

Clr SCCR1 \*mise à zéro du registre de contrôle1 de com SCI (situé à l'adresse \$102C)

\*8 bits de données, 1 bit de start et 1 bit de stop (page 239)

Ldaa #\$30 \*charge l'accumulateur A avec la valeur hexa \$30

Staa BAUD \*stocke le contenu de A dans le registre BAUD (situé à l'adresse \$102B) page 237

| BIT 7    | 6 | 5    | 4    | 3    | 2    | 1    | BIT 0 |
|----------|---|------|------|------|------|------|-------|
| *   TCLR | 0 | SCP1 | SCP0 | RCKB | SCR2 | SCR1 | SCR0  |

\*SCP0 et SCP1 à '1'

\*soit une prédivison de 13 ce qui donne une vitesse de transmission de 9600 Baud

\*lorsque la fréquence de cadencement du HC11 est de 8MHz.

Ldab #\$0C \*charge l'accumulateur B avec la valeur hexa \$0C

Stab SCCR2 \*stocke le contenu de B dans le registre SCCR2 (situé à l'adresse \$102D) page 240

| BIT 7   | 6    | 5   | 4    | 3  | 2  | 1   | BIT 0 |
|---------|------|-----|------|----|----|-----|-------|
| *   TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK   |

\*Emission et réception Activées (bits TE et RE à 1), pas d'interruption (bits TIE et RIE

\*à 0.

Clr \$2000 \*efface l'adresse mémoire \$2000.

Loop Ldaa SCSR \*charge le registre A avec le contenu du registre SCSR (situé à l'adresse \$102E)

\*page 242

| BIT 7    | 6  | 5    | 4    | 3  | 2  | 1  | BIT 0 |
|----------|----|------|------|----|----|----|-------|
| *   TDRE | TC | RDRF | IDLE | OR | NF | FE | 0     |

Anda #\$20 \*Effectue un ET logique entre A et la valeur hexa \$20 (pour tester le bit de poids 5)

\*

|      |    |      |      |    |    |    |   |
|------|----|------|------|----|----|----|---|
| TDRE | TC | RDRF | IDLE | OR | NF | FE | 0 |
| &    |    |      |      |    |    |    |   |
| 0    | 0  | 1    | 0    | 0  | 0  | 0  | 0 |
| =    |    |      |      |    |    |    |   |
| 0    | 0  | RDRF | 0    | 0  | 0  | 0  | 0 |

Beq loop \*branch loop tant que A est égal à zéro (tant que RDRF == 0 donc pas de donnée

\* arrivée)

Ldaa SCDR \*charge le registre A avec la donnée située dans le registre SCDR (situé à l'adresse  
\*\$102F)

Staa \$2000 \*stocke la donnée de A à l'adresse \$2000.

Mode de fonctionnement en mode polling : on vérifie en permanence le contenu du bit 5 de SCSR (RDRF) qui passera à 1 lorsque une donnée sera reçue sur la liaison SCI est sera donc présente dans le registre SCDR (registre réception DATA)

Contrairement au mode interruption où lorsque qu'une donnée est reçue le programme principal est interrompu pour traiter la donnée reçue.

L'inconvénient du mode polling est que le programme ne fait rien d'autre que d'attendre une donnée. Le CPU est utilisé à 100% pour vérifier l'arrivée d'une donnée potentielle (à 9600 Baud moins de 960 données par seconde (9600 / 10 parce que 1 bit de start + 8bits de data + 1 bit de stop = 10), cela fait beaucoup de perte pour un CPU tournant à 8MHz.

### Q18

Le vecteur d'interruption SCI se trouve aux adresses \$00C4 et \$00C6 (l'adresse tient sur 2 octets) à cette adresse \$00C4-\$00C6 se trouve l'adresse du point d'entrée du programme d'interruption traitant la liaison série asynchrone.

|             |         |                                                                                  |
|-------------|---------|----------------------------------------------------------------------------------|
| Org         | \$00C4  | ??? sincèrement je ne sais pas, il faut vérifier si cela s'écrit comme ça !      |
|             |         | *ou mettre l'adresse contenue aux adresses \$00C4 et \$00C6 ?                    |
| Org         | \$2180  | *ou comme ça : juste avant le org \$2200 ci-dessous ?                            |
| Jmp         | issprog | *saut à l'adresse de la routine d'interruption                                   |
| Org         | \$2200  |                                                                                  |
| Sei         |         | *Dévalide les interruptions (pour permettre la programmation des registres)      |
| Clr         | SCCR1   | *8 bits + 1 start et 1 stop                                                      |
| Clr         | SCSR    | *clear le registre d'état de la liaison SCI                                      |
| Ldaa        | \$30    | *b0011 0000                                                                      |
| Staa        | BAUD    |                                                                                  |
| Ldaa        | #\$2C   | *b0010 1100                                                                      |
| Staa        | SCCR2   | *Emission Réception activées, mais Interruption uniquement sur Réception         |
|             |         | *regarder le détail du registre SCCR2 page 240 du 68HC11                         |
| Clr         | \$2000  | *efface la donnée en mémoire adresse \$2000                                      |
| Cli         |         | *Autorise les interruptions                                                      |
| Issprog clr | SCSR    | *clear l'interruption                                                            |
| Ldab        | SCDR    | *charge le registre B avec la donnée reçue dans le registre Réception data de la |
|             |         | *liaison SCI                                                                     |

Stab    \$2000                    \*et la stocke en adresse mémoire \$2000

### Q19

Ajouter les adresses des registres du LCD dans les premières lignes du programme :

Ce que l'on doit ajouter **est surligné en jaune** dans ce qui suit

BAUD    equ    \$102b

SCDR    equ    \$102f

...

**LCDCMD equ    \$b5f0**

**LCDDAT equ    \$b5f1**

Org    \$2080    \*(celui que tu as fini par trouver dans la Q17 !)

Jmp    issprog

...

Org    \$2000

Clr    SCCR1

Ldaa    \$30

Staa    BAUD

Ldaa    #\$0C

Staa    SCCR2

**Jsr    INITLCD                    \*ajouter cette ligne pour initialiser le LCD (jump sous routine)**

Clr

Loop    bra    loop

Issprog    clr    SCSR

Ldab    SCDR

Stab    \$2000

**Stab    LCDDAT**

**Jsr    courte**

Org     \$2100

INITLCD   Ldaa     #L4xC20

      Staa     LCDCMD

Et la suite de l'init vue dans la séance 2 sans oublier les routines 'longue' et 'courte'

Jusqu'au

      Ldaa     #CSRHOM

      Staa     LCDCMD

      Jsr       courte

      Rts       \* pour retourner