

Q0 :

Z:\AxIDE3\Example\ProjetNumerique\Q0.lst					
0001	100e		TCNT	equ	\$100E
0002					
0003	2000			org	\$2000
0004					
0005	2000 fe 10 0e	[5]	ldx	TCNT	
0006	2003 01	[2]	nop		
0007	2004 18 fe 10 0e	[6]	ldy	TCNT	
0008					
0009	2008 39	[5]	rts		

Fichier.lst

On relève le nombre de cycles entre la fin du chargement de TCNT dans X et la fin du chargement de TCNT dans Y. A la fin de l'exécution de **ldx TCNT**, on a l'exécution de la commande **nop** qui prend 2 cycles et on exécute la commande **ldy TCNT** qui prend 6 cycles. On a donc $Y - X = 8$ cycles entre la fin des 2 chargements.

```
>call 2000  
  
P-2000 Y-0428 X-0420 A-FF B-FF C-90 S-0041  
>
```

$X = \$0420$

$Y = \$0428$

$Y - X = 8|_{10} = \$0008|_{16}$, pour les 8 cycles.

L'horloge du CPU a une fréquence de 2MHz. En effet, PR1 et PR0 sont initialisés à 0 (ils représentent la division de l'horloge du CPU) donc la division de l'horloge est de 1.

Donc le temps d'un cycle machine est de $T = 1/f = 500\text{ns}$. Le temps de entre les 2 chargements $500.10^{-9} * 8 \text{ cycles} = 4000\text{ns}$

L'horloge du CPU et celle du TCNT est donc la même.

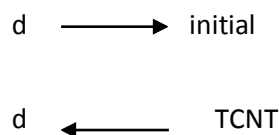
Q1 :

```
Z:\AxIDE3\Example\ProjetNumerique\Q1.lst
0001 100e          TCNT equ    $100E
0002
0003 2000          org      $2000
0004 2000          initial rmb 2
0005 2002          delta rmb  2
0006
0007 2050          org      $2050
0008
0009 2050 fc 10 0e    [ 5 ]    ldd    TCNT
0010 2053 fd 20 00    [ 5 ]    std    initial
0011 2056 01          [ 2 ]    nop
0012 2057 01          [ 2 ]    nop
0013 2058 01          [ 2 ]    nop
0014 2059 01          [ 2 ]    nop
0015 205a 01          [ 2 ]    nop
0016 205b fc 10 0e    [ 5 ]    ldd    TCNT
0017 205e b3 20 00    [ 6 ]    subd   initial
0018 2061 fd 20 02    [ 5 ]    std    delta
0019
```

Fichier .lst

Le résultat de la soustraction est stocké dans delta par la commande **std delta**. La commande **subd initial** effectue la soustraction de la valeur contenu dans le registre D et de la valeur contenue dans initial.

Nous chargeons l'adresse TCNT dans le registre d puis nous envoyons la valeur du registre d dans initial (5 cycles). Ce processus dure 10 cycles. Puis nous rechargeons l'adresse du TCNT dans le registre et nous effectuons la différence entre la valeur chargé (5 cycles) et initial pour la stocké dans delta (16 cycles). Ainsi pour calculer le temps de cycle d'un processus nous devons lui retirer 10 cycles celle du stockage initial du registre d à delta et celui du chargement du TCNT au registre d.



Cycle retardant le dénombrement du nombre de cycle d'un processus

```
>call 2050
P-2050 Y-F1D2 X-F1CA A-00 B-14 C-D0 S-0041
```

Etat des registre après exécution du programme

Lorsqu'on exécute le programme, on remarque que le registre d est égale à 0014_{16} donc 20_{10} c'est-à-dire le nombre de cycle calculé jusqu'à la fin deuxième chargement du registre d.

```

Z:\AxIDE3\Example\ProjetNumerique\Q1.lst
0001 100e          TCNT equ $100E
0002
0003 2000          org $2000
0004 2000          initial rmb 2
0005 2002          delta rmb 2
0006
0007 2050          org $2050
0008
0009 2050 fc 10 0e      [ 5 ] ldd TCNT
0010 2053 fd 20 00      [ 5 ] std initial
0011 2056 01          [ 2 ] nop
0012 2057 01          [ 2 ] nop
0013 2058 01          [ 2 ] nop
0014 2059 01          [ 2 ] nop
0015 205a 01          [ 2 ] nop
0016 205b fc 10 0e      [ 5 ] ldd TCNT
0017 205e b3 20 00      [ 6 ] subd initial
0018 2061 fd 20 02      [ 5 ] std delta
nn19

```

Fichier .lst

Calcul du nombre de cycle entre la fin du premier chargement du registre d et la fin du deuxième chargement d :

$$5+5+5*2 = 20$$

Valeur de delta adressé à
\$2002-\$2003

Valeur d'initial donc de l'état
TCNT adressé à \$2000-\$2001

Adresse du code à exécuté à
l'adresse \$2050

```

>call 2000
P-2000 Y-F1D2 X-F1CA A-50 B-59 C-C8 S-0041
>
BUFFALO 3.4 (ext) - Bit User Fast Friendly Aid to Logical Operation
>l t
done
>call 2050
P-2050 Y-F1D2 X-F1CA A-00 B-14 C-D0 S-0041
>md 2000
2000 39 8B 00 14 18 FE 10 0E 39 FD E7 EE D1 3F FF 7F 9 9 ?
2010 FD FF FF 7A D4 FE F9 FE FF FA C5 76 EF 2F 6E BB z v /n
2020 DE FF BF AF FE DF FA FF EE 7F F7 FB DE BC 3C 5F <
2030 FB BB FF 73 5F 7B B7 79 BD EE DF B6 FF F2 FE 9B s_ y
2040 FB EF DB 78 EF FF DF 69 BD 8F FD FF 7F FF DF FF x i
2050 FC 10 0E FD 20 00 01 01 01 01 FC 10 0E B3 20
2060 00 FD 20 02 39 FB BF FE DF FF F7 B9 6F BF FF FD 9 o
2070 79 FD B9 29 77 AF FB FF BC FD 69 FE 77 E6 6F F5 y )w i w o
2080 7F FD 5F B7 BB D9 F7 CF ED FF AF FB D7 5F F3 7F _ _
>

```

Contenu de la mémoire

On constate que l'adresse de TCNT est la même que le compteur TCNT on a donc affecté à la variable TCNT l'état du compteur.

Q2 :

PA2 utilise le registre TC1, un registre 16 bit commençant à l'adresse \$1010 à \$1011.

PA1 utilise le registre TC2, un registre 16 bit commençant à l'adresse \$1012 à \$1013.

PA0 utilise le registre TC3, un registre 16 bit commençant à l'adresse \$1014 à \$1015.

Q3 :

```
Z:\AxIDE3\Example\ProjetNumerique\q3.lst
0001 1010          TIC1 equ    $1010
0002 1023          TFLG1 equ   $1023
0003 1021          TCTL2 equ   $1021
0004
0005 2050          org      $2050
0006 2050          front rmb    2
0007 2052          periodrmb    2
0008 2054 10       ED1ba fcb    $10
0009
0010 2100          org      $2100
0011
0012 2100 f6 20 54  [ 4 ]      ldab  ED1ba
0013 2103 f7 10 21  [ 4 ]      stab  TCTL2
0014
0015 2106 b6 10 23  [ 4 ]      ldaa  TFLG1
0016 2109 84 04     [ 2 ]      anda  #$04
0017 210b b7 10 23  [ 4 ]      staa  TFLG1
0018
0019 210e b6 10 23  [ 4 ] lire1 ldaa  TFLG1
0020 2111 84 04     [ 2 ]      anda  #$04
0021 2113 27 f9     [ 3 ]      beq   lire1
0022
0023 2115 fc 10 10  [ 5 ] suite1 ldd  TIC1
0024 2118 fd 20 50  [ 5 ]      std   front
0025
0026 211b b6 10 23  [ 4 ]      ldaa  TFLG1
0027 211e 84 04     [ 2 ]      anda  #$04
0028 2120 b7 10 23  [ 4 ]      staa  TFLG1
0029
0030 2123 b6 10 23  [ 4 ] lire2 ldaa  TFLG1
0031 2126 84 04     [ 2 ]      anda  #$04
0032 2128 27 f9     [ 3 ]      beq   lire2
0033
0034 212a fc 10 10  [ 5 ] suite2 ldd  TIC1
0035 212d b3 20 50  [ 6 ]      subd   front
0036
0037 2130 fd 20 52  [ 5 ]      std   period
0038
0039 2133 39        [ 5 ]      rts
```

Fichier .lst

Le TCNT est codé sur 2 octets donc un tour complet est égal à $FFFF|_{16}$.

Description du programme :

- On définit un nom de variable pour 3 adresses correspondant :
 - TIC1 = \$1010
 - TFLG1 = \$1023
 - TCTL2 = \$1021
- On se place à l'adresse \$2050 :
 - org \$2050
- On alloue 2 octets à 2 variables front et period :
 - front rmb 2
 - period rmb 2

- On met la valeur \$10 à l'adresse \$2050
 - ED1ba fcb \$10
- On se place à l'adresse \$2100
 - org \$2100
- On charge la valeur contenu à l'adresse indiquée par ED1ba dans l'accu b :
 - ldab ED1ba
- On stocke la valeur contenue dans l'accu b à l'adresse indiquée par TCTL2 :
 - stab TCTL2
- Ldaa TFLG1 dans l'accu a la valeur contenu à TFLG1 (initialisation)
- anda #\$04 fait l'opération de masquage entre l'accu a et la valeur 4
- staa TFLG1 permet de stocker le résultat de l'opération anda (contenu dans l'accu a) dans l'adresse référencée par TFLG1
- **On initialise le front actif sur PA2 en mémorisant l'état du TCNT sur TIC1 : (on l'appellera cmd β)**

lire1 Ldaa TFLG1 mettre dans l'accu a la valeur contenu à TFLG1 (initialisation) et correspond à la nouvelle commande lire1

Ldaa TFLG1	0	0	1	0	0	1	0	0
Anda #\$04	0	0	0	0	0	1	0	0

Cet opération de masquage donne le résultat 0000 0100₂ et la commande **staa TFLG1** permet de stocker le résultat du masquage dans TFLG1. Cela permet d'initialiser le front actif sur PA2 en mémorisant l'état du TCNT sur TIC1.

- beq lire1 : on boucle sur lire1 si le test entre TFLG1 et #\$04 est égal à 0, sinon on va à suite1

beq fait un test entre le résultat de l'opération précédente et la valeur 0 contenu dans le registre 0.

- Suite1 est la commande ldd TIC1 qui correspond au chargement de l'adresse correspondant à TIC1 dans l'accu d.
- Std front copie la valeur contenue dans l'accu d dans front (front étant l'espace de 2 octets alloué en début de programme) à l'adresse \$2050.
- Ldaa TFLG1 /anda #\$04/ refait l'opération de masquage entre l'accu a et la valeur 4
- staa TFLG1 permet de stocker le résultat de l'opération anda (contenu dans l'accu a) dans l'adresse référencée par TFLG1
- lire2 ldaa TFLG1

anda #\$04

beq lire2

➔ **cmd B** est similaire à l'opération du même type qui débute à lire1

- suite2 ldd TIC1 correspond au chargement de l'adresse correspondant à TIC1 dans l'accumulateur.
- subd front effectue la différence entre la valeur contenue dans d et celle contenue dans front et met le résultat dans l'accumulateur
- std period stocke la valeur contenue dans l'accumulateur à l'adresse \$2052, c'est-à-dire period
- rts marque la fin du programme

On trouve donc un nombre total de cycle de 87 (à vérifier)

Configuration des fronts actifs sur le port A.

- Quel est le registre participant à cette opération ?
TCTL2
- Quelle est la polarité du front actif ?
Front montant
- Quelle est l'entrée sélectionnée ?
Activation PA2

Initialisation de la signalisation

- Quel est le registre participant à cette opération ?
TFLG1
- Comment est-il programmé ?
On effectue une opération de masquage entre la valeur contenue dans TFLG1 (que l'on charge dans l'accumulateur a) et la valeur 4. Le résultat est renvoyé dans TFLG1 et enregistre le front actif sur IC1F (qui signale l'arrivée d'un front actif sur PA2 en mémorisant l'état du TCNT sur TIC1)

Scrutation

Quel est le bit de signalisation qui est testé pour examiner l'arrivée d'un 1er front actif ?
Le 3^e bit en partant du bit de poids faible est testé (car 4 = 0000 0100₂)
Si le test beq renvoie 0 (réponse non, ce qui équivaut à PA2 qui n'est pas sur un front montant), alors on boucle sur lire1 sinon on passe à suite1 ➔ PA2 est sur front montant.

Datation de l'arrivée du premier front du signal

- Quel registre de capture participe à l'opération ?
TIC1
- Quelle est l'adresse de la mémoire qui contiendra la date d'arrivée du premier front ?
\$2050 car on a alloué 2 octets d'espace à front juste après le début du programme

Ré-initialisation de la signalisation

- Quel est le registre participant à cette opération ?
TFLG1
- Comment est-il programmé ?
De la même manière que précédemment

Scrutation

Quel est le bit de signalisation qui est testé pour examiner l'arrivée d'un 2ème front actif ?

Le 3^e bit en partant du bit de poids faible est testé (car 4 = 0000 0100|₂)

Si le test beq renvoie 0 (ce qui équivaut à PA2 qui n'est pas sur un front montant), alors on boucle sur lire1 sinon on passe à suite1 → PA2 est sur front montant.

Datation de l'arrivée du second front du signal

- Quelle est l'adresse de la mémoire qui contiendra la valeur de la période calculée ?
\$2052 car on a alloué 2octets d'espace sous l'étiquette period

Q4 :

Q5 :

Q6 & 7:

```
Z:\AXIDE3\Example\ProjetNumerique\q6.lst
0001 1010          TIC1 equ    $1010
0002 1023          TFLG1 equ    $1023
0003 1021          TCTL2 equ    $1021
0004 1022          TMSK1 equ    $1022 ;
0005
0006 2050          org    $2050
0007 2050          front rmb    2
0008 2052          periodrmb    2
0009 2054 10       ED1ba fcb    $10
0010
0011 00e8          org    $00E8 ;
0012 00e8 7e 21 20 [ 3 ]      jmp    issprog    ;
0013
0014 2100          org    $2100
0015
0016 2100 0f       [ 2 ]      sei    ;
0017
0018 2101 8e 02 00 [ 3 ]      lds    #$0200
0019
0020 2104 f6 20 54 [ 4 ]      ldab   ED1ba
0021 2107 f7 10 21 [ 4 ]      stab   TCTL2
0022
0023 210a b6 10 23 [ 4 ]      ldaa   TFLG1
0024 210d 84 04    [ 2 ]      anda   #$04
0025 210f b7 10 23 [ 4 ]      staa   TFLG1
0026
0027 2112 86 04    [ 2 ]      ldaa   #$04 ;
0028 2114 b7 10 22 [ 4 ]      staa   TMSK1 ;
0029 2117 0e       [ 2 ]      cli    ;
0030
0031 2118 cc 00 00 [ 3 ]      ldd    #$0000 ;
0032 211b fd 20 00 [ 5 ]      std    $2000
0033 211e 20 fe    [ 3 ]      bouclebra boucle ;
0034
0035
36: symbol Undefined on pass 2
0036 2120 cc 00 00 [ 3 ]      issprog ldd    #ACDC ;
0037 2123 fd 20 00 [ 5 ]      std    $2000 ;
0038
0039 2126 3b       [12 ]      rti
```

Fichier .lst

- La boucle infinie est : bra boucle

- entre sei et cli, on empeche toute activation d'un front actif.

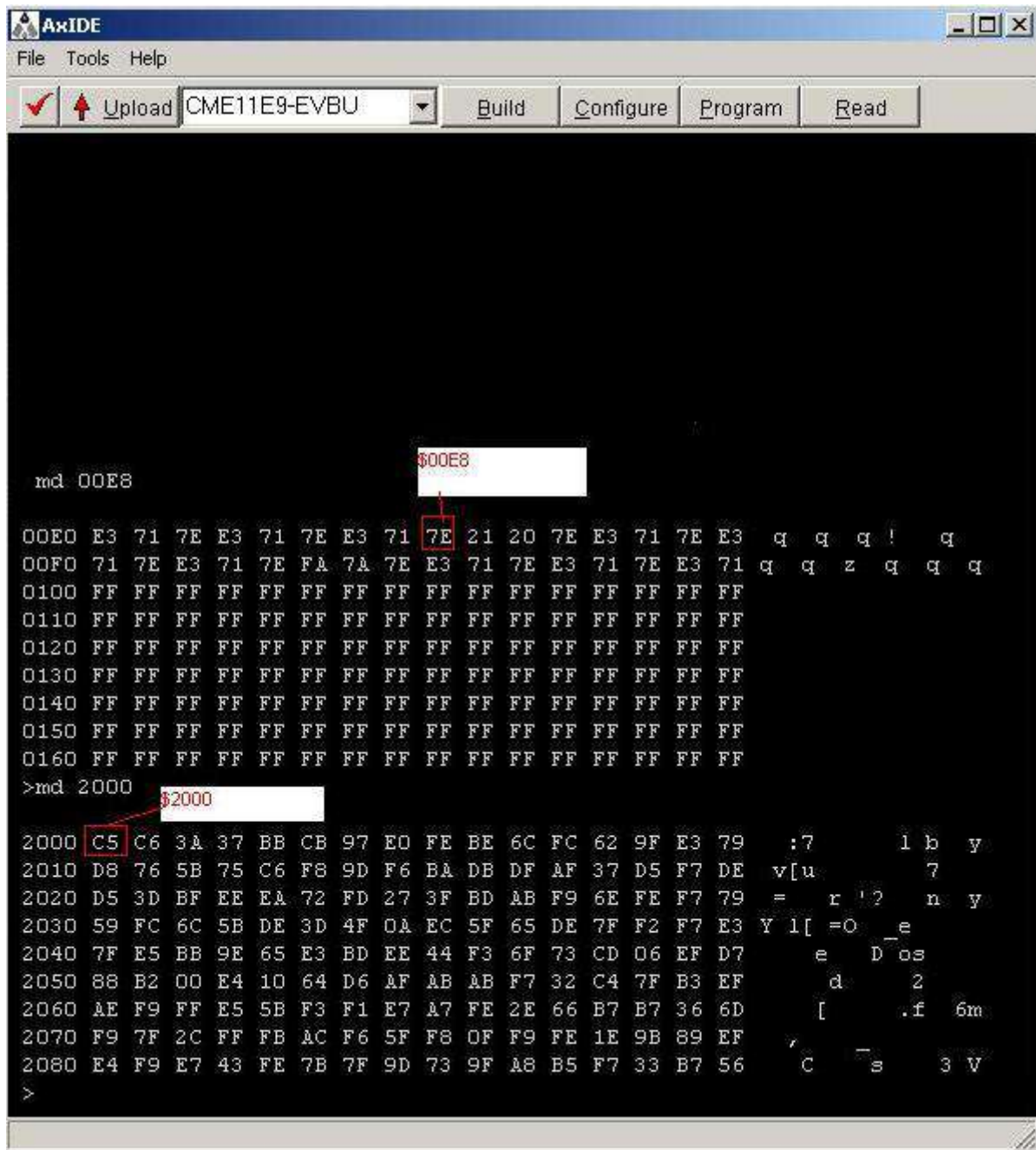
- vecteur d'interruption: 3 adresses: 1e instruction, 2e octet bas, 3e octet haut
vecteur d'interruption du registre TIC1 commence à l'adresse \$00E8
instruction jmp est à l'adresse \$00E8
sous-prog de l'adresse \$00E9 à \$00EA

- évènement capable d'arrêter la boucle = front montant de PA2

- le programme d'interruption

- ldd #\$ACDC : charge la valeur ACDC dans l'accu d

- std \$2000: stocke la valeur contenue dans l'accu d à l'adresse \$2000



```
>call 2100

BUFFALO 3.4 (ext) - Bit User Fast Friendly Aid to Logical Operation
>md 2000

2000 00 00 3A 37 BB CB 97 E0 FE BE 6C FC 62 9F E3 79 :7 1 b y
2010 D8 76 5B 75 C6 F8 9D F6 BA DB DF AF 37 D5 F7 DE v[u 7
2020 D5 3D BF EE EA 72 FD 27 3F BD AB F9 6E FE F7 79 = r '?' n y
2030 59 FC 6C 5B DE 3D 4F 0A EC 5F 65 DE 7F F2 F7 E3 Y l[ =O _e
2040 7F E5 BB 9E 65 E3 BD EE 44 F3 6F 73 CD 06 EF D7 e D os
2050 88 B2 00 E4 10 64 D6 AF AB AB F7 32 C4 7F B3 EF d 2
2060 AE F9 FF E5 5B F3 F1 E7 A7 FE 2E 66 B7 B7 36 6D [ .f 6m
2070 F9 7F 2C FF FB AC F6 5F F8 OF F9 FE 1E 9B 89 EF , _s
2080 E4 F9 E7 43 FE 7B 7F 9D 73 9F A8 B5 F7 33 B7 56 C _s 3 V
>_
```

On retrouve à l'adresse \$00E8 le code du jmp : 7E

A l'adresse \$2000, on a une valeur aléatoire car on n'a pas exécuté le programme

Après exécution, on retrouve à l'adresse \$2000 la valeur 00

En absence du signal PA2-IC1, la valeur de l'octet à l'adresse \$2000 est #0 c'est bien la valeur chargé avant que le programme ne tombe dans la boucle infini.

Après le signal PA2-IC1, la valeur de l'octet à l'adresse \$2000 est #ACDC c'est bien la valeur chargé après que le programme exécute l'instruction issprog.