Estructura de Computadores

El computador simple

Autores: David Guerrero. Isabel Gómez

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 y al nuevo sistema le llamaremos el computador simple 2 (CS2).
- Existen dos opciones para dotar al sistema de almacenamiento de datos:
  - Utilizar un único sistema de memoria para datos e instrucciones lo que se denomina arquitectura Von Neumann.
  - Utilizar sistemas de memoria distintos para datos e instrucciones, lo que es denominado 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 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) ← Rf
001	LD Rd, (Rb)	Rd ← MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) ← Rf
011	LDS Rd,dir	Rd ← MEMDAT(dir)
100	ADD Rd,Rf	Rd ← Rd+Rf
110	SUB Rd,Rf	Rd ← Rd-Rf
101	MOV Rd,Rf	Rd ← 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)←Rf
001	LD Rd, (Rb)	Rd←MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir)←Rf
011	LDS Rd,dir	Rd←MEMDAT(dir)
100	ADD Rd,Rf	Rd←Rd+Rf
110	SUB Rd,Rf	Rd←Rd-Rf
101	MOV Rd,Rf	Rd←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) ← Rf
001	LD Rd, (Rb)	Rd ← MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) ← Rf
011	LDS Rd,dir	Rd ← MEMDAT(dir)
100	ADD Rd,Rf	Rd ← Rd+Rf
110	SUB Rd,Rf	Rd ← Rd-Rf
101	MOV Rd,Rf	Rd ← 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) ← Rf
001	LD Rd, (Rb)	Rd ← MEMDAT(Rb)
010	STS dir, Rf	MEMDAT(dir) ← Rf
011	LDS Rd,dir	Rd ← MEMDAT(dir)
100	ADD Rd,Rf	Rd ← Rd+Rf
110	SUB Rd,Rf	Rd ← Rd-Rf
101	MOV Rd,Rf	Rd ← 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) ← Rf
001	LD Rd,(Rb)	Rd ← MEMDAT(Rb)
010	STS dir,Rf	MEMDAT(dir) ← Rf
011	LDS Rd,dir	Rd ← MEMDAT(dir)
100	ADD Rd,Rf	Rd ← Rd+Rf
110	SUB Rd,Rf	Rd ← Rd-Rf
101	MOV Rd,Rf	Rd ← Rf
111	STOP	NOP

Formas de direccionar la memoria
Direccionamiento registro
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

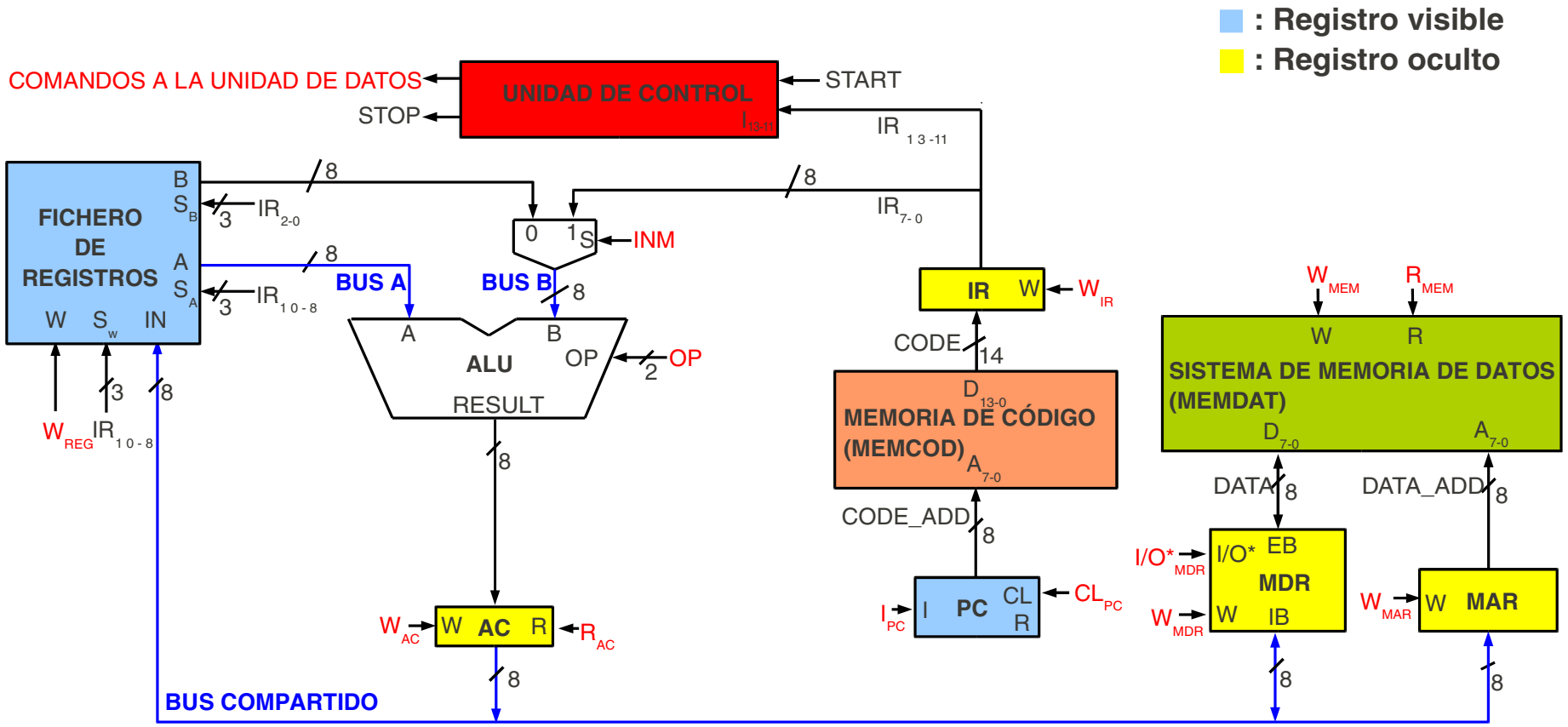
		dir	cont
$03	R1	0	$34
$58	R2	1	$56
		2	$78
		3	$00
		Memoria	

Después de la ejecución

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

Almacenamiento de datos

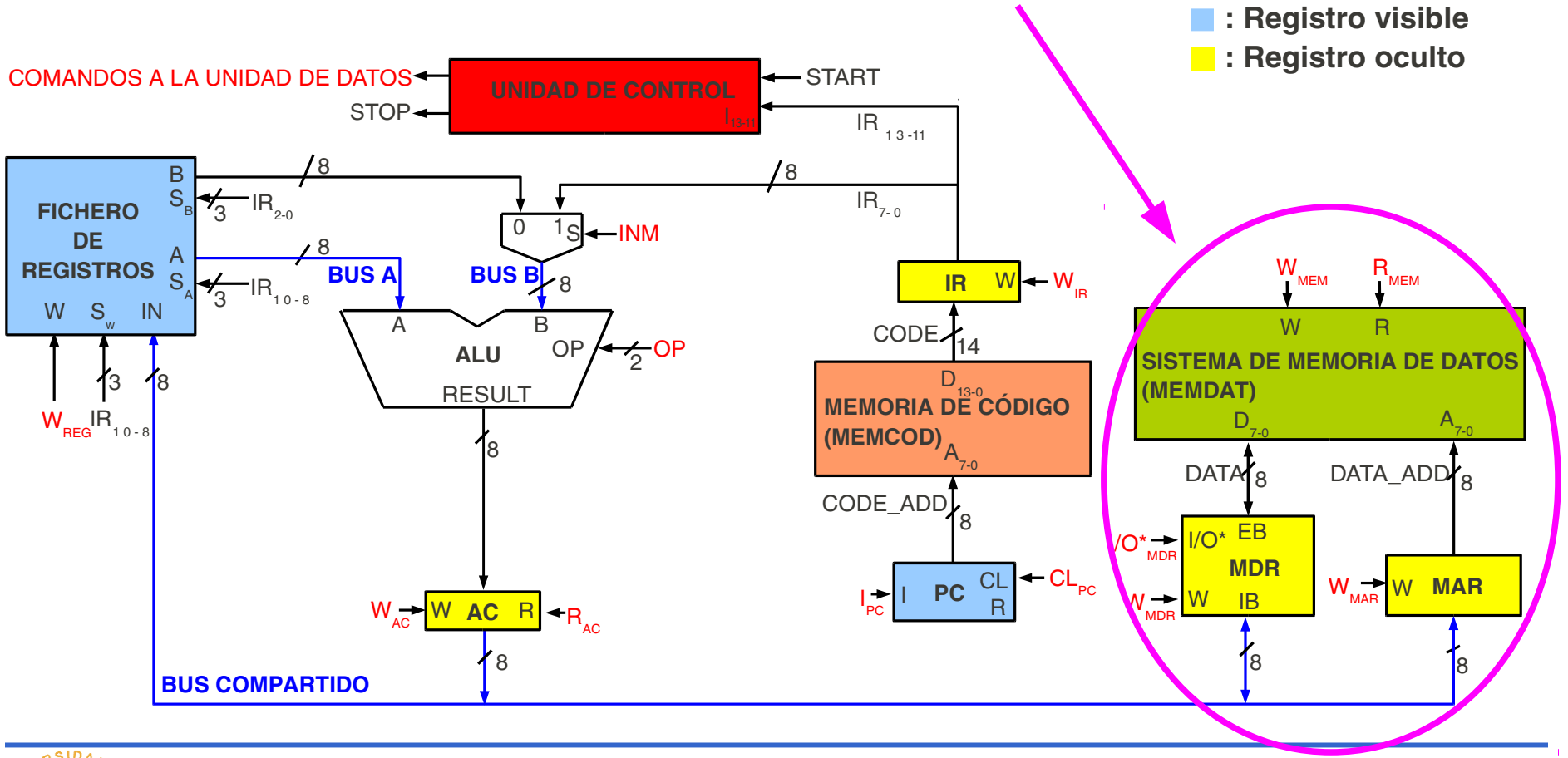
Implementación del CS2



Almacenamiento de datos

Parte añadida para almacenar más datos

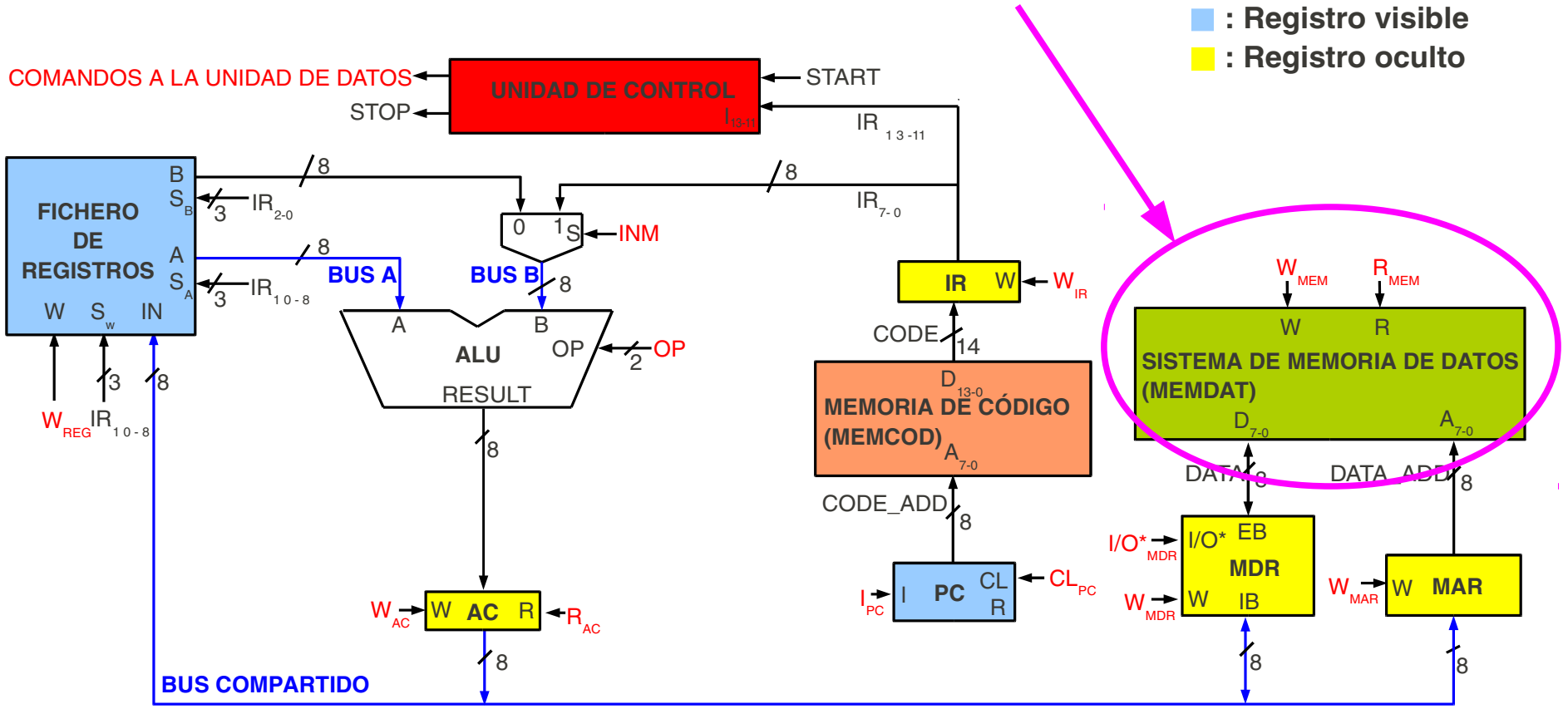
Implementación del CS2



Almacenamiento de datos

Implementación del CS2

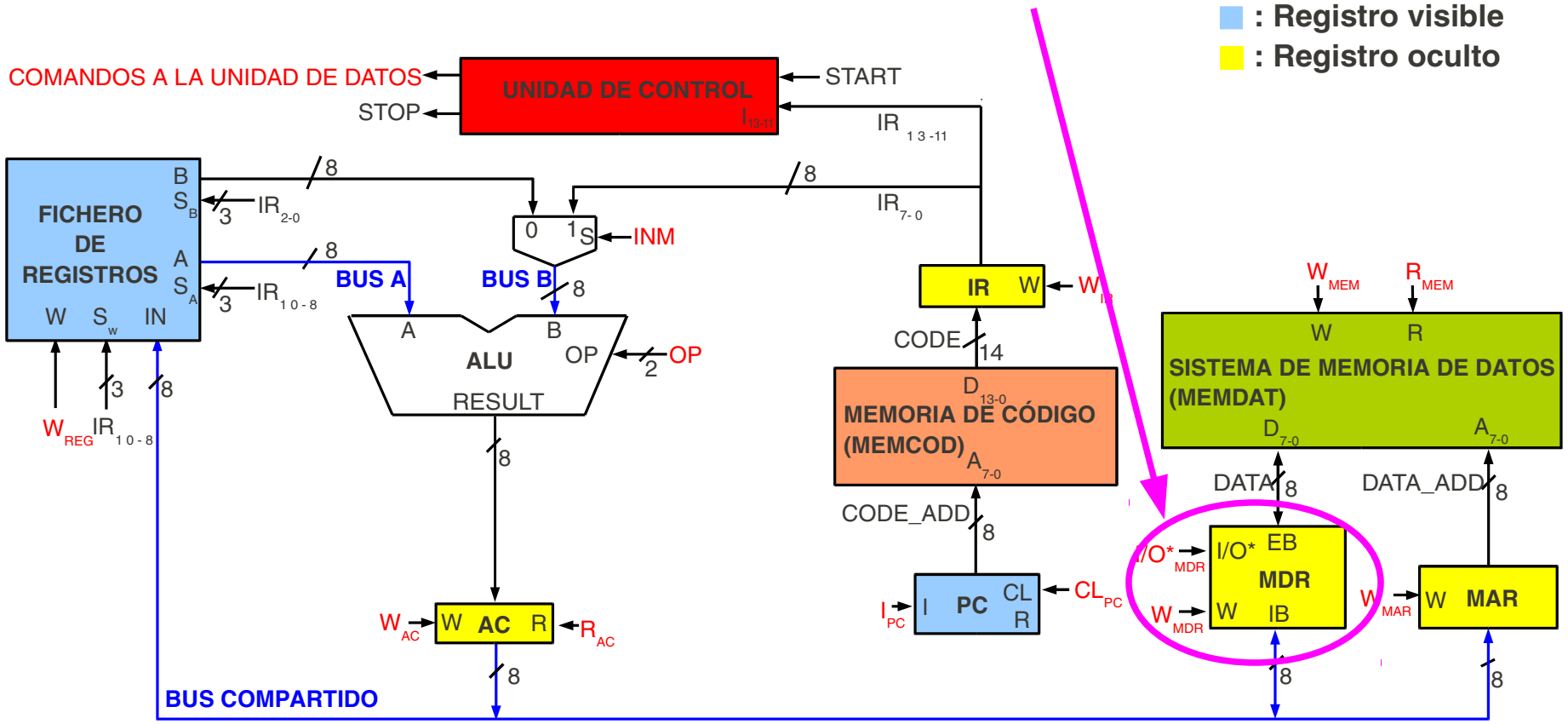
Memoria de lectura/escritura con capacidad 2^8x8 bits.
Terminal de datos bidireccional



Almacenamiento de datos

Implementación del CS2

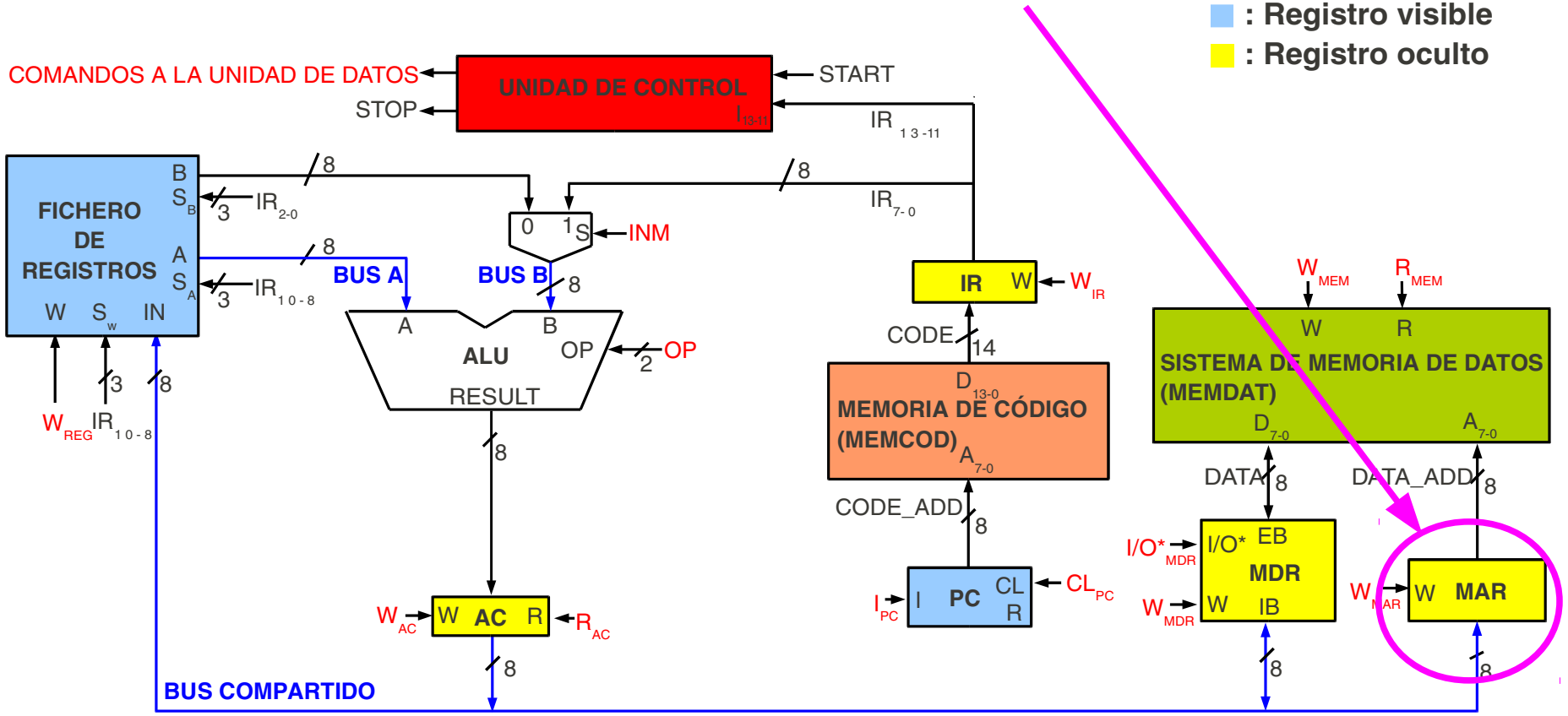
Registro que guarda de manera temporal los datos que van a intercambiarse con la memoria



Almacenamiento de datos

Implementación del CS2

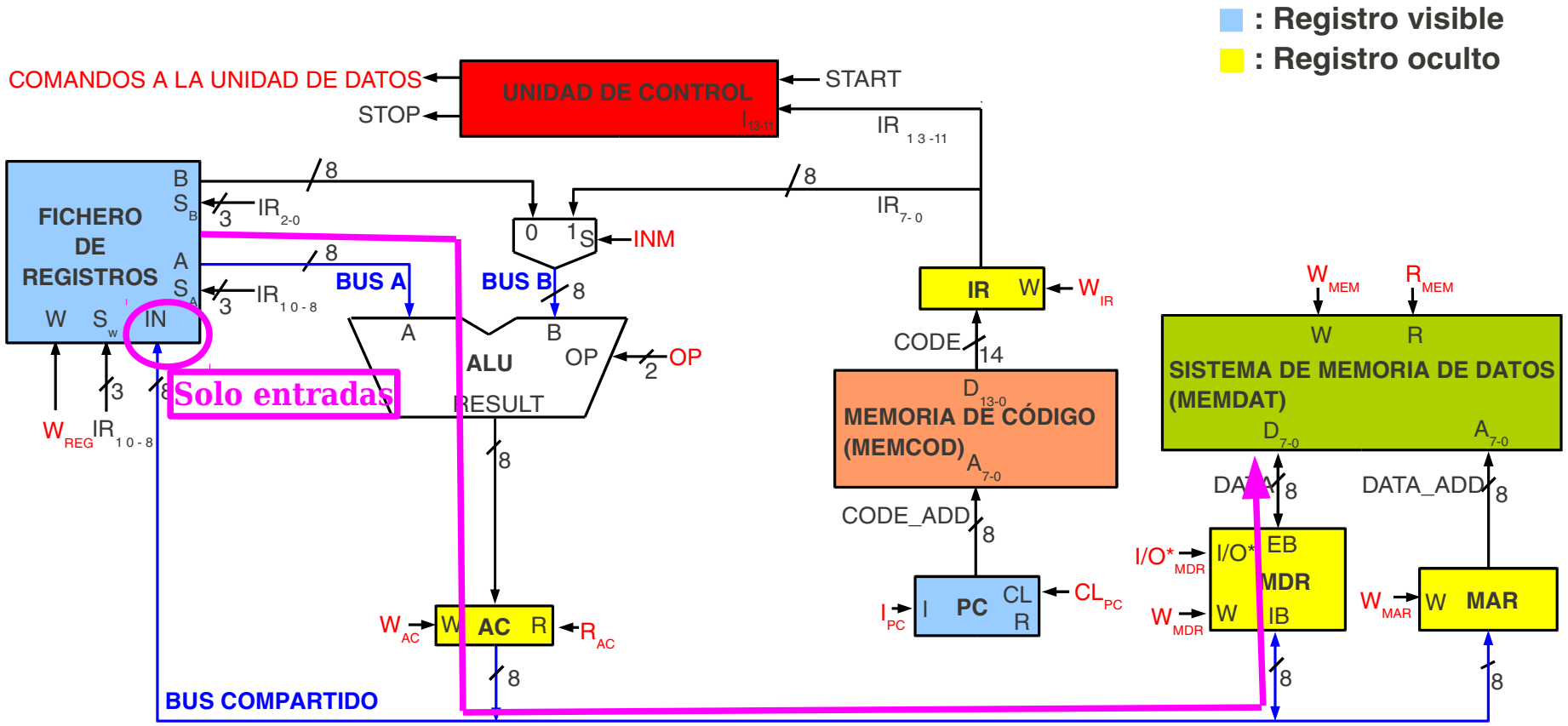
Registro que guarda de manera temporal las direcciones de las posiciones de la memoria de datos que serán accedidas.



Almacenamiento de datos

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

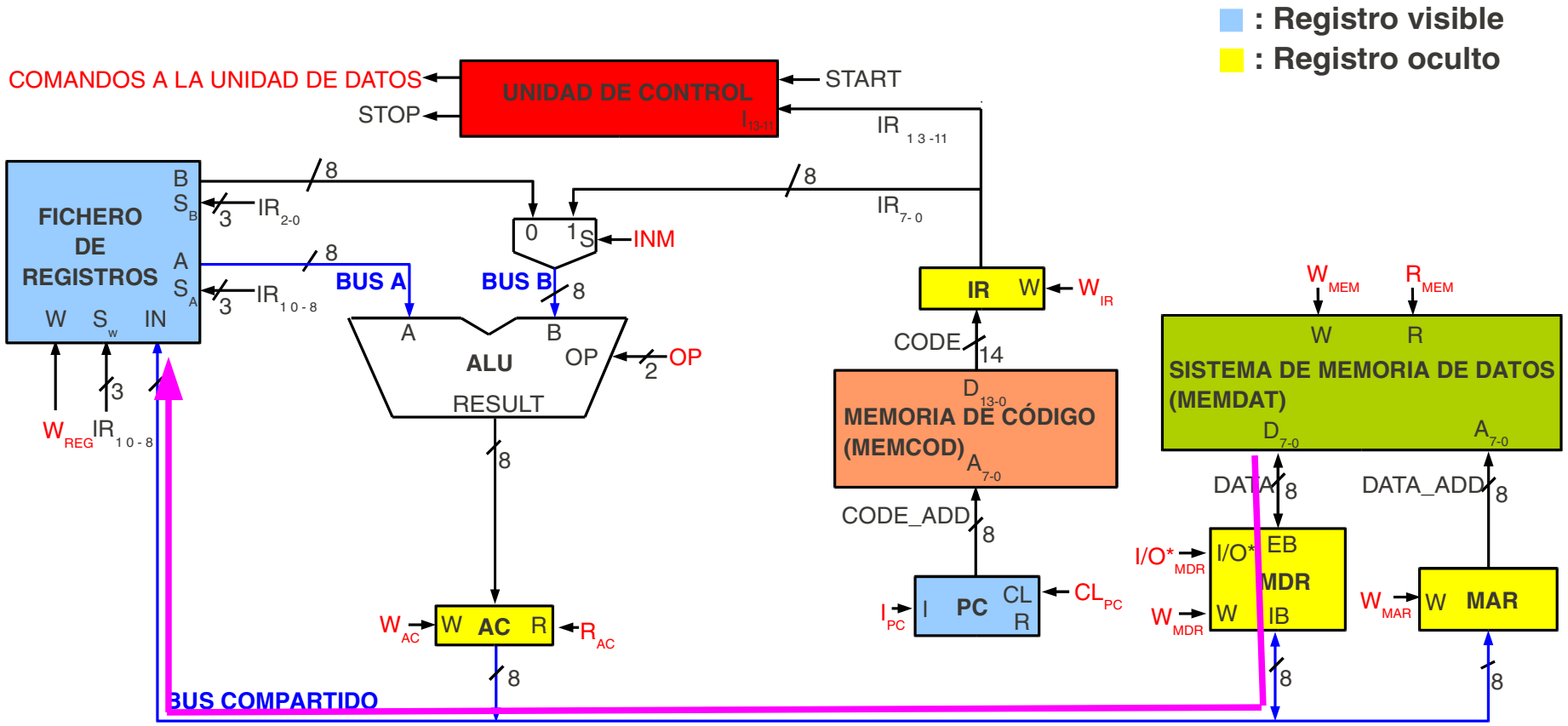
Implementación del CS2



Almacenamiento de datos

Implementación del CS2

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

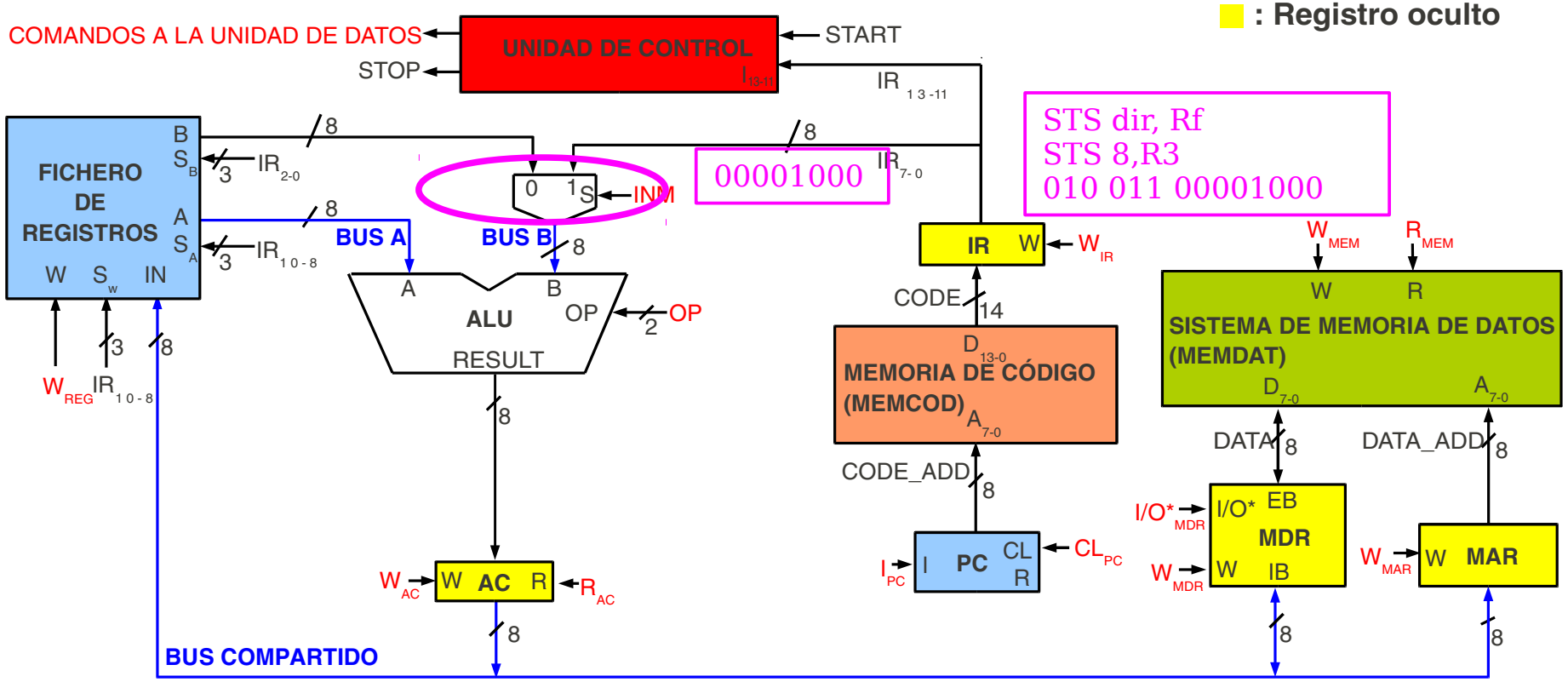


formato	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A	código de operación			registro destino (fuente en ST)			-	-	-	-	-	registro fuente (registro base en ST/LD)		
B							dirección del dato							

Implementación del CS2

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

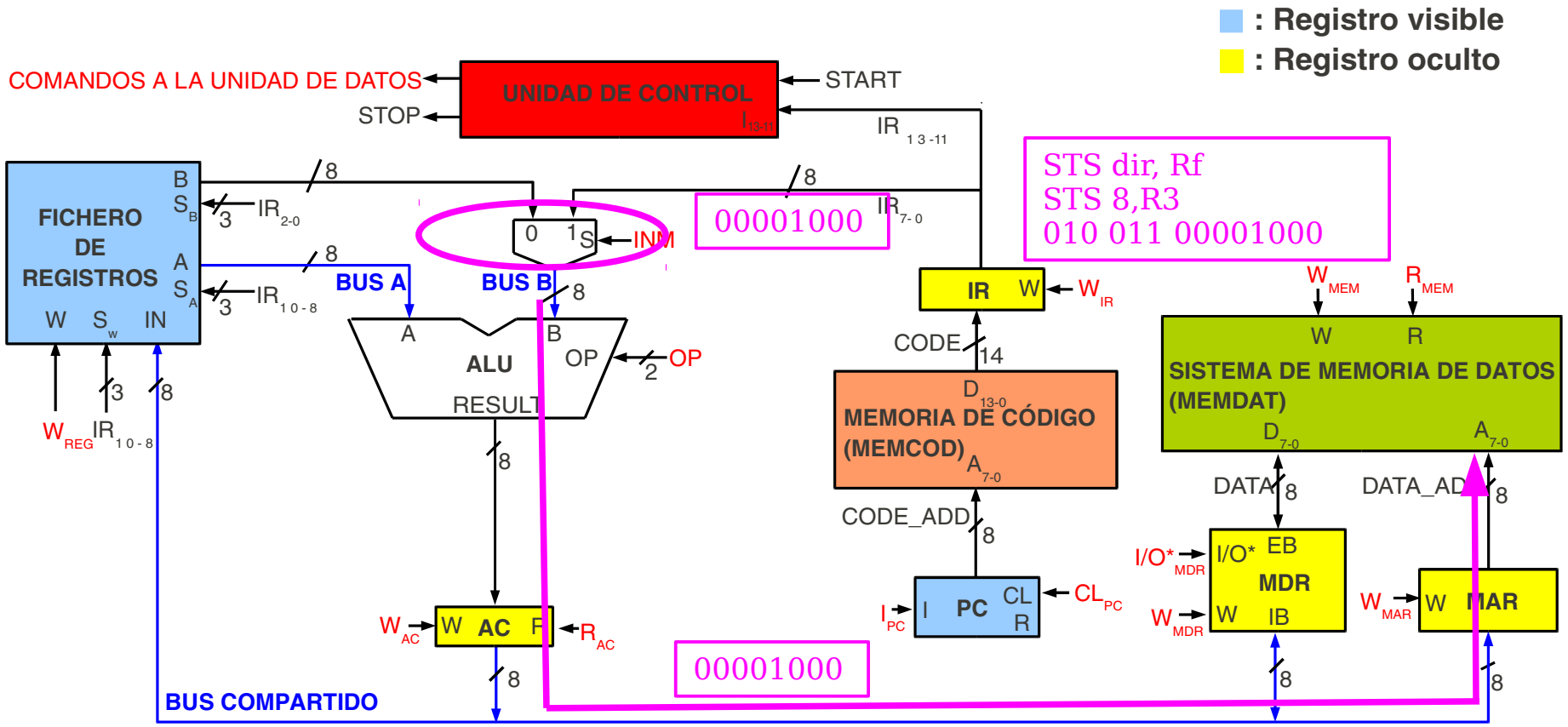
: Registro visible
: Registro oculto



formato	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



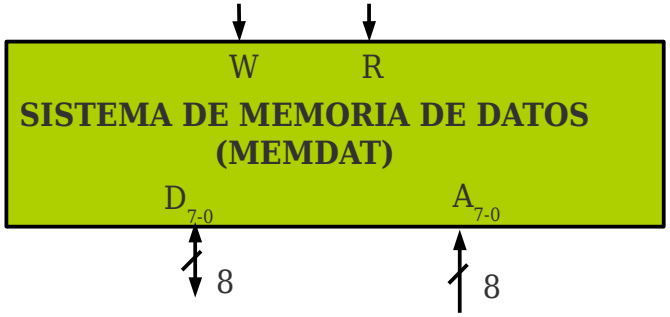
Almacenamiento de datos

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

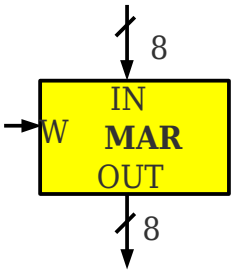
- Sistema de memoria de datos.
- Registro de datos de la memoria (MDR).
- Registro de direcciones de la memoria (MAR).
- Acumulador (AC). Se ha añadido este registro a la salida de la ALU porque esta debe compartir el bus.
- Multiplexor a la entrada de la ALU porque esta debe transferir datos tanto de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

Almacenamiento de datos

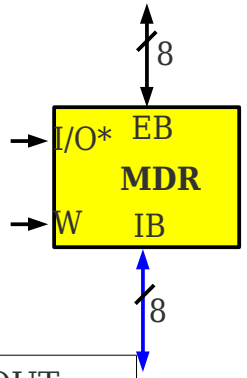
Descripción RT de los nuevos componentes



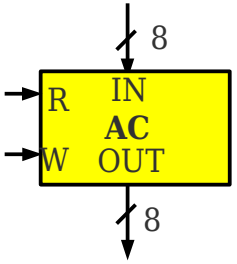
W R	MEMDAT[A_{7-0}] ←	D_{7-0} :=
00	MEMDAT[A_{7-0}]	H.I.
01	MEMDAT[A_{7-0}]	MEMDAT[A_{7-0}]
10	D_{7-0}	H.I.
11	PROHIBIDO	PROHIBIDO



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.

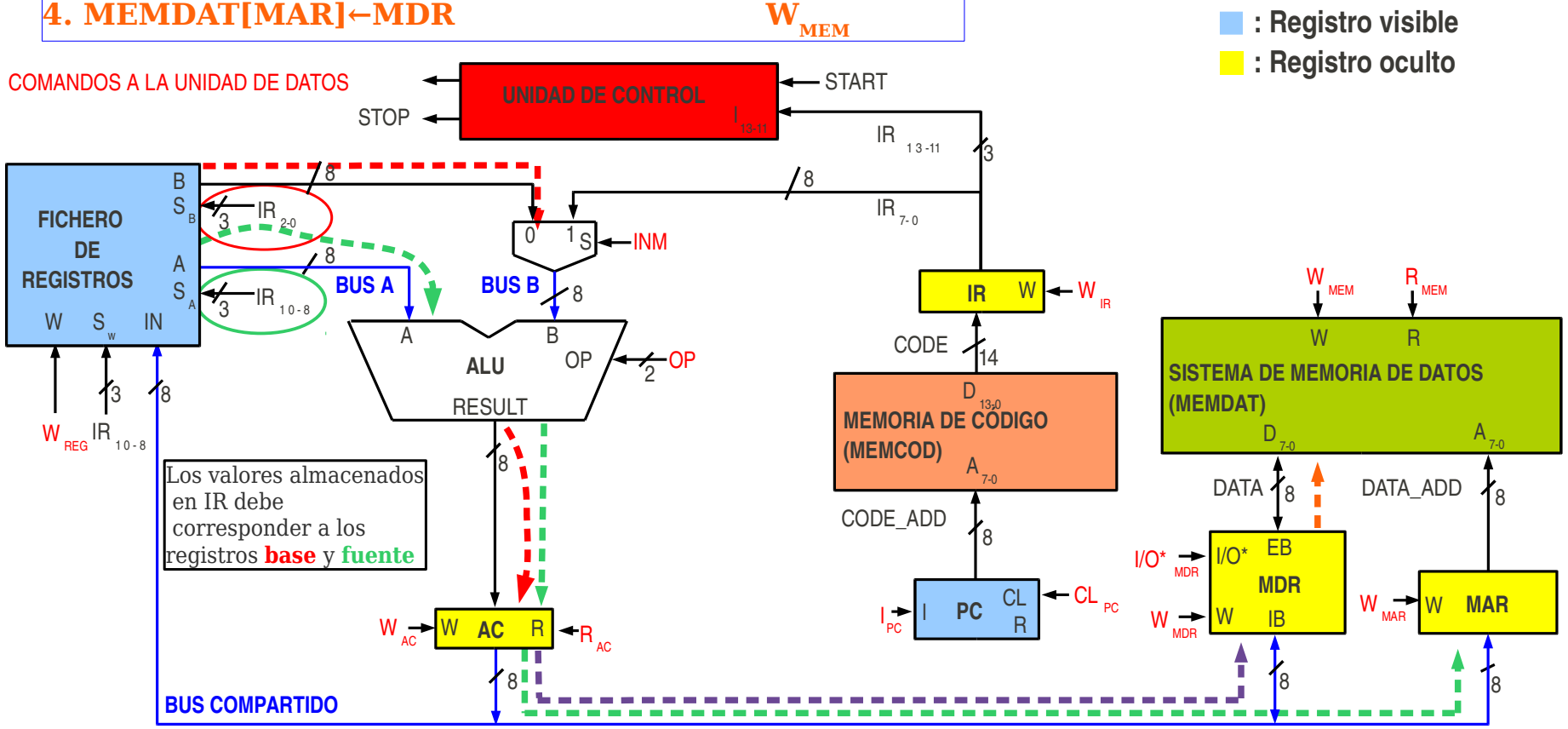


WR	AC ←	OUT:=
00	AC	H.I.
01	AC	AC
10	IN	H.I.
11	IN	AC

Almacenamiento de datos

ST (Rb),Rf

1. AC←REG[IR_{2-0}]	$OP_1 OP_0 W_{AC}$
2. MAR←AC; AC←REG[IR_{10-8}]	$R_{AC} W_{MAR} OP_0 W_{AC}$
3. MDR←AC	$R_{AC} W_{MDR}$
4. MEMDAT[MAR]←MDR	W_{MEM}



Almacenamiento de datos

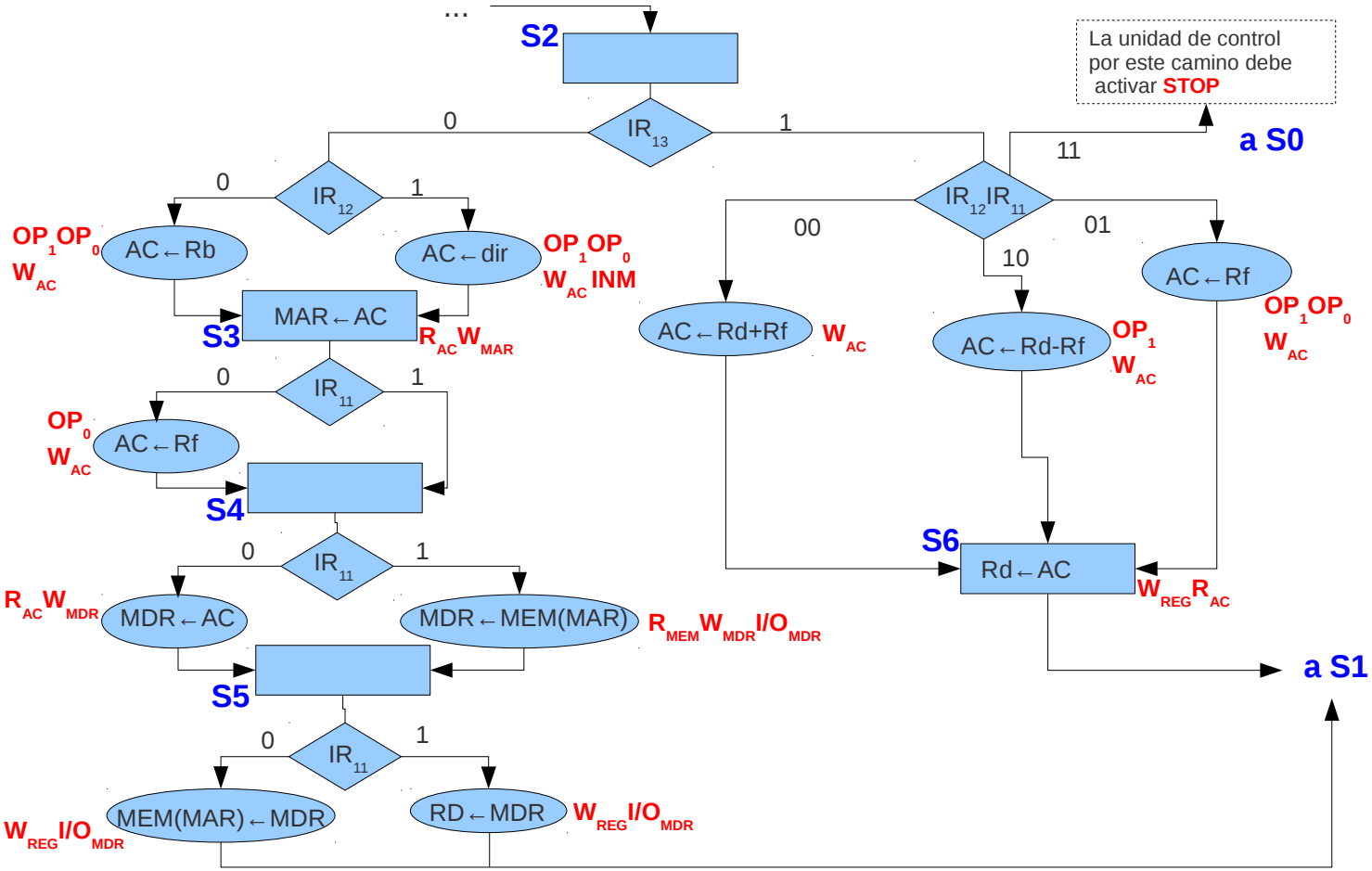
Descripción de las microoperaciones de las instrucciones añadidas

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←Rb $OP_1 OP_0 W_{AC}$	AC←Rb $OP_1 OP_0 W_{AC}$	AC←dir $OP_1 OP_0 W_{AC}$INM	AC←dir $OP_1 OP_0 W_{AC}$INM
Micro 2	MAR←AC $R_{AC} W_{MAR}$ AC←Rf $OP_0 W_{AC}$	MAR←AC $R_{AC} W_{MAR}$	MAR←AC $R_{AC} W_{MAR}$ AC←Rf $OP_0 W_{AC}$	MAR←AC $R_{AC} W_{MAR}$
Micro 3	MDR←AC $R_{AC} W_{MDR}$	MDR←MEM(MAR) $R_{MEM} W_{MDR} I/O^*_{MDR}$	MDR←AC $R_{AC} W_{MDR}$	MDR←MEM(MAR) $R_{MEM} W_{MDR} I/O^*_{MDR}$
Micro 4	MEM(MAR)←MDR W_{MEM}	RD←MDR $W_{REG} I/O^*_{MDR}$	MEM(MAR)←MDR W_{MEM}	RD←MDR $W_{REG} I/O^*_{MDR}$

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 R4,(R1)
ST (R1),R3
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)
ST (R1),R3
ST (R0),R4
ADD R0,R2
ADD R1,R2
LD R3,(R0)
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	DATO1TABLA1
R0+1	DATO2TABLA1
R0+2	DATO3TABLA1
R0+3…	DATO4TABLA1
R1	...
R1+1	DATO1TABLA2
R1+2	DATO2TABLA2
R1+3	DATO3TABLA2
...	DATO4TABLA2

MEMORIA DE DATOS