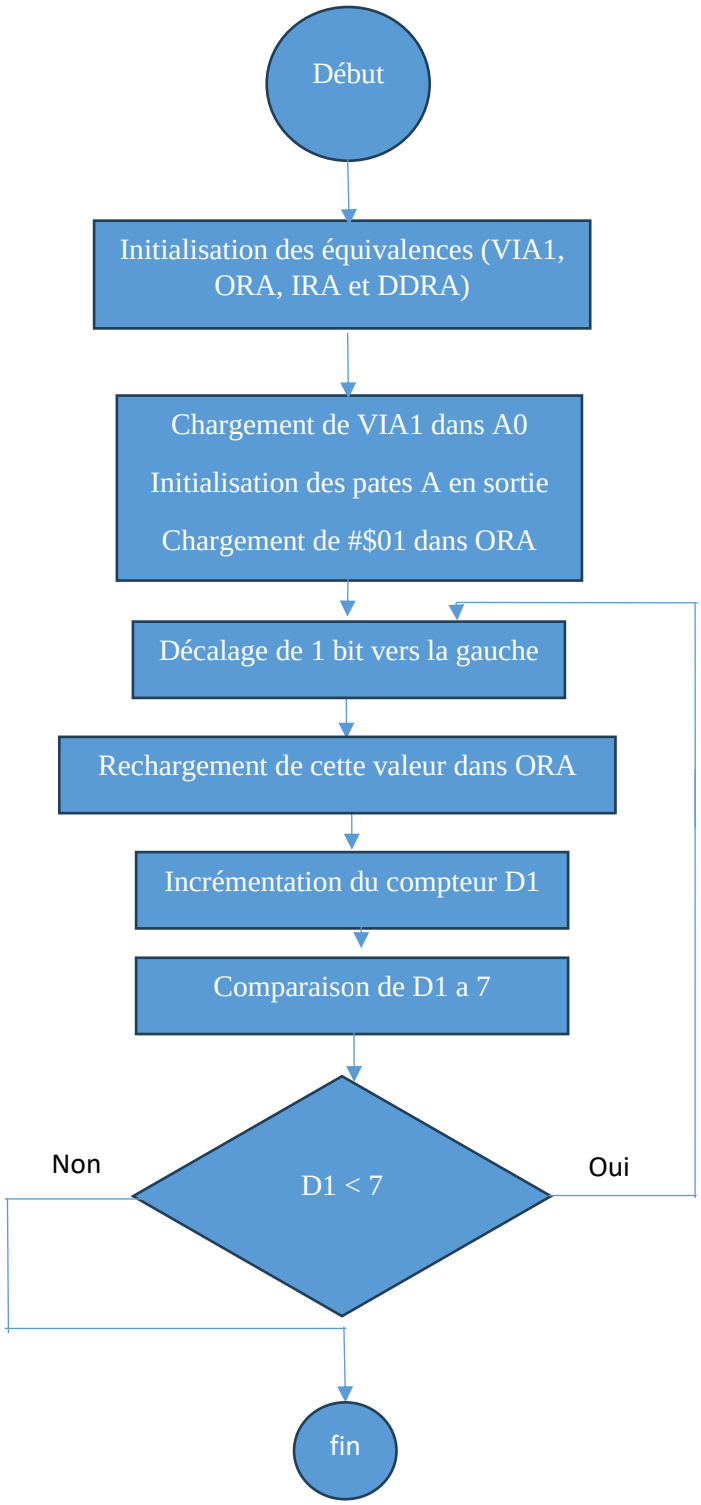
DACCACHE Christopher
ERRARD Mathéo

II0601
23/03/24

Compte rendu TP_2

Exercice 1 :

Organigramme :



Programme question 1.1 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 1.1
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 1.1
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                 *initialisaton des equivalences
ora equ $2                      *output port A
ira equ $2                      *input port A
ddra equ $6                     *status des ports A0 a A7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0             *chargement de via1 dans l'adresse de
registre A0
        move.b #$FF,ddra(A0)    *initialisation des port A en sortie (1111
1111)
        move.b  #$01, D2        *on met 01 en hex dans D2
        move.b D2, ORA(A0)      *on affiche sur la pate A0 la valeur 1 (led
allumé)
        clr D1                  *on effache le contenu de D1

boucle  rol #1, D2              *on effectue une rotation de 1 bit a gauche
pour passer de (0000 0001 à 0000 0010)
        move.b D2, ORA(A0)      *maintenant la pate A1 contient la valeur 1
(led alumé)
        add #1,D1               *on ajoute 1 a D1
        cmp #7,D1               *on compare D1 a 7
        bgt stop                *si D1 > 7 on arrete le programme (on aura
allumé les led 0 a 7 une par une)
        blt boucle              *si D1<7 on remote a l'etiquette boucle et
on allume la led suivante

stop    move #9,D0              *chargement de 9 dans D0
        trap #15                *arret de la simulation


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

Programme question 1.2:

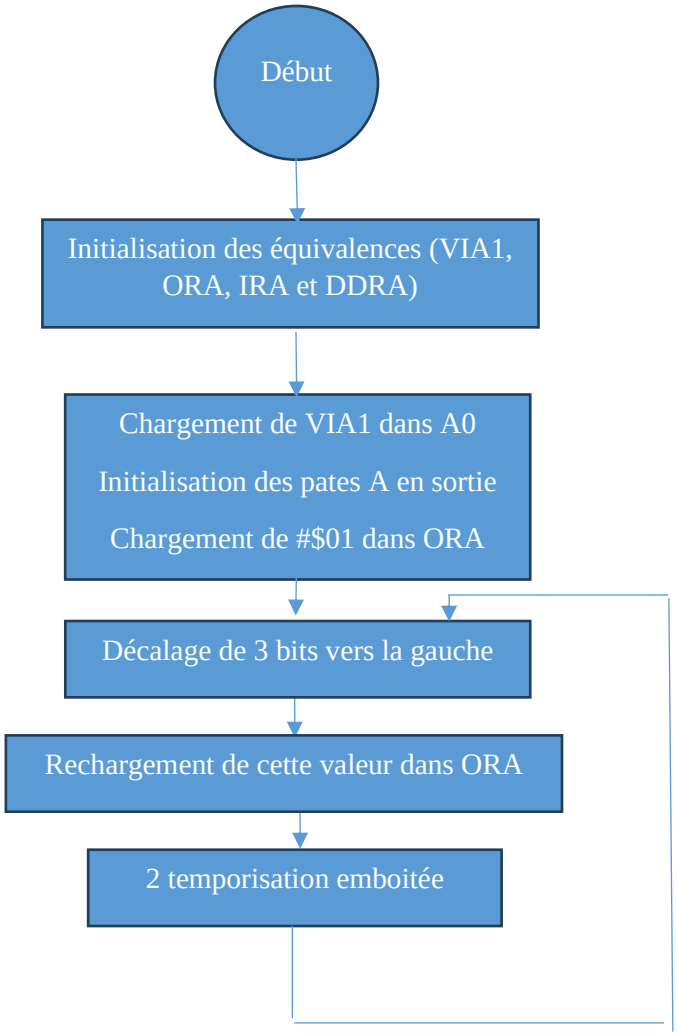
```
*-----------------------------------------------------------
* Title      :  TP3-EXO 1.2
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 1.2
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                 *initialisaton des equivalences
ora equ $2                      *output port A
ira equ $2                      *input port A
ddra equ $6                     *status des ports A0 a A7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0             *chargement de via1 dans l'adresse de
registre A0
        move.b #$FF,ddra(A0)    *initialisation des port A en sortie (1111
1111)
        move.b  #$01, D2        *on met 01 en hex dans D2
        move.b D2, ORA(A0)      *on affiche sur la pate A0 la valeur 1 (led
allumé)
        clr D1                  *on effache le contenu de D1

boucle  rol #1, D2              *on effectue une rotation de 1 bit a gauche
pour passer de (0000 0001 à 0000 0010)
        bsr affiche             *on execute le code de la sub routine
affiche
        bra boucle              *on reexecute le code de l'etiquette boucle


affiche move.b D2, ORA(A0)      *on charge D2 dans ORA pour allume la led
de la pate A correspondante au bit egale a 1

Tempo   move.l #$EE, D3         *on met #$EE (1110 1110 = 238) dans D3

cycle   dbf D3,cycle            *on effectue un cycle qui va se répeter 239
fois
        rts                     *apres 239 iteration on remonte a la
position apres l'execution du coe de l'etiquette affiche
                                *donc on remonte a la ligne de code: bra
boucle (ligne 27)

stop    move #9,D0              *chargement de 9 dans D0
        trap #15                *arret de la simulation


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

Programme question 1.3 et 1.4 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 1.3 et 1.4
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 1.3 et 1.4
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                 *initialisaton des equivalences
ora equ $2                      *output port A
ira equ $2                      *input port A
ddra equ $6                     *status des ports A0 a A7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0             *chargement de via1 dans l'adresse de
registre A0
        move.b #$FF,ddra(A0)    *initialisation des port A en sortie (1111
1111)
        move.b  #$01, D2        *on met 01 en hex dans D2
        move.b D2, ORA(A0)      *on affiche sur la pate A0 la valeur 1 (led
allumé)
        clr D1                  *on effache le contenu de D1

decale  rol.b #1, D2            *on effectue une rotation de 1 bit a gauche
pour passer de (0000 0001 à 0000 0010)
        bsr affiche             *on execute le code de la sub routine
affiche
        bra decale              *on reexecute le code de l'etiquette decale


affiche move.b D2, ORA(A0)      *on charge D2 dans ORA pour allume la led
de la pate A correspondante au bit egale a 1

Tempo   move.l #$2FF, D3        *on met #$2FF(0010 1111 1111 = 767) dans D3

cycle1  move.l D3,D4            *on met le contenue de D3 dans D4
cycle2  dbf D4, cycle2          *cette boucle de tempo va itérer 295296
fois (768+767+766+765+...+0)
        dbf D3, cycle1          *cette boucle cree une temporiation
d'environ 0.5s
        rts                     *on retourn a bra boucle (ligne 27)



stop    move #9,D0
        trap #15


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

Exercice 2 :

Organigramme :



Programme question 2.1 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 2.1
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 2.1
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                 *initialisaton des equivalences
ora equ $2                      *output port A
ira equ $2                      *input port A
ddra equ $6                     *status des ports A0 a A7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0             *chargement de via1 dans l'adresse de
registre A0
        move.b #$FF,ddra(A0)    *initialisation des port A en sortie (1111
1111)
        move.b  #$01, D2        *on met 01 en hex dans D2
        move.b D2, ORA(A0)      *on affiche sur la pate A0 la valeur 1 (led
allumé)
        clr D1                  *on effache le contenu de D1

decale  rol.b #3, D2            *on effectue une rotation de 3 bit a gauche
rol #3 pour la suite (1,4,7,2,5,0,3,6,1,...)
        bsr affiche             *on execute le code de la sub routine
affiche
        bra decale              *on reexecute le code de l'etiquette decale


affiche move.b D2, ORA(A0)      *on charge D2 dans ORA pour allume la led
de la pate A correspondante au bit egale a 1

Tempo   move.l #$2FF, D3        *on met #$2FF(0010 1111 1111 = 767) dans D3

cycle1  move.l D3,D4            *on met le contenue de D3 dans D4
cycle2  dbf D4, cycle2          *cette boucle de tempo va itérer 295296
fois (768+767+766+765+...+0)
        dbf D3, cycle1          *cette boucle cree une temporiation
d'environ 0.5s
        rts                     *on retourn a bra boucle (ligne 27)



stop    move #9,D0
        trap #15


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

Programme question 2.2 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 2.2
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 2.2
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                 *initialisaton des equivalences
orb equ $0                      *output port B
irb equ $0                      *input port B
ora equ $2                      *output port A
ira equ $2                      *input port A
ddra equ $6                     *status des ports A0 a A7 (entree ou
sortie)
ddrb equ $4                     *status des ports B0 a B7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0             *chargement de via1 dans l'adresse de
registre A0
        move.b #$FF,ddra(A0)    *on met les pates de A en mode Lecture
(0F0443 pour l'affichage dans hardware)
        move.b #$FF,ddrb(A0)    *on met les ports de B en mode Lecture
(0F0441 pour les bouton dans hardwar)
        clr D1                  *on efface le contenu de D1
        clr D2                  *on efface le contenu de D2
        clr D6                  *on efface le contenu de D6
        move.b  #$01, D2        *on emet #$01 dans D2
        move.b D2, ora(A0)      *on charge D2 dans ORA
        move.b #$00,irb(A0)     *on charge #$00 dans IRB
        add #$100,D6            *on ajout 100 a D6 pour que la vitesse ne
soit pas tres rapide au tout debut


decale  rol.b #3, D2        *rol #3 au lieu de 1 pour la suite
(1,4,7,2,5,0,3,6,1,...)
        bsr lecture         *on execute le sous programme lecture
        bsr affiche         *on execute le sous programme affiche
        bra decale          *on relance le code d'etiquette decale


lecture move.b irb(A0),D6   *on lit la valeur entré par l'utilisateur sur
les port B
        rts                 *on retourne a bsr affiche (ligne 36)
affiche move.b D2, ORA(A0)  *on charge D2 dans ORA pour allume la led de la
pate A correspondante au bit egale a 1



Tempo   move.l D6, D3       *on met D6 dans D3

cycle1  move.l D3,D4        *on met D3 dans D4
cycle2  dbf D4, cycle2      *on commence l'iteration par rapport au nombre
de bit et position de bits mis a 1
```

```
        dbf D3, cycle1      *exemple si on active les boutons 7,6,5 et 4
cela donne 1 1111 0000 en binaire donc une iteration de 8388608 fois
        rts                 *on retourne a bra decale (ligne 37)



stop    move #9,D0
        trap #15


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

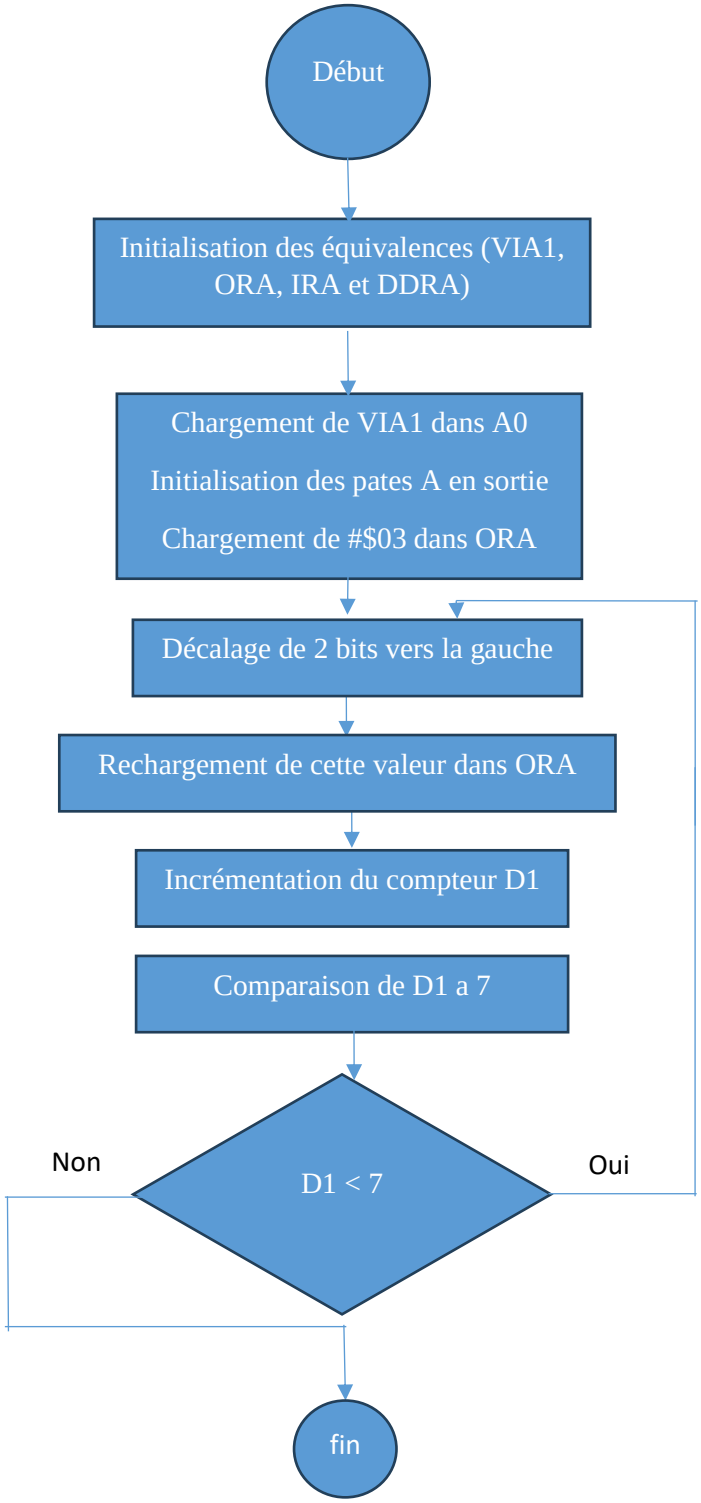
Programme question 2.3 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 2.3
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 2.3
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                 *initialisaton des equivalences
orb equ $0                      *output port B
irb equ $0                      *input port B
ora equ $2                      *output port A
ira equ $2                      *input port A
ddra equ $6                     *status des ports A0 a A7 (entree ou
sortie)
ddrb equ $4                     *status des ports B0 a B7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0             *chargement de via1 dans l'adresse de
registre A0
        move.b #$FF,ddra(A0)    *on met les pates de A en mode Lecture
(0F0443 pour l'affichage dans hardware)
        move.b #$FF,ddrb(A0)    *on met les ports de B en mode Lecture
(0F0441 pour les bouton dans hardwar)
        clr D1                  *on efface le contenu de D1
        clr D2                  *on efface le contenu de D2
        clr D6                  *on efface le contenu de D6
        move.b  #$01, D2        *on emet #$01 dans D2
        move.b D2, ora(A0)      *on charge D2 dans ORA
        move.b #$00,irb(A0)     *on charge #$00 dans IRB
        add #$100,D6            *on ajout 100 a D6 pour que la vitesse ne
soit pas tres rapide au tout debut


decale  cmp #$80,D7             *on compare le bit numero 7 au contenu de
D7
        beq droite              *si c'est egale on execute le code de
l'etiquette droite
        rol.b #1,D2             *sinon on fait un decalage de bit vers la
gauche
suite   bsr lecture             *on lance le sous programme lecture
        bsr affiche             *on lance le sous programme affiche
        bra decale              *on relance le code de l'etiquette decale

droite  ror.b #1,D2
        bra suite


lecture move.b irb(A0),D5       *on met irb dans D5
        move.b D5,D6            *on met le contenu de D5 dans D6
        and #$F7F,D6            *on fait une operation logique and entre
#$F7F et D6 et le resultat sera sauvegarder dans D6
        move.b D5,D7            *on met le contenu de D5 dans D7
```

```
        and #$80,D7             *on fait une operation logique and entre
#$80 et D7 et le resultat sera sauvegarder dans D7
                                *apres ces operations logique, si on met le
bit 7 des ports B a 1 le sens d'allumage des led se fera de gauche a droite
                                *et si on change les bits 6 a 0 des ports B
la vitesse change
        rts                     *on retourne a bsr affiche (ligne 38)
affiche move.b D2, ORA(A0)      *on charge D2 dans ORA pour allume la led
de la pate A correspondante au bit egale a 1



Tempo   move.l D6, D3           *on met D6 dans D3

cycle1  move.l D3,D4            *on met D3 dans D4
cycle2  dbf D4, cycle2          *on commence l'iteration par rapport au
nombre de bit et position de bits mis a 1
        dbf D3, cycle1          *exemple si on active les boutons 7,6,5 et
4 cela donne 1 1111 0000 en binaire donc une iteration de 8388608 fois
        rts                     *on retourne a bra decale (ligne 39)

stop    move #9,D0
        trap #15


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

Exercice 3 :

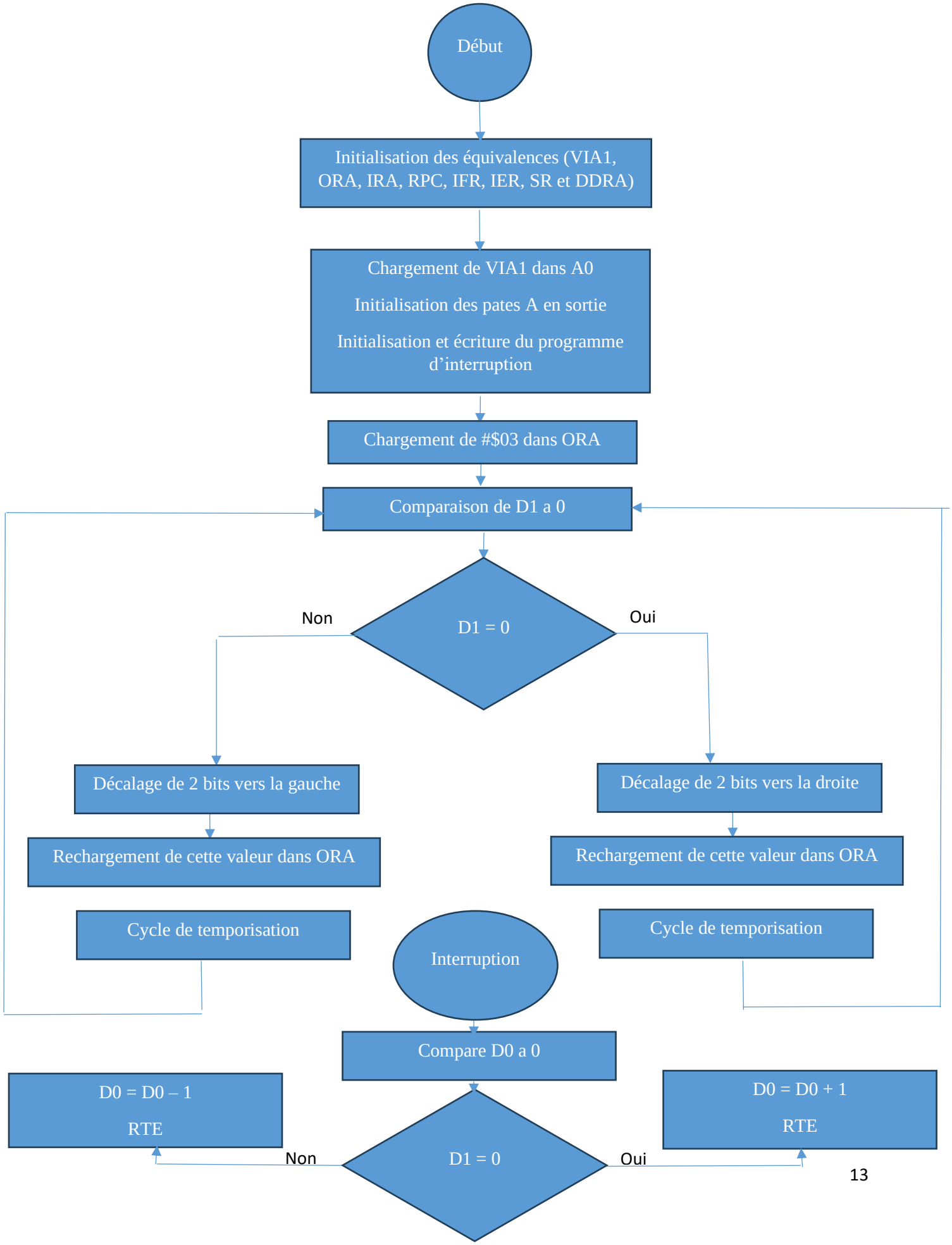
Organigramme :



Programme question 3.1 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 3.1
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 3.1
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                 *initialisaton des equivalences
ora equ $2                      *output port A
ira equ $2                      *input port A
ddra equ $6                     *status des ports A0 a A7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0             *chargement de via1 dans l'adresse de
registre A0
        move.b #$FF,ddra(A0)    *on met les pates de A en mode Lecture
(0F0443 pour l'affichage dans hardware)
        move.b  #$03, D2        *on emet #$03 dans D2
        move.b D2, ORA(A0)      *on charge D2 dans ORA
        clr D1                  *on efface le contenu de D1

boucle  rol.b #2, D2            *on effectue un décalage de 2 bits vers la
gauche
        move.b D2, ORA(A0)      *on recharge D2 dans ORA
        add #1,D1               *on incremente D1
        cmp #7,D1               *on compare D1 a 7
        bgt stop                *si D1>7 on arrete la simulation
        blt boucle              *si D1<7 on relance la boucle

stop    move #9,D0
        trap #15


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

Organigramme question 3.2 :

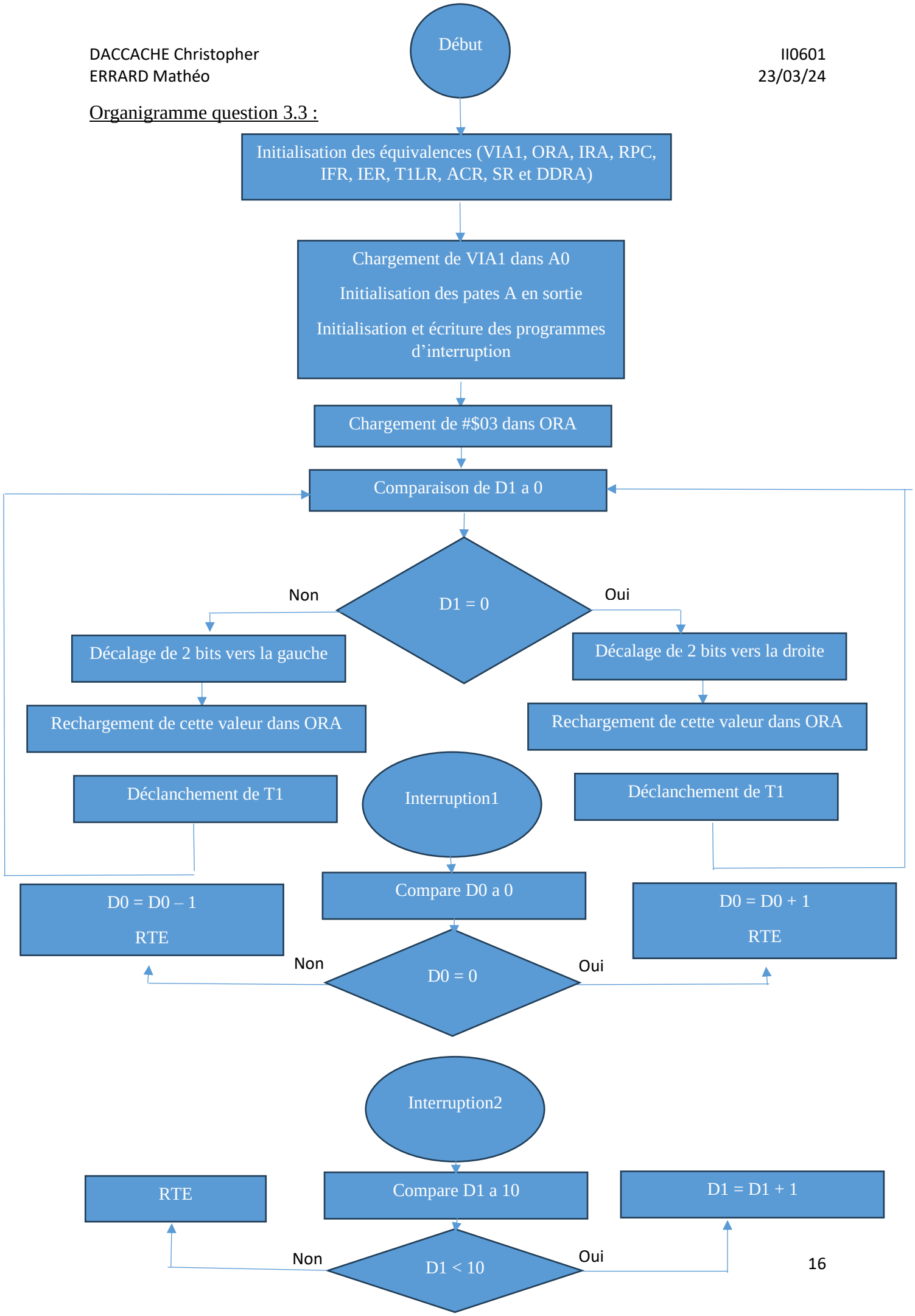


Programme question 3.2 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 3.2
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 3.2
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                     *initialisaton des equivalences
ora equ $2                          *output port A
ira equ $2                          *input port A
rpc equ $18                         *registre de controle des peripheriques
ifr equ $1A                         *registre d'interruption valide
ier equ $1C                         *registre d'indicateur d'interruption
sr equ $14                          *registre d'etat
ddra equ $6                         *status des ports A0 a A7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0                 *chargement de via1 dans l'adresse de
registre A0
        move.l #inter,$70           *on choisit l'adresse d'interuption
(niveau 4)
        move.b #$1,rpc(A0)          *on met a 1 le bit numero 0 de rpc qui
correspond a un front montant sur CA1
        move.b #%10000010,ier(A0)   *on met ier a 1000 0010, pour authorise
les interruption sur CA1
        move.w #$2300,sr            *on met les 8 bits de gauche de SR a
0010 0011, pour changer le masque d'interruption a 3 et authorisé mode
superviseur
        move.b #$FF,ddra(A0)        *on met les pates de A en mode Lecture
(0F0443 pour l'affichage dans hardware)
        move.b  #$03, D2            *on emet #$03 dans D2
        move.b D2, ORA(A0)          *on charge D2 dans ORA
        clr D1                      *on efface le contenu de D1


boucle  cmp #1,D1                   *on compare D1 a 1
        beq droite                  *si D1=1 on passe a programme de
l'etiquette droite
        rol.b #2, D2                *sinon on on fait un decalage de 2 bit
vers la gauche
gauche  bsr affiche                 *on passe ou sous programme affiche
        bra boucle                  *on relance le programme d'etiquette
boucle

droite  ror.b #2,D2                 *on fait un deplacement de 2 bit vers
la droite
        bsr affiche                 *on passe ou sous programme affiche
        bra boucle                  *on relance le programme d'etiquette
boucle
affiche move.b D2, ORA(A0)          *on recharge D2 dans ORA

Tempo   move.l #$2FF, D3            *on met#$2FF dans D3
```

```
cycle1  move.l D3,D4                *on met D3 dans D4
cycle2  dbf D4, cycle2              *on commence le cycle de temporisation
        dbf D3, cycle1
        rts                         *on retourne au programme bra boucle
(ligne 38 ou ligne 42) cela depand de D1
*----(programme d'interruption)-------------------------------
inter   cmp #0,D1                   *on compare D1 a 0 losqu'on a un front
montant sur CA1 (declanchement de l'interruption 4 sur le hardware)
        beq yes                     *si D1 = 0 on passe a l'etiquette yes
        bne no                      *sinon on passe a l'etiquette no
yes     add #1,D1                   *si D1 = 0 on ajoute 1 a D1
        bra out                     *on passe a l'etiquette out
no      sub #1,D1                   *si D1 n'est pas egale a 0 on soustrait
1 a D1 pour qu'il redeviet egale a 0
        bra out                     *on passe a l'etiquette out
out     bset #1,ifr(A0)             *apres avoir modifier la valeur de D1
on change le bit numeo 1 de IFR qui correspond a CA1 pour arreter
l'interruption
        rte                         *on sort de l'interruption et on
reprend le code d'ou on avait arreter


stop    move #9,D0
        trap #15


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```

Organigramme question 3.3 :



Programme question 3.3 :

```
*-----------------------------------------------------------
* Title      :  TP3-EXO 3.3
* Written by :  ERRARD-DACCACHE
* Date       :  20/03/2024
* Description:  EXO 3.3
*-----------------------------------------------------------
    *include c:\easy68K\equvia.X68
via1 equ $f0441                     *initialisaton des equivalences
ora equ $2                          *output port A
ira equ $2                          *input port A
rpc equ $18                         *registre de controle des peripheriques
ifr equ $1A                         *registre d'interruption valide
ier equ $1C                         *registre d'indicateur d'interruption
sr equ $14                          *registre d'etat
T1LR equ $8                         *registre de temporisation
acr equ $16                         *registre de controle auxiliaire
ddra equ $6                         *status des ports A0 a A7 (entree ou
sortie)

    ORG    $1000

START:                  ; first instruction of program

* Put program code here

        lea via1,A0                 *chargement de via1 dans l'adresse de
registre A0
        move.b #$1,rpc(A0)          *on met a 1 le bit numero 0 de rpc qui
correspond a un front montant sur CA1
        move.b #%10000010,ier(A0)   *on met ier a 1000 0010, pour authorise
les interruptions sur CA1
        move.l #inter1,$70          *on choisit l'adresse d'interuption
(niveau 4)
        move.w #$2300,sr            *on met les 8 bits de gauche de SR a
0010 0011, pour changer le masque d'interruption a 3 et authorisé mode
superviseur

        move.l #80000,D0            *on met 80000 dans D0 pour une
precision de 100ms
        rol.w #8,D0                 *on fait un deplacement de 8 bit vers
la gauche
        movep.w D0,T1LR(A0)         *on charge D0 dans T1LR
        move.b #$80,acr(A0)         *on met #$40 (0100 0000) dans ACR pour
metre T1 en mode libre
        move.b #%11000000,ier(A0)   *on met ier a 1000 0010, pour authorise
les interruptions sur T1
        move.l #inter2,$74          *on choisit l'adresse d'interuption
(niveau 4)
        move.w #$2300,sr            *on met les 8 bits de gauche de SR a
0010 0011, pour changer le masque d'interruption a 3 et authorisé mode
superviseur


        move.b #$FF,ddra(A0)        *on met les pates de A en mode Lecture
(0F0443 pour l'affichage dans hardware)
        move.b  #$03, D2            *on emet #$03 dans D2
        move.b D2, ORA(A0)          *on charge D2 dans ORA
        clr D1                      *on efface le contenu de D1
        clr D5
```

```
boucle  cmp #1,D1                   *on compare D1 a 1
        beq droite                  *si D1=1 on passe a programme de
l'etiquette droite
        rol.b #2, D2                *sinon on on fait un decalage de 2 bit
vers la gauche
gauche  bsr affiche                 *on passe ou sous programme affiche
        bra boucle                  *on relance le programme d'etiquette
boucle

droite  ror.b #2,D2                 *on fait un deplacement de 2 bit vers
la droite
        bsr affiche                 *on passe ou sous programme affiche
        bra boucle                  *on relance le programme d'etiquette
boucle
affiche move.b D2, ORA(A0)          *on recharge D2 dans ORA
tempo   cmp #10,D5                  *pour que la tempo se declanche il faut
activé le automatic disable 5
        blt tempo                   *verification que l'interruption timer
a bien fonctionner
        clr D5                      *on efface le contenu de D5 pour
repeter l'interruption timer
        rts                         *on retourne au programme bra boucle
(ligne 38 ou ligne 42) cela depand de D1
*----(programme d'interruption)-----------------------------
inter1  cmp #0,D1                   *on compare D1 a 0 losqu'on a un front
montant sur CA1 (declanchement de l'interruption 4 sur le hardware)
        beq yes                     *si D1 = 0 on passe a l'etiquette yes
        bne no                      *sinon on passe a l'etiquette no
yes     add #1,D1                   *si D1 = 0 on ajoute 1 a D1
        bra out                     *on passe a l'etiquette out
no      sub #1,D1                   *si D1 n'est pas egale a 0 on soustrait
1 a D1 pour qu'il redeviet egale a 0
        bra out                     *on passe a l'etiquette out
out     bset #1,ifr(A0)             *apres avoir modifier la valeur de D1
on change le bit numeo 1 de IFR qui correspond a CA1 pour arreter
l'interruption
        rte                         *on sort de l'interruption et on
reprend le code d'ou on avait arreter

inter2  add #1, D5                  *on ajoute 1 a D5
        cmp #10,D5                  *on compare D5 a 10 (on a besoin de 10
interruptions de 100ms pour avoir une tempo de 1s)
        blt inter2                  *si D5 plus petit que 10 on remonte a
inter2
        bset #6,ifr(A0)             *apres avoir fini la tempo on sort de
l'interruption on mettant le bit 6 a 1 de IFR
        rte                         *on remonte au programme


stop    move #9,D0
        trap #15


    SIMHALT             ; halt simulator

* Put variables and constants here

    END    START        ; last line of source
```