B1. Echantillonnage

B1.1. Théorème de Shannon

Pour échantillonner correctement un signal *x(t)* et éviter le phénomène de repliement (aliasing), il faut choisir une fréquence d'échantillonnage telle que :

$$\boxed{F_e \geq 2F_{MAX}}$$

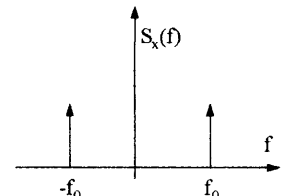
F_{MAX} représentant la fréquence maximale contenue dans le signal *x(t)* soit :

$$S_x(f) = 0 \text{ pour } |f| \geq F_{MAX}$$

S_x représentant la densité spectrale de puissance (spectre) du signal à échantillonner *x(t)*.

B1.2. D'après le théorème de Shannon, pour un signal sinusoidal de fréquence f_0,, la fréquence d'échantillonnage minimale est :

$$\boxed{(F_e)_{\min} = 2f_0}$$



Densité spectrale de puissance d'un signal sinusoidal

D'où la période d'échantillonnage maximale :

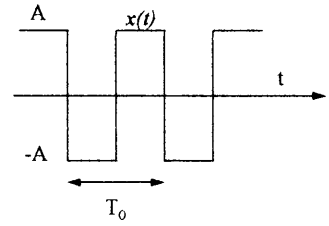
$$\boxed{(T_e)_{\max} = \frac{1}{(F_e)_{\min}} = \frac{1}{2f_0} = \frac{T_0}{2}}$$

avec T_0 la période du signal sinusoidal.

Cela revient à prendre **au minimum 2 points (échantillons) par période**.

B1.3. En théorie, un signal carré n'est pas échantillonnable car il possède une bande spectrale infinie (pas de fréquence maximale !)

Calcul des coefficients de Fourier pour un signal carré de période T_0 et de dynamique *(-A,A)*



On peut considérer un signal centré sans perte de généralité ; s'il n'est pas centré, cela rajoute une composante continue, c'est-à-dire que cela modifiera uniquement le coefficient d'ordre 0. On a :

$$x_n = \frac{1}{T_0}\int_{T_0} x(t)\exp\left(-j2\pi n\frac{t}{T_0}\right)dt = -\frac{A}{T_0}\int_{-T_0/2}^{0}\exp\left(-j2\pi n\frac{t}{T_0}\right)dt + \frac{A}{T_0}\int_{0}^{T_0/2}\exp\left(-j2\pi n\frac{t}{T_0}\right)dt$$

Ce qui donne en intégrant :

$$x_n = \frac{A}{T_0}\left[\frac{1}{j2\pi\frac{n}{T_0}}\exp\left(-j2\pi n\frac{t}{T_0}\right)\right]_{-T_0/2}^{0} + \frac{A}{T_0}\left[-\frac{1}{j2\pi\frac{n}{T_0}}\exp\left(-j2\pi n\frac{t}{T_0}\right)\right]_{0}^{T_0/2}$$

$$x_n = -\frac{A}{j2\pi n}\left(\exp\left(j2\pi n\frac{1}{2}\right) + \exp\left(-j2\pi n\frac{1}{2}\right)\right) + \frac{2A}{j2\pi n} = \frac{A}{j\pi n}\left(1-\cos(\pi n)\right)$$

$$x_n = \frac{A}{j\pi n}\left(1-(-1)^n\right)$$

D'où le résultat suivant :

$$x_n = 0 \text{ si } n \text{ est pair}$$

$$x_n = \frac{2A}{j\pi n} \text{ si } n \text{ est impair}$$

Ainsi, le spectre d'un signal carré contient un nombre infini de raies à f_0 , 3 f_0 , ..., (2k+1) f_0 mais avec une puissance qui diminue et tend vers zéro. Donc il existe n_0 tel que

$$x_{n_0} \leq \frac{x_1}{100} \Rightarrow n_0 \geq 100$$

On considère qu'au delà, les coefficients sont négligeables

$$n \geq n_0 \quad x_n \approx 0$$

Alors la fréquence d'échantillonnage minimale est :

$$\boxed{F_{e_{\min}} = 200 f_0}$$

B1.4. On a :

$$X(f) = \int_{-\infty}^{+\infty} x(t)\exp(-j2\pi ft)dt = \int_0^{+\infty} A\exp(-\alpha t)\exp(-j2\pi ft)dt = \frac{A}{\alpha + j2\pi f}$$

Il s'agit d'un signal déterministe à énergie finie. Son spectre (ou plus exactement sa densité spectrale d'énergie) est défini par :

$$S_x(f) = |X(f)|^2 = \frac{A^2}{\alpha^2 + 4\pi^2 f^2}$$

<u>Ce signal est donc à support spectral infini. En théorie, il n'est donc pas échantillonnable.</u>

En pratique,

$$\frac{|X(0)|}{|X(\Delta f)|} = 100 \Rightarrow \frac{|X(0)|^2}{|X(\Delta f)|^2} = \frac{\alpha^2 + 4\pi^2 \Delta f^2}{\alpha^2} = 10000$$

Soit

$$\alpha^2(10000 - 1) = 4\pi^2 \Delta f^2$$

Soit encore, en négligeant 1 devant 10000 :

$$\boxed{\Delta f \approx \frac{100\alpha}{2\pi}}$$

La fréquence d'échantillonnage minimale est donc :

$$\boxed{F_{e_{\min}} = 2\Delta f = \frac{100}{\pi}\alpha}$$

Pour couvrir une durée de $T = 4/\alpha$, en échantillonnant à cette fréquence d'échantillonnage minimale, il faut prélever N échantillons tels que

$$NT_e = T$$

avec $T_e = 1/F_{e_{\min}}$. D'où

$$\boxed{N = \frac{4}{\alpha}\frac{100}{\pi}\alpha = \frac{400}{\pi} \approx 128}$$

<u>B2. Quantification</u>

B2.1. Le rapport signal à bruit est défini en décibels par :

$$\boxed{S/B = 10 \times \log_{10}\left(\frac{\sigma_x^2}{\sigma_b^2}\right) = 20 \times \log_{10}\left(\frac{\sigma_x}{\sigma_b}\right)} \text{ en dB}$$

σ_x^2 et σ_b^2 désignant les puissances respectives du signal et du bruit.

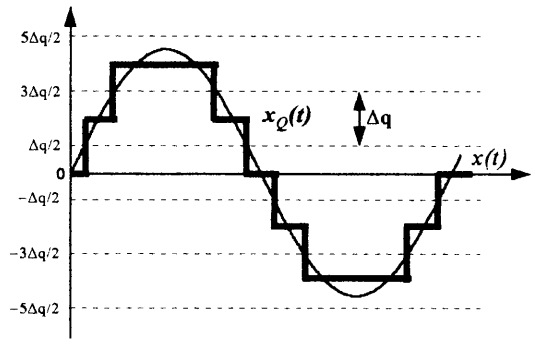
B2.2.

50 dB : plus de signal que de bruit, très peu de bruit, très bonne qualité de signal, puissance de bruit égale à 0.00001 fois celle du signal.

0 dB : autant de signal que de bruit, puissances de bruit et de signal égales.

-10 dB : plus de bruit que de signal, puissance de bruit égale à 10 fois celle du signal.

B2.3.



Bruit de quantification : $b(t) = x_Q(t) - x(t)$

D'où

$$\boxed{|b(t)| \leq \Delta q / 2}$$

B2.4. Hypothèse : $\sigma_b^2 = \dfrac{\Delta q^2}{12}$

Or, si B est la dynamique crête-à-crête du quantificateur de n bits, soit 2^n niveaux, on a :

$$\Delta q \times 2^n = B \Rightarrow \Delta q^2 = \frac{B^2}{2^{2n}}$$

et

$$\sigma_b^2 = \frac{B^2}{12 \times 2^{2n}}$$

D'où le rapport signal à bruit :

$$\boxed{S/B = 10 \times \log_{10}\left(\sigma_x^2\right) - 10 \times \log_{10}\left(\frac{B^2}{12}\right) + n \times 20 \times \log_{10}(2)}$$

$$\boxed{S/B \approx 6.02n + 10 \times \log_{10}\left(\sigma_x^2\right) - 10 \times \log_{10}\left(\frac{B^2}{12}\right)}$$

Le rapport signal à bruit, exprimé en décibels, est une fonction linéaire du nombre de bits : pour un bit de quantification supplémentaire, on gagne 6 dB dans le rapport signal à bruit.

B2.5. Pour un signal sinusoïdal d'amplitude (-A,+A), on a, par rapport aux notations de la question précédente,

$\sigma_x^2 = \dfrac{A^2}{2}$ et $B=2A$

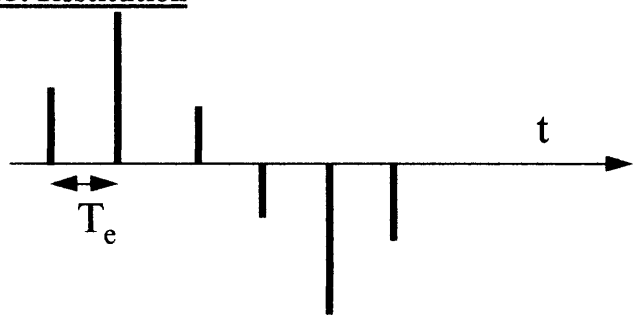
D'où

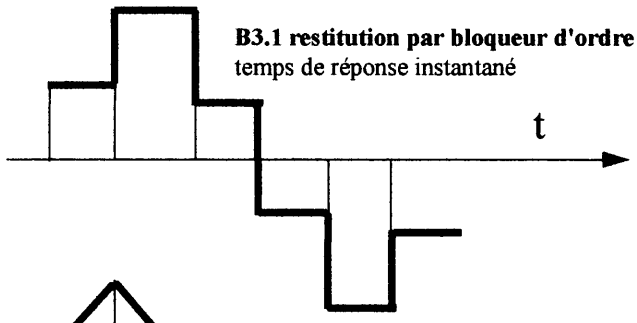
$$\boxed{S/B \approx 6.02n + 10 \times \log_{10}\left(\frac{A^2}{2}\right) - 10 \times \log_{10}\left(\frac{4A^2}{12}\right)}$$

$$\boxed{S/B \approx 6.02n + 10 \times \log_{10}\left(\frac{A^2}{2} \times \frac{3}{A^2}\right) \approx 6.02n + 10 \times \log_{10}\left(\frac{3}{2}\right) \approx 6.02n + 1.76 \quad dB}$$

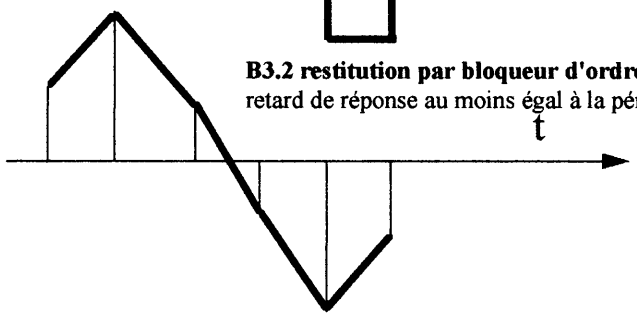
Pour *n=8* bits, on obtient un rapport signal à bruit de 49.92 dB ce qui signifie que le bruit de quantification n'est pas perceptible. Le signal en sortie du quantificateur peut être considéré comme pratiquement le même que celui en entrée. La quantification est dite de haute résolution.

B3. Restitution



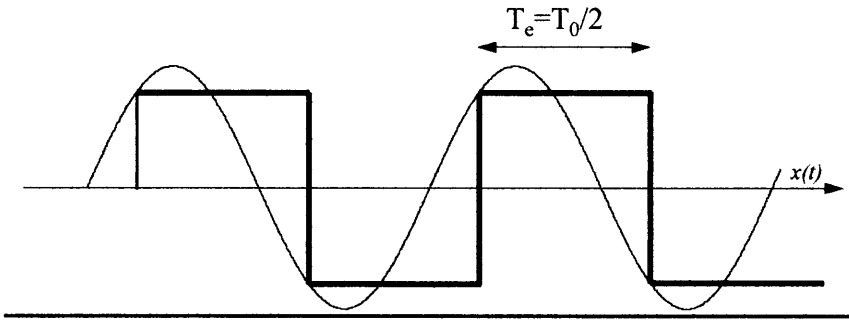


B3.1 restitution par bloqueur d'ordre 0
temps de réponse instantané



B3.2 restitution par bloqueur d'ordre 1 (interpolateur linéaire)
retard de réponse au moins égal à la période d'échantillonnage

B3.3 Restitution par bloqueur à la fréquence d'échantillonnage minimale d'un signal sinusoidal : on retrouve un signal carré, de fréquence fondamentale égale à la fréquence du signal sinusoidal.
Il ne faut donc pas travailler autour de la fréquence d'échantillonnage minimale si le système de restitution est un bloqueur. **Dans le cas de restitution par bloqueur d'ordre 0, il faut suréchantillonner.**



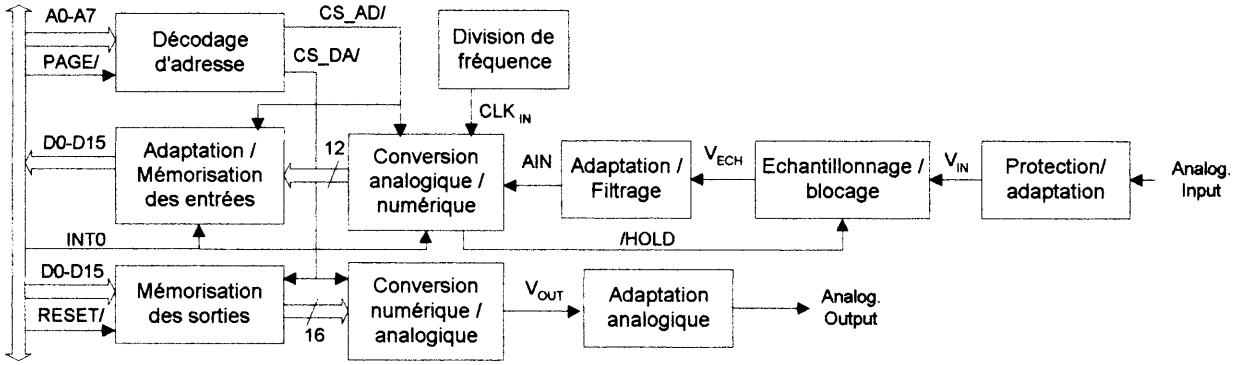
signal restitué par bloqueur d'ordre 0 (en gras) : on est loin d'un signal sinusoidal !

C . ETUDE DE LA CARTE D'ACQUISITION RESTITUTION

C.1. Analyse fonctionnelle

C.1.1.

C.1.2. Les acquisitions et les restitutions se font à l'adresse $3FF7. Le CAN et le CNA seront sélectionnés en fonction des lignes de lecture ou d'écriture. Le décodage de page fixe les adresses A8-A15.



$CS\_AD/ = PAGE/ + DMRD/ + \overline{DA7} + \overline{DA6} + \overline{DA5} + \overline{DA4} + DA3 + \overline{DA2} + \overline{DA1} + \overline{DA0}$

$CS\_DA/ = PAGE/ + DMRW/ + \overline{DA7} + \overline{DA6} + \overline{DA5} + \overline{DA4} + DA3 + \overline{DA2} + \overline{DA1} + \overline{DA0}$

C2. Catactéristiques du signal d'entrée

C.2.1. Conversion courant-tension.

C.2.2.
f_{in} = 250 kHz >> fréquence maximale des harmoniques (1KHz)

C.3. Echantillonneur-bloqueur

C.3.1. Protection contre les surtensions positives ou négatives (V_{DD} + 0,7 V et V_{SS} - 0,7 V) et adaptation d'impédance.

C.3.2. V_{OUT} / V_{IN} = 1. Les résistances internes permettant de fixer l'amplification à 2 ou -1 ne sont pas utilisées.

C.3.3. Ajustement de la compensation de la tension de décalage (V_{OUT} = 0 V pour V_{IN} = 0 V)

C.3.4. L'échantillonneur est commandé sur un niveau haut du signal /BUSY issu du CAN, ce qui correspond à l'état inoccupé pour le convertisseur. Un niveau bas (/HOLD) permet le blocage de l'échantillon pendant la conversion.

C.4. Conversion analogique-numérique

C.4.1. Filtre passe-bas et ajout d'une tension de décalage.

C.4.2.

$$\frac{A_{IN}}{V_{OUT}} = \frac{R}{R+R_5}\left[1+\frac{R_2}{R_3}\times\frac{1}{1+j\dfrac{f}{f_0}}\right]$$

$$avec \quad R = R_4 + P_3 \quad et \quad f_0 = \frac{1}{2\pi R_2 C_{14}}$$

La fréquence de coupure du filtre passe-bas du premier ordre est de 2,57 MHz, il n'intervient donc pas sur les harmoniques traités.

C.4.3.
P3 permet d'ajuster la valeur pleine échelle.
Pour une tension V_{OUT} égale à la valeur pleine échelle – 3/2LSB (+2.49817 V) le code de sortie doit osciller entre $FFE et $FFF.

C.4.4.

$$A_{IN} = \frac{R}{R+R_5}\times\frac{R_3+R_2}{R_3}V_{OUT} - \frac{R_2}{R_3}V_{REF}$$

$$A_{IN} \approx V_{OUT} + 2{,}5V$$

Une tension de décalage de 2,5V est ajoutée pour convertir des tensions bipolaires comprises entre –2,5V et +2,5V en entrée de la carte.

C.4.5. Conversion par approximations successives. Principe de la pesée avec une balance à double plateau. Des comparaisons successives sont effectuées entre la tension d'entrée et une tension délivrée par un CNA interne assurant la conversion des valeurs successives présentées, des poids forts aux poids faibles.

C.4.6. CAN à rampe ou sigma-delta pour les basses fréquences (<100 kHz) et les hautes résolutions (>12 bits), CAN parallèle ou flash pour les hautes fréquences (>1 MHz) et les moyennes résolutions (<12 bits).
Le CAN utilisé présente des performances intermédiaires (3 µs, 12 bits).

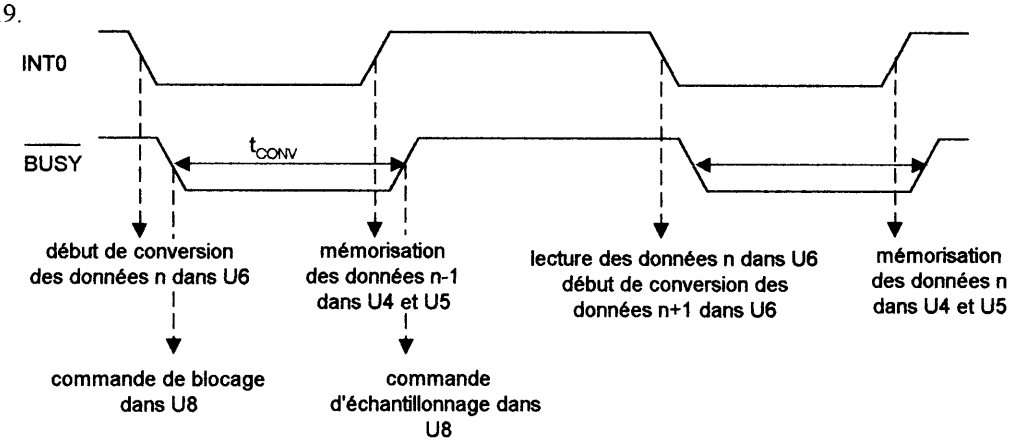
C.4.7. Référence d'horloge pour le registre à approximations successives et le multiplexeur. f_{CLKIN} = 4 MHz.

C.4.8.

$$t_{CONV} = \frac{n}{f_{CLKIN}}$$

Pour 12 bits, $t_{CONV} = \dfrac{12}{4\cdot 10^6} = 3\,\mu s$

C.4.9.



C.4.10. Durée liée aux caractéristiques du CAN, soit 170 ns (t_3).

C.4.11. Il faut suréchantilonner l'harmonique de rang 20, d'où f_{INT0} min = 2 kHz.
Pour une durée à l'état bas de 0,170 µs et un temps de conversion de 3 µs, on obtient $f_{INT0} \approx 315$ kHz.
Les documentations donnent par ailleurs une fréquence maximale d'échantillonnage de 250 kHz,
125 fois supérieure à f_{INT0} min.

C.5. Conversion numérique-analogique

C.5.1.
U1 et U2 permettent la mémorisation des sorties numériques à chaque accès à la carte.

C.5.2. Dans la fonction de transfert du CNA, l'erreur de linéarité différentielle correspond à l'écart maximal entre la valeur effective d'un quantum (tension analogique correspondant à un LSB) et sa taille idéale FSR/2^n (FSR étant la tension pleine échelle et n la résolution en bits).
Pour le CNA utilisé, l'erreur de linéarité est égale à 0,001 % de FSR, soit 0,06 mV.
Cette erreur peut être exprimée en fraction de LSB :
0,001% . 2^{16} = 0,65536 LSB.

C.5.3.

$$\frac{A_{OUT}}{V_{OUT}} = \frac{R_9}{R_9 + R_7}\left[1 + \frac{R_8}{R_{10}} \times \frac{1}{1 + j\frac{f}{f_0}}\right]$$

$$avec \quad f_0 = \frac{1}{2\pi R_8 C_{30}}$$

C.5.4. La fréquence de coupure est de 26,79 kHz. La fréquence maximale de conversion de la carte est de 250 KHz. Le filtrage passe-bas permet de restituer un signal lissé.

D. TRAITEMENT NUMERIQUE

D.1. Adressage

D.1.1. SW2-1 = 1 (fermé)
SW2-2 à SW2-8 = 0 (ouvert)

D.1.2.

Registre de commande	Adress L Register	Adress H Register	DMA Register	INTERRUP Register	STATUS Register
Adresse	$D0000	$D0001	$D0002	$D0003	$D0004

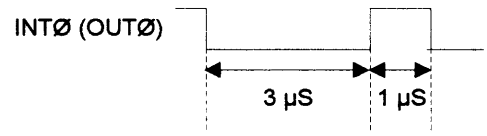
Mémoire programme : $D8000-$DBFFF (DMA Register = $7E ou $7D ou $7B)
Mémoire données : $D8000-$DBFFF (DMA Register = $77 ou $6F)

D.1.3.
Demande d'accès aux poids faibles de la mémoire des données (DMA Register = $77)
FAIRE TANT QUE <bus non disponible>
    Lecture du registre d'état (STATUS Register)
    Test disponibilité du bus (bit D1)
FIN FAIRE
Commande d'arrêt du DSP (DMA Register = $37)
Lecture de l'octet
Commande de libération du DSP (DMA Register = $FF)

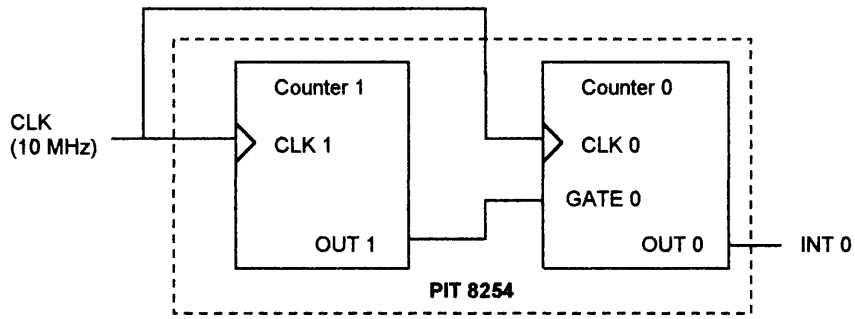
D.2. Horloges programmables

D.2.1.
Counter 0 = 60 (6 µs)
Control Word Register = $16 (mode 3)

D.2.2.
Durée maximale du traitement de l'interruption : 4µs



D.2.3.



D.2.4.
Counter 0 = 30 (3 µs)
Control Word Register = $12 (mode 1)

Counter 1 = 40 (4µs)
Control Word Register = $54 (mode 2)

D.3. Programmes

D.3.1. f_E = 153,85 kHz

D.3.2.
Cntl_Word = b#00010110 : le registre de contrôle du compteur 0 est programmé pour un fonctionnement en mode 8 bits, en mode 3 (signaux de rapport cyclique ½) et pour un comptage binaire.
Timer_Count = 65 : le compteur 0 est chargé avec la valeur 65, ce qui correspond à une période de 6,5 µs pour une fréquence d'entrée de 10 Mhz.

D.3.3.
La routine de traitement de l'interruption doit être insérée entre la lecture de la donnée convertie par le CAN et l'écriture dans le CNA, soit entre les lignes 24 et 25.

D.3.4.
Les DSP utilisent une unité arithmétique et logique de type **MAC** (Multiplicateur-Accumulateur) permettant d'effectuer simultanément les deux opérations élémentaires présentes dans les algorithmes de traitement du signal.
Plusieurs unités de **mémoires distinctes** (bus d'adresses et de données distincts) sont intégrées, ce qui permet la gestion indépendante des valeurs d'entrée, de sortie et des coefficients d'une équation de récurrence, ou encore de séparer la mémoire programme de la mémoire des données.
Le format des **registres internes** est adapté au traitement des nombres réels (n bits pour la partie entière, m bits pour la partie fractionnaire).

D.3.5.

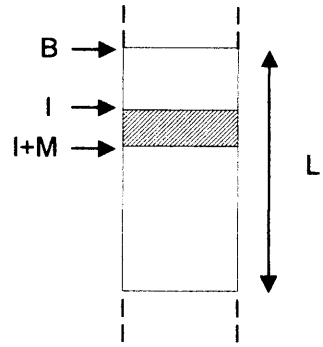
- Opérations sur des mots de 16 bits en virgule fixe.
- Acquisition des 2 opérandes, multiplication et addition en un cycle d'horloge.
- Séparation des mémoires programme et données (séparation des bus).
- Tampon circulaire gérée en interne.
- Mémoire cache.

D.3.6. Les deux générateurs d'adresse indépendants permettent d'adresser simultanément les mémoires de données et de programmes : DAG 1 pour les données, DAG2 pour les données ou le programme.
DAG1 et DAG2 possèdent 4 jeux de 3 registres de 14 bits :

- I0 à I3 (DAG1) et I4 à I7 (DAG2) sont des registres d'index qui contiennent l'adresse courante de la mémoire à consulter ;
- M0 à M3 (DAG1) et M4 à M7 (DAG2) sont des registres de modification qui contiennent la valeur du post-incrément du registre d'index ;
- L0 à L3 (DAG1) et L4 à L7 (DAG2) sont des registres de longueur permettant un adressage circulaire.

Pour un adressage direct en mémoire des données, les registres des DAG ne sont pas utilisés.
Pour un adressage indirect en mémoire des données ou des programmes, un des jeux I-M-L est utilisé.
Lorsque ces 3 registres sont chargés, la prochaine adresse de la donnée ou de l'instruction est calculée par :
I ← (I+M) modulo L (avec L=0 pour un adressage linéaire).

Un tampon circulaire est une zone mémoire définie par son adresse de base B, sa taille L, l'adresse courante de la donnée I et le format des données adressées M. Lorsque la fin du tampon est atteinte l'adresse de début est rechargée.



La prochaine adresse est calculée par (I+M-B) modulo L + B

D.3.7. La fonction R=MR+X*Y réalise dans le MAC en un cycle d'horloge la multiplication des opérandes X et Y et l'addition du résultat au registre MR.
L'opérande X est un registre 16 bits (MX0, MX1, AR, MR0, MR1, MR2, SR0 ou SR1).
L'opérande Y est un registre 16 bits (MY0, MY1 ou MF).
Le résultat est un registre 40 bits (MR ou MF).

D.3.8

La fréquence d'échantillonnage étant de 2 kHz, la période d'échantillonnage correspondante est égale à 0.5 ms soit 5000 fois 100 nanosecondes : **le compteur doit compter jusqu'à 5000**.
Or 5000 ne tient pas sur 8 bits : il faut donc **l'envoyer en deux fois**, dans un premier temps les 8 bits de poids faibles puis dans un deuxième temps les 8 bits de poids forts.
En binaire, 5000 s'écrit : 0001001100001000
soit : $5000 = 1x2^{12} + 1x2^{9} + 1x2^{8} + 1x2^{7} + 1x2^{3}$
D'où les modifications :
ligne 2 :
. const Cntl_Word = b#00110110; {compteur 0,8 bits poids faibles puis 8 bits poids forts, mode 3, binaire}
ligne 3 :
. const Timer_Count1= b#00001000; { poids faibles }

ligne 3 bis (à rajouter) :
. const Timer_Count2 = b#00010011; { poids forts }
ligne 21 :
ax0=Timer_Count1 ; {pour écrire dans Timer0, nécessité de passer par des registres }
{ ax0 n'est qu'un exemple, on peut employer n'importe quel registre }
ligne 22 :
dm(Timer0)=ax0; {envoi des 8 bits poids faibles }
ligne 22 bis (à rajouter) :
ax0=Timer_Count2;
ligne 22 ter (à rajouter) :
dm(Timer0)=ax0; {envoi des 8 bits poids forts }

D.3.9

L'équation de récurrence représente p opérations de multiplications, additions et accumulations (MAC).
Ici $p=154$.
L'opération de MAC se fait en un seul cycle machine, soit 100 nanosecondes.
Donc le calcul d'un échantillon filtré nécessite $px100\ ns$ soit, dans notre cas, **15.4 microsecondes**.
Or la période d'échantillonnage (temps entre deux échantillons) est de :

$$T_e = \frac{1}{2000} = 500\,\mu s >> 15.4\ \mu s$$

Entre deux échantillons, l'opération de filtrage peut s'effectuer facilement, le temps réel est donc possible.

D.3.10 et D.3.11 : la programmation

Ce qu'il est important d'avoir compris :

- la routine de filtrage s'insère dans le sous-programme d'interruption soit :

ligne 24 :
in_out : ay0=dm(DXM2106_1);
ligne 24 bis (à rajouter) :
call filt;
ligne 25 :
dm(DXM2106_1)=mr1; {résultat du filtrage dans un des registres du MAC }
ligne 26 :
rti;

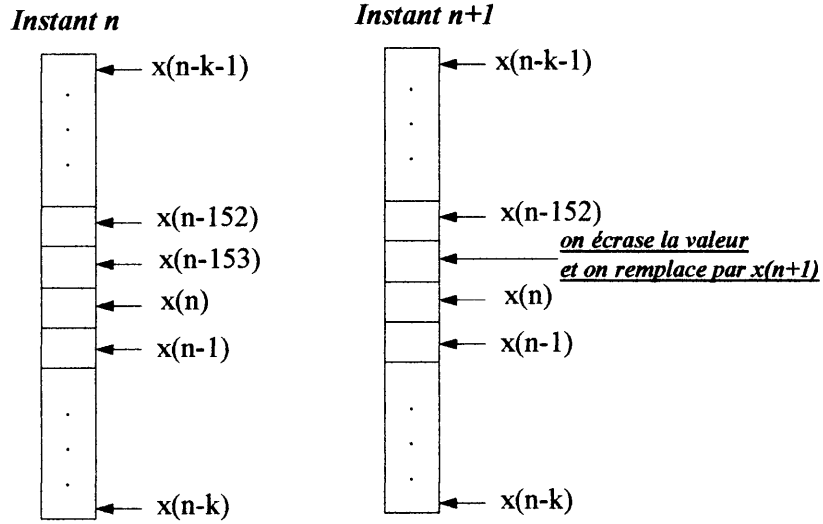
- les coefficients du filtre doivent être mis en mémoire de données pour permettre à l'opération de MAC de s'exécuter en un cycle machine (et utiliser les deux pointeurs d'adresses (mémoire de programme et mémoire de données)). Donc, dans les déclarations, on doit trouver :
  . var/dm coefficients[154];
- de même, le vecteur d'échantillons du signal doit être en mémoire de programme car ainsi, le DSP sait pointer en un cycle machine à la fois sur le coefficient (en mémoire donnée) et sur l'échantillon (en mémoire programme) et faire l'opération de multiplication et accumulation. Remarque : rien n'empêche de mettre les coefficients en mémoire programme et les échantillons du signal en mémoire donnée, l'essentiel est d'avoir ces deux tableaux dans des zones mémoires différentes, l'un en mémoire donnée, l'autre en mémoire programme.
  MR=MR+MX0*MY0 (ss), MX0=DM(I0,M1),MY0=PM(I4,M5);
  {mr=mr+coef*sample , get next coef , get next sample}
- penser à mettre le tableau d'échantillons contenant les 154 derniers échantillons dans un **buffer circulaire**.

Ainsi, avec une gestion du pointeur d'adresse (en jouant sur les registres modificateurs), on peut écrire la routine de filtrage.

Le buffer circulaire assure que dans l'écriture du résultat du filtrage, on ne se préoccupe que de la position de départ (sur l'échantillon qui vient d'être acquis à l'instant *n*, *x(n)*) et on lit ensuite avec un registre modificateur M5=1 pour faire les 154 opérations de MAC. En entrée de la routine, il suffit alors d'écrire l'échantillon (contenu dans ay0 - voir ligne 24 - :

PM(I4,M4)= ay0;

avec M4=0. On reste ainsi pointé sur *x(n)*.



On trouvera ci-après un programme répondant au cahier des charges. On distingue le programme principal effectuant l'échantillonnage et faisant appel à la routine de filtrage (module sampling) et le sous-programme de filtrage (module filtrage).

{ LOGABEX Toulouse 2086.91 / Octobre 1991
********** Module DXM2106 **********
This module performs acquisitions in a DXM2106 card (12 bits)
and restore the data to the analog output (16 bits).
Only the most significant bits are used.
The DXM2106 card works in interrupt mode.
The interruption is provided by the Timer of the DXM2106 card. }

{Programme modifie pour fonctionner à 2 kHz et effectuer le filtrage }

```
1       . Module        sampling ;

2       . const         Cntl_Word=b#00110110 ;          {écriture poids faible, poids forts}
3       . const         Timer_Count1=b #00001000 ;      {poids faibles}
3 bis   . const         Timer_Count2=b #00010011 ;      {poids forts }
4       . const         Hard_Mask=1 ;

5       . port          Cntl_Irq ;                      {IRQ Hard control}
6       . port          Cntl_Tmr ;                      {Timer control}
7       . port          Timer0 ;                        {Timer0 access}
8       . port          DXM2106_1 ;                     {Base address of DXM2106 card}

8 bis   . external      filt ;                          {routine de filtrage dans autre fichier}

        {Interrupt Vectors}
9                       jump in_out ;
10                      rti ;
11                      rti ;
12                      rti ;
        {Start of Code}
13      start :         icntl=b #10001 ;                {interruption soft configuration}
```

```
                imask=b#1111 ;              {interruption soft activation}
                ay0=Hard_Mask ;             {interruption hard activation}
                dm(Cntl_Irq)=ay0 ;
                call timer1 ;

loop :          jump loop ;

timer1 :        ax0=Cntl_Word ;             {Timer configuration}
                dm(Cntl_Tmr)=ax0 ;
                ax0=Timer_Count1;
                dm(Timer0)=ax0 ;
                ax0=Timer_Count2;
                dm(Timer0)=ax0 ;
                rts ;

in_out :        ay0=dm(DXM2106_1) ;
                call filt ;                 {appel à la routine de filtrage}
                dm(DXM2106_1)=mr1 ;
                rti ;
.endmod.
```

```
.module filtrage ;
{en entrée : ay0 contient le dernier échantillon acquis, x(n) }
{en sortie : mr1 contient le résultat du filtrage }
.CONST  Length=154;           { Length of impulse response }
.VAR/DM Coefficients[Length];

.INIT  Coefficients: 52,51,47,39,27,14,-1, -17,-32,-46,-57,-64,-67,-65, -58,-47,-32,-13,7,27,46,
63,76,85,87,83,73,57, 36,12,-15,-42,-68,-90,-106, -116,-118,-111,-96,-73,-43,-8,
30,68,104,134,157,170,171,
160,136,100,53,-2,-62,-123, -181,-231,-270,-294,-298,-280,-238, -171,-81,32,164,311,468,630,
790,942,1080,1198,1291,1356,1388, 1388,1356,1291,1198,1080,942,790, 630,468,311,164,32,-81,-
171,
-238,-280,-298,-294,-270,-231,-181, -123,-62,-2,53,100,136,160, 171,170,157,134,104,68,30,
-8,-43,-73,-96,-111,-118,-116,-106,-90,-68,-42,-15,12,36,57,73,83,87,85,76,63,46,27,7,-13,-32,-47,-58,
-65,-67,-64,-57,-46,-32,-17,-1,14,27,39,47,51,52;

.var/pm/circ out_data[length] ;
.INIT   out_data:     0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                      0,0,0,0,0;

.var/dm pointeur[1] ;
.init pointeur :^out_data ;
.entry filt ;

filt :    m1=1 ;          {registre modificateur des coefficients}
```

```
        m5=1 ;          {registre modificateur du tableau de sortie}
        m4=0 ;          {registre modificateur du tableau de sortie}
        l0=0 ;          {registre longueur des coefficients non circulaire}
        l4=length ;     {registre longueur du tableau de sortie CIRCULAIRE}
        i0=^Coefficients ;{registre pointeur du tableau de coefficients}
        i4=dm(pointeur) {registre pointeur du tableau de sortie : position sauvegardée dans
« pointeur »}

        pm(i4,m4)=ay0 ;     { échantillon x(n) lu sur ay0 (cf routine d'interruption}
        cntr=length-2 ;
        mr=0,mx0=dm(i0,m1) ,my0=pm(i4,m5) ;   {initialisation mr, mx0=coef, my0= échantillon}
        do conv until ce ;
conv :  mr=mr+mx0*my0 (ss), mx0=dm(i0,m1),my0=pm(i4,m5) ;
        { mr=mr + coef* echantillon, prochain coefficient, prochain echantillon}

        mr=mr+mx0*my0 (ss), mx0=dm(i0,m1),my0=pm(i4,m4) ;
{tableau de sortie reste pointé sur dernier échantillon de la convolution (x(n-153)}
        dm(pointeur)=i4 ;{sauvegarde de cette position : on placera x(n+1) au prochain appel de
filtrage}
        mr=mr+mx0*my0 (ss) ;

        sr=lshift mr1 by 1 (hi) ;
        sr=sr or lshift mr0 by 1 (lo) ;     {récupération de la valeur en sortie du MAC}
        mr1=sr1 ;

        rts ;

.endmod ;
```