

---

# Estructura de Computadores

## *El computador simple*

---

Autores: David Guerrero. Isabel Gómez

Usted es libre de copiar, distribuir y comunicar públicamente la obra y de hacer obras derivadas siempre que se cite la fuente y se respeten las condiciones de la licencia Attribution-Share alike de Creative Commons.

Texto completo de la licencia: <http://creativecommons.org/licenses/by-nc-sa/3.0/es/>

---

---

# Guión

- ▶ **El punto de partida: La calculadora**
- ▶ **Automatización en la ejecución**
- ▶ **Almacenamiento de los datos**
- ▶ **Diversificación de instrucciones**
- ▶ **Una posible implementación**
- ▶ **Ejemplos de uso**

---

# Almacenamiento de datos

- ▶ Es necesario aumentar la capacidad de almacenamiento de datos del sistema y que no quede únicamente limitada a sus 8 registros.
- ▶ Para ello se ampliará la arquitectura anterior añadiéndole una memoria RAM. Al nuevo sistema le llamaremos el computador simple 2 (CS2).
- ▶ Existen dos opciones para dotar al sistema de memoria RAM:
  - ▶ Utilizar un único sistema de memoria para datos e instrucciones (arquitectura Von Neumann).
  - ▶ Utilizar sistemas de memoria distintos para datos e instrucciones (arquitectura Harvard).

---

# Almacenamiento de datos

- ▶ En la arquitectura Harvard, las características de las memorias y los buses de interconexión de las mismas pueden diferir. Normalmente los datos requieren memoria de lectura y escritura
- ▶ En la arquitectura de Von Neuman el sistema de memoria es único y por lo tanto tanto memoria como buses son únicos.
- ▶ El disponer de dos sistemas de memoria separados dota de eficiencia al sistema ya que permite acceder a instrucciones y datos simultáneamente.
- ▶ Muchas CPU modernas incorporan aspectos de ambas arquitecturas ya que disponen de una memoria principal común pero disponen de memorias caches separas para datos e instrucciones.

---

# Almacenamiento de datos

- ▶ El CS2 dispondrá de una arquitectura Harvard.
- ▶ El conjunto de instrucciones debe ser ampliado ya que se requiere manejar los datos almacenados en la memoria.

# Almacenamiento de datos

Conjunto de instrucciones (ISP) del CS2

CO	SINTAXIS	FUNCIÓN
000	ST (Rb),Rf	MEMDAT(Rb) $\leftarrow$ Rf
001	LD Rd, (Rb)	Rd $\leftarrow$ MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) $\leftarrow$ Rf
011	LDS Rd,dir	Rd $\leftarrow$ MEMDAT(dir)
100	ADD Rd,Rf	Rd $\leftarrow$ Rd+Rf
110	SUB Rd,Rf	Rd $\leftarrow$ Rd-Rf
101	MOV Rd,Rf	Rd $\leftarrow$ Rf
111	STOP	NOP

- ▶ Las 4 primeras instrucciones son para intercambio de datos con la memoria.
- ▶ Es necesario aumentar los bits del código de operación.
- ▶ Se han añadido nuevas formas de acceso a los operandos (modos de direccionamiento).

# Almacenamiento de datos

Conjunto de instrucciones (ISP) del CS2

CO	SINTAXIS	FUNCIÓN
000	ST (Rb),Rf	MEMDAT(Rb) $\leftarrow$ Rf
001	LD Rd, (Rb)	Rd $\leftarrow$ MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) $\leftarrow$ Rf
011	LDS Rd,dir	Rd $\leftarrow$ MEMDAT(dir)
100	ADD Rd,Rf	Rd $\leftarrow$ Rd+Rf
110	SUB Rd,Rf	Rd $\leftarrow$ Rd-Rf
101	MOV Rd,Rf	Rd $\leftarrow$ Rf
111	STOP	NOP

**Arquitectura load/store**  
Las únicas instrucciones con operandos en memoria son las de carga (load) y almacenamiento (store).

- ▶ Las 4 primeras instrucciones son para intercambio de datos con la memoria.
- ▶ Es necesario aumentar los bits del código de operación.
- ▶ Se han añadido nuevas formas de acceso a los operandos (modos de direccionamiento).

# Almacenamiento de datos

Conjunto de instrucciones (ISP) del CS2

CO	SINTAXIS	FUNCIÓN
000	ST (Rb),Rf	MEMDAT(Rb) $\leftarrow$ Rf
001	LD Rd, (Rb)	Rd $\leftarrow$ MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) $\leftarrow$ Rf
011	LDS Rd, dir	Rd $\leftarrow$ MEMDAT(dir)
100	ADD Rd,Rf	Rd $\leftarrow$ Rd+Rf
110	SUB Rd,Rf	Rd $\leftarrow$ Rd-Rf
101	MOV Rd,Rf	Rd $\leftarrow$ Rf
111	STOP	NOP

Las operaciones aritméticas usan operandos de tipo registro.

- ▶ Las 4 primeras instrucciones son para intercambio de datos con la memoria.
- ▶ Es necesario aumentar los bits del código de operación.
- ▶ Se han añadido nuevas formas de acceso a los operandos (modos de direccionamiento).



# Almacenamiento de datos

Conjunto de instrucciones (ISP) del CS2

CO	SINTAXIS	FUNCIÓN
000	ST (Rb),Rf	MEMDAT(Rb) $\leftarrow$ Rf
001	LD Rd, (Rb)	Rd $\leftarrow$ MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) $\leftarrow$ Rf
011	LDS Rd,dir	Rd $\leftarrow$ MEMDAT(dir)
100	ADD Rd,Rf	Rd $\leftarrow$ Rd+Rf
110	SUB Rd,Rf	Rd $\leftarrow$ Rd-Rf
101	MOV Rd,Rf	Rd $\leftarrow$ Rf
111	STOP	NOP

- ▶ Las 4 primeras instrucciones son para intercambio de datos con la memoria.
- ▶ Es necesario aumentar los bits del código de operación.
- ▶ Se han añadido nuevas formas de acceso a los operandos (modos de direccionamiento).

# Almacenamiento de datos

Conjunto de instrucciones (ISP) del CS2

CO	SINTAXIS	FUNCIÓN
000	ST (Rb), Rf	MEMDAT(Rb) $\leftarrow$ Rf
001	LD Rf, (Rb)	Rd $\leftarrow$ MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) $\leftarrow$ Rf
011	LDS Rf, dir	Rd $\leftarrow$ MEMDAT(dir)
100	ADD Rd, Rf	Rd $\leftarrow$ Rd + Rf
110	SUB Rd, Rf	Rd $\leftarrow$ Rd - Rf
101	MOV Rd, Rf	Rd $\leftarrow$ Rf
111	STOP	NOP

Formas de direccionar la memoria  
Direccionamiento absoluto  
Direccionamiento indirecto de registro

- ▶ Las 4 primeras instrucciones son para intercambio de datos con la memoria.
- ▶ Es necesario aumentar los bits del código de operación.
- ▶ Se han añadido nuevas formas de acceso a los operandos (modos de direccionamiento)

# Almacenamiento de datos

Modos de direccionamiento del CS2:

- ▶ Directo de registro:

- ▶ El dato se encuentra en un registro. `MOV Rd,Rf`

- ▶ Indirecto de registro:

- ▶ El dato se encuentra en una posición de memoria cuya dirección está almacenada en un registro denominado “registro base”. `ST (Rb),Rf`

- ▶ Absoluto:

- ▶ El dato se encuentra en una posición de memoria cuya dirección se da como operando. `STS dir, Rf`

No permite direccionamiento inmediato, es decir, usar el propio operando como dato en las operaciones.

MOV Rd,Rf

MOV R1,R2

Antes de la ejecución

		dir	cont
\$03	R1	0	\$34
\$58	R2	1	\$56
		2	\$78
		3	\$00
Memoria			

Después de la ejecución

		dir	cont
\$58	R1	0	\$34
\$58	R2	1	\$56
		2	\$78
		3	\$00
Memoria			

ST (Rb),Rf

ST (R1),R2

Antes de la ejecución

		dir	cont
\$03	R1	0	\$34
\$58	R2	1	\$56
		2	\$78
		3	\$00
Memoria			

Después de la ejecución

		dir	cont
\$03	R1	0	\$34
\$58	R2	1	\$56
		2	\$78
		3	\$58
Memoria			

STS dir, Rf

STS 2,R2

Antes de la ejecución

		dir	cont
\$03	R1	0	\$34
\$58	R2	1	\$56
		2	\$78
		3	\$00
Memoria			

Después de la ejecución

		dir	cont
\$03	R1	0	\$34
\$58	R2	1	\$56
		2	\$58
		3	\$00
Memoria			

# Almacenamiento de datos

## Formato de instrucción del CS2

<u>formato</u>	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>A</b>	código de operación			registro destino (fuente en ST)			-	-	-	-	-	registro fuente (registro base en ST/LD)		
<b>B</b>							dirección del dato							

Este formato es necesario para el modo de direccionamiento absoluto.

Campo para guardar dirección de 8 bits.

Las instrucciones son de 14 bits y siguen ocupando una palabra de memoria. En el CS2 el ancho de la memoria de código será de 14 bits. También será necesario que el registro IR sea de este tamaño.

# Almacenamiento de datos

## Formato de instrucción del CS2

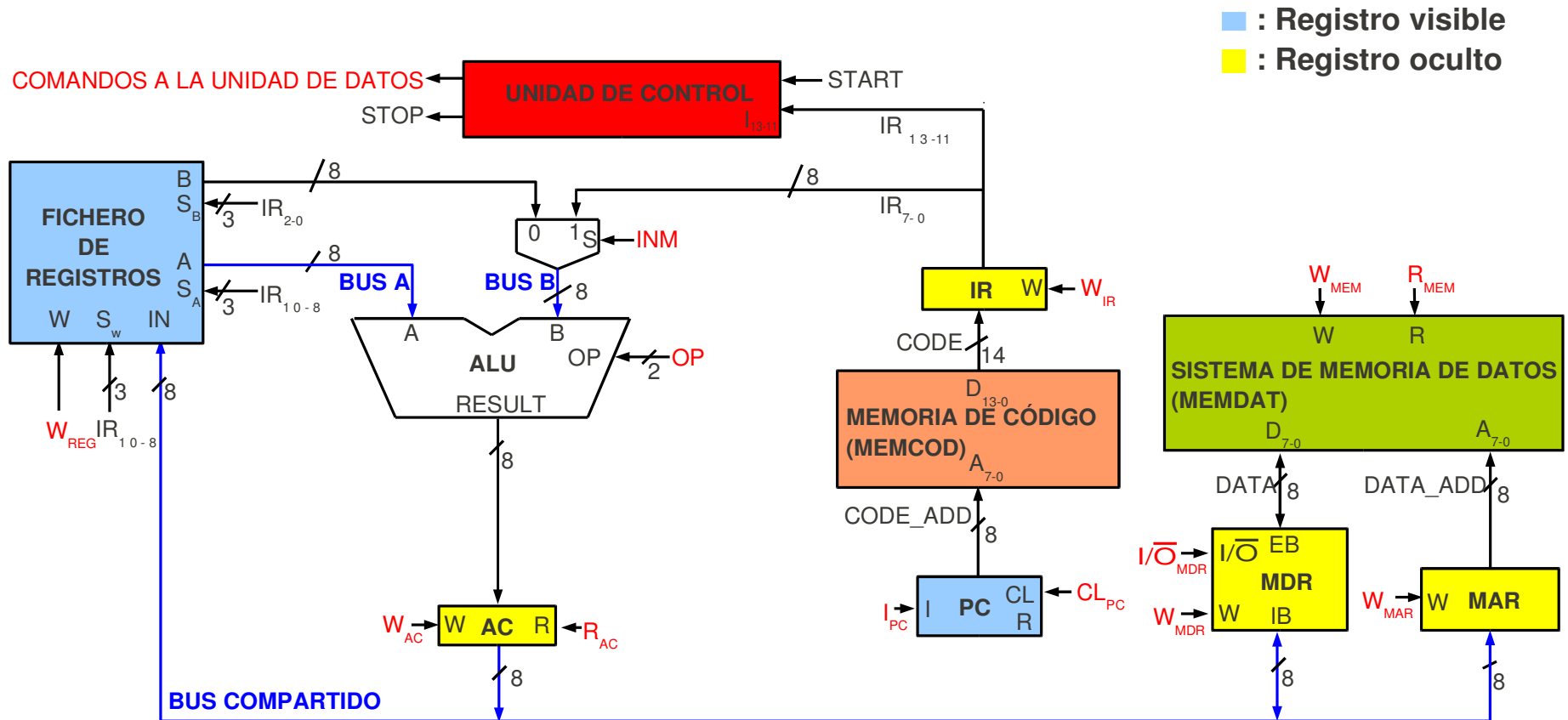
<u>formato</u>	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<b>A</b>	código de operación			registro destino (fuente en ST)			-	-	-	-	-	registro fuente (registro base en ST/LD)		
<b>B</b>							dirección del dato							

MOV R3,R2	101	011	- - - - -	010
ST (R4),R2	000	010	- - - - -	100
STS \$F3,R2	010	010	1111	0011

Las instrucciones son de 14 bits y siguen ocupando una palabra de memoria. En el CS2 el ancho de la memoria de código será de 14 bits. También será necesario que el registro IR sea de este tamaño.

# Almacenamiento de datos

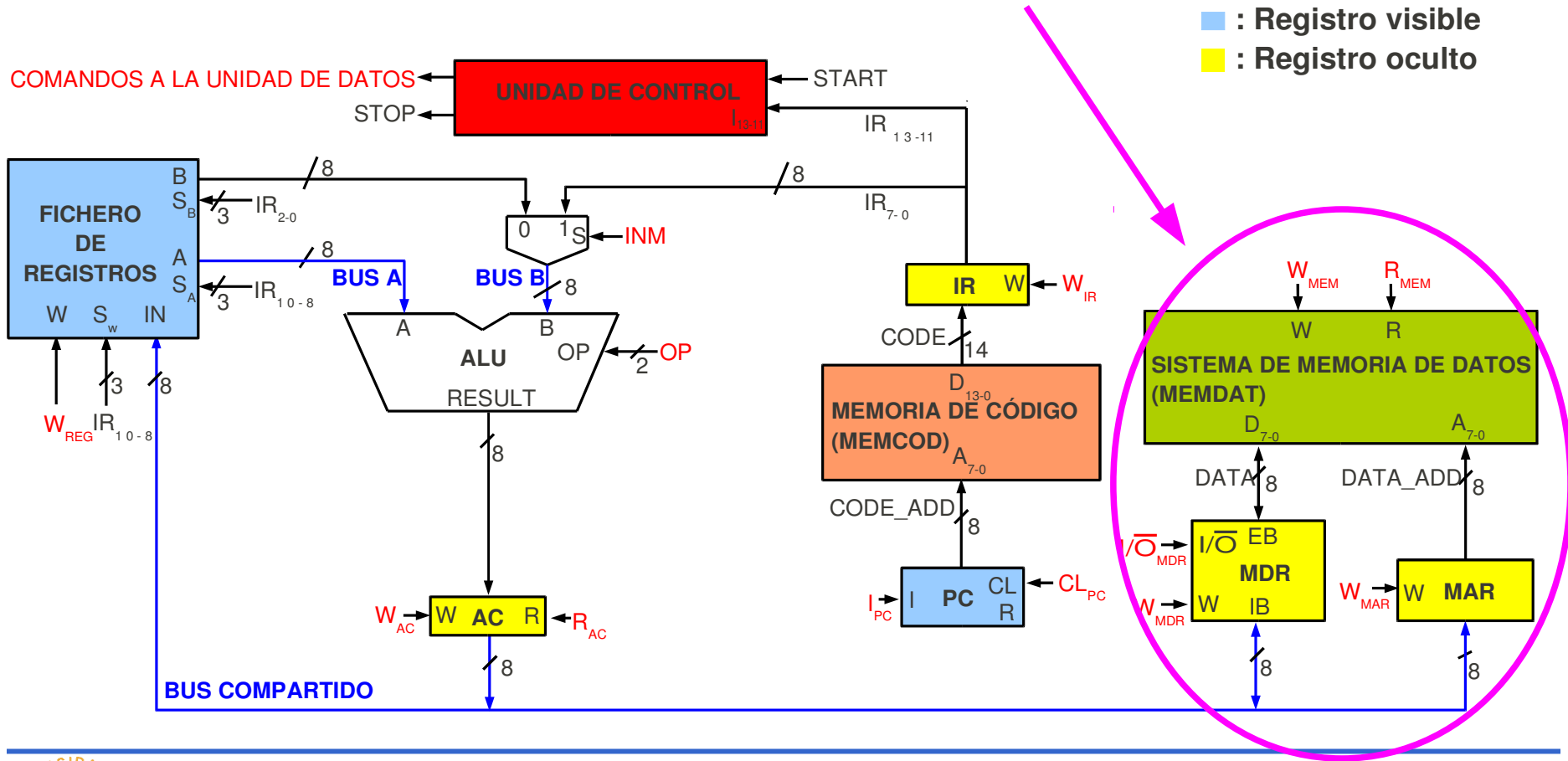
## Implementación del CS2



# Almacenamiento de datos

# Implementación del CS2

## Parte añadida para almacenar más datos

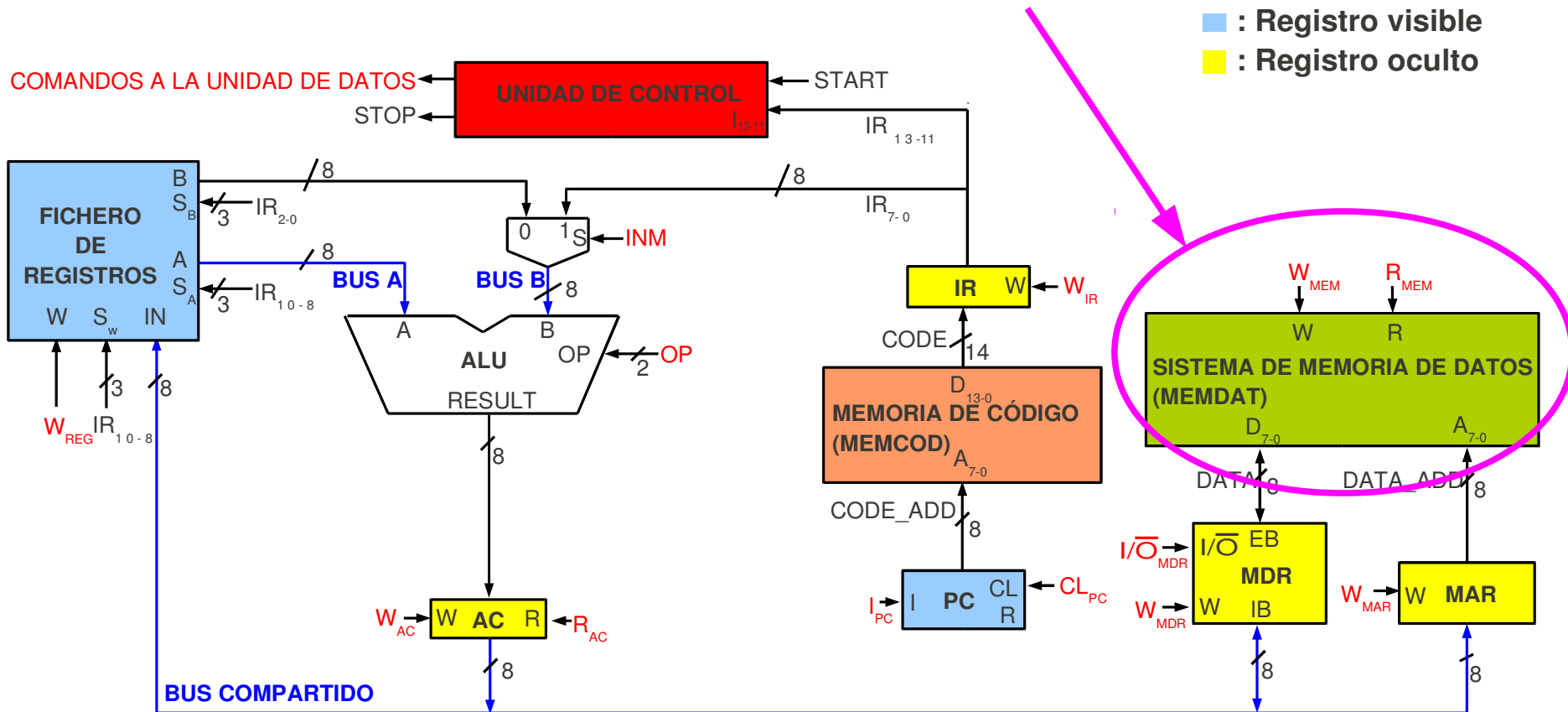




# Almacenamiento de datos

# Implementación del CS2

**Memoria de lectura/escritura con capacidad 2<sup>8</sup>x8 bits.**  
**Terminal de datos bidireccional**

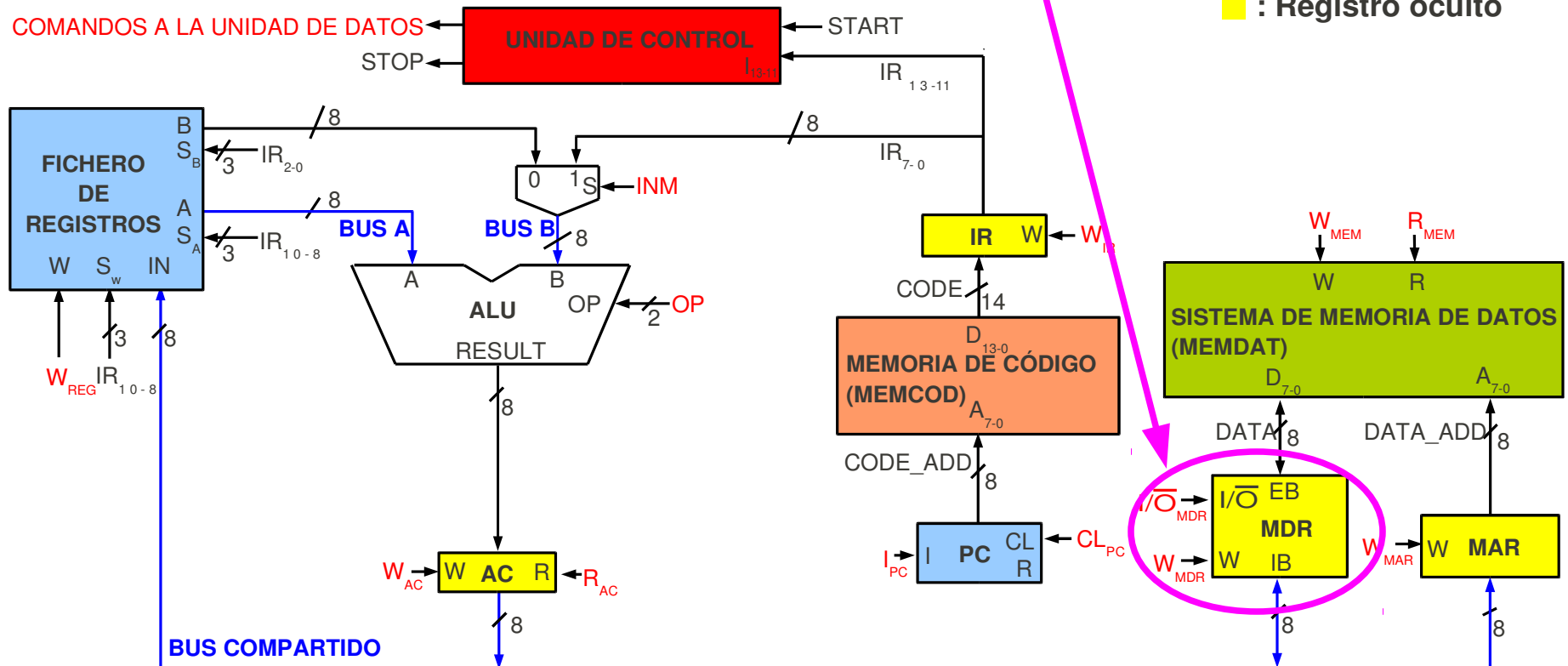


# Almacenamiento de datos

## Implementación del CS2

Registro que guarda de manera temporal los datos que van a intercambiarse con la memoria

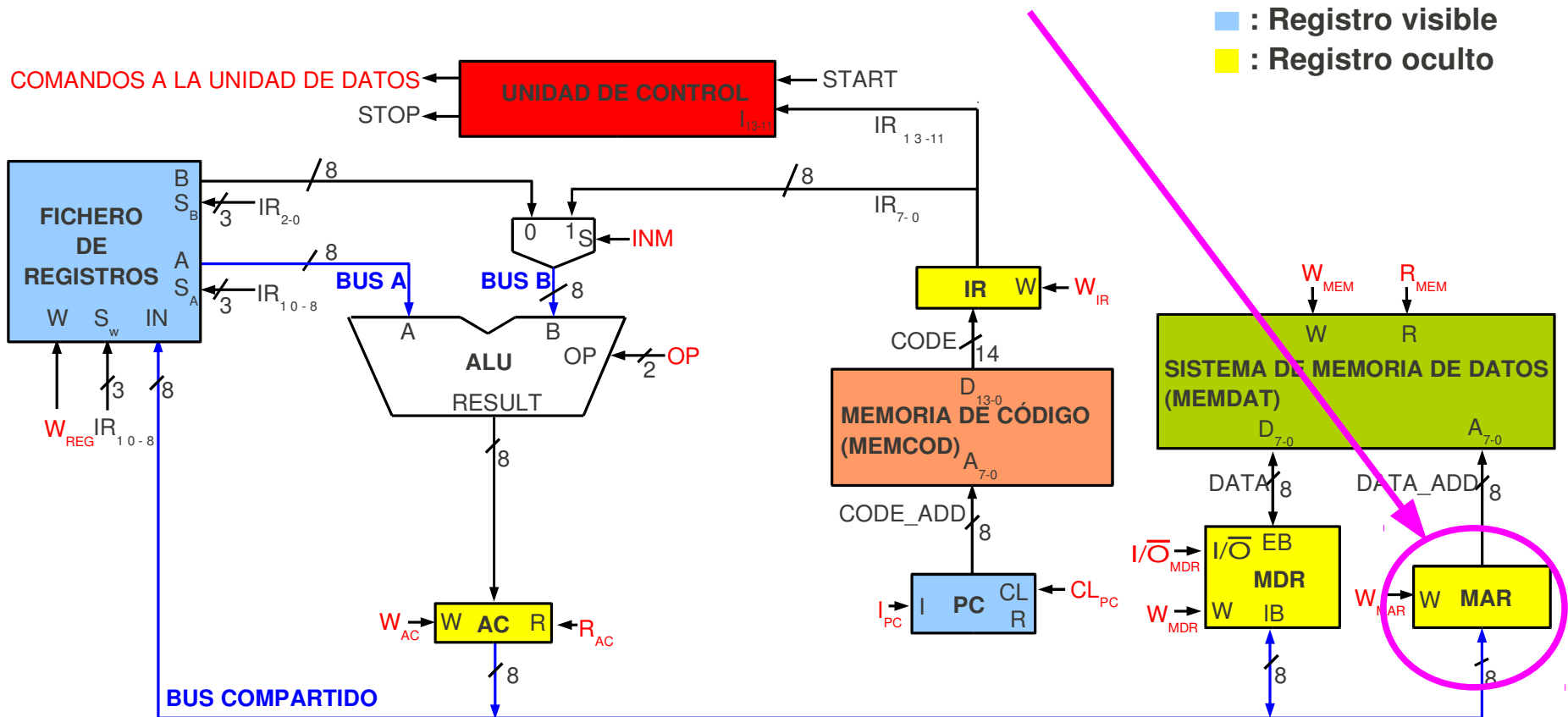
■ : Registro visible  
■ : Registro oculto



# Almacenamiento de datos

## Implementación del CS2

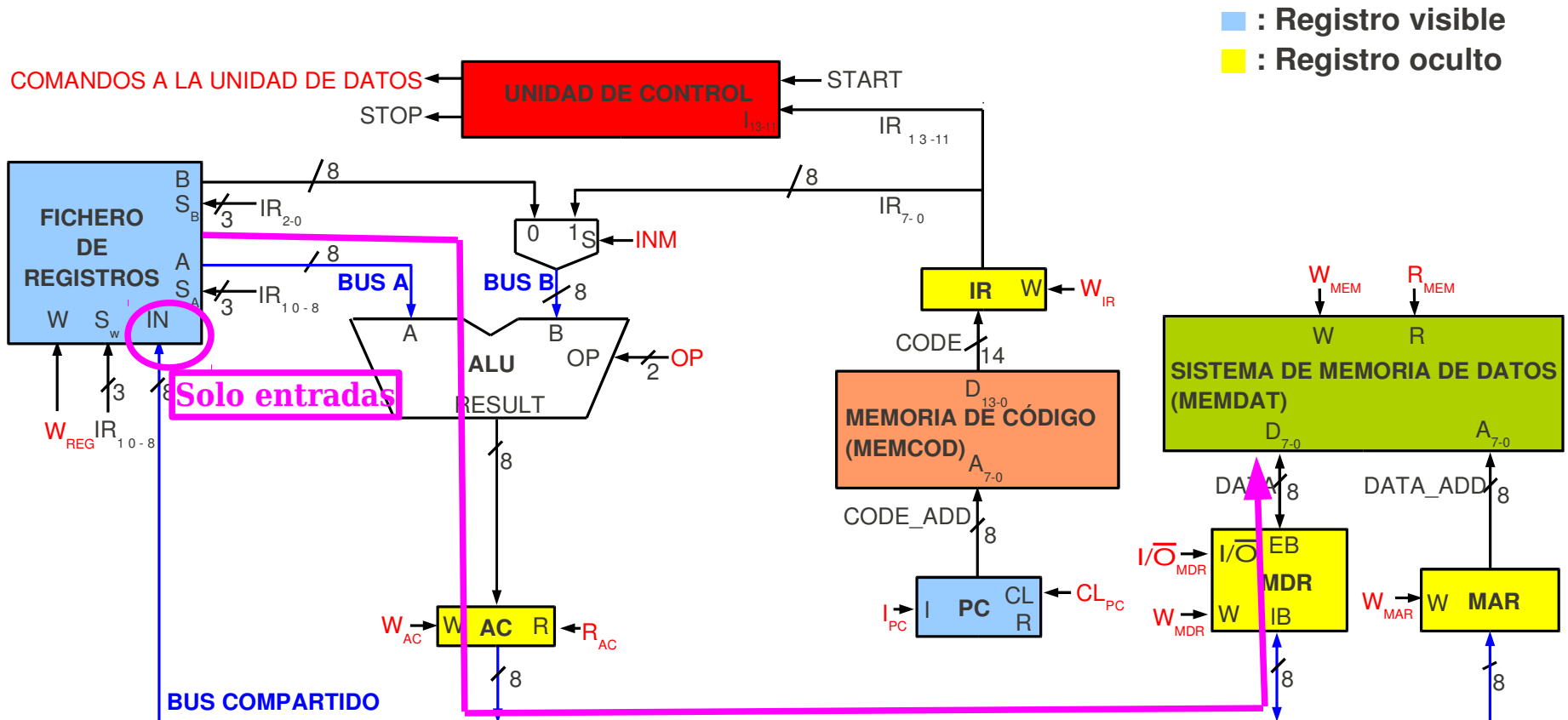
Registro que guarda de manera temporal las direcciones de las posiciones de la memoria de datos que serán accedidas.



# Almacenamiento de datos

El camino para pasar datos del banco de registros a la memoria pasa siempre por la ALU y el registro AC.

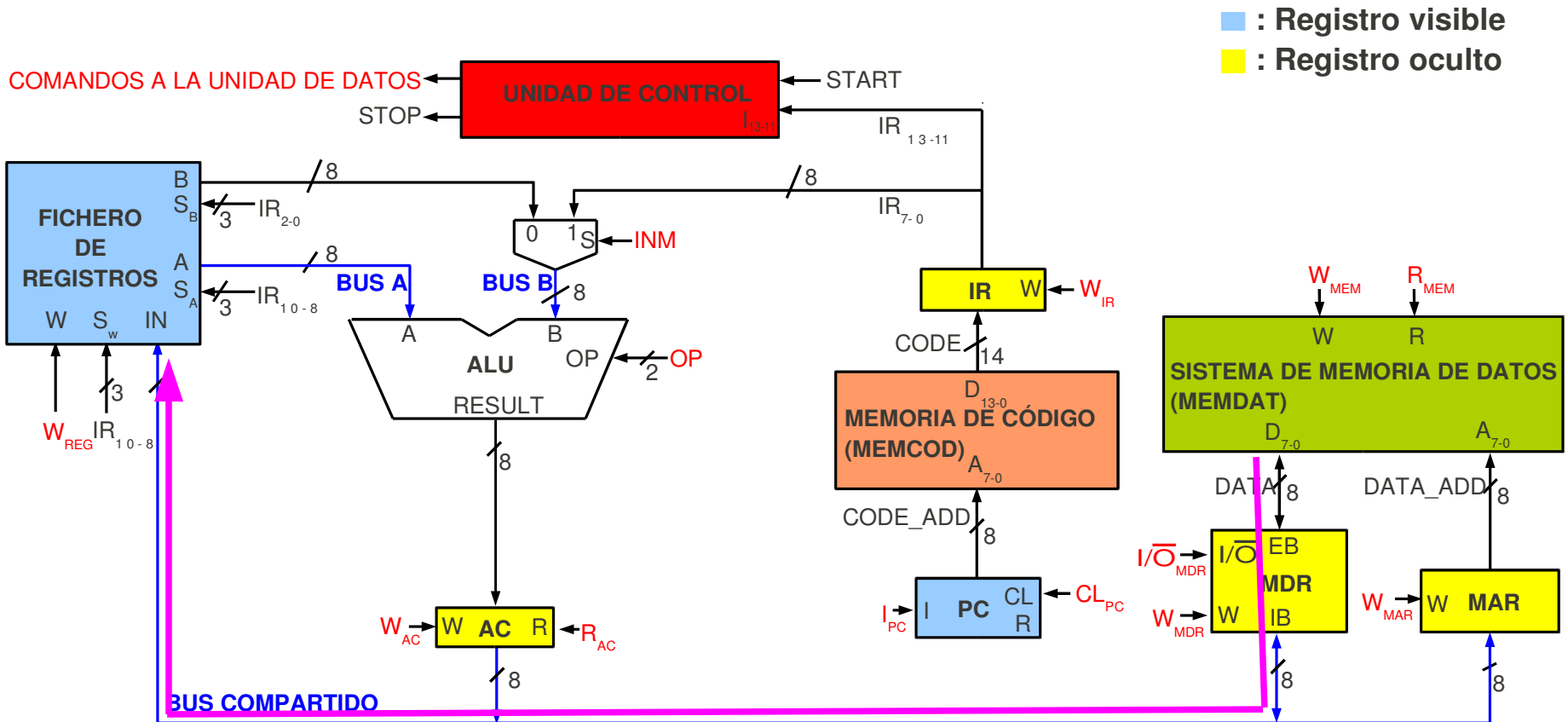
## Implementación del CS2



# Almacenamiento de datos

## Implementación del CS2

Camino para que los datos vayan de la memoria a los registros. Usa el bus compartido.

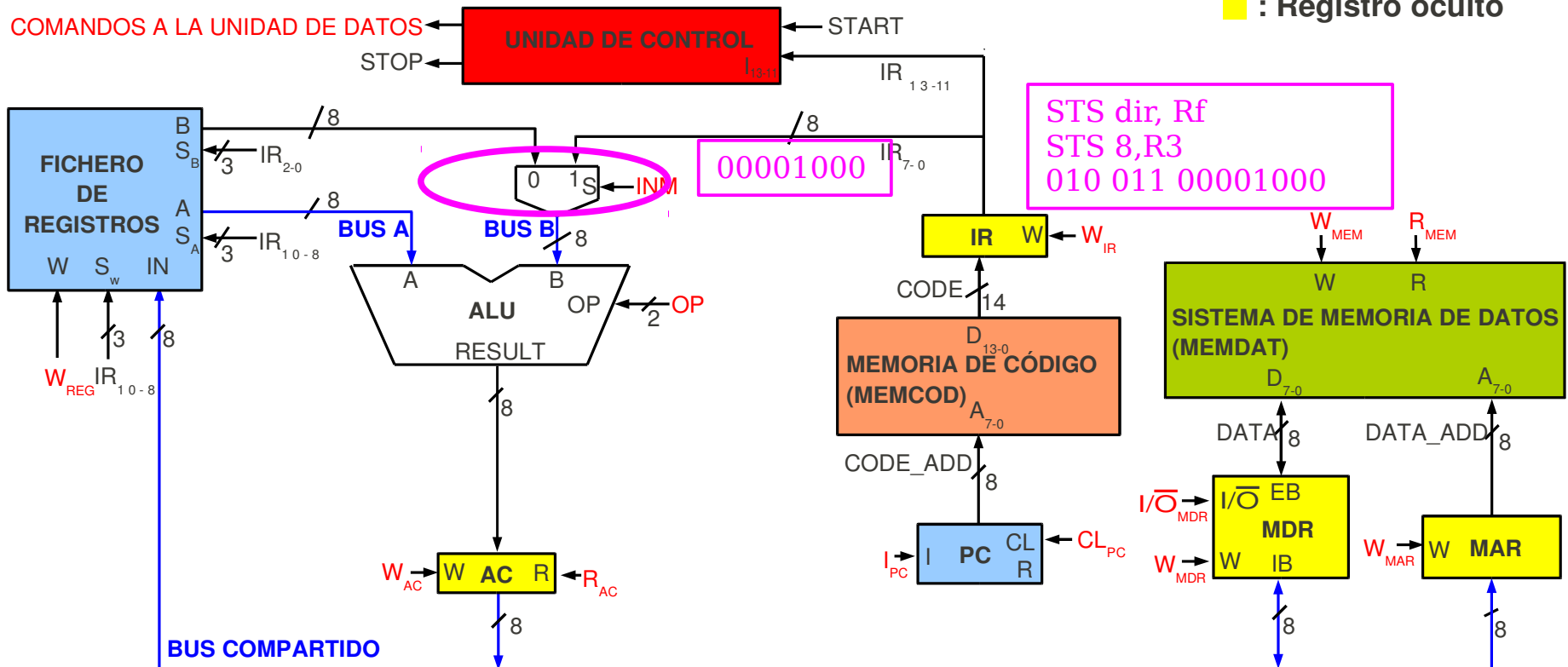


<u>formato</u>	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A	código de operación			registro destino (fuente en ST)			-	-	-	-	-	registro fuente (registro base en ST/LD)		
B							dirección del dato							

## Implementación del CS2

**Direccionamiento absoluto.**  
**Necesidad del MUX a la entrada de la ALU**

■ : Registro visible  
■ : Registro oculto

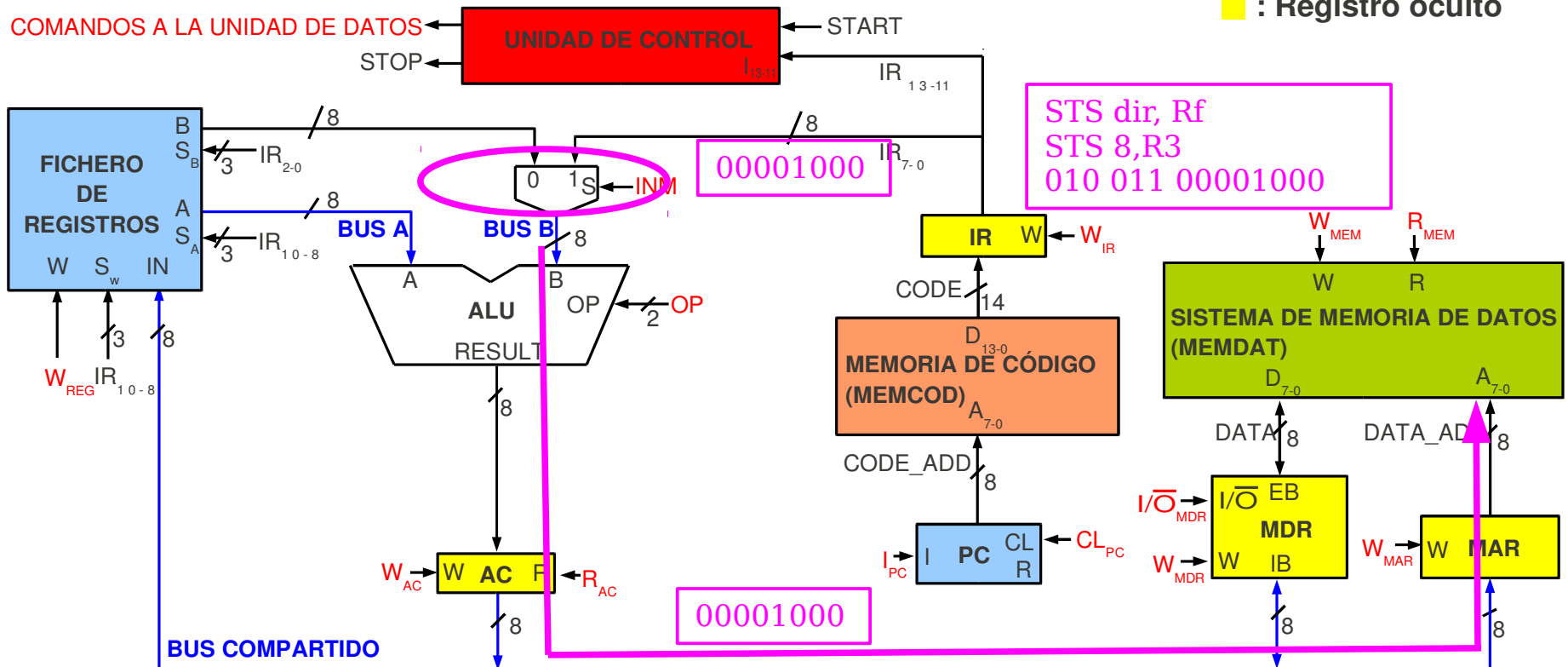


<u>formato</u>	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A	código de operación			registro destino (fuente en ST)			-	-	-	-	-	registro fuente (registro base en ST/LD)		
B							dirección del dato							

## Implementación del CS2

**Direccionamiento absoluto.**  
**Necesidad del MUX a la entrada de la ALU**

■ : Registro visible  
■ : Registro oculto



---

# Almacenamiento de datos

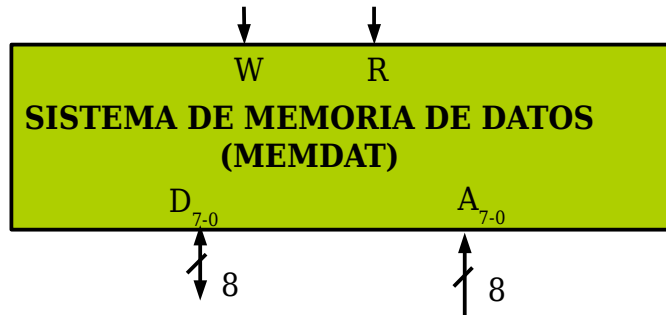
Modificaciones en la arquitectura del CS2: todo lo que se ha añadido es para dotar al sistema de almacenamiento de datos en memoria.

- ▶ Sistema de memoria de datos.
- ▶ Registro de datos de la memoria (MDR).
- ▶ Registro de direcciones de la memoria (MAR).
- ▶ Acumulador (AC). Se ha añadido este registro a la salida de la ALU porque esta debe compartir el bus.
- ▶ Multiplexor a la entrada de la ALU porque esta debe transferir datos tanto de los registros como de los 8 bits menos significativos del registro IR a MAR en el caso de direccionamiento absoluto.

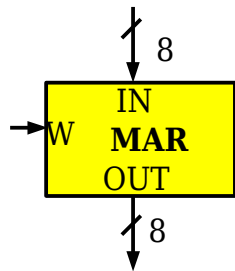


# Almacenamiento de datos

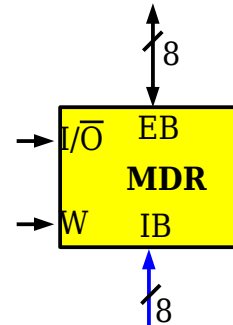
Descripción RT de los nuevos componentes



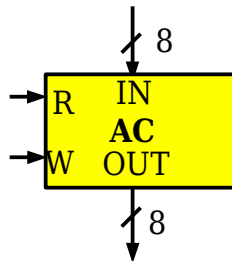
W R	MEMDAT[A <sub>7-0</sub> ] ←	D <sub>7-0</sub> :=
00	MEMDAT[A <sub>7-0</sub> ]	H.I.
01	MEMDAT[A <sub>7-0</sub> ]	MEMDAT[A <sub>7-0</sub> ]
10	D <sub>7-0</sub>	H.I.
11	PROHIBIDO	PROHIBIDO



W	MAR ←	OUT :=
0	MAR	MAR
1	IN	MAR



W I/O	MDR ←	IB :=	EB :=
00	MDR	H.I.	MDR
01	MDR	MDR	H.I.
10	IB	H.I.	H.I.
11	EB	H.I.	H.I.



W R	AC ←	OUT :=
00	AC	H.I.
01	AC	AC
10	IN	H.I.
11	IN	AC

# Almacenamiento de datos

ST (Rb),Rf

1.  $AC \leftarrow REG[IR_{2-0}]$

2.  $MAR \leftarrow AC; AC \leftarrow REG[IR_{10-8}]$

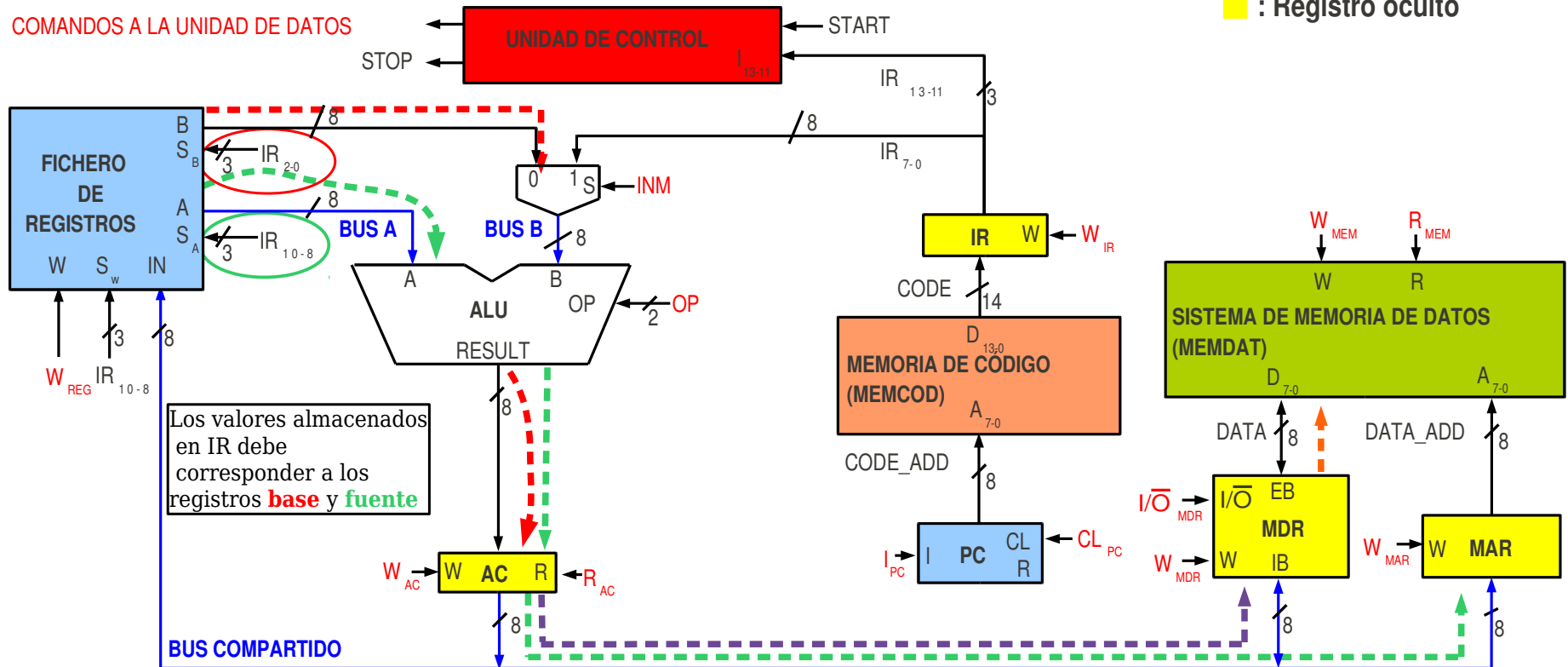
3.  $MDR \leftarrow AC$

4.  $MEMDAT[MAR] \leftarrow MDR$

OP<sub>1</sub> OP<sub>0</sub> W<sub>AC</sub>  
R<sub>AC</sub> W<sub>MAR</sub> OP<sub>0</sub> W<sub>AC</sub>  
R<sub>AC</sub> W<sub>MDR</sub>  
W<sub>MEM</sub>

■ : Registro visible  
■ : Registro oculto

COMANDOS A LA UNIDAD DE DATOS



# Almacenamiento de datos

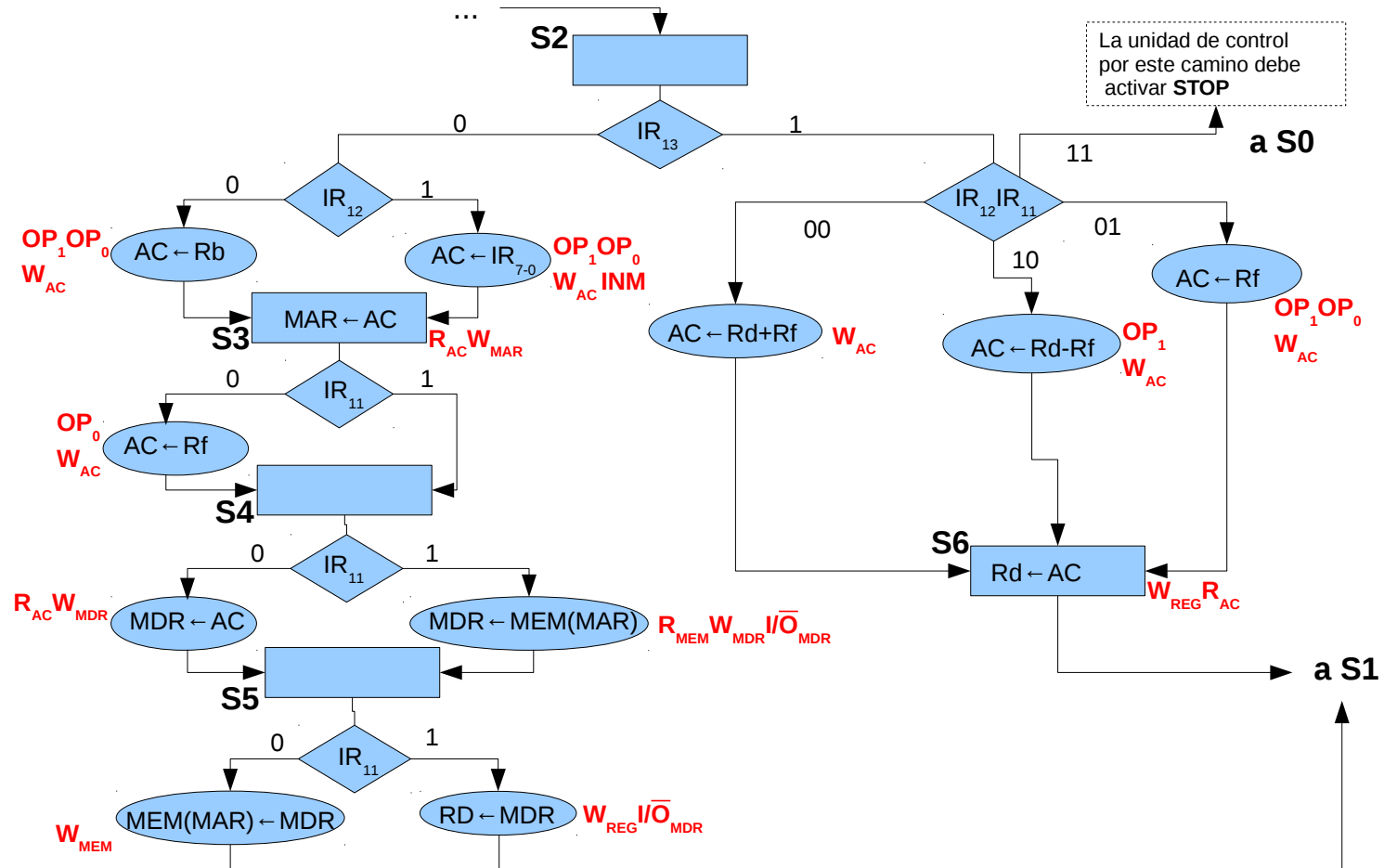
Descripción de las microoperaciones de las instrucciones añadidas

CO ( $I_{13} I_{12} I_{11}$ )	000	001	010	011
Instrucción	ST (Rb),Rf	LD Rd,(Rb)	STS dir,Rf	LDS Rd,dir
Micro 1	$AC \leftarrow Rb$ $OP_1 OP_0 W_{AC}$	$AC \leftarrow Rb$ $OP_1 OP_0 W_{AC}$	$AC \leftarrow dir$ $OP_1 OP_0 W_{AC} INM$	$AC \leftarrow dir$ $OP_1 OP_0 W_{AC} INM$
Micro 2	$MAR \leftarrow AC$ $R_{AC} W_{MAR}$ $AC \leftarrow Rf$ $OP_0 W_{AC}$	$MAR \leftarrow AC$ $R_{AC} W_{MAR}$	$MAR \leftarrow AC$ $R_{AC} W_{MAR}$ $AC \leftarrow Rf$ $OP_0 W_{AC}$	$MAR \leftarrow AC$ $R_{AC} W_{MAR}$
Micro 3	$MDR \leftarrow AC$ $R_{AC} W_{MDR}$	$MDR \leftarrow MEM(MAR)$ $R_{MEM} W_{MDR} I/\overline{O}_{MDR}$	$MDR \leftarrow AC$ $R_{AC} W_{MDR}$	$MDR \leftarrow MEM(MAR)$ $R_{MEM} W_{MDR} I/\overline{O}_{MDR}$
Micro 4	$MEM(MAR) \leftarrow MDR$ $W_{MEM}$	$Rd \leftarrow MDR$ $W_{REG} I/\overline{O}_{MDR}$	$MEM(MAR) \leftarrow MDR$ $W_{MEM}$	$Rd \leftarrow MDR$ $W_{REG} I/\overline{O}_{MDR}$

Las otras 4 instrucciones tienen una microoperación más que en el CS1 debido al AC. Esto ha sido reflejado en la carta ASM.

# Almacenamiento de datos

Carta ASM del CS2



# Almacenamiento de datos

Ejemplo de uso del CS2: Realizar un programa que intercambie dos tablas de 4 datos que se encuentran almacenadas en la memoria y cuyas direcciones base están guardadas en los registros R0 y R1 respectivamente. El registro R2 tiene almacenado el valor 1.

**Programa**  
LD R3,(R0)  
LD R3,(R0)  
LD R4,(R1)  
ST (R1),R3  
ST (R0),R4  
ST (R0),R4  
ADD R0,R2  
ADD R1,R2  
LD R3,(R0)  
LD R4,(R1)  
LD R4,(R1)  
ST (R1),R3  
ST (R0),R4  
ST (R0),R4  
ADD R0,R2  
ADD R1,R2

LD R3,(R0)  
LD R4,(R1)  
ST (R1),R3  
ST (R0),R4  
ADD R0,R2  
ADD R1,R2  
LD R3,(R0)  
LD R4,(R1)  
LD R4,(R1)  
ST (R1),R3  
ST (R0),R4  
STOP

dirección	contenido
0	001 011-----000
1	001 100-----001
2	000 011-----001
3	000 100-----000
4	100 000-----011
5	100 001-----011
...	...
22	11- - - - -

MEMORIA DE CÓDIGO

Dir.	contenido
0	.....
...	...
R0	DAT01TABLA1
R0+1	DAT02TABLA1
R0+2	DAT03TABLA1
R0+3...	DAT04TABLA1
R1	...
R1+1	DAT01TABLA2
R1+2	DAT02TABLA2
R1+3	DAT03TABLA2
...	DAT04TABLA2

MEMORIA DE DATOS