Informe de la pràctica #4

Introducció

Aquesta cuarta practica está enfocada en les microinstruccions, és a dir, en cada part de la execució de una instrucció en els diferents cicles de rellotge.

- Visualitzar els diferents passos que ha de realitzar un microprocessador per tal de executar una instrucció.
- Aprendre a comptar el numero de cicles que tarden en executar-se diferents tipus de instruccions al microprocessador.

Exercicis

Primer de tot, i tal com està al guió de pràctiques, copiem el primer codi que apareix per tal de observar el seu funcionament.

```
.begin start
start:
      ADD R0,R0,R3
      ADD R0,R0,R7
      ADDI R0, #4, R2
      ADD R2, R3, R3
      SUBI R7, #1,R7
      BG salto
salto:
      STORE R3, guardaResultat(R0)
.end
```

Aquest codi, tot i no tenir una funció real, ens funciona perfectament per a estudiar els temps de execució de les microinstruccions de cada una de les instruccions. Com que tenim instruccions de tots els tipus, podrem observar els diferents temps i cicles que tarda en cada cas.

Més endevant, en el guió de pràctiques, trobem un segón codi per a executar, aquest més detallat i ha de ser executat microinstrucció a microinstrucció.

```
valorDada: .dw 7
guardaResultat: .rw 1
.begin start
start:
      LOAD valorDada(R0), R1
      ADDI R0, #9, R2
      ADD R0, R0, R3
loop:
      ADD R2, R3, R3
      SUBI R7, #1,R7
      BG loop
      STORE R3, guardaResultat(R0)
.end
```

Aquest codi, que conté tot tipus de instruccions (aritmetico-lógiques, de salt i de accés a memòria), ens ajudará a comprendre els cicles de execució de les instruccions al processador.

El simulador SiMR, amb el que nosaltres treballem, consta de uns cicles ja definits, el FETCH, on es carrega la instrucció de memória al processador, el DECO, on es decodifica i comença la execució de la instrucció, els LOAD i STORE, on es fan les instruccions de carrega i guardat a memória respectivament, el ARIT, on es realitzen les operacions aritmetico-lógiques i el BR, on es fan les operacions de salt.

Nosaltres contarem com que el cicle FETCH de cada instrucció no entra dins del temps de execució, aixi que sense comptar-lo, tenim que les instruccions aritmetico-lógiques tarden 2 cicles de rellotge a executar-se. Per altra banda, les instruccions de salt, tenen diferents temps de execució: en cas de que la condició no es compleixi (com p.e. Les BG o BL) la instrucció únicament tarda 1 cicle de rellotge en executar-se, ja que només la ha de descodificar, en canvi si la instrucció compleix la condició, tarda 2 cicles de rellotge com les anterior dites. Finalment, les instruccions de accés a memória tarden també 2 cicles, ja que després de esser descodificades, s'han de carregar o guardar dades únicament.

Si agafem la instrucció ADD R2, R3, R3, podem veure que al primer cicle s'activen els bits “Ld_RA” i “Ld_R@”, que porten la instrucció a la ALU per tal de operar amb aquesta instrucció ARIT. Tot i ser una instrucció de tipus diferent, la instrucció LOAD valorDada(R0), R1, activa exactament els mateixos bits de control en el seu primer cicle de execució.

A la instrucció BG loop, en cas de que el salt no es prodüeixi, tenim que la entrada del MUX que s'activa per a apuntar a la memória és la de “-”, la provinent del PC. Aquesta entrada apunta a la direcció de memória “0008”, és a dir, la següent instrucció. Al contrari, si el salt si s'ha produït, la entrada del MUX és la de “+”, que provenint del R@ apunta a la direcció de memória “0005”, és a dir, la posició del *loop*.

Conclusions

Per finalitzar, en aquesta practica s'han aprés i arribat a les següents conclusions:

- S'ha aprés a observar els bits de la UC i interpretar-los.
- S'ha aprés a comptar els cicles de rellotge en temps de execució de instruccions.
- S'ha aprés el diagrama de estats del simulador SiMR aixi com el funcionament dels seus estats.