

Q11 initialisation

Il y a 2 latences en fonctions des commandes :

- Latence de 1.65ms :
 - Clear display, cursor to home
 - Cursor to home
- Latence de 40µs
 - Les autres commandes.

Ldaa #data (charge l'accumulateur A avec la valeur juste après le # (load immédiat))

Staa LCDCMD (stock l'accumulateur A à l'adresse LCDCMD (registre de commande du LCD.

Jsr longue (jum to subroutine 'longue') : saut à un sous programme sans condition.

Sous programmes 'longue' et 'courte' :

Ldx #50000 (charge le registre X avec 50000 (en décimal contrairement aux précédentes valeurs qui débutent par \$ qui sont en hexadécimal)

Dex décrémente registre X

Bne loop_l Branch if not equal to zero à l'adresse loop_l donc 50000 fois

Rts return from subroutine.

Si vous exécutez l'init pas a pas, après la dernière instruction jsr courte vous allez rentrer dans la subroutine 'longue' et à la fin de cette subroutine le logiciel devrait planter parce que l'instruction 'rts' ne saura pas où retourner puisque vous ne serez pas arrivé là par un appel de type 'jsr' (ou autre appel de subroutine)

Ajouter :

Bra suite

juste après le dernier 'jsr courte' par exemple. (le 'suite' qui sera l'adresse des instructions de Q12.

Q12 affichage d'un caractère

Suite ldaa #'A' *charge le caractère 'A' dans l'accumulateur A (chargement immédiat grâce au #)

 staa LCDDAT *stock 'A' dans le registre 'data' du LCD

 jsr courte *délai de 40µs

 ldaa #'B'

 staa LCDDAT

 jsr courte

Q13 Affichage d'une chaîne

La directive fcc permet de déclarer une chaîne de caractères de type constante.

Texto fcc 'La période ='

Fin fcb #0 *octet null juste à la fin de la chaîne de caractères (comme en langage 'C' c'est le marqueur de fin de chaîne de caractère)

Ldy #texto *on charge le registre Y avec l'adresse du premier caractère de 'texto'

Chaîne ldaa 0, y *on charge l'accumulateur A de façon indirecte : adresse contenue dans Y + 0

Beq suite *branche à 'suite' si l'accumulateur A contient le caractère null (la chaîne est complètement transférée vers le LCD)

Staa LCDDAT *sinon on écrit le contenu de l'accumulateur A dans le registre data du LCD

Jsr courte *délai de 40µs

Iny *incrément de Y (donc caractère suivant de 'texto' jusqu'au caractère 'fin' le caractère null)

Bra chaîne *branche toujours à chaîne pour charger le caractère suivant.

Se méfier : il risque d'y avoir 2 labels 'suite', changer le nom de l'un des 2.

Q14 Conversion d'un entier sur 1 octet

Conv ldx #table *charge le registre X avec l'adresse du tableau de caractères '0123...DEF'
* remarque : les registres X et Y peuvent être utilisés pour des adressages indirects

Ldaa octet * octet de test = \$36 en hexa dans l'accumulateur A

Anda #\$0F * on fait un & (immédiat (#)) pour mettre à 0 les 4 bits de poids fort
(\$0F=b00001111)

Adda 0,X * on ajoute à cette valeur le premier caractère du tableau (dont l'adresse est dans X)
cela doit donner '6'

Staa q_bas * le résultat est stocké dans l'octet réservé à l'affichage du quartet bas

Ldaa octet * on charge de nouveau A avec \$36

Asra * décalage de 1 bit vers la droite

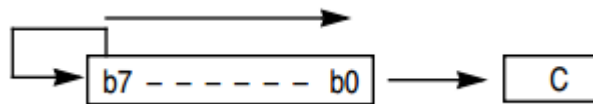
Asra

Asra

Asra * décalage de 4 bits vers la droite, maintenant il y a \$03 dans A

Anda #\$0F

* au cas où les poids forts seraient différents de 0 (lors du décalage le bit de poids le



plus fort reste dans le bit de poids 7

Adda 0,X

* on ajoute au resultat la valeur du contenu du premier octet du tableau. Donc i y a

maintenant '6' dans A

Staa q_haut

*le résultat est stocké dans l'octet réservé à l'affichage du quartet haut

A toi de faire l'organigramme.

Q15 Conversion d'un entier sur 2 octets

Mot Fcb \$45, \$67

Q_hh rmb

Q_hb rmb

Q_bh rmb

Q_bb rmb

Org \$2100

Conv ldx #table

Ldaa mot

Anda #\$0F

adda 0, X

staa q_hb

ldaa mot

asra

asra

asra

asra

anda #\$0F

adda 0,X

staa q_hh

ldaa mot+1

Anda #\$0F

adda 0, X

```
staa    q_bb  
ldaa    mot  
  
asra  
  
asra  
  
asra  
  
asra  
  
anda    #$0F  
adda    0,X  
  
staa    q_bh
```

Remarque :

Il me semble que l'instruction 'adda 0,X' risque de ne pas faire ce que vous souhaitez (à moins que je ne me trompe)

Pour moi le but de créer un tableau avec les caractères '01234 ... BCDEF' est de charger de façon indirecte le caractère en nieme position pour l'afficher.

Or dans le cas présent j'ai plutôt l'impression que l'on charge toujours le caractère '0' pour lui ajouter la valeur contenue dans notre registre, cela va bien marcher pour des valeurs inférieures à 9 ('0' + 1 = '1', '0' + 2 = '2' ...)

Mais pour une valeur de b00001010 soit \$0A soit 10 en décimal '0' + 10 = ' ' (voir la table1 page 1 de votre sujet)

A moins que j'ai mal compris l'instruction 'ADDA (IND, X)'