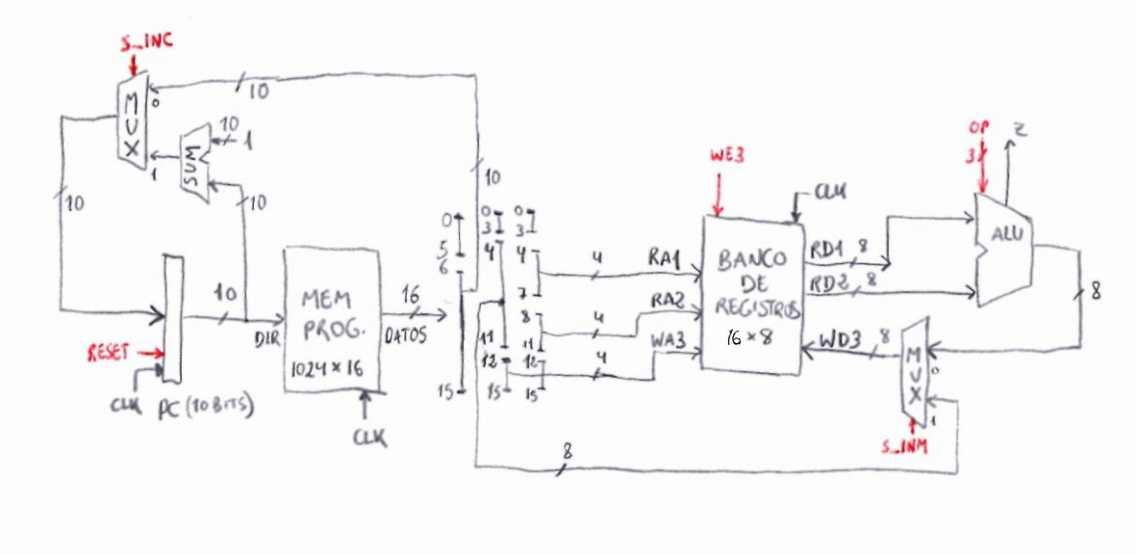
DISEÑO DE LA UNIDAD DE CONTROL DE UNA CPU SIMPLE DE UN SOLO CICLO

El objetivo de la práctica es diseñar la unidad de control de un procesador lo más sencillo posible. Para ello, nos centraremos en un procesador de un sólo ciclo. Para que un procesador pueda ejecutar instrucciones en un solo ciclo sin recurrir al paralelismo en su implementación debemos separar las memorias de instrucciones y de datos de forma que se pueda realizar el acceso a ambas dentro del mismo ciclo (al estilo de la arquitectura Harvard). En este ejemplo el procesador no va a tener una memoria de datos propiamente dicha, sino que operará con su banco de registros como memoria de datos. Esta estructura es típica de algunos microcontroladores, procesadores muy sencillos con una memoria de programa no volátil, diseñados para funcionar integrados en otro artefacto como una lavadora o un coche, realizando el control del mismo.

Para analizar el funcionamiento del procesador, estudiaremos separadamente el camino de datos de la unidad de control que lo gobierna y los modelaremos por separado también.



La figura representa el camino de datos del procesador. Se aprecia el registro PC de 10 bits que sirve de dirección a la memoria de programa. El dato obtenido de esta es la instrucción, de 16 bits. Esos 16 bits se toman de tres maneras diferentes en función de la instrucción de que se trate:

Instrucción de salto: Opcode de 6 bits (5-0) y los 10 bits restantes (15-6) serán el nuevo PC si el multiplexor que controla la entrada al PC tiene su entrada de selección s_inc a cero. En las demás instrucciones, s_inc se pondrá a 1 provocando que el nuevo PC sea el PC previo incrementado en 1.

Instrucción de carga de una constante inmediata: Opcode de 4 bits (3-0), constante inmediata de 8 bits (11-4) y campo de registro de destino de 4 bits (15-12) indicando el registro donde se escribirá la constante (wa3), siempre que el multiplexor que provee el dato a escribir tenga la entrada s_inm a 1.

Instrucción de operación aritmética o lógica: Opcode de 4 bits (3-0), campo de primer registro operando de 4 bits (7-4, ra1), campo de segundo registro operando de 4 bits (11-8, ra2) y campo de registro de destino de 4 bits (15-12) donde se almacenará el resultado (siempre que el multiplexor tenga s_inm a cero).

En la figura se han señalado en rojo las señales que provienen de la unidad de control, que debe tener la siguiente definición de módulo

module uc(**input wire** clk, reset, z, **input wire** [5:0] opcode, **output wire** s_inc, s_inm, we3, **output wire** [2:0] op);

Es decir, serán entradas las señales comunes de reloj y reset, los 6 bits menos significativos de la instrucción (opcode mayor posible), el flag de cero para posibles saltos condicionales y generará las señales de control de ambos multiplexores, la habilitación de escritura del banco de registros y las señales de selección de operación de la ALU. La misión de la unidad de control será activar correctamente las señales de salida a lo largo del ciclo que dura la instrucción de forma que se ejecute correctamente la instrucción determinada por los 6 (o menos) bits de opcode.

OBJETIVO: DISEÑO DE LA UNIDAD DE CONTROL

a) Estudiar y familiarizarse con el funcionamiento de los módulos de partida: ALU, Banco de registros, multiplexores, registro PC

b) Realizar un módulo que represente el camino de datos, con la siguiente definición

```
module microc(input wire clk, reset, s_inc, s_inm, we3, input wire [2:0] op, output wire z,
output wire [5:0] opcode);
```

c) Implementar la unidad de control que permita ejecutar las siguientes instrucciones

- Salto absoluto incondicional
- Salto condicional (a las condiciones cero y no cero)
- Carga de un valor inmediato en un registro
- Instrucción(es) Aritmética(s)/Lógica(s)

Visualizar su correcto funcionamiento con el gtkwave. Realizar un programa simple demostrando el uso de las instrucciones implementadas (ej: con un bucle similar al bucle for)