

 Institut Sabadell	Departamento de Informàtica		ASIR	M5 + M17	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador				
Daniel Martínez					Curso 2024 - 25

Fundamentos de Hardware y Sostenibilidad

BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador

 Institut Sabadell	Departamento de Informàtica		ASIR	M5 + M17	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador				
Daniel Martínez					Curso 2024 - 25

Índice

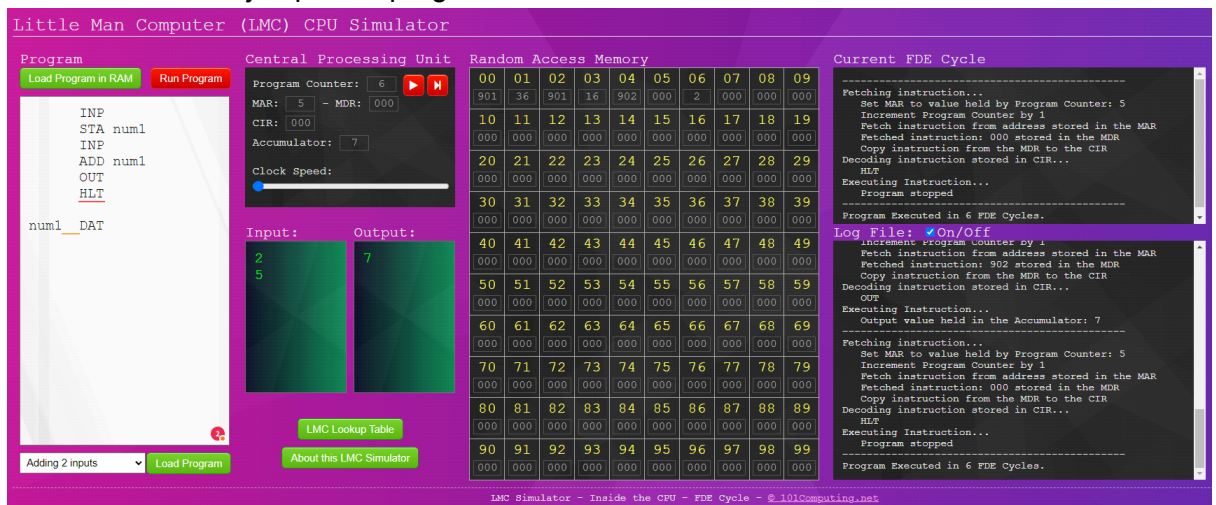
Ejercicio 1	2
Ejercicio 1a	2
Ejercicio 1b	3
Ejercicio 2	7
Ejercicio 3	8
Ejercicio 4	8
Ejercicio 5	9

Ejercicio 1

Para visualizar el funcionamiento interno del procesador, incluyendo la Unidad Aritmético-Lógica (UAL), los registros y el flujo de datos podemos utilizar simulador de procesadores online como, por ejemplo, [Little Man Computer \(LMC\)](#).

- a) Escribe instrucciones en código máquina (LOAD, STORE, ADD, cálculos matemáticos, etc.) para ver cómo el procesador gestiona los datos. Observa cómo el procesador accede a los registros, mueve datos entre la memoria y la UAL, y realiza cálculos.

Probamos con el ejemplo del programa de la suma de dos números:



Little Man Computer (LMC) CPU Simulator

Program

Load Program in RAM Run Program

```

INP
STA num1
INP
ADD num1
OUT
HLT
num1 DAT
  
```

Central Processing Unit

Program Counter: 5

MAR: 5 - MDR: 000

CIR: 000

Accumulator: 7

Clock Speed: [Slider]

Input: 2, 5

Output: 7

Random Access Memory

00	01	02	03	04	05	06	07	08	09
901	36	901	16	902	000	2	000	000	000
10	11	12	13	14	15	16	17	18	19
000	000	000	000	000	000	000	000	000	000
20	21	22	23	24	25	26	27	28	29
000	000	000	000	000	000	000	000	000	000
30	31	32	33	34	35	36	37	38	39
000	000	000	000	000	000	000	000	000	000
40	41	42	43	44	45	46	47	48	49
000	000	000	000	000	000	000	000	000	000
50	51	52	53	54	55	56	57	58	59
000	000	000	000	000	000	000	000	000	000
60	61	62	63	64	65	66	67	68	69
000	000	000	000	000	000	000	000	000	000
70	71	72	73	74	75	76	77	78	79
000	000	000	000	000	000	000	000	000	000
80	81	82	83	84	85	86	87	88	89
000	000	000	000	000	000	000	000	000	000
90	91	92	93	94	95	96	97	98	99
000	000	000	000	000	000	000	000	000	000



Current FDE Cycle

Log File: On/Off

```

Fetching instruction...
Set MAR to value held by Program Counter: 5
Increment Program Counter by 1
Fetch instruction from address stored in the MAR
Fetch instruction: 000 stored in the MDR
Copy instruction from the MDR to the CIR
Decoding instruction stored in CIR...
HLT
Executing instruction...
Program stopped
Program Executed in 6 FDE Cycles.
  
```

LMC Simulator - Inside the CPU - FDE Cycle - @101Computing.net

 Institut Sabadell	<i>Departamento de Informàtica</i>			<i>ASIR</i>	<i>M5 + M17</i>	 <div>Generalitat de Catalunya Departament d'Ensenyament</div>
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador					
Daniel Martínez						Curso 2024 - 25

b) Una vez ejecutado el programa, explica el proceso, identificando las operaciones principales del procesador.

Recuperamos el contador a 0 y lo establecemos como valor del registro de dirección de memoria (MAR), e incrementamos el contador del programa en 1. Ahora, obtenemos la instrucción almacenada en el registro de dirección de memoria (MAR). Ahora, la instrucción 901 que hemos encontrado, la almacenamos en el registro de datos de memoria (MDR). Lo copiamos al registro de instrucción actual (CIR), y lo decodifica. Detecta que es un input (INP), por lo que espera a la entrada del usuario, y almacena el número dado (1) en el acumulador:

```

-----
Fetching instruction...
  Set MAR to value held by Program Counter: 0
  Increment Program Counter by 1
  Fetch instruction from address stored in the MAR
  Fetched instruction: 901 stored in the MDR
  Copy instruction from the MDR to the CIR
Decoding instruction stored in CIR...
  INP
Executing Instruction...
  Waiting for user input
  Store user input in Accumulator: 1
-----

```

Recuperamos el contador a 0 y lo establecemos como valor del registro de dirección de memoria (MAR), e incrementamos el contador del programa en 1. Ahora, obtenemos la instrucción almacenada en el registro de dirección de memoria (MAR). Ahora, la instrucción 36 que hemos encontrado, la almacenamos en el registro de datos de memoria (MDR). Lo copiamos al registro de instrucción actual (CIR), y lo decodifica. Detecta que es un store accumulator (STA), por lo que establece el registro de dirección de memoria (MAR) al que siguiente libre (6). Establece el registro de datos de memoria (MDR) el valor del acumulador (1), y almacena el valor del registro de datos de memoria (MDR) en la localización del registro de dirección de memoria (MAR), que en este caso es el 6:

```

-----
Fetching instruction...
  Set MAR to value held by Program Counter: 1
  Increment Program Counter by 1
  Fetch instruction from address stored in the MAR
  Fetched instruction: 36 stored in the MDR
  Copy instruction from the MDR to the CIR
Decoding instruction stored in CIR...
  STA
Executing Instruction...
  Set MAR to the operand of the current instruction: 6
  Set MDR to the value held in the Accumulator: 1
  Store MDR value 1 at the memory location held in the MAR: 6
-----

```


 Institut Sabadell	Departamento de Informàtica		ASIR	M5 + M17	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador				
Daniel Martínez					Curso 2024 - 25

Recuperamos el contador a 2 y lo establecemos como valor del registro de dirección de memoria (MAR), e incrementamos el contador del programa en 1. Ahora, obtenemos la instrucción almacenada en el registro de dirección de memoria (MAR). Ahora, la instrucción 901 que hemos encontrado, la almacenamos en el registro de datos de memoria (MDR). Lo copiamos al registro de instrucción actual (CIR), y lo decodifica. Detecta que es un input (INP), por lo que espera a la entrada del usuario, y almacena el número dado (2) en el acumulador:

```

-----
Fetching instruction...
  Set MAR to value held by Program Counter: 2
  Increment Program Counter by 1
  Fetch instruction from address stored in the MAR
  Fetched instruction: 901 stored in the MDR
  Copy instruction from the MDR to the CIR
Decoding instruction stored in CIR...
  INP
Executing Instruction...
  Waiting for user input
  Store user input in Accumulator: 2
-----

```

 Institut Sabadell	<i>Departamento de Informàtica</i>			<i>ASIR</i>	<i>M5 + M17</i>	 <div>Generalitat de Catalunya Departament d'Ensenyament</div>
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador					
Daniel Martínez						Curso 2024 - 25

Recuperamos el contador a 3 y lo establecemos como valor del registro de dirección de memoria (MAR), e incrementamos el contador del programa en 1. Ahora, obtenemos la instrucción almacenada en el registro de dirección de memoria (MAR). Ahora, la instrucción 16 que hemos encontrado, la almacenamos en el registro de datos de memoria (MDR). Lo copiamos al registro de instrucción actual (CIR), y lo decodifica. Detecta que es una adición (ADD), por lo que establece el registro de dirección de memoria (MAR) al que siguiente libre (6). Obtenemos los datos en la ubicación del registro de dirección de memoria (MAR) y se almacenan en el registro de datos de memoria (MDR) el valor que tenía guardado (1). Añade el registro de datos de la memoria (MDR) al acumulador y almacena el resultado de la suma en el acumulador ($1+2=3$)

```

-----
Fetching instruction...
  Set MAR to value held by Program Counter: 3
  Increment Program Counter by 1
  Fetch instruction from address stored in the MAR
  Fetched instruction: 16 stored in the MDR
  Copy instruction from the MDR to the CIR
Decoding instruction stored in CIR...
  ADD
Executing Instruction...
  Set MAR to the operand of the current instruction: 6
  Fetch data at the location held by the MAR and store it in the MDR:
1
  Add MDR value to the Accumulator and store the result in the
Accumulator: 1+2=3
-----

```

Recuperamos el contador a 4 y lo establecemos como valor del registro de dirección de memoria (MAR), e incrementamos el contador del programa en 1. Ahora, obtenemos la instrucción almacenada en el registro de dirección de memoria (MAR). Ahora, la instrucción 902 que hemos encontrado, la almacenamos en el registro de datos de memoria (MDR). Lo copiamos al registro de instrucción actual (CIR), y lo decodifica. Detecta que es un output (OUT), por lo que imprime como salida el valor del acumulador (3):

```

-----
Fetching instruction...
  Set MAR to value held by Program Counter: 4
  Increment Program Counter by 1
  Fetch instruction from address stored in the MAR
  Fetched instruction: 902 stored in the MDR
  Copy instruction from the MDR to the CIR
Decoding instruction stored in CIR...
  OUT
Executing Instruction...
  Output value held in the Accumulator: 3
-----

```

 Institut Sabadell	Departamento de Informàtica		ASIR	M5 + M17	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador				
Daniel Martínez					Curso 2024 - 25

Recuperamos el contador a 5 y lo establecemos como valor del registro de dirección de memoria (MAR), e incrementamos el contador del programa en 1. Ahora, obtenemos la instrucción almacenada en el registro de dirección de memoria (MAR). Ahora, la instrucción 000 que hemos encontrado, la almacenamos en el registro de datos de memoria (MDR). Lo copiamos al registro de instrucción actual (CIR), y lo decodifica. Detecta que es un halt (HLT), por lo que finaliza el programa y nos muestra que se ha ejecutado en 6 ciclos FDE (Fetch, Decode, Execute):

```

-----
Fetching instruction...
  Set MAR to value held by Program Counter: 5
  Increment Program Counter by 1
  Fetch instruction from address stored in the MAR
  Fetched instruction: 000 stored in the MDR
  Copy instruction from the MDR to the CIR
Decoding instruction stored in CIR...
  HLT
Executing Instruction...
  Program stopped
-----
Program Executed in 6 FDE Cycles.

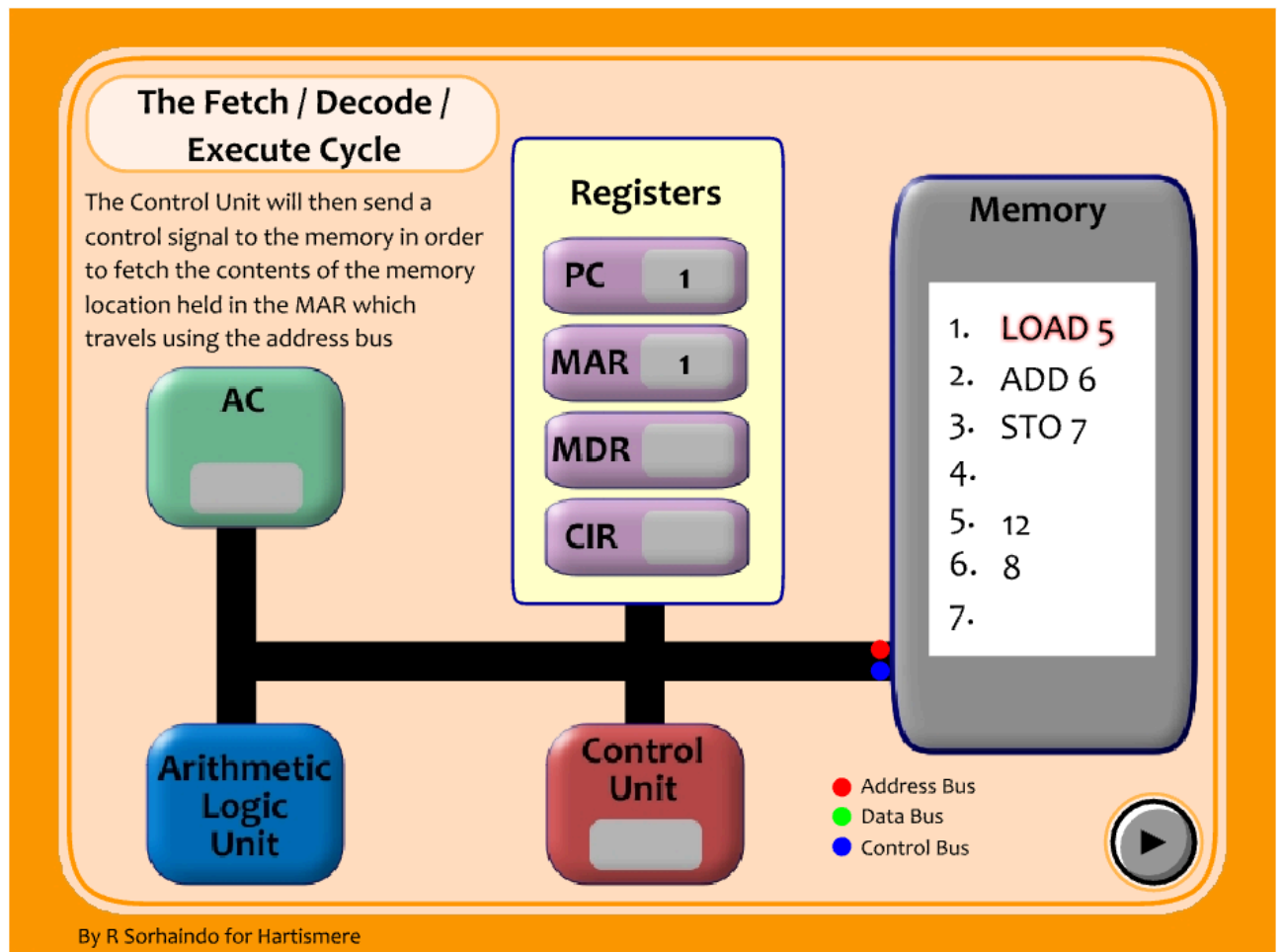
```


 Institut Sabadell	Departamento de Informàtica			ASIR	M5 + M17	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador					
Daniel Martínez						Curso 2024 - 25

Ejercicio 2

Para comprender cómo la Unidad de Control (UC) dirige el flujo de datos e instrucciones dentro del procesador, podemos utilizar un [simulador](#) de CPU animado o recursos online que muestran visualmente el ciclo de instrucción.

Abrimos el recurso:



 Institut Sabadell	<i>Departamento de Informàtica</i>			<i>ASIR</i>	<i>M5 + M17</i>	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador					
Daniel Martínez						Curso 2024 - 25

Ejercicio 3



Analiza cómo la Unidad de Control gestiona paso a paso las instrucciones (fetch, decode, execute) y cómo dirige el flujo de datos al UAL o la memoria.

- **Fetch:** El flujo de datos va desde la memoria a través del registro de posición de memoria (MAR) y el registro de datos de memoria (MDR) hacia el registro de instrucción actual (CIR).
- **Decode:** La unidad de control (UC) analiza la instrucción en el registro de instrucción actual (CIR) y dirige el flujo de datos hacia la memoria, a través del registro de posición de memoria (MAR), para preparar la ejecución.
- **Execute:** Los datos fluyen desde la memoria, a través del registro de datos de memoria (MDR) y el acumulador (AC) hacia la unidad aritmético lógica (ALU) para la operación, y el resultado regresa al acumulador (AC).

Ejercicio 4

Explica las diferentes etapas del proceso y cómo se coordinan para realizar una instrucción simple.

- **Fetch:** La unidad de control comienza cargando la dirección de la siguiente instrucción desde el contador de programa. Accede a la memoria usando esa dirección, recupera la instrucción y actualiza el contador de programa para señalar la próxima instrucción a ejecutar. Traducido sería la fase de obtención.
- **Decode:** Una vez obtenida la instrucción, la unidad de control la interpreta, descomponiéndola. Determina el tipo de operación a realizar, preparando los datos necesarios, que están en la memoria. Traducido sería la fase de decodificación.
- **Execute:** La unidad de control envía las señales necesarias para ejecutar la operación en la Unidad Aritmético Lógica (ALU) o en otros componentes. Después de que la ALU realiza el cálculo, la unidad de control gestiona el almacenamiento del resultado. Traducido sería la fase de ejecución.

 Institut Sabadell	Departamento de Informàtica		ASIR	M5 + M17	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador				
Daniel Martínez					Curso 2024 - 25

Ejercicio 5

Ver en detalle cada paso del ciclo de instrucción del procesador (fetch, decode, execute).

Paso 1:

- **Fetch:** El contador (PC) apunta a la dirección 1 (Posición de memoria que contiene la instrucción LOAD 5). La dirección 1 se envía al registro de posición de memoria (MAR), y la instrucción LOAD 5 se transfiere de la memoria al registro de datos de memoria (MDR). La instrucción se copia del MDR al registro de instrucción actual (CIR). El contador (PC) se incrementa para apuntar a la siguiente instrucción en la posición 2.
- **Decode:** La unidad de control (UC) decodifica la instrucción LOAD 5, identificando que debe cargar el valor de la dirección 5 en el acumulador (AC). La unidad de control (UC) prepara el registro de posición de memoria (MAR) con la dirección 5.
- **Execute:** El valor en la dirección 5 de la memoria es 12. La unidad de control (UC) transfiere este valor desde la memoria al registro de datos de memoria (MDR). El valor 12 se mueve del registro de datos de memoria (MDR) al acumulador (AC), almacenando el valor en el acumulador.

Cómo resultado, el acumulador ahora contiene el valor 12.

Paso 2:

- **Fetch:** El contador (PC) ahora apunta a la dirección 2, donde se encuentra la instrucción ADD 6. La dirección se envía al registro de posición de memoria (MAR), y la instrucción ADD 6 se transfiere de la memoria al registro de datos de memoria (MDR), luego se copia al registro de instrucción actual (CIR). El PC se incrementa para apuntar a la siguiente instrucción en la posición 3.
- **Decode:** La unidad de control (UC) decodifica la instrucción ADD 6, identificando que debe sumar el valor almacenado en la dirección 6 al valor del acumulador (AC). La unidad de control (UC) prepara el registro de posición de memoria (MAR) con la dirección 6.
- **Execute:** El valor en la dirección 6 de la memoria es 8. La unidad de control (UC) transfiere este valor desde la memoria al registro de datos de memoria (MDR). El valor 8 se suma al valor actual del acumulador (AC), que es 12, lo que da como resultado 20. El acumulador (AC) se actualiza con el valor 20.

Cómo resultado, el acumulador ahora contiene el valor 20.

 Institut Sabadell	Departamento de Informàtica		ASIR	M5 + M17	 Generalitat de Catalunya Departament d'Ensenyament
	BA1 - RA1 - Actividad 2 - Funcionamiento de un Procesador				
Daniel Martínez					Curso 2024 - 25

Paso 3:

- **Fetch:** El contador (PC) apunta a la dirección 3, donde se encuentra la instrucción STO 7. La dirección se envía al registro de posición de memoria (MAR), y la instrucción STO 7 se transfiere al registro de datos de memoria (MDR) y luego al registro de instrucción actual (CIR). El contador (PC) se incrementa para finalizar el programa.
- **Decode:** La unidad de control (UC) decodifica la instrucción STO 7, identificando que debe almacenar el valor del acumulador, que es 20, en la dirección 7 de la memoria. La unidad de control (UC) prepara el registro de posición de memoria (MAR) con la dirección 7.
- **Execute:** El valor 20 del acumulador (AC) se transfiere al registro de datos de memoria (MDR) y luego a la dirección de memoria 7.

Cómo resultado, la dirección 7 de la memoria ahora contiene el valor 20.