Documentació de l'actual estat del processador del dia 27/03/2023:

1. Instruccions Implementades:

I. Instruccions lògiques i aritmètiques:

Les operacions lògiques conformades per AND, OR, XOR, NOT ADD, SUB, SHA i SHL han passat els jocs de prova bàsics satisfactòriament a més dels nostres.

II. Comparacions:

Les operacions de comparació conformades per CMPLT, CMPLE, CMPEQ, CMPLTU, CMPLEU han passat els jocs de prova bàsics satisfactòriament a més dels nostres.

III. Moves:

Les operacions de moviment de dades a nivell de registre conformades per MOVI i MOVHI han passat els jocs de prova bàsics satisfactòriament a més dels nostres.

IV. Operació suma/resta immediats:

La operació ADDI ha superat els jocs de prova satisfactòriament.

V. Loads:

Les operacions d'accés de memòria per carregar dades a registres, ja sigui a nivell de *byte* com a *word* has satisfet els jocs de prova.

VI. Stores:

Les operacions de càrrega a memòria amb dades provinents de registres, ja sigui a nivell de *byte* com a *word* han satisfet els jocs de prova.

VII. Extensió aritmética:

Les operacions d'extensió aritmética conegudes com MUL, MULH, MULHU, DIV i DIVU han superat els jocs de prova.

VIII. Instrucció especial:

La instrucció HALT funciona satisfactòriament.

IX. Salts condicionals i incondicionals:

Les instruccions de salts condicionals BZ i BNZ estan implementades pero no les hem comprovat.

Les instruccions de salts incondicionals a saber: JMP, JAL, JZ i JNZ no estan comprovades.

Joan Teruel; Albert Azcárate

2. Controlador de Memoria:

Varem decidir implementar el controlador de memoria sense estats, nomes basant-nos en els flancs de relltoge, ja que, siguent tant rapida la memoria comparada amb el nostre processasdor, no ens cal anar amb cura sobre els timings ni tindre multiples estats per fer les lectures o escriptures.

3. Decisions de Diseny:

I. Unitat Aritmetico-Logica:

Hem decidit implementar la ALU de manera unitaria. No te móduls interns, sino que esta implementat tot directament, multiplexant les sortides segons la Operacio i la Funcio de Operacio. Aquesta decisió no es massa critica i facilment es podrien fer modules per fer certes operacions com els SHL o SHA sense generar tant creuament de llibreries.

II. Unitat de Multicicle:

La necesitat d'un estat Fetch i un Decode ens ha portat a una maquina d'estats binaria i pocs vertexs. Sempre passem de Fetch a Decode i viceversa. En general sempre passem al data path les senyals de control en estat Decode i les bloquegem en estat Fetch.

III. Controladors de memoria i SRAM:

Obviament hem escollit el esquema de SRAM de words de 16bits i 32k files, ja que, a part de que es el que se'ns demana, es la més optima segons l'esquema Fecth Decode. Com ja s'ha especificat abans, no tenim dificultats en complir els cronogrames de la SRAM, aixi que la implementacio es bastant directe, actualitzant entrades i sortides segons flancs ascendents de rellotge.

IV. Instruccions de Salts:

Per instruccions de control de flux hem decidit ampliar el ldpc a un std_logic_vector de tamamy 2, per distingir quan hem de fer Pc = Pc + 2, Pc = RegN, Pc = Pc + 2 + Imm8, Pc = Pc. Hem augmentat el multiplexor d'entrada del Banc de Registres per que tingui entrades desde Memoria, ALU i el Pc