**Explication de pourquoi on choisit T2 au lieu de T1 sur le LDR/STR.**

Parce que en T1, il faut coder un registre et un immédiat.

On prend T1 pour toutes instructions sauf LDR/STR où on choisit T1 et T2.

T2 c’est plus simple car on a directement l’adresse. En T1 faut faire la somme entre le contenu d’un registre et l’immédiat.

Donc, on choisit l’encoding T2 car plus simple.

Sinon, le registre R1 il faut le faire quelque part l’addition entre R1 et l’immédiat. (Dans l’alu ? on ne sait pas…)

**Qu’est-ce que Rt dans LDR/STR ?**

L’emplacement pour coder le registre est le même pour LDR/STR, 3 bits pour coder l’adresse, on n’a ni Rd ni Rn car soit c’est une destination si c’est load, soit c’est la source si c’est un store. Donc pour éviter de mettre Rd et Rn, on met RT pour dire soit l’un ou soit l’autre.

**Y'a plusieurs registres de destination Rdn, Rdm, et Rd :**

**Sur les instructions, on a =/= registres qui vont dans différentes destinations.**

**Faut-il séparer ou les regrouper ?**

Quand on a Rdn, cela fait référence au registre de destination et au deuxième registre entré en paramètre.

(Pas la place pour écrire 3 registres sur 16 bits)

Rdn pour dire Rd et Rn.

On a Rn qui prend deux registres et Rm opérande.

RDN donc RN destination du résultat.

Si on met dans Rdn toutes les destinations Rd.

Dans Rdn on envoie sur rd et rn les 3 bits. O envoie aux deux.

**Program counter ?**

Un registre. La mémoire ROM à gauche. On envoie sur le port d’adresse l’arobase, en même temps ça fait incrémenter de 1 pour le prochain cycle ; ça c’est la boucle standard.

Quand on regarde les instructions en branchement, soit on fait « +1 » soit on fait directement « PC = un immédiat sur 8 bits » car la condition de branchement aura été respectée (pour faire un saut et non la prochaine instruction, aller plus long en avant ou arrière).

On aura un multiplexeur devant qui prend branche haut et bas en fonction de la condition de source soit validés.