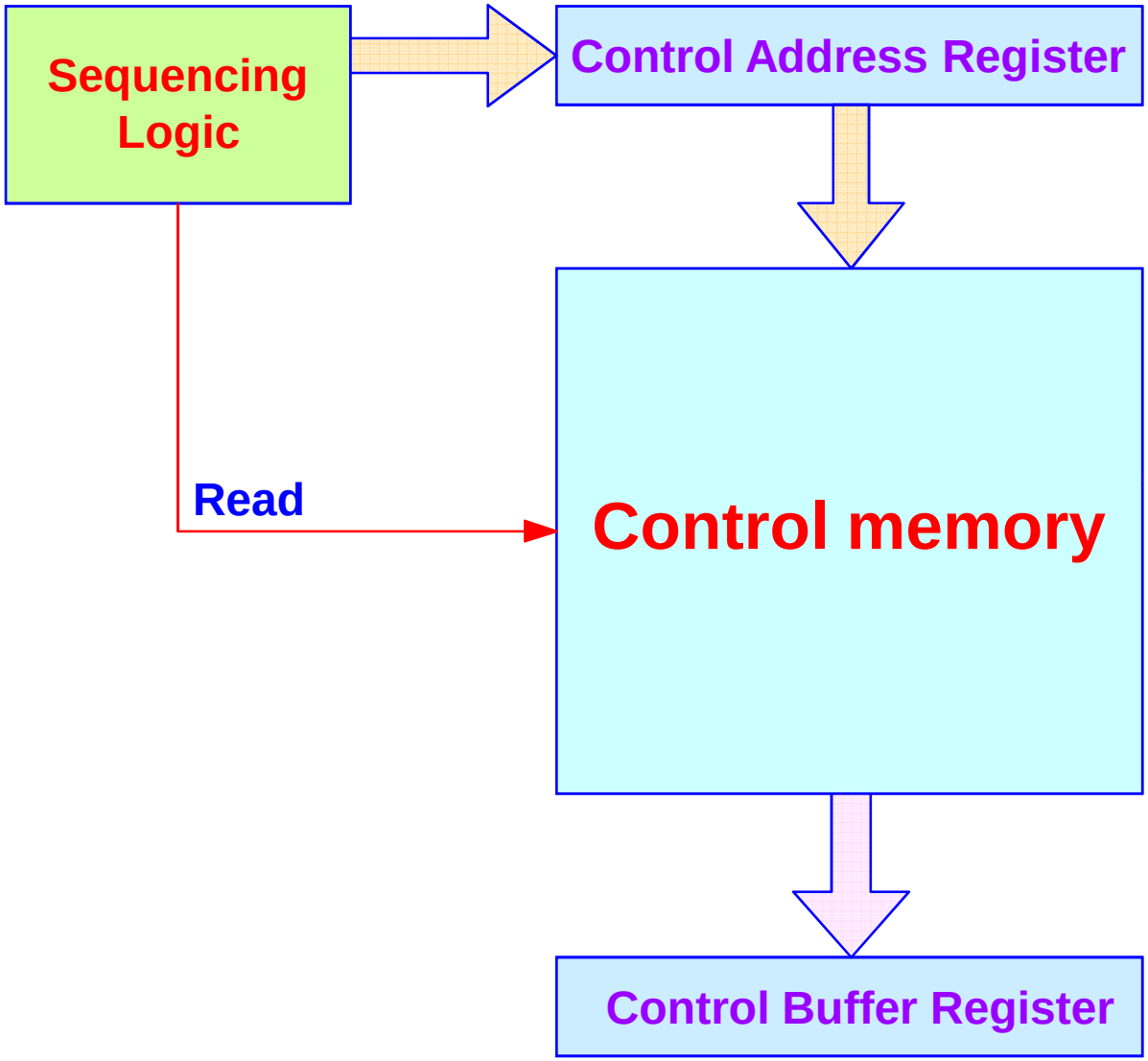
AND microsequence

e.g. Fetch
Decode: calc operand addr
Fetch operand(s)
Execute: calculate
Store results: save answer(s)

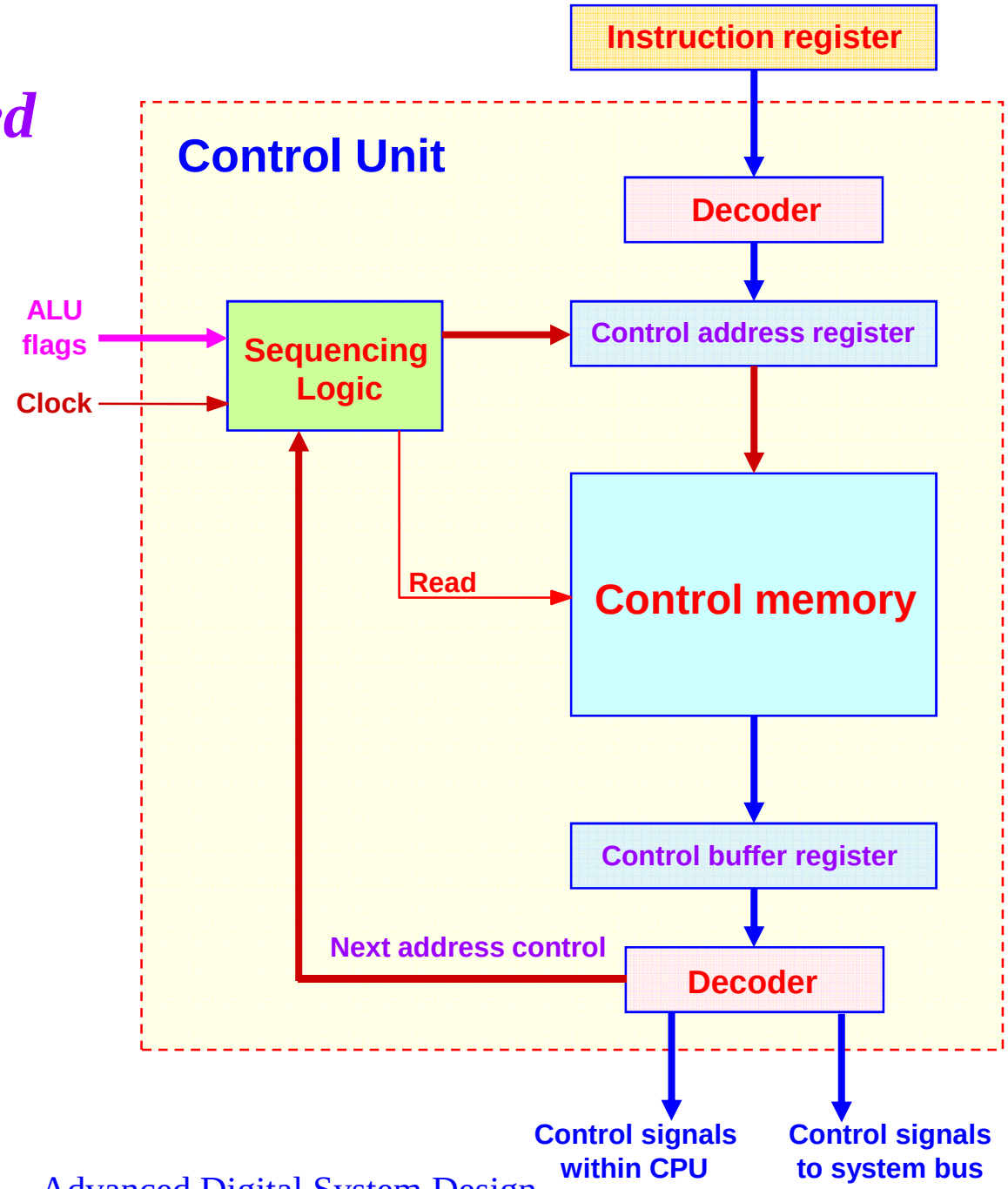
1. Organization of Control Memory



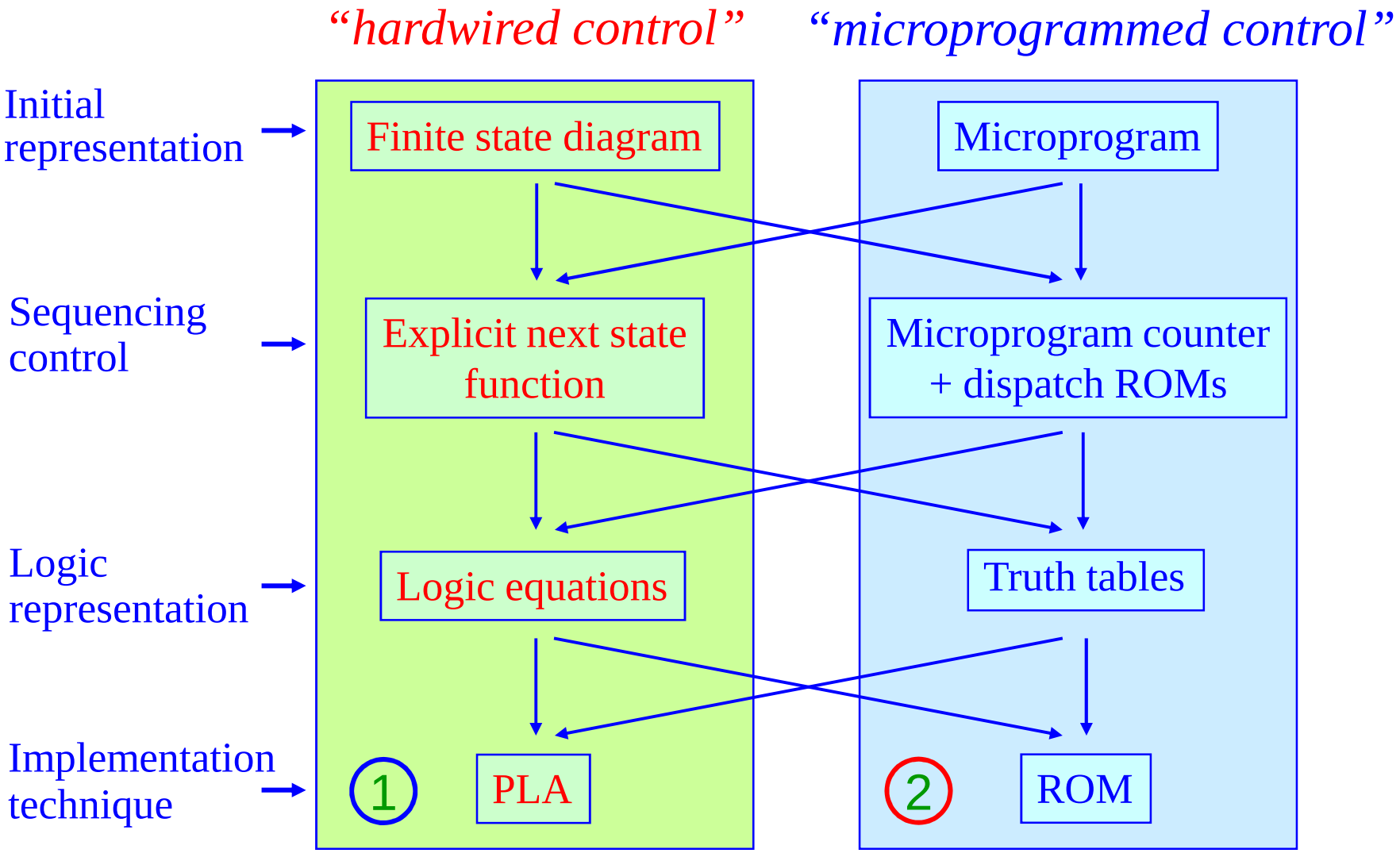
1. Control Unit Micro architecture



Functioning of Microprogrammed Control Unit



1. Controller Design

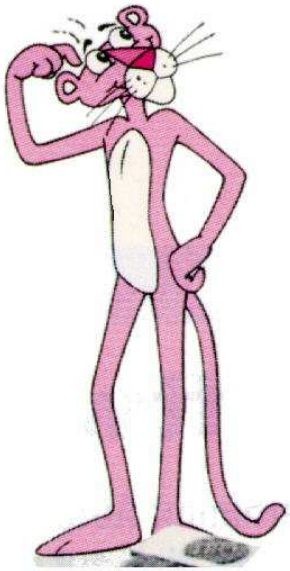


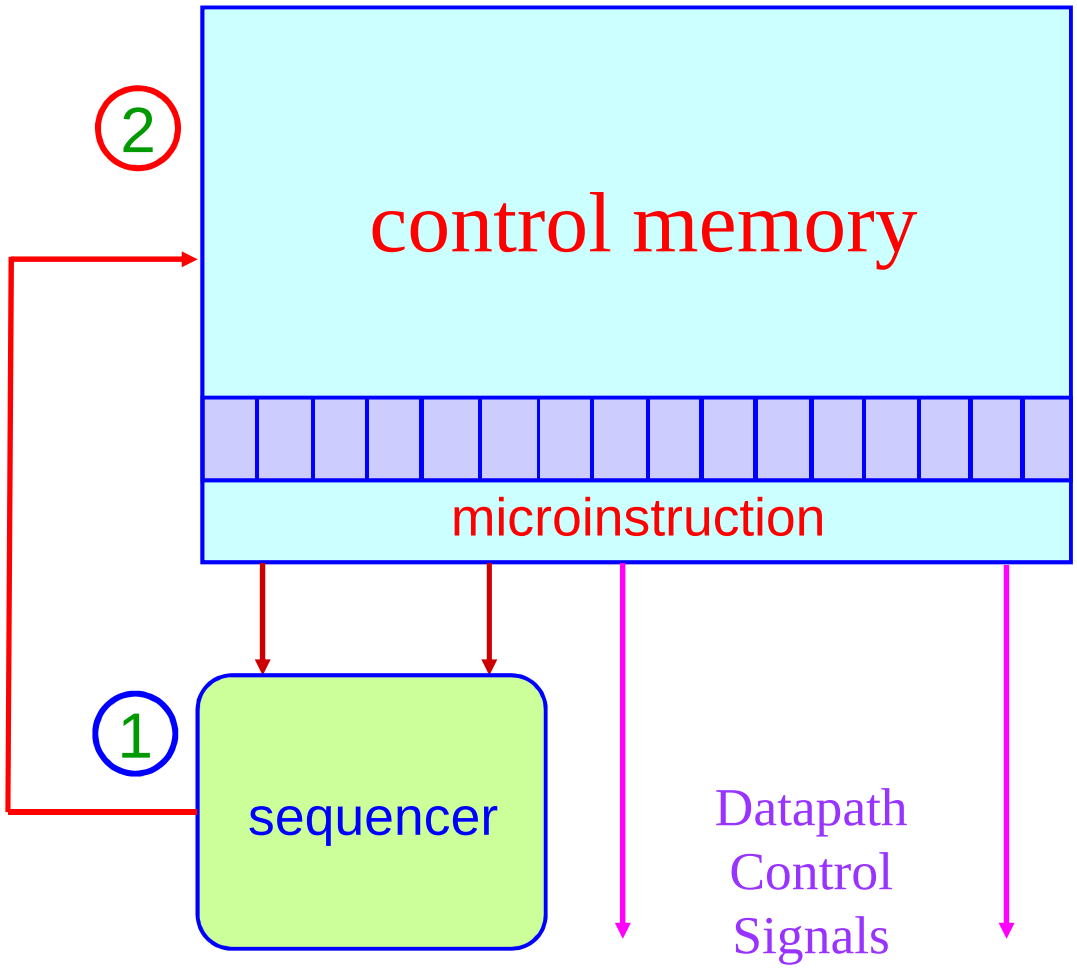
1. Micro-programmed Control Unit

- *Control* may be designed using one of several initial *representations*
- Choice of *sequence control*, and how logic is represented, can be determined independently
- the *control* can be *implemented* with one of *several methods* using a structured logic technique
- *Control reduces to programming "micro-sequencer"*

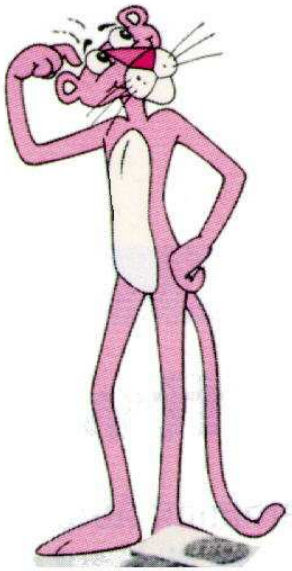
  ⇨ *microprogramming*

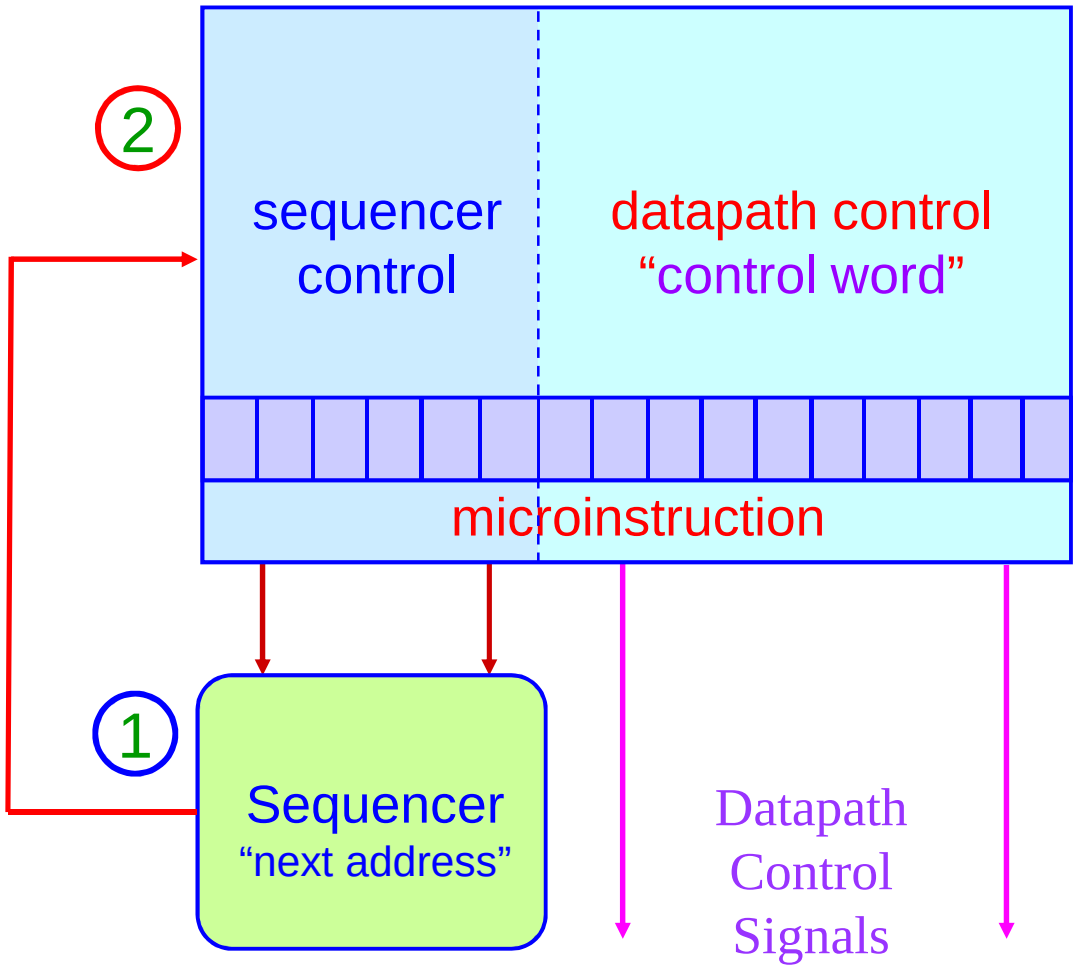
1. Micro-programmed Control Unit



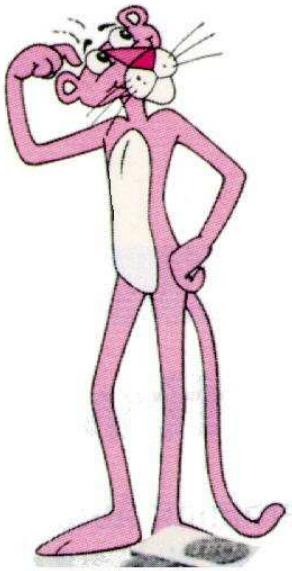


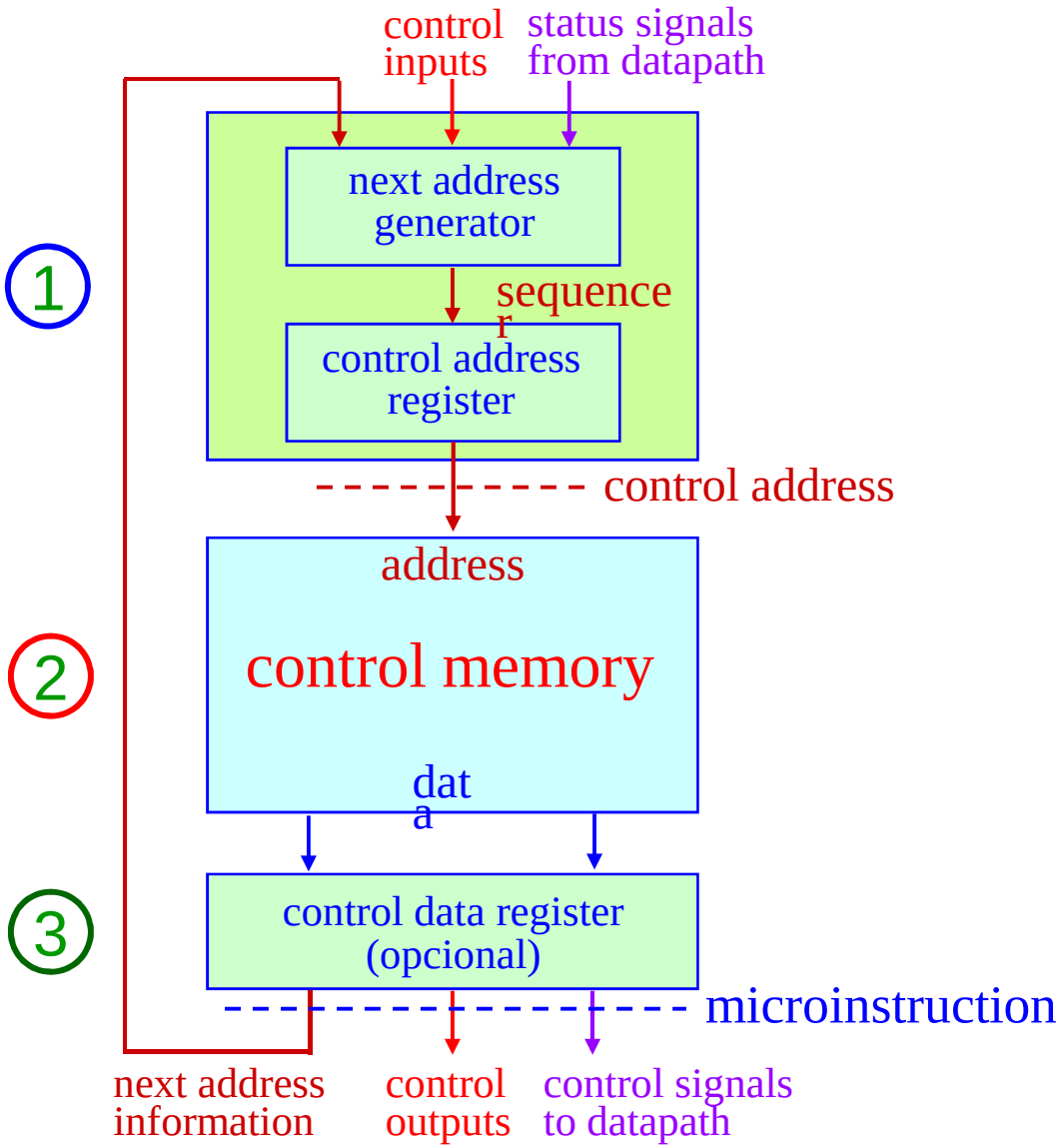
1. Micro-programmed Control Unit





1. Micro-programmed Control Unit





1. Micro-programmed Control Unit

- *Micro programmed control units* are *implemented* as a *microprogram* which is *stored* in a *control store memory*
- *Microprogram* is a *program* which consists of a *microcode* that *controls* the different parts of a processor.
- *The memory* in which the microprogram *resides* is called a *control store memory*
- *Words of the microprogram* are usually *accessed* or *selected* by a *microsequencer* or *sequencer,* which generates *the addresses* for the *memory*

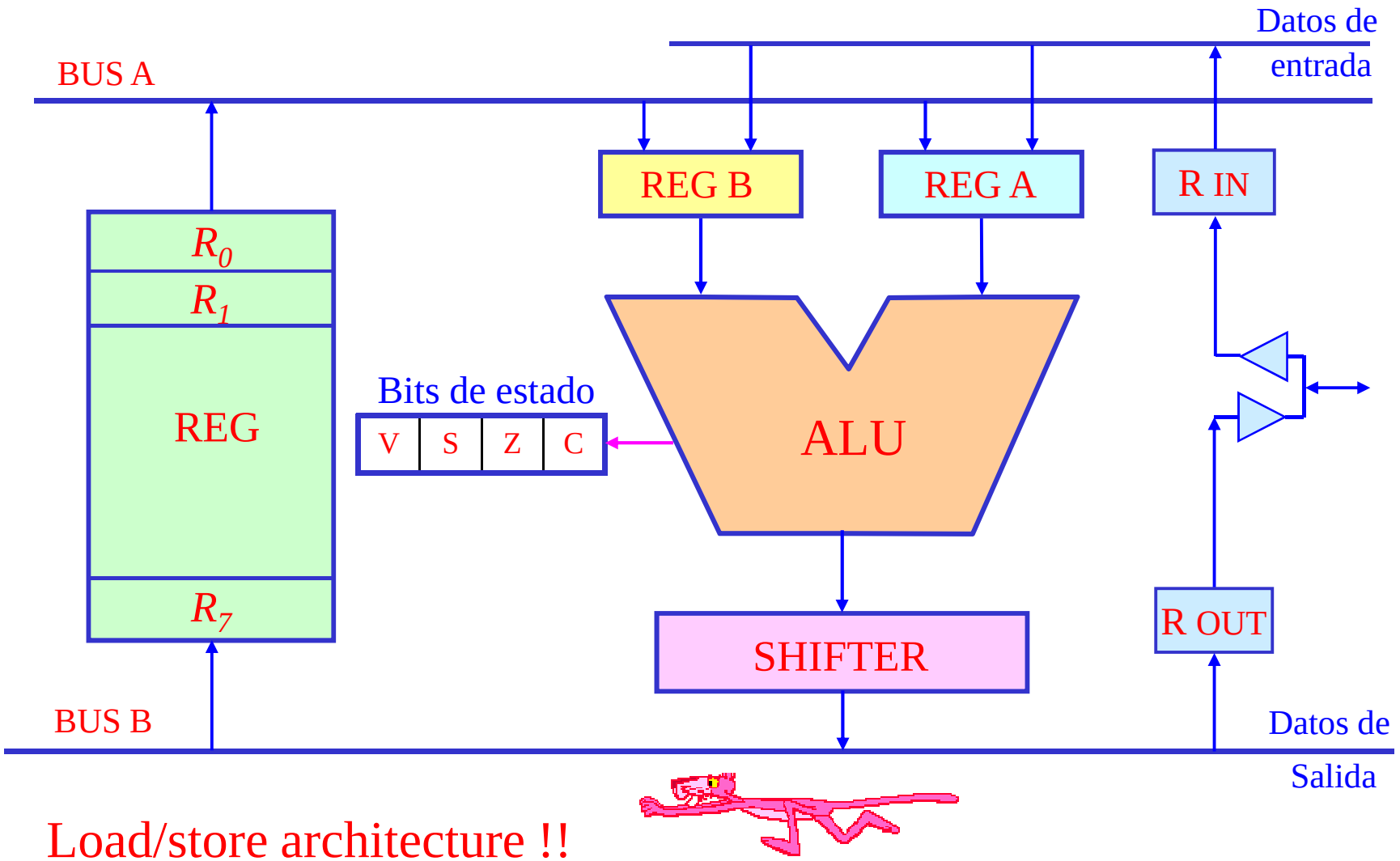
1. Micro-programmed Control Unit

- *The bits from words of the microprogram* directly *control* the different parts of the processor or device
- *Address are generated* by some *combination* of a *counter*, a *field from a microinstruction*, and some *subset of the instruction register*
- *Counter* is used for the *typical case*, which generates the address of the next microinstruction
- The *simplest sequencer* is just a *register* loaded from a *few bits* of the *control store memory*

2. Micro-programmed Control Unit Design

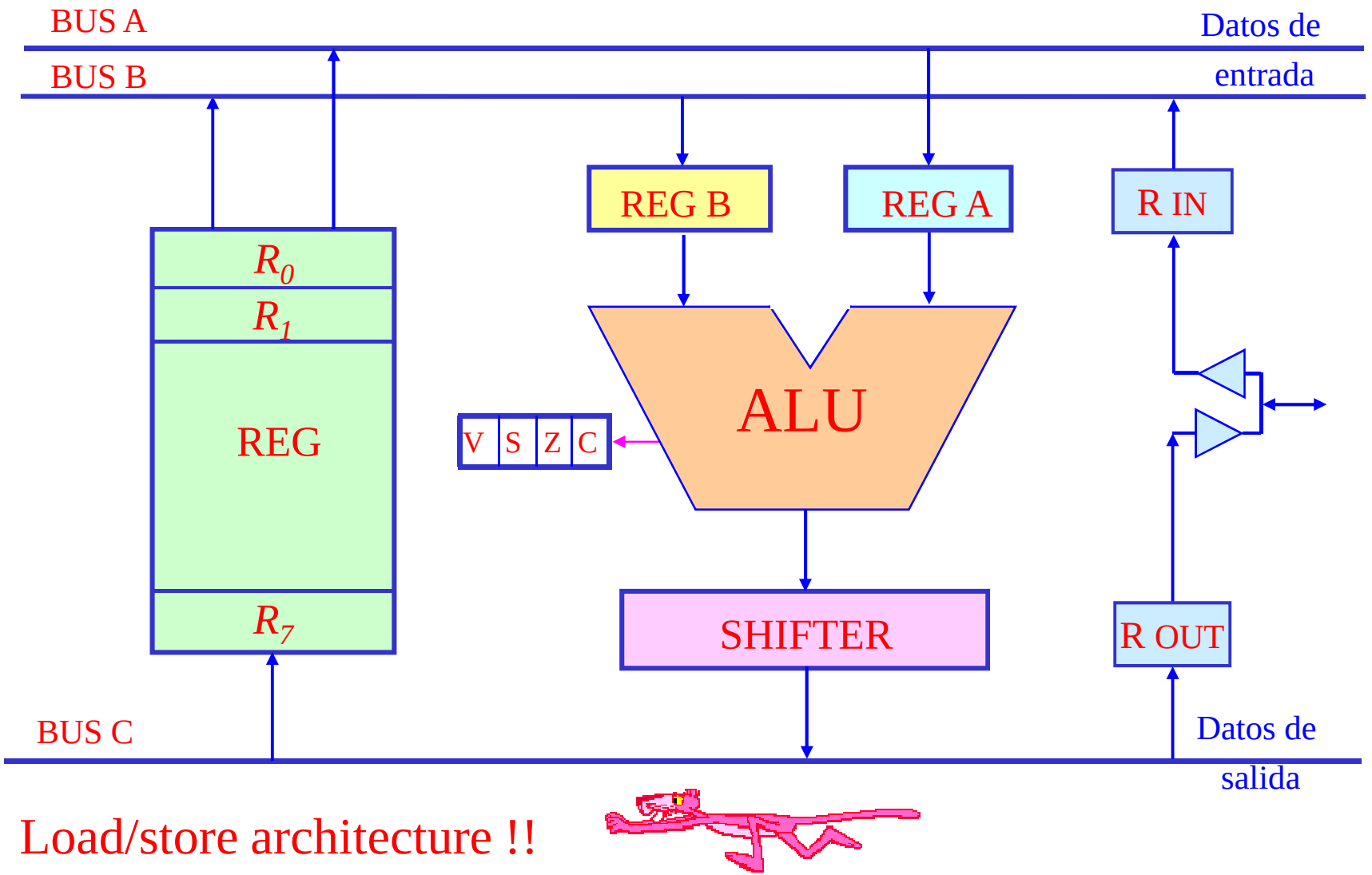
- *Data path1: simple processor*
  - Ocho *registros* R_0–R_7 de 16 bits
  - Un *shifter-barrel shifter* de 16 bits
  - Una *ALU* de 16 bits con cuatro *bits de estado*: *C, Z, S y V*
  - La *micro-instrucción* se selecciona con una *palabra de control* de *16 bits*
  - La *palabra de control* se divide en *cinco campos*: *A, B, D, F y H*

2. Data Path: Design 1



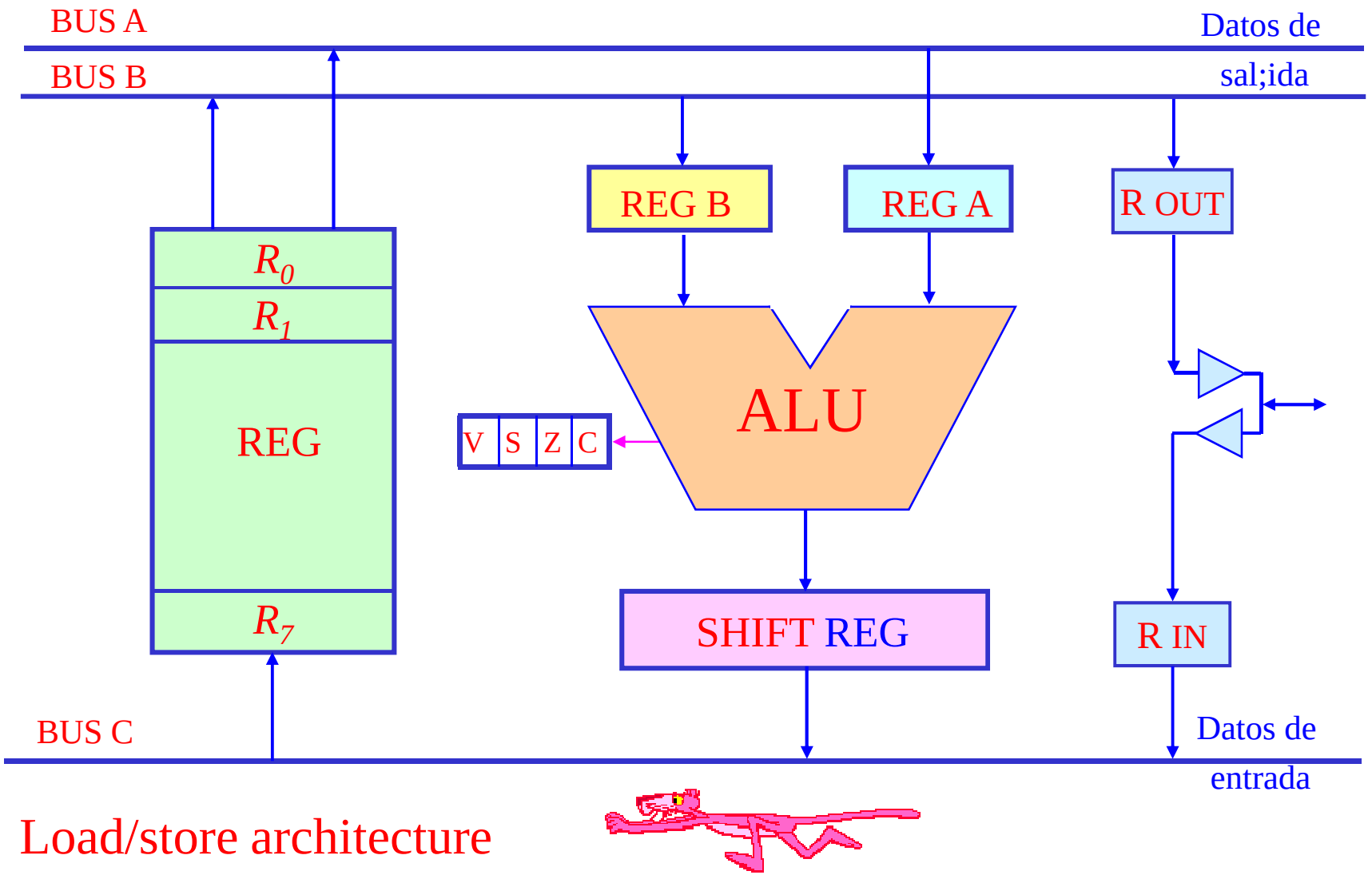
Load/store architecture !!

2. Data Path: Design 2



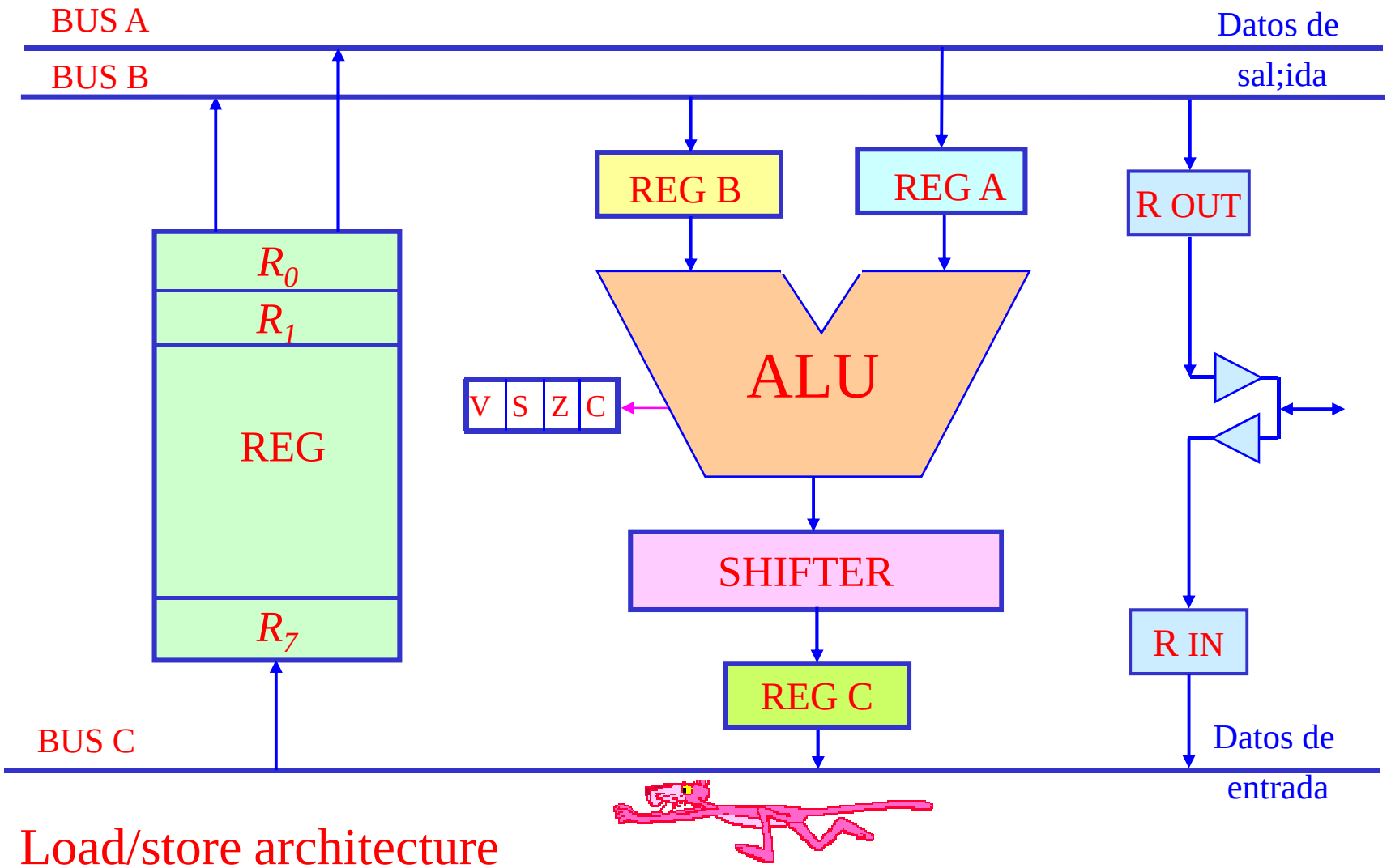
Load/store architecture !!

2. Data Path: Design 3



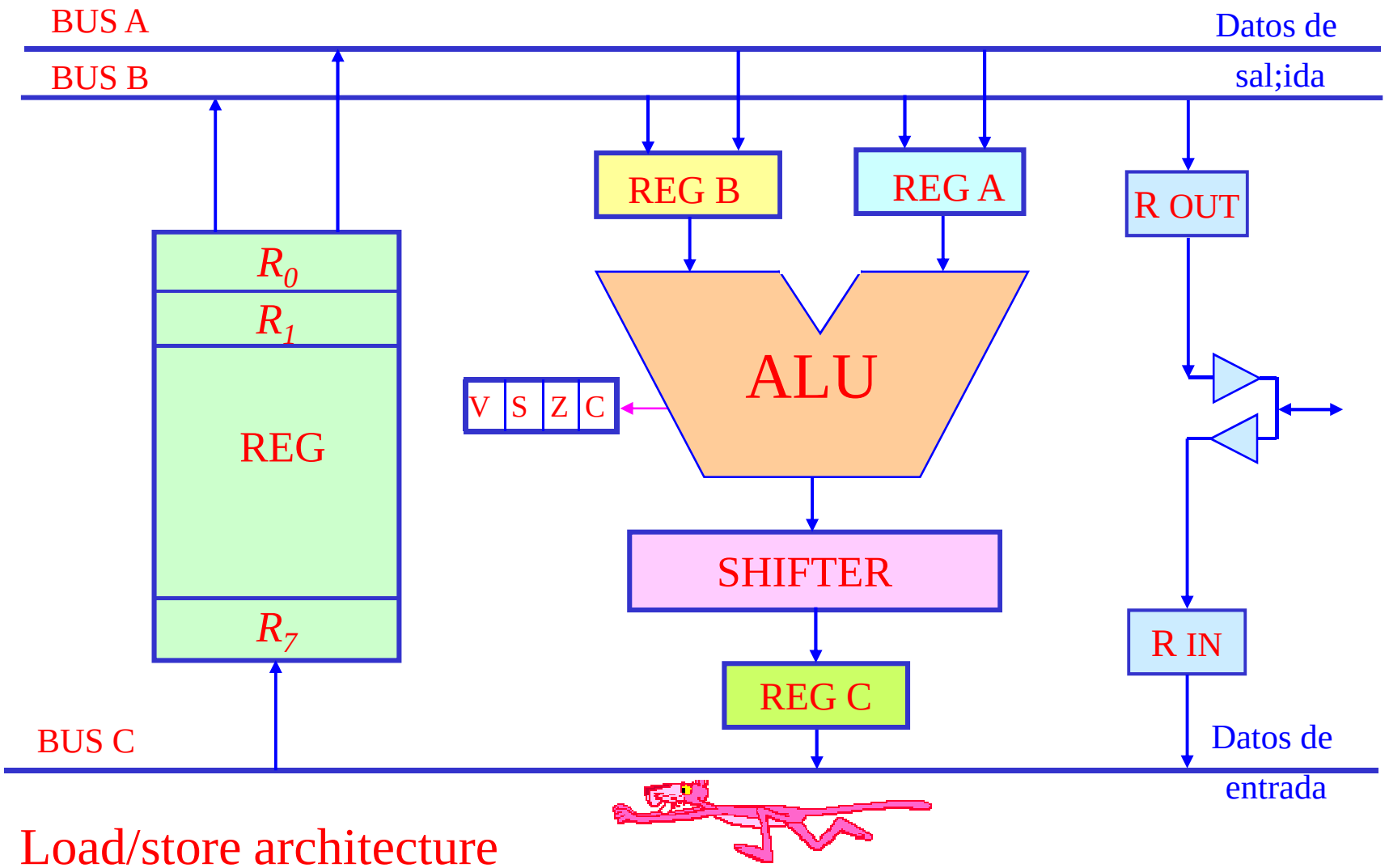
Load/store architecture

2. Data Path: Design 4



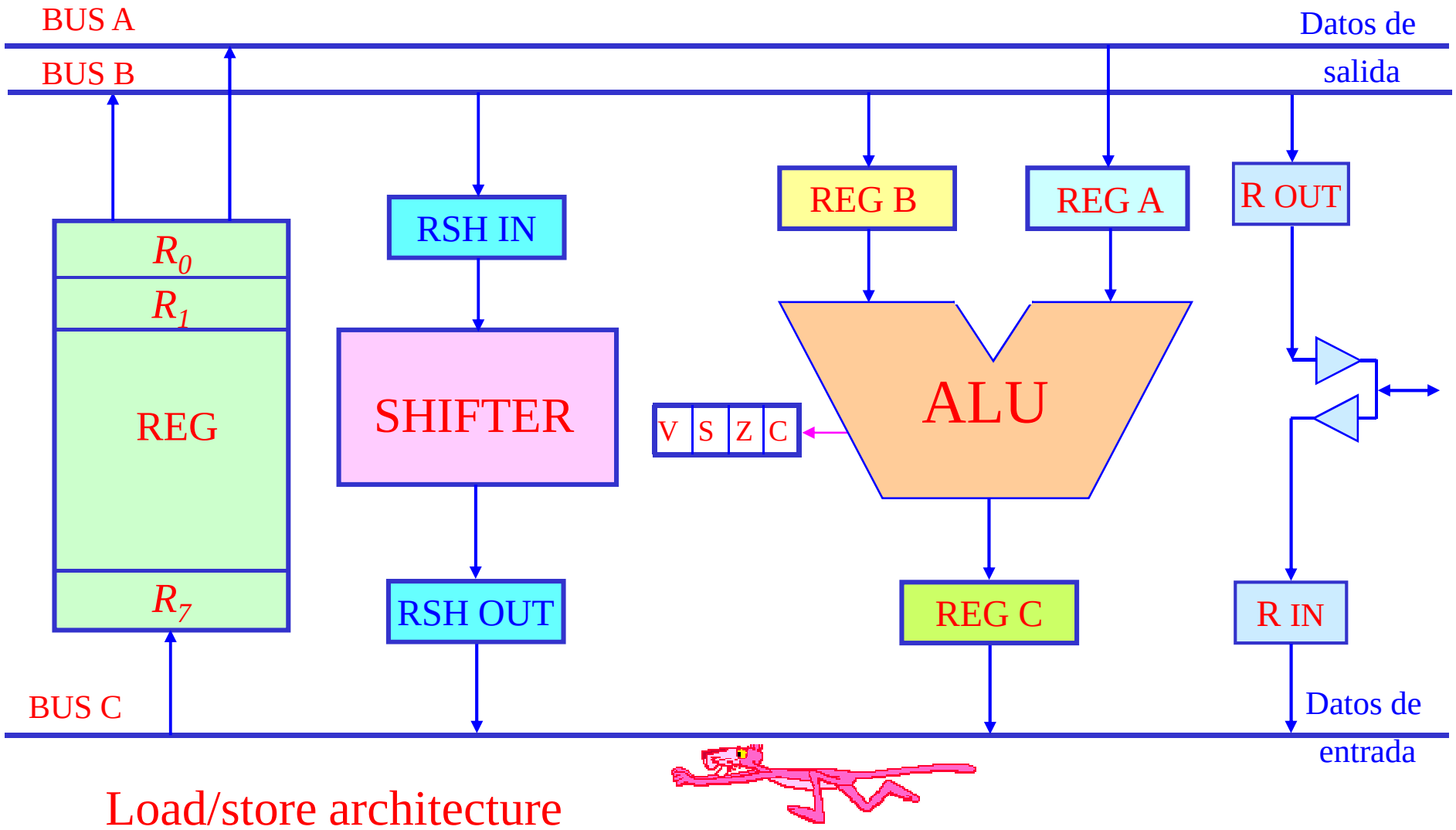
Load/store architecture

2. Data Path: Design 4-1



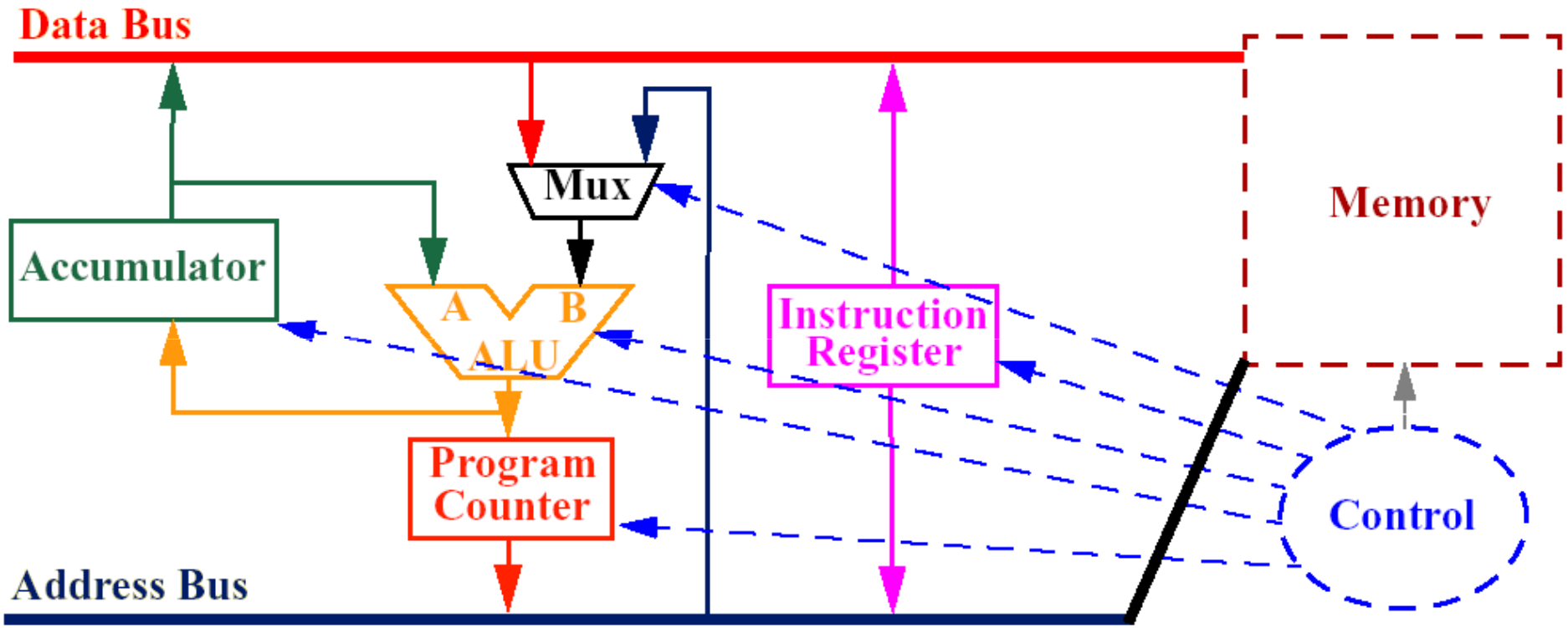
Load/store architecture

2. Data Path: Design 5



Load/store architecture

2. Data Path: Design 6

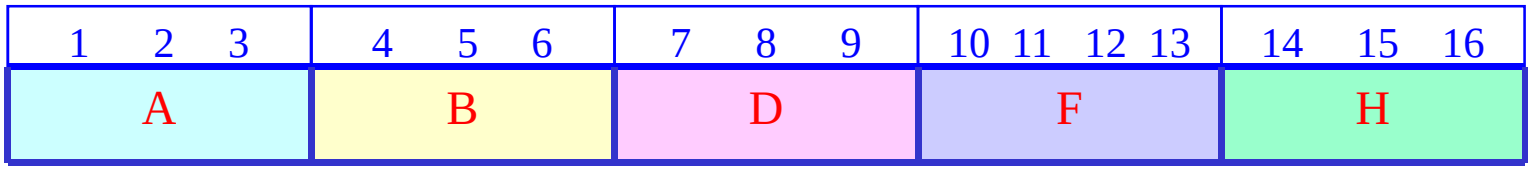


2. Micro-programmed Control Unit Design

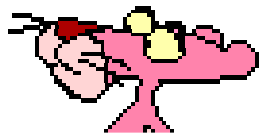
- *Control unit:*
  - Unidad *micro-programada*
    - una *memoria ROM* de *64* palabras de *26* bits
    - un *registro CAR*
    - dos *multiplexores* MUX1 y MUX2
  - La microinstrucción tiene *26 bits*
    - *16 bits* de la palabra de control
    - *10 bits* para seleccionar la siguiente dirección

2. Micro-programmed Control Unit Design

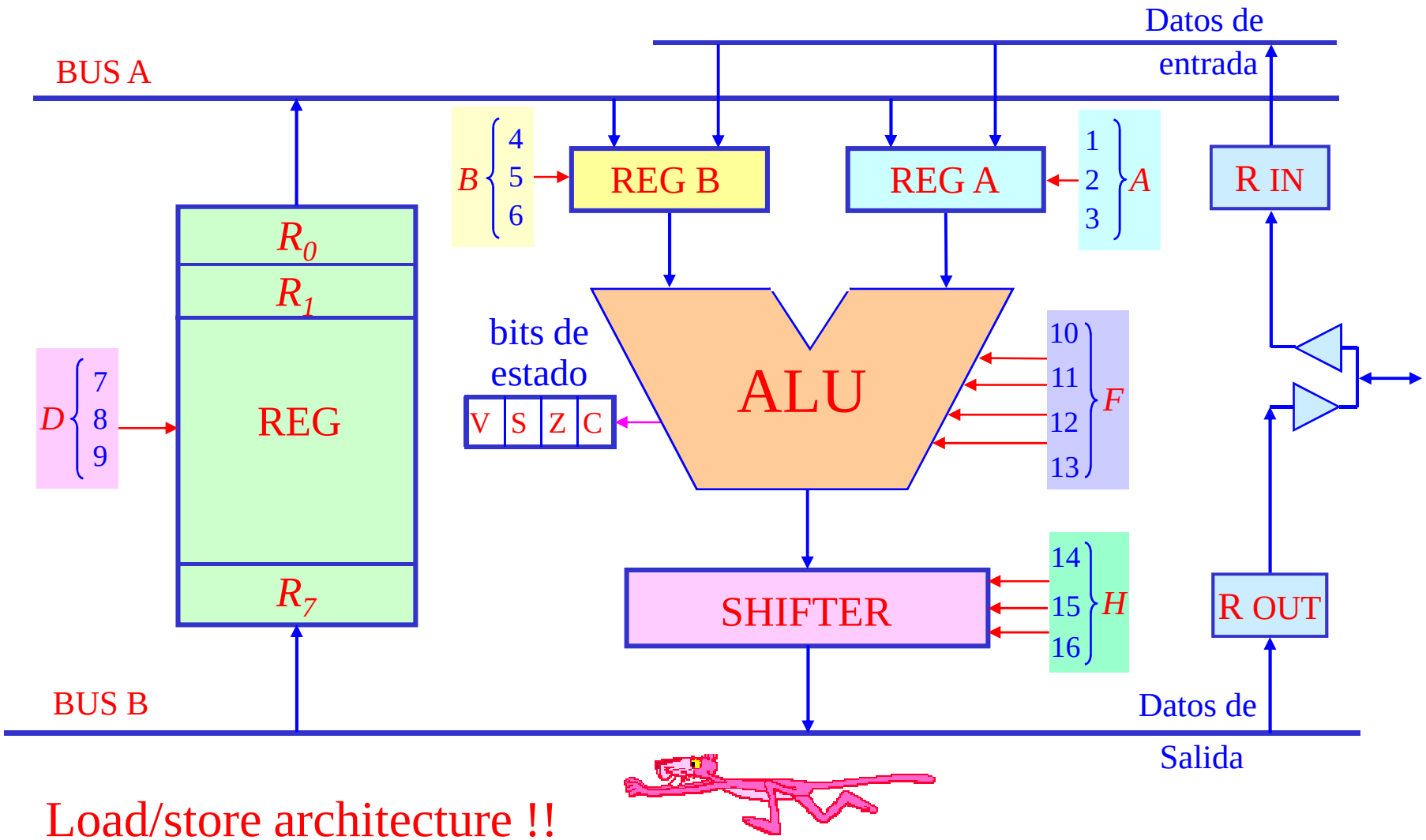
❑ *Palabra de control*



- Campo A (3 bits) : RS1, Banco de registros, Registro A
- Campo B (3 bits) : RS2, Banco de registros, Registro B
- Campo D (3 bits) : RD1, Banco de registros
- Campo F (4 bits) : ALU
- Campo H (3 bits) : Shifter

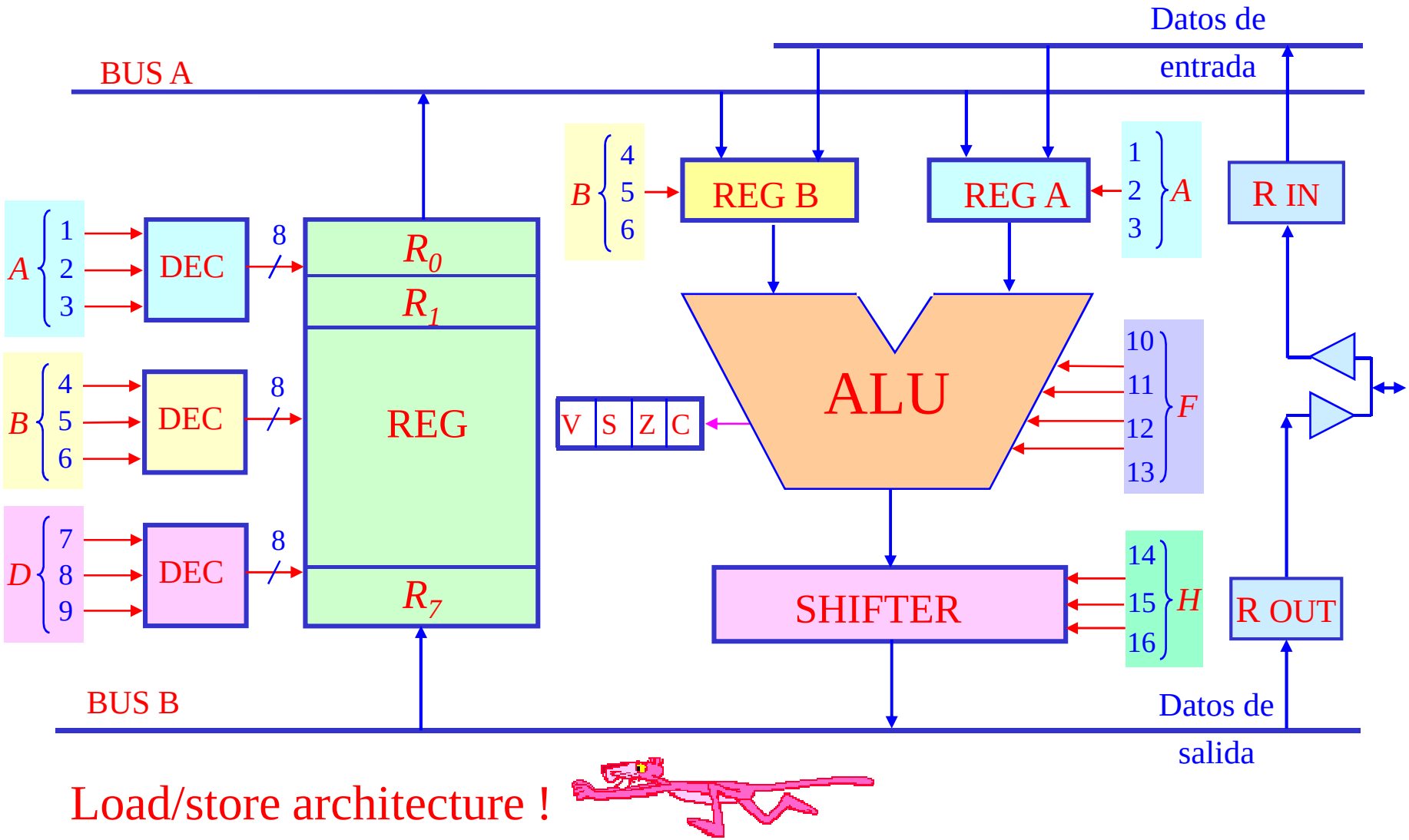


2.1 Data–path1 con señales de control



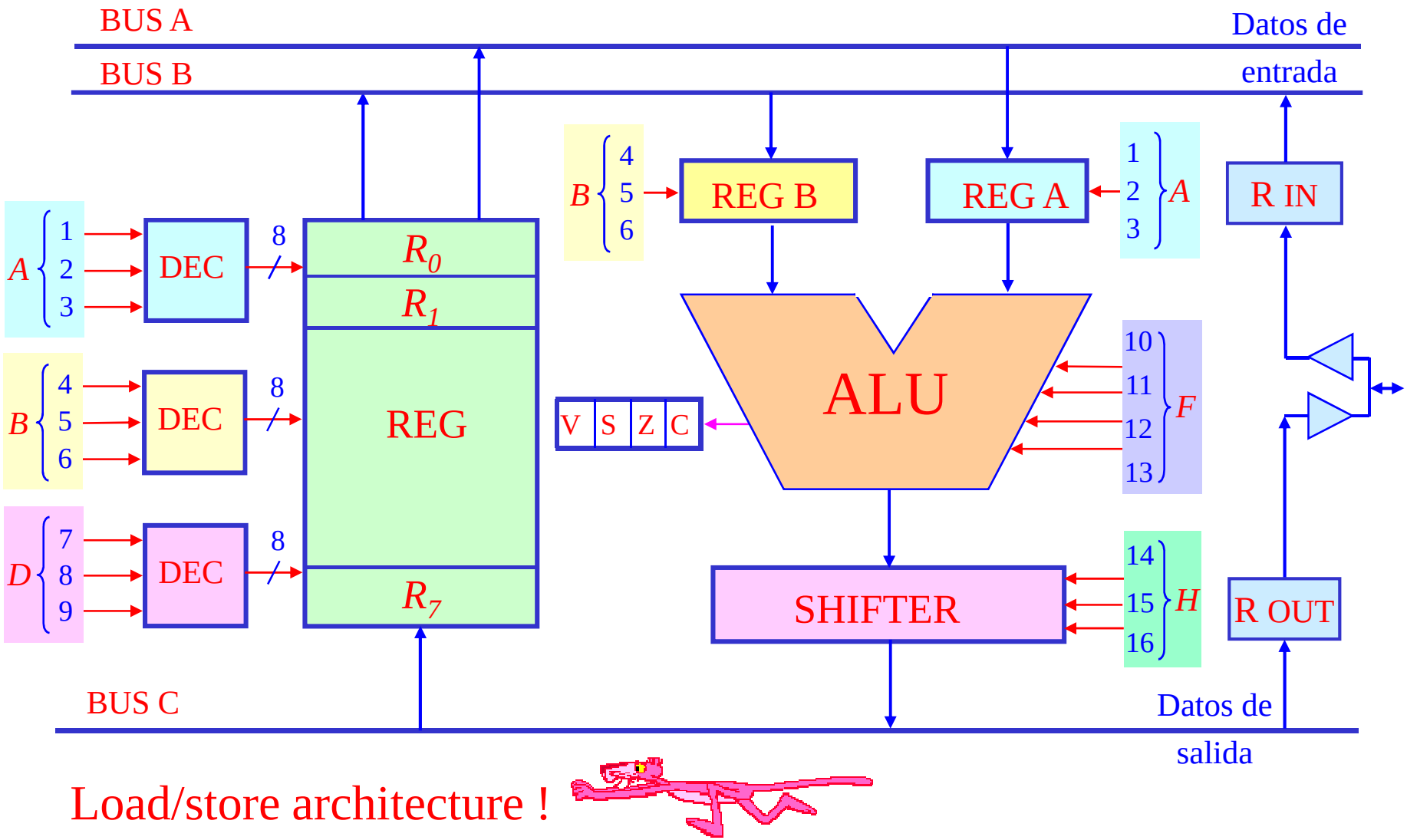
Load/store architecture !!

2.1 Datapath1 con señales de control



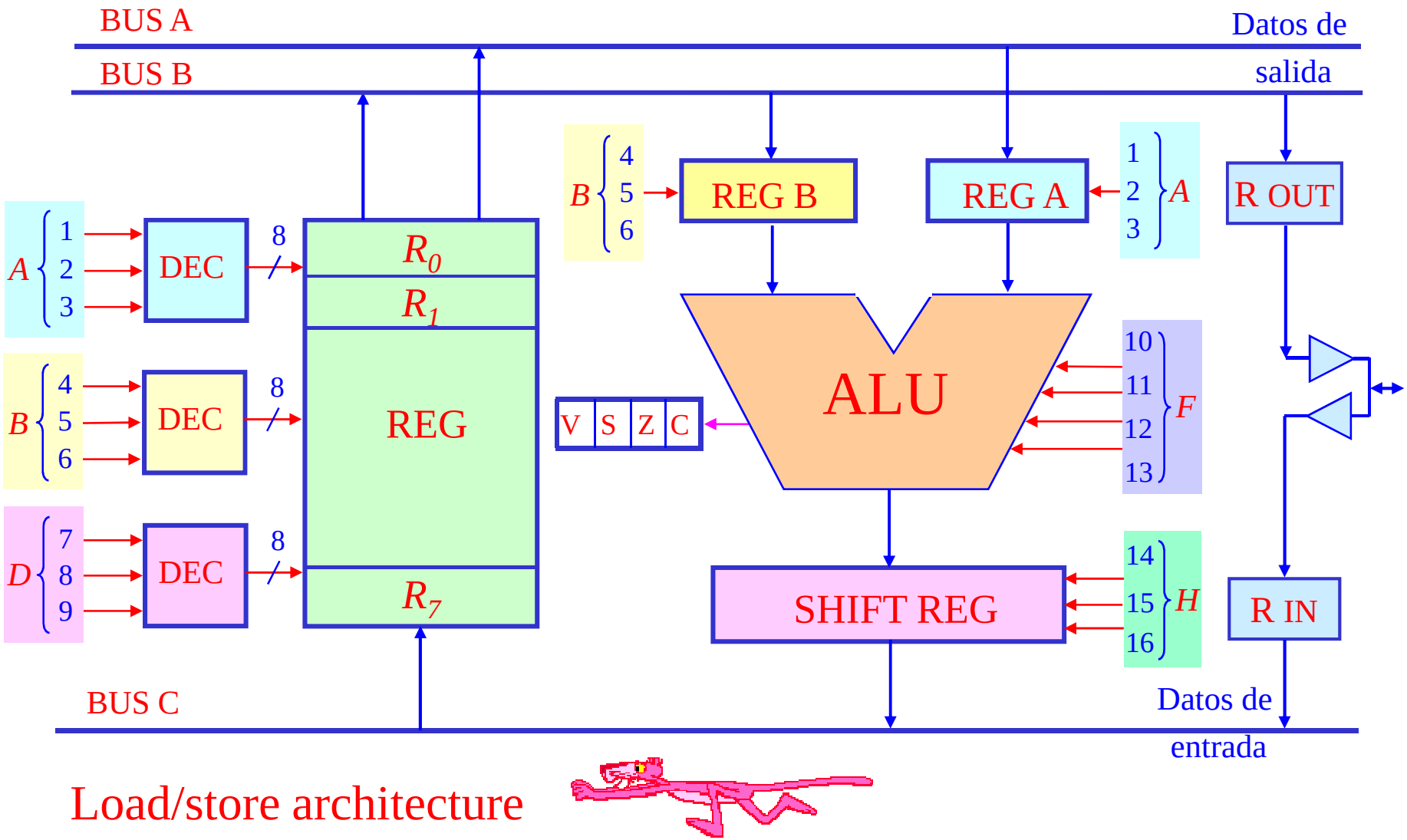
Load/store architecture !

2.1 Datapath2 con señales de control



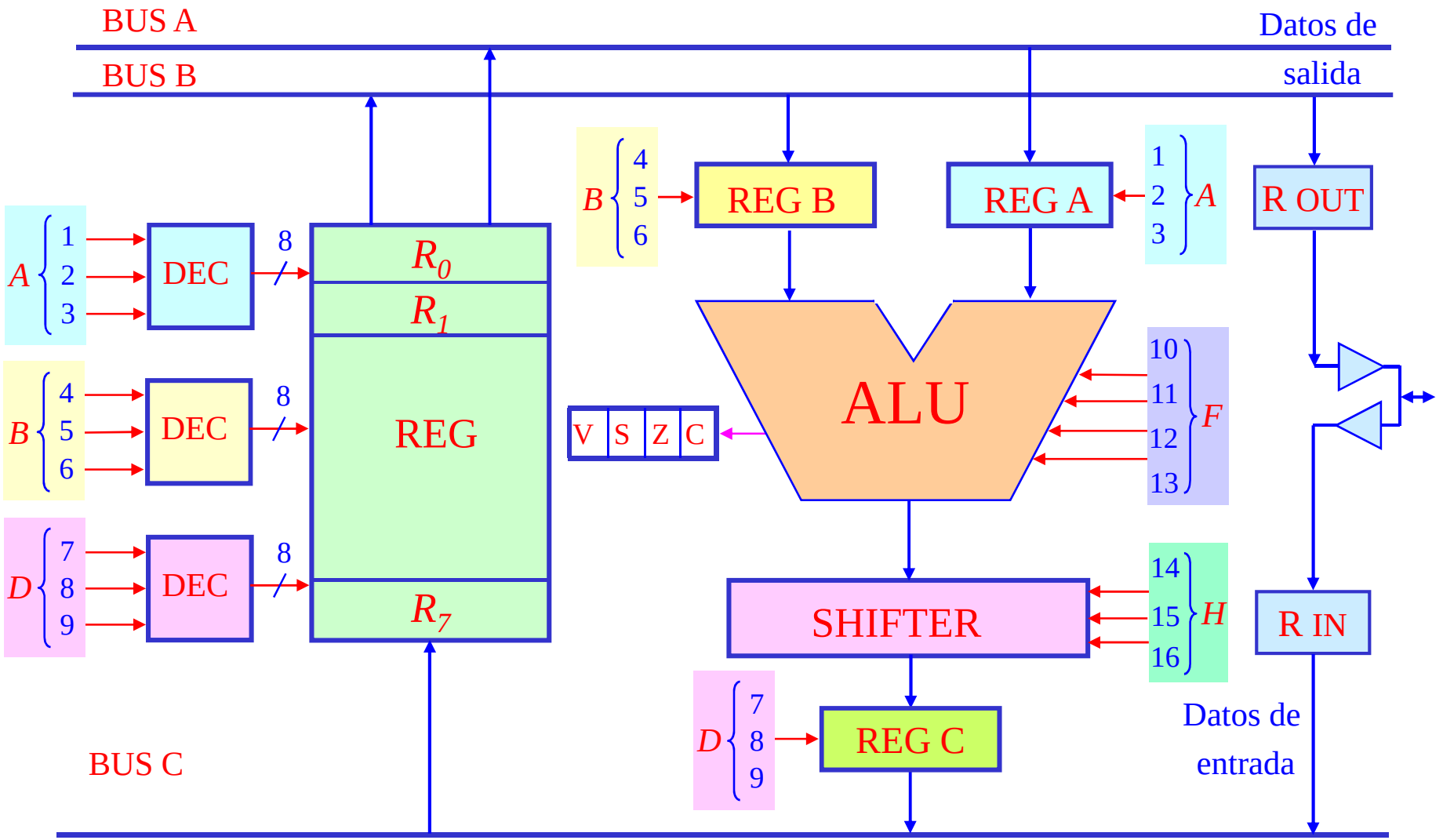
Load/store architecture !

2.1 Datapath3 con señales de control



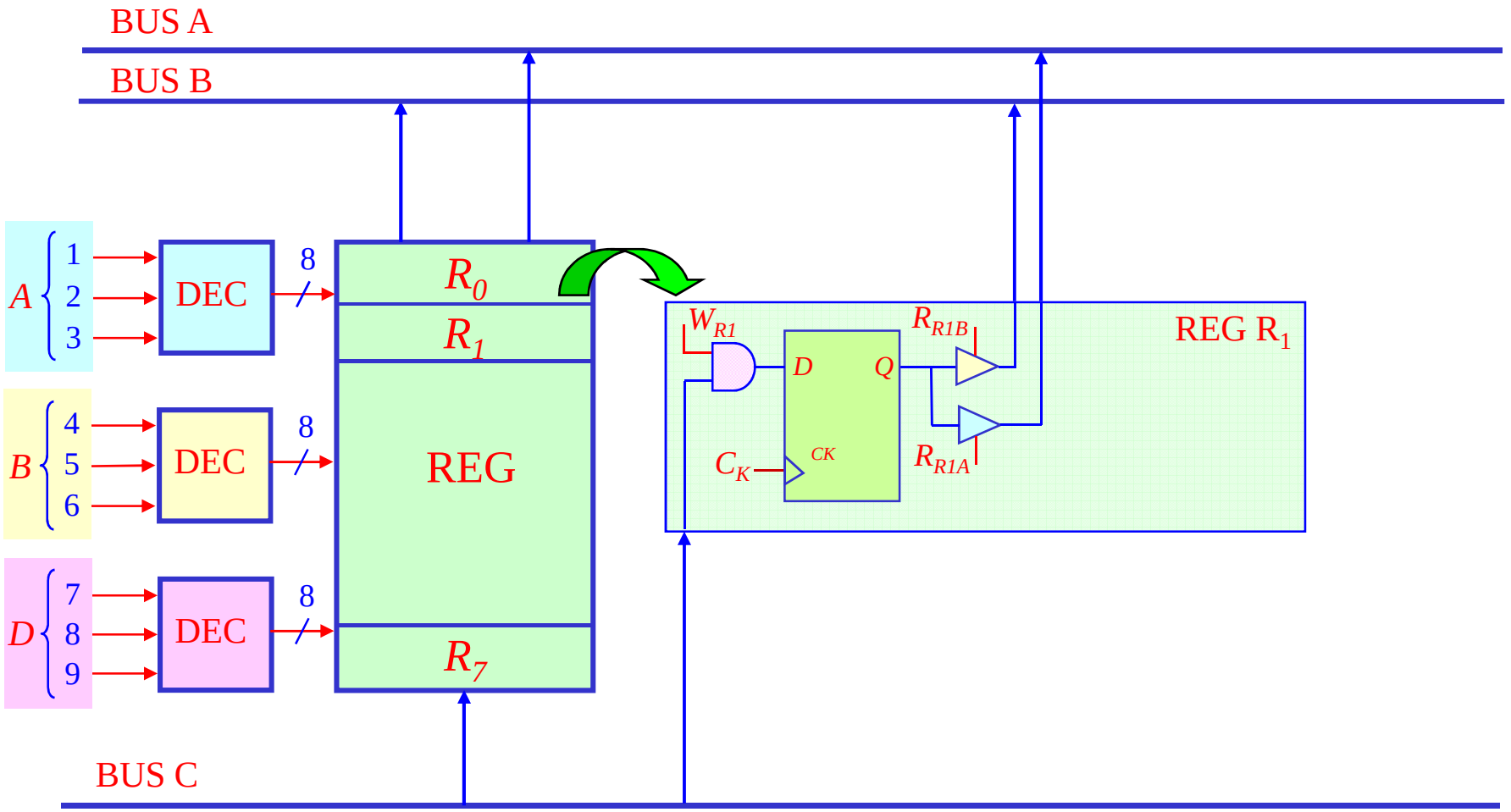
Load/store architecture

2.1 Datapath4 con señales de control



Load/store architecture

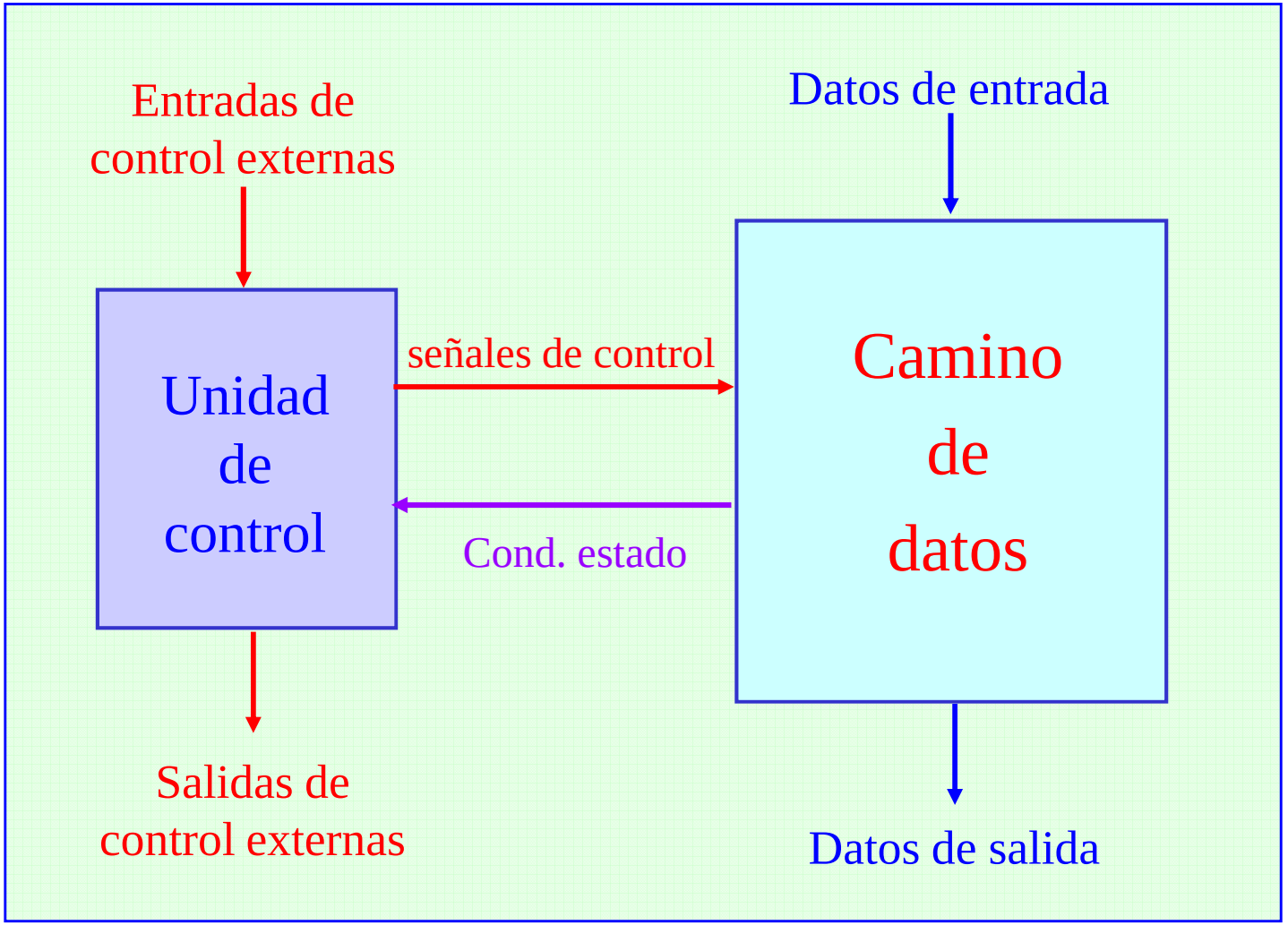
2.1 Datapath: registros



2.2 Diseño de la Unidad de Control

- El diseño de un *sistema digital complejo se divide*:
  - Diseño de los circuitos digitales para efectuar las *operaciones de procesamiento de datos*: *Data Path Design*
  - Diseño del circuito de control que *supervisa las operaciones y determina la secuencia de ejecución*: *Control Unit Design*
- La unidad de control es básicamente un *circuito secuencial con estados internos*, que depende de las *condiciones de estado y entradas externas*
- En un sistema digital, la *sincronización* de los *registros* es controlada por una *señal de reloj*

2.2 Diseño de la Unidad de Control



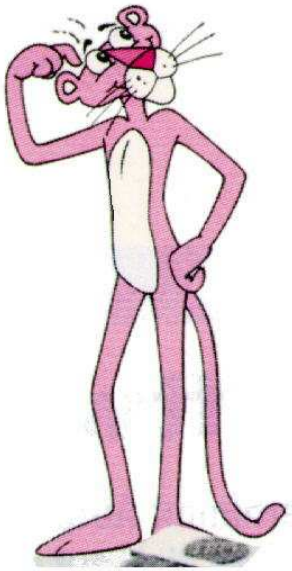
2.2 Diseño de la Unidad de Control

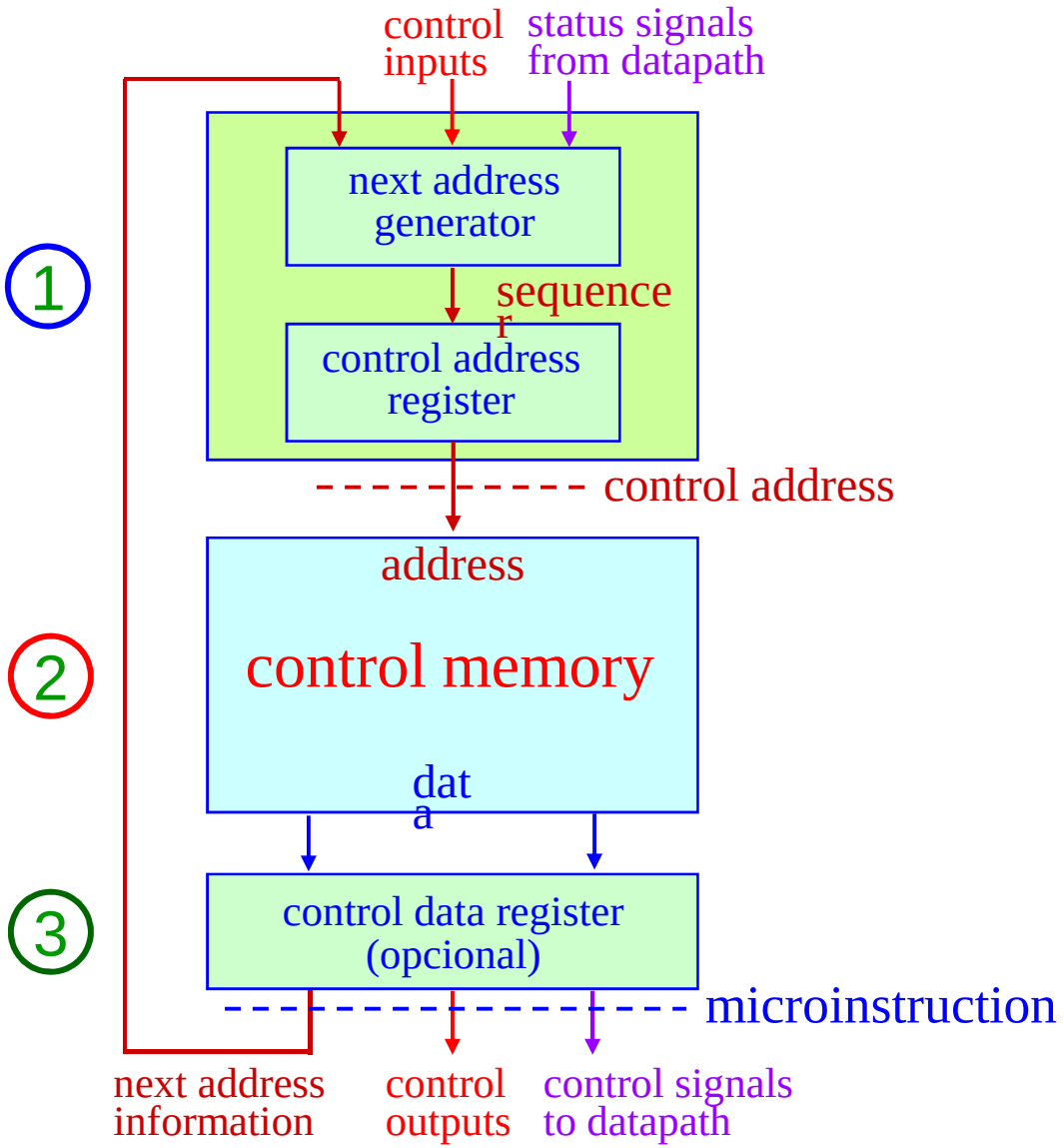
- *Arquitecturas para la unidad de control*
  - *Unidad de control dedicada (hardwired control)* basada en usar *gates* y *flip-flops*
  - *Unidad de control micro-programada* basada en usar una memoria *ROM-EEPROM* o *RAM*
  - La unidad de *control dedicada* usa una maquina de estados finito (FSM) compleja: *controlador*
  - La unidad de *control micro-programada* usa una maquina de estados finito simple: *secuenciador*

2.2 Diseño de la Unidad de Control

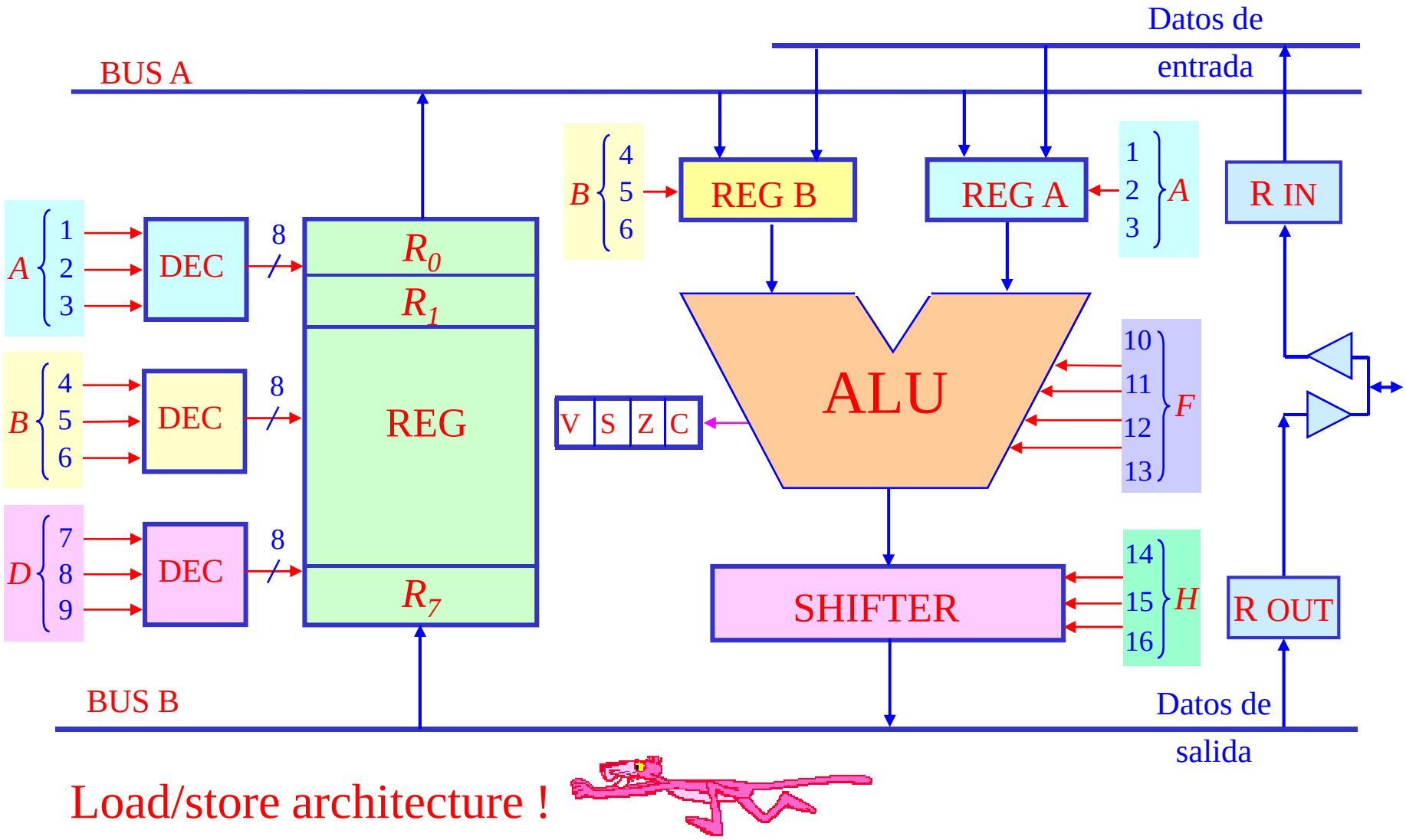
- *Unidad de control micro-programada*
  - La unidad de control micro-programada almacena la información de control (variables de control) en la memoria (memoria de control)
  - Cada palabra de la memoria de control contiene una *microinstrucción*
  - La microinstrucción especifica *una o más micro-operaciones* para el sistema digital
  - Permite *micro-operación dinámica*, es decir permite cargar un *micro-programa* desde el exterior (teclado/consola o un disco magnético)

2.2 Micro-programmed Control Unit





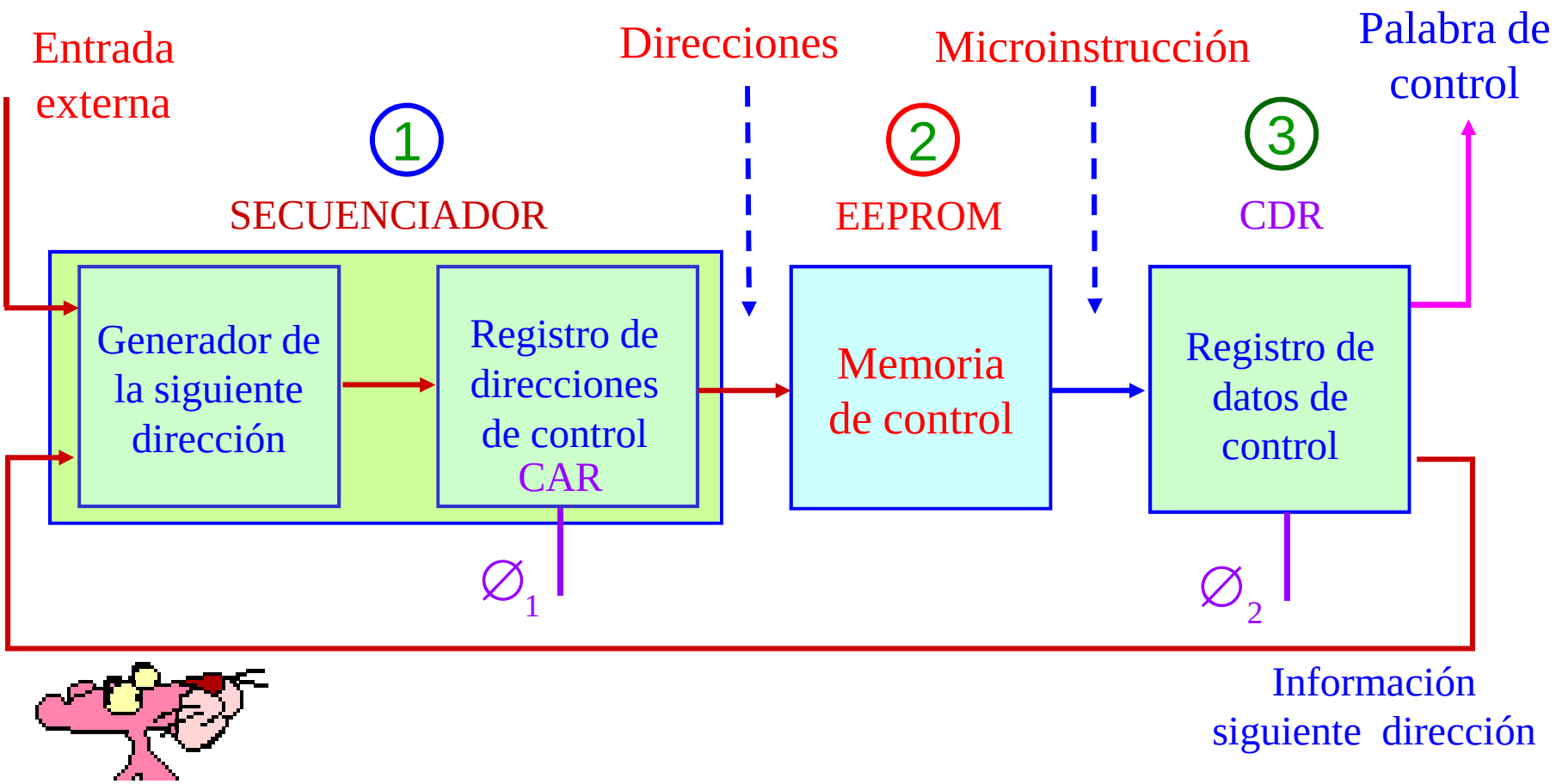
2.2 Datapath1 con señales de control



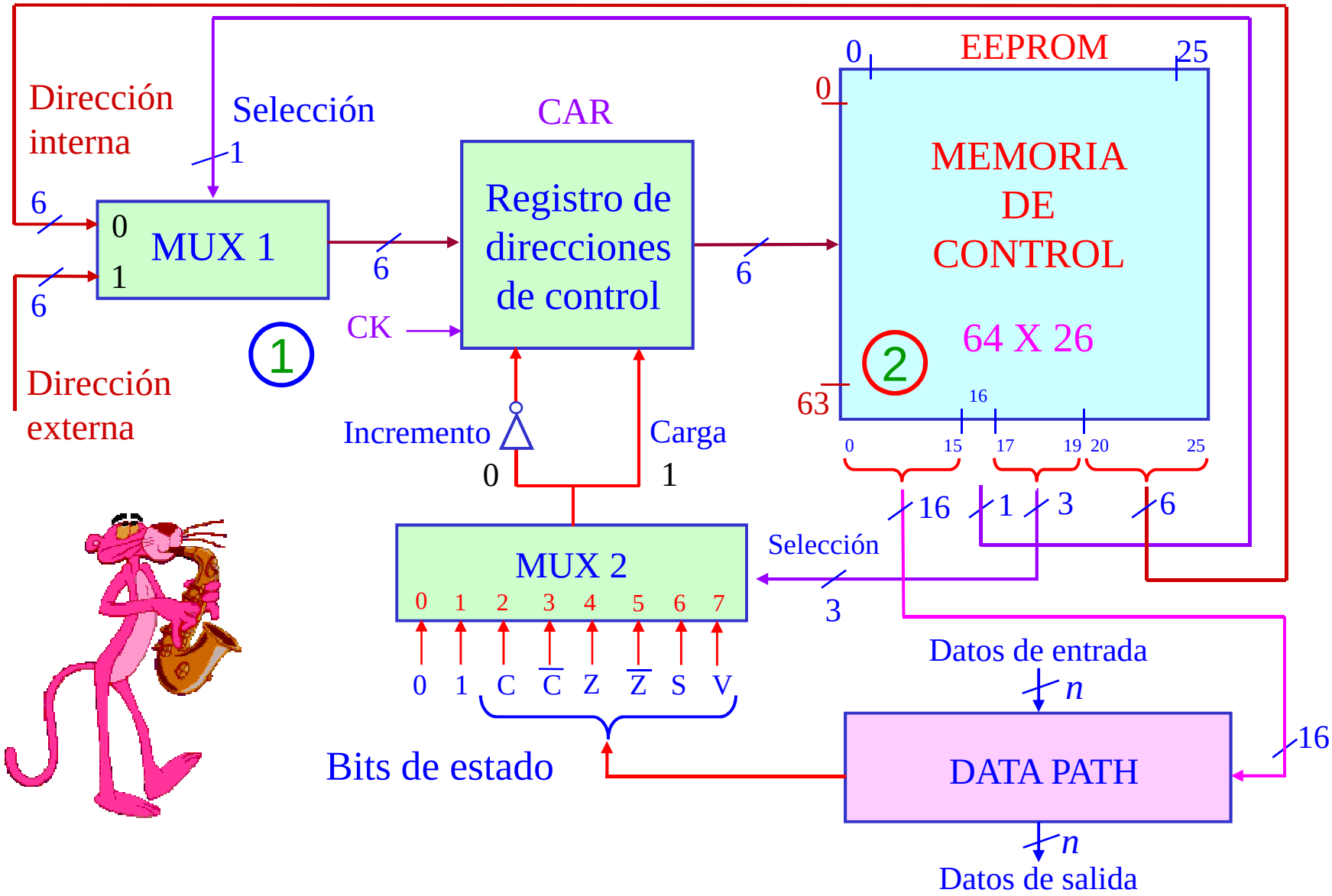
Load/store architecture !

2.2 Diseño de la Unidad de Control

- *Diagrama de bloques de unidad de control microprogramada*



2.2 Diseño de la Unidad de Control



2.3 Operación de la Unidad de Control

- En cada *flanco/pulso de reloj*, el *registro CAR* recibe una *nueva dirección*
- La *microinstrucción de 26 bits* se lee desde la *memoria de control*
- La *palabra de control de la microinstrucción* especifica la *micro-operación* en el procesador
- Los *multiplexores MUX1 y MUX2* determinan la siguiente *operación del registro CAR*
- La *próxima transición del reloj* transfiere el *resultado de la micro-operación* al *registro destino* del procesador y *actualiza los bits de estado* y coloca una *nueva dirección en el registro CAR*

2.3 Operación de la Unidad de Control

- La *microinstrucción* contiene la *palabra de control* que especifica *una o más micro-operaciones*
- La *dirección* de la *siguiente microinstrucción* puede ser la *siguiente dirección* u otra *nueva dirección*
- La *microinstrucción* tiene *bits* para *generar la dirección de la siguiente microinstrucción*
- La *próxima dirección* puede ser función de *las entradas externas*
- La *microinstrucción* especifica *simultáneamente la palabra de control y la generación de la próxima microinstrucción*

2.3 Operación de la Unidad de Control

- La *dirección* de la *próxima microinstrucción* se puede obtener:
  - *Incremento* del *registro CAR en uno*
  - *Transferencia* de una *dirección externa*
  - Cargar en el *registro CAR* una *dirección de la memoria de control*
  - *Cargar* una *dirección inicial*: operaciones de control

2.4 Palabra de Control

- Las *variables de selección-control* del data-path *controlan las micro-operaciones*, es decir, controlan la ALU, el shifter y los registros
- La combinación de las variables de selección-control especifican la palabra de control
- *La palabra de control se divide en campos* y esta palabra cuando se aplica a *las entradas de selección-control permite realizar una micro-operación en particular*
- La *palabra de control de una micro-operación* se obtiene a partir de *las variables de selección-control*

2.4.1 Formato de la Palabra de Control

	Función de los campos de selección					
Código Binario	A	B	D	F : Cin = 0	F: Cin = 1	H
000	Inp	Inp	None	F = A	F = A+1	No desplaz.
001	R1	R1	R1	F = A+B	F = A+B+1	SHL
010	R2	R2	R2	F = A+$\overline{B}$	F = A-B	SHR
011	R3	R3	R3	F = A - 1	F = A	Bus = 0
100	R4	R4	R4	F = A $\wedge$ B	--	--
101	R5	R5	R5	F = A $\vee$ B	--	ROL
110	R6	R6	R6	F = A $\oplus$ B	--	ROR
111	R7	R7	R7	F = $\overline{A}$	--	--

2.4.2 Codificación de la Palabra de Control

- *Micro-operación de resta*

RTL	R1 ←	R2 − R3			
Campo:	**A**	**B**	**D**	**F**	**H**
Símbolo:	**R2**	**R3**	**R1**	**F = A-B**	No desplazamiento
Palabra de contr.	**010**	**011**	**001**	**0101**	**000**

- *Micro-operación de suma y desplazamiento*

RTL	R4 ←	Shr (R5 + R6)			
Campo:	**A**	**B**	**D**	**F**	**H**
Símbolo:	**R5**	**R6**	**R4**	**F = A+B**	**Shr**
Palabra de contr.	**101**	**110**	**100**	**0010**	**010**

2.4.2 Codificación de la Palabra de Control

	Designación simbólica					Palabra de control				
Micro-operación	A	B	D	F	H	A	B	D	F	H
R1 ← R2-R3	R2	R3	R1	F= A-B	No desplaza.	010	011	001	0101	000
R4 ← shr(R5+R6)	R5	R6	R4	F=A+B	SHR	101	110	100	0010	010
R7 ← R7+1	R7	--	R7	F= A+1	No desplaza.	111	000	111	0001	000
R1 ← R2	R2	--	R1	F= A	No desplaza.	010	000	001	0000	000
Salida ← R3	R3	--	None	F= A	No desplaza.	011	000	000	0000	000
R4 ← Rol R4	R4	--	R4	F= A	ROL	100	000	100	0000	101
R5 ← 0	--	--	R5	--	Bus = 0	000	000	101	0000	011

2.5 Formato de la Microinstrucción

Bits	Campo	Símbolo	Función
0-2	A	Inp, $R_1 – R_7$	Entrada derecha a la ALU
3-5	B	Inp, $R_1 – R_7$	Entrada izquierda a la ALU
6-8	D	None, $R_1 – R_7$	Registro destino
9-12	F	-	Operación de la ALU
13-15	H	-	Operación del Shifter
16	MUX 1	INT, EXT	Selección del MUX:INT=0, EXT=1
17-19	MUX 2	-	Selección del MUX 2
20-25	ADRS	Número de dirección	Campo de dirección

2.5 Formato de la Microinstrucción

INP : Datos de entrada al procesador

NONE : No selecciona ninguno de los registros; R_0

INT : Dirección interna de la microinstrucción

EXT : Dirección externa.

— : Observar tablas

2.5.1 Codificación de la Microinstrucción

- *Microprograma: micro-operaciones en RTL*

Dirección	Micro-operaciones y condiciones de bifurcación
36	$R_1 \leftarrow R_1 \wedge R_2$, CAR ← CAR + 1
40	$R_3 \leftarrow R_3 - 1$, CAR ← 43
45	No existe operación para el procesador, ROUT ← RIN, CAR ← CAR + 1
52	$R_4 \leftarrow 0$, si S = 1 entonces CAR ← 37, si no CAR ← CAR + 1
56	$R_5 \leftarrow$ Shl R_5 , si C = 0 entonces CAR ← 62, si no CAR ← CAR + 1
59	$R_2 \leftarrow R_6 + R_7$, CAR ← Dirección externa

2.5.1 Codificación de la Microinstrucción

Dirección CAR	Palabra de control					Selección		Campo de dirección
	A	B	D	F	H	MUX1	MUX2	
36	R_1	R_2	R_1	AND	NSH	-	NEXT	-
	001	010	001	1000	000	0	000	000000
40	R_3	-	R_3	DEC	NSH	INT	LAD	43
	011	000	011	0110	000	0	001	101011
45	-	-	none	TSF	NSH	-	NEXT	-
	000	000	000	0000	000	0	000	000000
52	-	-	R_4	TSF	Zero	INT	LS	37
	000	000	100	0000	011	0	110	100101
56	R_5	-	R_5	TSF	SHL	INT	LNC	62
	101	000	101	0000	001	0	011	111110
59	R_6	R_7	R_2	ADD	NSH	EXT	LAD	-
	110	111	010	0010	000	1	001	000000

2.6.1 Operaciones de la ALU : Campo F

Código Binario	Símbolo	Bits de Estado				Función
		Z	S	C	V	
0000	TSF	N	N	N	N	Transferencia de A
0001	INC	Y	Y	N	N	Incremento de A en uno
0010	ADD	Y	Y	Y	Y	Sumar A + B
0101	SUB	Y	Y	Y	Y	Restar A – B
0110	DEC	Y	Y	N	N	Decremento de A en uno
0111	TRC	Y	Y	O	N	Transferencia de A, reiniciar el carry
1000	AND	Y	Y	N	N	A AND B
1010	OR	Y	Y	N	N	A OR B
1100	XOR	Y	Y	N	N	A OR exclusiva B
1110	COM	Y	Y	N	N	Complemento de A

N : Bit de estado no afectado
Y : Bit de estado afectado
O : Re-iniciar a cero

2.6.2 Operaciones del Shifter: Campo H

Código Binario	Símbolo	Función
000	NSH	No hay desplazamiento
001	SHL	Desplazamiento a la izquierda con In serie = 0
010	SHR	Desplazamiento a la derecha con In serie = 0
011	ZERO	Solo ceros en la salida
100	RLC	Rotación a la izquierda con carry
101	ROL	Rotación a la izquierda
110	ROR	Rotación a la derecha
111	RRC	Rotación a la derecha con carry

2.6.3 Entrada de Selección: MUX 2

Código Binario	Símbolo	Función
000	NEXT	Ir a la siguiente dirección incrementando CAR
001	LAD	Cargar la dirección en CAR/bifurcar incondicional
010	LC	Carga / Bifurcar si Carry = 1
011	LNC	Carga / Bifurcar si Carry = 0
100	LZ	Carga / Bifurcar si Zero = 1
101	LNZ	Carga / Bifurcar si Zero = 0
110	LS	Carga / Bifurcar si Signo = 1
111	LV	Carga / Bifurcar si Overflow = 1

2.7 Microprograma: Ejemplo 1

- *Microprograma en RTL*
  - *Micro-operaciones en RTL*
    - $R_3 \leftarrow R_1 - R_2$: actualizar C y Z
    - Si $R_1 < R_2$: detectado por C = 0, entonces $R_4 \leftarrow R_4 + 1$

      Si $R_1 > R_2$: detectado por C = 1 y Z = 0, entonces $R_4 \leftarrow R_4 + R_2$

      Si $R_1 = R_2$: detectado por Z = 1, entonces $R_4 \leftarrow R_4 + R_1$
    - Salida $\leftarrow R_4$ y bifurcar a dirección externa

2.7.1 Microprograma: Descripción RTL

Dirección	Micro-operaciones y condiciones de bifurcación
20	$R_3 \leftarrow R_1 - R_2$, CAR ← CAR + 1
21	Si C = 1 entonces CAR ← 23, si no CAR ← CAR + 1
22	$R_4 \leftarrow R_4 + R_1$, CAR ← 26
23	Si Z = 0 entonces CAR ← 25, si no CAR ← CAR + 1
24	$R_4 \leftarrow R_4 + 1$, CAR ← 26
25	$R_4 \leftarrow R_4 + R_2$, CAR ← CAR + 1
26	Salida ← R_4, CAR ← Dirección externa

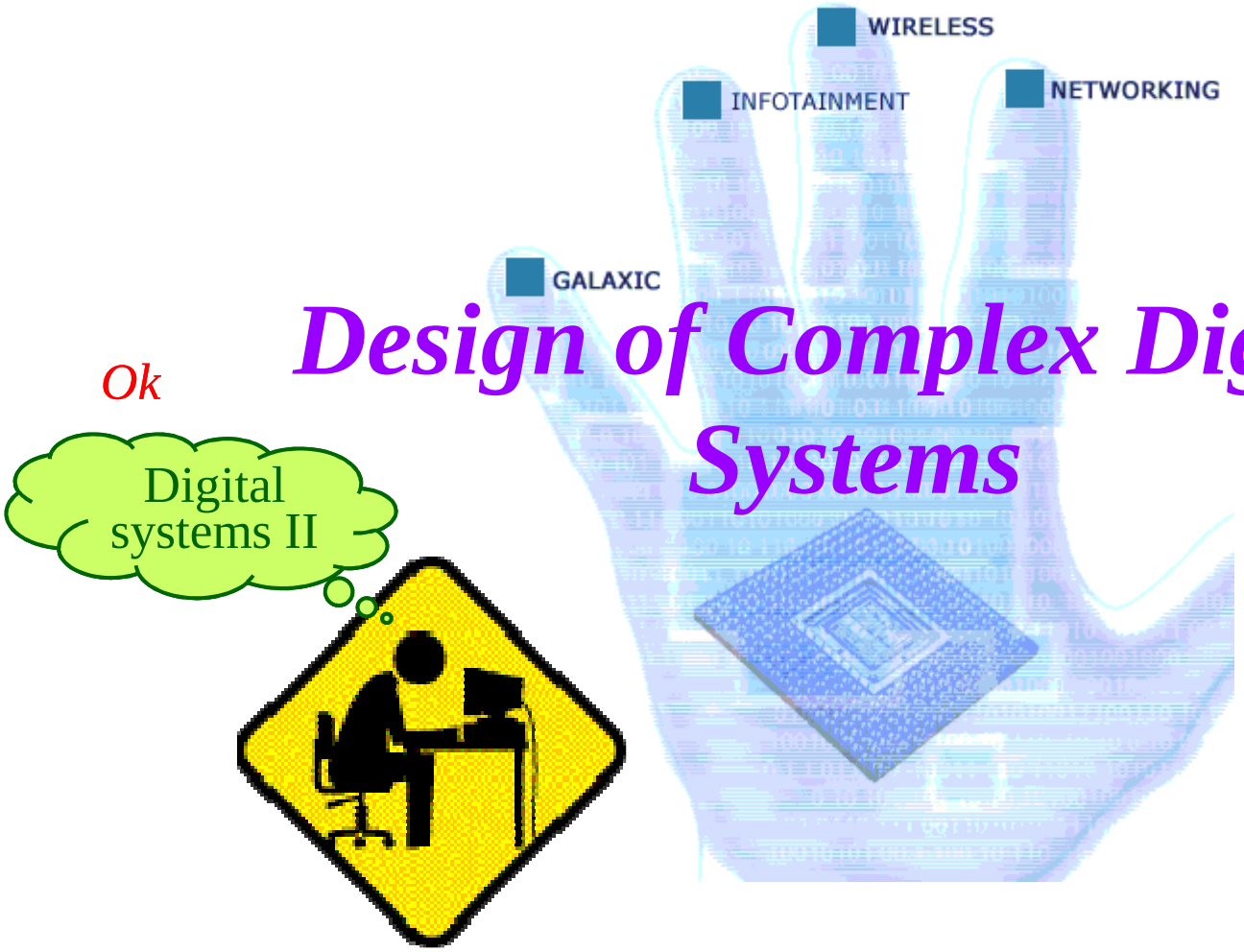
2.7.2 Microprograma: Descripcion Simbólica

Dirección CAR	Palabra de control							Campo de dirección
	A	B	D	F	H	MUX1	MUX2	
20	R_1	R_2	R_3	SUB	NSH	-	NEXT	-
21	-	-	none	TSF	NSH	INT	LC	23
22	R_4	R_1	R_4	ADD	NSH	INT	LAD	26
23	-	-	none	TSF	NSH	INT	LNZ	25
24	R_4	-	R_4	INC	NSH	INT	LAD	26
25	R_4	R_2	R_4	ADD	NSH	-	NEXT	-
26	R_4	-	none	TSF	NSH	EXT	LAD	-

2.7.3 Microprograma: Código Binario

Dirección EEPROM	Contenido en la memoria EEPROM						
	A	B	D	F	H	MUX	ADRS
010100	001	010	011	0101	000	0 000	000000
010101	000	000	000	0000	000	0 010	010111
010110	100	001	100	0010	000	0 001	011010
010111	000	000	000	0000	000	0 101	011001
011000	100	000	100	0001	000	0 001	011010
011001	100	010	100	0010	000	0 000	000000
011010	100	000	000	0000	000	0 001	000000

Design of Complex Digital Systems



Design of Complex Digital Systems

Digital Systems II