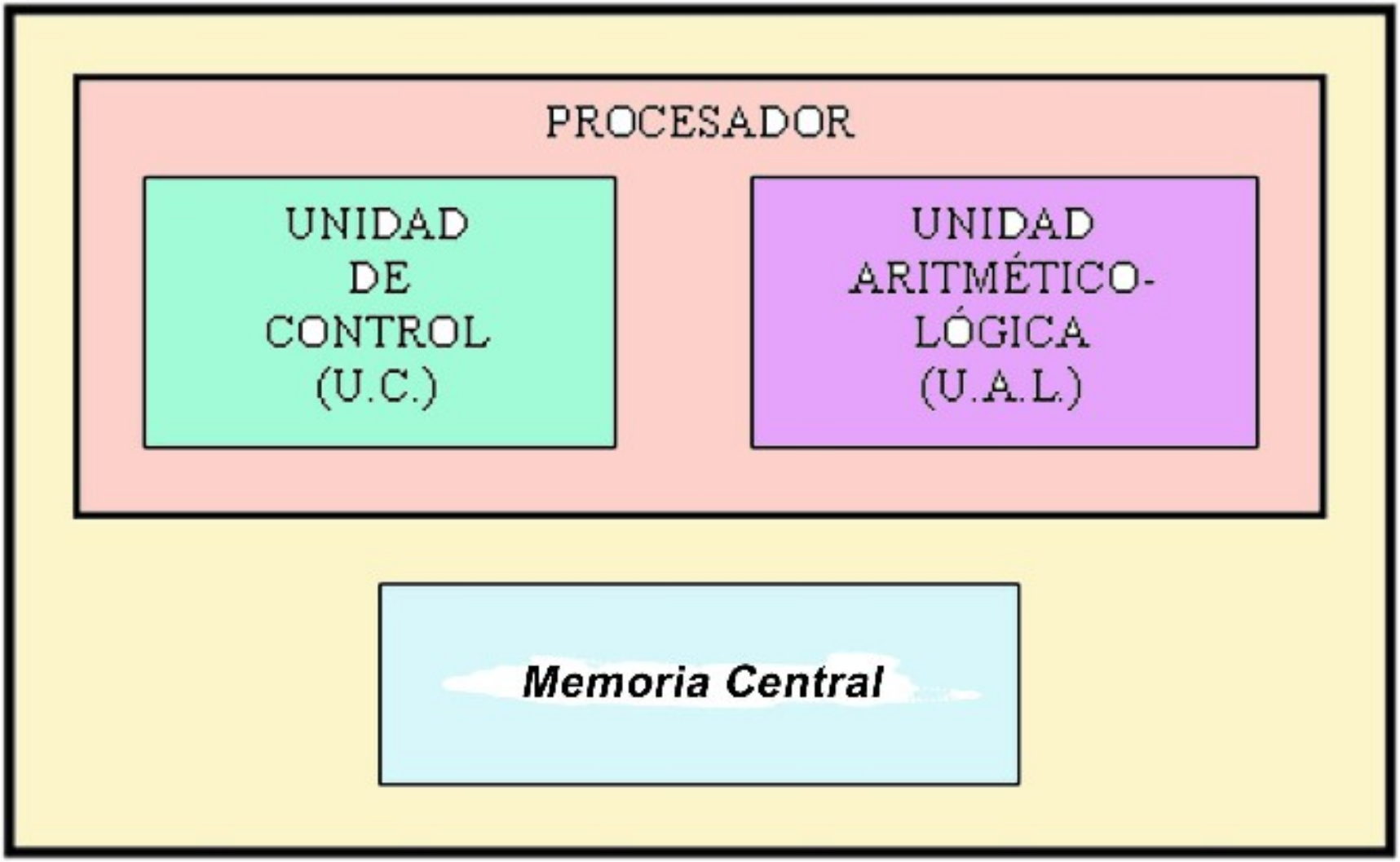
SEGUIMIENTO DE UNA INSTRUCCIÓN EN UNA CPU

- Vamos a simular a los grandes fabricantes de CPUs:
  - INTEL
  - MOTOROLA

- Los dos se basan en el modelo de Von Neumann:
  - El programa y los datos están almacenados en la memoria del ordenador.
- Crearemos la CPU BLUME

- La CPU BLUME está desarrollada como una máquina de Von Newmann:



- La CPU BLUME tendrá las siguientes **instrucciones**:

1ª) 001 → Borrar registro AX

2ª) 010 → Borrar registro BX

3ª) 011 → Suma AX + BX y lo deja en AX

4ª) 100 → Cambia contenido de AX y BX

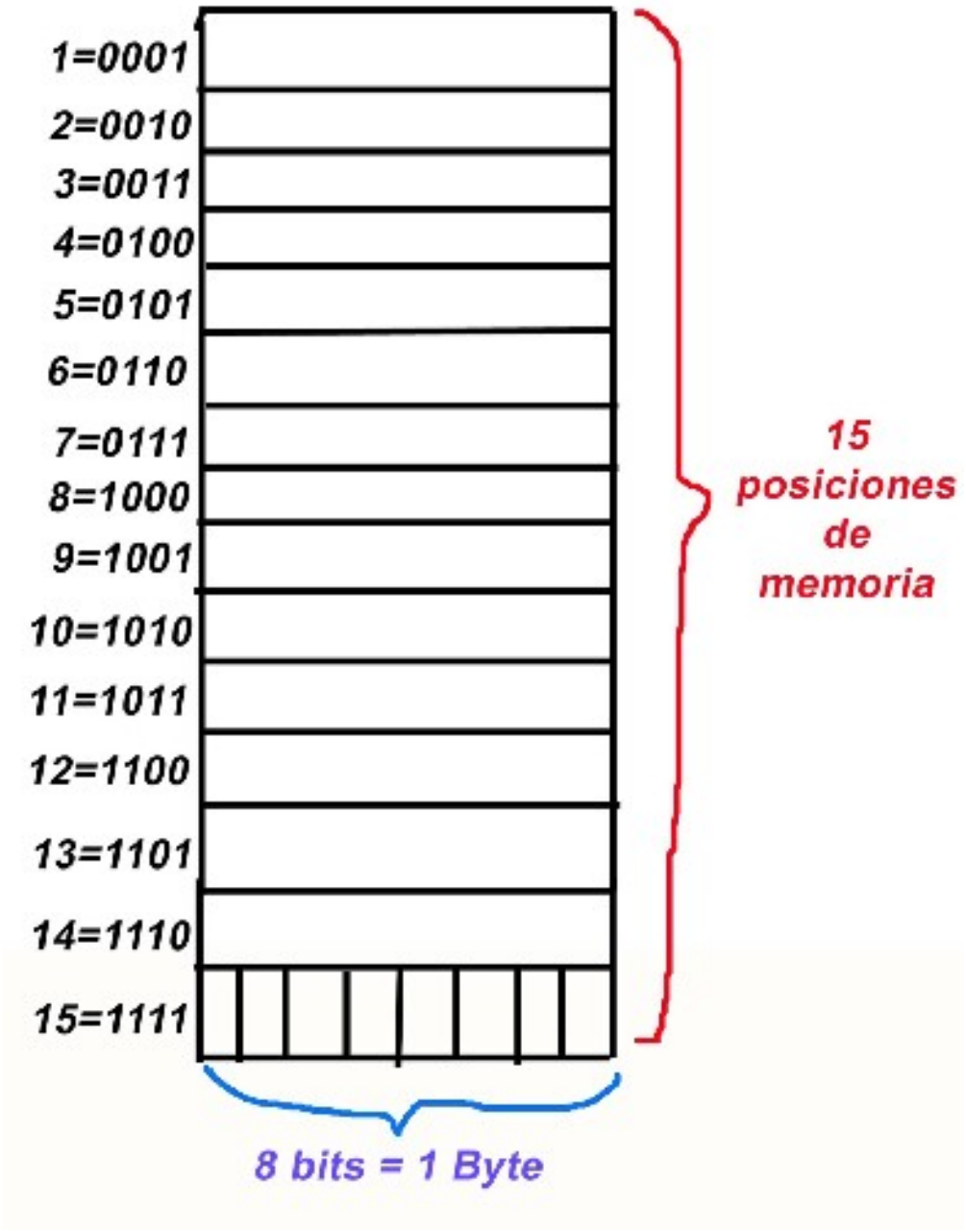
5ª) 101 → Suma AX + Contenido de una posición de memoria y el resultado lo deja en AX

6ª) 110 → Salta a la posición de memoria que viene a continuación

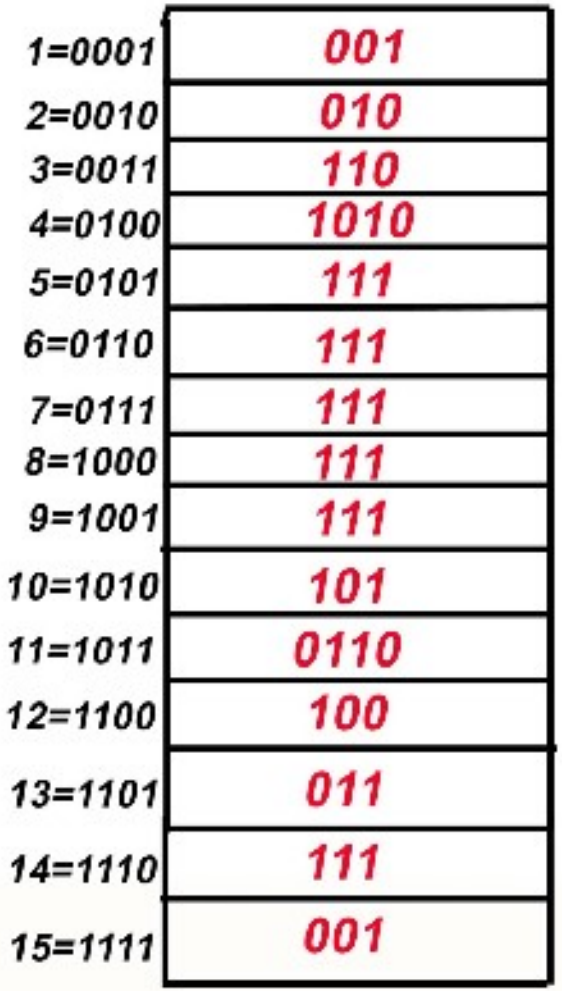
7ª) 111 → Fin de programa

- **LA MEMORIA** de nuestra CPU BLUME contiene:

- 15 posiciones de memoria.
- En cada posición de memoria cabe 1 Byte



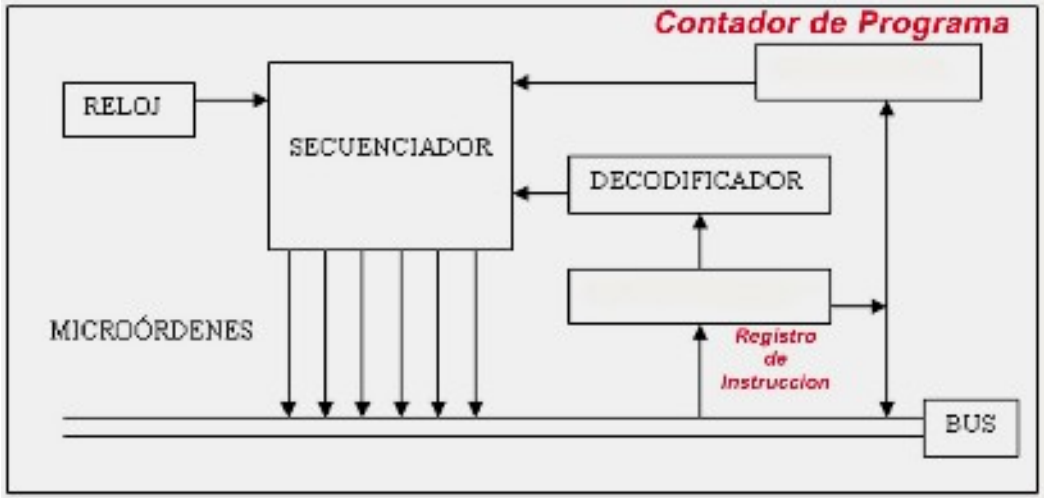
- Cargamos **LA MEMORIA** de la CPU BLUME con el siguiente contenido:



Y le damos al botón de Empezar la ejecución de nuestra CPU

Nos fijaremos en que pasa en la unidad de control.

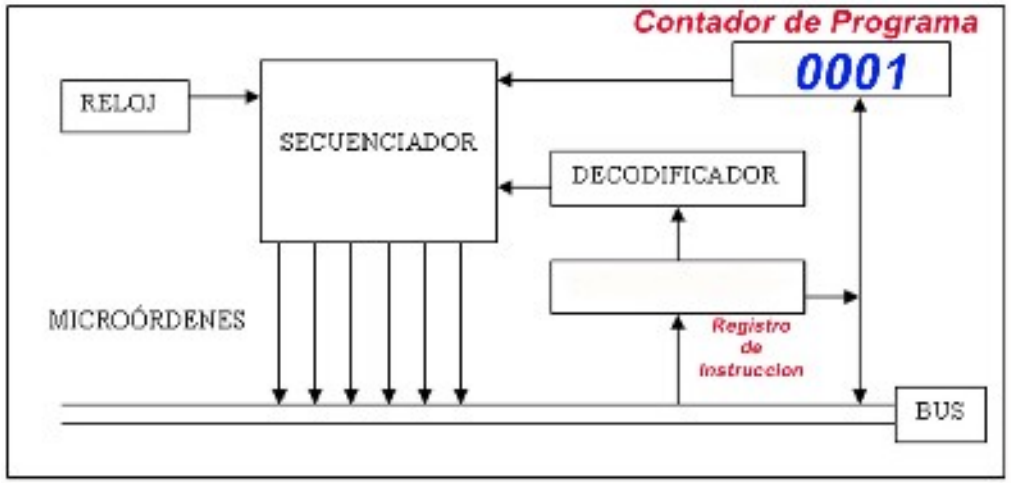
Aquí tenemos la Unidad de Control y la Memoria Central cargada



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

Primer ciclo de Reloj:

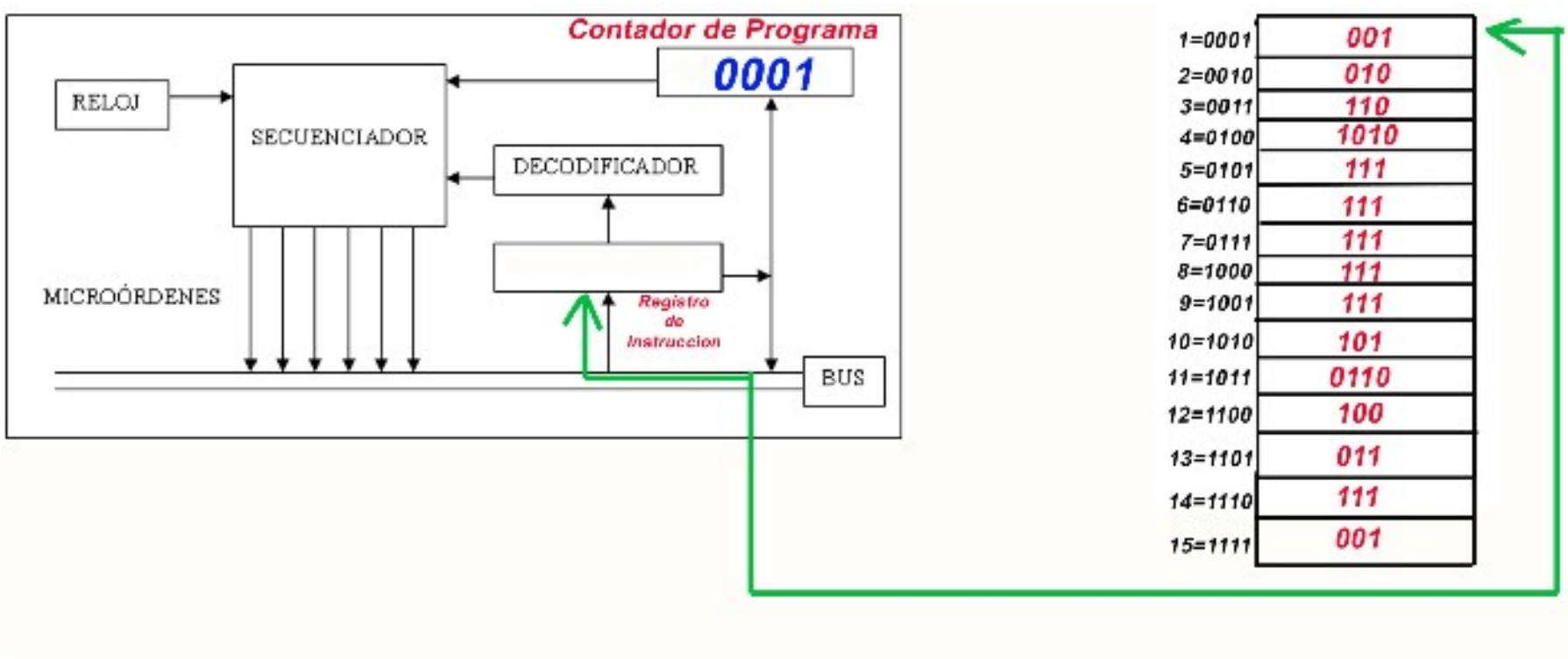
- El contador de programa contiene **0001**



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

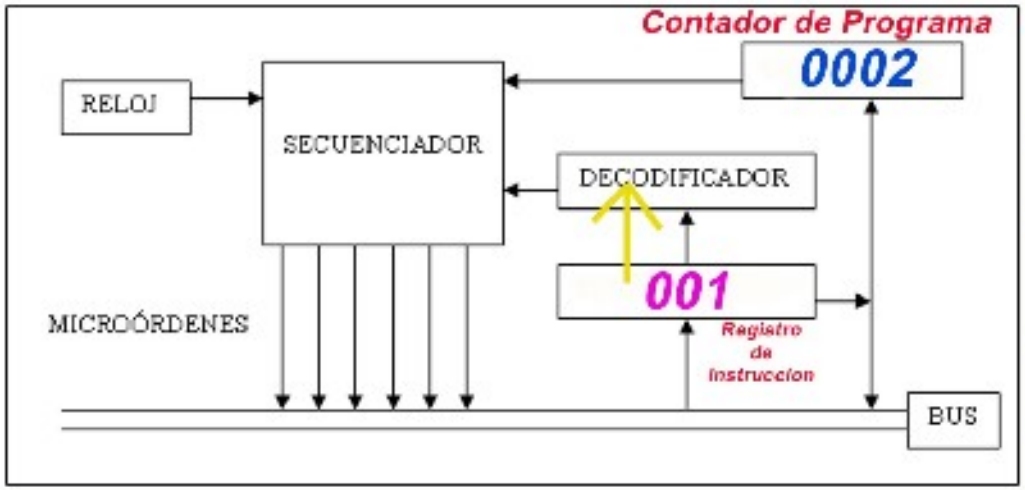
Primer ciclo de Reloj:

- El contador de programa contiene **0001**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción:



Segundo ciclo de Reloj:

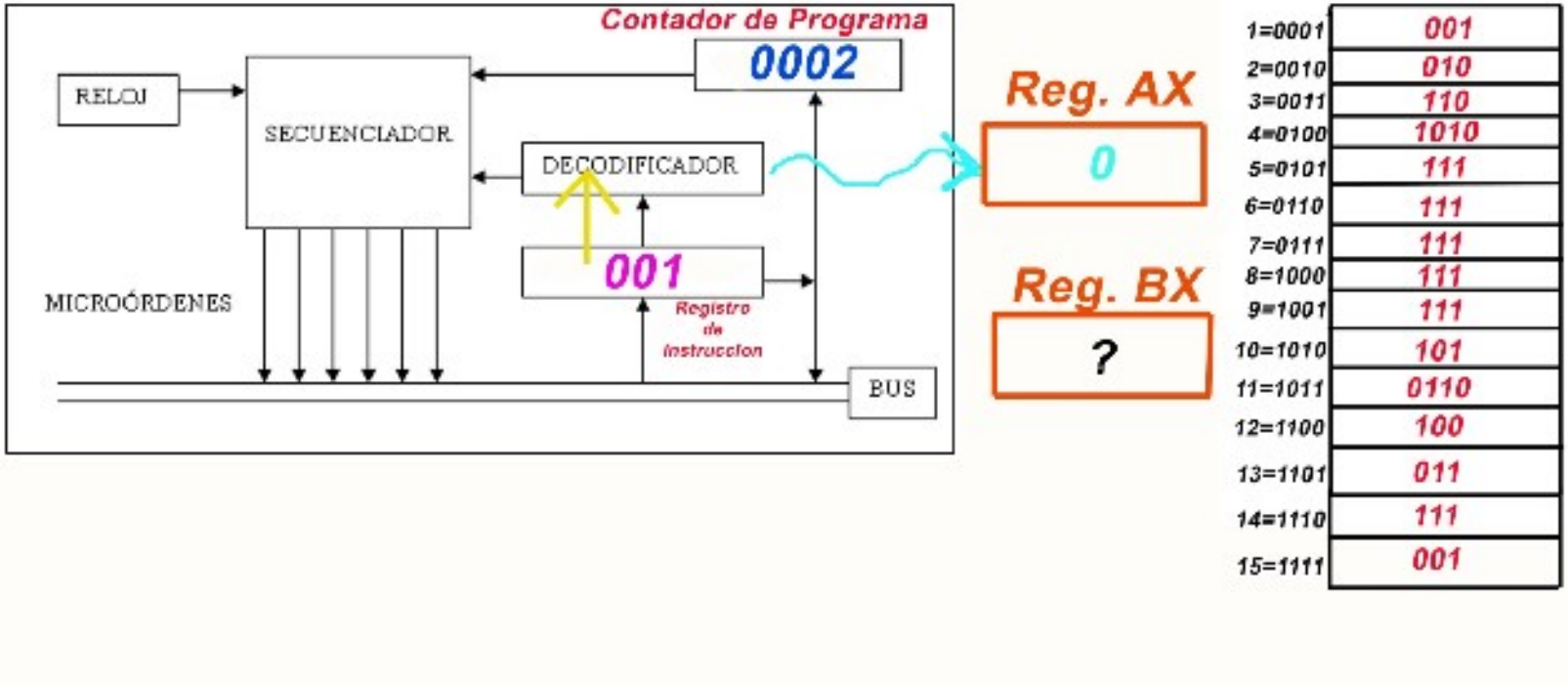
- El contador de programa se incremente automáticamente. Tenia **0001 y pasa a tener 0002.**
- Lo que hay en el Registro de instrucción, pasa a ser evaluado por el **Decodificador**.
  - El decodificador sabe que lo que hay es una instrucción, la 001.
- **Los Registros AX y BX pueden tener cualquier cosa.**



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

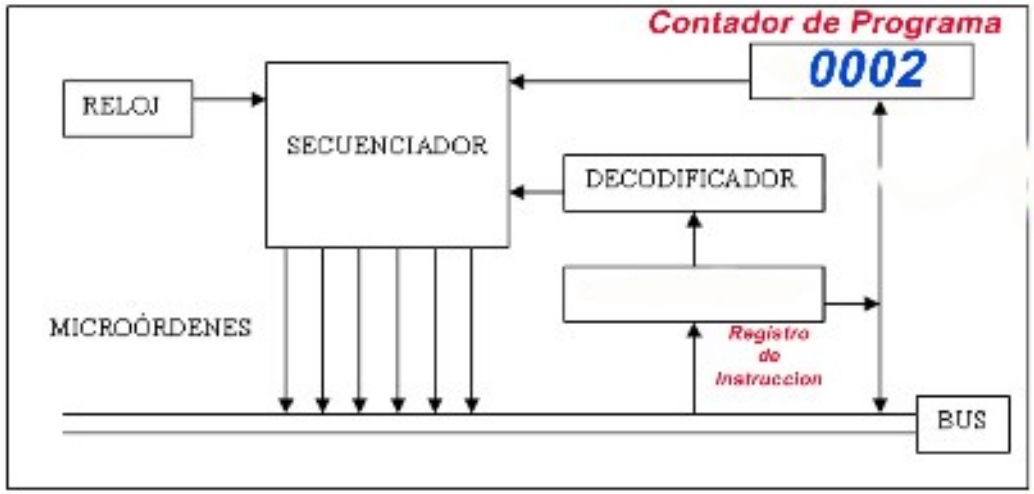
Segundo ciclo de Reloj:

- El decodificador sabe que la instrucción 001 es:
  - 1 → 001 →Borrar registro AX
- **El decodificador** se encarga directamente de cargar un 0 en el Registro AX
- Aquí ha acabado la primera instrucción.



- La instrucción: 1 → 001 →Borrar registro AX
- Ha sido ejecutada en 2 ciclos de Reloj.

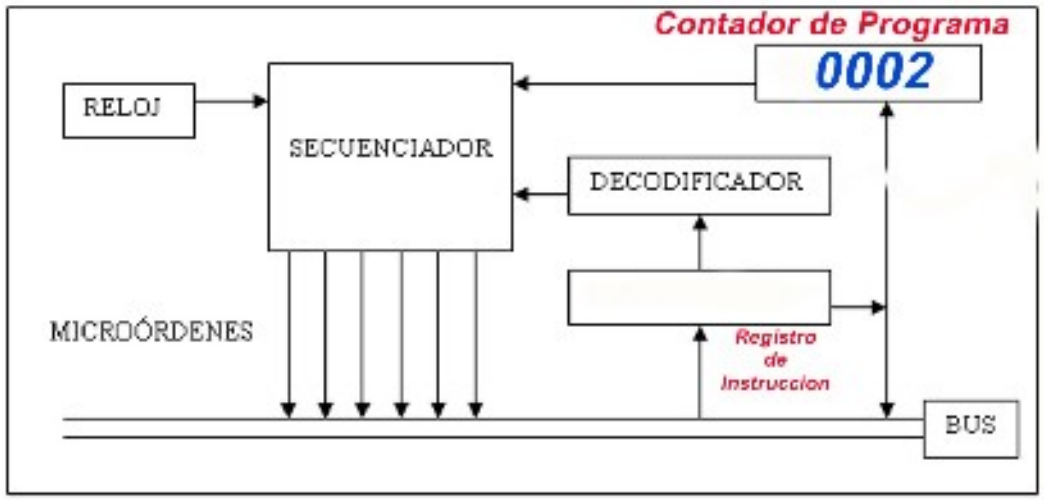
- La CPU sigue ejecutando la siguiente instrucción.



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

Tercer ciclo de Reloj:

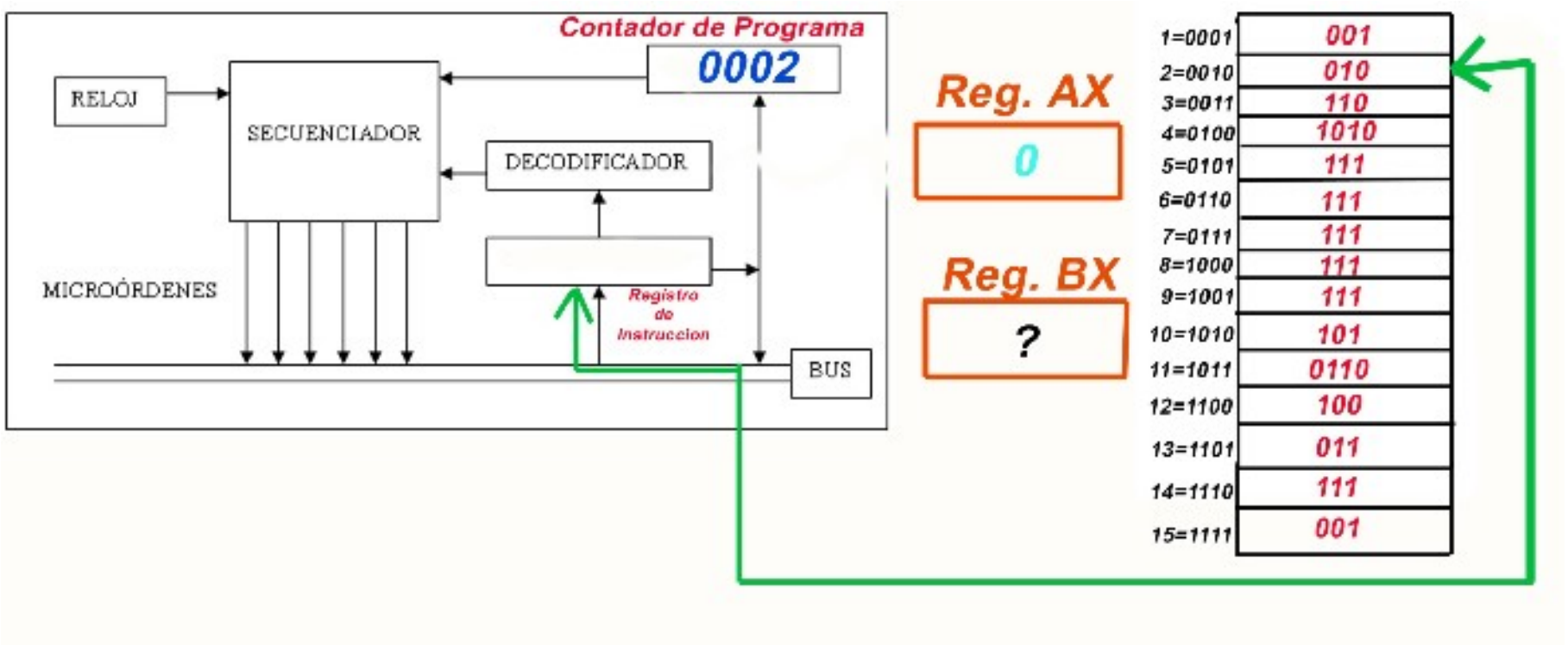
- El contador de programa contiene **0002**



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

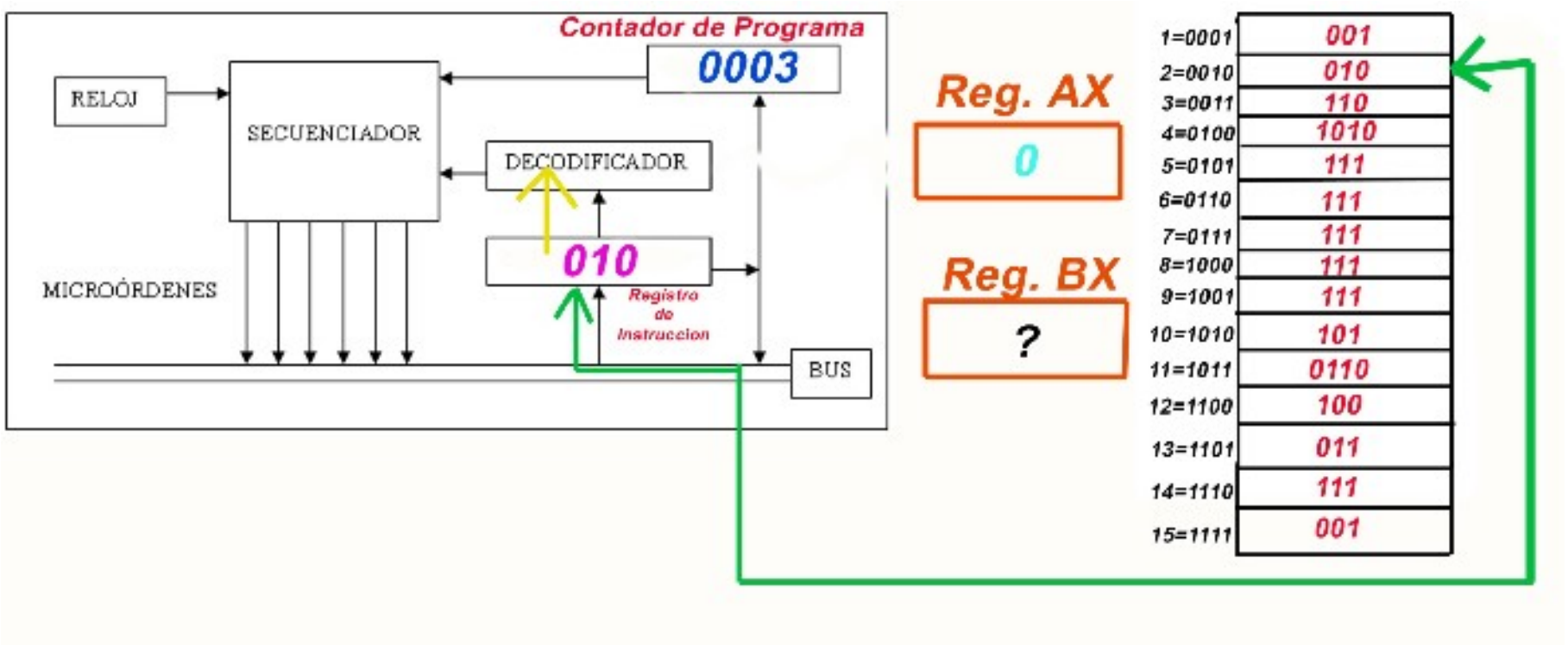
Tercer ciclo de Reloj:

- El contador de programa contiene **0002**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción:



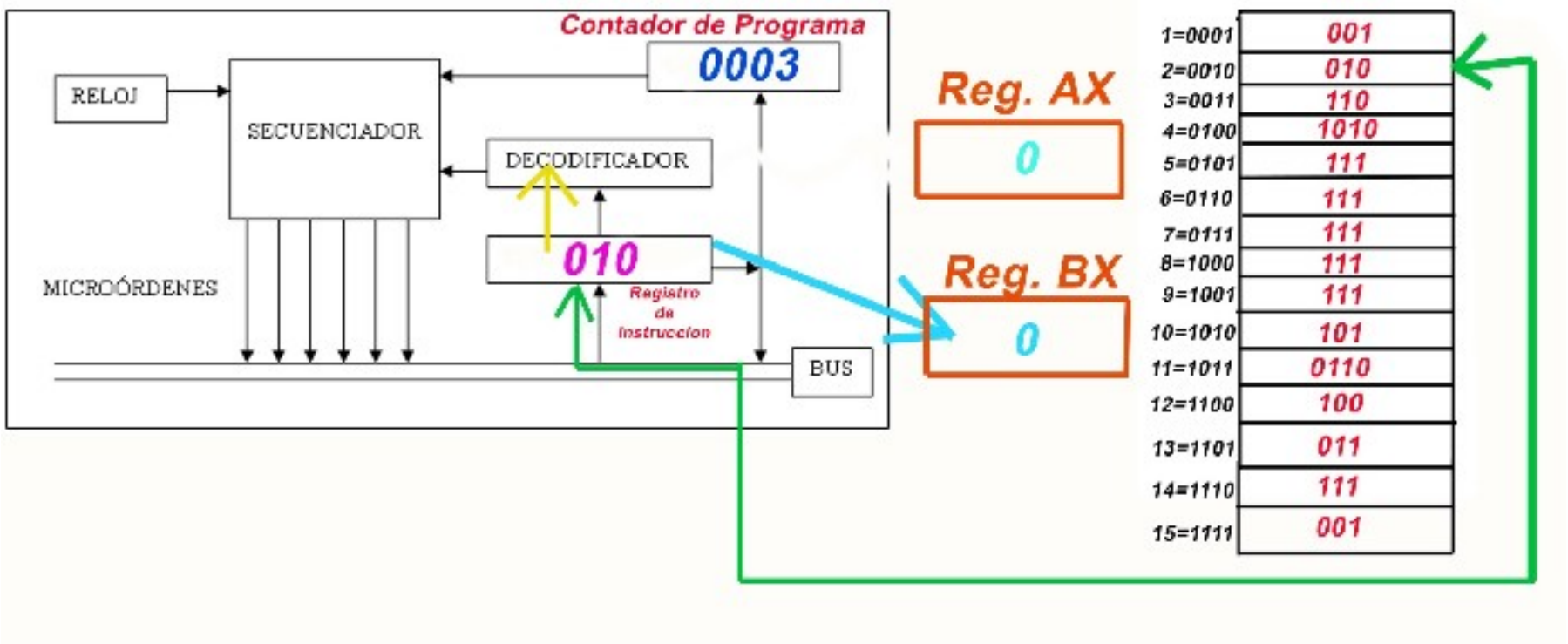
Cuarto ciclo de Reloj:

- El contador de programa se incremente automáticamente. Tenía **0002 y pasa a tener 0003.**
- Lo que hay en el Registro de instrucción, pasa a ser evaluado por el **Decodificador**.
  - El decodificador sabe que lo que hay es una instrucción, la **010**.



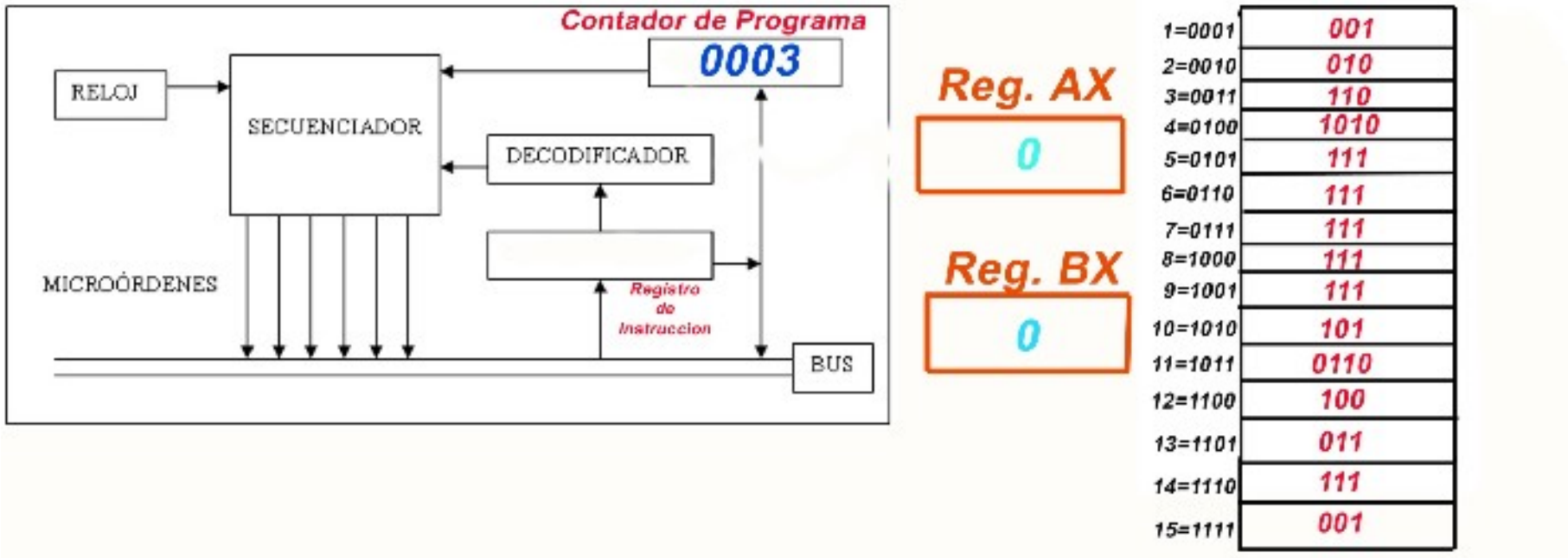
Cuarto ciclo de Reloj:

- El decodificador sabe que la instrucción **010** es:
  - 2 → 010 →Borrar registro BX
- **El decodificador se encarga directamente de cargar un 0 en el Registro BX**
- Aquí ha acabado la Segunda instrucción.



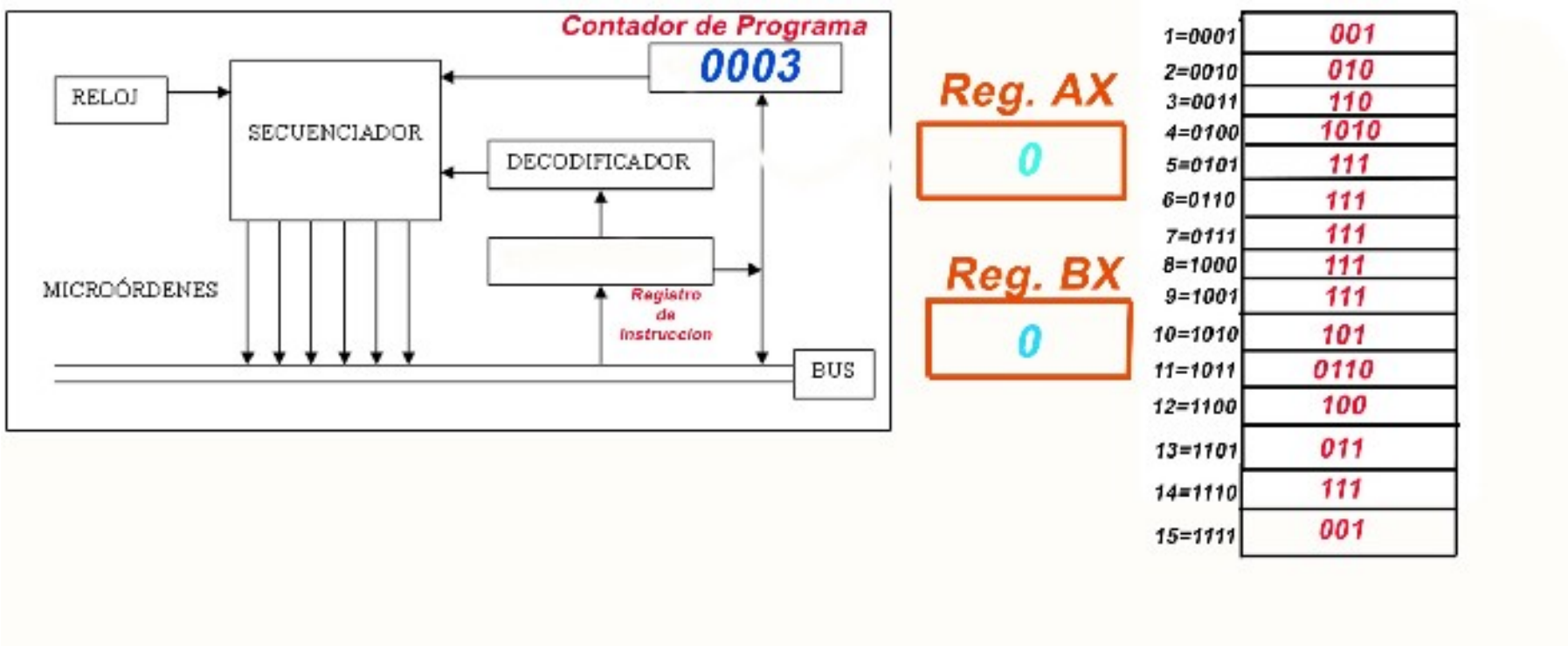
- La instrucción: 2 → 010 →Borrar registro BX
- Ha sido ejecutada en 2 ciclos de Reloj.

- La CPU sigue ejecutando la siguiente instrucción.



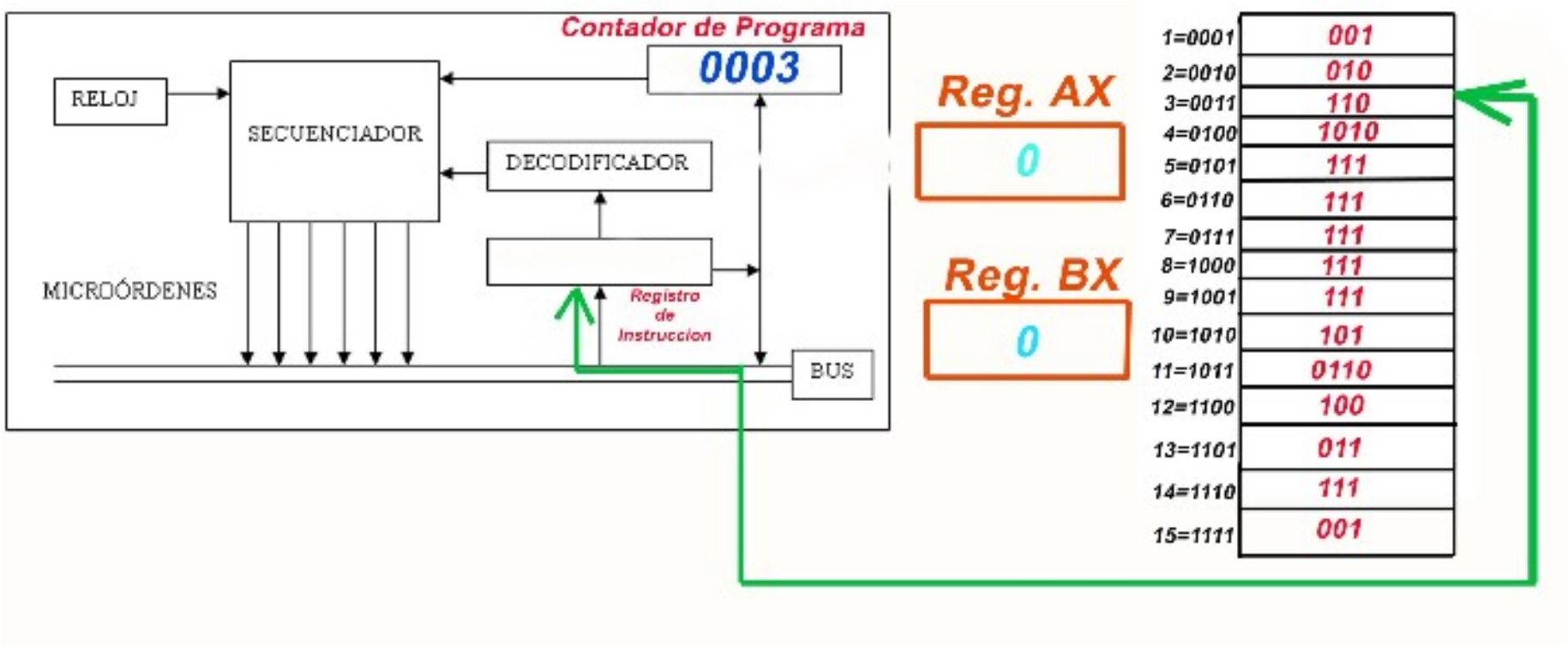
Quinto ciclo de Reloj:

- El contador de programa contiene **0003**



Quinto ciclo de Reloj:

- El contador de programa contiene **0003**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción:

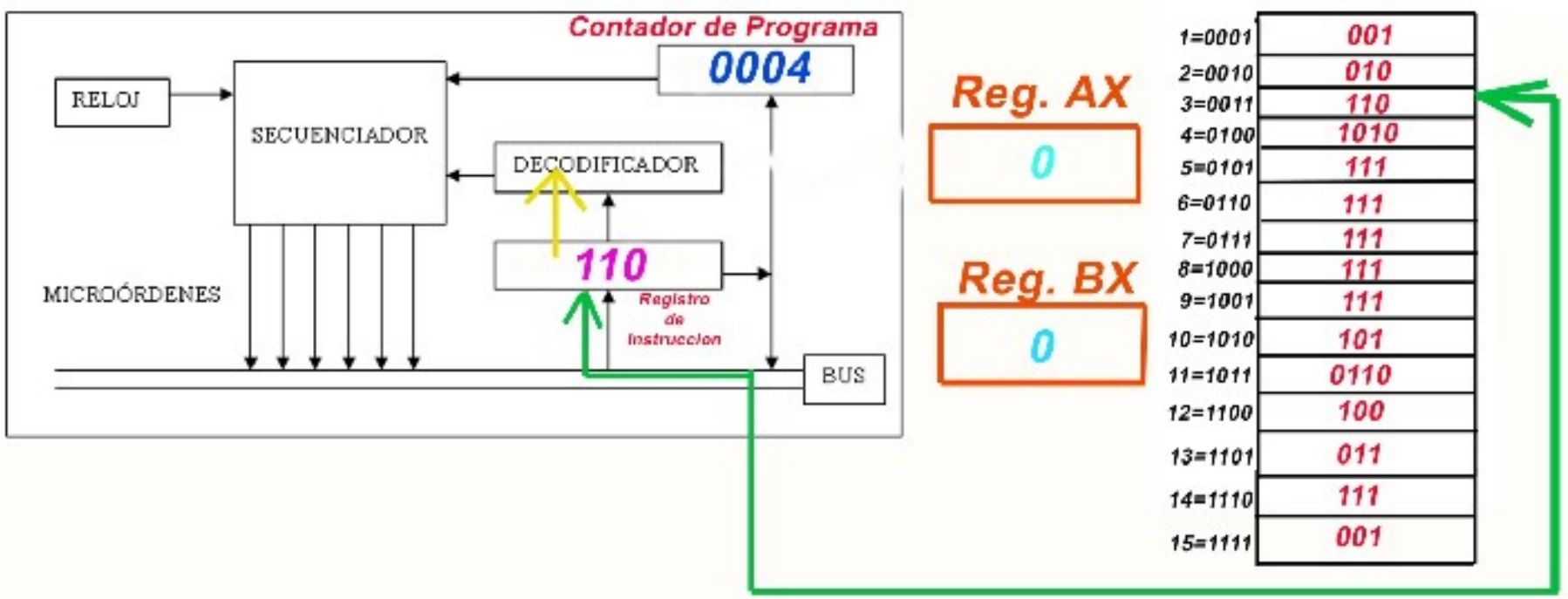


Sexto ciclo de Reloj:

•El contador de programa se incremente automáticamente. Tenía **0003 y pasa a tener 0004.**

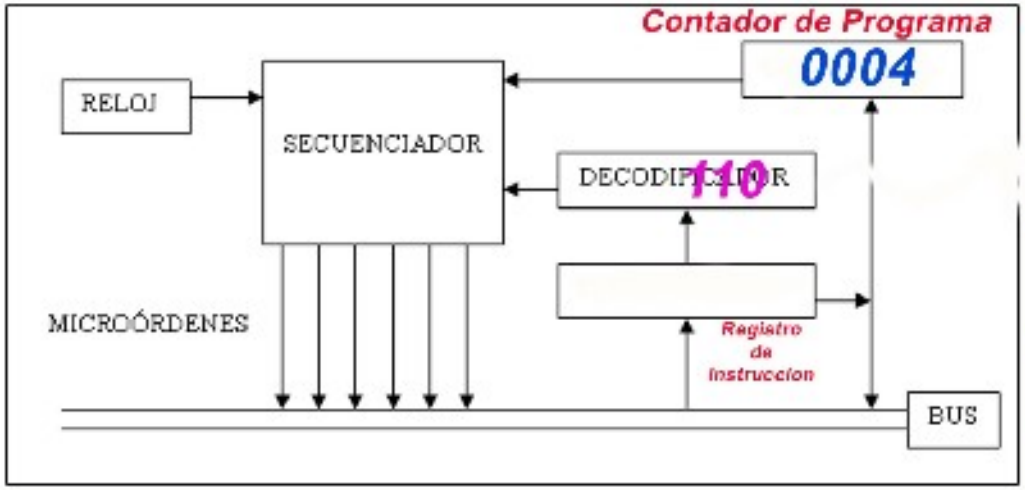
•Lo que hay en el Registro de instrucción, pasa a ser evaluado por el **Decodificador**.

•El decodificador sabe que lo que hay es una instrucción, la **110**.



Sexto ciclo de Reloj:

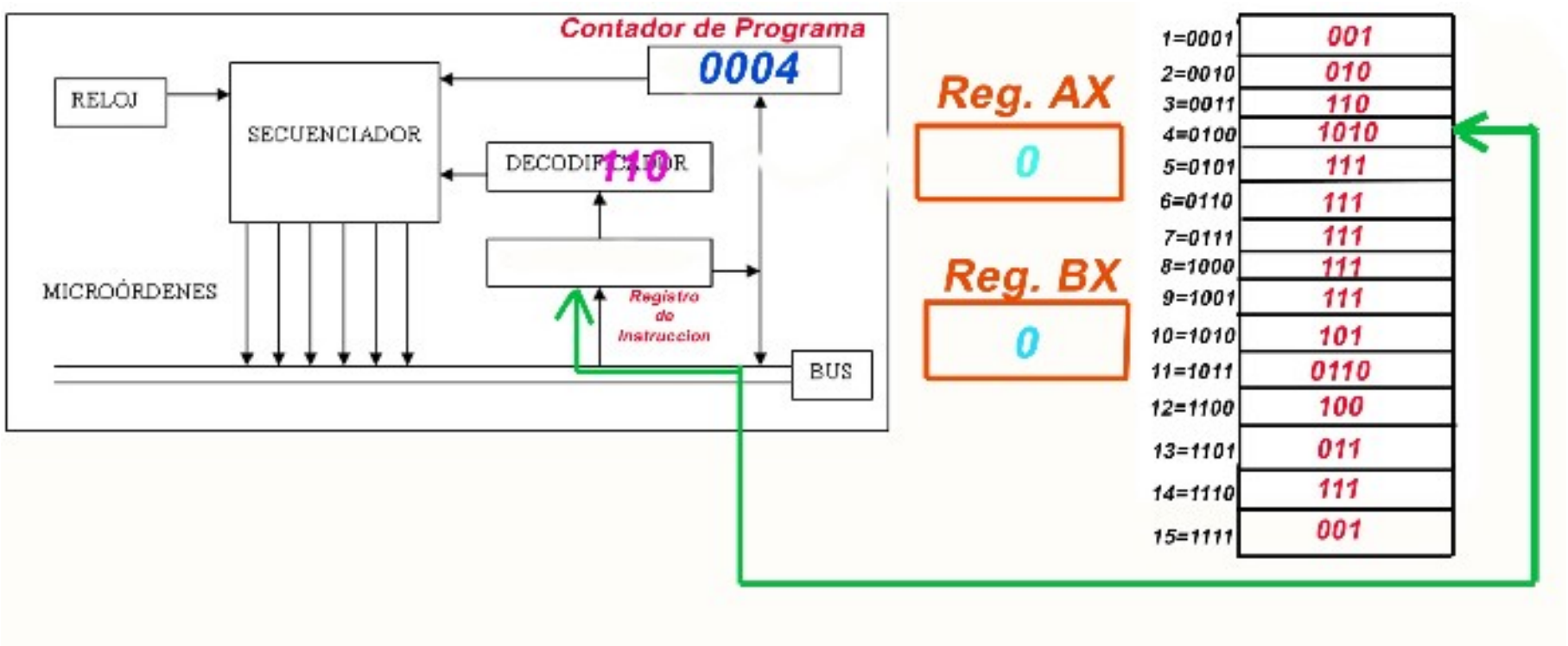
- El decodificador sabe que la instrucción **110** es:
  - 6 → 110 ➔ Salta a la posición de memoria que viene a continuación.
- Por tanto, necesito otra lectura a la memoria, a lo que marca el Contador de Programa, la posición **0004** pero en otro Ciclo de Reloj



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

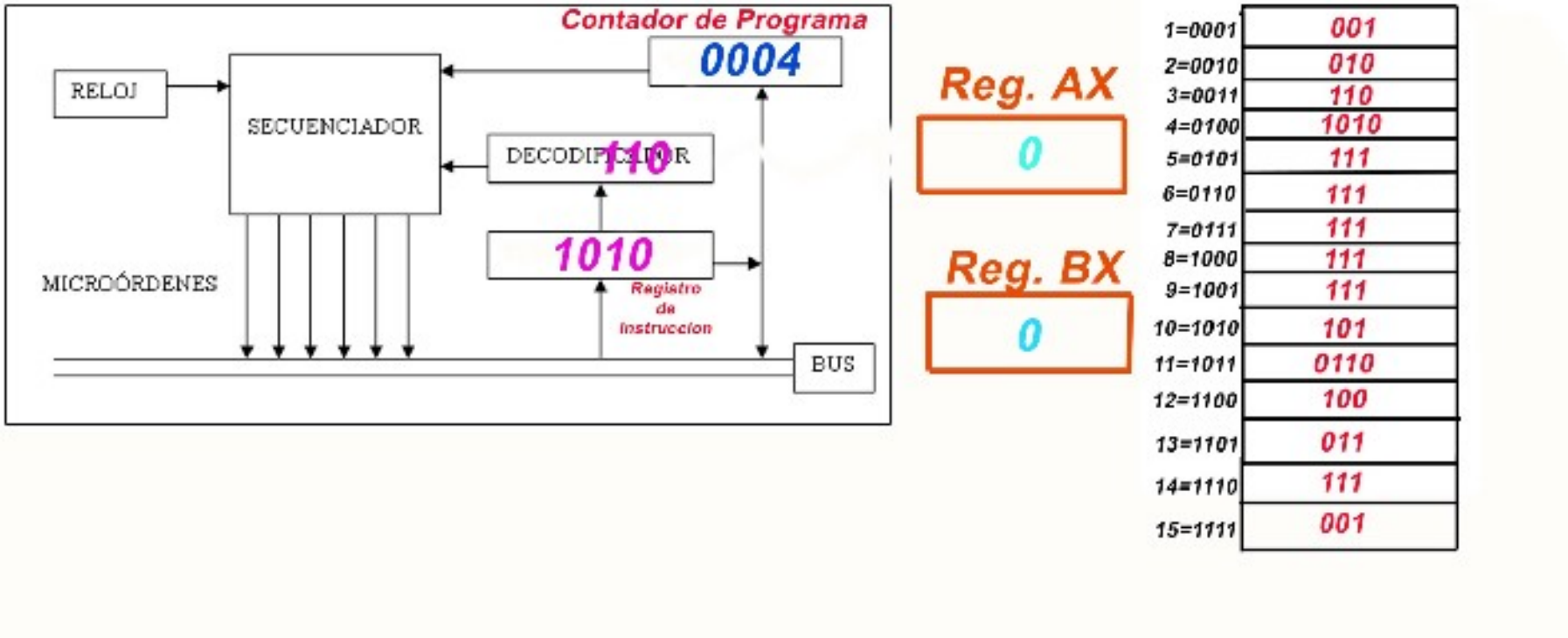
Séptimo ciclo de Reloj:

- El contador de programa contiene **0004**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción



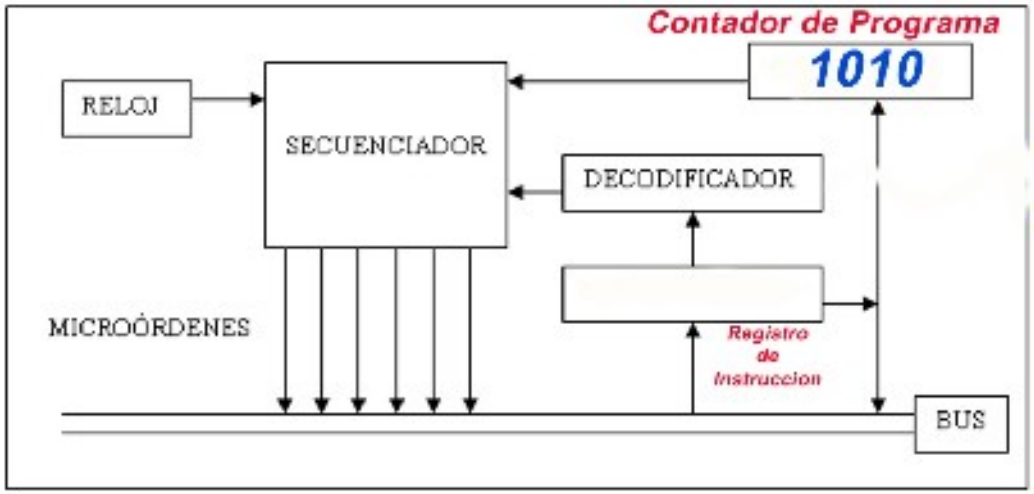
Séptimo ciclo de Reloj:

•El decodificador sabe que la instrucción 110 debe dar un salto a otra instrucción.
•El contenido **1010** va a ser a lo que tiene que apuntar el contador de programa



Séptimo ciclo de Reloj:

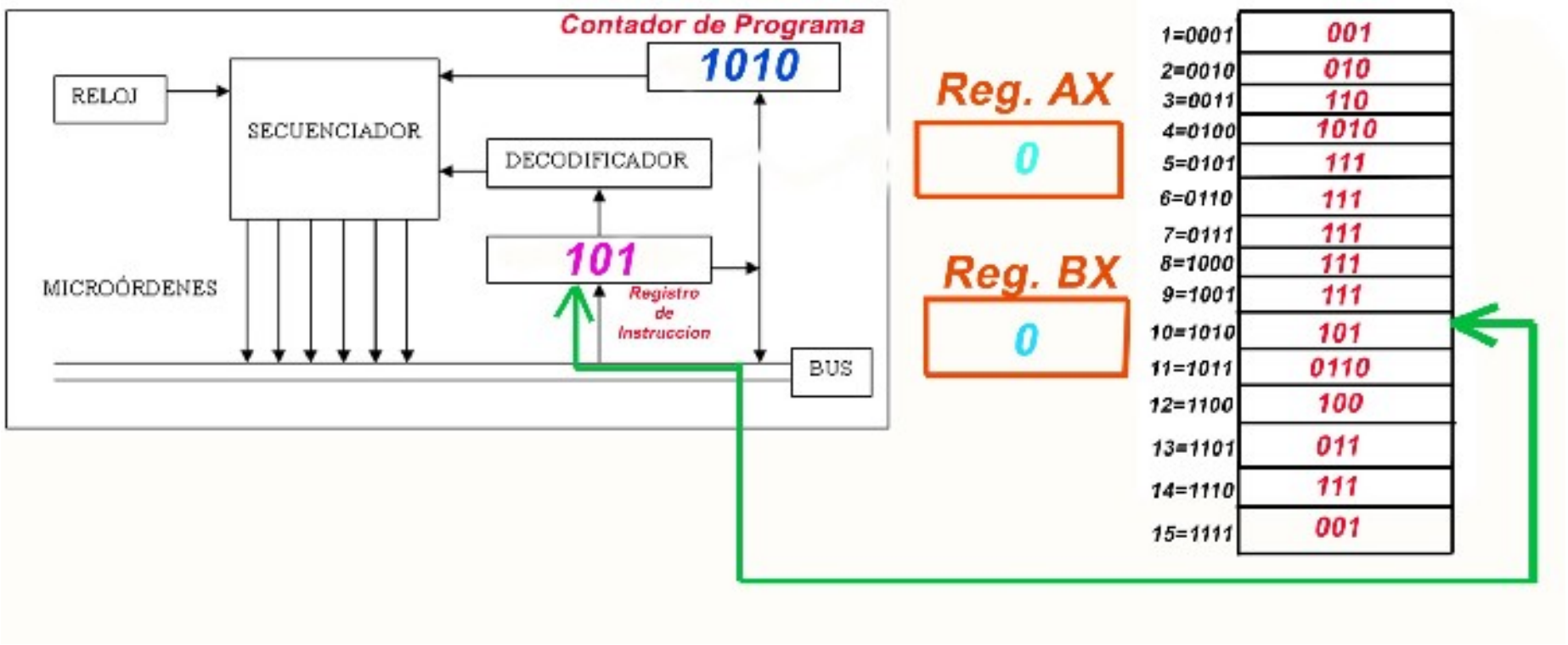
- Por lo tanto, el Contador de programa pasa a tener el valor **1010**
- **Ya ha terminado esta tercera instrucción.**
- La instrucción ha usado 3 ciclos de reloj para ejecutarse: quinto, sexto y séptimo.



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

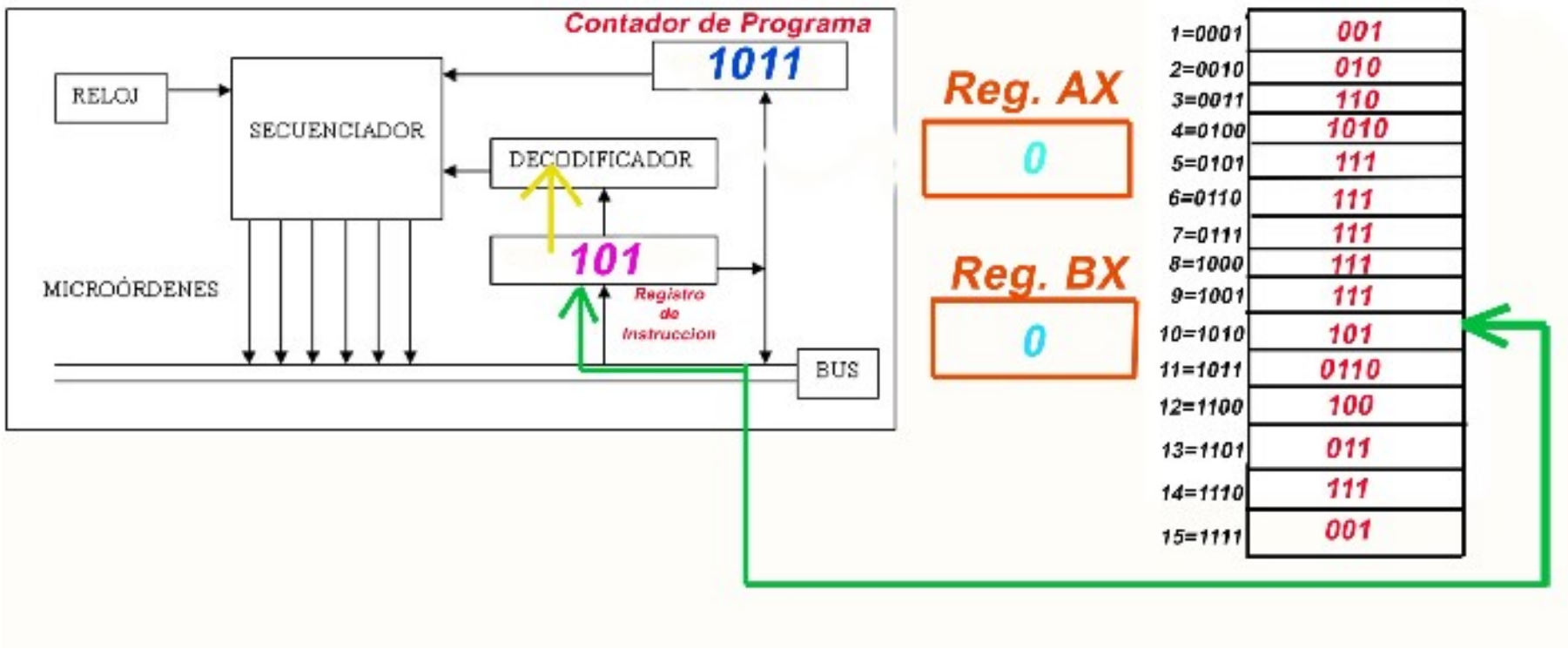
Octavo ciclo de Reloj:

- El contador de programa contiene **1010**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción:



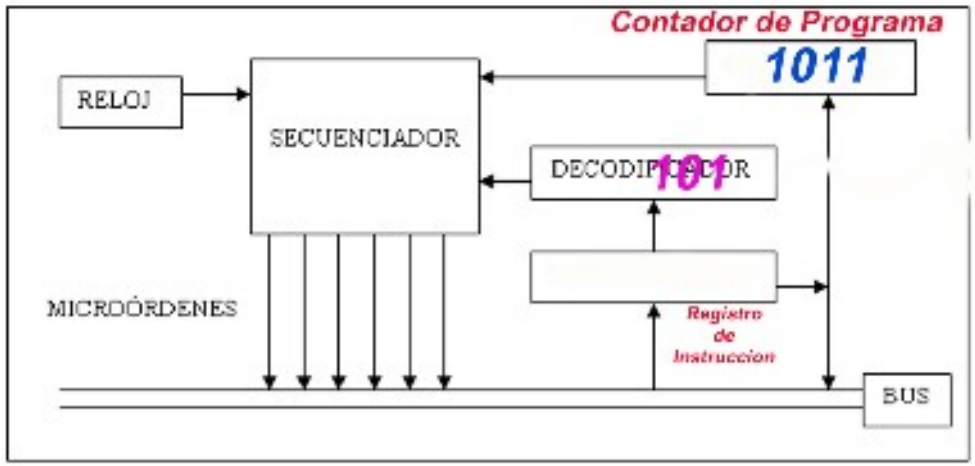
Noveno ciclo de Reloj:

- El contador de programa se incremente automáticamente. Tenía **1010 y pasa a tener 1011.**
- Lo que hay en el Registro de instrucción, pasa a ser evaluado por el **Decodificador**.
  - El decodificador sabe que lo que hay es una instrucción, la **101**.



Noveno ciclo de Reloj:

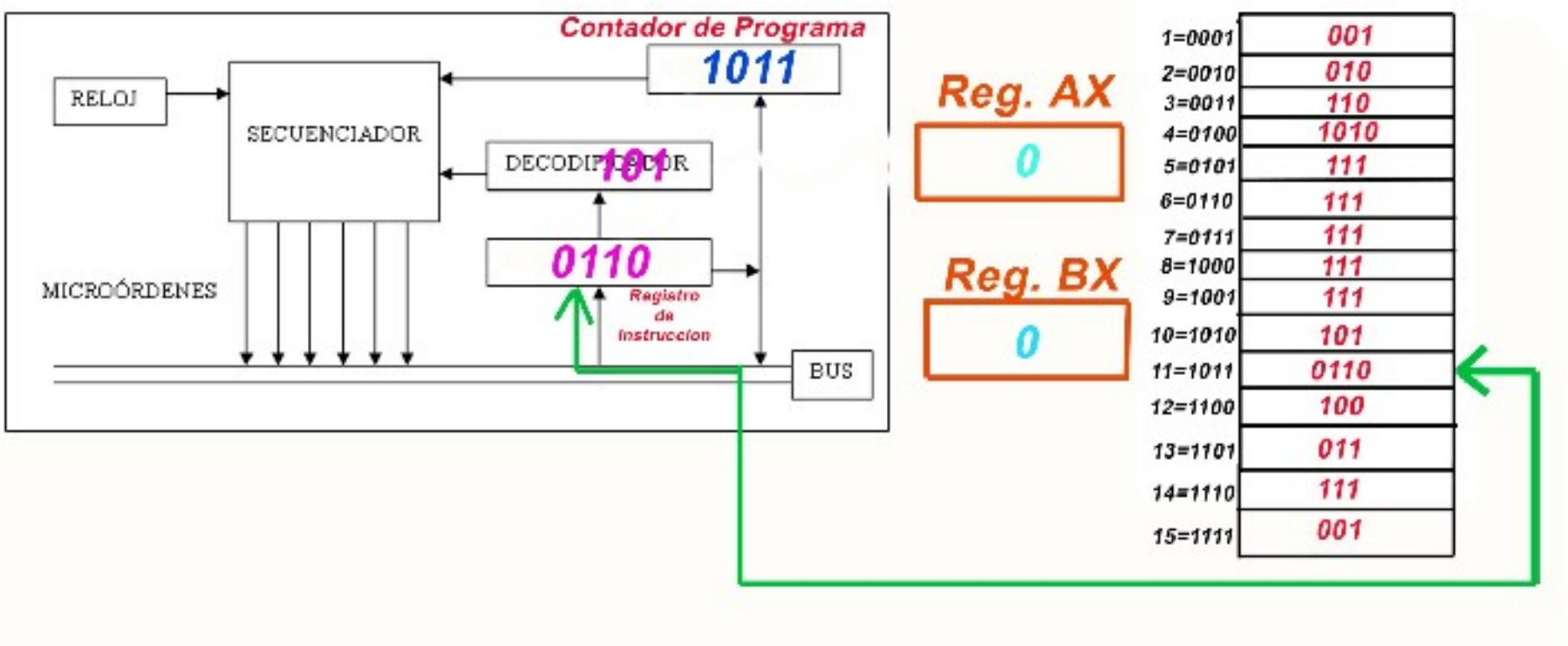
- la instrucción **101** es:
  - 5 → 101 ➔Suma AX + Contenido de una posición de memoria y el resultado lo deja en AX.
- Necesito otro dato ➔ debo hacer otra lectura a la memoria, a lo que marca el Contador de Programa, la posición **1011** pero en otro Ciclo de Reloj



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

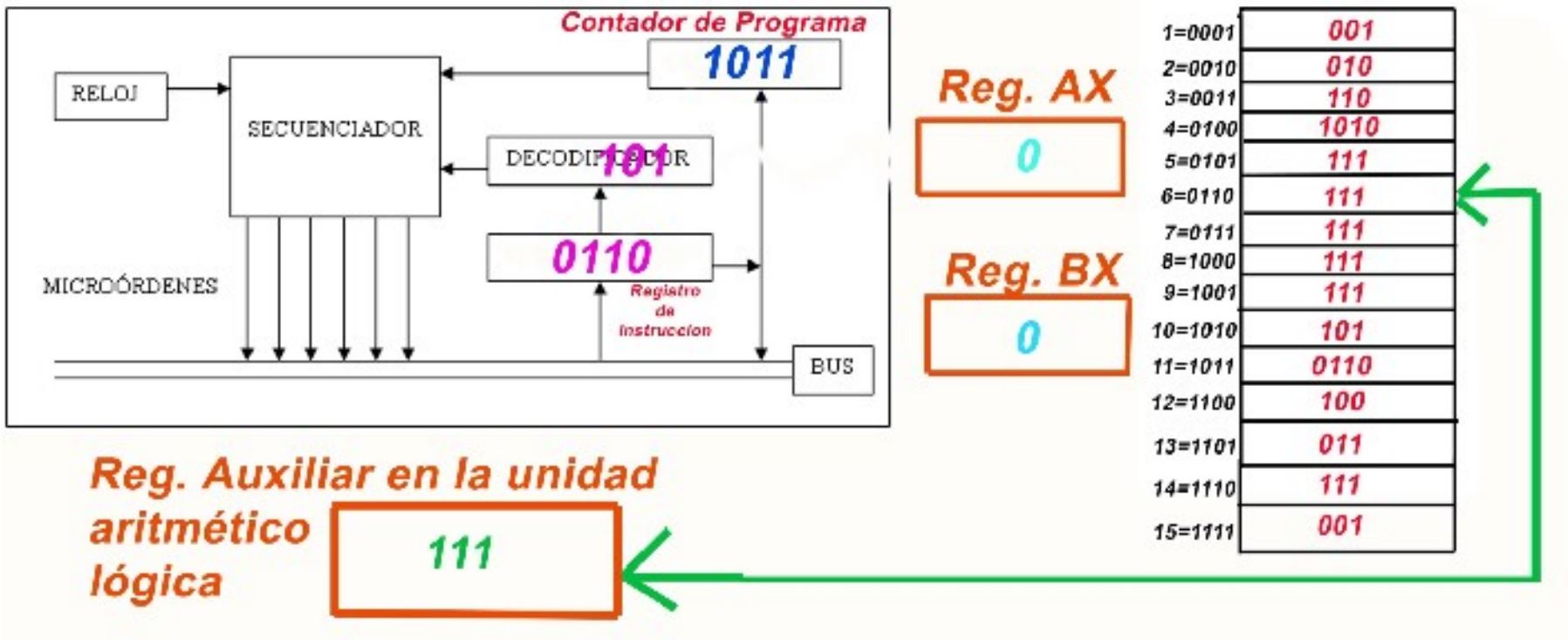
Décimo ciclo de Reloj:

- El contador de programa contiene **1011**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción:



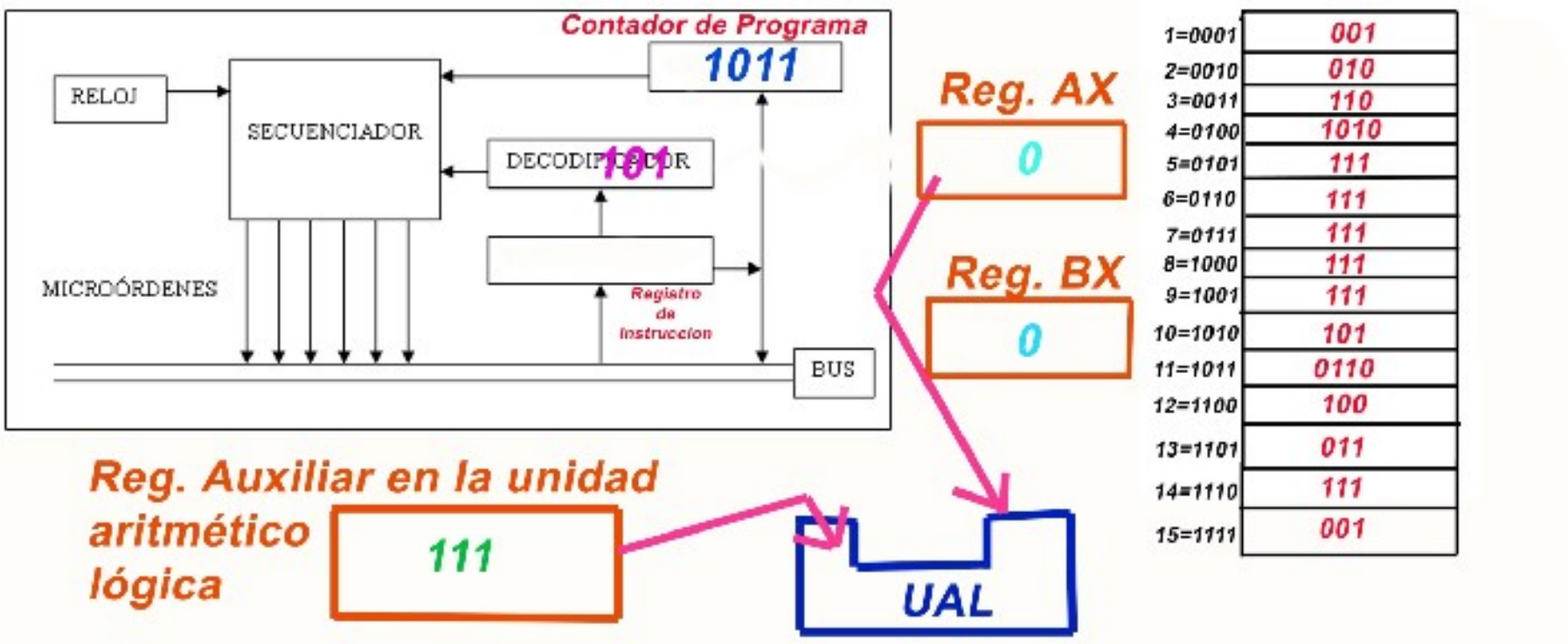
Undécimo ciclo de Reloj:

- Tenemos que saber el contenido de la dirección **0110**:
  - Por tanto, en otro ciclo de reloj, se carga en otro registro auxiliar de la unidad aritmético lógica, el contenido pedido de la dirección 0110, en nuestro caso **111**



Decimosegundo ciclo de Reloj:

- Ya tenemos todos los datos para ejecutar la instrucción **101**:
  - Suma AX + Contenido de una posición de memoria y el resultado lo deja en AX.
  - Suma **0** + **111** ➔ el resultado lo deja en AX
  - Esta suma la realiza la unidad aritmético lógica.

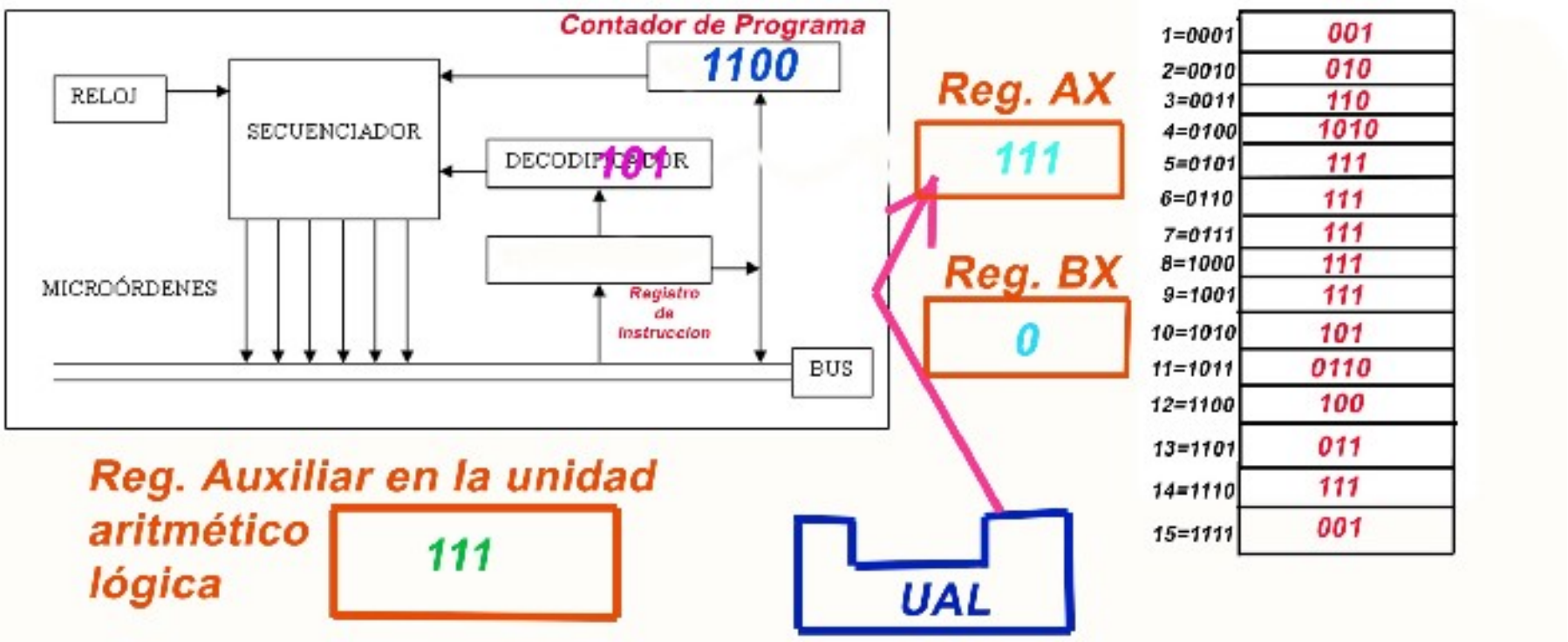


Decimosegundo ciclo de Reloj:

El contador de Programa se incrementa.
El Registro AX pasa a tener el valor del resultado de la operación: 111

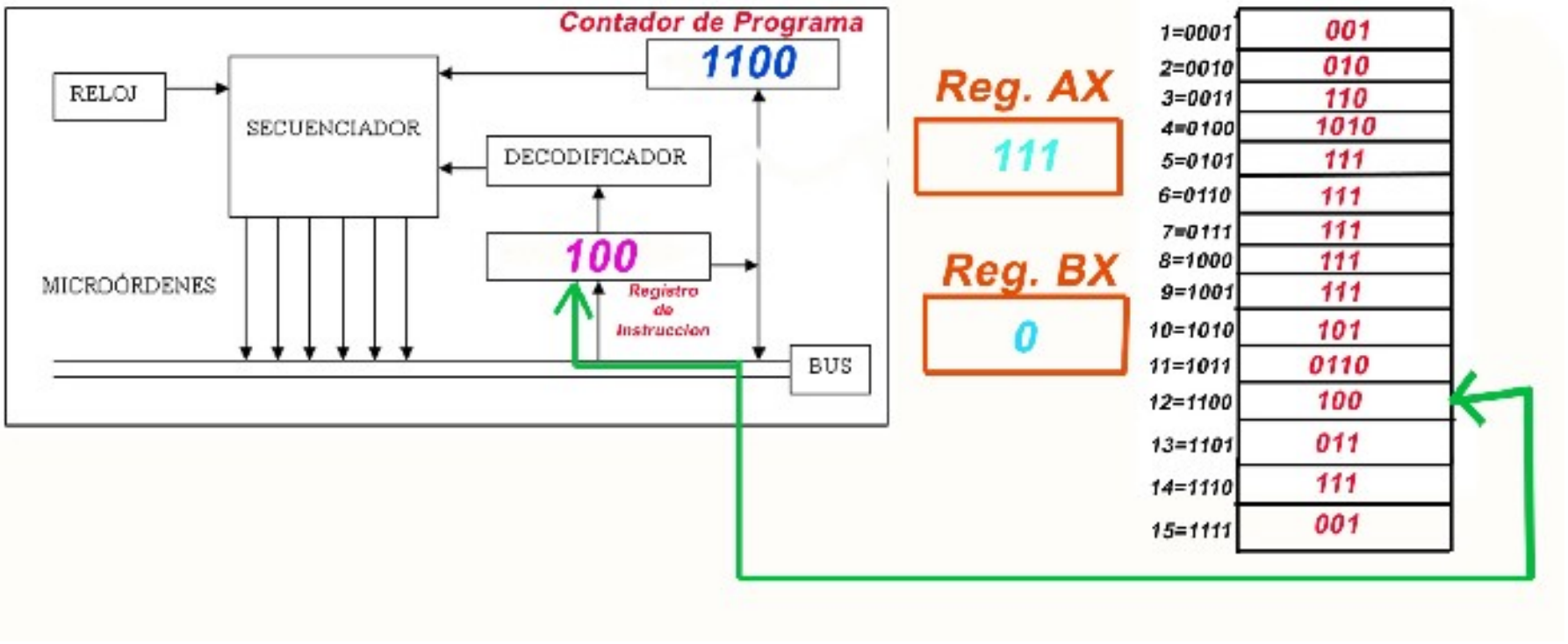
Ya ha terminado esta cuarta instrucción.

La instrucción ha usado 5 ciclos de reloj para ejecutarse: octavo, noveno, décimo, undécimo y decimosegundo.



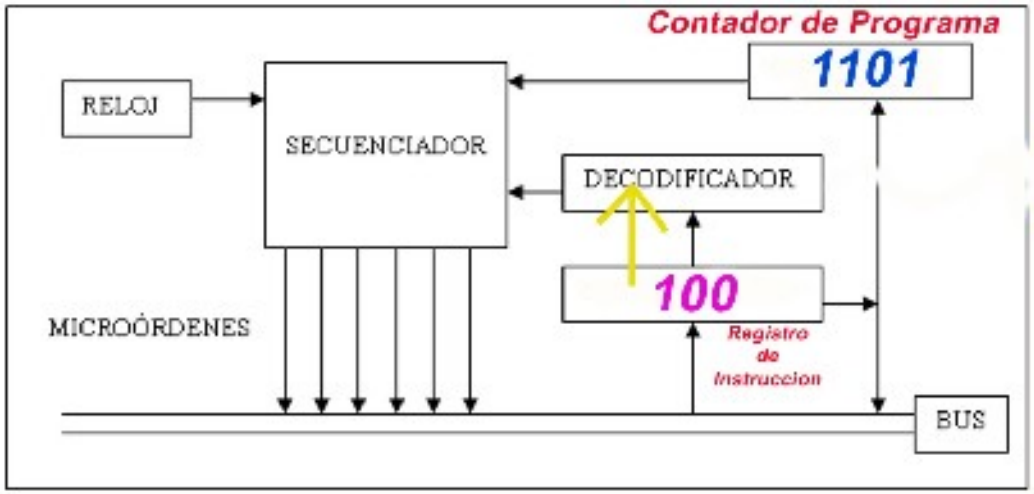
13 ciclo de Reloj:

- El contador de programa contiene **1100**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción



14 ciclo de Reloj:

- El contador de programa se incremente automáticamente. Tenía **1100 y pasa a tener 1101.**
- Lo que hay en el Registro de instrucción, pasa a ser evaluado por el **Decodificador**.
  - El decodificador sabe que lo que hay es una instrucción, la **100**.



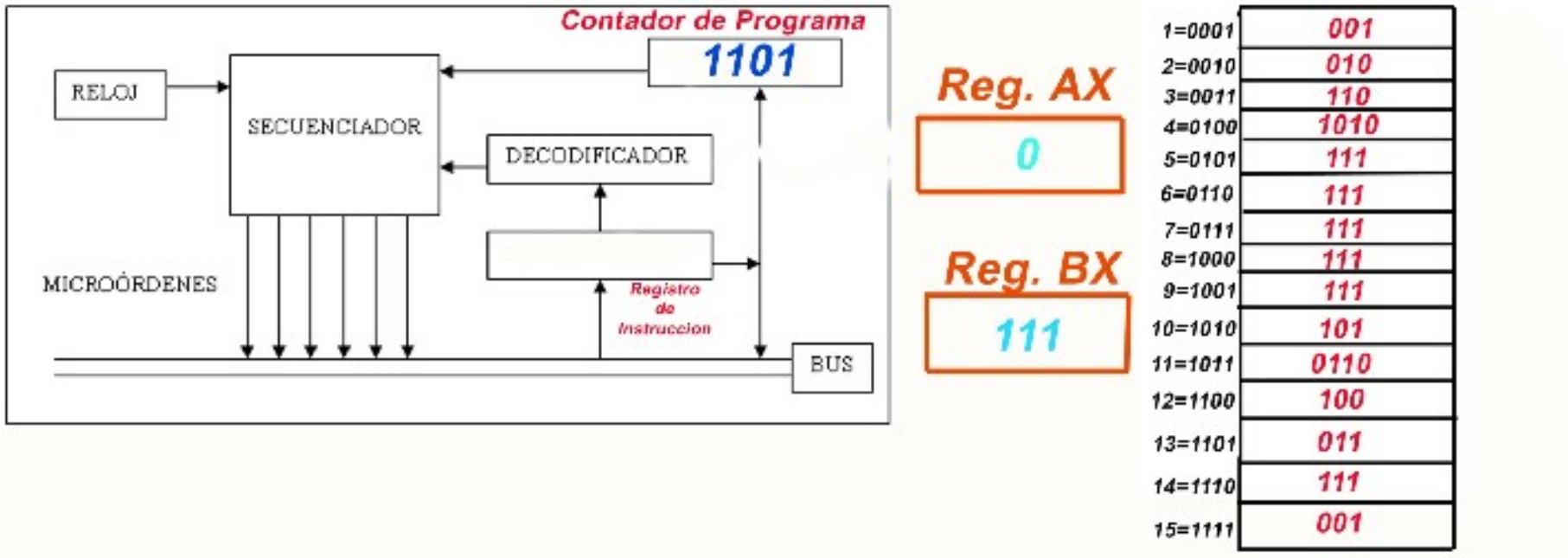
1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

14 ciclo de Reloj:

•la instrucción **100** es:

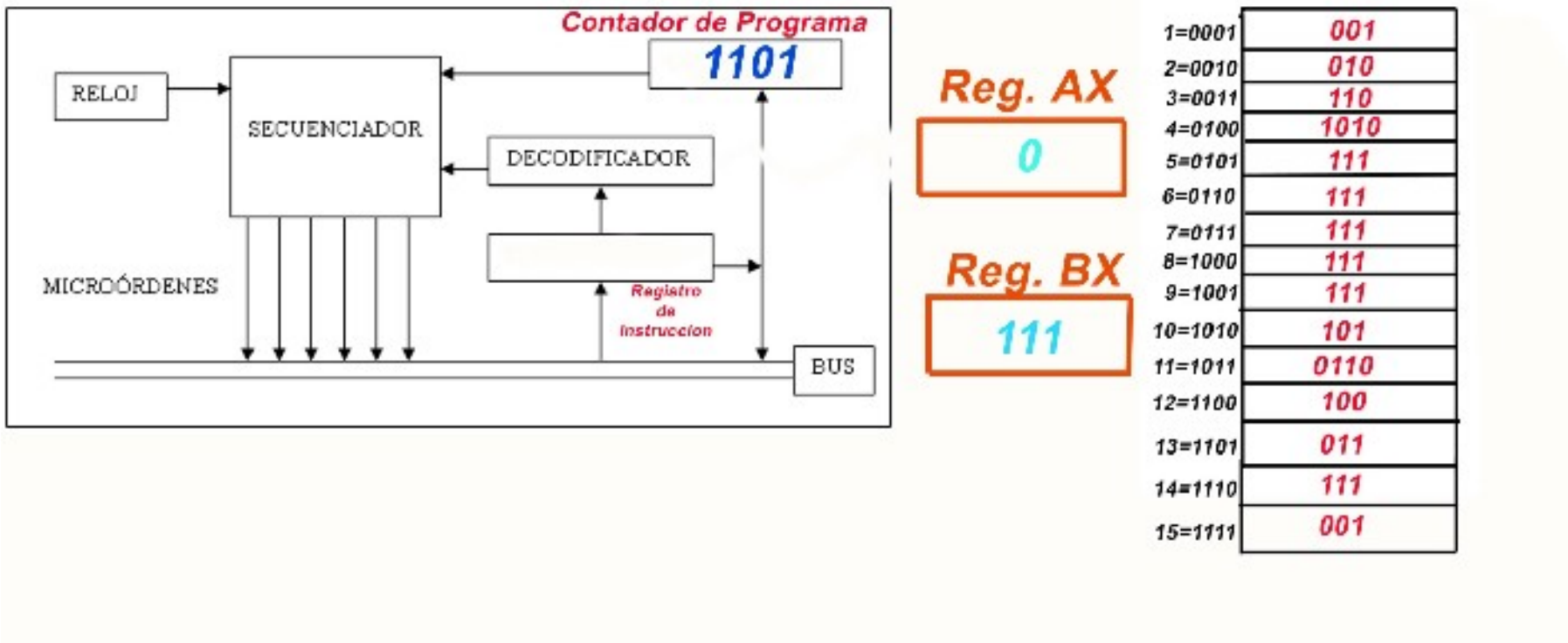
4 → 100 ➔ Cambia contenido de AX y BX

Ya puedo hacerlo directamente en este ciclo de reloj.



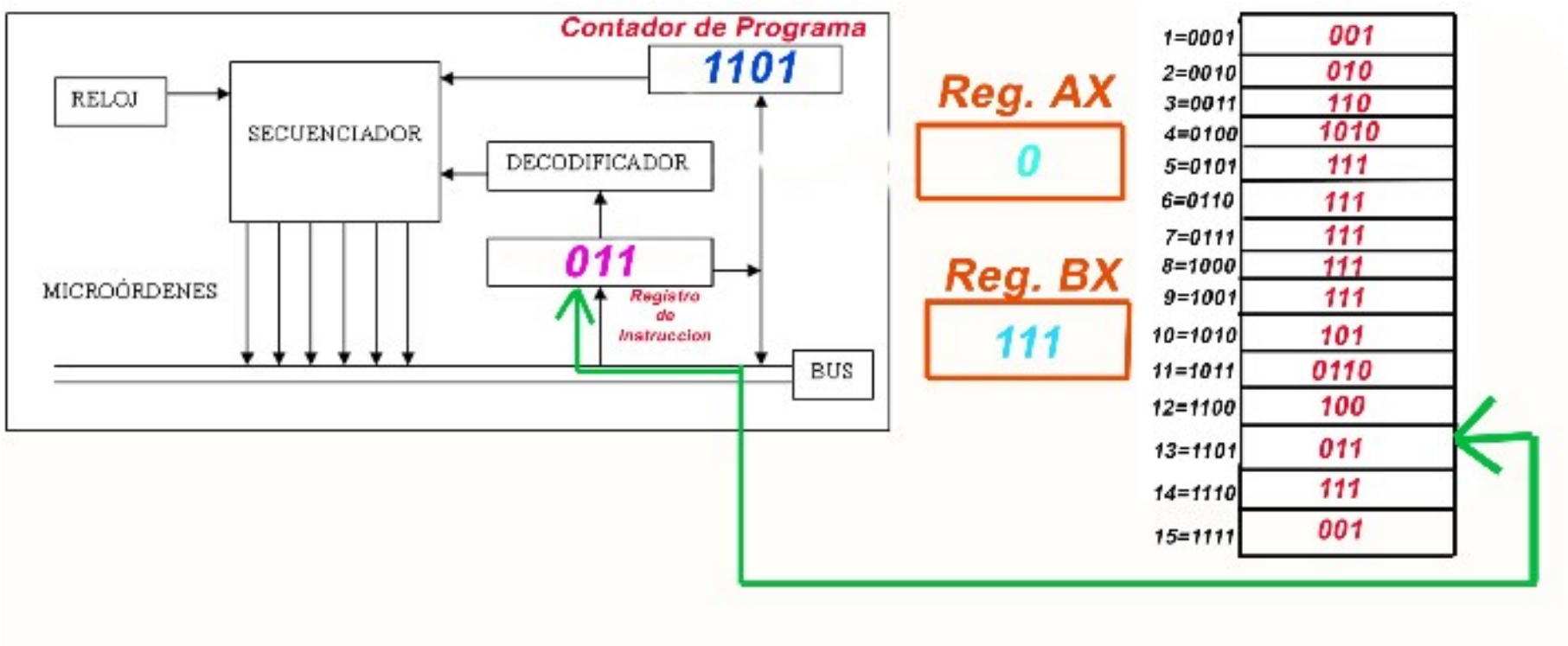
- La instrucción: 100 ➔ Cambia contenido de AX y BX
- Ha sido ejecutada en 2 ciclos de Reloj.

- La CPU sigue ejecutando la siguiente instrucción.



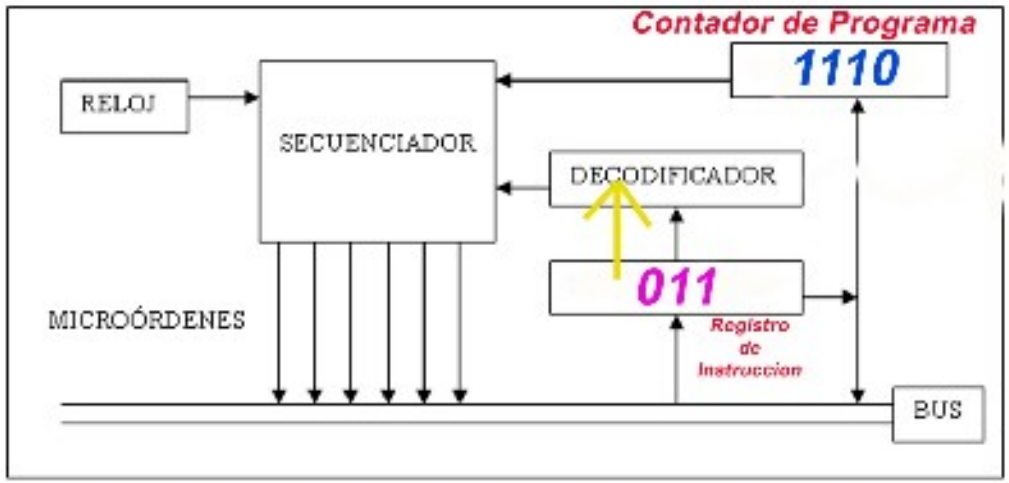
15 ciclo de Reloj:

- El contador de programa contiene **1101**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción



16 ciclo de Reloj:

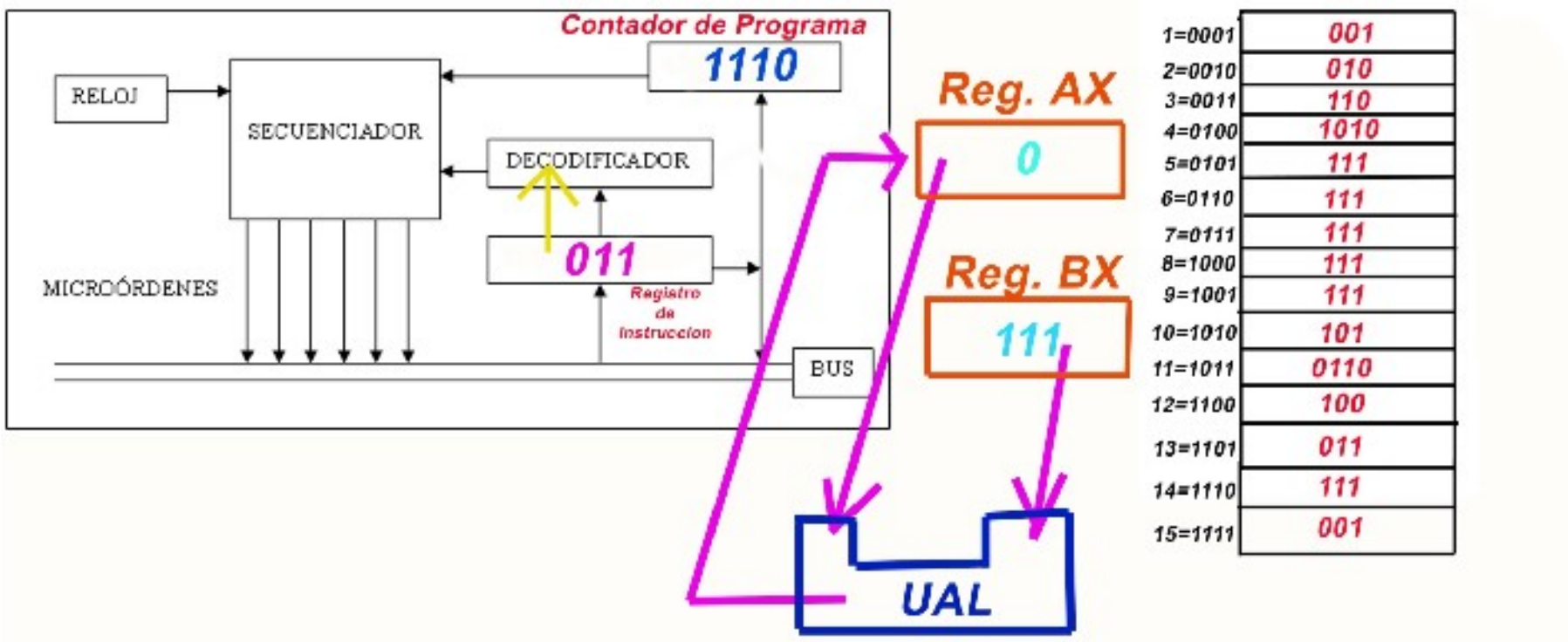
- El contador de programa se incremente automáticamente. Tenía **1101 y pasa a tener 1110.**
- Lo que hay en el Registro de instrucción, pasa a ser evaluado por el **Decodificador**.
  - El decodificador sabe que lo que hay es una instrucción, la **011**.



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

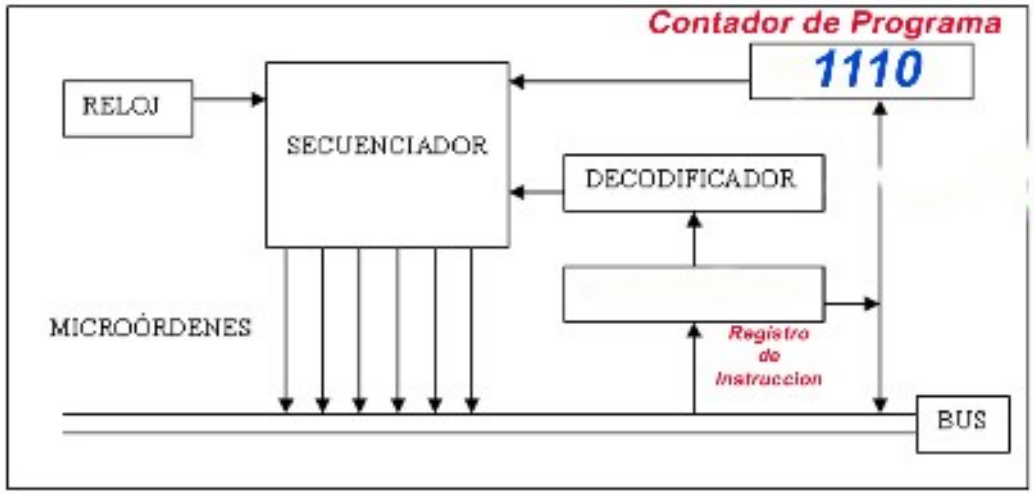
16 ciclo de Reloj:

•la instrucción **011** es:
3 → 011 →Suma AX + BX y lo deja en AX
Ya puedo hacerlo directamente en este ciclo de reloj ayudándome de la unidad aritmético lógica.



16 ciclo de Reloj:

- Quedando los registro AX y BX como vemos.
- Esta instrucción se ha ejecutado en dos ciclos de reloj
- La CPU sigue trabajando.



Reg. AX

111

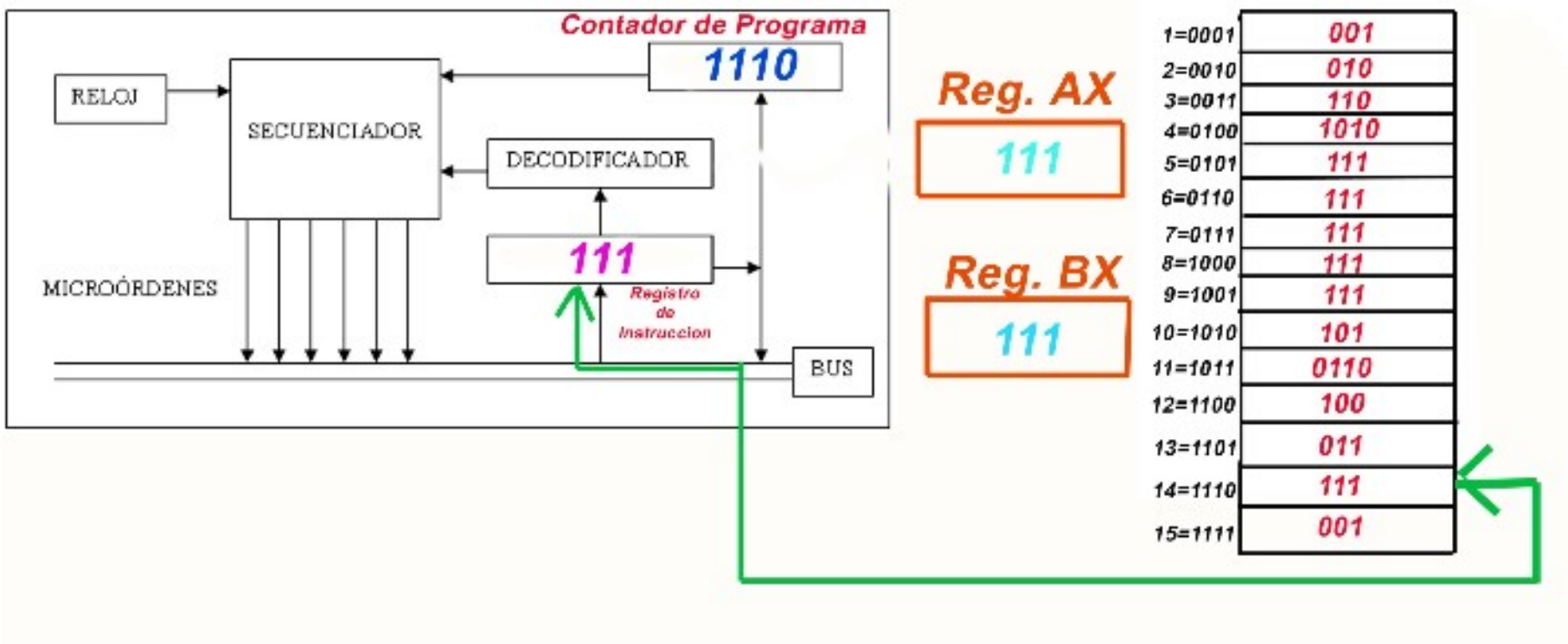
Reg. BX

111

1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

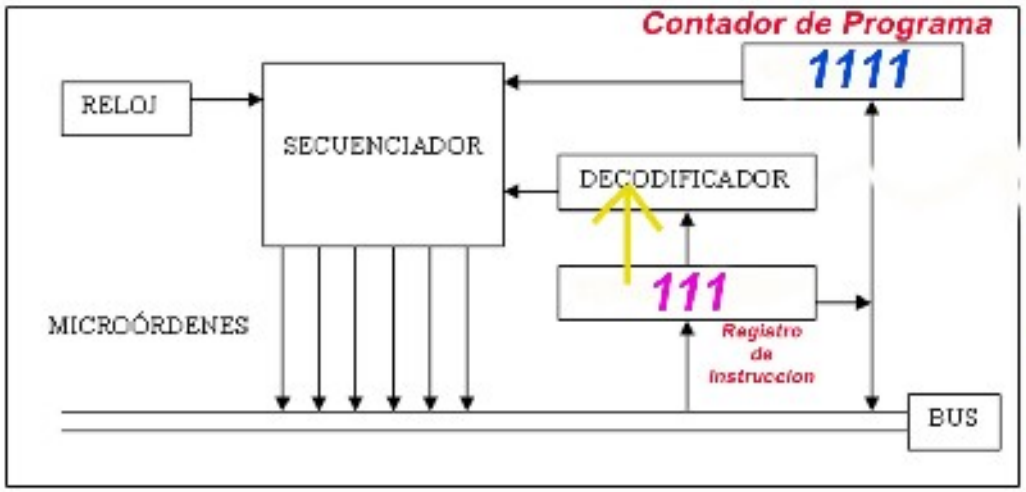
17 ciclo de Reloj:

- El contador de programa contiene **1110**
- Los buses llevan el contenido de la posición de memoria que marca el contador de programa al Registro de Instrucción



17 ciclo de Reloj:

- El contador de programa se incremente automáticamente. Tenía **1110 y pasa a tener 1111.**
- Lo que hay en el Registro de instrucción, pasa a ser evaluado por el **Decodificador**.
  - El decodificador sabe que lo que hay es una instrucción, la **111**.



1=0001	001
2=0010	010
3=0011	110
4=0100	1010
5=0101	111
6=0110	111
7=0111	111
8=1000	111
9=1001	111
10=1010	101
11=1011	0110
12=1100	100
13=1101	011
14=1110	111
15=1111	001

17 ciclo de Reloj:

•la instrucción **111** es:
111 ➔ Fin de programa
La CPU para el programa. Fin de ejecución.

El resultado final de los registros es el siguiente: