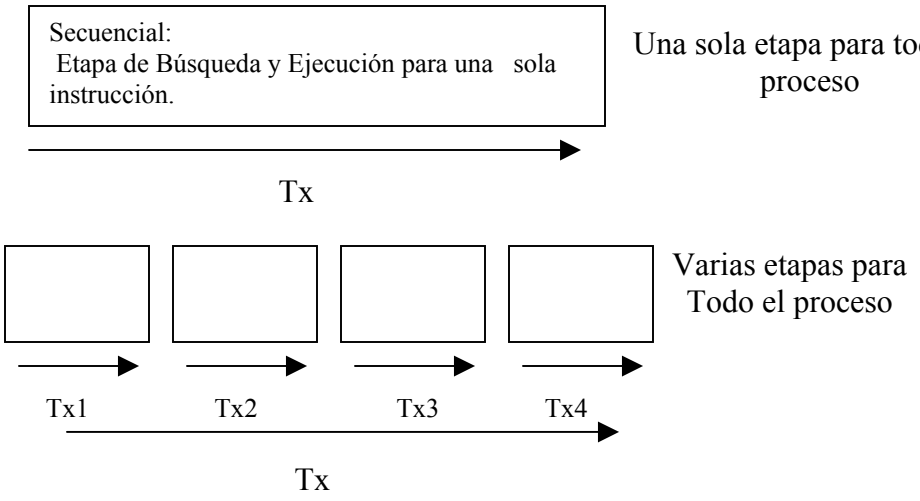
Introducción a la Segmentación.

Es el proceso de ejecutar varias instrucciones sin necesidad de que la primera instrucción se ejecute en su totalidad. Un ejemplo ilustrativo es una línea de ensamblaje de una planta de vehículo, en donde existen varias estaciones secuenciales que en se encargan de obtener un resultado final: El Vehículo.

Por lo tanto podemos decir que es una técnica en la cual múltiples instrucciones se realicen simultáneamente durante todo el proceso de ejecución de un programa. El trabajo que implica realizar una instrucción se descompone en partes pequeñas, cada una de las cuales necesita una fracción del tiempo total para completar la instrucción. Esta serie de pasos se denomina "etapa de la segmentación" o "segmento". Es de hacer notar que el tiempo total de ejecución de todas las instrucciones disminuye pero no se reduce el tiempo que se necesita para completar una instrucción individual.



Una de las cosas que se debe tener en cuenta es que para mantener el flujo de instrucciones a la salida del pipeline (segmentación), es que debe mantenerse también el flujo de instrucciones a la entrada, así se puede evitar que haya vacíos trayendo como consecuencia que se tenga que esperar a que se llene de nuevo el pipeline. Además la segmentación tarda cierto tiempo en producir resultados, el mismo que tarda en llenarse dependiendo del número de etapas, una vez estando lleno por cada pulso de reloj se obtendrá un resultado. En cierto tiempo habrá tantas instrucciones en paralelo como etapas tenga la segmentación, lo que hace que todas las unidades funcionales estén trabajando.

Estaciones básicas de un pipeline:

- Fletch de Instrucción
- Decodificación de la Instrucción
- Fletch de Datos

- Ejecución de la Instrucción
- Escribir Resultados.

La descripción del pipeline que se ha hecho y el aumento y el aumento de la eficiencia del procesador que se ha mostrado realmente están basados en condiciones ideales de operación. Hay varias circunstancias en las cuales no es posible mantener el flujo de entrada al pipeline. Hay 3 categorías de los pipeline hazards (peligros del pipeline).

- Peligros Estructurales.

Significa que el hardware no puede soportar la combinación de instrucciones que se desean ejecutar en el mismo ciclo de reloj. El problema estructural estriba en el hecho de que un componente tiene que ser usado por dos o mas segmentos distintos simultáneamente.

- Peligros de Control

Uno de los aspectos fundamentales detrás de la eficiencia lograda con el pipeline es la ejecución secuencial de las instrucciones. Se busca una instrucción y se procede a buscar la instrucción que sigue a la buscada anteriormente. Pero ¿pero que tal si la instrucción buscada anteriormente es una instrucción de esas que rompen la ejecución secuencial de instrucciones? (saltos condicionales, llamadas a funciones o procedimientos, etc...). En este caso que sigue, que debiera ser la próxima ser introducida en el pipeline no necesariamente es la que sigue en secuencia a la instrucción buscada.

- Peligros de Data

Se intenta usar un dato antes de que sea valido. Ocurre cuando una instrucción depende de un dato que escribe una instrucción previa que también está en el pipeline, es decir, los resultados de una instrucción depende de los resultados de otra que se encuentra en el pipeline.

Algunas soluciones viables a los problemas del pipeline son: (investigar otros)

- Tener mas de una unidad funcional para evitar el problema estructural, aun cuando es muy complejo y costoso y aun así hay que prever que haya mas pedido de esa unidad funcional que unidades disponibles por lo que habrá que estar preparado por software cuando esta solución por hardware sea insuficiente.
- Para el pipeline cuando se detecte un problema de Control o de Data, indudablemente esta solución es la menos deseada y la menos implementada, ya que se requiere para un manejo eficiente del pipeline es que haya un flujo constante de entrada y salida de instrucciones.