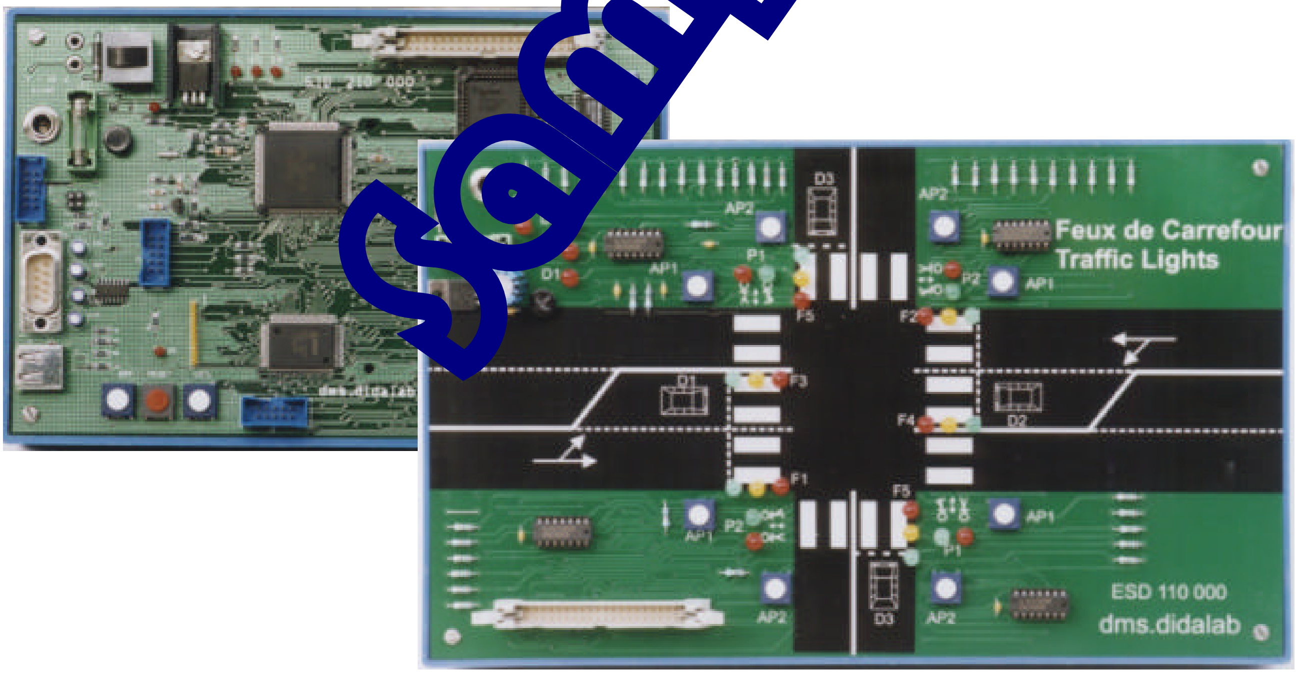
# EXPERIMENTS

# MANUAL

## EID210 System
## +
## "Traffic Lights" Module



Didalab Matelco Stci

Z.A. de la clé St Pierre
5, rue du groupe Manoukian
78990 Elancourt
tél. : (33) 01 30 66 08 88
fax : (33) 01 30 66 72 20

Issued on 30/08/05

Reference : EID212040

# SUMMARY

# EXP N°1: SINGLE CYCLE WITHOUT PEDESTRIAN CALL & CAR DETECTION

## 1.1 Topic

| | |
|---|---|
| ***Purpose :*** | Being capable of activating the different lights of the " Traffic Lights " module.<br>Being capable of representing by a “Grafcet” the specified sequential linking.<br>Being capable of programming in Assembler language a sequential linking represented by a “Grafcet”.<br>Being capable of carrying out time delays by using “Software-type” waiting loop.. |
| ***Specifications :*** | The following cycle has to be carried out :<br>- main lanes (lights [illegible] F2 [illegible] 2 ) at green during 12Sec.then,<br>- passing at yellow [illegible] during 3Sec.then,<br>- secondary lanes (lights [illegible] P1 ) at green during 3Sec. then,<br>- passing at yellow [illegible] during 3Sec. Then,<br>- loop again.<br>Waiting is carried out by programmed loop. |

## Necessary Equipment :

Micro Computer PC-type, with Windows 95 ® or later,

16/32 bits, 68332 micro-controller mother Board , Ref. : EID 100 000

USB link cable or if not available, RS232 cable, Ref. : EGD 000 003

AC/DC 8V 1 A Power Supply, Ref. : EGD000001,

“Traffic Lights” Board, ref. : EID 002 000,

## Allocated time duration : 4 hours

# 1.2 Elements of solution

## *1.2.1 Outputs activation*

**Lights assignment chart to the ports bits :**

| | | | | | |
|---|---|---|---|---|---|
| Light **F3**<br>Bit Port A<br>Bit HSRR1 | Green Yellow Red<br>▨ 7 6<br>15 14 13 12 | Light **F2**<br>Bit Port A<br>Bit HSRR1 | Green Yellow Red<br>5 4 3<br>11 10 9 8 7 6 | Light **F1**<br>Bit Port A<br>Bit HSRR1 | Green Yellow Red<br>2 1 0<br>5 4 3 2 1 0 |
| Light **F5**<br>Bit Port B<br>Bit HSRR0 | Green Yellow Red<br>6 5 4<br>13 12 11 10 9 8 | Light **F4**<br>Bit Port B<br>Bit HSRR0 | Green Yellow Red<br>3 2 1<br>7 6 5 4 3 2 | Light **F3**<br>Bit Port B<br>Bit HSRR0 | Green Yellow Red<br>0 ▨<br>1 0 |
| Light **P2**<br>Bit Port C | Green Red<br>2 1 | Light **P1**<br>Bit Port C | Green Red<br>0 ▨ | Light **P1**<br>Bit Port B<br>Bit HSRR0 | Green Red<br>▨ 7<br>15 14 |

Lights on A & B are switched on by binary couple « 0 1 » into the corresponding location of HSRR register. One light is off is the value « 1 0 » is given to the same location.
Lights on port C are switched on by the value 1 into the data register linked to port C ( Label specified Port_C).

**Example :**
We wish to allow cars crossing only through lanes F1 & F2 and pedestrians crossing through P2, as :

- Lights F1 , F2 & P2 at green
- Lights F3, F4, F5 & P1 at red

Then, we must write the following binary combinations :
-> For register HSRR1 enabling the specify the states of port A

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Light **F3**<br>Bit Port A<br>Bit HSRR1 | Green Yellow<br>▨ 0<br>1 0 | Red<br>1<br>0 1 | Light **F2**<br>Bit Port A<br>Bit HSRR1 | Green<br>1<br>0 1 | Yellow Red<br>0 0<br>1 0 1 0 | Light **F1**<br>Bit Port A<br>Bit HSRR1 | Green<br>1<br>0 1 | Yellow Red<br>0 0<br>1 0 1 0 |

HSRR1 = 1001 0110 1001 1010 in binary code = $969A (Hexadecimal encoding)

-> For register HSRR1 enabling to specify the states of port B

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Light **F5**<br>Bit Port B<br>Bit HSRR0 | Green Yellow<br>0 0<br>1 0 1 0 | Red<br>1<br>0 1 | Light **F4**<br>Bit Port B<br>Bit HSRR0 | Green Yellow<br>0 0<br>1 0 1 0 | Red<br>1<br>0 1 | Light **F3**<br>Bit Port B<br>Bit HSRR0 | Green Yellow<br>0 ▨<br>1 0 | Red<br>▨ |
| Light **P1**<br>Bit Port B<br>Bit HSRR0 | Green<br>▨ | Red<br>0<br>1 0 | | | | | | |

HSRR0 = 1010 1001 1010 0110 in binary code = $A9A6 (Hexadecimal encoding )

-> For data register of port C (label specified Port_C)

| | | | | | |
|---|---|---|---|---|---|
| Light **P2**<br>Bit Port C | Green<br>0 | Red<br>1 | Light **P1**<br>Bit Port C | Green<br>0 | Red<br>▨ |

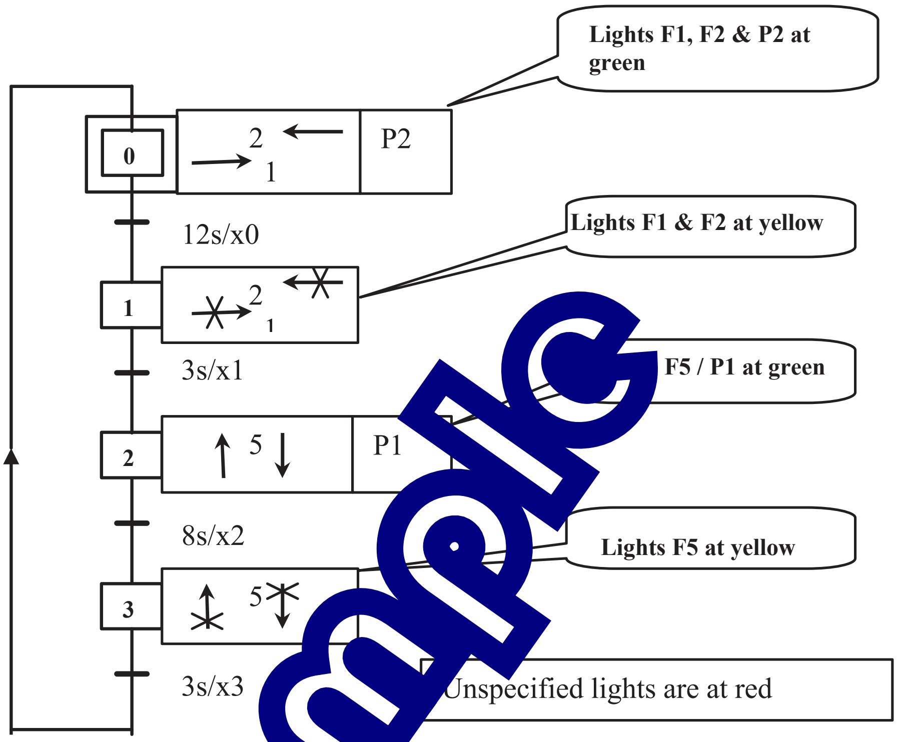
Port_C = xxxxx010xxxxxxxx = $0200

## 1.2.2 *Specifications "Grafcet" display*

For simplifying the display, the following agreements are adopted :

- The lane with the green light is represented by a pointer.
- The lane with the yellow light is represented by a. crossed pointer.
- Unrepresented or unspecified lights are at red.

Which gives the specified topic :



**Lights F1, F2 & P2 at green**

**Lights F1 & F2 at yellow**

**F5 / P1 at green**

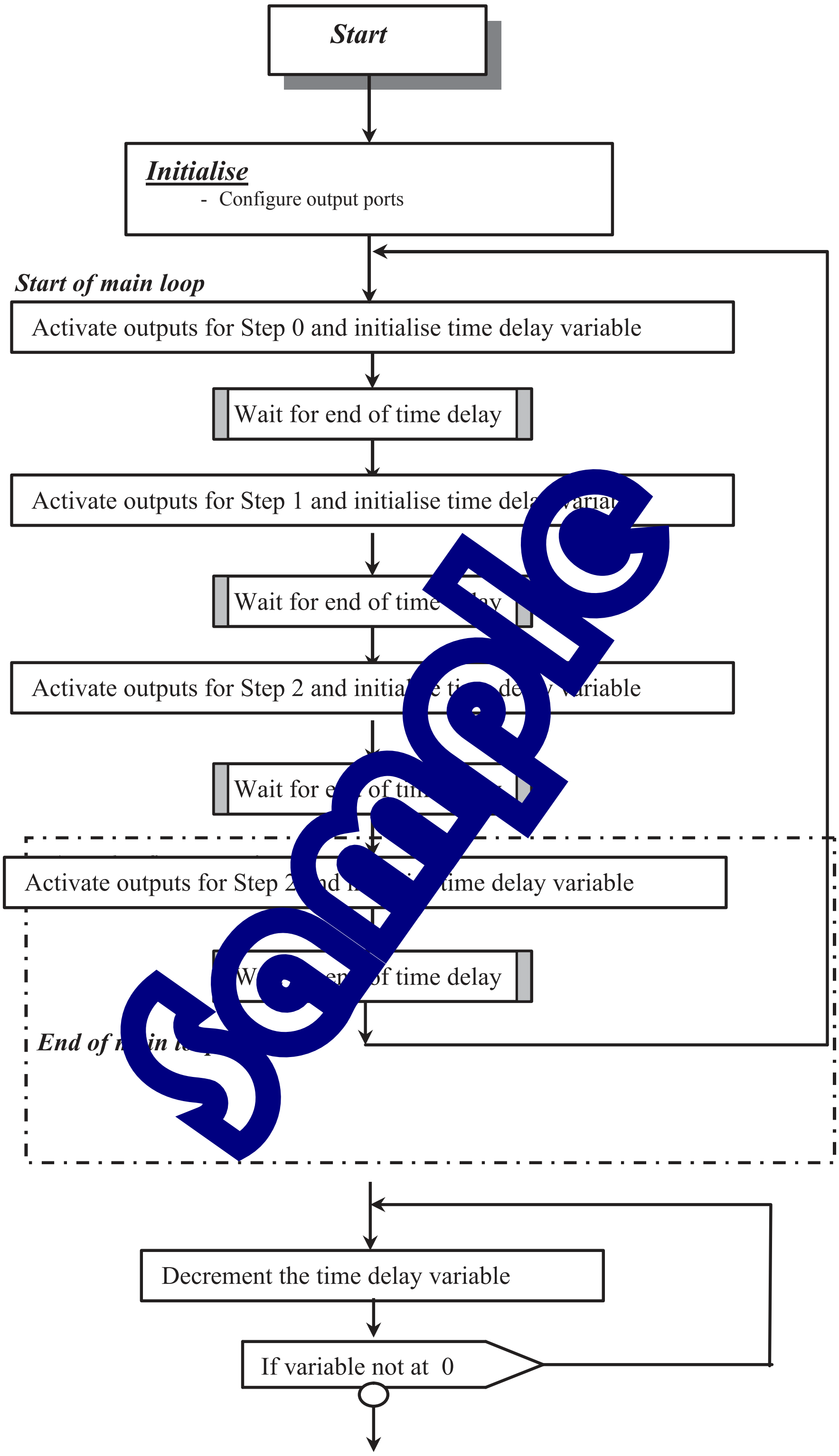
**Lights F5 at yellow**

Unspecified lights are at red

The chart giving in detail the determination of the binary words to be loaded into the different registers, is shown in ANNEX.

| **Step N°** | REGISTERS CONTENTS (In hexadecimal) | | |
|---|---|---|---|
| | HSRR0 | HSRR1 | Port C |
| 0 | 69A6 | 969A | 0400 |
| 1 | 69A6 | 99A6 | 0200 |
| 2 | 9AA6 | 9A69 | 0300 |
| 3 | 66A6 | 9A69 | 0200 |

## 1.2.3 *"Grafcet" programming flowchart*


Start
Initialise
- Configure output ports
Start of main loop
Activate outputs for Step 0 and initialise time delay variable
Wait for end of time delay
Activate outputs for Step 1 and initialise time dela variab
Wait for end of time lay
Activate outputs for Step 2 and initia e ti de riable
Wait for e of tim
Activate outputs for Step 2 d time delay variable
W of time delay
End of in
Decrement the time delay variable
If variable not at 0

## 1.2.4  A68xxx Assembler Program

```
          **************************************************************************
          *                         TP EID210 + TRAFFIC LIGHTS                       *
          **************************************************************************
          *     Specifications:                                                    *
          *    *****************                                                   *
          *    - Scheduled  permutations : main lanes – secondary lanes            *
          *    - Pedestrian calls and car detection are not taken into account     *
          *    - Time delays are carried out by programmed waiting loops           *
          *  FILE NAME:  Feu_Carf_1.SRC                                            *
          **************************************************************************
          * File inclusion specifying the different labels
          include 68332.def
*   Definition of constants          *
*********************************
*******************************
*  Start of execute program   *
*******************************
          section                    code
*         INITIALISE
*************************
* Configure port A in "Discrete Input Output"   (DIO) mode-> code $8
DEBUT move.w                 #$8888,CFSR3         * from CHA0 to CHA3 in "DIO" mode
                    move.w            #$8888,CFSR2        * from CHA4 to CHA7 [illegible]ode
                    move.w            $8888,CFSR1         * from CHA8 to CHA[illegible] in "DI[illegible] mode
                    move.w            #$8888,CFSR0        * from CHA12 to C[illegible]15[illegible]ode
* Specify priorities
                    move.w            #$FFFF,CPR1         * All PA[illegible] in hi[illegible]ity
                    move.w            #$FFFF,CPR0         * All PB bit[illegible]high p[illegible]
* All Lights are at red
                    move.w            #$9A69,HSRR1        *  For Lig[illegible] at[illegible]
                    move.w            #$69A6,HSRR0        * For L[illegible]ts on B ([illegible])
                    move.w            #$0700,DIR_Port_C  *  The[illegible]sb[illegible]f po[illegible]C on output
                    move.w            #$0200,Port_C       *  For[illegible]ght[illegible]ort
*          MAIN LOOP
*************************
Deb_BP
*  STEP n°0  Authorisation main lanes (Lights 1 & 2 at green)
                    move.w            #$969A,HSRR[illegible]        *  Fo[illegible]ts on port A (CH0 at 7)
                    move.w            #$69A6,HSR[illegible]        [illegible]r the[illegible]ghts on port B (CH8 at 15)
                    move.w            #$0400,Port_C[illegible]ans 2 at GREEN
*  Waiting loop of about 12 seconds
                    move.l            #$00D[illegible]FF,d[illegible]
ATT1                sub.l             #1,d2
                    bne               ATT[illegible]
* STEP n°1  Lights 1 & 2 passing at y[illegible]w
                    move.w            #[illegible]9A[illegible]HSRR1        * For  the Lights on port A (CH0 at 7)
                    move.w            #$0[illegible]              * For  the Lights on port B (CH8 at 15)
                    move.w            #$0200,P[illegible]_C        *  Pedestrians 2 pass at RED
* Waiting loop of about 3 seconds
                    move.l            #[illegible]FC[illegible]F,d2
ATT2                sub.l             #1,d[illegible]
                    bne               ATT2
* STEP n°2  Lights 5 pass at green
                    move.w            #$9A69,HSRR1        * For  the Lights on port A (CH0 at 7)
                    move.w            #$9AA6,HSRR0        * For  the Lights on port B (CH8 at 15)
                    move.w            #$0300,Port_C       * Pedestrians 2 pass at GREEN
* Waiting loop of about 8 seconds
                    move.l            #$009FCFFF,d2
ATT3                sub.l             #1,d2
                    bne               ATT3
* STEP n°3 Lights 5 pass at yellow
                    move.w            #$9A69,HSRR1        * For  the Lights on port A (CH0 at 7)
                    move.w            #$66A6,HSRR0        * For  the Lights on port B (CH8 at 15)
                    move.w            #$0200,Port_C       * Pedestrians 2 pass at RED
* Waiting loop of about 3 seconds
                    move.l            #$004FCFFF,d2
ATT4                sub.l             #1,d2
                    bne               ATT4
* loop
                    bra               Deb_BP
*         End of main loop and end of main program
                    end                                   *          End of listing
```

# TP 2 : FULL CYCLE WITHOUT ANY PEDESTRIANS CALL PROCESSING OR CAR DETECTION

## 2.1 Topic

| | |
|---|---|
| ***Purpose :*** | **Additional abilities :**<br>Being capable of programming a full pre-specified sequential linking.<br>Being capable of carrying out a time delay loop with the micro-controller built-in timer. |
| ***Specifications :*** | The cycle must be the following :<br>- main lanes (Lights F1, F2 and P2 at green) during 10 Sec. then,<br>- Light F1 passing at yellow during 3 Sec. then,<br>- Main lane n°2 (Light F2) with fork n° 4 (Light F4) during 8 Sec. then,<br>- Lights F2 and F4 passing at yellow during 3 Sec. then,<br>- Secondary lanes (Lanes F5 and P1 at green) during 3 Sec. then,<br>- Passing at yellow during 3 Sec. then,<br>- Main lanes (Lights F1, F2 and P2 at green) during 10 Sec. then,<br>- Light F2 passing at yellow during 3 Sec. then,<br>- Main lane n°1 (Light F1) with fork n°3 (F3) during 8Sec. then,<br>- Lights F1 and F3 passing at yellow during 3 Sec. then,<br>- Secondary lanes (Lanes F5 and P1 at green) during 3 Sec. then,<br>- LOOP<br>W[illegible] will be carried out by programmed loops and micro-controller built-in timer. |

Necessary Equipment :

Micro Computer PC-type, with Windows 95 ® or later,

16/32 bits, 68332 micro-controller mother Board , Ref. : EID 100 000

USB link cable or if not available, RS232 cable, Ref. : EGD 000 003

AC/DC 8V 1 A Power Supply, Ref. : EGD000001,

“Traffic Lights” Board, ref. : EID 002 000,

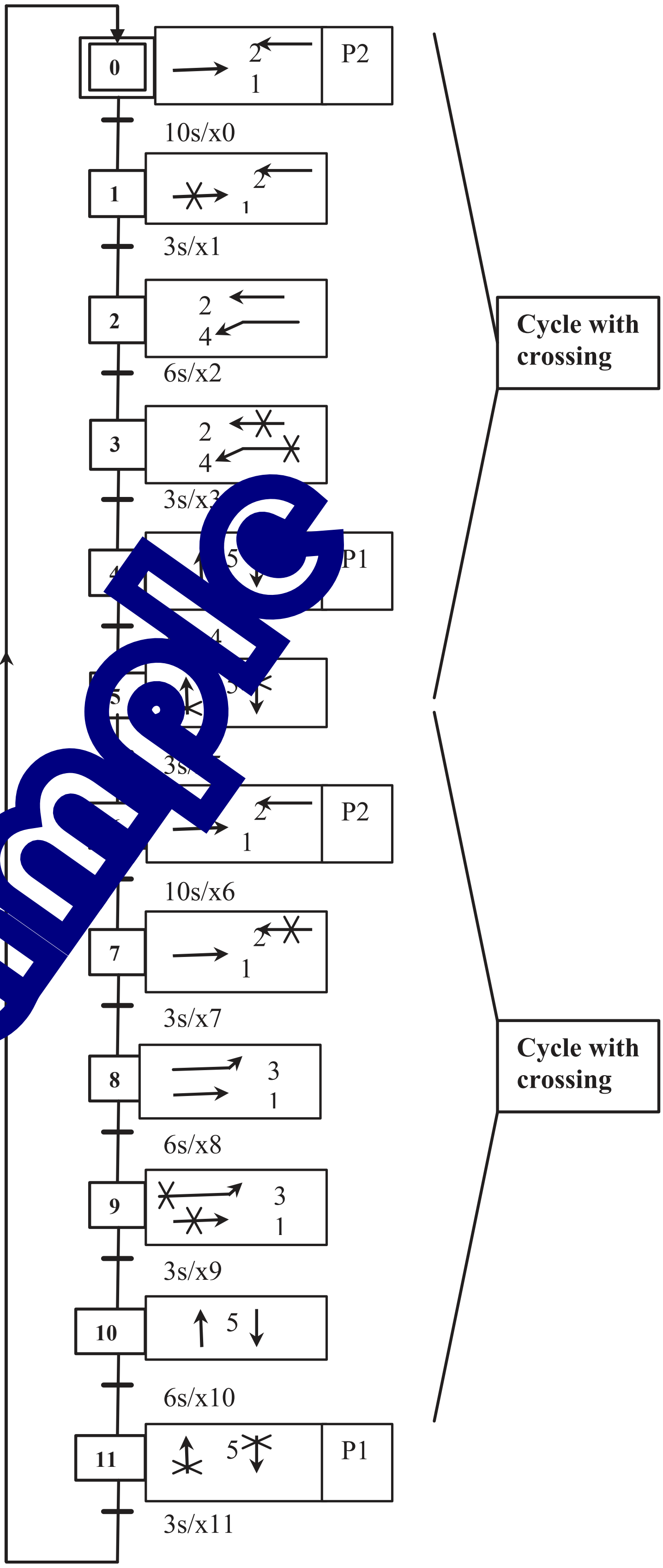
Allocated time duration : 4 hours

# 2.2 Elements of solution

## *2.2.1 Grafcet*

The table specifying the determination of the binary words to be loaded onto the different registers, is given in ANNEX.

| Step N° | REGISTERS CONTENTS (In hexadecimal)<br>HSRR0 | HSRR1 | Port C |
|---|---|---|---|
| 0 | 69A6 | 969A | 0400 |
| 1 | 69A6 | 96A6 | 0200 |
| 2 | 696A | 96A9 | 0200 |
| 3 | 699A | 99A9 | 0200 |
| 4 | 9AA6 | 9A69 | 0300 |
| 5 | 66A6 | 9A69 | 0300 |
| 6 | 69A6 | 969A | 0400 |
| 7 | 69A6 | 999A | 0200 |
| 8 | 69A5 | AA5A | 0200 |
| 9 | 69A6 | 6A66 | 0200 |
| 10 | 9AA6 | 9A69 | 0300 |
| 11 | 66A6 | 9A69 | 0300 |

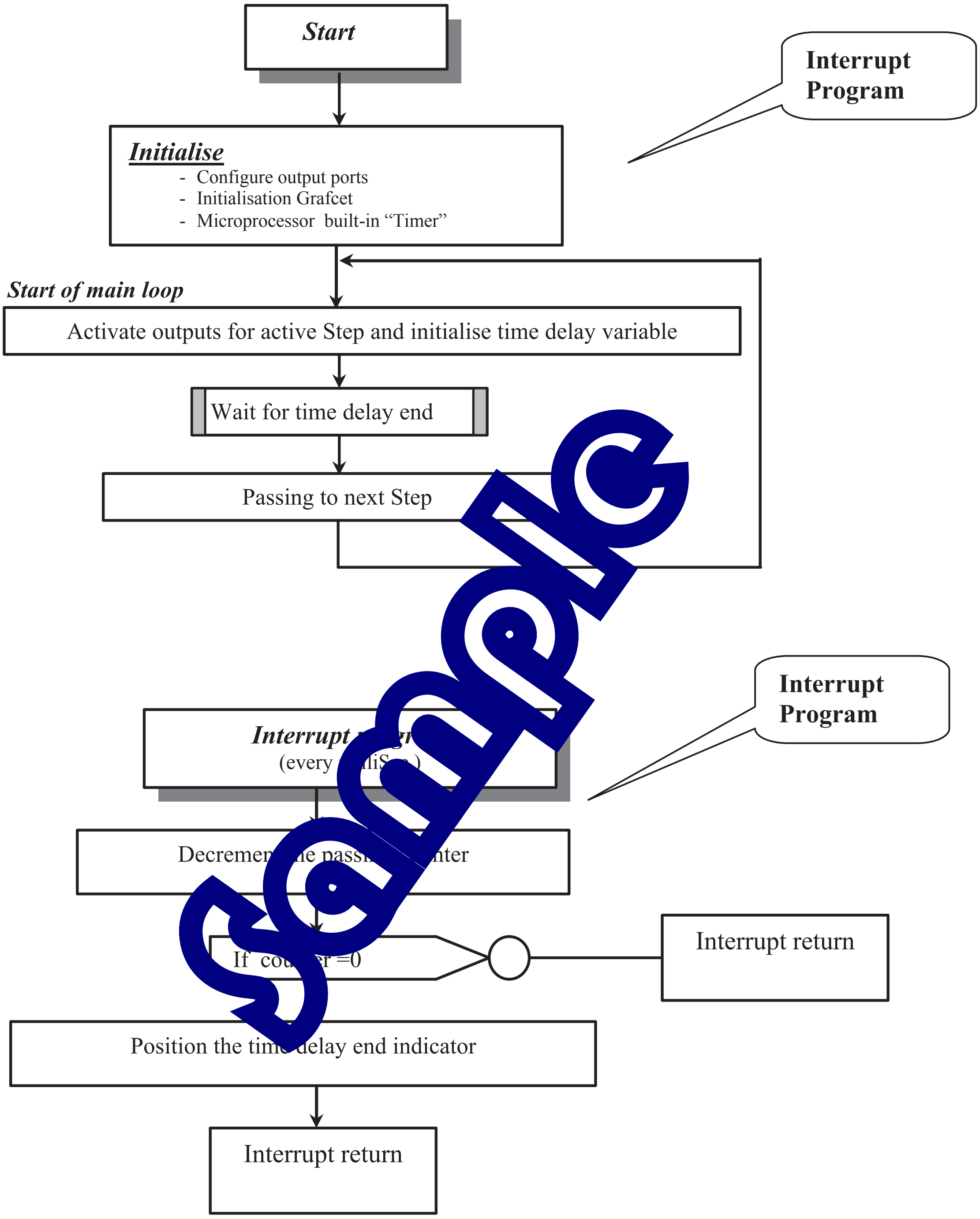


## 2.2.2 A68xxx Assembler Program with ‘programmed’ time delay

```
          ***************************************************************************
          *                         EXPERIMENT EID210 + TRAFIC LIGHTS                *
          ***************************************************************************
          *     Specifications:                                                     *
          *   *****************                                                     *
          *   - Main lanes Scheduled permutations  -  Crossing  n° 4 – secondary lanes   *
          *             main lanes -  Crossing n° 3 – secondary lanes ..etc          *
          *   - Pedestrian calls and car detection are not taken into account       *
          *   - Time delays are carried out by programmed waiting loops             *
          *  FILE NAME:  Feu_Carf_2.SRC                                             *
          *******************                                                       *
          **************************************************************************
          * File inclusion specifying the different labels
          include 68332.def
*   Definition of constants          *
*************************************
*  Start of execute program         *
*************************************
          section                   code
*         INITIALISE
*************************
* Configure port A   in "Discrete Input Output" mode  (DIO)-> code $8
START     move.w                    #$8888,CFSR3        * CHA0 to CHA3 in  "DIO" mode
          move.w                    #$8888,CFSR2        * CHA4 to CHA7 in  "DIO" mo
          move.w                    #$8888,CFSR1        * CHA8 to CHA11 in  "DIO" m   e
          move.w                    #$8888,CFSR0        * CHA0 to CHA3 in  "      " mo
Specify priorities
          move.w                    #$FFFF,CPR1         * All bits of  PA in hig     iority
          move.w                    #$FFFF,CPR0         * All bits of  PB in high pr
* All Lights are at red
          move.w                    #$9A69,HSRR1        *  For Lights on p    A (CH0 a
          move.w                    #$69A6,HSRR0        *  For Lights on p   t B     at    )
          move.w                    #$0700,DIR_Port_C   *  The 3 lsb bits o    or     ou   t
          move.w                    #$0200,Port_C       *  For Lights on p     C
*          MAIN LOOP
****************************
Deb_BP
*  STEP n°0  Authorisation main lanes (Lights 1 and 2 at re      (  Star         ncluding crossing n° 4)
*********************************************     ***     ***     ***************************
          move.w                    #$969A,HSRR1        * F    ight          A (CH0 at 7)
          move.w                    #$69A6,HSRR0        * For     hts o        B (CH8 at 15)
          move.w                    #$0400,Port_C                        2 at GREEN
*  Waiting loop of about 12 seconds
          move.l                    #$00DF    FFF,d
ATT1      sub.l                     #1,d2
          bne                       ATT
* ETAPE n°1  Light n°1 pass at yello
          move.w                    #$96A     SRR1       For Lights on port A (CH0 at 7)
          move.w                    #$69A6,H             For Lights on port B (CH8 at 15))
          move.w                    #$0200,Por           Pedestrians n° 2 at RED
* Waiting loop of about 3 seconds
          move.l                    #$004FCFFF,d2
ATT2      sub.l                     #1,d2
          bne                       ATT2
*  STEP n°2  Authorisation crossing n°4 (Lights 2 and 4 at green)
*************************************************
          move.w                    #$96A9,HSRR1        * For Lights on port A (CH0 at 7)
          move.w                    #$696A,HSRR0        * For Lights on port B (CH8 at 15)
          move.w                    #$0200,Port_C       * Pedestrians at RED
* Waiting loop of about 8 seconds
          move.l                    #$009FCFFF,d2
ATT3      sub.l                     #1,d2
          bne                       ATT3
* STEP n°3  Lights 2 and 4 pass at yellow
          move.w                    #$99A9,HSRR1        * For Lights on port A (CH0 at 7)
          move.w                    #$699A,HSRR0        * For Lights on port B (CH8 at 15)
          move.w                    #$0200,Port_C       * Pedestrians at RED
* Waiting loop of about 3 seconds
          move.l                    #$004FCFFF,d2
ATT4      sub.l                     #1,d2
          bne                       ATT4
```

```
* STEP n°4  Lights n°5 pass at GREEN
************************************
          move.w            #$9A69,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$9AA6,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0300,Port_C      *  Pedestrians n°1 pass at GREEN
* Waiting loop of about 8 seconds
          move.l            #$009FCFFF,d2
ATT5      sub.l             #1,d2
          bne               ATT5
* STEP n°5 Lights n°5 pass at YELLOW
          move.w            #$9A69,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$66A6,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C      *  Pedestrians n°1 pass at RED
* Waiting loop of about 3 seconds
          move.l            #$004FCFFF,d2
ATT6      sub.l             #1,d2
          bne               ATT6
*  STEP n°6  Authorisation main lanes (Lights 1 and 2 at GREEN)
*   ( Start of cycle including crossing n°3)
**********************************************************
          move.w            #$969A,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0400,Port_C      *  Pedestrians n°2 at GREEN
* Waiting loop of about 12 seconds
          move.l            #$00DFCFFF,d2
ATT7      sub.l             #1,d2
          bne               ATT7
* STEP n°7  Light n°2 passes at YELLOW
          move.w            #$999A,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C      * Pedestrians n°2 at ROUGE
* Waiting loop of about 3 seconds
          move.l            #$004FCFFF,d2
ATT8      sub.l             #1,d2
          bne               ATT8
*  STEP n°8  Authorisation crossing n°3  (Light 1 and 3 at GREEN)
****************************************************
          move.w            #$AA5A,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$69A5,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C      *  Pedestrians at RED
* Waiting loop of about 8 seconds
          move.l            #$009FCFFF,d2
ATT9      sub.l             #1,d2
          bne               ATT9
* STEP n°9  Lights n°1 et 3 pass at YELLOW
          move.w            #$6A66,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C      *  Pedestrians at RED
* Waiting loop of about 3 seconds
          move.l            #$004FCFFF,d2
ATT10     sub.l             #1,d2
          bne               ATT10
* STEP n°10  Lights n°5 pass at GREEN
************************************
          move.w            #$9A69,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$9AA6,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0300,Port_C      *  Pedestrians n°1 pass at GREEN
* Waiting loop of about 8 seconds
          move.l            #$009FCFFF,d2
ATT11     sub.l             #1,d2
          bne               ATT11
* ETAPE n°11  Lights n°5 pass at YELLOW
          move.w            #$9A69,HSRR1       * For Lights on port A (CH0 at 7)
          move.w            #$66A6,HSRR0       * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C      *  Pedestrians n°1 pass at RED
* Waiting loop of about 3 seconds
          move.l            #$004FCFFF,d2
ATT12     sub.l             #1,d2
          bne               ATT12
          bra               Deb_BP             * loop
*         End of main loop, end of main program
******************************************************
            end                                * End of listing
```

## 2.2.3 Flowchart with time delay carried out by Microprocessor built-in “Timer”.



## *2.2.4 A68xxx Assembler Program  with 'Timer' use*

```
          *******************************************************************************
          *                          EXPERIMENT  EID210 + TRAFIC LIGHT                  *
          ****************************************************************** ***** ****
          *     Specifications:                                                        *
          *    ****************                                                        *
          *    - Scheduled permutations: main lanes, then Crossing n° 4 , then main lanes, then Crossing n° 4,   *
          *      then secondary lanes, then main lanes, then Crossing n° 3,  then secondary lanes,  etc.         *
          *    - Pedestrian calls and car presence detection are not controlled                                  *
          *    - Time delays are carried out with the 68332 Timer                                                *
          *                                                                             *
          *  FILE NAME:  Feu_Carf_3.SRC                                                 *
          **********************                                                        *
          ********************************************************************************
          * File inclusion specifying the different labels
          include 68332.def
*         Declaration of the variables        *
*************************************
          section        var
COUNTER        ds.l         1
INDICATOR      ds.b         1
*  Start of execute program                   *
*************************************
          section                  code
*         INITIALISE
************************
* Configure port A    in "Discrete Input Output" mode (DIO)-> code $8
START     move.w                 #$8888,CFSR3                * From C[illegible]0 to [illegible]n "DIO" [illegible]de
            move.w               #$8888,CFSR2                * From CH[illegible]o CH[illegible] mode
            move.w               #$8888,CFSR1                * From CHA8 t[illegible]HA1[illegible]DIO" mode
            move.w               #$8888,CFSR0                * From C[illegible]in "[illegible]" mode
* Specify the priorities
            move.w               #$FFFF,CPR1                 * All [illegible] of [illegible] hi[illegible] priority
            move.w               #$FFFF,CPR0                 * All [illegible] of [illegible] hi[illegible] priority
* All Lights are at RED
            move.w               #$9A69,HSRR1                [illegible]n po[illegible] (CH0 at 7)
            move.w               #$69A6,HSRR0                For Lig[illegible]rt [illegible]H8 at 15)
            move.w               #$0700,DIR_Port_C           * [illegible] lsb [illegible] C in output
            move.w               #$0200,Port_C               * Fo[illegible] port C
* Configure the time base
          move.l                 #96,d0                      [illegible] interrupt vector n°
          move.l                 #it_bt,a1                   * f[illegible] is the interrupt function address
          asl.l                  #2,d0
          add.l                  #tab_vect,d0                [illegible]tialise the vectors table
          move.l                 d0,a0
          move.l                 a1,(a[illegible]
          move.l                 #100[illegible]CO[illegible]E[illegible]    * 1000*1mS = 1S
          move.b                 #$00[illegible]DIC[illegible]    * of end of counting
          move.w                 #$000[illegible]TR              * 1 interrupt every ms
          move.w                 #$0760,P[illegible]
************************************
*          MAIN LOOP                   *
************************************
Deb_BP
*  Authorisation main lanes (Lights 1 and 2 at GREEN)
* (  Start of cycle including crossing n° 4)
********************************************
          move.w                 #$969A,HSRR1                * For Lights on port A (CH0 at 7)
          move.w                 #$69A6,HSRR0                * For Lights on port B (CH8 at 15)
          move.w                 #$0400,Port_C               *  Pedestrians  2 at GREEN
* Time delay initialisation of about 12 seconds
          move.l                 #12000,COMPTEUR             * 12000*1mS = 12S
          move.b                 #$00,INDICATEUR             * of end of counting
*  Time delay end waiting loop
ATT1      move.b                 INDICATEUR,D2
          cmp.b                  #01,D2
          bne                    ATT1
* Light n°1 passes at YELLOW
          move.w                 #$96A6,HSRR1                * For Lights on port A (CH0 at 7)
          move.w                 #$69A6,HSRR0                * For Lights on port B (CH8 at 15)
          move.w                 #$0200,Port_C               *  Pedestrians 2 pass at RED
```

```
* Time delay initialisation of about 3 seconds
          move.l              #3000,COMPTEUR            * 3000*1mS = 3S
          move.b              #$00,INDICATEUR           * of end of counting
* Time delay end waiting loop
ATT2      move.b              INDICATEUR,D2
          cmp.b               #01,D2
          bne                 ATT2
*  Authorisation crossing n° 4 (Lights 2 and 4 at GREEN)
****************************************
          move.w              #$96A9,HSRR1              * For Lights on port A (CH0 at 7)
          move.w              #$696A,HSRR0              * For Lights on port B (CH8 at 15)
          move.w              #$0200,Port_C             *  Pedestrians at RED
* Time delay initialisation of about 8 seconds
          move.l              #12000,COMPTEUR           * 8000*1mS = 8S
          move.b              #$00,INDICATEUR           * of end of counting
* Time delay end waiting loop
ATT3      move.b              INDICATEUR,D2
          cmp.b               #01,D2
          bne                 ATT3
* Lights 2 and 4 pass at YELLOW
          move.w              #$99A9,HSRR1              * For Lights on port A (CH0 at 7)
          move.w              #$699A,HSRR0              * For Lights on port B (CH8 at 15)
          move.w              #$0200,Port_C             * Pedestrians at RED
* Time delay initialisation of about 3 seconds
          move.l              #3000,COMPTEUR            * 3000*1mS = 3S
          move.b              #$00,INDICATEUR           * of end of counting
* Time delay end waiting loop
ATT4      move.b              INDICATEUR,D2
          cmp.b               #01,D2
          bne                 ATT4

* Lights 5 pass at GREEN
*************************
          move.w              #$9A69,HSRR1              * For Lights on port A (CH0 at 7)
          move.w              #$9AA6,HSRR0              * For Lights on port B (CH8 at 15)
          move.w              #$0300,Port_C             *  Pedestrians 1 pass at GREEN
* Time delay initialisation of about 8 seconds
          move.l              #8000,COMPTEUR            * 8000*1mS = 8S
          move.b              #$00,INDICATEUR           * of end of counting
* Time delay end waiting loop
ATT5      move.b              INDICATEUR,D2
          cmp.b               #01,D2
          bne                 ATT5
* Lights 5 pass at YELLOW
          move.w              #$9A69,HSRR1              * For Lights on port A (CH0 at 7)
          move.w              #$66A6,HSRR0              * For Lights on port B (CH8 at 15)
          move.w              #$0200,Port_C             * Pedestrians 1 pass at RED
* Time delay initialisation of about 3 seconds
          move.l              #3000,COMPTEUR            * 3000*1mS = 3S
          move.b              #$00,INDICATEUR           * of end of counting
* Time delay end waiting loop
ATT6      move.b              INDICATEUR,D2
          cmp.b               #01,D2
          bne                 ATT6

*  Authorisation main lanes (Lights 1 and 2 at GREEN)
*   ( Cycle start including crossing n°3)
*******************************************
          move.w              #$969A,HSRR1              * For Lights on port A (CH0 at 7)
          move.w              #$69A6,HSRR0              * For Lights on port B (CH8 at 15)
          move.w              #$0400,Port_C             *  Pedestrians 2 at GREEN
* Time delay initialisation of about 12 seconds
          move.l              #12000,COMPTEUR           * 12000*1mS = 12S
          move.b              #$00,INDICATEUR           * of end of counting
* Time delay end waiting loop
ATT7      move.b              INDICATEUR,D2
          cmp.b               #01,D2
          bne                 ATT7
* Light 2 passes at YELLOW
          move.w              #$999A,HSRR1              * For Lights on port A (CH0 at 7)
          move.w              #$69A6,HSRR0              * For Lights on port B (CH8 at15)
          move.w              #$0200,Port_C             *  Pedestrians  2 pass at RED
```

```
* Time delay initialisation of about 3 seconds
          move.l            #3000,COMPTEUR             * 3000*1mS = 3S
          move.b            #$00,INDICATEUR            * of end of counting
*  Time delay end waiting loop
ATT8      move.b            INDICATEUR,D2
          cmp.b             #01,D2
          bne               ATT8
*  Authorisation crossing n° 3  (Lights 1 and 3 at GREEN)
*******************************************
          move.w            #$AA5A,HSRR1               * For Lights on port A (CH0 at 7)
          move.w            #$69A5,HSRR0               * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C              *  Pedestrians at RED
* Time delay initialisation of about 8 seconds
          move.l            #8000,COMPTEUR             * 8000*1mS = 8S
          move.b            #$00,INDICATEUR            * of end of counting
* Time delay end waiting loop
ATT9      move.b            INDICATEUR,D2
          cmp.b             #01,D2
          bne               ATT9
* Lights 1 and 3 pass at YELLOW
          move.w            #$6A66,HSRR1               * For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0               * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C              * Pedestrians at RED
* Time delay initialisation of about 3 seconds
          move.l            #3000,COMPTEUR             * 3000*1mS = 3S
          move.b            #$00,INDICATEUR            * of end of counting
* Time delay end waiting loop
ATT10  move.b       INDICATEUR,D2
          cmp.b             #01,D2
          bne               ATT10
* Lights 5 pass at GREEN
*************************************
          move.w            #$9A69,HSRR1               * For Lights on port A (CH0 at 7)
          move.w            #$9AA6,HSRR0               * For Lights on port B (CH8 at 15)
          move.w            #$0300,Port_C              *  Pedestrians 1 pass at GREEN
* Time delay initialisation of about 8 seconds
          move.l            #8000,COMPTEUR             * 8000*1mS = 8S
          move.b            #$00,INDICATEUR            * of end of counting
* Time delay end waiting loop
ATT11  move.b       INDICATEUR,D2
          cmp.b             #01,D2
          bne               ATT11
* Lights 5 pass at YELLOW
          move.w            #$9A69,HSRR1               * For Lights on port A (CH0 at 7)
          move.w            #$66A6,HSRR0               * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C              * Pedestrians 1 pass at RED
* Time delay initialisation of about 3 seconds
          move.l            #3000,COMPTEUR             * 3000*1mS = 3S
          move.b            #$00,INDICATEUR            * of end of counting
* Time delay end waiting loop
ATT12     move.b            INDICATEUR,D2
          cmp.b             #01,D2
          bne               ATT12
* loop
          bra               Deb_BP
*         End of main loop
**********************************
*         End of main program
******************************************
**********************************************
*         INTERRUPT FUNCTION                   *
*         linked to the time base              *
**********************************************
it_bt     sub.l             #$00000001,COMPTEUR
          cmp.l             #$00000000,COMPTEUR
          bne               it_ret                     * Return if it is not equals to 0
          move.b            #$01,INDICATEUR            * End of time delay
          move.l            #1000,COMPTEUR             * Time delay re-initialisation
it_ret    rte                                          * Interrupt return
*  End of interrupt function
**************************************
*         End of Assembler source file
**************************************
             end
```

# TP 3 : FULL CYCLE WITH PEDESTRIANS CALL PROCESSING AND WITHOUT CAR DETECTION

## 3.1 Topic

| | |
|---|---|
| ***Purpose :*** | **Additional capabilities :**<br>Being capable of acquiring inputs forcing sequence jumps.<br>Being capable of structuring a "Grafcet" micro-program including sequence jumps. |
| ***Specification :*** | The cycle with forks (cf. previous experiment) must be broken in case of pedestrian call occurring.<br>If a call button is pressed down, the normal cycle is interrupted in order to allow pedestrians crossing. Then we go to a state where all "cars" Lights are at red and both "pedestrians" Lights at green.<br>This state lasts for 10 seconds.<br>The coming into this state is only possible after the passing at yellow of the "cars" Lights which were before at green..<br>Waits are carried out by micro-controller built-in Timer. |

Necessary Equipment :

Micro Computer PC-type, with Windows 95 ® or later,

16/32 bits, 68332 micro-controller mother Board , Ref. : EID 100 000

USB link cable or if not available, RS232 cable, Ref. : EGD 000 003

AC/DC 8V 1 A Power Supply, Ref. : EGD000001,

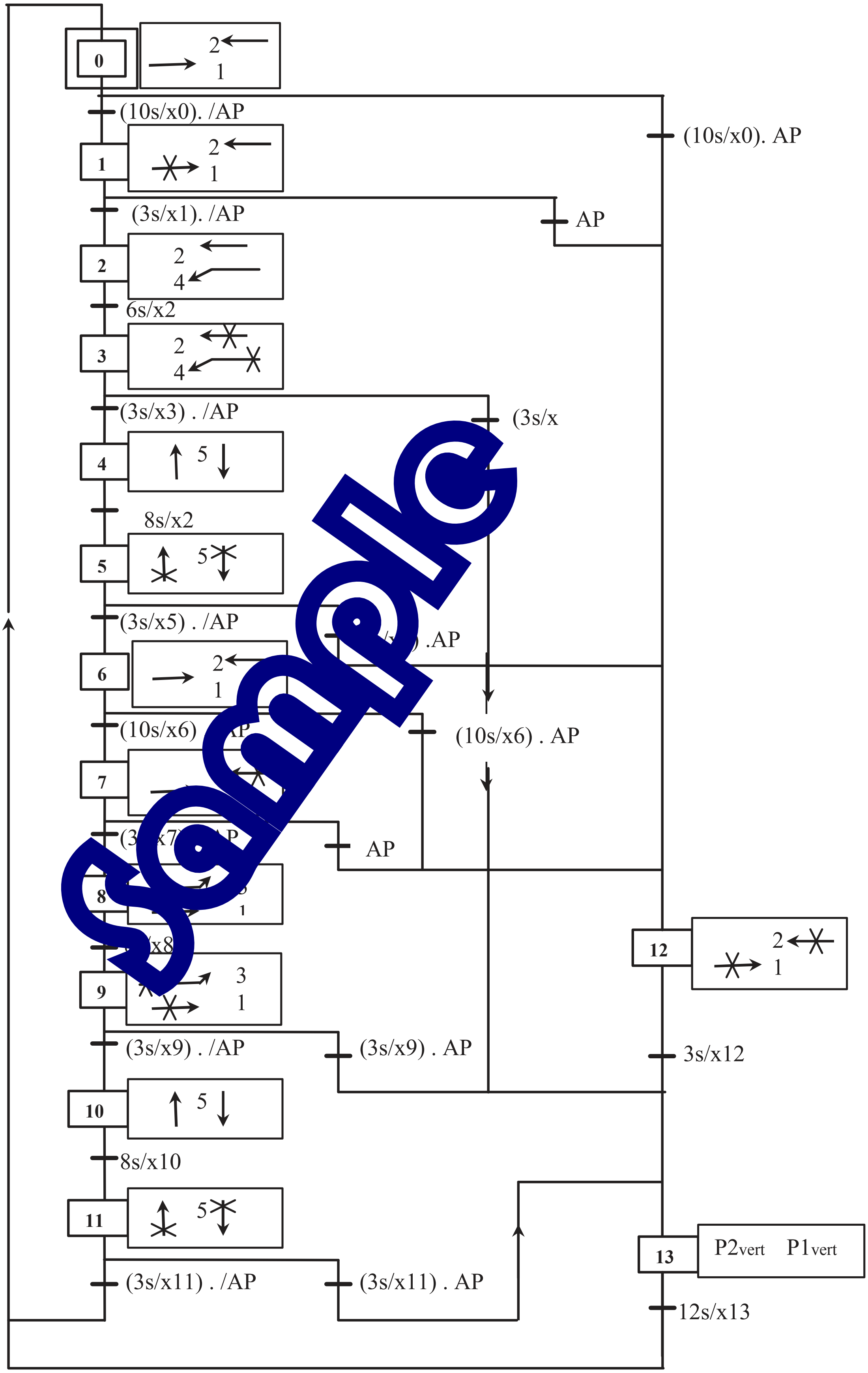
"Traffic Lights" Board, ref. : EID 002 000,

Allocated time duration : 4 hours

# 3.2 Elements of solution

## *3.2.1 Grafcet*

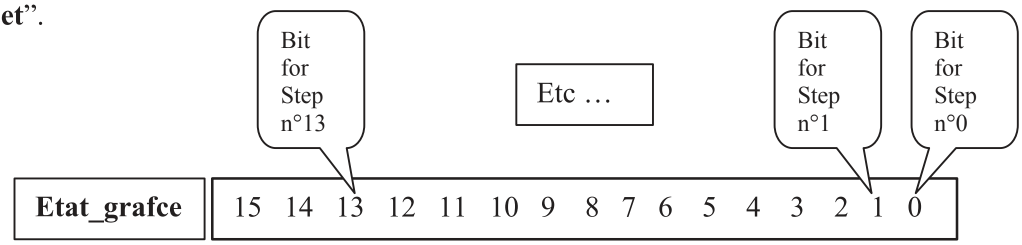
The table specifying the determination of the binary words to be loaded onto the different registers, is given in ANNEX.



## 3.2.2 Grafcet programming flowchart

**Principle:**

To each step is allocated a binary variable, ordered in a global variable, which selected label is: "**Etat_grafcet**".

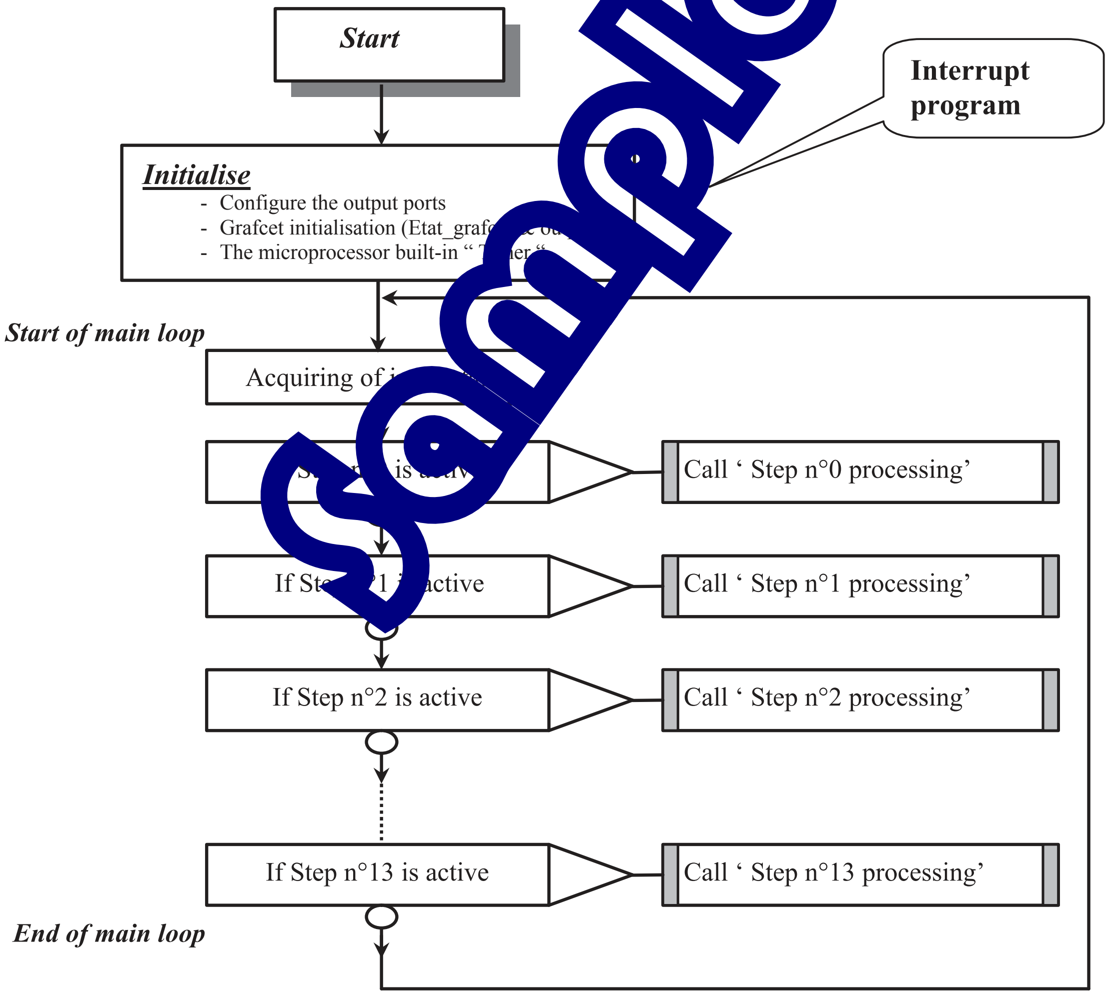


When a step is active, the linked bit is levelled at the logic state '1'. Of course, when a step is not active, it is levelled at'0'.

Thus, the '**Etat_grafcet'** variable initialisation is $0001 ( Hexadecimal value).

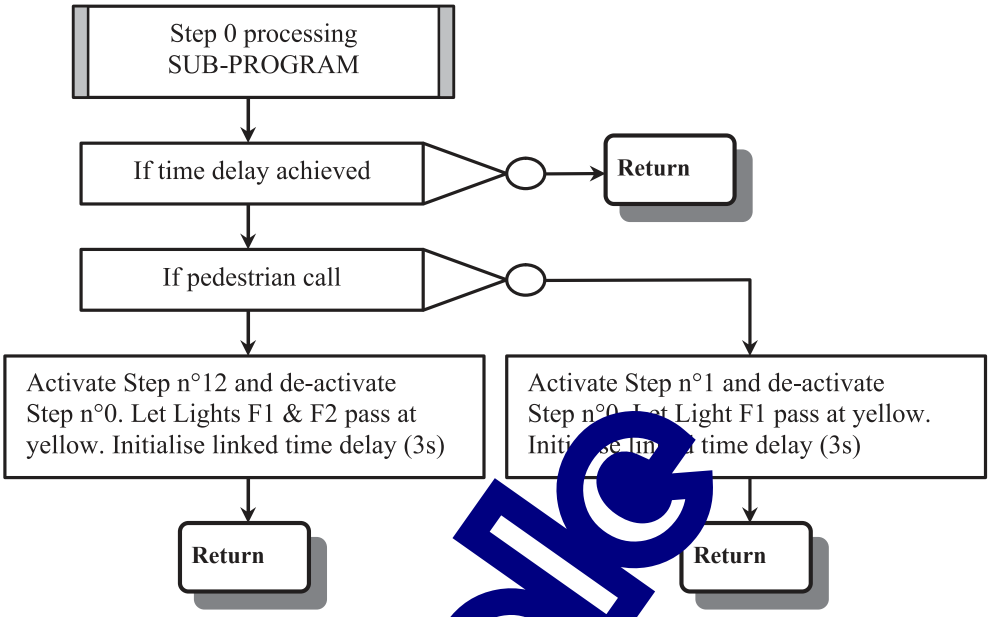
The main loop of the main program includes a search for the active step, where the '**Etat_grafcet'** variable bits are successively checked.

When a bit at level 1 is found, this means that the corresponding step is active. A sub-program, where previous receptivities linked to this active step, are checked.


Start
Interrupt program
Initialise
- Configure the output ports
- Grafcet initialisation (Etat_graf
- The microprocessor built-in "
Start of main loop
Acquiring of i
is activ
Call ' Step n°0 processing'
If Ste 1 i active
Call ' Step n°1 processing'
If Step n°2 is active
Call ' Step n°2 processing'
If Step n°13 is active
Call ' Step n°13 processing'
End of main loop

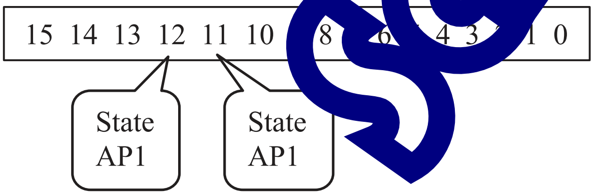
In a step processing sub-program the previous receptivities are tested.
If any of these checked receptivities is true then the **‘Etat_grafcet’** variable is developed, as well as the linked values.

***Example for the Step 0 processing sub-program:***


Step 0 processing SUB-PROGRAM
If time delay achieved
Return
If pedestrian call
Activate Step n°12 and de-activate Step n°0. Let Lights F1 & F2 pass at yellow. Initialise linked time delay (3s)
Activate Step n°1 and de-activate Step n°0. Let Light F1 pass at yellow. Initialise linked time delay (3s)
Return
Return

**Pedestrians call acquisition**

Inputs linked to the pedestrian calls are available on port C : bits of rank 4 and 3. The port C data register is read (label ‘**Port_C')** on 16 bits. The port C state is on the 8 most significant bits.



When a key is pressed down, then we read one ‘0’.

The following operations can be carried out:

- Read Port_C (one ‘word’)
- Carry out a logic AND with $1800
- Compare the result with $1800
- If it is not equal, it is because a call occurred.

**Registers values for outputs activation**

See Annex the table of values

| Step N° | REGISTERS CONTENT (in hexadecimal)<br>HSRR0 | HSRR1 | Port C |
|---|---|---|---|
| 0 | 69A6 | 969A | 0400 |
| 1 | 69A6 | 96A6 | 0200 |
| 2 | 696A | 96A9 | 0200 |
| 3 | 699A | 99A9 | 0200 |
| 4 | 9AA6 | 9A69 | 0300 |
| 5 | 66A6 | 9A69 | 0300 |
| 6 | 69A6 | 969A | 0400 |
| 7 | 69A6 | 999A | 0200 |
| 8 | 69A5 | AA5A | 0200 |
| 9 | 69A6 | 6A66 | 0200 |
| 10 | 9AA6 | 9A69 | 0300 |
| 11 | 66A6 | 9A69 | 0300 |
| 12 | 69A6 | 99A6 | 0300 |
| 13 | A9A6 | 9A69 | 0500 |

## 3.2.3 A68xxx Assembler Program

```
          ************************************************************************************
          *              EXPERIMENT EID210 + TRAFIC LIGHT                                     *
          ************************************************************************************
          *    Specifications:                                                                *
          *    ****************                                                               *
          *    - With the taking into account of the Pedestrian calls                         *
          *    - If there is no Pedestrians calls, there is Lights regular schedule: main lanes then, crossing n°4  *
          *      then, secondary lanes, then, main lanes, then, crossing n°3, then secondary lanes, etc.            *
          *    - Car presence is not controlled                                               *
          *    - The operation is described by a "grafcet"                                    *
          *    - Time delays are carried out with the 68332 timer                             *
          *                                                                                   *
          *  FILE NAME:  Feu_Carf_5.SRC                                                       *
          ********************                                                                *
          ************************************************************************************
          * File inclusion specifying the different labels
          include EID210.def
****************************************
*         Declaration of the variables        *
****************************************
          section        var
COMPTEUR          ds.l      1
Etat_grafcet      ds.w      1
INDICATEUR        ds.b      1            * for indication of the time delay end
MEM_AP ds.b       1                      * for Pedestrian call  MEMory
***********************************
*  Start of the execute program            *
***********************************
          section                   code
*         INITIALISE
*********************
* Configure port A    in  "Discrete Input Output" mode  (DIO)-> code $8
DEBUT    move.w             #$8888,CFSR3             * Fror  HA    CH    in "DIO" mode
         move.w             #$8888,CFSR2             * From  IA4       7 in "DIO" mode
         move.w             #$8888,CFSR1                      2 to C    11 in "DIO" mode
         move.w             #$8888,CFSR0               From C        CH   5 in "DIO" mode
* Specify priorities
         move.w             #$FFFF,CPR1              * Al          A in high priority
         move.w             #$FFFF,CPR0                 ll bi      PB in high priority
* Configure the time base
          move.l            #96,d0                   * 96     e interrupt vector n°
          move.l            #it_bt,a1                       bt is the interrupt function address
          asl.l             #2,d0
          add.l             #tab_ve   d0                 nitialise the vectors table
          move.l            d0,a0
          move.l            a1,(a
          move.l            #800   COM               * 8000*1mS = 8S
          move.b            #$00,I   ICATEUI         * of end of counting
          move.w            #$0008,P                 * 1 interruption every  ms
          move.w            #$0760,PIC
* For configuring port C
          move.w            #$0700,DIR_Port_C        *  The 3 lsb bits of port C in output
* Initialisation of the grafcet
**************************
*  Step n° 0, active at the initialisation
          move.w            #$0001,Etat_grafcet      * Step activation memory
*  Initialisation of actions linked to step n°0
*  Authorisation main lanes (Lights 1 and 2 at green)
          move.w            #$969A,HSRR1             *  For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0             * For Lights on port port B (CH8 at 15)
          move.w            #$0400,Port_C            *  Pedestrians  2 at GREEN
          move.b            #0,MEM_AP                * Init MEMory Pedestrians call
***********************************
*          MAIN LOOP                         *
***********************************
Deb_BP
* Reading of the imputs state " Pedestrians call "
         move.w             Port_C,d0
         andi.w             #$1800,d0                * For isolating the 2 bits of  pedestrians call
         cmp.w              #$1800,d0
         beq                Test_E0                  * Go out if  no detection button pressure
         move.b             #1,MEM_AP                * Set to 1 of pedestrian call memory
```

```
* Search loop of active Step
Test_E0   cmp.w     #$0001,Etat_grafcet            * Check if Step n°0 is active
          bne       Test_E1
          bsr       T_E0                           * Towards processing of Step 0
Test_E1   cmp.w     #$0002,Etat_grafcet            * Check if Step n°1 is active
          bne       Test_E2
          bsr       T_E1                           * Towards processing of Step 1
Test_E2   cmp.w     #$0004,Etat_grafcet            * Check if Step n°2 is active
          bne       Test_E3
          bsr       T_E2                           * Towards processing of Step 2
Test_E3   cmp.w     #$0008,Etat_grafcet            * Check if Step n°3 is active
          bne       Test_E4
          bsr       T_E3                           * Towards processing of Step 3
Test_E4   cmp.w     #$0010,Etat_grafcet            * Check if Step n°4 is active
          bne       Test_E5
          bsr       T_E4                           * Towards processing of Step 4
Test_E5   cmp.w     #$0020,Etat_grafcet            * Check if Step n°5 is active
          bne       Test_E6
          bsr       T_E5                           * Towards processing of Step 5
Test_E6   cmp.w     #$0040,Etat_grafcet            * Check if Step n°6 is active
          bne       Test_E7
          bsr       T_E6                           * Towards processing of Step 6
Test_E7   cmp.w     #$0080,Etat_grafcet            * Check if Step n°7 is active
          bne       Test_E8
          bsr       T_E7                           * Towards processing of Step 7
Test_E8   cmp.w     #$0100,Etat_grafcet            * Check if Step n°8 is active
          bne       Test_E9
          bsr       T_E8                           * Towards processing of Step 0
Test_E9   cmp.w     #$0200,Etat_grafcet            * Check if Step n°9 is active
          bne       Test_E10
          bsr       T_E9                           * Towards processing of Step 9
Test_E10  cmp.w     #$0400,Etat_grafcet            * Check if Step n°10 is active
          bne       Test_E11
          bsr       T_E10                          * Towards processing of Step 10
Test_E11  cmp.w     #$0800,Etat_grafcet            * Check if Step n°11 is active
          bne       Test_E12
          bsr       T_E11                          * Towards processing of Step 11
Test_E12  cmp.w     #$1000,Etat_grafcet            * Check if Step n°12 is active
          bne       Test_E13
          bsr       T_E12                          * Towards processing of Step 12
Test_E13  cmp.w     #$2000,Etat_grafcet            * Check if Step n°13 is active
          bne       Test_Fin
          bsr       T_E13                          * Towards processing of Step 0
* END of search loop of active step
Test_Fin  bra       Deb_BP                         * Go to beginning of main loop
*         End of main loop
*         End of main program
************************************************************
************************************************************
*  STEP PROCESSING SUB- PROGRAM                            *
************************************************************
*         Processing of step n°0                 *
******************************************
T_E0      move.b    INDICATEUR,D2
          cmp.b              #01,D2
          bne                T_E0_r                * Go out if time delay not achieved
          cmp.b              #0,MEM_AP                       * To know if pedestrians call
          bne                T_E0_1                * Go out if pedestrians call
          *If we get the time delay end without pedestrians call then, go to step 1(Light 1 passes at yellow)
          move.w             #$96A6,HSRR1          *  For Lights on  port A (CH0 at 7)
          move.w             #$69A6,HSRR0          *  For Lights on port B (CH8 at 15)
          move.w             #$0200,Port_C                   *  Pedestrians 2 pass at RED
          * Go to step n°1
          move.w             #$0002,Etat_grafcet   * bit of rank  1 go to 1 and others are at  0

          move.l             #3000,COMPTEUR        *  Time delay initialisation of about 3 seconds        3000*1mS = 3S
          move.b             #$00,INDICATEUR       * of end of counting
          rts                                               * Return to main loop
        * We get the end of time delay and pedestrians call  -> go to step n°12 , Lights 1 and 2  pass  at YELLOW
T_E0_1    move.b             #0,MEM_AP                       * Reset of pedestrians call memory
          move.w             #$1000,Etat_grafcet   * Bit of rank 12  passes at 1 and others bits stay at  0
          move.w             #$99A6,HSRR1          *  For Lights on port A (CH0 at 7)
          move.w             #$69A6,HSRR0                    *  For Lights on port B (CH8 at 15)
          move.w             #$0200,Port_C                   *  Pedestrians 2 at RED
          *Time delay initialisation of about 3 seconds
```

```
            move.l              #3000,COMPTEUR              *  3000*1mS = 3S

            move.b              #$00,INDICATEUR             * of end of counting
T_E0_r      rts                 *End of step n°0 processing, back in the main loop.
*****************************************************************************************
*           Step n°1 processing            *
****************************************
T_E1        cmp.b               #0,MEM_AP                   * For knowing if pedestrians call
            bne                 T_E1_1                      * Go out if pedestrians call
            move.b      I       NDICATEUR,D2                * Check if time delay end
            cmp.b               #01,D2
            bne                 T_E1_r                      * Go out if time delay not achieved
            * We have got the time delay end without pedestrians call  thus, we go to step 2
            * Light 4 passes at green
            move.w              #$96A9,HSRR1                *  For Lights on port A (CH0 at 7)
            move.w              #$696A,HSRR0                *  For Lights on port B (CH8 at 15)
            move.w              #$0200,Port_C               *  Pedestrians 2 passes at RED
            *  Time delay initialisation of about 6 seconds
            move.l              #6000,COMPTEUR              * 6000*1mS = 6S
            move.b              #$00,INDICATEUR             * of end of counting
            * Go to step n°2
            move.w              #$0004,Etat_grafcet         * bit of rank  2  passes at level 1 and other bits stay at 0
            rts                                             Return to main loop
T_E1_1  * There is pedestrians call, thus passing at step 12  * Passing at step n°12 , Light 2  passes at yellow
            move.b              #0,MEM_AP                   * Reset of pedestrians call memory
            move.w              #$1000,Etat_grafcet         * Bit of  rank 12  passes at level 1 and other bits stay at 0
            move.w              #$99A6,HSRR1                *  For Lights on port A (CH0 at 7)
            move.w              #$69A6,HSRR0                *  For Lights on port B (CH8 at 15)
            move.w              #$0200,Port_C               *  Pedestrians 2 passes at RED
            move.l              #3000,COMPTEUR              * Time delay initialisation 3000*1mS = 3S
            move.b              #$00,INDICATEUR             * of end of counting
T_E1_r      rts                 * End of step n° 1 processing, return to main loop
******************************************************************************************
*           Processing of step n°2          *
*****************************************
T_E2        move.b      INDICATEUR,D2
            cmp.b               #01,D2
            bne                 T_E2_r                      * Go out if time delay not achieved
            * Time delay achieved thus, passing at step n° 3
            move.w              #$0008,Etat_grafcet         * bit of rank 3 passes at level 1 and other bits stay at 0
            * Lights 2 and 4 pass at yellow
            move.w              #$99A9,HSRR1                *  For Lights on port A (CH0 at 7)
            move.w              #$699A,HSRR0                *  For Lights on port B (CH8 at 15)
            move.w              #$0200,Port_C               *  Pedestrians 1 passes at red
            move.l              #3000,COMPTEUR              * Time delay initialisation 3000*1mS = 3S
            move.b              #$00,INDICATEUR             * of end of counting
T_E2_r      rts                 * End of step n° 2 processing, return to main loop
*****************************************************************************************
*           Processing of step n°3
****************************************
T_E3        move.b      INDICATEUR,D2
            cmp.b               #01,D2
            bne                 T_E3_r                      * Go out if time delay not achieved
            *cmp                #$0,MEM_AP                  * For knowing if there is no pedestrians call
            *bne                T_E3_1                      * Go out if pedestrian call
            * We have got the time delay end without pedestrians call  thus, we go to step 4*  Light 5 passes at green
            move.w              #$0010,Etat_grafcet         * bit of rank 4 passes at level 1 and other bits stay at 0
            move.w              #$9A69,HSRR1                *  For Lights on  port A (CH0 at 7)
            move.w              #$9AA6,HSRR0                *  For Lights on port B (CH8 at 15)
            move.w              #$0300,Port_C               *  Pedestrian Lights  P 1 at green
            move.l              #8000,COMPTEUR              * Time delay initialisation  8000*1mS = 8S
            move.b              #$00,INDICATEUR             * of end of counting
            rts                                             *return to main loop
            * We have got the time delay end and pedestrian call  thus, we go to step n° 13
T_E3_1 move.w                   #$2000,Etat_grafcet   * bit of rank 13 passes at level 1 and other bits stay at 0
            move.w              #$9A69,HSRR1                * For Lights on port A (CH0 à 7)
            move.w              #$A9A6,HSRR0                * For Lights on port B (CH8 à 15)
            move.w              #$0500,Port_C               *  Pedestrian call at green
            move.b              #0,MEM_AP                   * Pedestrian call memory reset
            move.l              #8000,COMPTEUR              * Time delay initialisation of about 10000*1mS = 10S
            move.b              #$00,INDICATEUR             * of end of counting
T_E3_r      rts                 * End of step n°3 processing, return to main loop
****************************************************************************************
```

```
*          Step n°4 processing                     *
************************************
T_E4      move.b     I             NDICATEUR,D2
          cmp.b                    #01,D2
          bne                      T_E4_r                      * Go out if time delay not achieved
          * End of time delay, thus go to step n°5, Lights 5 pass at yellow
          move.w                   #$0020,Etat_grafcet         * Bit of rank  5  passes at level 1 and other bits stay at  0
          move.w                   #$9A69,HSRR1                *  For Lights on port A (CH0 at 7)
          move.w                   #$A6A6,HSRR0                *  For Lights on port B (CH8 at 15)
          move.w                   #$0200,Port_C               *  Pedestrians at red
          move.l                   #3000,COMPTEUR              * Time delay initialisation 3000*1mS = 3S
          move.b                   #$00,INDICATEUR             * of end of counting
T_E4_r    rts                      * End of step n°4 processing, return to main loop
********************************************************************************
*          Step n°5 processing     *
************************************
T_E5      move.b                   INDICATEUR,D2
          cmp.b                    #01,D2
          bne                      T_E5_r                      * Go out if time delay not achieved
          cmp                      #$0,MEM_AP                  * For knowing if there is pedestrians call
          bne                      T_E5_1                      * Go out if pedestrian  call
          * We have got the time delay end without pedestrian call  thus, we go to step n° 6 (1 and 2 at green)
          move.w                   #$0040,Etat_grafcet         * Bit of rank  6  passes at level 1 and other bits stay at  0
          move.w                   #$969A,HSRR1                * For Lights on port A (CH0 at 7)
          move.w                   #$69A6,HSRR0                * For Lights on port B (CH8 at 15)
          move.w                   #$0200,Port_C               * Pedestrian Lights at red
          move.l                   #10000,COMPTEUR             * Time delay initialisation 10000*1mS = 10S
          move.b                   #$00,INDICATEUR             * of end of counting
          rts
          * We have got the time delay end with pedestrian call  thus, we go to step n° [illegible]
T_E5_1    move.w                   #$2000,Etat_grafcet         * * Bit of rank [illegible] passes at level 1 and other bits stay at  0
          move.w                   #$9A69,HSRR1                * For Lights on port A (CH0 at 7)
          move.w                   #$A9A6,HSRR0                * For Lights on port B (CH8 at 15)
          move.w                   #$0500,Port_C               * Pedestrian Lights at green
          move.b                   #0,MEM_AP                   * Pedestrian call memory reset
          move.l                   #12000,COMPTEUR             * Time delay initialisation 12000*1mS = 12S
          move.b                   #$00,INDICATEUR             * of end of counting
T_E5_r    rts                      * End of step n°5 processing, return to main loop
********************************************************************************
*          Step n°6  processing     *
************************************
T_E6      move.b                   INDICATEUR,D2
          cmp.b                    #01,D2
          bne                      T_E6_r                      * Go out if time delay not achieved
          cmp                      #$0,MEM_AP                  * For knowing if there is pedestrians call
          bne                      T_E6_1                      * Go out if pedestrian  call
          * We have got the time delay end without pedestrian call  thus, we go to step 7  (Light 2 passes at yellow)
          move.w                   #$00[illegible],Etat_grafcet         * Bit of rank 7 passes at level 1 and other bits stay at  0
          move.w                   #$99[illegible],HSRR1                * For Lights on port A (CH0 at 7)
          move.w                   #$69[illegible],HSRR0                * For Lights on port B (CH8 at 15)
          move.w                   #$0200,Port_C               * Pedestrian Lights at red
          move.l                   #3000,COMPTEUR              * Time delay initialisation 30000*1mS = 3S
          move.b                   #$00,INDICATEUR             * of end of counting
          rts
          * We have got the time delay end with pedestrian call  thus, we go to step 12 (Lights 1 and 2 at yellow)
T_E6_1    move.w                   #$1000,Etat_grafcet         * Bit of rank 7 passes at level 1 and other bits stay at  0
          move.w                   #$999A,HSRR1                * For Lights on port A (CH0 at 7)
          move.w                   #$69A6,HSRR0                * For Lights on port B (CH8 at 15)
          move.w                   #$0200,Port_C               * Pedestrian Lights at red
          move.b                   #0,MEM_AP                   * Pedestrian call  memory reset
          move.l                   #3000,COMPTEUR              * Time delay initialisation 3000*1mS = 3S
          move.b                   #$00,INDICATEUR             * of end of counting
T_E6_r    rts                      * End of step n°6 processing, return to main loop
********************************************************************************
*          Step n°7 processing *
************************************
T_E7      move.b                   INDICATEUR,D2
          cmp.b                    #01,D2
          bne                      T_E7_r                      * Go out if time delay not achieved
          cmp                      #$0,MEM_AP                  * For knowing if there is pedestrians call
          bne                      T_E7_1                      * Go out if pedestrian  call
        * We have got the time delay end without pedestrian call  thus, we go to step 8,  Lights 1 and 3 pass at green
          move.w          #$0100,Etat_grafcet         * Bit of rank 8 passes at level 1 and other bits stay at  0
          move.w          #$AA5A,HSRR1                * For Lights on port A (CH0 at 7)
          move.w          #$69A5,HSRR0                * For Lights on port B (CH8 at 15)
          move.w          #$0200,Port_C               * Pedestrian Lights at red
          move.l          #8000,COMPTEUR              * Time delay initialisation 8000*1mS = 8S
          move.b          #$00,INDICATEUR             * of end of counting
          rts
          * We have got one pedestrian call, then we go to step n° 12 (Lights 1 and 2 at yellow)
T_E7_1      move.w          #$1000,Etat_grafcet         * Bit of rank 12 passes at level 1 and other bits stay at  0
          move.w          #$99A6,HSRR1                * For Lights on port A (CH0 à 7)
          move.w          #$69A6,HSRR0                * For Lights on port B (CH8 à 15)
          move.w          #$0200,Port_C               * Pedestrian Lights at red
          move.b          #0,MEM_AP                   * Pedestrian call  memory reset
          move.l          #3000,COMPTEUR              * Time delay initialisation 3000*1mS = 3S
          move.b          #$00,INDICATEUR             * of end of counting
T_E7_r      rts             * End of step n°7 processing, return to main loop
**************************************************************************
********************************
*         Step n°8 processing     *
********************************
T_E8        move.b          INDICATEUR,D2
          cmp.b           #01,D2
          bne             T_E8_r                      * Go out if time delay not achieved
          * End of time delay, thus, we go to step n°9 , Lights 1 and 3 pass at yellow
          move.w          #$0200,Etat_grafcet         * Bit of rank 9 passes at level 1 and other bits stay at  0
          move.w          #$6AA6,HSRR1                * For Lights on port A (CH0 at 7)
          move.w          #$69A6,HSRR0                * For Lights on port B (CH8 at 15)
          move.w          #$0200,Port_C               * Pedestrian Lights at red
          move.l          #3000,COMPTEUR              * Time delay initialisation 3000*1mS = 3S
          move.b          #$00,INDICATEUR             * of end of counting
T_E8_r      rts             * End of step n°8 processing, return to main loop
**************************************************************************
********************************
*         Step n°9 processing               *
********************************
T_E9        move.b    INDICATEUR,D2
          cmp.b           #01,D2
          bne             T_E9_r                      * Go out if time delay not achieved
          cmp             #$0,MEM_AP                  * F[illegible] if there is pedestrians call
          bne             T_E9_1                      * Go out if pedestrian call
          * We have got the time delay end without pedestrian call thus, we go to step n°10, Lights 5 pass at green
          move.w          #$0400,Etat_grafcet         * Bit of rank 9 and others bits stay at 0
          move.w          #$9A69,HSRR1                * For Lights on port A (CH0 at 7)
          move.w          #$9AA6,HSRR0                * For Lights on port B (CH8 at 15)
          move.w          #$0300,Port_C               * Pedestrians 1  at green
          move.l          #0800[illegible]MPT[illegible]UR  * Time delay initialisation 8000*1mS = 8S
          move.b          #$00,INDICATEUR             * of end of counting
          rts             * Return to main loop
          * We have got the time delay end with pedestrian call  thus, we go to step 13 (Car Lights at red)
T_E9_1  move.w          #$2000,Etat_grafcet         * Bit of rank 13 passes at level 1 and other bits stay at  0

          move.w          #$9A69,HSRR1                * For Lights on port A (CH0 at 7)
          move.w          #$A9A6,HSRR0                * For Lights on port B (CH8 at 5)
          move.w          #$0500,Port_C               *  Pedestrians Lights at green
          move.b          #0,MEM_AP                   * Pedestrian call memory reset
          move.l          #12000,COMPTEUR             * Time delay initialisation 12000*1mS = 12S
          move.b          #$00,INDICATEUR             * of end of counting
T_E9_r      rts             * End of step n°9 processing, return to main loop
*****************************************************************************************
********************************
*         Step n°10 processing    *
********************************
T_E10       move.b    INDICATEUR,D2
          cmp.b           #01,D2
          bne             T_E10_r                     * Go out if time delay not achieved
          * End of time delay, thus we go to step n°11 , Lights n°5 pass at yellow
          move.w          #$0800,Etat_grafcet         * Bit of rank 11 passes at level 1 and other bits stay at  0
          move.w          #$9A69,HSRR1                * For Lights on port A (CH0 à 7)
          move.w          #$A6A6,HSRR0                * For Lights on port B (CH8 à 15)
          move.w          #$0200,Port_C               *  Pedestrians Lights at red
          move.l          #3000,COMPTEUR              * Time delay initialisation 3000*1mS = 3S
          move.b          #$00,INDICATEUR             * of end of counting
T_E10_r     rts             * End of step n°10 processing, return to main loop
```

```
************************************
*         Step n°11 processing     *
************************************
T_E11     move.b            INDICATEUR,D2
          cmp.b             #01,D2
          bne               T_E11_r                * Go out if time delay not achieved
          cmp               #$0,MEM_AP             * For knowing  if pedestrians call
          bne               T_E11_1                * Go out if pedestrians call
          * We have got the time delay end without pedestrian call  thus, we go to step 0 (1 et 2 at green)
          move.w            #$0001,Etat_grafcet    * Bit of rank 0 passes at level 1 and other bits stay at  0
          move.w            #$969A,HSRR1           *  For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0           * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C          *  Pedestrians Lights at red
          move.l            #10000,COMPTEUR        * Time delay initialisation 10000*1mS = 10S
          move.b            #$00,INDICATEUR        * of end of counting
          rts
          * We have got the time delay end with pedestrian call  thus, we go to step 13
T_E11_1   move.w            #$2000,Etat_grafcet    * Bit of rank 0 passes at level 1 and other bits stay at  0
          move.w            #$9A69,HSRR1           * For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0           * For Lights on port B (CH8 at 15)
          move.w            #$0500,Port_C          * Pedestrians Lights at green
          move.b            #0,MEM_AP              * Pedestrian call memory reset
          move.l            #10000,COMPTEUR        * Time delay initialisation 10000*1mS = 10S
          move.b            #$00,INDICATEUR        * of end of counting
T_E11_r   rts               * End of step n°10 processing, return to main loop
************************************
*         Step n°12 processing     *
************************************
T_E12     move.b    INDICATEUR,D2
          cmp.b             #01,D2
          bne               T_E12_r                * Go out if time delay achieved
          * We have got the time delay end without pedestrian call  thus, we go to step 13 ([illegible] lights at red)
          move.w            #$2000,Etat_grafcet    * Bit of rank 13 passes at level 1 and other bits stay at  0
          move.w            #$9A69,HSRR1           * For Lights on port A (CH0 à 7)
          move.w            #$A9A6,HSRR0           * For Lights on port B (CH8 à 15)
          move.w            #$0500,Port_C          * Pedestrians Lights at green
          move.l            #10000,COMPTEUR        * Time delay initialisation 10000*1mS = 10S
          move.b            #$00,INDICATEUR        of end of counting
T_E12_r   rts               * End of step n°12 processing, return to main loop
************************************
*         Step n°13 processing*
************************************
T_E13     move.b    INDICATEUR,D2
          cmp.b             #01,D2
          bne               T_E13_r                * Go out if time delay achieved
          * We have got the time delay end without pedestrian call  thus, we go to step 0 (1 et 2 at green)
          move.w            #$0001,Etat_grafcet    * Bit of rank 13 passes at level 1 and other bits stay at  0
          move.w            #$969A,HSRR1           * For Lights on port A (CH0 at 7)
          move.w            #$69A6,HSRR0           * For Lights on port B (CH8 at 15)
          move.w            #$0200,Port_C          * Pedestrians Lights at red
          move.l            #8000,COMPTEUR         * Time delay initialisation 8000*1mS = 8S
          move.b            #$00,INDICATEUR        * of end of counting
T_E13_rrts                  * End of step n°13 processing, return to main loop
************************************************
*         INTERRUPT FUNCTION                   *
*         linked to the time base              *
************************************************
it_bt     sub.l             #$00000001,COMPTEUR
          cmp.l             #$00000000,COMPTEUR
          bne               it_ret                 * Return if not equal to 0
          move.b    #$01,INDICATEUR  * End of time delay
          move.l            #$5000,COMPTEUR * Time delay re-initialisation
it_ret    rte                                      * Interrupt return
*  End of interrupt function
*************************************
*         End of Assembler source file
*********************************************
          end
```

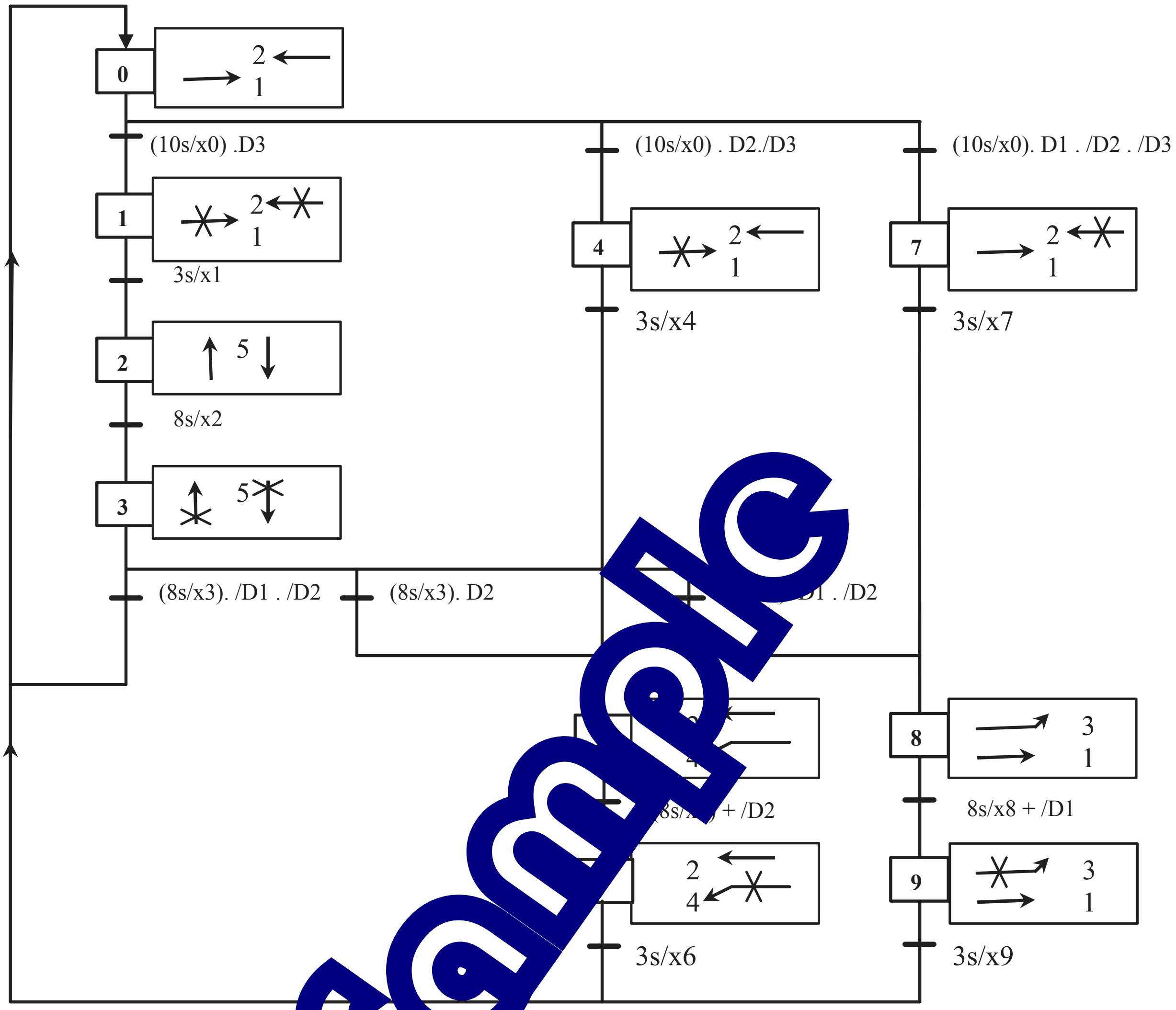
# TP 4 : CYCLE TAKING INTO ACCOUNT PEDESTRIAN CALL WITHOUT CAR DETECTION

## 4.1 Topic

|  |  |
|---|---|
| ***Purpose :*** | Additional abilities:<br>Being capable of representing by a “grafcet” flowchart complex specifications.<br>Being capable of carry out a “grafcet” program presenting many OR-type divergences and et convergences. |
| ***Specifications:*** | In a current use, only both main lanes are auth[illegible]sed [illegible]ts F1a[illegible]F2 at green), pedestrian lanes n°2 also (P2 at green).<br>Cars turning right must respect pedestrians priori[illegible]<br>This state only changes when a car is detect[illegible] on [illegible]e [illegible] th[illegible] secondary lanes (sensor D3 on lanes n°5) or on the fork lanes (sensor D1on lane n° 3 [illegible]nd [illegible]or [illegible]2 on lane n°4).<br>From the current use, as described pre[illegible] can [illegible] detection on one (or many) of the 3 sensors D1, D2, D3.<br>The following  hierarchy will be t[illegible] i[illegible]cco[illegible]t:<br>P(D3) > P(D2) > P(D1)<br>➔ 1- If a car is detected by D3, [illegible]ights [illegible]d F2 pass at yellow then, after 3 Sec., pass at red in the same time than Lights [illegible]. This state goes on 8 Sec. before Lights F5 pass at yellow during 3 Sec. and be[illegible]re the ini[illegible]ate is on again (duration of this state: 10 Sec. at least).<br>➔ 2- If a car a[illegible] is not activated), Light F1 passes at yellow then, after 3 Sec., passes at r[illegible] in [illegible] tha[illegible] passes at green. This state goes on 8 Sec. before Light F4 passes at yellow d[illegible]ng 3 Se[illegible] before the initial state is on again (duration of this state: 10 Sec. at least).<br>➔ 3- If a car activate[illegible] while D3 is not activated), Light F2 passes at yellow then, after 3 Sec., passes at red in the same than F3 passes at green. This state goes on 8 Sec. before Light F3 passes at yellow during 3 Sec. and before the initial state is on again (duration of this state: 10 Sec. at least).<br>➔ From the state while Lights F5 are at yellow since 3Sec., if a car activate D2, we meet up with the 2nd condition as described before-.<br>➔ From the state while Lights F5 are at yellow since 3Sec., if a car activate D2 (while D1 is not activated, then, we meet up with the 3rd condition as described before. |

# 4.2 Elements of solution

## *4.2.1 “Grafcet” flowchart*



**Values of registers for outputs activation**
See in Annex the binary values table

| Step N° | REGISTERS CONTENTS (In hexadecimal) HSRR0 | HSRR1 | Port C |
|---|---|---|---|
| 0 | 69A6 | 969A | 0400 |
| 1 | 69A6 | 99A6 | 0200 |
| 2 | 9AA6 | 9A69 | 0300 |
| 3 | 66A6 | 9A69 | 0300 |
| 4 | 69A6 | 96A6 | 0200 |
| 5 | 696A | 96A9 | 0200 |
| 6 | 699A | 96A6 | 0400 |
| 7 | 69A6 | 999A | 0200 |
| 8 | 69A5 | AA5A | 0200 |
| 9 | 69A6 | 6A5A | 0200 |

**Acquisition of the car detection sensors states**
Inputs linked to car detection sensors are available on port C :

- bits of rank 5 for sensor labelled D1
- bits of rank 6 for sensor labelled D2
- bits of rank 7 for sensor labelled D3

The data register of port C is read (label ‘**Port_C'**) on 16 bits. The states of port C being on the 8 most significant bits.
When a car is detected then, we read a ‘0’ level.
The following actions can be carried out :

- Read Port_C (one ‘word’)
- Carry out a logic AND with $E000
- Compare the result with $E000
- If there is not equality, it is because a car is detected.

## 4.2.2 A68xxx Assembler Program

```
          *********************************************************************
          *              EXPERIMENT  EID210 + TRAFIC LIGHTS                    *
          *********************************************************************
          *     Specifications:                                                *
          *    *****************                                               *
          *    -  With car presence detection                                  *
          *    - Car presence is not controlled                                *
          *    -  The operation is described by a "grafcet"                    *
          *    -  Time delays are carried out with the 68332 timer             *
          *                                                                    *
          *  FILE NAME:  Feu_Carf_4.SRC                                        *
          ***********************                                              *
          *********************************************************************
          * File inclusion specifying the different labels
          include 68332.def
***************************************
*         Declaration of variables          *
***************************************
          section       var
COMPTEUR            ds.l        1
Etat_grafcet        ds.w        1
INDICATEUR          ds.b        1
**************************************
*  Start of execute program                *
**************************************
          section             code
*         INITIALISE
*************************
* Configure port A    in "Discrete Input Output" mode  (DIO)-> code $8
DEBUT    move.w             #$8888,CFSR3               * From C[illegible] in [illegible]O" mode
         move.w             #$8888,CFSR2               * From [illegible]A4 to C[illegible]DIO" mode
         move.w             #$8888,CFSR1               * Fro[illegible]H[illegible]CH[illegible]1 in "DIO" mode
         move.w             #$8888,CFSR0               * Fro[illegible]HA[illegible]C[illegible]15 in "DIO" mode
* Specify priorities
         move.w             #$FFFF,CPR1                [illegible]A in[illegible] priority
         move.w             #$FFFF,CPR0                All  bi[illegible] hi[illegible]priority
* Configure the time base
         move.l             #96,d0                     * 96[illegible]rrupt  vector n°
         move.l             #it_bt,a1                  [illegible]it-bt[illegible]e interrupt function address
         asl.l              #2,d0
         add.l              #tab_vect,d0               * In[illegible]se the vectors table
         move.l             d0,a0
         move.l             a1,(a0)
         move.l             #1000,COMPT[illegible]R           [illegible]000*1mS = 1S
         move.b             #$00[illegible]AT[illegible]R            [illegible] of end of counting
         move.w             #$00[illegible],PIT[illegible]            * 1 interrupt every 1 msec.
         move.w             #$07[illegible]PICR
* Initialisation of the "grafcet"
****************************
*  Step n° 0 active at the initialisation
         move.w             #$0001,Etat_gr[illegible]          * Steps activation memory
*  Initialisation of actions linked to step n°0
*  Authorisation main lanes (Lanes n°1 and 2 at green)
         move.w             #$969A,HSRR1               *  For Lights on port A (CH0 at 7)
         move.w             #$69A6,HSRR0               * For Lights on port B (CH8 at 15)
         move.w             #$0400,Port_C              *  Pedestrians n°2 at GREEN
* Configure in outputs the 3 bits of port QS where the diodes are connected
*  For the display of the "grafcet" activation
         move.w             #$0070,PQSCTR              * 3 outputs on LED
         move.w             #$0070,d3                  * For displaying the Step 0 n°
**************************************
*        MAIN LOOP
**************************************
Deb_BP
* Reading of inputs state
         move.w             Port_C,d0
         and.w              #$E000,d0                  * For isolating the bits of car detection
         eor.w              #$E000,d0                  * For having levels 1 if car detection
* Active step search loop
Test_E0  cmp.w              #$0001,Etat_grafcet        * Check if Step n°0 is active
         bne                Test_E1
         bsr                T_E0                       * Towards Step n°0 processing
```

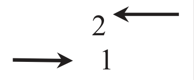
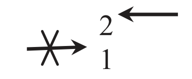
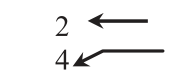
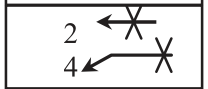
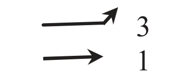
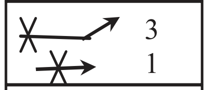
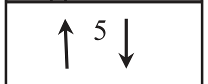
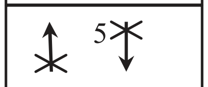
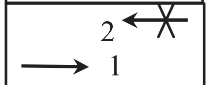
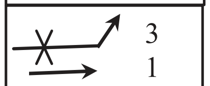
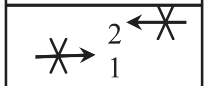
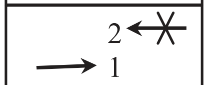
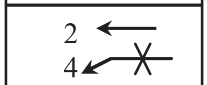
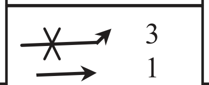
```
Test_E1   cmp.w            #$0002,Etat_grafcet        * Check if Step n°1 is active
          bne              Test_E2
          bsr              T_E1                       * Towards Step n° 1 processing
Test_E2   cmp.w            #$0004,Etat_grafcet        * Check if Step n°2 is active
          bne              Test_E3
          bsr              T_E2                       * Towards Step n° 2 processing
Test_E3   cmp.w            #$0008,Etat_grafcet        * Check if Step n°3 is active
          bne              Test_E4
          bsr              T_E3                       * Towards Step n° 3 processing
Test_E4   cmp.w            #$0010,Etat_grafcet        * Check if Step n°4 is active
          bne              Test_E5
          bsr              T_E4                       * Towards Step n° 4 processing
Test_E5   cmp.w            #$0020,Etat_grafcet        * Check if Step n°5 is active
          bne              Test_E6
          bsr              T_E5                       * Towards Step n° 5 processing
Test_E6   cmp.w            #$0040,Etat_grafcet        * Check if Step n°6 is active
          bne              Test_E7
          bsr              T_E6                       * Towards Step n° 6 processing
Test_E7   cmp.w            #$0080,Etat_grafcet        * Check if Step n°7 is active
          bne              Test_E8
          bsr              T_E7                       * Towards Step n° 7 processing
Test_E8   cmp.w            #$0100,Etat_grafcet        * Check if Step n°8 is active
          bne              Test_E9
          bsr              T_E8                       * Towards Step n° 8 processing
Test_E9   cmp.w            #$0200,Etat_grafcet        * Check if Step n°9 is active
          bne              Visu_EG
          bsr              T_E9                       * Towards Step n° 9 processing

* Display of  "GRAFCET" State on port "QS" (LEDs D12, D11, D10  of Central Processing Unit)
Visu_EG   move.w    d3,PORTQS            * Load on port QS
          bra       Deb_BP
*         End of main loop
***************************************
*         End of main program
*****************************************
**********************************************************
*  STEP PROCESSING SUB PROGRAM                  *
**********************************************************
*         Step n°0 processing                   *
*****************************************
T_E0      move.b           INDICATEUR,D2
          cmp.b            #01,D2
          bne              T_E0_3                     * Go out if the time delay is not achieved
          cmp              #$2000,d0
          bne T_E0_1                 * Go out if [illegible] and D1 are not available
          * We get the combination /D3 and D2 and D1
          * Light n° 2 passes at yellow
          move.w           #$99[illegible],HSRR1      *  For Lights on  port A (CH0 at 7)
          move.w           #$69A6,HSRR0               *  For Lights on port B (CH8 at 15)
          move.w           #$0200,Port_C              *  Pedestrian n°2 passes at RED
          *  Time delay initialisation of about 3 seconds
          move.l           #3000,COMPTEUR  * 3000*1mS = 3S
          move.b           #$00,INDICATEUR            * of end of counting
          * Passing to Step n°7
          move.w           #$0080,Etat_grafcet        * Bit of rank 7 passes at level 1 and bit 0 at level  0
          move.w           #$0000,d3                  * For the display of  Step n°7
          * Return to main loop
          rts
T_E0_1    and              #$C000,d0                  * Mask of input D1
          cmp              #$4000,d0                  * D2 has priority on D1
          bne              T_E0_2                     * Go out if  /D3 and D2 are not available
          * We get the combination /D3 and D2
          * Passing at Step °4 , Light n°1 passes at yellow
          move.w           #$0010,Etat_grafcet        * Bit of rank 4 passes at level 1 and bit 0 at level  0
          move.w           #$0030,d3                  * For the display of  Step n°4
          move.w           #$96A6,HSRR1               * For Lights on port A (CH0 at 7)
          move.w           #$69A6,HSRR0               * For Lights on port B (CH8 at 15)
          move.w           #$0200,Port_C              *  Pedestrian n°2 passes at RED
          * Time delay initialisation of about 3 seconds
          move.l           #3000,COMPTEUR             * 3000*1mS = 3S
          move.b           #$00,INDICATEUR            * of end of counting
          * Return to main loop
          rts
```

```
T_E0_2  and             #$8000,d0                * Masking of input D2 (D1 already masked)
        cmp             #$8000,d0                * D3 has priority on D2 and on D1
        bne             T_E0_3                   * Go out if D3 not available
        * On a D3
        * Passing at Step n°1, Lights n°1 and n° 2 pass at yellow
        move.w          #$0002,Etat_grafcet      * Bit of rank 1 passes at level 1 and bit 0 at level  0
        move.w          #$0060,d3                * Display of Step n°1
        move.w          #$99A6,HSRR1             * For Lights on port A (CH0 at 7)
        move.w          #$69A6,HSRR0             * For Lights on port B (CH8 at 15)
        move.w          #$0200,Port_C            * Pedestrian n°2 passes at RED
        * Time delay initialisation of about 3 seconds
        move.l          #3000,COMPTEUR  * 3000*1mS = 3S
        move.b          #$00,INDICATEUR  * of end of counting
        * Return to main loop
T_E0_3  rts             *End of Step n°0 processing, return to main loop
***********************************************************************************
*       Step n° 1 processing            *
*************************************
T_E1    move.b          INDICATEUR,D2
        cmp.b           #01,D2
        bne             T_E1_1                   * Go out if the time delay is not achieved
        *Time delay is achieved thus, passing at Step n°2
        move.w          #$0004,Etat_grafcet      * Bit of rank 2 passes at level 1 and bit 1 at level  0
        move.w          #$0050,d3                * Display of Step n°2
        * Lights n°5 pass at green
        move.w          #$9A69,HSRR1             * For Lights on port A (CH0 at 7)
        move.w          #$9AA6,HSRR0             * For Lights on port B (CH8 at 15)
        move.w          #$0300,Port_C            * Pedestrian n°1 passes at RED
        * Time delay initialisation of about 6 seconds
        move.l          #6000,COMPTEUR           * 6000*1mS = 6S
        move.b          #$00,INDICATEUR          * of end of counting
T_E1_1  rts             * End of Step n°1 processing, return to main loop
*************************************************************************************
*       Step n°2  processing    *
******************************************
T_E2    move.b          INDICATEUR,D2
        cmp.b           #01,D2
        bne             T_E2_1                   * Go out if the time delay is not achieved
        * Time delay is achieved thus, passing at Step n°3
        move.w          #$0008,Etat_grafcet      * Bit of rank 3 passes at level 1 and bit 2 at level  0
        move.w          #$0040,d3                * Display of Step n°3
        * Lights n°5 pass at yellow
        move.w          #$9A69,HSRR1             * For Lights on port A (CH0 at 7)
        move.w          #$66A6,HSRR0             * For Lights on port B (CH8 at 15)
        move.w          #$0200,Port_C            * Pedestrian n°1 passes at RED
        * Time delay initialisation of about 3 seconds
        move.l          #3000,COMPTEUR  * 3000*1mS = 3S
        move.b          #$00,INDICATEUR  * of end of counting
T_E2_1  rts             * End of Step n°2 processing, return to main loop
***********************************************************************************
*       Step n°3  processing    *
**************************************
T_E3    move.b          INDICATEUR,D2
        cmp.b           #01,D2
        bne             T_E3_3                   * Go out if the time delay is not achieved
        and             #$6000,d0                * Masking of D3
        cmp             #$2000,d0
        bne T_E3_1                               * Go out if  /D2 and D1 not available
        * If D1 and /D2 are available thus, go to Step n°8 , Lights n°1 and n°3 pass at green
        move.w          #$0100,Etat_grafcet      * Bit of rank 8 passes at level 1 and bit 3 at level  0
        move.w          #$AA5A,HSRR1             * For Lights on port A (CH0 at 7)
        move.w          #$69A5,HSRR0             * For Lights on port B (CH8 at 15)
        move.w          #$0200,Port_C            *  Pedestrians at red
        * Time delay initialisation of about 6 seconds
        move.l          #6000,COMPTEUR           * 6000*1mS = 6S
        move.b          #$00,INDICATEUR          * of end of counting
        rts                                      * return to main loop
```

```
T_E3_1   and             #$4000,d0                * Masking of input D1
         cmp             #$4000,d0
         bne T_E3_2                               * Go out if  D2 not available
         *  D2 is available thus, we go to Step n°5, Lights n°2 and n°4 pass at green
         move.w          #$0020,Etat_grafcet      * Bit of rank 5 passes at level 1 and bit 3 at level  0
         move.w          #$0020,d3                * Display of Step n°5
         move.w          #$96A9,HSRR1             *  For Lights on port A (CH0 at 7)
         move.w          #$696A,HSRR0             *  For Lights on port B (CH8 at 15)
         move.w          #$0200,Port_C            * Pedestrians at red
         * Time delay initialisation of about 6 seconds
         move.l          #6000,COMPTEUR           * 6000*1mS = 6S
         move.b          #$00,INDICATEUR          * of end of counting
         rts                                      *return to main loop
T_E3_2 * Both D1 and D2 are not available thus, return to Step n°0, Lights n°1 and 2 pass at green
         move.w          #$0001,Etat_grafcet      * Bit of rank 0 passes at level 1 and bit 3 at level  0
         move.w          #$0070,d3                * Display of Step n°0
         move.w          #$969A,HSRR1             * For Lights on port A (CH0 at 7)
         move.w          #$69A6,HSRR0             * For Lights on port B (CH8 at 15)
         move.w          #$0400,Port_C            *  Pedestrians n°2 at GREEN
         * Time delay initialisation of about 12 seconds
         move.l          #12000,COMPTEUR * 12000*1mS = 12S
         move.b          #$00,INDICATEUR  * of end of counting
T_E3_3 rts               * End of Step n°3 processing, return to main loop
******************************************************************************
*        Step n°4 processing           *
*************************************
T_E4     move.b          INDICATEUR,D2
         cmp.b           #01,D2
         bne             T_E4_2                   * Go out i[illegible] not achiev[illegible]
         * End of time delay, thus passing at Step n°5, Lights n°2 and 4 pa[illegible] green
         move.w          #$0020,Etat_grafcet      * Bit of rank [illegible]es a[illegible] 1 and bit 4 at level  0
         move.w          #$0020,d3                * Display of [illegible]
         move.w          #$96A9,HSRR1             * For Li[illegible]s on po[illegible] at 7)
         move.w          #$696A,HSRR0             * For [illegible]nts [illegible]ort [illegible]C[illegible]8 at 15)
         move.w          #$0200,Port_C            *  Ped[illegible]ria[illegible]d
         * Time delay initialisation of about 6 seconds
         move.l          #6000,COMPTEUR           * [illegible] = 6[illegible]
         move.b          #$00,INDICATEUR          [illegible]of en[illegible]ting
T_E4_2 rts               * End of Step n°4 processing, [illegible]ur[illegible] main[illegible]
******************************************************************************
*        Step n°5 processing           *
*****************************************
T_E5     move.b          INDICATEUR,D2
         cmp.b           #01,D2
         beq             T_E5_2                   [illegible] out if time delay is achieved
T_E5_1 and               #$4000,d0                [illegible]nput D2 is selected
         cmp             #$000[illegible]
         bne             T_E[illegible]           * Go out if D2 is not back to level 0
         * End of time delay OR D[illegible]0, th[illegible]n°6 , Light n°4 passes at yellow
T_E5_2 move.w            #$004[illegible]Etat_grafce[illegible]  * Bit of rank 6 passes at level 1 and bit 5 at level  0
         move.w          #$0010,[illegible]       * Display of Step n°6
         move.w          #$96A9,HSR[illegible]    * For Lights on port A (CH0 at 7)
         move.w          #$699A,HSR[illegible]    * For Lights on port B (CH8 at 15)
         move.w          #$0200,Port_C            * Pedestrians at red
         * Time delay initialisation of about 6 seconds
         move.l          #003000,COMPTEUR         * 3000*1mS = 3S
         move.b          #$00,INDICATEUR          * of end of counting
T_E5_3 rts               * End of Step n°5 processing, return to main loop
*****************************************************************************
*        Step n°6 processing    *
*****************************************
T_E6     move.b          INDICATEUR,D2
         cmp.b           #01,D2
         bne             T_E6_1                   * Go out if time delay is not achieved
         * Return to Step 0, Lights n° 1 and 2 pass at green
         move.w          #$0001,Etat_grafcet      * Bit of rank 0 passes at level 1 and bit 3 at level  0
         move.w          #$0070,d3                * Display of Step n°0
         move.w          #$969A,HSRR1             * For Lights on port A (CH0 at 7)
         move.w          #$69A6,HSRR0             * For Lights on port B (CH8 at 15)
         move.w          #$0200,Port_C            *  Pedestrians n°2 at green
         * Time delay initialisation of about 10 seconds
         move.l          #10000,COMPTEUR * 10000*1mS = 10S
         move.b          #$04,INDICATEUR          * of end of counting
T_E6_1   rts             End of Step n°6 processing, return to main loop
******************************************************************************
```

```
*         Step n°7 processing                *
*************************************
T_E7      move.b                INDICATEUR,D2
          cmp.b                 #01,D2
          bne                   T_E7_1               * Go out if time delay is not achieved
          * Passing at Step n° 8, Lights n°1 and 3 pass at green
          move.w                #$0100,Etat_grafcet  * Bit of rank 08 passes at level 1 and bit 7 at level  0
          move.w                #$0070,d3            * Display of Step n°0  (instead of 8)
          move.w                #$AA5A,HSRR1         *  For Lights on port A (CH0 at 7)
          move.w                #$69A5,HSRR0         * For Lights on port B (CH8 at 15)
          move.w                #$0200,Port_C        *  Pedestrians at RED
          * Time delay initialisation of about 6 seconds
          move.l                #6000,COMPTEUR  * 8000*1mS = 6S
          move.b                #$00,INDICATEUR  * of end of counting
T_E7_1 rts                      * End of Step n°7 processing, return to main loop
********************************************************************************
*         Step n°8 processing     *
*************************************
T_E8      move.b                INDICATEUR,D2
          cmp.b                 #01,D2
          bne                   T_E8_1               * Go out if time delay is not achieved
          bra                   T_E5_2
T_E8_1    and                   #$2000,d0            * Input D1 is selected
          cmp                   #$0000,d0
          bne                   T_E8_3               * Go out if D2 is not  ck         0
          * End of time delay OR D3=0, thus passing at Step n°9 , Light n°3    es at  llo
T_E8_2    move.w                #$0200,Etat_grafcet  * Bit of ra  9 p     s  level 1 an   t 8 at level  0
          move.w                #$0070,d3            * Display    tep n               )
          move.w                #$6A5A,HSRR1         * For Lights o    t A (     at 7)
          move.w                #$69A6,HSRR0         * For Lights         (CH    15)
          move.w                #$0200,Port_C        * Pedes    s at R
          Time delay initialisation of about 3 seconds
          move.l                #3000,COMPTEUR       * 300    m
          move.b                #$00,INDICATEUR      * of en   f co
T_E8_3    rts                   * End of Step n°7 processing, ret          oop
********************************************************************************
*         Step n°9 processing     *
*****************************************
T_E9      move.b                INDICATEUR,D2
          cmp.b                 #01,D2
          bne                   T_E9_1               * G      if time delay is not achieved
          * Return to Step n°0, Lights n°1 and 2 pa
          move.w                #$0001,Etat_gra         of rank 1 passes at level 1 and bit 3 at level  0
          move.w                #$0070,d3             isplay of Step n°0
          move.w                #$969    RR          For Lights on port A (CH0 at 7)
          move.w                #$69   ,H    0       * For Lights on port B (CH8 at 15)
          move.w                #$02   Por           * Pedestrians n°2 at green
          * Time delay initialisation   bout 10 se   ds
          move.l                #10000,     EUR  10000*1mS = 10S
          move.b      #$04,INDICATEUR  *    nd   ounting
T_E9_1 rts                      * End of Step     essing, return to main loop
***********************************************************************************
*********************************************
*         INTERRUPT FUNCTION                 *
*         linked to the time base            *
*********************************************
it_bt     sub.l                 #$00000001,COMPTEUR
          cmp.l                 #$00000000,COMPTEUR
          bne                   it_ret               * Return if it is not equal to 0
          move.b                #$01,INDICATEUR      * End of time delay
          *move.l               #1000,COMPTEUR       * Re-initialisation
it_ret    rte                                        * Interrupt return
*  End of interrupt function
***********************************
*         End of Assembler source file
***********************************
          end
```

| ANNEX<br>Table of values | Light P2 | | Light P1 | | Light F5 | | | Light F4 | | | Light F3 | | | Light F2 | | | Light F1 | | | CONTENTS OF REGISTERS<br>(In Hexadecimal) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Port C | | | Port B | | | | | | | | Port A | | | | | | | | HSRR0 | HSRR1 | Port C |
| | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
|  | G | R | G | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | 69A6 | 969A | 0400 |
| | 1 | 0 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 01 | 10 | 10 | 01 | 10 | 10 | | | |
|  | G | R | G | R | G | O | R | V | O | R | G | O | R | G | Y | R | G | Y | R | 69A6 | 96A6 | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 01 | 10 | 10 | 10 | 01 | 10 | | | |
|  | G | R | G | R | G | O | R | V | O | R | G | O | R | G | Y | R | G | Y | R | 696A | 96A9 | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 01 | 10 | 10 | 10 | 10 | 01 | 01 | 10 | 10 | 10 | 10 | 01 | | | |
|  | G | R | G | R | G | [illegible] | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | 699A | 99A9 | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | [illegible] | 10 | 01 | 10 | 10 | 10 | 01 | 10 | 01 | 10 | 10 | 10 | 01 | | | |
|  | G | R | G | R | G | Y | [illegible] | [illegible] | Y | R | G | Y | R | G | Y | R | G | Y | R | 69A5 | AA5A | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 01 | 10 | 10 | 10 | 10 | 01 | 01 | 10 | 10 | | | |
|  | G | R | G | R | G | Y | R | G | [illegible] | [illegible] | G | Y | R | G | Y | R | G | Y | R | 69A6 | 6A66 | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | [illegible] | 10 | 01 | 10 | 01 | 10 | 10 | 10 | 01 | 10 | 01 | 10 | | | |
|  | G | R | G | R | G | Y | R | [illegible] | [illegible] | R | [illegible] | Y | R | G | Y | R | G | Y | R | 9AA6 | 9A69 | 0300 |
| | 0 | 1 | 1 | 10 | 01 | 10 | 10 | 10 | [illegible] | [illegible] | [illegible] | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | | | |
|  | G | R | G | R | G | Y | R | G | Y | [illegible] | G | [illegible] | [illegible] | G | Y | R | G | Y | R | 66A6 | 9A69 | 0300 |
| | 0 | 1 | 1 | 10 | 10 | 01 | 10 | 10 | 10 | 01 | [illegible] | 10 | [illegible] | 10 | 10 | 01 | 10 | 10 | 01 | | | |
|  | G | R | G | R | G | Y | R | G | Y | R | G | [illegible] | R | [illegible] | [illegible] | R | G | Y | R | 69A6 | 999A | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | [illegible] | 10 | [illegible] | 10 | 01 | 10 | 10 | | | |
|  | G | R | G | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | 69A6 | 6A5A | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 01 | 10 | 10 | 10 | 01 | 01 | 10 | 10 | | | |
|  | G | R | G | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | 69A6 | 99A6 | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 01 | 10 | 10 | 01 | 10 | | | |
|  | G | R | G | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | 69A6 | 999A | 0300 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 01 | 10 | 01 | 10 | 10 | | | |
|  | G | R | G | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | 699A | 96A6 | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 01 | 10 | 10 | 10 | 01 | 01 | 10 | 10 | 10 | 10 | 01 | | | |
|  | G | R | G | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | G | Y | R | 69A6 | 6A5A | 0200 |
| | 0 | 1 | 0 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 01 | 10 | 10 | 10 | 01 | 01 | 10 | 10 | | | |
| No F Lights<br>But P Light | G | R | G | R | G | Y | R | G | Y | R | G | Y | R | G | O | R | G | Y | R | A9A6 | 9A69 | 0500 |
| | 1 | 0 | 1 | 10 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | 10 | 10 | 01 | | | |