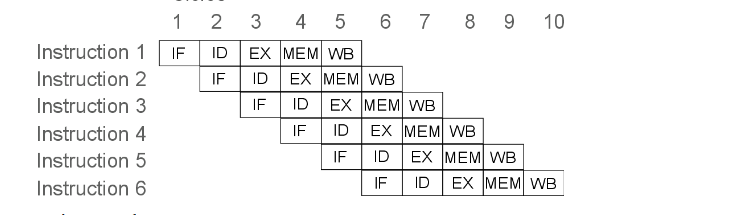
**TRABAJO PRÁCTICO FINAL ARQUITECTURA DE COMPUTADORAS**

**Objetivo**

Implementar el pipeline del proceso MIPS en nuestra placa adquirida Basys 3

**PIPELINE MIPS**

Implementación de etapas para ejecutar instrucciones, donde una instrucción empieza a ejecutarse antes de que hayan terminado las anteriores y, por lo tanto, que haya varias instrucciones procesando simultáneamente.

****

**Etapas**:

* **IF (Instruction Fetch):** Búsqueda de la instrucción en la memoria de programa.
* **ID (Instruction Decode):** Decodificación de la instrucción y lectura de registros.
* **EX (Execute):** Ejecución de la instrucción propiamente dicha.
* **MEM (Memory Access):** Lectura o escritura desde/hacia la memoria de datos.
* **WB (Write back):** Escritura de resultados en los registros.

**Instrucciones permitidas:**

* Instrucciones **tipo R:**

**SLL, SRL, SRA, SLLV, SRLV, SRAV, ADDU, SUBU, AND, OR, XOR, NOR, SLT**

* Instrucciones **tipo I:**

**LB, LH, LW, LWU, LBU, LHU, SB, SH, SW, ADDI, ANDI, ORI, XORI, LUI, SLTI, BEQ, BNE, J, JAL**

* Instrucciones **tipo J:**

**JR, JALR**

**Riesgos:**

Nuestro procesador cuenta con la implementación para atender los siguientes tipos de riesgo:

* **Estructurales.** Se producen cuando dos instrucciones tratan de utilizar el mismo recurso en el mismo ciclo.
* **De datos.** Se intenta utilizar un dato antes de que esté disponible. Mantenimiento del orden estricto de lecturas y escrituras.
* **De control.** Se intenta tomar una decisión sobre una condición todavía no evaluada.

**Requerimientos de operación:**

El programa a ejecutar debe ser cargado en la memoria de programa mediante un archivo ensamblado. El procesador estará a la espera para recibir un programa mediante la **Debug Unit** (es decir mediante la interfaz de UART). Una vez cargado el programa, debe permitir dos **modos de operación:**

* **Continuo**, se envía un comando a la FPGA por la UART y esta inicia la ejecución del programa hasta llegar al final del mismo (Instrucción HALT). Llegado ese punto se muestran todos los valores indicados en pantalla.
* **Paso a paso**: Enviando un comando por la UART se ejecuta un ciclo de Clock. Se debe mostrar a cada paso los valores indicados.

**ETAPA IF**

La etapa IF contiene una instancia del módulo PC para manejar el contador de programa y calcular la siguiente address de instrucción. También permite almacenar instrucciones dentro de la memoria. Ya que las instrucciones son cargadas por la Unidad de de Debug.

Mediante un Multiplexor se obtiene el PC correspondiente y en el módulo de PC se incrementa el PC para la siguiente instrucción

**Módulos de etapa IF:**

* **IF.v:** Inicializa la etapa.
* **pc.v:** Se obtiene el PC, PC+4 y PC+8 en caso de JAL
* **mux\_pc.v:** Se setea un incremento en el PC si hubo un branch o un jump a partir de la instrucción que tiene.
* **memory\_instruc.v:** Recibe una address de la instrucción de que debe recuperar de los registros de la memoria de instrucciones. También escribe las instrucciones que son enviadas desde la pc por la **Debug Unit.**