Université BADJI Mokhtar Annaba

Faculté des sciences de l’ingénieur

Département d’électronique

Master 1 Système embarqué

Matière : DSP

**TP 4 : Programmation du produit scalaire de 2 vecteurs (.ASM)**

Ce TP reprend le produit scalaire de 2 vecteurs de taille 4 (voir TP3) mais le main et la fonction sont implémentés en assembleur.

Le projet appelé dotp4a inclut:

1. **Le programme dotp4a\_init.asm qui permet d’initialiser les variables et fait appel à la fonction dotp4afunc.asm.**
2. **Le programme dotp4afunc.asm qui calcule le produit scalaire des 2 vecteurs.**
3. **Le programme vectors\_dotp4a.asm**

**N.B:** Aucun fichier library n’est nécessaire pour ce projet.

1- **;dotp4a\_init.asm**

**.def init ;adresse de début**

**.ref dotp4afunc ;la fonction appelée**

**.text ;section du code**

**x\_adr .short 1,2,3,4**

**y\_adr .short 0,2,4,6**

**result\_adr .short 0**

**Init MVK result\_adr, A4**

**MVK 0,A3**

**STH A3,\*A4**

**MVK x\_adr, A4**

**MVK y\_adr, B4**

**MVK 4,A6**

**B dotp4afunc**

**MVK ret\_adr, B3**

**NOP 3**

**ret\_adr MVK result\_adr,A0**

**STW A4,\*A0**

**wait B wait**

**NOP 5**

**2- ;Dotp4afunc.asm Multiplication de 2 vecteurs**

**;A4=adresse x, B4=adresse y , A6=count, B3=adresse de retour**

**.def dotp4afunc ;fonction produit**

**.text ;section text**

**Dotp4afunc MV A6,A1**

**ZERO A7**

**loop LDH \*A4++,A2**

**LDH \*B4++,B2**

**NOP 4**

**MPY A2,B2,A3**

**NOP**

**ADD A3,A7,A7**

**SUB A1,1,A1**

**[A1] B loop**

**NOP 5**

**MV A7,A4**

**B B3**

**NOP 5**

1. **;vectors\_dotp4a.asm**

**.ref init ;adr de début dans le fichier init**

**.sect "vectors" ; section vectors**

**rst: mvkl .s2 init,B0**

**mvkh .s2 init,B0**

**B B0**

**NOP 5**

**Création et compilation du projet dotp4a**

* Créer le projet **dotp4a .**
* **Modifier l’option du Linker (Project**→ Options) en sélectionnant “**No** **Autoinitialization**” dans **automodelinit** et éviter le message (warning): “ entry point symbol *\_c\_int00 undefined ” du au fait que le point d’entrée conventionnel n’est pas utilisé dans ce projet.*

*(pas de fonction main en assembleur).*

Placer un breakpoint à la première instruction branch dans le programme dotp4a\_init.asm :

B dotp4afunc

Exécuter le programme (**Run**)

Sélectionner: **View** → **Memory**,

*Choisir l’adresse:* ***result\_adr*** *, et*

***16 - bit signed* integer.**

L’exécution s’arrête au niveau de l’instruction où se trouve le breakpoint.

Le contenu de l’adresse mémoire ***result\_adr*** *est zéro.* ***Pourquoi ?***

*Sélectionner* Run à nouveau, puis halt, puisque l’exécution est dans la boucle infinie wait :

wait B wait ;attendre ici

Vérifier le résultat du produit des 2 vecteurs.

Vérifier que A0 contient l’adresse du résultat (*result\_addr ). Sélectionner* ***View*** *→* ***Registers*** *→* ***Core Registers*** *et vérifier cette adresse en hexadécimal.*

*Dans quel registre se trouve le résultat ?*

*Vérifier ce résultat.*

