UNIVERSITAT DE BARCELONA

Examen final d'Estructura de Computadors (EI)
Gener de 2021

Nom:
Cognoms :

RISC-V (3 punts)

1. A diferència del RISC-V tradicional de 5 etapes, aquesta CPU alterada ha dividit la segona etapa (decode) en dues etapes diferenciades: decode i registre de lectura. A més, aquesta CPU té una ALU pels càlculs de coma flotant (FP ALU) junt amb una ALU tradicional (no s'afegeixen registres al registre file per guardar valors per a la FP ALU). El senyal de control afegit, FPOp, determina quina ALU s'ha d'utilitzar per a una instrucció determinada.
   Aquesta ALU de coma flotant funciona interpretant els seus operands de 32 bits en format de coma flotant IEEE 754. A diferència del RISC-V tradicional, suposem que les operacions de coma flotant utilitzen els mateixos registres que les operacions normals de coma no flotant.
   En la figura 1 mostrem un diagrama de la CPU modificada i les seves etapes corresponents. En la taula 1, proporcionem els retards de cada etapa:

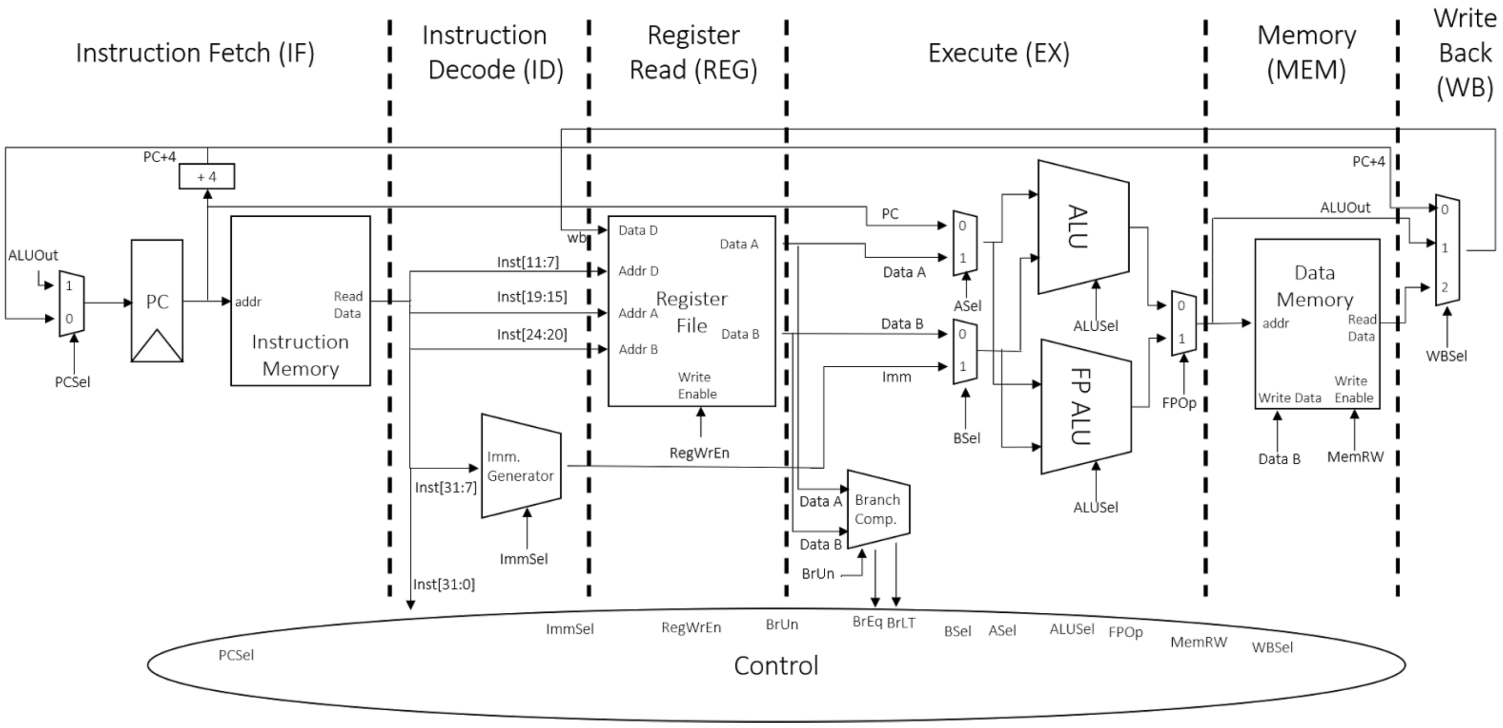

Fig.1: Esquema simplificat del nostre processador

Element	Reg CLK-to-Q*	Register Setup	MUX	ALU	FP ALU	Mem Read	Mem Write	RegFile Read	RegFile Setup	Imm. Gen
Delay	20	25	10	150	215	225	230	100	35	75

*Retard, temps que triga el registre en donar-nos la dada un cop arriba CLK

.

TIP: El temps de setup s'introdueix a L03_Optimitzacio_processadors_5stages_PIPELINED.pdf, tot i que la definició exacta apareix a L08_Memoria_Principal_DRAM.pdf. El temps de setup és el temps en què una dada ha d'estar estable abans de l'arribada del rellotge. En aquest problema en particular, el temps de setup afecta als registres: la dada ha d'estar estable N ps abans de l'arribada del rellotge; teniu un exemple a les pàgines 30 i 31 de la lectura L03. Penseu quins registres teniu en els processadors de l'apartat a i b, en el cas a, la lectura de RegFile és asíncrona.

Nom:
Cognoms :

a) Penseu en la instrucció de coma flotant, faddi, que és similar a l'addi, tret que considera operands en format de coma flotant i executa una operació d'adició de coma flotant. Suposant que aquesta CPU NO té pipeline (és a dir, és una CPU single cycle), quin és el període de rellotge més curt possible per executar la instrucció *faddi t0, s0, 2.71* correctament? (0.85 punts)

b) Quin és el període de rellotge més curt possible per executar instruccions en aquesta CPU si **SÍ** tenim pipeline? (1.25 punts)

c) Seguint amb la CPU amb pipeline modificat, executeu CORRECTAMENT el programa següent seguint les següent suposicions:

1. No hi ha optimitzacions de pipeline (no forwarding, ni load delay slot, ni branch prediction, ni buidat de pipeline, etc)
2. No podem llegir i escriure els registres en un mateix cicle de rellotge.
3. Un integer (100) es guarda a la direcció de memòria 0x61C61C61, i que R[a0] = 0x61C61C61
4. Un hazard entre dues instruccions s'ha de comptar com 1 sol hazard

Codi: Registres amb nomenclatura RIPES (o MIPS)

```
lw t0, 0(a0) # R[a0] = 0x61C61C61

srli s0, t0, 4

faddi s1, t0, 1.7

beq a0, s1, Label

add a1, t2, t3

Label ...
```

No importa el tipus de diagrama que feu servir per a completar l'execució, o si no en necessiteu cap. El resultat us ha de servir per contestar les següents preguntes:

i. Quants cicles de rellotge triga en executar-se el programa correctament?
ii. Quants stalls/noop heu hagut d'afegir per executar correctament el programa?
iii. Quants data hazards hi ha en el programa?
iv. Quants control hazard hi ha en el programa?

Nom:
Cognoms :

d) Reordeneu el codi per aconseguir el mateix resultat minimitzant el nombre total de cicles d'execució:

i. Quants cicles de rellotge triga en executar-se el programa correctament?
ii. Quants stalls/noop heu hagut d'afegir per executar correctament el programa?