ARCHI - TP4 : Processeur

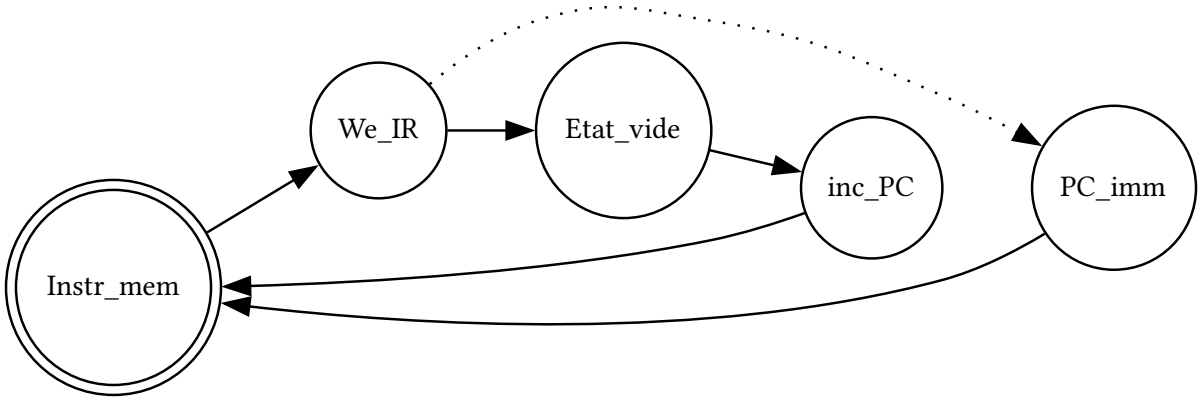
Réponses aux question 4 et 5 du sujet

4. Toutes les instructions s'exécutent en 4 cycles (4 états de l'automate), sauf l'instruction d'accès mémoire qui prend un cycle de plus, *ie* 5 cycles.
   On ne compte pas les instructions F comme de vrais instructions, elles sont plus des marqueurs de fin de programme.
5. En considérant ce que l'on a dit à la question précédente, on pourrait considérer que notre processeur a un CPI moyen qui vaut la moyenne des CPI, *ie* un CPI moyen de $\frac{4\times13+5}{14} = 4,07$.

Les états de base de la FSM

Voici la table des valeurs modifiée dans chaque état. L'automate représenté est le squelette de la FSM et toutes les transisions sont celles par défaut sauf celle en pointillé, ce qui signifie que l'état est accessible sous certaines condiditions (*cf* Les *opcod* de saut conditionnel). En page 3, on peut retrouver la FSM en entier.
Ainsi, l'état de départ est *Instr_mem* où la prochaine opération est écrite dans *IR* grace à **WeIR**. À *We_IR*, on n'autorise plus l'écriture dans *IR* puis on fait le switch case selon l'*opcod* pour choisir vers quel état transitionner.
Par défaut, on transitionne vers l'état vide qui ne fait rien, utile pour l'*opcod* E qui n'est pas assigné, puis il transitionne vers *inc_PC*.
Les états *inc_PC* et *PC_imm* servent à incrémenter le compteur *PC*, dans le cas il est incrémenté de 1 et dans l'autre de la valeur de $\text{sext}_{16}(\text{IMM8})$, et ce grâce à **WeReg**, **SelA**, **SelImm**, **cmd** et **selSext**.

États	SelM	SelD	WeReg	SelB	SelA	SelImm	cmd	WeMem	WeIR	selSext	selImmByte
Instr_mem	0	11	0	11	11	00	111	0	1	0	0
We_IR			0					0	0		
Etat_vide											
inc_PC	0	11	1	00	11	10	000	0	0	0	0
PC_imm	0	11	1	00	11	11	000	0	0	1	0



Les *opcod* qui n'utilisent que l'ALU (0 à 9)

Chaque état représente un *opcod*, l'état *Si* pour l'*opcod i*. Tous ces états transitionnent vers *inc_PC*.

Attention : on a remarqué que la commande 110 de l'ALU ne marchait pas.

Par exemple, pour *S0* l'opération correspondante est $\mathrm{Rd} = \mathrm{sext}_{16}(\mathrm{IMM8})$. Avec **SelM**, **SelD** et **WeReg** on permet l'écriture du résultat de l'ALU dans le registre *rd*. **SelImm** et **selSext** permettent de récupérer l'extension de signe de *imm*, qui est envoyé à l'ALU, et qui ressort immédiatement grâce à **cmd**. Pour les autres *opcod* c'est similaire.

État	SelM	SelD	WeReg	SelB	SelA	SelImm	cmd	WeMem	WeIR	selSext	selImmByte
S0	0	rd	1	00		11	111	0	0	1	0
S1	0	rd	1	rs	rd	00	000	0	0	0	0
S2	0	rd	1		rs	11	000	0	0	1	0
S3	0	rd	1	rs	rd	00	001	0	0	0	0
S4	0	rd	1	rs	rd	00	100	0	0	0	0
S5	0	rd	1	rs	rd	00	011	0	0	0	0
S6	0	rd	1	rs	rd	00	010	0	0	0	0
S7	0	rd	1	rs	rd	00	110	0	0	0	0
S8	0	rd	1	rs	rd	11	011	0	0	0	1
S9	0	rd	1	rs	rd	11	011	0	0	0	0

Les *opcod* qui intéragissent avec la mémoire (A et B)

Pour récupérer un valeur en mémoire, on a besoin de deux cycles (donc deux états) :

- un premier (*SA_mem*) qui récupère *rs* et qui le laisse passer au travers de l'ALU grâce à **SelB**, **SelImm** et **cmd** pour être envoyé comme adresse à la mémoire, qui renverra au cycle suivant la valeur de la case *mem[rs]*,
- un second (*SA_WeReg*) qui récupère *mem[rs]* et qui l'écrit dans le registre *rd*.

Pour écrire en mémoire (*SB*), on récupère *rs* et on le laisse passer au travers de l'ALU grâce à **SelB**, **SelImm** et **cmd** pour être envoyé comme adresse à la mémoire, où l'on y écrit *rd* grâce à **SelA** et **WeMem**. Dans les deux cas, on transitionne ensuite vers *inc_PC*.

États	SelM	SelD	WeReg	SelB	SelA	SelImm	cmd	WeMem	WeIR	selSext	selImmByte
SA_mem			0	rs		00	111	0	0		
SA_WeReg	1	rd	1					0	0		
SB	0	11	0	rs	rd	00	111	1	0	0	0

Les *opcod* de saut conditionnel (C et D)

Dans ces deux états, on fait la différence entre *rd* et *rs* grâce à **selB**, **selA**, **SelImm** et **cmd**. Ensuite, selon le résultat, soit on transitionne vers :

- *PC_imm* si le résultat est négatif (*resp.* zéro) pour l'*opcod* C (*resp.* D),
- *inc_PC* sinon, *ie* le résultat est n'est pas négatif (*resp.* zéro) pour l'*opcod* C (*resp.* D).

État	SelM	SelD	WeReg	SelB	SelA	SelImm	cmd	WeMem	WeIR	selSext	selImmByte
SC	0	rd	0	rs	rd	00	001	0	0	0	0
SD	0	rd	0	rs	rd	00	001	0	0	0	0

FIN

Si l'*opcod* vaut *F* (*ie* 1111), alors l'on va dans un état tel qu'on oscille entre lui et un autre, où plus rien ne se passe. C'est la fin du programme, il faut *reset* la machine et potentiellement mettre un nouveau code en mémoire.

États	SelM	SelD	WeReg	SelB	SelA	SelImm	cmd	WeMem	WeIR	selSext	selImmByte
SF_fin			0					0	0		
SFIN			0					0	0