Ana Karla Caballero González C-213

Para el funcionamiento de este microprocesador se ha utilizado diferentes circuitos. En este documento se nombra y describe cada uno de los circuitos implementados para un mejor entendimiento de lo que ocurre dentro de él.

**Register File**

Este circuito fue creado con el propósito de almacenar datos que se necesitan para realizar las instrucciones que se realizaran. Este contiene 32 registros de 32 bits (R0,R1…R31) de propósito general , donde R0 siempre tiene el valor 0 ya que a este su enable estará siempre en y el R31 actuará como puntero de pila (SP) . Estos registros se les conecta a todos por igual un reloj, un reset y una entrada de dato, y cada registro tiene un enable independiente de dirección que se activa cuando se desea escribir en dicho registro por medio de un demultiplexor, se tiene dos salidas (O1 y O2) cuya dirección entran como S1 y S2 respectivamente. Se tiene además dos registros adicionales Hi y Lo donde se almacena el resultado de la división y la multiplicación que almacena el resultado y resto de la división o la parte alta y baja del resultado de la multiplicación, estos cuentan con entradas independientes y un mismo RW dado que estos se escriben de forma simultánea (como estos registros solo se necesitan en las operaciones “move from hi/Lo” se tiene entonces su salida conectada a un multiplexor que en dependencia de lo que se pida se dará la salida del registro correspondiente).

**ALU**

Esta es una unidad lógica aritmética (ALU) donde se realiza la operación seleccionada. Esta contiene un sumador, un restador, dos multiplicadores, dos divisores (uno de los multiplicadores y uno de los divisores son circuitos que realizan la operación y división respectivamente con la diferencia que estos dividen números con signos) y dos comparadores (Operaciones aritméticas) y además tiene un and, un or, un xor y un nor (Operaciones lógicas).

Claramente esta unidad recibe dos entradas (A y B) y un código de operación a ejecutar(op) que este es el que funciona como selector en el multipexor que une a todas mis operaciones y además al valor de A dado que peste falicita el trabajo de ciertos moves y branches. Este tiene dos salidas, la salida del multiplexor (Out) y otra que solo se activa si ocurre una división o multiplicación (MulorDiv), también se setean los flags de zero y negative.

**Instruction Decoder**

Este recibe la instrucción a ejecutar y separa el Op-Code, Func-Code, las direcciones de los registros de lectura y de escritura, la constante y el destino de salto. En dependencia del Op-Code y/o del Func-Code mediante operaciones lógicas se realiza una clasificación de las diferentes instrucciones para liberar señales que permitan determinar el selector de operaciones de la ALU, clasificar las operaciones en una forma más general para pasar esta información al Control Unit y también determinar el valor de los selectores de inputs para el Register File, el Branch Control y el ALU.

**Branch Control**

Este determina en dependencia de si la operación es un salto, y si se cumple la condición de salto entonces determina el nuevo valor del Program Counter, como los posibles cambios son hacer la próxima operación, hacer un Branch de salto K, un Jump de destino D o un Jump R, entonces recibe como entrada si es un salto, el tipo de salto y los flags que se usan en caso de que sea un salto con condición (Branch) para determinar si se cumple la condición de salto.

**Program Counter**

Este es un registro de 32 bits que almacena el valor dela dirección en la que se encuentra la próxima instrucción en memoria. Su enable se activa por el Control Unit para que se mantenga en la instrucción actual hasta que se haya terminado de ejecutar todo lo que esta debe hacer.

**Instruction Memory**

Está es una memoria Caché Direct Mapped, que contiene 8 líneas de 16 bytes, y de tamaño de palabra 4 bytes y direcciones de 18 bits por lo que destino 2 bits a offset, 3 bits a index y 13 bits a tag, como por línea de la caché caben 16 bytes se puede colocar un bloque entero de la RAM cada vez que se vaya a ejecutar una instrucción se pide esta a caché, si la dirección está en caché y es válido entonces se le da automáticamente la instrucción al Instruction Decoder y se inicia el ciclo actual en la Unidad de Control, en cambio si la dirección que se pide no está en caché se pasa la dirección de memoria a la RAM y se espera a que la RAM lea los datos de esa dirección, copia este bloque a la caché y da la instrucción actual, por tanto como la caché hace una búsqueda en la RAM tiene como salidas una dirección para la RAM, el chip select, el RW y el mask, así como un END que indica si se tiene un ya un dato válido disponible.

Como las operaciones push y store modifican los datos de la RAM es posible que el dato que se cambie en RAM esté también en caché, por tanto, la caché recibe un bit que indica si se está escribiendo en memoria y en que dirección, de modo que si esa dirección está en caché se desactiva el metadato valid forzando a que se vuelva a leer de la RAM en caso de necesitar luego el valor de ese bloque de la RAM.

**Data Path**

Aquí es donde se interrelacionan el ALU, BC, ID, RF y PC, conecta las salidas y entradas de estos componentes y del I-Memory, tiene como entradas los Outputs dela RAM y el Read Time de la misma, las señales de control del Control Unit que controlan los “caminos” por los que circula la instrucción, recibe también la entrada del Kbd para poder guardar los caracteres. Como salidas devuelve la dirección a escribir o leer en RAM, el dato a escribir en RAM, el código ASCII de los caracteres a escribir en el TTY, el R/W, el CS y el Mask de la RAM y las señales que necesita el Control Unit para direccionar las operaciones.

**Control Unit**

Esto es un circuito que dirige la ejecución de la instrucción, recibe el tipo de operación que se va a realizar ([Jump,Jump R,Nop],[ALU],[Load],[Store], [Push],[Pop],[Especiales],[Branches]) así como el RT y WT para controlar en las operaciones de escritura y lectura de datos de la RAM el tiempo en que se debe permanecer en el estado lectura-escritura, el Start de la unidad e control es el end de la caché de instrucciones lo que indica que como ya se tiene la instrucción a realizar los datos que recibe el CU son válidos. Como salidas devuelve señales de control para el Data Path así como CH, R/W para trabajocon la RAM.