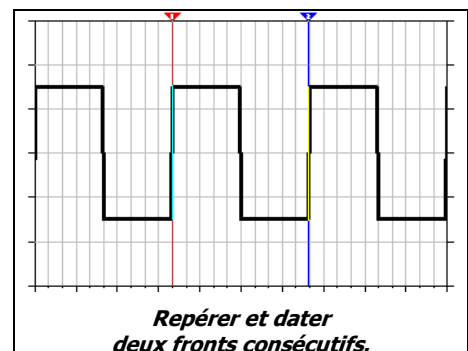


Première séance : Mesure de la Période d'un signal

POSITION DU PROBLEME

La réalisation du fréquencemètre exploite essentiellement la minuterie du 68HC11 dont le cœur est constitué d'un compteur 16 bits appelé le TCNT (voir description en annexe page 11). L'état de ce compteur nous servira à dater l'arrivée des fronts de même nature du signal sous test. Sa période (donc sa fréquence) sera enfin déduite par le calcul d'une différence. L'implémentation logicielle conduit les élèves à développer et à comparer plus exactement deux méthodes programmation :

- La méthode de polling ou de scrutation :
Le programme scrute en permanence l'arrivée du premier front du signal sous test, puis du second, de même nature, et déduit sa période.
- La méthode optimisée par interruption.
Le programme principal est interrompu par l'arrivée d'un premier front et déduit la période du signal à l'arrivée du second front de même nature.



MODALITES DE DEROULEMENT

*Les questions marquées d'un * peuvent être préparées avant la séance mais contrairement aux séances de TP habituelles, aucun compte rendu ne sera exigé à la fin des séances du projet. Les élèves doivent donc consigner les résultats de leurs manipulations et de leurs réflexions avec le plus grand soin dans le but de les présenter dans un rapport final.*

EVALUATION DES ELEVES

L'évaluation du rapport tiendra essentiellement compte de la capacité de synthèse, et bien entendu de l'originalité de cet effort rédactionnel. La note de revue de projet sera composée au fur et à mesure des séances au moyen de validations ponctuelles sur les aspects techniques des réalisations. Les enseignants se réservent toutes les possibilités d'interrogations en rapport avec ces réalisations.

PROGRAMMATION D'UNE E/S PAR SCRUTATION : LE POLLING

CYCLE CPU & CYCLE TIMER

- Q0. Considérons le petit programme expérimental ci-contre dans lequel l'instruction nop (no opération) ne réalise aucune opération.
- Assembler ce programme en activant l'option de comptage des cycles machines. Exécuter ce programme.
 - Combien de cycles séparent la fin des 2 chargements **ldx** et **ldy** ?
 - Relever l'état des registres X et Y et justifier le résultat de Y-X.
 - Justifier le rapport entre l'horloge du CPU et celui du TCNT.

TCNT	equ	\$100E
	org	\$2000
	ldx	TCNT
	nop	
	ldy	TCNT
	rts	

DUREE D'UN PROCESSUS

Q1. La durée d'un processus peut donc être mesurée au moyen d'un encadrement avec deux lectures du TCNT. Or, le jeu d'instruction du 68HC11 ne prévoit pas de soustraction sur les registres X et Y. Cette opération (sur 16b) ne reste possible que sur l'accumulateur D qui est la concaténation des registres A et B.

- Assembler le programme ci-contre et relever le nombre de cycles qui séparent les deux opérations de chargement.
- Relever le contenu des registres ainsi que le contenu de la mémoire à l'adresse \$2002 après l'exécution du programme.
- Quelle serait l'erreur commise dans le calcul de la durée du processus encadré ci-contre.

TCNT	equ	\$100E
initial	org	\$2000
delta	rmb	2
	rmb	2
	org	\$2050
	ldd	TCNT
	std	initial
Processus		
	ldd	TCNT
	subd	initial
	std	delta
	rts	

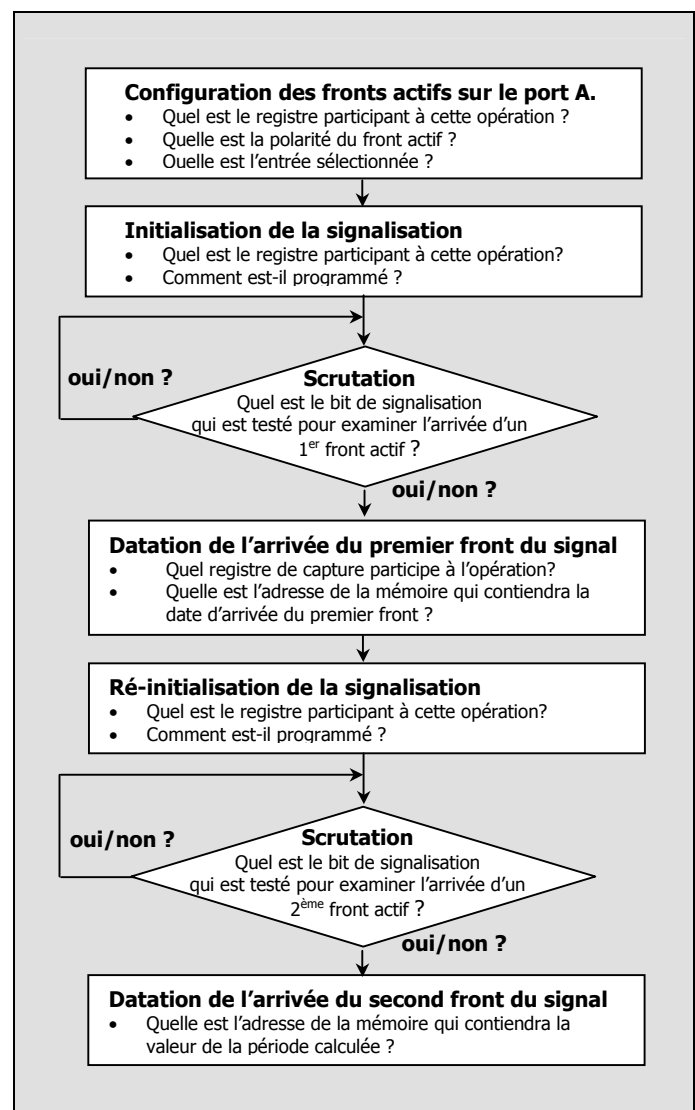
MESURE D'UNE PERIODE

Q2*. Se reporter au schéma fonctionnel de la minuterie (Le timer en annexe page 11) et noter tous les registres acteurs d'un scénario utilisant PA2 comme entrée du signal sous test. Refaire le même travail pour PA1 et pour PA0.

Q3. Examinons maintenant le programme ci-dessous.

- Compléter tout d'abord les deux lignes incomplètes du programme.
- Commenter ce programme ligne par ligne et compléter son organigramme en répondant aux questions.
- Assembler ce programme et relever le total de tous les cycles machines qui séparent la fin des deux opérations de chargement dans l'accumulateur D.

TIC1	equ	\$1010
TFLG1	equ	\$1023
TCTL2	equ	\$1021
	org	\$2050
front	rmb	2
period	rmb	2
ED1ba	fcbl	\$10
	org	\$2100
	ldab	ED1ba ; commenter
	stab	TCTL2 ; la valeur du
		; TCTL2
	ldaa	TFLG1 ; initialisation
	anda	#\$04 ; de ???
	staa	TFLG1 ; ???
lire1	ldaa	TFLG1 ; décrire le
	anda	#\$04 ; test réalisé
	beq	lire1 ; ici
suite1	ldd	TIC1
	std	front
	ldaa	TFLG1 ; initialisation
	anda	#\$04 ; de ???
	staa	TFLG1
lire2	ldaa	TFLG1
	anda	
	beq	
suite2	ldd	TIC1
	subd	front
	std	period
	rts	



- Q4. Exécuter ce programme pour des fréquences allant de 50Hz à 50 kHz
- Consigner le contenu de l'accumulateur D ainsi que sa conversion en base 10 dans le tableau ci-contre.
 - Reporter ces mesures dans un repère log-log et justifier l'allure de cette représentation graphique.
 - Valider et commenter** ces résultats devant un enseignant.

```
>call 2100
P-2100 Y-FFFF X-FFFF A-07 B-CE C-D0 S-004
```

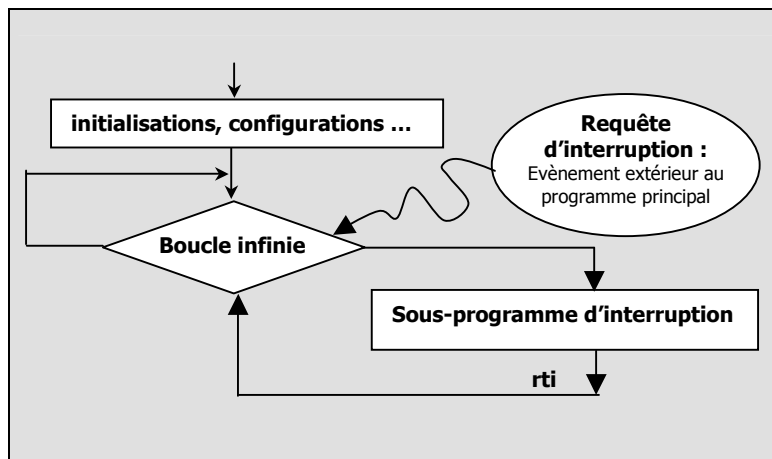
f (Hz)	Polling
40	
100	
400	
1k	$(07CE)_{16} = (1988)_{10}$
4k	
10k	
40k	

- Q5. Effectuer quelques mesures en BF et en HF afin de cerner les limites de cette méthode.
- Justifier la limite en HF (vers 65kHz) en comparant le résultat de la mesure au nombre de cycles machines qui séparent la fin des deux opérations de chargement dans l'accumulateur D.
 - Justifier la limite en BF (vers 30kHz) en comparant le résultat de la mesure à limite toute simple d'un compteur 16b.

PROGRAMMATION D'UNE E/S PAR INTERRUPTION

INTERRUPTION D'UN PROGRAMME

- Q6. Le programme expérimental ci-contre est conçu pour permettre une prise en main des processus d'interruptions. Après une première phase de paramétrage, le programme s'enferme dans une boucle infinie.
- Identifier cette boucle. Décrire le résultat de son exécution.
 - Identifier le vecteur d'interruption à partir du fragment de code encadré ci-contre. Quel est l'évènement capable d'interrompre la boucle ?
 - Décrire le résultat du programme d'interruption.
 - Détailler l'organigramme du programme.
- Q7. Charger le programme et relever l'état de la mémoire aux adresses : \$2000 et \$00E8 **avant** l'exécution. Commentaires.
- Lancer le programme et relever l'adresse \$2000 en **l'absence** de signal sur PA2-IC1. Commentaires.
 - Relancer le programme et relever l'adresse \$2000 en présence d'un TTL (1000Hz) sur PA2-IC1. **Validation.**



```

TIC1 equ $1010
TFLG1 equ $1023
TCTL2 equ $1021
TMSK1 equ $1022 ;

front org $2050
period rmb 2
ED1ba rmb 2
      fcb $10

      org $00E8 ;
      jmp issprog ;

      org $2100

      sei ;

      lds #$0200

      ldab ED1ba
      stab TCTL2

      ldaa TFLG1
      anda #$04
      staa TFLG1

      ldaa #$04 ;
      staa TMSK1 ;
      cli ;

      ldd #$0000 ;
      std $2000 ;
boucle bra boucle ;

issprog ldd #$ACDC ;
        std $2000 ;

        rti
  
```

- Q8.** Examinons maintenant le programme ci-contre.
- Compléter tout d'abord les deux lignes incomplètes du programme.
 - Commenter ce programme ligne par ligne et donner tous les **détails** de son organigramme en distinguant les tâches réalisées par le programme principal des tâches de la routine d'interruption.
- Q9.** Exécuter ce programme pour des fréquences allant de 50Hz à 50 kHz
- Consigner le résultat des périodes mesurées ainsi que leur conversion en base 10 dans le tableau ci-dessous.
 - Reporter ces mesures dans le même repère log-log.
 - Valider et commenter ces résultats devant un enseignant.

f (Hz)	Interruption
40	
100	
400	
1k	
4k	
10k	
40k	

- Q10.** Effectuer quelques mesures en BF et en HF afin de cerner les limites de cette méthode.
- Comparaison des limites et des avantages de chaque méthode de programmation (polling/interruption).

```

TIC1 equ $1010
TFLG1 equ $1023
TCTL2 equ $1021
TMSK1 equ $1022 ;

        org $2050
front1 rmb 2
front2 rmb 2
period rmb 2
ED1ba fcb $10
N rmb 1

        org $00E8 ;
        jmp issprog ;

        org $2100

        sei ;

        lds #$0200

        ldab ED1ba
        stab TCTL2 ;

        ldaa TFLG1
        anda #$04
        staa TFLG1 ;

        ldaa #$04 ;
        staa TMSK1 ;
        cli ;

        ldab #2
        stab N

boucle  tst N
        bne boucle ;

        clr TMSK1

        ldd TIC1
        std .....

        subd .....
        std period

        rts

issprog ldaa TFLG1
        anda #$04
        staa TFLG1 ;

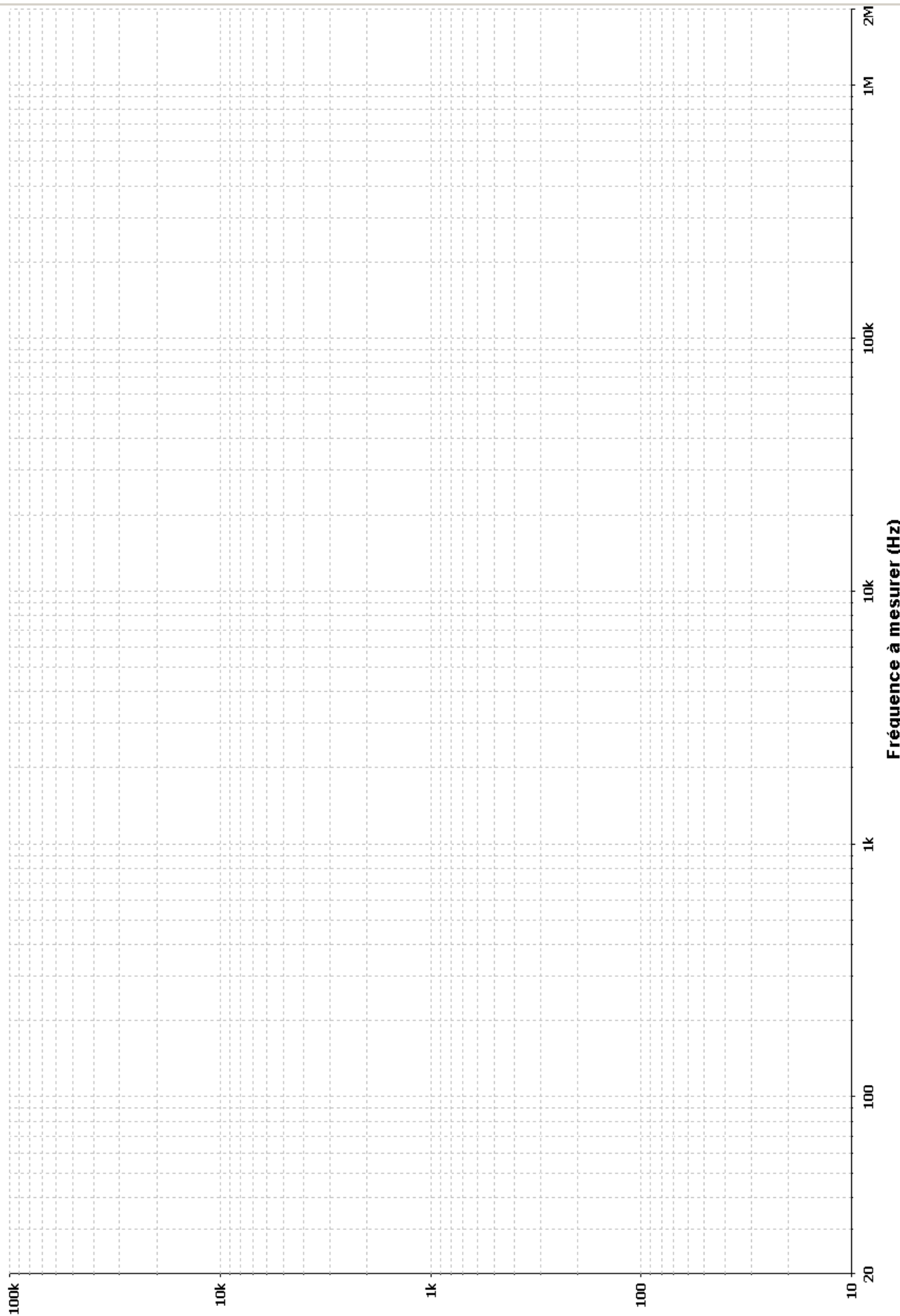
        dec N
        beq sortie

        ldx TIC1
        stx front1

sortie  rti

```

Linéarité de la mesure



2^{ème} et 3^{ème} séance : Programmation de fonctionnalités d'E-S

Les élèves les plus avancés ont réussi à implémenter la mesure d'une période sur la maquette expérimentale 68HC11. Il s'agit maintenant de mettre en œuvre l'affichage de cette mesure sur un écran LCD et d'étudier un moyen pour lancer une mesure à partir d'un PC.

PROGRAMMATION D'UN LCD

Ce dispositif dont la connectique est représentée ci-contre est accessible au moyen de deux registres :

- Le registre de commande à l'adresse \$B5F0
- Le registre de données à l'adresse \$B5F1

B5F0 Command Register	+5V	2	1	GND
B5F1 Data Register	A0	4	3	Vee
	/LCDCS	6	5	/RW
	D1	8	7	D0
	D3	10	9	D2
	D5	12	11	D4
	D7	14	13	D6

Le constructeur réunit dans les tables ci-dessous la listes des commandes du LCD ainsi que son jeu de caractères. La table des commandes met aussi en évidence les latences liées à la réponse du dispositif et la table2 ne résulte que d'un codage particulier de caractères.

TABLE 1. LCD Command Codes

Command codes are used for LCD setup and control of character and cursor position. All command codes are written to LCD panel address \$B5F0. The BUSY flag (bit 7) should be tested before any command updates to verify that any previous command is completed. A read of the command address \$B5F0 will return the BUSY flag status and the current display character location address.

Command	Code	Delay
Clear Display, Cursor to Home	\$01	1.65ms
Cursor to Home	\$02	1.65ms
Entry Mode:		
Cursor Decrement, Shift off	\$04	40us
Cursor Decrement, Shift on	\$05	40us
Cursor Increment, Shift off	\$06	40us
Cursor Increment, Shift on	\$07	40us
Display Control:		
Display, Cursor, and Cursor Blink off	\$08	40us
Display on, Cursor and Cursor Blink off	\$0C	40us
Display and Cursor on, Cursor Blink off	\$0E	40us
Display, Cursor, and Cursor Blink on	\$0F	40us
Cursor / Display Shift: (nondestructive move)		
Cursor shift left	\$10	40us
Cursor shift right	\$14	40us
Display shift left	\$18	40us
Display shift right	\$1C	40us
Display Function (default 2x40 size)	\$3C	40us
Character Generator Ram Address set	\$40-\$7F	40us
Display Ram Address set	\$80-\$FF	40us

TABLE 2. LCD Character Codes

\$20 Space	\$2D -	\$3A :	\$47 G	\$54 T	\$61 a	\$6E n	\$7B {
\$21 !	\$2E .	\$3B ;	\$48 H	\$55 U	\$62 b	\$6F o	\$7C
\$22 "	\$2F /	\$3C {	\$49 I	\$56 V	\$63 c	\$70 p	\$7D }
\$23 #	\$30 0	\$3D =	\$4A J	\$57 W	\$64 d	\$71 q	\$7E >
\$24 \$	\$31 1	\$3E }	\$4B K	\$58 X	\$65 e	\$72 r	\$7F <
\$25 %	\$32 2	\$3F ?	\$4C L	\$59 Y	\$66 f	\$73 s	
\$26 &	\$33 3	\$40 Time	\$4D M	\$5A Z	\$67 g	\$74 t	
\$27 '	\$34 4	\$41 A	\$4E N	\$5B [	\$68 h	\$75 u	
\$28 (	\$35 5	\$42 B	\$4F O	\$5C Yen	\$69 i	\$76 v	
\$29)	\$36 6	\$43 C	\$50 P	\$5D]	\$6A j	\$77 w	
\$2A *	\$37 7	\$44 D	\$51 Q	\$5E ^	\$6B k	\$78 x	
\$2B +	\$38 8	\$45 E	\$52 R	\$5F _	\$6C l	\$79 y	
\$2C ,	\$39 9	\$46 F	\$53 S	\$60 `	\$6D m	\$7A z	

PROCEDURE INITIALISATION

Q11. Le programme LCDinit1 ci-contre, illustre une façon d'initialiser le LCD

- Cette procédure fait appel à deux sous-programmes de temporisation **longue** et **courte** après chaque écriture dans le registre de commande. Pourquoi ?
- Assembler ce programme en activant l'option de comptage des cycles CPU. Corriger la durée des boucles de temporisation si nécessaire.
- Examiner la fonction réalisée à chaque étape de cette procédure au moyen d'une exécution pas-à-pas.

AFFICHAGE D'UN CARACTERE

Q12. Compléter le programme ci-dessous de manière à afficher l'un après l'autre les caractères A, B, C et D.

Init			
suite	ldaa	#'A'	*commentaires
	staa		
	jsr	courte	
		#'B'	
	staa		
		courte	
		#'C'	
	staa		
		courte	
		#'D'	
	staa		
		courte	

LCDCMD	equ	\$b5f0
LCDDAT	equ	\$b5f1
L4xC20	equ	\$3c * 4x20
CLRCH	equ	\$01
DCBON	equ	\$0f
CINSHOF	equ	\$06
SCSHR	equ	\$14
CSRHOM	equ	\$02
	ORG	\$2100
init	ldaa	#L4xC20
	staa	LCDCMD
	jsr	longue
	ldaa	#CLRCH
	staa	LCDCMD
	jsr	longue
	ldaa	#DCBON
	staa	LCDCMD
	jsr	courte
	ldaa	#CINSHOF
	staa	LCDCMD
	jsr	courte
	ldaa	#SCSHR
	staa	LCDCMD
	jsr	courte
	ldaa	#CSRHOM
	staa	LCDCMD
	jsr	courte
* longue	ldx	#50000
loop_l	dex	
	bne	loop_l
	rts	
* courte	ldx	#1000
loop_c	dex	
	bne	loop_c
	rts	

AFFICHAGE D'UNE CHAINE

Q13. Examinons maintenant le programme ci-contre dans lequel les labels **init** et **suite** font appel aux procédures décrites en Q11 et en Q12.

- Quelle est la fonction réalisée par la directive **fcc** ?
- Examiner la fonction réalisée par l'instruction de chargement en adressage indexé (voir la petite doc du HC11). Commenter chaque ligne du programme.
- Intégrer ce programme dans celui de Q2 et procéder à son exécution.
- Validation.

texto	org	\$2000
fin	fcc	'La periode='
	fcbl	#0
	ORG	\$2100
init		* Voir q11
chaîne	ldy	#texto
	ldaa	0,y
	beq	suite
	staa	LCDDAT
	jsr	courte
	iny	
	bra	chaîne
suite	...	* Voir q12

CONVERSION D'UN ENTIER SUR 1 OCTET

- Q14. Nous complétons le dernier programme avec une table de correspondance à l'adresse \$2020 est un algorithme pour la conversion d'un entier de 1 octet (2 quartet) en deux caractères (quartet haut et quartet bas) à partir du label **conv**.
- Compléter les lignes manquantes et décrire cet algorithme au moyen d'un organigramme.
 - Assembler et exécuter ce programme.
 - Validation.

CONVERSION D'UN ENTIER SUR 2 OCTET

- Q15. Nous savons que la précision de notre mesure de fréquence atteint 2 octets.
- Modifier alors le programme ci-contre de manière à réaliser une conversion sur 2 octets.
 - Test sur 2 octets quelconques.
 - Validation.
- Q16. Intégrer ce programme dans celui de la première séance de manière à pouvoir appliquer cette conversion à la période mesurée par une des deux méthodes étudiée à la première séance.
- Affichage de la période mesurée, comparaison avec les mesures de la première séance.

org	\$2000	
texto	fcc	'La periode='
fin	fcf	#0
	org	\$2020
table	fcc	'0'
	fcc	'1'
	fcc	'2'
	fcc	'3'
	fcc	'4'
	fcc	'5'
	fcc	'6'
	fcc	'7'
	fcc	'8'
	fcc	'9'
	fcc	'A'
	fcc	'B'
	fcc	'C'
	fcc	'D'
	fcc	'E'
	fcc	'F'
octet	fcf	#\$36
q_haut	rmb	1
q_bas	rmb	1
	ORG	\$2100
conv	ldx	#table
	ldaa	octet
	anda	#\$0F
	adda	0,x
	staa	q_bas
		*quartet bas
	ldaa	octet
	asra	
		
		
	asra	
	anda	#\$0F
		
	staa	q_haut
		*quartet haut
init	...	* voir q1
chaîne	ldy	#texto
	ldaa	0,y
	beq	suite
	staa	LCDDAT
	jsr	courte
	iny	
	bra	chaîne
suite	ldaa	q_haut
	staa	LCDDAT
	jsr	courte
	ldaa	q_bas
	staa	LCDDAT
	jsr	courte
	ldaa	#'.'
	staa	LCDDAT
	jsr	courte
	ldaa	#'.'
	staa	LCDDAT
	jsr	

Troisième séance : Communication par liaison série asynchrone

Après l'étude de deux méthodes d'E/S appliquée à mesure de la période d'un signal, les élèves les plus avancés ont réussi à mettre en œuvre un module d'affichage au moyen d'un LCD. Pour finir, le travail d'aujourd'hui vise, le déclenchement de l'affichage à partir de la saisie d'un caractère dans AxIDE.

L'interface de communication série, le SCI (Serial Communication Interface), utilise les bits 0 et 1 du port D et communique avec le processeur au moyen de cinq registres que nous définissons partiellement dans ce qui suit.

Le constructeur illustre le schéma fonctionnel de l'interface série avec les figures 7.1 et 7.2. Enfin, les connexions PD0(RxD) et PD1(TxD) avec un terminal utilisateur sont mis en évidence sur le circuit de la figure B1 fournit par le constructeur (cf. doc HC11E9).

REGISTRES DE VITESSE ET DE DONNEES

La vitesse de la liaison est programmable au moyen des bits SCP0,1,2 et SCR0,1,2 du registre BAUD. La table 7-1 de la section 7 consacré au SCI donne tous les détails de programmation nécessaire en fonction de l'horloge du processeur.

BAUD \$102B			SCP1	SCP0		SCR2	SCR1	SCR0			R-T7	R-T6	R-T5	R-T4	R-T3	R-T2	R-T1	R-T0	SCDR \$102F

La donnée à transmettre ou à recevoir, quand à elle, sera consignée dans le registre SCDR (Serial Communication Data Register) à l'adresse \$102F.

REGISTRE D'ETAT : SCSR

Le Serial Com. Status Register consigne l'état de la liaison. Plus exactement :

TDRE		RDRE						SCSR \$102E
Etat du SCDR								

- RDRF=1 : Receive Data Register Full Flag.
Signale la disponibilité d'une donnée dans le SCDR pour être lue.
- TDRE=1 : Transmitt Data Register Empty Flag.
Signale que le registre de donnée est vide.

REGISTRE DE CONTROLE SCCR1

On note seulement le rôle du bit M sur le mode de transmission.

			M					SCCR1 \$102C
M=0 : 8 bits données + 1 bit de start + 1 stop								

REGISTRE DE CONTROLE SCCR2

TE et TR activent l'émission ou la réception.

- RE=1 : Emission activée.
- TE=1 : Réception activée.
- RE=0 : Emission désactivée.
- TE=0 : Réception désactivée.

TIE		RIE		TE	RE		SBK	SCCR2 \$102D
Interruptions				Activation			0	

Les bits TIE (Transmitt interrupt enable) et RIE (Receive interrupt enable) permettent d'activer ou non les interruptions provenant du SCI.

- TIE=1 : Interruption autorisée par le bit TDRE du registre d'état.
- TIE=0 : Interruption par TDRE interdite.
- RIE=1 : Interruption autorisée par le bit RDRF (RDRF =1).
- RIE=0 : Interruption par RDRF interdite.

- Q17. Le programme ci-contre montre une façon de communiquer avec le port série en mode polling (scrutation).
- Commenter ce programme et dresser son organigramme.
 - Noter la vitesse de la communication (cf. section 7 p-156 doc HC11E9) ainsi que la configuration du SCI en émission ou en réception.
 - Justifier le mode de fonctionnement en mode Polling.
 - Assembler et exécuter ce programme.
 - Sortir de la boucle infinie au moyen de la saisie d'un caractère au clavier. Noter le code associé à ce caractère et relever l'état de la mémoire à l'adresse \$2000.
 - Quel est l'inconvénient de ce mode de programmation ?

BAUD	equ	\$102b
SCDR	equ	\$102f
SCSR	equ	\$102e
SCCR1	equ	\$102c
SCCR2	equ	\$102d
	org	\$2200
	clr	SCCR1
	ldaa	#\$30
	staa	BAUD
	ldab	#\$0C
	stab	SCCR2
	clr	\$2000
loop	ldaa	SCSR
	anda	#\$20
	beq	loop
	ldaa	SCDR
	staa	\$2000
	rts	

- Q18. S'inspirer du code ci-dessous pour modifier ce programme de manière à fonctionner en interruption.
- Noter d'abord l'adresse du vecteur d'interruption du SCI.
 - Assembler et exécuter ce programme.
 - Sortir de la boucle infinie au moyen de la saisie d'un caractère au clavier. Noter le code associé à ce caractère et relever l'état de la mémoire à l'adresse \$2000.
 - **Valider** le bon fonctionnement de ce programme.

- Q19. Adapter ce programme pour lancer un affichage sur le LCD après la saisie d'un caractère sur le clavier.
Validation

BAUD	equ	\$102b
SCDR	equ	\$102f
SCSR	equ	\$102e
SCCR1	equ	\$102c
SCCR2	equ	\$102d
	org	\$.....
	jmp	issprog
	org	\$2200
	sei	
	clr	SCCR1
	clr	SCSR
	ldaa	#\$30
	staa	BAUD
	ldaa	#\$2C
	staa	
	clr	\$2000
	cli	
loop	bra	loop
issprog	clr	SCSR
	ldab	SCDR
	stab	\$2000
	rti	

Les principales fonctionnalités de la minuterie du microcontrôleur sont résumées ici sous une forme très synthétique dans le but de réunir les informations nécessaires à la conduite des prochaines manipulations. Il convient de préciser que l'emploi de cet outil n'est pas très difficile, cependant, sa maîtrise demande beaucoup d'exercices et dépasse sans doute le cadre de notre étude. Pour plus de détails, nous invitons donc les élèves à consulter la Section-9 de la documentation du 68HC11E9.

**LE CŒUR DE
LA MINUTERIE**

Au centre du Timer se trouve un compteur 16b, le TCNT, à lecture seule, sur lequel aucune action d'écriture, d'interruption ou de mise à zéro n'est possible.

TCNT Compteur 16b	B ₁₅	B ₁₄	B ₁₃	B ₁₂	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀	Adresse : \$100E-100F
	MSB à l'adresse 100E								LSB à l'adresse 100F								

**REGISTRES DE
COMPARAISON**

Le contenu du TCNT peut être comparé à celui de 4 registres 16b, (Timer Output Compare Register), TOC1, TOC2, TOC3 et TOC4.

TOC1 Registre 16b	B ₁₅ B ₁₄ B ₁₃ B ₁₂ B ₁₁ B ₁₀ B ₉ B ₈ B ₇ B ₆ B ₅ B ₄ B ₃ B ₂ B ₁ B ₀	Adresse : \$1016-1017
	MSB à l'adresse 1016 LSB à l'adresse 1017	
TOC2 Registre 16b	B ₁₅ B ₁₄ B ₁₃ B ₁₂ B ₁₁ B ₁₀ B ₉ B ₈ B ₇ B ₆ B ₅ B ₄ B ₃ B ₂ B ₁ B ₀	Adresse : \$1018-1019
	MSB à l'adresse 1018 LSB à l'adresse 1019	
TOC3 Registre 16b	B ₁₅ B ₁₄ B ₁₃ B ₁₂ B ₁₁ B ₁₀ B ₉ B ₈ B ₇ B ₆ B ₅ B ₄ B ₃ B ₂ B ₁ B ₀	Adresse : \$101A-101B
	MSB à l'adresse 101A LSB à l'adresse 101B	
TOC4 Registre 16b	B ₁₅ B ₁₄ B ₁₃ B ₁₂ B ₁₁ B ₁₀ B ₉ B ₈ B ₇ B ₆ B ₅ B ₄ B ₃ B ₂ B ₁ B ₀	Adresse : \$101C-101D
	MSB à l'adresse 101C LSB à l'adresse 101D	

**MEMORISATION
EVENEMENTIELLE**

Le contenu du TCNT peut être mémorisé dans 3 registres dits de capture d'entrée (Timer Input Capture Register) TIC1, TIC2, et TIC3 à l'arrivée d'un front actif (montant ou descendant) sur les entrées respectives PA0, PA1 et PA2 du port A. Le RESET n'altère pas le contenu de ces registres.

TIC1 Registre 16b	<table><tr><td>B₁₅</td><td>B₁₄</td><td>B₁₃</td><td>B₁₂</td><td>B₁₁</td><td>B₁₀</td><td>B₉</td><td>B₈</td><td>B₇</td><td>B₆</td><td>B₅</td><td>B₄</td><td>B₃</td><td>B₂</td><td>B₁</td><td>B₀</td></tr><tr><td colspan="8">MSB à l'adresse 1010</td><td colspan="8">LSB à l'adresse 1011</td></tr></table>	B ₁₅	B ₁₄	B ₁₃	B ₁₂	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀	MSB à l'adresse 1010								LSB à l'adresse 1011								Adresse : \$1010-1011
B ₁₅	B ₁₄	B ₁₃	B ₁₂	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀																			
MSB à l'adresse 1010								LSB à l'adresse 1011																										
TIC2 Registre 16b	<table><tr><td>B₁₅</td><td>B₁₄</td><td>B₁₃</td><td>B₁₂</td><td>B₁₁</td><td>B₁₀</td><td>B₉</td><td>B₈</td><td>B₇</td><td>B₆</td><td>B₅</td><td>B₄</td><td>B₃</td><td>B₂</td><td>B₁</td><td>B₀</td></tr><tr><td colspan="8">MSB à l'adresse 1012</td><td colspan="8">LSB à l'adresse 1013</td></tr></table>	B ₁₅	B ₁₄	B ₁₃	B ₁₂	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀	MSB à l'adresse 1012								LSB à l'adresse 1013								Adresse : \$1012-1013
B ₁₅	B ₁₄	B ₁₃	B ₁₂	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀																			
MSB à l'adresse 1012								LSB à l'adresse 1013																										
TIC3 Registre 16b	<table><tr><td>B₁₅</td><td>B₁₄</td><td>B₁₃</td><td>B₁₂</td><td>B₁₁</td><td>B₁₀</td><td>B₉</td><td>B₈</td><td>B₇</td><td>B₆</td><td>B₅</td><td>B₄</td><td>B₃</td><td>B₂</td><td>B₁</td><td>B₀</td></tr><tr><td colspan="8">MSB à l'adresse 1014</td><td colspan="8">LSB à l'adresse 1015</td></tr></table>	B ₁₅	B ₁₄	B ₁₃	B ₁₂	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀	MSB à l'adresse 1014								LSB à l'adresse 1015								Adresse : \$1014-1015
B ₁₅	B ₁₄	B ₁₃	B ₁₂	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀																			
MSB à l'adresse 1014								LSB à l'adresse 1015																										

**LES FRONTS ACTIFS
SUR PA0, 1, 2 ET PA3**

Les fronts (Edge) actifs arrivant sur PA0,1,2 et PA3 sont indépendamment configurables au moyen du registre de contrôle TCTL2 comme cela est indiqué ci-contre.

TCTL2 \$1021	ED4b	ED4a	ED1b	ED1a	ED2b	ED2a	ED3b	ED3a
	Config PA2		Config PA1		Config PA0			

EDxb	EDxa	Configuration
0	0	Désactivation de tous les fronts
0	1	Activation des fronts montants
1	0	Activation des fronts descendants
1	1	Activation de tous les fronts

MASQUAGE ET SIGNALISATION - I

Les résultats de comparaisons entre le TCNT et les TOCx ainsi que l'arrivée de fronts actifs sur les entrées PA0, PA1 et PA2, marqués par un 1 ou un 0, sont consignés dans un registre de signalisation, le TFLG1 (Timer Flag Register 1). Le registre TMSK1 (Timer Mask Register) permet à son tour d'inhiber ou non, une requête d'interruption du programme qui occupe le processeur.

Autrement dit :

- ICxi=1 : Interruption autorisée suite à l'enregistrement d'un front actif par ICxF.
- ICxi=0 : Interruption défendue même après l'enregistrement d'un front actif par ICxF.

TMSK1 \$1022	OC1i OC2i OC3i OC4i	O5i I4i	IC1i IC2i IC3i	OC1F OC2F OC3F OC4F	O5F I4F	IC1F IC2F IC3F	TFLG1 \$1023
	Autorisations des interrup. par OCxF		Aut. interrup. pa ICxF	Si TOCx = TCNT Alors OcxF=1		Enregistrement des fronts actifs	

Un examen rapide du schéma fonctionnel du timer (page 3) nous conduit aussi à noter que :

- IC3F signale l'arrivée d'un front actif sur PA0 en mémorisant l'état du TCNT sur TIC3.
- IC2F signale l'arrivée d'un front actif sur PA1 en mémorisant l'état du TCNT sur TIC2.
- IC1F signale l'arrivée d'un front actif sur PA2 en mémorisant l'état du TCNT sur TIC1.

Le bit O5F/I4F de TFLG configure les adresses 101E-101F en TIC4 ou en TOC5. Dans le cas d'une programmation en registre de capture, le bit O5F/I4F indiquera la présence d'un front actif sur PA3 en mémorisant l'état du TCNT sur TIC4/TOC5.

TIC4/TOC5 Registre 16b	B ₁₅ B ₁₄ B ₁₃ B ₁₂ B ₁₁ B ₁₀ B ₉ B ₈ B ₇ B ₆ B ₅ B ₄ B ₃ B ₂ B ₁ B ₀	Adresse : \$101E-101F
	MSB à l'adresse 101E	LSB à l'adresse 101F

MASQUAGE ET SIGNALISATION - II

A partir d'un RESET, le compteur TCNT s'incrémente de \$0000 à \$FFFF et repasse par \$0000 en mettant un bit de dépassement à 1. Ce bit de dépassement que nous désignons par TOF (Timer Overflow Flag) est consigné dans le registre TFLG2 à l'adresse \$1025 de la mémoire.

TMSK2 \$1024	TOI	PR1	PRO	TOF	TFLG2 \$1025
	Autorisation des interruptions par TOF	Division horloge		Dépassement du TCNT	

La vitesse d'incrémentation du compteur TCNT est quantifiée sur 4 niveaux et peut être programmée au moyen des bits PR0 et PR1 du registre TMSK2 comme cela est résumé sur le tableau ci-contre. Le bit de poids fort de ce registre, le TOI (Timer Overflow Interrupt Enable Bit) autorise l'interruption du programme qui occupe le CPU par le TOF.

PR1	PR0	Division de E-clock	Cycle TCNT	Cycle Overflow
0	0	1 (2MHz)	500 ns	32,77 ms
0	1	4 (500 kHz)	2 µs	131,07 ms
1	0	8 (250 kHz)	4 µs	262,14 ms
1	1	16 (125 kHz)	8 µs	524,29 ms

Attention : L'écriture sur TMSK2 n'est possible que durant 64 cycles du CPU après un RESET.

VECTEURS D'INTERRUPTION

Les vecteurs d'interruptions en rapport avec la minuterie sont encadrés ci-contre sur les données fournies par le constructeur.

Chaque vecteur possède 3 coordonnées qui sont des adresses consécutives de un octet (V1, V2, V3) dans la mémoire. Toute requête d'interruption pointe le programme à l'adresse V1.

Par conséquent, suite à une requête extérieure, l'appel d'un sous-programme peut être simplement réalisé en chargeant le code d'un saut (JMP) à l'adresse V1 et l'adresse du sous-programme en V2 (MSB) et en V3 (LSB).

Interrupt Vector Jump Table	
INTERRUPT VECTOR	FIELD
Serial communications Interface (SCI)	\$00C4-\$00C6
Serial Peripheral Interface (SPI)	\$00C7-\$00C9
Pulse Accumulator Input Edge	\$00CA-\$00CC
Timer Overflow	\$00DD-\$00DE
Timer Output Compare 5	\$00D3-\$00D5
Timer Output Compare 4	\$00D6-\$00D8
Timer Output Compare 3	\$00D9-\$00DB
Timer Output Compare 2	\$00DC-\$00DE
Timer Output Compare 1	\$00DF-\$00E1
Timer Input Capture 3	\$00E2-\$00E4
Timer Input Capture 2	\$00E5-\$00E7
Timer Input Capture 1	\$00E8-\$00EA
Real Time Interrupt	\$00EB-\$00ED
IRQ	\$00EE-\$00F0
XIRQ	\$00F1-\$00F3
Software Interrupt (SWI)	\$00F4-\$00F6
Illegal Opcode	\$00F7-\$00F9
Computer Operating Properly (COP)	\$00FA-\$00FC
Clock Monitor	\$00FD-\$00FF

To use vectors specified in the table, the user must insert a jump extended opcode in the byte field of the vector required. For example, for the IRQ vector, the following is performed:

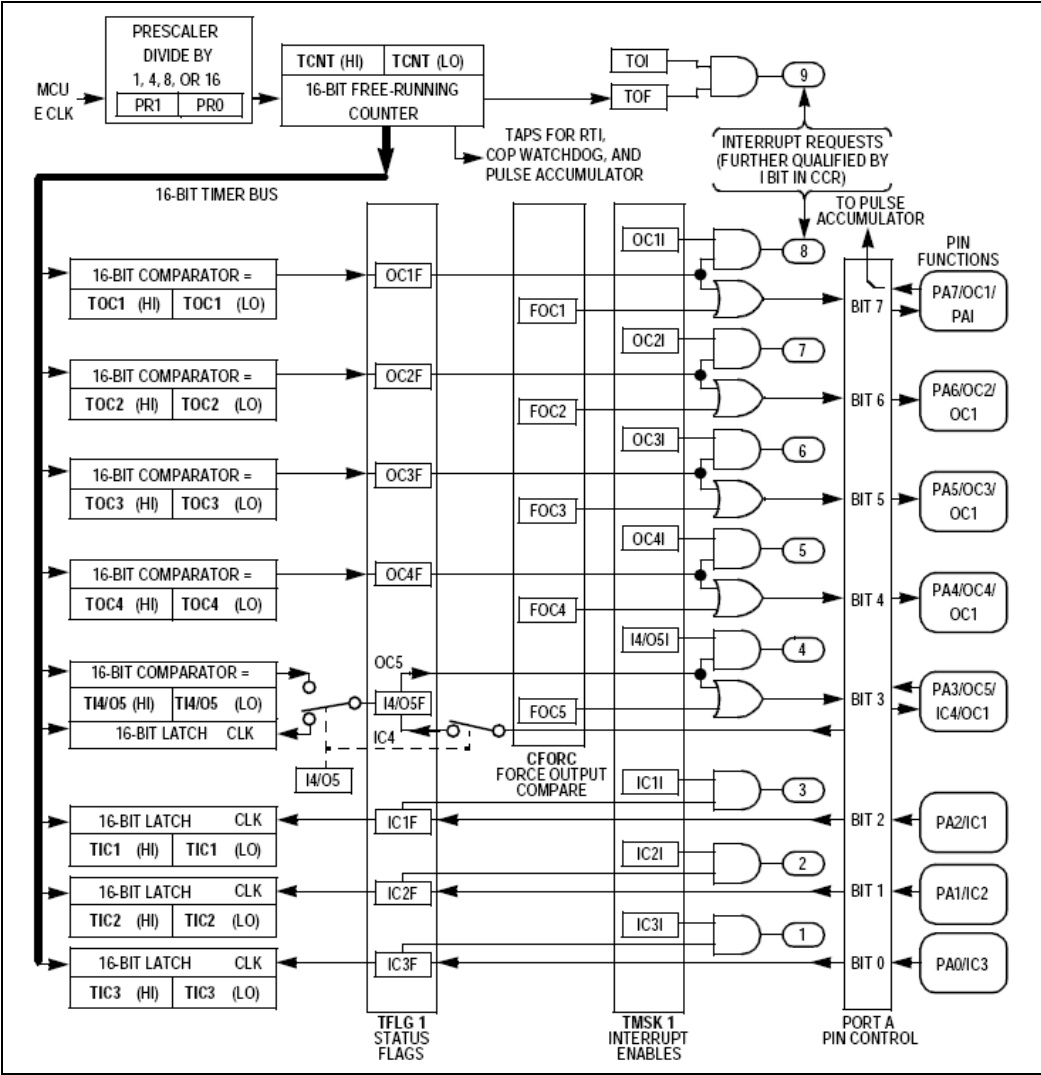
a. Place \$7E (JMP) at location \$00EE.
b. Place IRQ service routine address at locations \$00EF and \$00F0

```
$00EE 7E 80 00 JMP IRQ SERVICE
```

**Table des vecteurs d'interruption du 68HC11E9:
Extrait de la documentation de Buffalo**

SCHEMA FONCTIONNEL

Schéma simplifié de la minuterie.
Extrait de la documentation technique du 68HC11E9



CARTOGRAPHIE DE LA MEMOIRE

