

# LES DATES SUR $\mu$ CONTRÔLEURS

ACCU32 est déclaré dans le segment DATA de \_div32

; ACCU32: DS 4

; additionne R2:R3 a R4:R5:R6:R7 résultat dans R4:R5:R6:R7

addR2R3:

```
MOV    A,R7
ADD    A,R3    ; additionne le LSB
MOV    R7,A
MOV    A,R6
ADDC   A,R2    ; puis le MSB
MOV    R6,A
RET
```

; multiplie R4:R5:R6:R7 par deux

```
_mul2: MOV    A,R7
ADD    A,R7
MOV    R7,A
MOV    A,R6
ADDC   A,R6
MOV    R6,A
MOV    A,R5
ADDC   A,R5
MOV    R5,A
MOV    A,R4
ADDC   A,R4
MOV    R4,A
RET
```

; divise R2:R3 par deux

```
_divR2R3: CLR    C
MOV    A,R2
RRC    A
MOV    R2,A
MOV    A,R3
RRC    A
MOV    R3,A
RET
```

ANNEE EQU 1

MOIS EQU 2

JOUR EQU 3

; conversion de la date en nombre de jour comptée depuis le 1 mars 1960

; la date est dans ACCU32[1..3]

; ACCU32+ANNEE -> offset de l'année (0 pour 1960, 1 pour 1961, 2 pour 1962...)

; ACCU32+MOIS -> mois

; ACCU32+JOUR -> jour dans le mois

; le résultat est dans R6:R7

\_datejour:

```
MOV    R6,#0
INC    ACCU32+MOIS    ; Incrémente le numéro du mois
MOV    A,ACCU32+MOIS
CJNE   A,#4,date0     ; teste si inférieur au mois de mars
date0: JNC    date1
ADD    A,#12           ; non...
MOV    ACCU32+MOIS,A   ; oui, additionne un ans (en mois)
DEC    ACCU32+ANNEE    ; et enlève l'année correspondante
date1: MOV    R6,#0
MOV    R7,A            ; MSB a 0 (pour la multiplication)
MOV    R4,#1EH
MOV    R5,#9AH         ; valeur hexa de 7834
CALL   _mul16           ; multiplie R4:R5 par R6:R7
```

; résultat dans R4:R5:R6:R7, divise par 256 en prenant R5:R6

```
MOV    A,R5            ; prend le MSB
MOV    R2,A            ; sauve le MSB
MOV    A,R6            ; prend le LSB
MOV    R3,A            ; et le sauve
```

; dans R2:R3 le nombre de jour correspondant aux mois

```
MOV    A,ACCU32+ANNEE ; prend les années
MOV    R7,A
MOV    R6,#0           ; moins de 256 années ! (au max 136)
CALL   _mul2           ; X 2
MOV    R4,#0B6H
MOV    R5,#0A0H        ; valeur hexa de 46752 = 365.25 * 128
CALL   _mul16          ; multiplie R4:R5 par R6:R7
```

; dans R5:R6:R7 le nombre de jours correspondant aux années X 256

```
MOV    A,R5            ; X 2 X 128 = X 256
```

suite page suivante

La manipulation des dates

est toujours une opération

délicate pour le

programmeur. Dans de

nombreux programmes, on

est amené à faire des calculs

sur les dates, pour connaître

par exemple le nombre de

jours séparant deux dates ou

plus simplement pour

déterminer le jour de la

semaine d'une date donnée.

On doit alors choisir entre

deux représentations de

date : soit on manipule des

dates sous forme de nombre

de jours écoulés, soit les

dates sont représentées

sous forme année, mois, jour.

Dans le premier cas, les calculs de distance entre deux dates sont triviaux mais leur représentation sous forme habituelle AA/MM/JJ est problématique ; dans le second cas, les calculs sur les dates sont extrêmement difficiles à réaliser. Le programmeur qui souhaite réaliser les fonctions de conversion d'une représentation à une autre se heurtera à deux problèmes : le calcul des années bissextiles et le calcul de la longueur des mois. Nous savons

que la durée de rotation de la terre autour du soleil est approximativement de 365,2422 jours. Jules César a introduit un calendrier basé sur des années de 365,25 jours. L'erreur sur la vitesse de rotation de la terre introduit une erreur qui a été corrigée sous le Pape Grégoire XIII à qui nous devons le calcul actuel des années bissextiles :

Les années bissextiles sont les années divisibles par 4 exceptées les années divisibles par 100 sauf si elles sont aussi divisibles par 400. Ainsi 1900 n'était pas une année bissextile, mais 2000 sera bissextile et 2100 ne le sera pas. Ainsi pour la période de 1901 à 2099, le calendrier basé sur une année de 365,25 jours sera exact. La majorité des applications à microprocesseur ayant à traiter des dates dans cet intervalle là, nous ne nous intéresserons pas aux corrections centenaires.

Les années bissextiles sont plus longues d'un jour. Malheureusement pour les programmeurs, ce jour est inséré juste avant le premier jour de mars, alors que l'on aurait préféré le dernier jour de l'année. Aussi dans les calculs, nous considérerons que le premier jour de l'année est le premier mars.

Les problèmes liés à la longueur irrégulière des différents mois peuvent se résoudre de différentes manières : soit on utilise une table qui contient la longueur des différents mois, soit on tire profit des propriétés du résultat entier de la multiplication par 30,6. Le tableau ci dessous illustre ce propos :

M	J = int(M X 30,6)	L = J <sub>i+1</sub> - J <sub>i</sub>	MOIS
0	0	30	
1	30	31	
2	61	30	
3	91	31	
4	122	31	mars
5	153	30	avril
6	183	31	mai
7	214	30	juin
8	244	31	juillet
9	275	31	août
10	306	30	septembre
11	336	31	octobre
12	367	30	novembre
13	397	31	décembre
14	428	31	janvier
15	459	30	février

Les formules ci-dessous illustrent la conversion nombre de jours N en date AA, MM, JJ. Nous prendrons comme année de référence 1960, le premier jour que nous pourrions manipuler sera donc le 1 mars 1960. On pourra prendre n'importe quelle année bissextile comme année de référence.

Conversion JOUR / DATE :

AA = (N - 0,1) / 365,25

JJ = reste de la division précédente + 1

MM = (122 + JJ) / 30,6

MM = MM - 1

si MM >= 13 {

MM = MM - 12

AA = AA + 1

}

JJ = reste de la division précédente + 1

AA = 1960 + AA

```

XCH      A,R6      ; il ne reste plus qu'à diviser le résultat
MOV      R7,A      ; par 256
CALL     addR2R3    ; additionne les deux précédents résultats
MOV      R3,A
MOV      R2,#0
MOV      R3,ACCU32+JOUR ; prend le jour
CALL     addR2R3    ; additionne au reste
CLR      C
MOV      A,R7
SUBB     A,#122
MOV      R7,A
MOV      A,R6
SUBB     A,#0
MOV      R6,A
RET

```

; conversion du nombre de jour compté depuis le 1 mars 1960 en date  
; le nombre de jour est passé dans R6:R7  
; la date est renvoyée dans ACCU32[1..3]  
; ACCU32+ANNEE -> offset de l'année (0 pour 1960, 1 pour 1961, 2 pour 1962...)  
; ACCU32+MOIS -> mois  
; ACCU32+JOUR -> jour dans le mois

```

_jourdate:
CLR      A
MOV      R4,A
MOV      R5,A
MOV      ACCU32+0,A
CALL     _mul2
CALL     _mul2      ; multiplie R6:R7 par 4
MOV      A,R7      ; la retenue est déjà a 0
SUBB     A,#1      ; soustrait 1
MOV      R7,A      ; à la valeur 24 bits R5:R6:R7
MOV      A,R6
SUBB     A,#0
MOV      R6,A
MOV      A,R5
SUBB     A,#0
MOV      R5,A
CLR      A
MOV      ACCU32+0,A
MOV      ACCU32+1,R5 ; transfère (R6:R7 X 4) - 1 dans ACCU32
MOV      ACCU32+2,R6
MOV      ACCU32+3,R7
MOV      R4,A
MOV      R5,A
MOV      R6,#05
MOV      R7,#0B5H ; valeur hexa de 1461 (365.25 X 4)
CALL     _div32     ; divise le nombre de jour(-epsilon) par 365.25
; dans ACCU32+3, le nombre d'année (< 256)
PUSH     ACCU32+3; sauve le nombre d'années
; dans R2:R3 le nombre de jour restant (X 4 -le numérateur étant X 4-)
CALL     _divR2R3 ; / 2
CALL     _divR2R3 ; / 2
MOV      A,R3
ADD      A,#1+122 ; +1 pour avoir le nombre de jour
MOV      ACCU32+2,A ; +122 pour calculer le numéro du mois
MOV      A,R2      ; résultat dans ACCU32+1:ACCU32+2 (X 256)
ADDC     A,#0
MOV      ACCU32+1,A
CLR      A
MOV      ACCU32+0,A
MOV      ACCU32+3,A
MOV      R4,A
MOV      R5,A
MOV      R6,#1EH
MOV      R7,#9AH ; valeur hexa de 30.6 X 256
CALL     _div32     ; dans ACCU32+3 le numéro du mois
; dans R2:R3 le numéro du jour X 256 (donc numéro du jour dans R2)
MOV      A,ACCU32+3 ; prend le numéro du mois
POP      ACCU32+ANNEE ; restaure le numéro de l'année
DEC      A          ; mois = mois - 1
CJNE     A,#13,jour0 ; si mois < 13
jour0:   JC          jour1 ; pas de changement
SUBB     A,#12      ; sinon janvier ou février, soustrait un an
INC      ACCU32+ANNEE ; et incrémente l'année
jour1:   MOV      ACCU32+MOIS,A ; sauve le numéro du mois
INC      R2        ; prend le numéro du jour + 1
MOV      ACCU32+JOUR,R2 ; et sauve le numéro du jour définitif
RET

```

Conversion DATE / JOUR :

AA = AA - 1960

si MM > 2 {

MM = MM + 1

}

sinon {

MM = MM + 13

AA = AA - 1

}

NN = int(MM X 30,6)

+ int(AA X 365,25) + JJ - 122

Le listing 1 présente les deux sous-programmes de conversion \_jourdate et

\_datejour. Ils sont écrit pour des microcontrôleurs de type 80C5X. Les routines \_div32 et \_mul16 ont déjà été présentées dans ces colonnes (numéro 556). Elles réalisent respectivement la division de ACCU32 par R4:R5:R6:R7 avec résultat dans ACCU32 et reste dans R0:R1:R2:R3, et la multiplication de R4:R5 par R6:R7. Ces différents sous-programmes sont disponibles sur le serveur ERP.

J.L. VERN