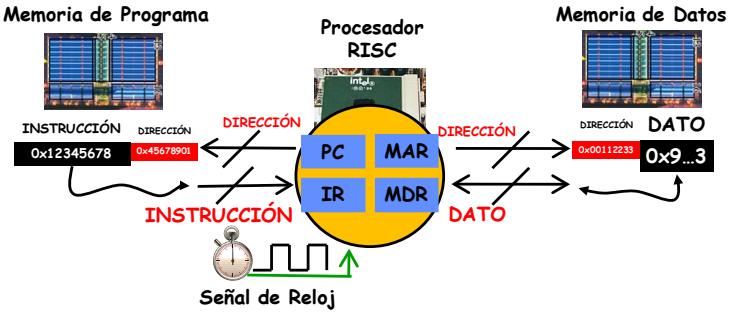
Arquitectura de Computadores
Tema 1-2 – Técnicas de Segmentación - Parte 1

Sumario

- Estructura interna del procesador
- Cómo se ejecutan las instrucciones
- Ejecución segmentada de instrucciones

Conexiones Externas del Procesador



¿Para qué se utilizan las señales externas de un procesador denominadas de forma conjunta "Dirección"? ¿Qué relación existe con el denominado Espacio de Direccionamiento del procesador?

Un programa informático se divide en una serie de instrucciones. Estas instrucciones están sujetas a unas reglas que se indican en lo que hemos denominado en el primer tema de la asignatura la **Arquitectura del Repertorio de Instrucciones** del procesador.

El procesador es la parte de un computador encargada de ejecutar las instrucciones de un programa informático. Los objetivos de esta lección son:

- Describir la organización hardware interna del procesador.
- Describir cómo el procesador realiza la ejecución de cada una de las instrucciones, básicamente dividiéndola en **etapas de ejecución**.
- Por último, describir un tipo de ejecución de instrucciones denominada **Segmentada**, que está muy extendida en los procesadores actuales.

Antes de que el procesador empiece a ejecutar instrucciones, las instrucciones de un programa se almacenan en la **Memoria de Programa** (parte de

la memoria principal que puede estar temporalmente replicada en la cache de instrucciones). Cada instrucción ocupa un número de bytes, cada uno de ellos tiene asignado una dirección memoria que es distinta para cada instrucción. Para aquellos procesadores en los que cada instrucción ocupa la misma cantidad de bytes, a veces se asigna una única dirección de memoria a cada grupo de bytes que representa a una única instrucción. Este es el caso que se puede observar en la transparencia.

Cuando el procesador va a ejecutar una nueva instrucción, empieza accediendo a su dirección en la memoria de programa, utilizando un registro denominado **Contador de Programa**, y representado por las siglas **PC**. El contenido de una dirección de la memoria de programa corresponde a lo que se denomina **código de la instrucción**, el cual incluye toda la información necesaria para que el procesador realice la correspondiente operación.

Después de que se acceda a la memoria de programa utilizando el contador de programa PC, se copia el código de la instrucción desde esta memoria de programa a un registro interno del procesador denominado **Registro de instrucción**, y que está representado por las siglas **IR**. Al direccionamiento de la memoria de programa y la posterior copia de la instrucción en un registro interno del procesador se denomina **Etapa de búsqueda de la instrucción**.

Estas y todas las demás actividades del procesador, tanto internas como externas, están sincronizadas por la denominada **señal de reloj**, que es una señal que tiene dos niveles de tensión denominados **alto** y **bajo**, los cuales se repiten periódicamente cada cierto tiempo. El tiempo de repetición se denomina **ciclo** o **periodo de reloj**, y el número de repeticiones por segundo se denomina **frecuencia de funcionamiento** del procesador.

Ciertas instrucciones denominadas de **almacenamiento** se caracterizan porque la operación asociada consiste en almacenar alguna información que se encuentra en un registro interno del procesador en la memoria del computador, la cual está mapeada en el espacio de direccionamiento del procesador. A esta parte de la memoria externa del procesador se le denomina **Memoria de Datos**. De forma semejante a la memoria de programa, la memoria de datos asigna una dirección distinta a cada bloque de datos.

Otras instrucciones denominadas de **carga** tienen asociadas una operación parecida, pero realizan la transferencia de la información en sentido contrario. Es decir, copian el contenido de una parte de la memoria de datos externa en un registro interno del procesador.

Como resultado intermedio de la ejecución de todas estas instrucciones que acceden a memoria, se genera una dirección que apunta a un bloque de datos. Esta dirección se guarda temporalmente en el denominado **Registro de Dirección de Memoria**, el cual está representado por las siglas **MAR**.

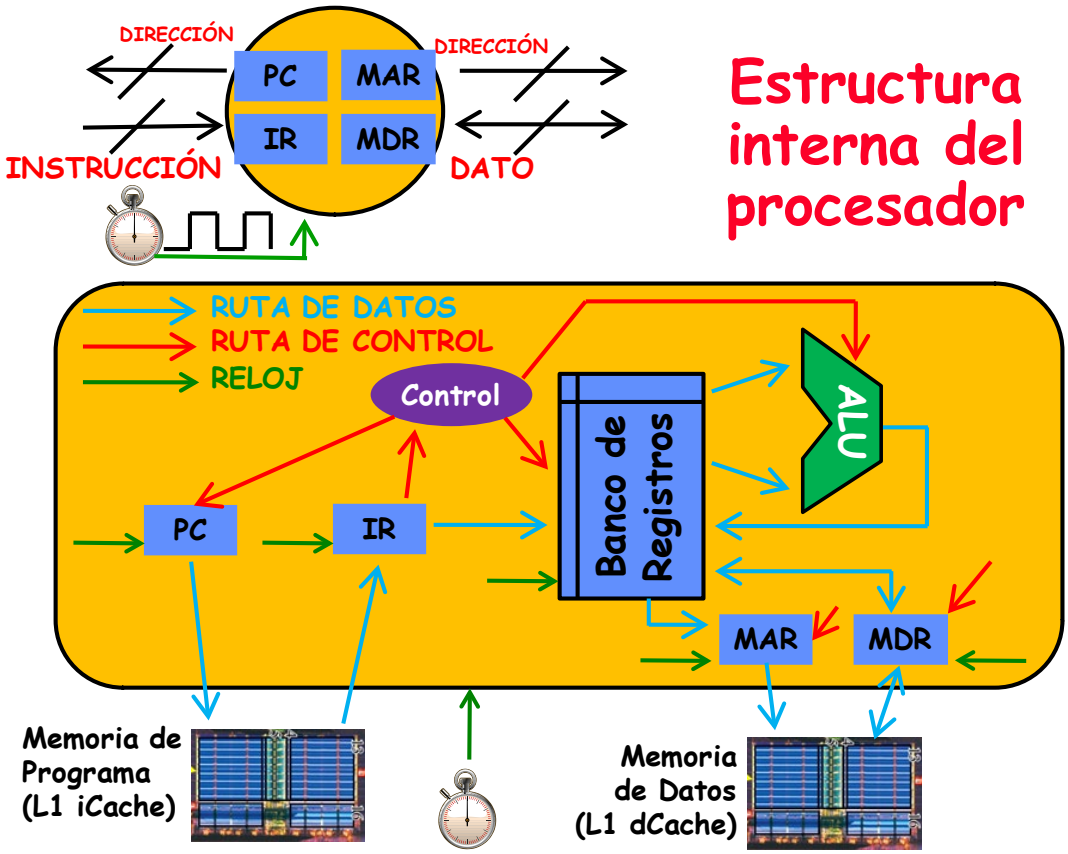
En las instrucciones de almacenamiento en memoria, el procesador utiliza otro registro interno denominado **Registro del Dato de Memoria**, el cual está representado por las siglas **MDR**. Este registro interno guarda temporalmente el dato que se va a almacenar en la memoria de datos a través de una instrucción de almacenamiento.

En las otras instrucciones de acceso a memoria denominadas de carga, se copia el contenido de la memoria de datos apuntado por el registro interno del procesador MAR en el registro MDR del procesador.

Cuando una instrucción termina de ejecutarse, el registro contador de programa PC se actualiza con la dirección de la siguiente instrucción a ejecutar. De esta forma, las instrucciones se van llevando **secuencialmente** desde la memoria de programa hacia el interior del procesador hasta que se termina de ejecutarse la última instrucción del programa informático.

En la mayoría de los casos, la siguiente instrucción en la secuencia de ejecución corresponde a la instrucción que se almacena en la dirección de memoria que está a continuación de la anterior en la memoria de programa. En ciertas ocasiones, cuando se ejecuta una **instrucción de salto**, el contador de programa PC se actualiza con la dirección de la memoria de programa donde se encuentra la instrucción que se debe ejecutar a continuación, que no tiene por qué ser la que se encuentra en la posición de memoria siguiente a la de la anterior instrucción.

A este tipo de ejecución de instrucciones se le denomina **secuencial** porque la secuencia que sigue la búsqueda de instrucciones corresponde a direcciones consecutivas en la memoria de programa. Además se le denomina **ejecución escalar** porque el procesador va a buscar una sola instrucción en la etapa de búsqueda de instrucciones. La ejecución escalar genera como máximo un resultado a la vez como fruto de la terminación de una única instrucción.



A continuación vamos a describir los principales elementos internos del procesador que permiten realizar finalmente la operación asociada a cada instrucción.

Se denominan **unidades funcionales** a las distintas partes hardware internas del procesador que interaccionan con los registros conectados directamente a sus conexiones externas y que presentamos en la transparencia anterior: PC, IR, MAR, MDR.

Se vuelve a mostrar en esta transparencia las conexiones entre las memorias externas del procesador, de programa y datos, con los registros internos del procesador. Los registros internos del procesador que se representan en azul están sincronizados por la señal de reloj del procesador. Esto significa que la actualización de estos registros se realiza en una de las transiciones de la señal de reloj.

Una unidad funcional muy importante del procesador es el denominado **banco de registros**. Su función consiste en implementar los registros de la arquitectura del repertorio de instrucciones del procesador y proporcionar los operandos de la operación asociada con la instrucción. Esta unidad funcional es como una pequeña memoria dividida en registros, cada uno de ellos

se identifica con una **dirección de registro**. Esta dirección de registro está normalmente incluida en el código de instrucción que se guarda temporalmente en el registro de instrucción IR durante lo que denominamos en la transparencia anterior la **etapa de búsqueda de la instrucción**. Esta etapa se corresponde con la primera de las cuatro etapas en las que vamos a dividir la ejecución de una instrucción.

Una segunda etapa de ejecución de la instrucción se denomina **búsqueda de operandos** y está también sincronizada con la señal de reloj. En esta segunda etapa, la unidad funcional banco de registros proporciona hasta dos operandos.

En las instrucciones aritmético-lógicas, estos dos operandos son recogidos por otra unidad funcional denominada **unidad aritmético-lógica**, la cual está representada por las siglas **ALU**.

En las instrucciones de acceso a memoria, el banco de registros proporciona la dirección de memoria que se almacena temporalmente en el registro MAR. Adicionalmente, el banco de registros proporciona el dato que se almacena a través de las instrucciones de almacenamiento. Este dato se guarda temporalmente en el registro MDR.

Una tercera fase en la que se divide la ejecución de una instrucción se denomina explícitamente **etapa de ejecución**. La parte interna del procesador que se activa en esta fase depende del tipo de instrucción que se esté ejecutando. En este sentido, clasificamos las instrucciones en dos grupos distintos: las que utilizan la unidad funcional denominada aritmético-lógica (ALU), y las que acceden a la memoria de datos.

En la etapa de ejecución de las instrucciones aritmético-lógicas, la ALU transforma los datos de entrada para generar un resultado que se corresponde con el de una operación aritmética (suma, resta, etc.) o lógica (Y-lógica, O-lógica, etc.).

En la etapa de ejecución de las instrucciones de carga, la memoria de datos es direccionada por la dirección almacenada temporalmente en el registro MAR y devuelve un dato que se guarda temporalmente en el registro MDR. En las instrucciones de almacenamiento, en la dirección apuntada por el registro MAR se guarda el contenido del registro MDR.

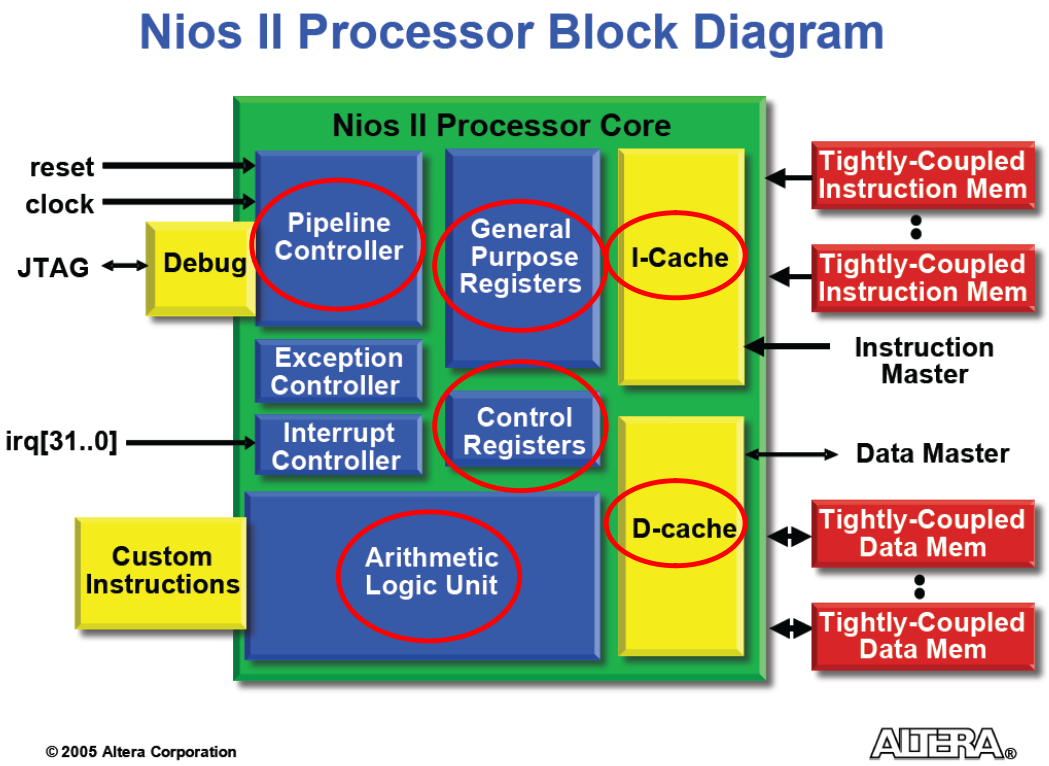
El resultado de la etapa de ejecución de las instrucciones aritmético-lógicas y de carga produce un dato que es guardado temporalmente en uno de los registros del banco de registros. Este almacenamiento interno en el procesador se realiza en una cuarta y última etapa de ejecución de la instrucción de se denomina **post-escritura**.

Observar en la transparencia que la propagación y transformación de la información obtenida en el interior del procesador y que es posteriormente almacenada en la memoria de datos sigue unas trayectorias representadas por líneas azules. A estas trayectorias se les denomina conjuntamente la **ruta de datos** del procesador.

Para que la propagación de los datos se realice siguiendo la secuencia de las cuatro etapas que hemos descrito previamente: Búsqueda de Instrucción, Búsqueda de Operandos, Ejecución, y Post-escritura, se necesitan otras señales que denominamos **señales de control** y que las representamos en rojo. Estas señales de control son generadas por otra unidad funcional interna del procesador denominada **unidad de control**.

El instante en el que se generan correctamente estas señales de control ocurre a continuación de cuando la instrucción se guarda en el registro de instrucción IR. Hasta ese momento el procesador no conoce qué instrucción es la que se va a ejecutar. Por ello, la unidad de control es la que interpreta el código de la instrucción guardado temporalmente en el registro IR, y seguidamente genera las señales de control. A este proceso se le denomina **decodificación de la instrucción** y se realiza a la vez que la búsqueda de los operandos. Por eso la segunda etapa de ejecución de la instrucción se le denomina **decodificación y búsqueda de operandos**.

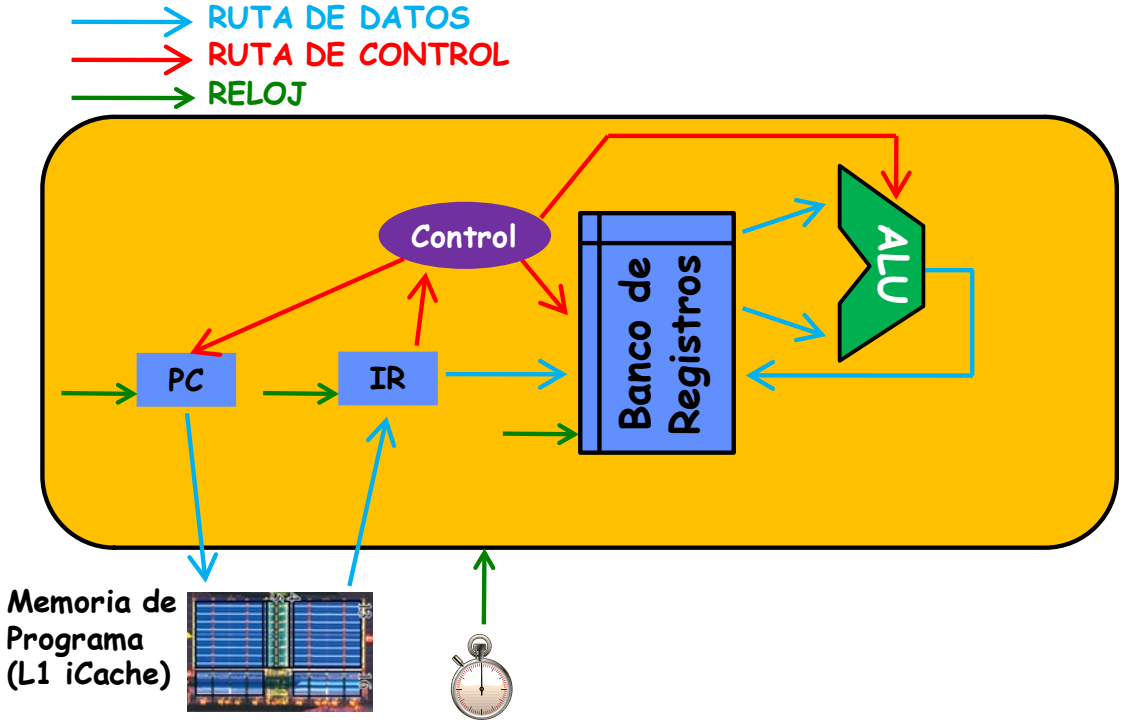
Estructura interna de NIOS II



En esta transparencia observamos las principales unidades funcionales que se encuentran en el interior del procesador que hemos utilizado en prácticas denominado NIOS II. Observar aquí cómo se nombran en inglés las distintas partes internas del procesador que presentamos en la transparencia anterior.

- **I-Cache** es la Memoria de Programa y en NIOS II se implementa en forma de memoria cache.
- **D-Cache** es la Memoria de Datos y en NIOS II se implementa en forma de memoria cache también.
- **General Purpose Registers** es el Banco de Registros.
- La unidad de control se implementa en NIOS II con el módulo denominado **Pipeline Controller** además de los registros denominados Control Registers.
- Y por último, se destaca la unidad aritmético-lógica que se implementa en NIOS II con el módulo denominado **Arithmetic Logic Unit**.

Ejecución de una instrucción aritmético-lógica



Vamos a ver ahora las distintas unidades funcionales internas del procesador que se activan en la ejecución de los distintos tipos de instrucciones. En primer lugar describimos las instrucciones aritmético-lógicas; es decir, las que tienen codificadas las operaciones de suma, resta, etc., así como las de Y-logica, O-lógica, etc. Esta descripción la vamos a realizar en el orden que se suceden las sucesivas etapas de ejecución de las instrucciones que hemos presentado en anteriores transparencias.

En la primera etapa de búsqueda de la instrucción, el contador de programa PC apunta a la dirección de memoria de la siguiente instrucción a ejecutar, que en este caso suponemos que es de tipo aritmético-lógica. La memoria de programa proporciona el código de la instrucción, el cual se almacena temporalmente en el registro de instrucción IR.

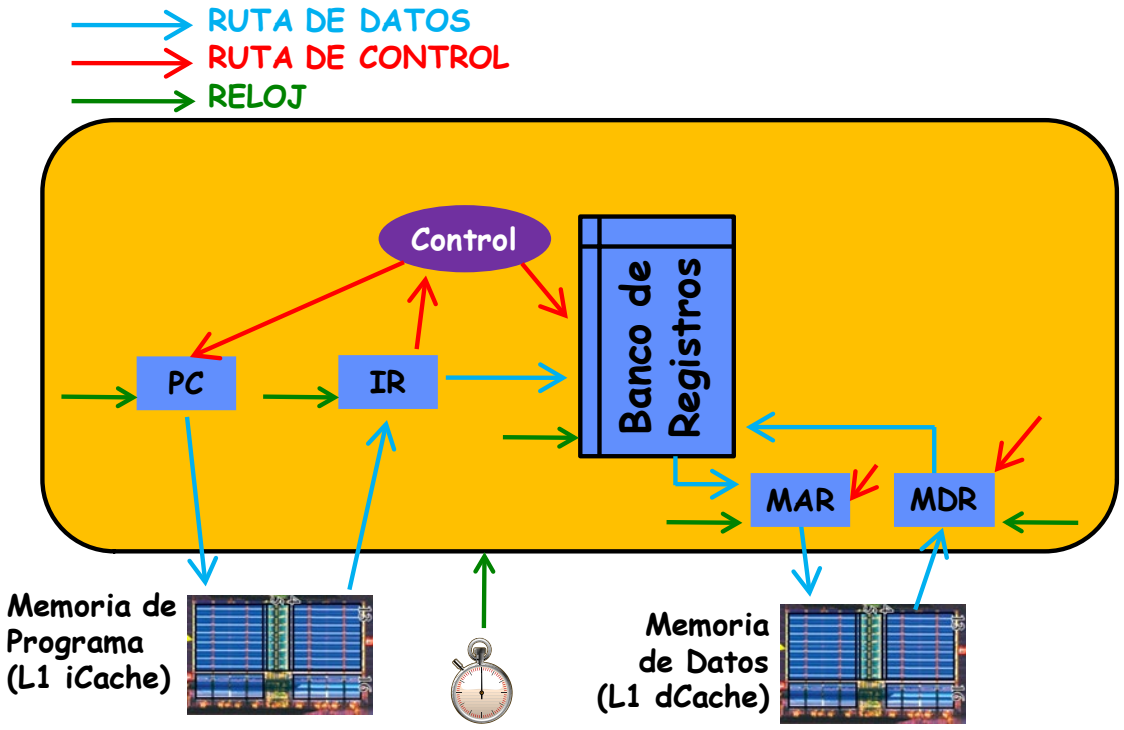
En la segunda etapa de decodificación y búsqueda de operandos, la instrucción almacenada en el registro de instrucción IR indica en qué registros del banco de registros se encuentran los operandos sobre los que se va a realizar la operación aritmético-lógica en la siguiente (tercera) etapa de la ejecución de la instrucción.

En la tercera etapa de ejecución de la instrucción, la unidad aritmético-lógica realiza la operación que le indica la unidad de control. Esta operación se establece a partir de la decodificación del código almacenado en el registro de instrucción IR. La operación se realiza sobre los operandos que le proporciona el banco de registros en la etapa anterior de decodificicación.

En la cuarta y última etapa de post-escritura, el resultado que proporciona la unidad aritmético-lógica se guarda en otro registro del banco de registros. Este registro donde se guarda el resultado también está indicado en el código de instrucción. Con esta etapa termina la ejecución de la instrucción aritmético-lógica.

Para poder ejecutar la siguiente instrucción, el registro interno contador de programa PC se actualiza con la siguiente dirección consecutiva de la memoria de programa.

Ejecución de una instrucción de carga desde memoria



Vamos a describir ahora la evolución de la ejecución de las instrucciones de carga; es decir, las que copian el contenido de un bloque de la memoria de

datos en un registro interno del banco de registros. La descripción se realizará también en el orden que determinan las cuatro sucesivas etapas de ejecución de las instrucciones.

En la etapa de búsqueda de la instrucción de carga, el contador de programa PC apunta a la dirección de memoria de programa donde suponemos que existe una instrucción de carga. La memoria de programa proporciona el código de la instrucción que se almacena temporalmente en el registro de instrucción IR.

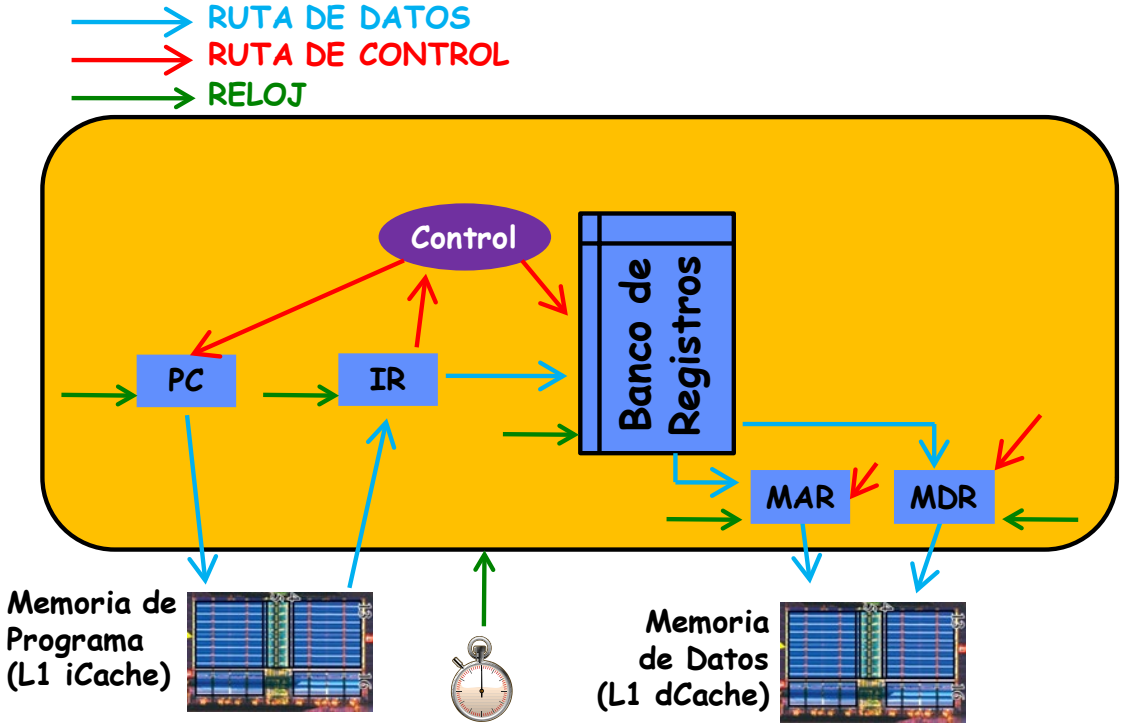
En la segunda etapa de decodificación y búsqueda de operandos, la instrucción almacenada en el registro de instrucción IR indica en qué registro del banco de registros se encuentra la dirección de la memoria de datos que apunta al bloque de memoria que se copiará posteriormente en un registro del banco de registros. Esta dirección de la memoria de datos se almacena al final de esta segunda etapa de ejecución de la instrucción en el registro interno MAR para ser utilizado en la siguiente (tercera) etapa de ejecución de la instrucción de carga.

En la tercera etapa de ejecución de la instrucción, la memoria de datos toma la dirección apuntada por el registro de dirección MAR y proporciona un bloque de datos, quedando este almacenado temporalmente en el registro interno de datos de memoria MDR al finalizar la tercera etapa de ejecución de la instrucción de carga.

En la cuarta y última etapa de post-escritura, el bloque de datos almacenado temporalmente en el registro de datos de memoria MDR se guarda en un registro del banco de registros. Este registro resultado se indica en el código de instrucción. Con esta etapa termina la ejecución de la instrucción de carga.

Para poder ejecutar la próxima instrucción, el registro interno contador de programa PC se actualiza con la siguiente dirección consecutiva de la memoria de programa.

Ejecución de una instrucción de almacenamiento en memoria

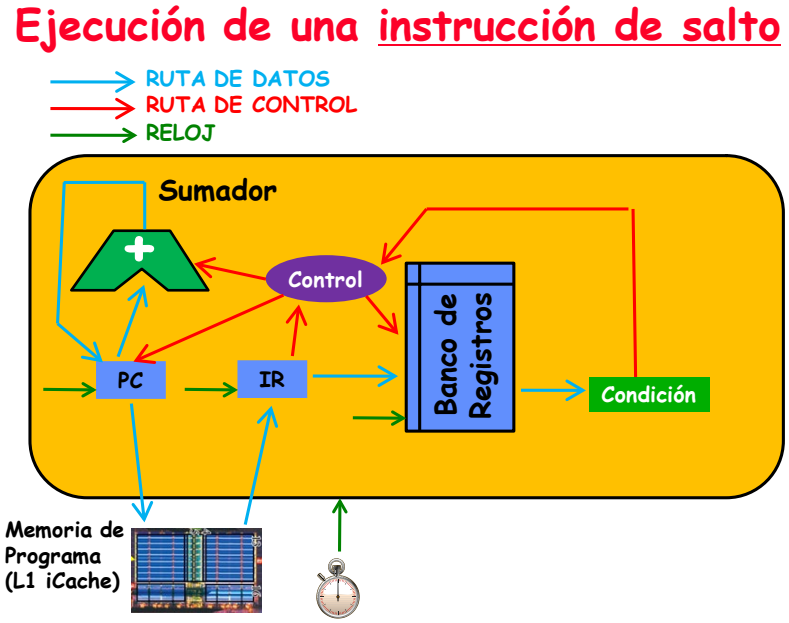


Vamos a describir ahora las fases de ejecución de las instrucciones de almacenamiento; es decir, las que copian el contenido de un registro interno del banco de registros en un bloque de la memoria de datos. La descripción se realizará también en el orden que determinan las sucesivas etapas de ejecución de las instrucciones.

En la primera etapa de búsqueda de la instrucción, el contador de programa PC apunta a la dirección de la memoria de programa donde suponemos la existencia de una instrucción de almacenamiento. La memoria de programa proporciona el código de la instrucción que se almacena temporalmente en el registro de instrucción IR.

En la segunda etapa de decodificación y búsqueda de operandos, el código de la instrucción almacenado en el registro de instrucción indica en qué registro se encuentra la dirección de la memoria de datos que apunta al bloque de memoria donde se copiará el contenido de otro registro del banco de registros. Esta dirección y dato a almacenar se guardan temporalmente al final de esta segunda etapa de ejecución en los registros MAR y MDR respectivamente, para ser utilizados en la siguiente etapa de ejecución de esta instrucción.

En la tercera etapa de ejecución de la instrucción, la memoria de datos toma el bloque de datos guardado temporalmente en el registro MDR y lo almacena en la dirección apuntada por el registro de dirección MAR. Esto se termina de hacer al finalizar esta tercera etapa de ejecución de la instrucción. En la última y cuarta etapa de post-escritura, esta instrucción no realiza ninguna operación. Para poder ejecutar la próxima instrucción, el registro interno contador de programa PC se actualiza con la siguiente dirección consecutiva de la memoria de programa al finalizar la cuarta etapa de post-escritura.



¿Cuáles son las unidades funcionales del procesador que se activan cuando se ejecutan las instrucciones de salto? Describe la funcionalidad que realiza cada unidad funcional en la ejecución de este tipo de instrucción.

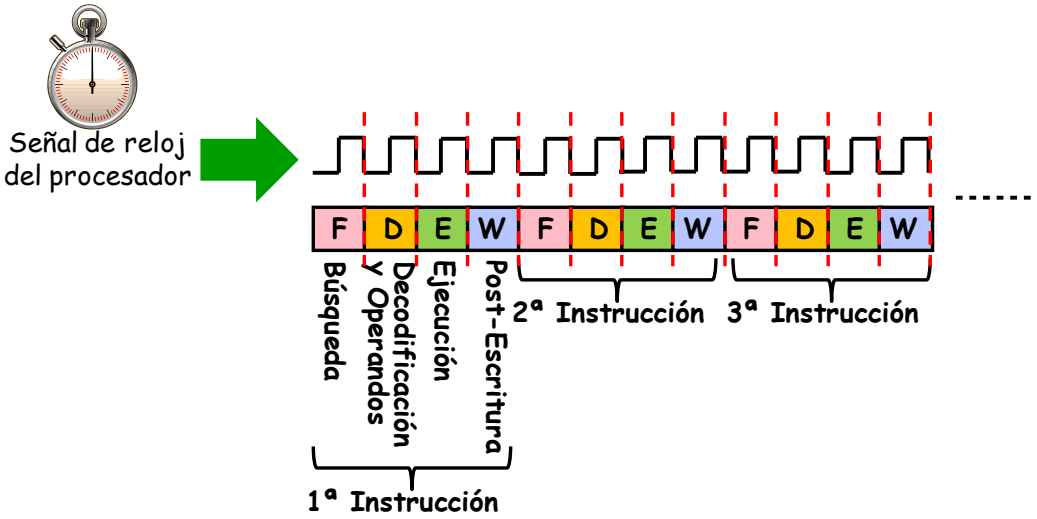
Vamos a describir ahora las instrucciones de salto; es decir, las que indican explícitamente la dirección de la memoria de programa donde se encuentra la siguiente instrucción que se debe ejecutar dentro del procesador. La descripción la vamos a realizar también en el orden que determinan las sucesivas etapas de ejecución de las instrucciones.

En la primera etapa de búsqueda de la instrucción, el contador de programa PC apunta a la dirección de memoria de la instrucción de salto. La memoria de programa proporciona el código de la instrucción que se almacena temporalmente en el registro de instrucción IR.

En la segunda etapa de decodificación y búsqueda de operandos, la instrucción almacenada en el registro de instrucción IR indica qué registro del banco de registros es el que se utiliza para saber si se cumple la condición de las instrucciones de salto condicional. En las instrucciones de salto incondicional, el banco de registros no se usa.

En la tercera etapa de ejecución de la instrucción, la unidad de control recibe el resultado del **test de la condición** que caracteriza a las instrucciones de salto condicional y si se cumple, se activa un pequeño sumador que calcula la dirección de la memoria de programa donde se encuentra la siguiente instrucción a ejecutar. En el caso de saltos incondicionales la unidad de control directamente activa el sumador que calcula la siguiente dirección del contador de programas PC. Al final de esta tercera etapa de ejecución es cuando se actualiza el PC y por tanto, termina la ejecución de este tipo de instrucción. La próxima instrucción a ejecutar es la que indica el nuevo contenido del registro contador de programa PC.

Ejecución MULTICICLO de instrucciones



Esta transparencia muestra el resumen de las etapas que secuencialmente atraviesa la ejecución de cada instrucción y que fueron descritas en transparencias anteriores. A partir de ahora y hasta el final de esta lección, vamos a utilizar un conjunto de siglas para representar a cada etapa de ejecución de una instrucción.

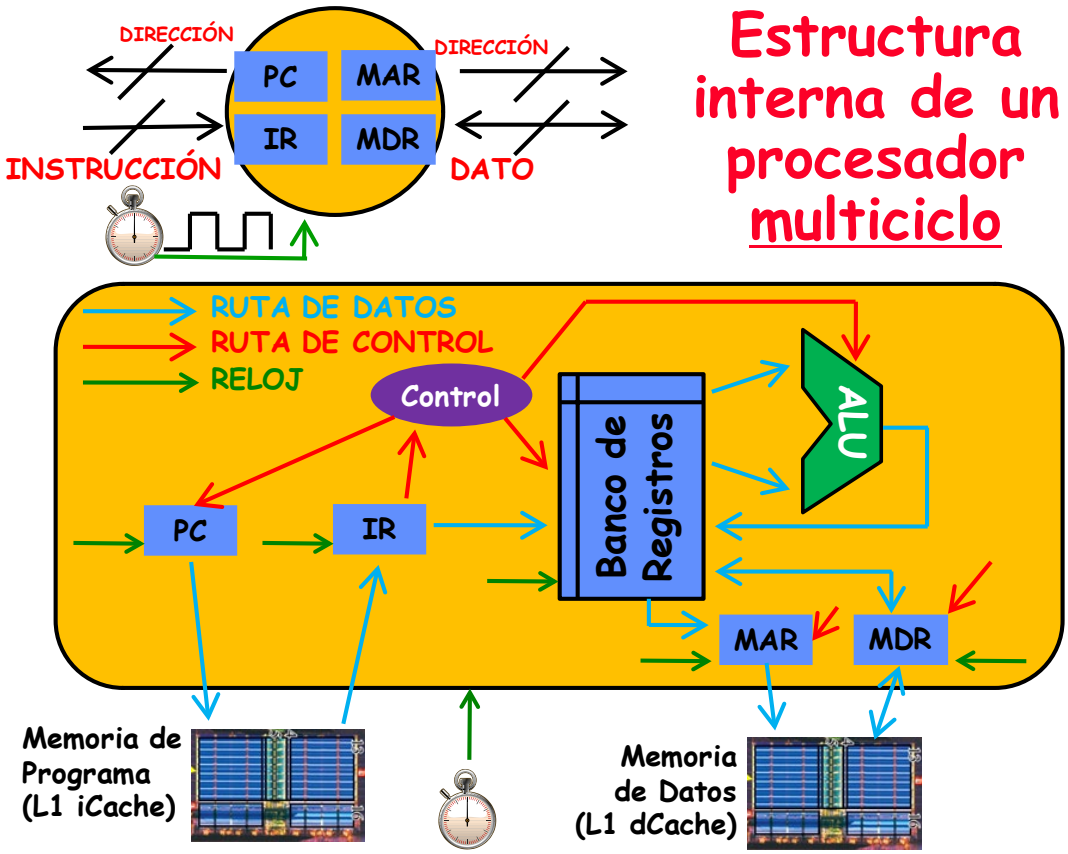
- **F** representa a la etapa de Búsqueda de la Instrucción
- **D** representa a la Descodificación y Búsqueda de Operandos
- **E** representa a la etapa de Ejecución

- **W** representa a la etapa de Post-escritura

Después de una primera instrucción, el contador de programa PC se actualiza para buscar, decodificar, ejecutar y post-escribir el resultado de una segunda instrucción. Y a esto le sigue una tercera instrucción, y sucesivas instrucciones hasta terminar todas las instrucciones del programa informático, en el caso de que todo funcione correctamente.

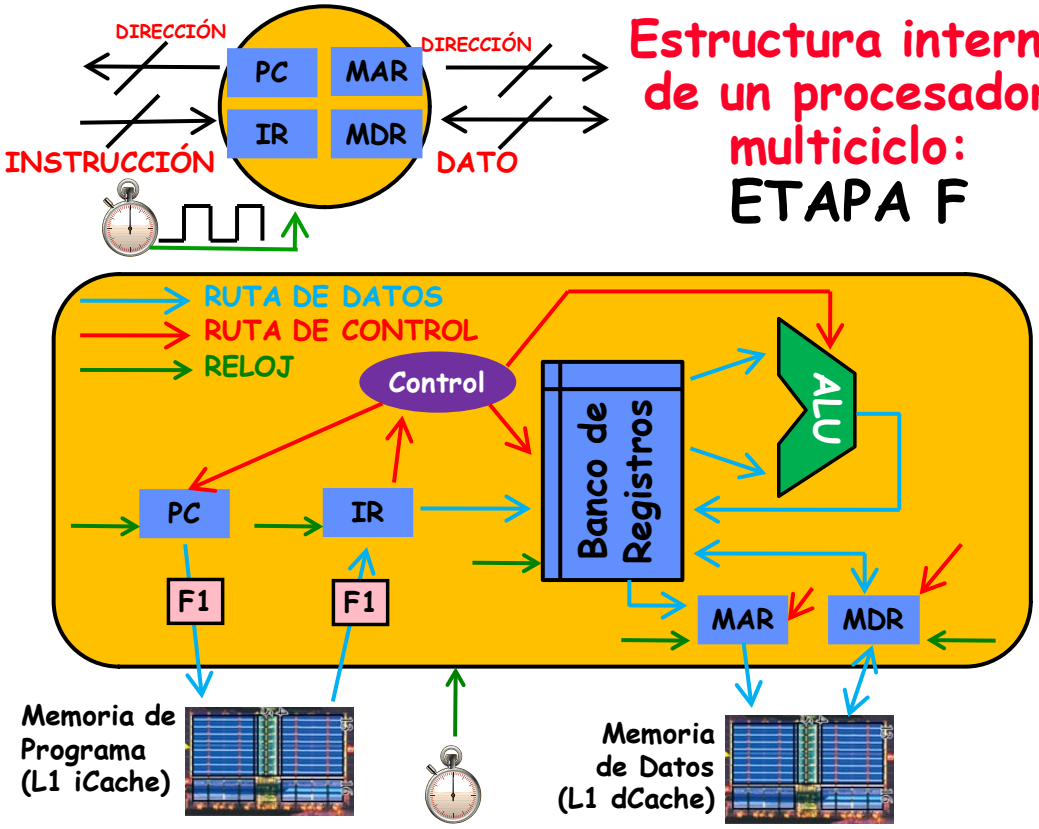
Este tipo de ejecución de las instrucciones es secuencial y escalar como se ha indicado previamente. Pero además, se dice que es **multiciclo** si el tiempo en el que se realiza cada etapa de ejecución de las instrucciones coincide con un ciclo distinto pero consecutivo de la señal de reloj del procesador (ver la transparencia).

En las siguientes transparencias se indica cuál es la parte interna del procesador que se activa en cada ciclo de reloj durante una ejecución multiciclo de las instrucciones.

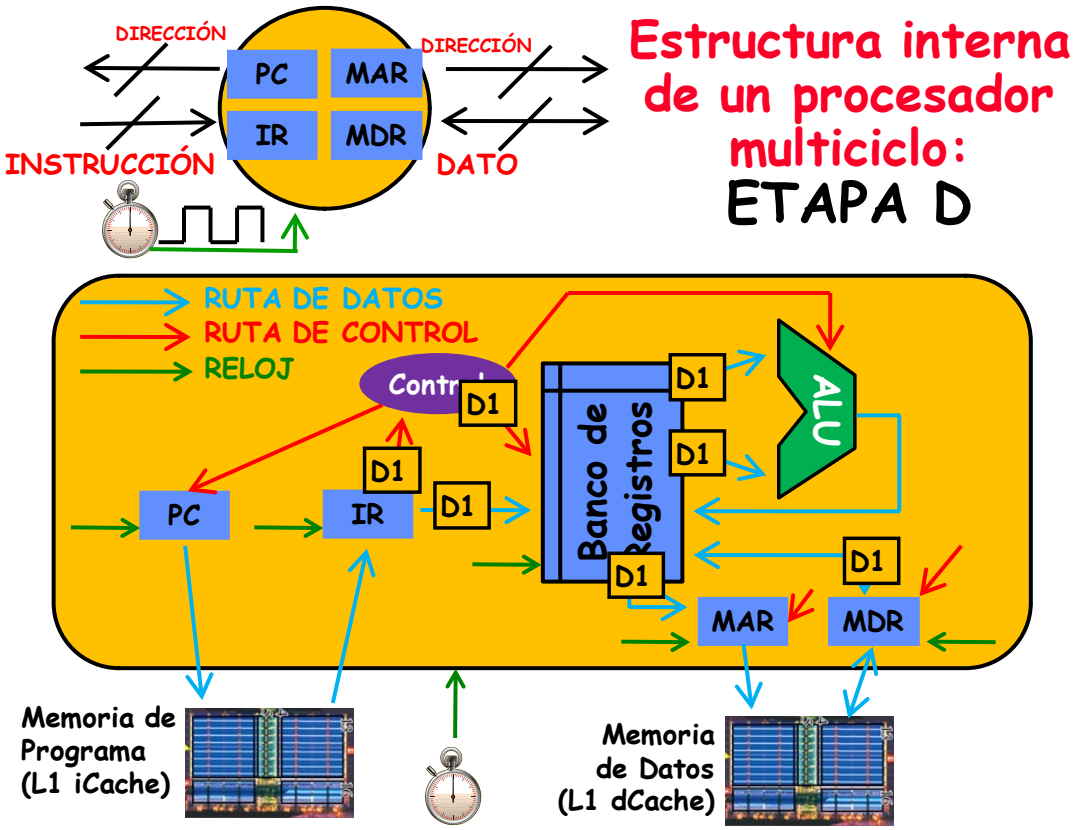


Para ello vamos a utilizar el diagrama que se muestra en esta transparencia, en el cual aparecen representadas todas las unidades funcionales del procesador que intervienen en las etapas de ejecución de las instrucciones.

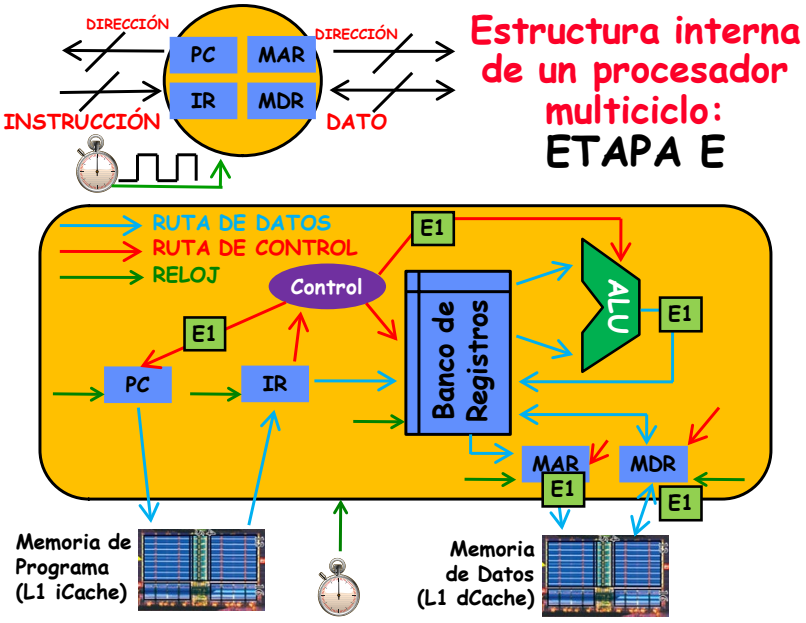
Suponer ahora que se va a ejecutar una nueva instrucción. Vamos a suponer que esta nueva instrucción es la primera de una secuencia de varias instrucciones.



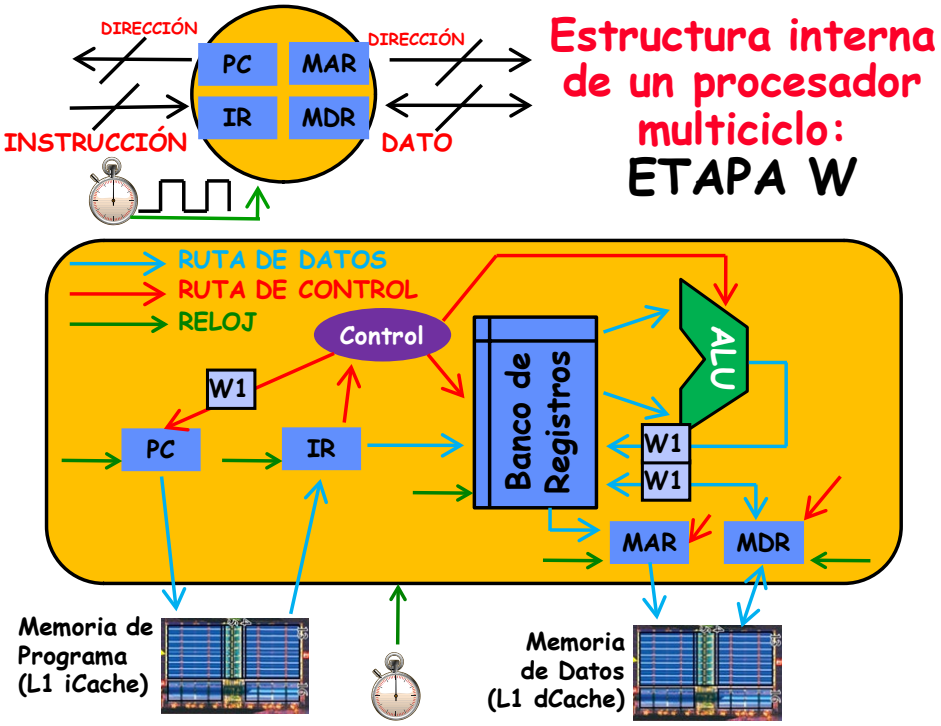
En un primer ciclo de reloj se activa la parte del procesador que se dedica a la búsqueda de esta primera instrucción (etapa F). Observar que F1 resalta la parte de la ruta de datos del procesador que está involucrada en esta primera fase de la ejecución de esta primera instrucción.



En un <u>segundo ciclo de reloj</u>, esta primera instrucción se decodifica y busca los operandos (etapa D) que necesitará en la tercera etapa de ejecución de la instrucción. Observar que D1 resalta la parte de la ruta de datos que se activa en esta segunda etapa de ejecución de la primera instrucción.

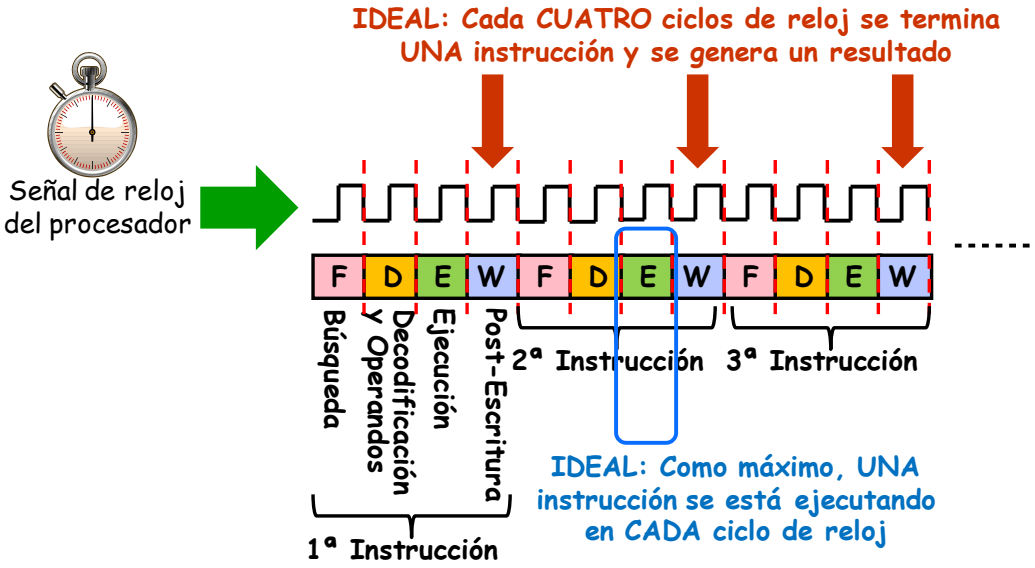


En el tercer ciclo de reloj la primera instrucción se ejecuta, bien en la unidad funcional que hemos denominado aritmético-lógica (ALU), bien accediendo a la memoria de datos a través de los registros MAR y MDR, o bien generando el nuevo contador de programa PC a través de un instrucción de salto. Observar que E1 resalta la parte de la ruta de datos que se activa en esta tercera etapa de ejecución de la primera instrucción.



Y por último en el cuarto ciclo de reloj, se guarda el resultado de la instrucción aritmético-lógica o de carga desde memoria en el banco de registros, a la vez que se actualiza el contador de programa PC para empezar a ejecutar una segunda instrucción. Esta actualización del contador de programa se realiza en el ciclo anterior cuando la instrucción es de tipo salto, tanto condicional como incondicional. Observar que W1 resalta la parte de la ruta de datos que se activa en esta cuarta etapa de ejecución de la primera instrucción.

Ejecución SECUENCIAL de instrucciones



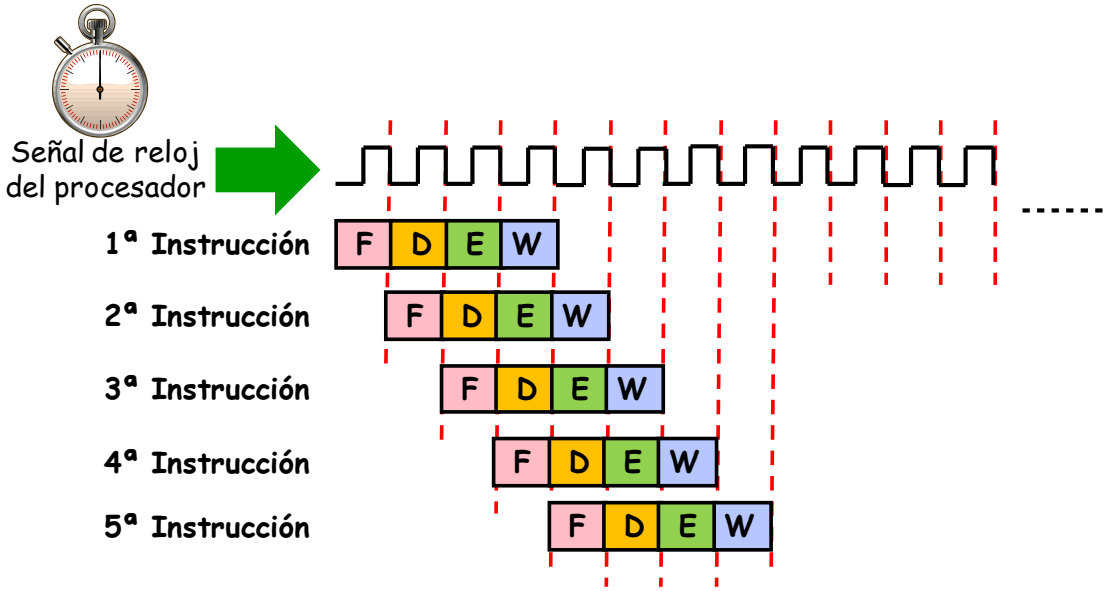
Volvemos a observar aquí el resumen de la ejecución de instrucciones de tipo secuencial, escalar y multiciclo. Sobre este diagrama se resaltan dos resultados de este tipo de ejecución de las instrucciones.

- El primero de ellos es que un nuevo resultado se genera cada vez que termina la ejecución de una instrucción, es decir, cada cuatro ciclos en el caso del procesador que hemos descrito previamente.

- El segundo resultado que se destaca es que en cada ciclo de reloj, dentro del procesador solo se encuentra ejecutándose una sola instrucción.

Ambos resultados son característicos de la ejecución secuencial, escalar y multiciclo de las instrucciones.

Ejecución SEGMENTADA de instrucciones



Veamos ahora otro tipo de ejecución secuencial y escalar de las instrucciones dentro del procesador. Sin embargo en este caso, en vez de ser multiciclo se denomina **segmentada**. Este tipo de ejecución segmentada de las instrucciones también se sincroniza con la señal de reloj del procesador.

La ejecución segmentada de la primera instrucción coincide a la forma indicada en la ejecución multiciclo. Durante cuatro ciclos sucesivos, esta primera instrucción atraviesa las cuatro partes internas del procesador que se activan en las cuatro etapas en las que se ejecutan las instrucciones.

Es a partir de la segunda instrucción cuando se observan las diferencias entre la ejecución multiciclo y la ejecución segmentada. La segunda instrucción se introduce en el procesador iniciando su etapa de búsqueda (F) cuando la instrucción anterior se encuentra en la segunda etapa de decodifición y búsqueda de operandos (D). A medida que la instrucción anterior pasa a la siguiente fase de ejecución, la segunda instrucción también avanza a la siguiente fase de ejecución.

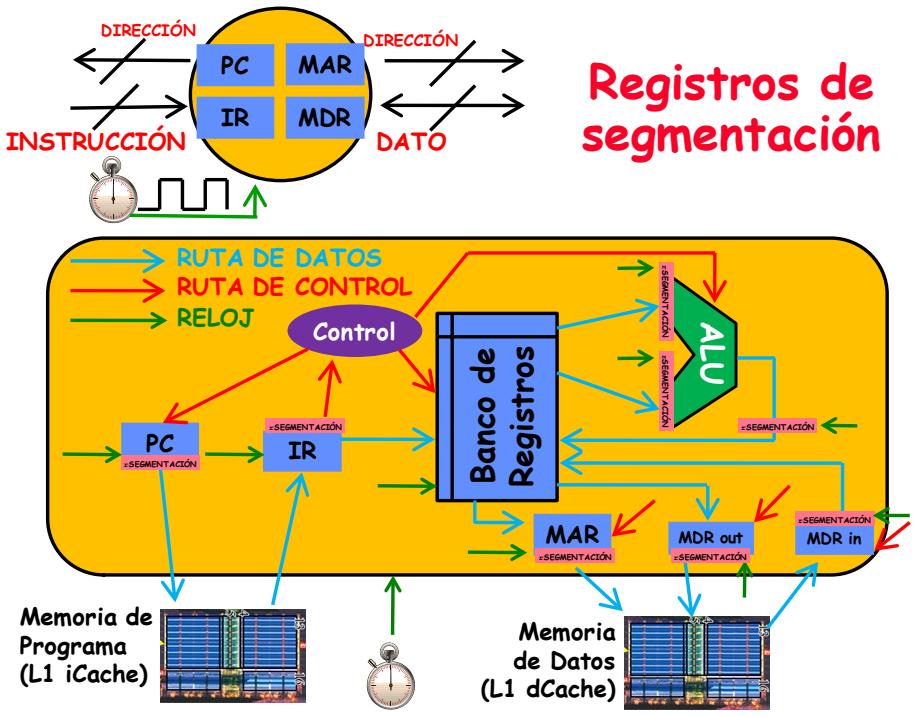
Una tercera instrucción entra en el procesador cuando la segunda instrucción se encuentra en la etapa de decodificación y búsqueda de operandos (D) y la primera instrucción se encuentra en la etapa de ejecución (E). De la misma forma que la segunda instrucción, la tercera instrucción avanza a la siguiente etapa de ejecución cuando la instrucción anterior ha avanzado una fase de ejecución.

Observar en la transparencia que en cada ciclo de reloj de la ejecución segmentada, se solapa en el tiempo la ejecución de más de una instrucción dentro del procesador, pero cada instrucción se encuentra en una etapa de ejecución distinta.

En un primer ciclo de reloj, sólo la primera instrucción se encuentra ejecutándose dentro del procesador, concretamente en la etapa de búsqueda (F). En un segundo ciclo de reloj, la primera instrucción se encuentra ejecutándose en la etapa de decodificación y búsqueda de operandos (D), a la vez que una segunda instrucción entra a ejecutarse en el procesador, permaneciendo esta segunda instrucción en la primera etapa de ejecución (F).

En el tercer ciclo de reloj, la primera instrucción pasa a la etapa de ejecución (E), la segunda instrucción pasa a decodificiación y búsqueda de operandos (D) y una tercera instrucción empieza a ejecutarse en la etapa de búsqueda de la instrucción (F).

En un cuarto ciclo de reloj, la primera instrucción pasa a la cuarta etapa de post-escritura (W), la segunda instrucción pasa a la tercera etapa de ejecución (E), la tercera instrucción pasa a la etapa de descodificación y búsqueda de operandos (D), y una cuarta instrucción entra a ejecutarse en la etapa de búsqueda de instrucción (F). Y así sucesivamente con las siguientes instrucciones.



Para poder realizar la ejecución segmentada de las instrucciones tal y como se acaba de describir, es necesario aislar cada parte de la ruta de datos del procesador. Para ello, se utilizan los denominados **registros de segmentación**, cuya posición dentro del procesador se indica en esta transparencia.

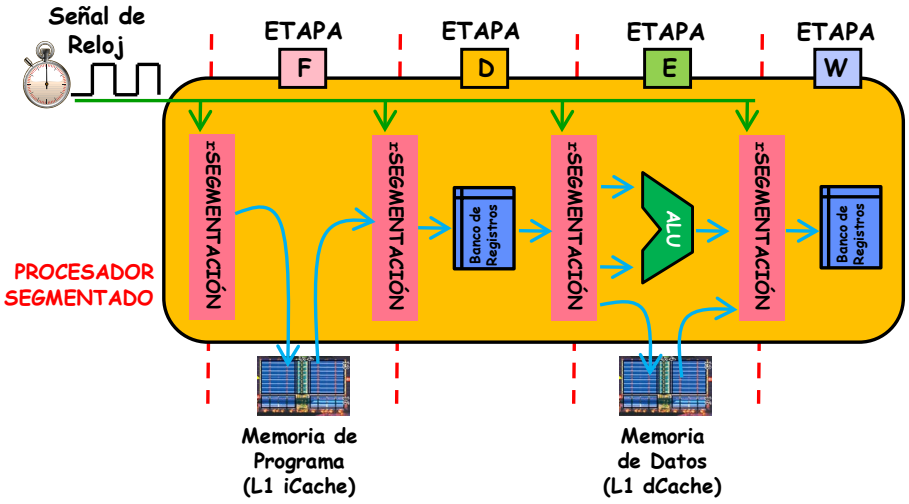
Observar que existen cinco registros de segmentación (rSEGMENTACIÓN):

- A la salida del contador de programa PC. Este registro de segmentación permite separar la actividad de las señales de la etapa de post-escritura (W) de la actividad de las señales involucradas en la etapa de búsqueda (F). De esta forma pueden convivir dentro del procesador una instrucción en la etapa de post-escritura con otra instrucción diferente en la etapa de búsqueda.
- A la salida del registro de instrucciones IR. Este registro de segmentación permite separar la actividad de las señales de la etapa de búsqueda (F) de la actividad de las señales involucradas en la etapa de decodificación y búsqueda de operandos (D). De esta forma pueden convivir dentro del procesador una instrucción en la etapa de búsqueda con otra instrucción diferente en la etapa de descodificación y búsqueda de operandos.
- A la salida del banco de registros, que coincide con la entrada de la unidad aritmético-lógica, y con la salida de los registros MAR y MDR. Este último registro MDR se utiliza para alojar temporalmente el dato que se guarda en las instrucciones de almacenamiento. En la transparencia se renombra a este registro como *MDR out* cuando se utiliza para guardar el dato involucrado en la instrucción de almacenamiento. Este registro de segmentación permite separar la actividad de las señales de la etapa de decodificación y búsqueda de operandos (D) de la actividad de las señales involucradas en la etapa de ejecución (E). De esta forma pueden convivir dentro del procesador una instrucción en la etapa de decodificación y búsqueda de operandos con otra instrucción diferente en la etapa de ejecución.
- A la salida de la unidad aritmético-lógica. Este registro de segmentación permite separar la actividad de las señales de la etapa de ejecución (E) de la actividad de las señales involucradas en la etapa de post-escritura (W). De esta forma pueden convivir dentro del procesador una instrucción en la etapa de ejecución con otra instrucción diferente en la etapa de post-escritura.
- A la salida del registro MDR cuando se ejecuta una instrucción de carga. En la transparencia se renombra a este registro como *MDR in* cuando se utiliza para guardar el dato enviado desde la memoria en la instrucción

de carga. Este registro de segmentación también permite separar la actividad de las señales de la etapa de ejecución (E) de la actividad de las señales involucradas en la etapa de post-escritura (W).

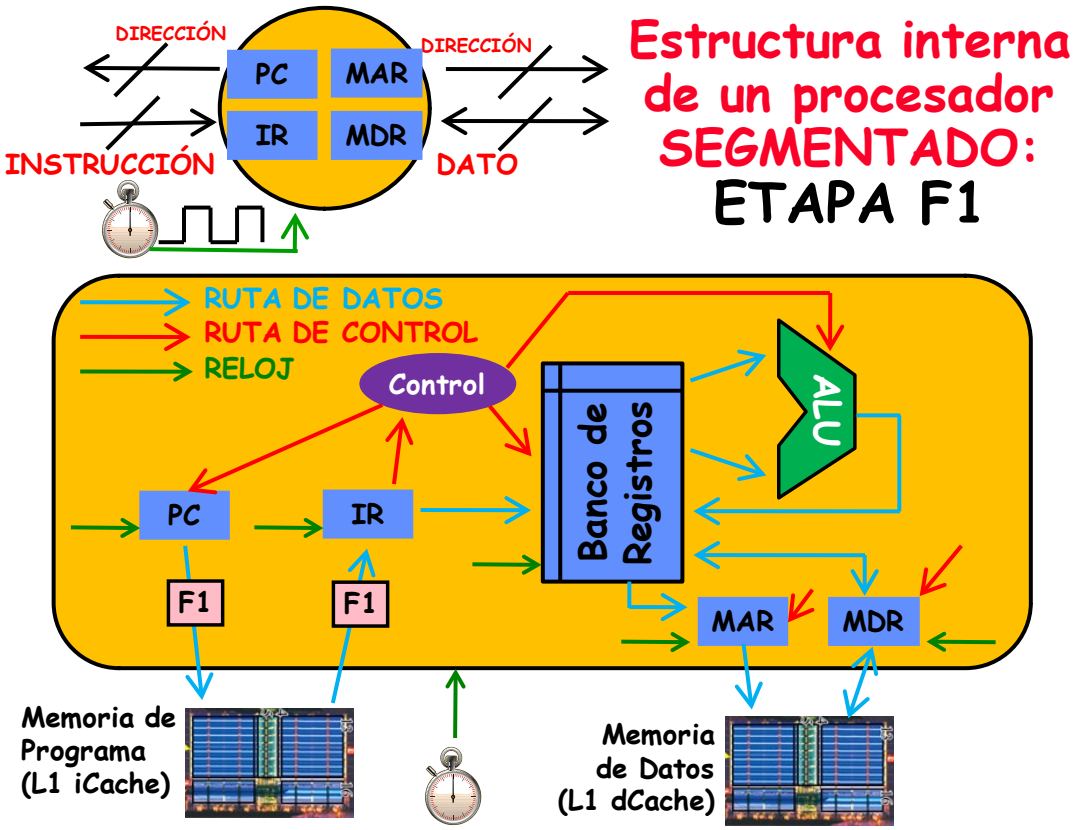
La actualización de estos registros de segmentación se realiza de forma unísona y sincronizada con un flanco de la señal de reloj del procesador.

Estructura interna simplificada de un procesador segmentado



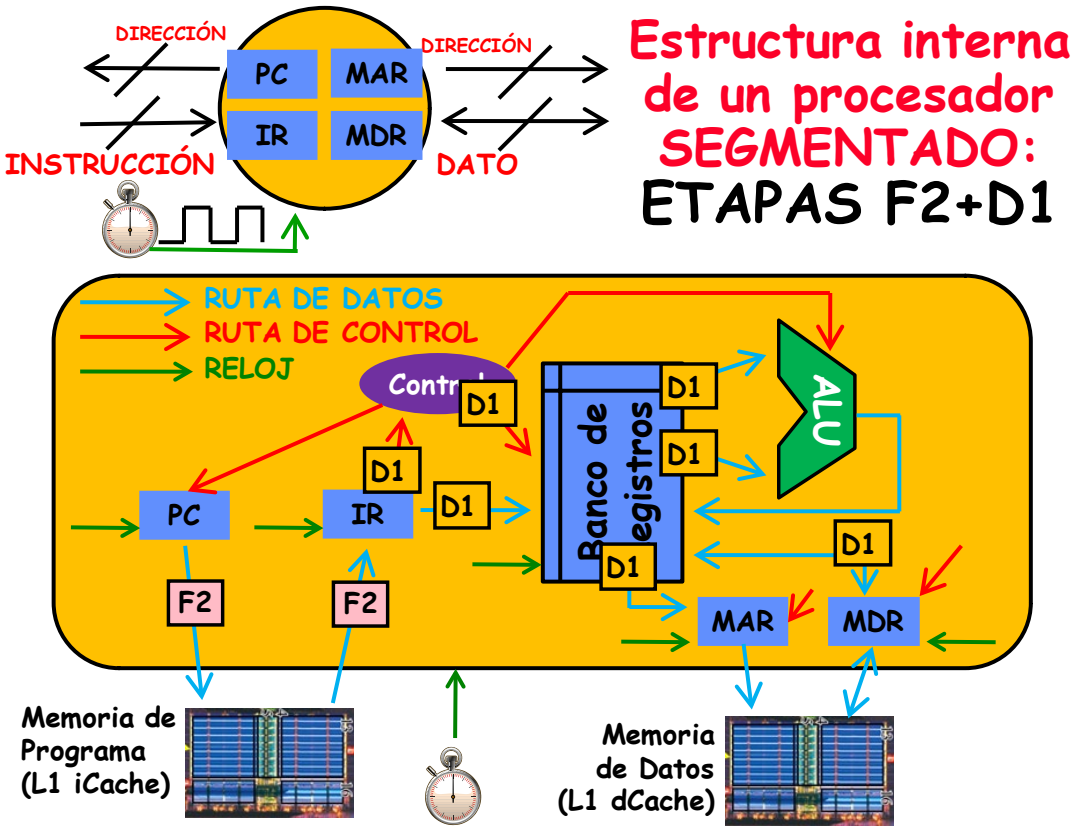
En esta transparencia se representa la misma ruta de datos segmentada de transparencias anteriores pero de forma aún más simplificada. En ella se resaltan exclusivamente los registros de segmentación (rSEGMENTACIÓN), la señal de reloj que sincroniza al unísono estos registros de segmentación, las memorias de programa y datos (en este caso las hacemos coincidir con la memorias cache de instrucciones y datos respectivamente), el banco de registros, y la unidad aritmético-lógica (ALU).

Observar que en esta representación aparece una letra que identifica a cada una de las etapas de ejecución de la instrucción situada entre dos registros de segmentación. Esto quiere representar que el hardware situado entre dos registros de segmentación se encarga de implementar exclusivamente las actividades asociadas a la correspondiente fase de ejecución de las instrucciones.



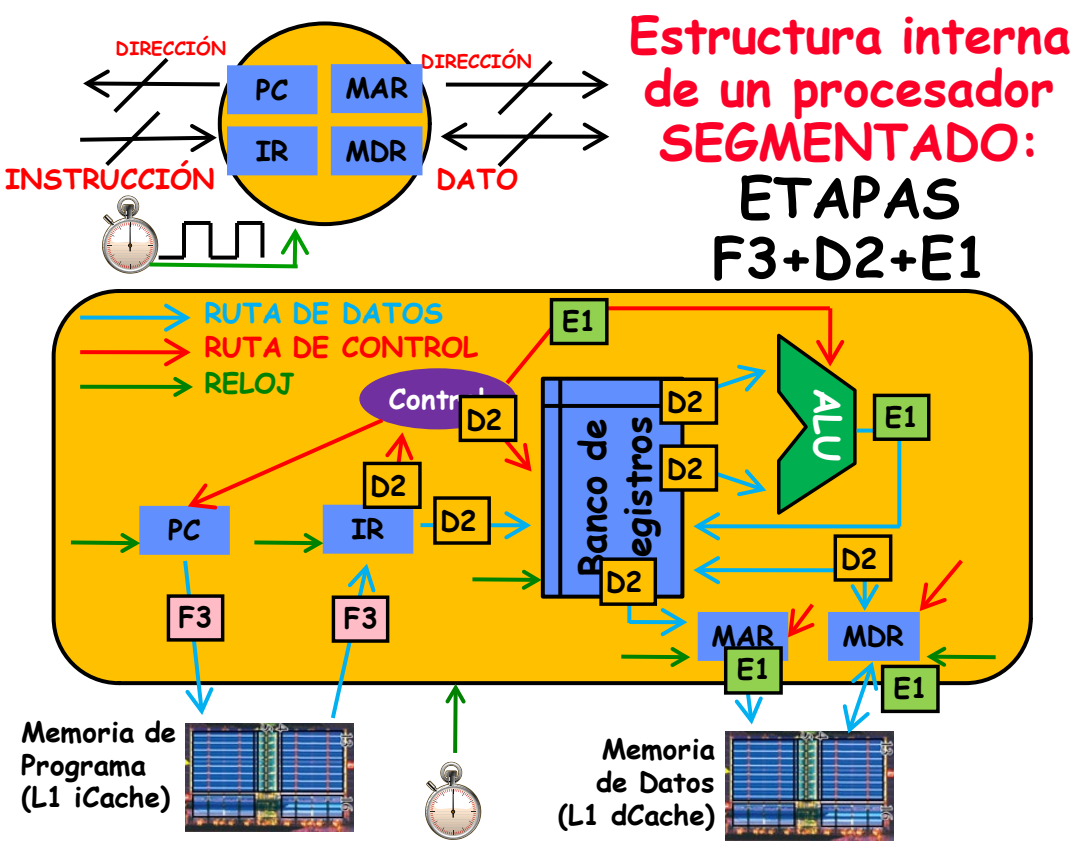
Vamos a utilizar a partir de ahora la misma representación del procesador que hemos utilizado en esta lección para indicar qué partes del procesador se activan durante la ejecución segmentada de las instrucciones.

En un primer ciclo de reloj, la primera instrucción se encuentra en la etapa de búsqueda de instrucción y por tanto la parte de la ruta de datos interna del procesador que agrupa los registros PC, la memoria de programa y el registro IR se activan para la primera instrucción. El símbolo F1 quiere resaltar esta parte de la ruta de datos del procesador.



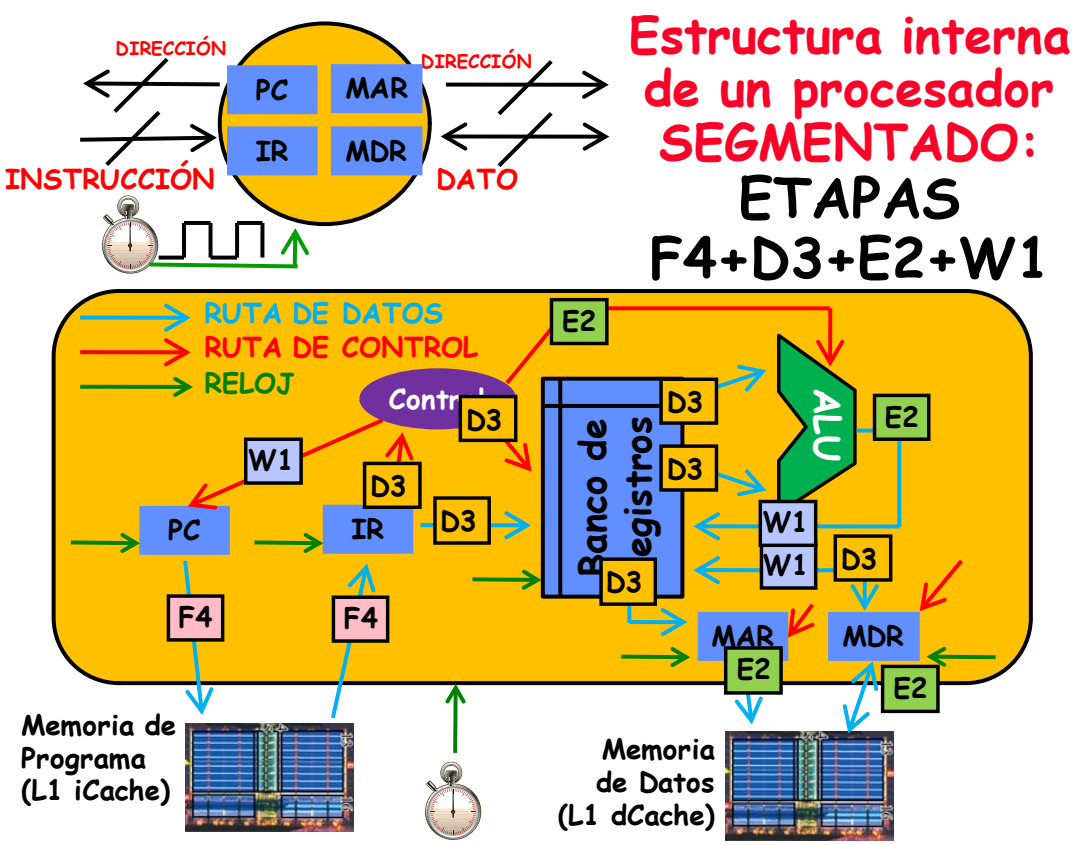
En un <u>segundo ciclo de reloj</u>, la primera instrucción se encuentra en la etapa de decodificación y búsqueda de operandos (D1) y por eso se activa la parte del procesador correspondiente. Y en ese mismo ciclo, se activa la parte del procesador correspondiente a la búsqueda de instrucciones para una segunda instrucción (F2).

D1 resalta la parte de la ruta de datos del procesador que se activa para la primera instrucción, y F2 la parte que se activa para la segunda instrucción. Ambas partes <u>se activan a la vez durante un mismo ciclo de reloj</u> en la ejecución segmentada.



En un tercer ciclo de reloj, la primera instrucción pasa a la etapa de ejecución y por eso se activa para esa instrucción la parte del procesador correspondiente (E1). En ese mismo ciclo de reloj, la segunda instrucción se encuentra en la etapa de decodificación y búsqueda de operandos. Por eso se activa la parte del procesador correspondiente (D2). Y en ese mismo ciclo de reloj, se activa la parte del procesador correspondiente a la búsqueda de instrucciones para una tercera instrucción (F3).

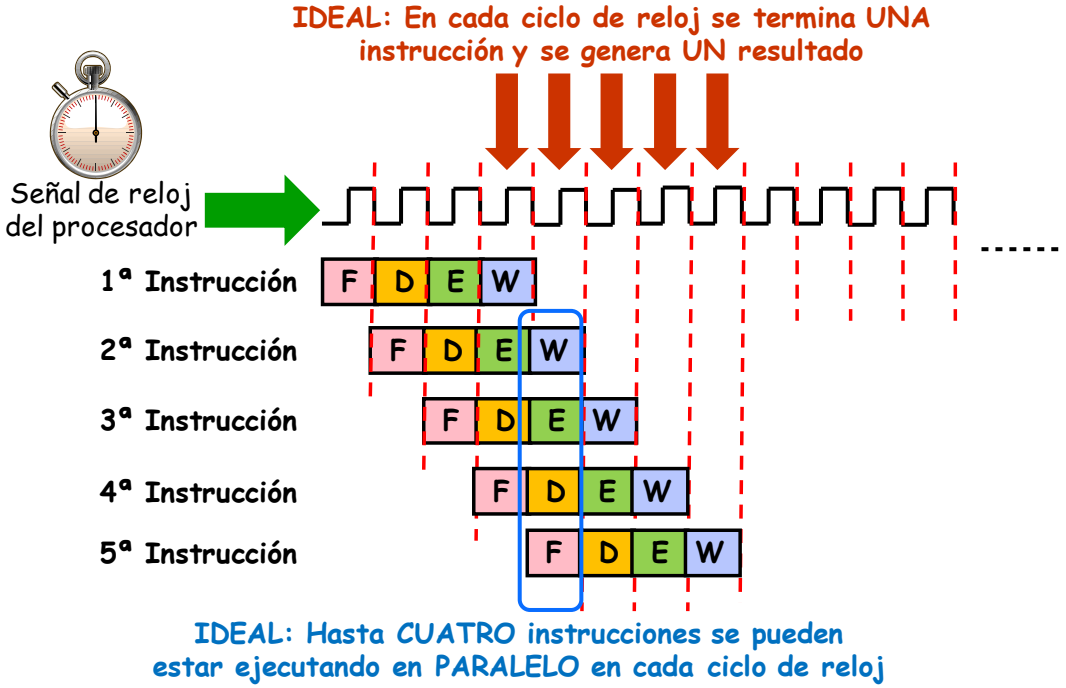
E1 resalta la parte de la ruta de datos del procesador que se activa para la primera instrucción, D2 la parte de la ruta de datos que se activa para la segunda instrucción y F3 la parte de la ruta de datos que se activa para la tercera instrucción. Las tres partes internas del procesador se activan a la vez durante un mismo ciclo de reloj en la ejecución segmentada de las instrucciones.



En un cuarto ciclo de reloj, la primera instrucción pasa a la última etapa de post-escritura y se actualiza el contador de programa (W1). En ese mismo ciclo de reloj, la segunda instrucción pasa a la etapa de ejecución y por eso se activa para esa instrucción la parte del procesador correspondiente (E2). En ese mismo ciclo de reloj, la tercera instrucción se encuentra en la etapa de decodificación y búsqueda de operandos. Por eso se activa la parte del procesador correspondiente (D3). Y en ese mismo ciclo de reloj, se activa la parte del procesador correspondiente a la búsqueda de instrucciones para una cuarta instrucción (F4).

Observar que en este ciclo de reloj, dentro del procesador existen cuatro instrucciones ejecutándose a la vez, cada una de ellas en una etapa de ejecución distinta.

Ejecución SEGMENTADA de instrucciones



En esta transparencia queremos destacar dos resultados de la ejecución segmentada de las instrucciones.

El primero es que los resultados proporcionados por las distintas instrucciones se obtienen en ciclos de reloj sucesivos, y no cada cuatro ciclos como en el caso de la ejecución multiciclo de instrucciones. Esto es debido a que en cada ciclo de reloj a partir de uno dado se termina de ejecutar una instrucción distinta.

Por otro lado, se observa que en cada ciclo de reloj se están ejecutando dentro del procesador hasta cuatro instrucciones en paralelo, y no una como en el caso de ejecución multiciclo de las instrucciones. Este número coincide con el número de etapas en las que se divide de forma generalizada la ejecución de instrucciones. A las partes internas del procesador segmentado que se activan independientemente para distintas instrucciones se denominan **etapas de segmentación** o simplemente **segmentos del procesador**. Cada etapa de segmentación está separada de la siguiente por un registro de segmentación.

NIOS II

Nios II	Nios II/f "Fast"	Nios II/s "Standard"	Nios II/e "Economy"
Pipeline	6-stage	5-stage	none
Max Frequency$_1$	200 MHz	180 MHz	210 MHz
Max D-MIPS$_1$	225	130	30
Size (4-input LUTs)	1800	1200	600
Branch Prediction	Dynamic	Static	no
I-Cache	Up to 64K	Up to 64K	no
D-Cache	Up to 64K	no	no

1. Characteristics in Stratix II 90nm FPGA

Volvemos a mostrar una tabla resumen con las características del procesador NIOS II en la que se observan los tipos de ejecución de instrucciones de tres tipos diferentes de implementaciones del procesador: dos de ellas segmentadas, denominadas **fast** (NIOS II/f) y **standard** (NIOS II/s), y la otra multiciclo, denominada **economy** (NIOS II/e).

Observar que la implementación fast (NIOS II/f) tiene 6 etapas de segmentación, y la implementación standard (NIOS II/s) tiene 5 etapas de segmentación.

Observar también que de las versiones segmentadas, la que tiene más etapas de segmentación consigue alcanzar una mayor frecuencia de reloj: 200 MHz para el procesador fast y 180 MHz para el procesador standard. La justificación de este resultado la descubriremos en la siguiente lección.

Por último, observar que el procesador segmentado con 6 etapas de segmentación es el mejor en cuanto a prestaciones, es decir, que consigue realizar mayor cantidad de operaciones por segundo; 225 millones de las operaciones denominadas **D-MIPS** por segundo se pueden ejecutar en la versión fast (NIOS II/f), frente a 130 millones y 30 millones de operaciones D-MIPS por segundo en las otras dos implementaciones del procesador NIOS II.

El resultado que se observa en el procesador fast puede ser un poco extraño: consigue realizar una mayor cantidad de millones de operaciones por segundo (225 D-MIPS) que el número de millones de ciclos por segundo que indica su frecuencia máxima de funcionamiento (200 MHz). La justificación de este resultado se fundamenta en las optimizaciones que introduce el compilador en el momento de generar el código ejecutable. Muchas operaciones originales del programa de prueba son combinadas en el código ejecutable en un número menor de ellas, no contabilizándose el número de operaciones final que realmente se realizan durante la ejecución sino contabilizándose el número original de operaciones indicadas por programa.

EJERCICIOS DE AUTOEVALUACIÓN

1. ¿A qué se denomina Ruta de Control de un procesador? ¿Por qué es necesaria su existencia?
2. ¿A qué se denomina Ruta de Datos de un procesador? ¿Por qué es necesaria su existencia?
3. ¿Cómo afecta la segmentación del procesador en la frecuencia de reloj del procesador?
4. ¿Cómo afecta la segmentación del procesador en las prestaciones del procesador? Muestra un ejemplo en el que calcules la aceleración o speed-up de un procesador segmentado frente a uno multiciclo.
5. ¿Cómo se llaman las etapas que se atraviesan cuando se ejecuta una instrucción dentro del procesador?
6. ¿Cuáles son las principales unidades funcionales del procesador NIOS II?
7. ¿Cuáles son las unidades funcionales del procesador que se activan cuando se ejecutan las instrucciones aritmético-lógicas?
8. ¿Cuáles son las unidades funcionales del procesador que se activan cuando se ejecutan las instrucciones de carga?
9. ¿Cuáles son las unidades funcionales del procesador que se activan cuando se ejecutan las instrucciones de almacenamiento?
10. ¿Cuáles son las unidades funcionales del procesador que se activan cuando se ejecutan las instrucciones de salto?
11. ¿Cuáles son los registros internos del procesador que interaccionan directamente con las partes del computador que se conectan al procesador? ¿Qué función tiene asignada cada uno de ellos?
12. ¿Para qué se utiliza dentro del procesador la señal de reloj?
13. ¿Qué diferencias existen entre la ejecución de instrucciones de tipos multiciclo y segmentada?
14. Describe la estructura interna de un procesador cuya ejecución de instrucciones es secuencial, escalar y multiciclo.
15. En la ejecución multiciclo, ¿cuántas instrucciones se ejecutan en cada ciclo de reloj?
16. En la ejecución multiciclo, ¿cuántos resultados se obtienen en cada ciclo de reloj?
17. En la ejecución segmentada, ¿cuántas instrucciones se ejecutan en cada ciclo de reloj?
18. En la ejecución segmentada, ¿cuántos resultados se obtienen en cada ciclo de reloj?

Bibliografía:

Nios II Core Implementation Details, http://www.altera.com/literature/hb/nios2/n2cpu_nii51015.pdf

Dhrystone, http://en.wikipedia.org/wiki/Dhrystone

Dhrystone Benchmark Results On PCs, http://www.roylongbottom.org.uk/dhrystone%20results.htm

Imagen alu: http://t0.gstatic.com/images?q=tbn:ANd9GcQpFxOEMzPy4AlxReq-XYEvib1_NQpSYQIwB3mt7kbcHtb1CnYm6A