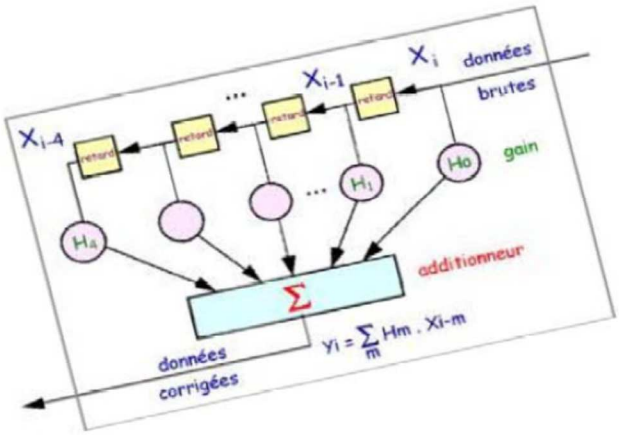
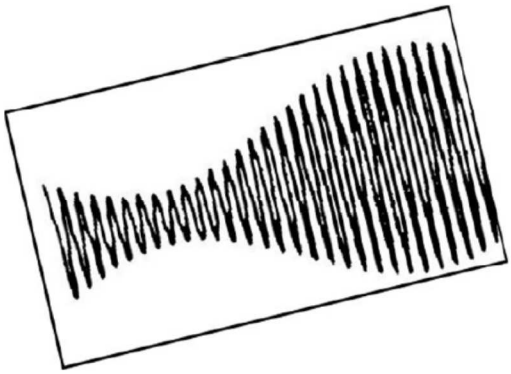
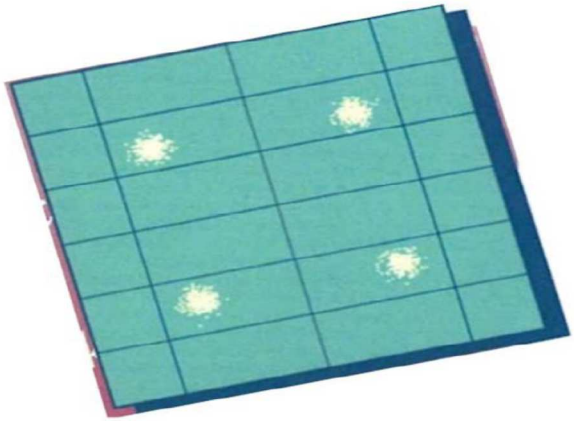




TP
Modulations numériques

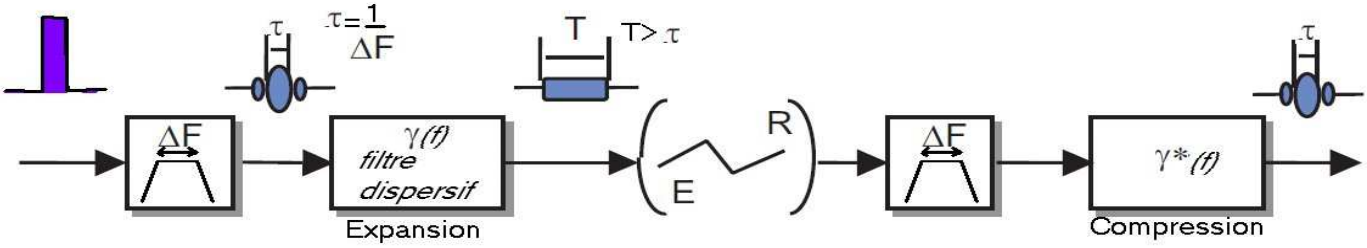




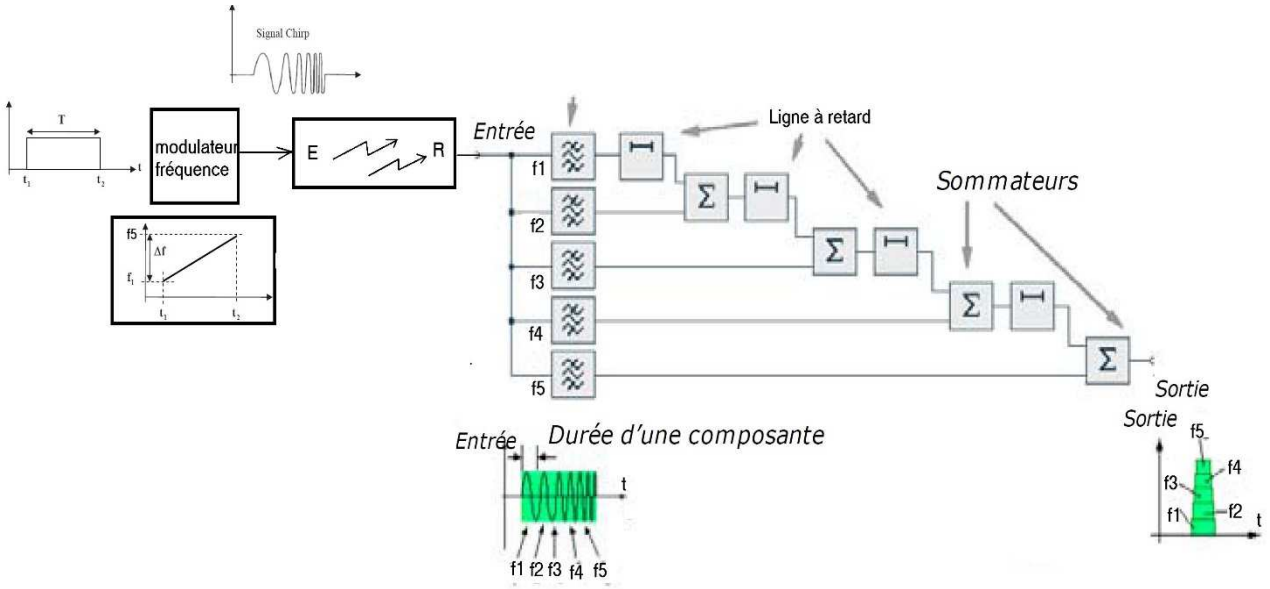
1.1) La modulation BPSK et le code de Barker

1.1.1) La technique de compression d'impulsion

En technique radar, Il existe différentes techniques de compression d'impulsion le concept général est le suivant :



Soit un signal court de bande ΔF et de durée voisine de 1/ΔF. Il est soumis à l'action d'un filtre « dispersif » γ (f), qui n'apporte pas de modifications aux amplitudes du spectre du signal ([γ (f)] = 1), mais a comme propriété de déphaser différemment les différentes fréquences du spectre du signal émis, déformation qui a pour effet d'allonger ce signal. En effet, l'action du filtre est d'ajouter de l'information au signal donc d'augmenter le nombre d'échantillons indépendants composant ce signal. A la réception, après passage dans un filtre de bande ΔF, qui fixe la bande de bruit accompagnant le signal, il ne reste plus qu'à faire passer le signal dans un filtre complémentaire **γ *(f),** qui aura pour effet de compenser les déphasages apportés par le filtre **γ (f).**
Une modulation de fréquence linéaire permet aussi de rajouter de l'information au signal afin d'étendre ce signal et de le compresser par la suite pour obtenir une brève impulsion en sortie.



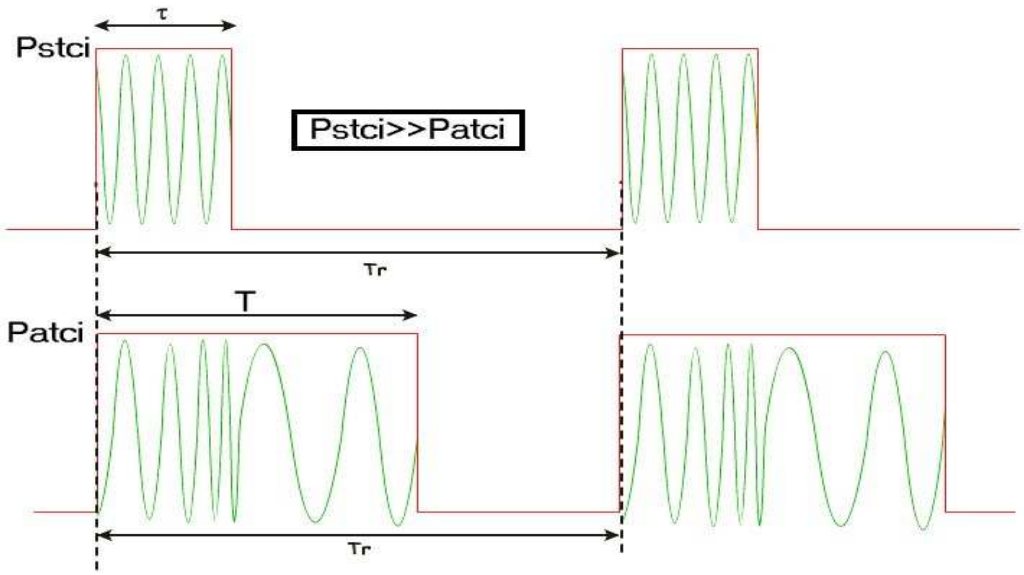
La modulation de phase du type BPSK est aussi utilisée en technique de compression d'impulsion RADAR, ce code de Barker en est un exemple. Ce code possède différentes séquences qui sont données ci-dessous :

N	Séquence de phase
5	0 0 0 π 0
7	0 0 0 π π 0 π
11	0 0 0 π π π 0 π π 0 π
13	0 0 0 0 0 π π 0 0 π 0 π 0

Pstci : puissance émise sans technique de compression d'impulsion
Pstci : puissance émise avec technique de compression d'impulsion

1.1.2) Etude théorique

Soient les chronogrammes suivants :



1.1.2.1) En sachant que l'énergie transmise sur le temps Tr est constant (portée identique) avec ou sans technique de compression d'impulsion, donnez la relation qui lie τ, T , Pstci , Patci.

1.1.2.2) Quel est l'intérêt d'utiliser la technique de compression d'impulsion sur les étages de sortie de la chaine d'émission .

1.1.2.3) En fonction de la vitesse de propagation de la lumière **c**, de la durée $\boldsymbol{\tau}$ de l'impulsion émise et de la période de répétition **Tr**, en déduire la distance maximale de détection Dmax et la distance minimale de détection Dmin.

D'un point de vue comportemental la technique de compression d'impulsion revient à émettre une impulsion brève de durée et de puissance crête Pscti. Le pouvoir discriminateur en distance est la distance minimale en deçà de laquelle le radar ne peut distinguer deux cibles dans la même direction. Le pouvoir discriminateur en distance est aussi égal à Dmin.

1.1.2.4) Quel est donc l'intérêt d'utiliser une impulsion d'émission très brève.

1.1.2.5) Complétez document réponse n°1, le signal BPSK émis avec le code de BARKER (N=7).

'0' aucun déphasage du signal émis avec la fréquence porteuse de référence
'π ' déphasage de π du signal émis avec la fréquence porteuse de référence

1.1.2.6) En raisonnant en terme de niveau logique, en déduire graphiquement document réponse N°2, la fonction d'autocorrélation s(τ) du signal Sbarker.

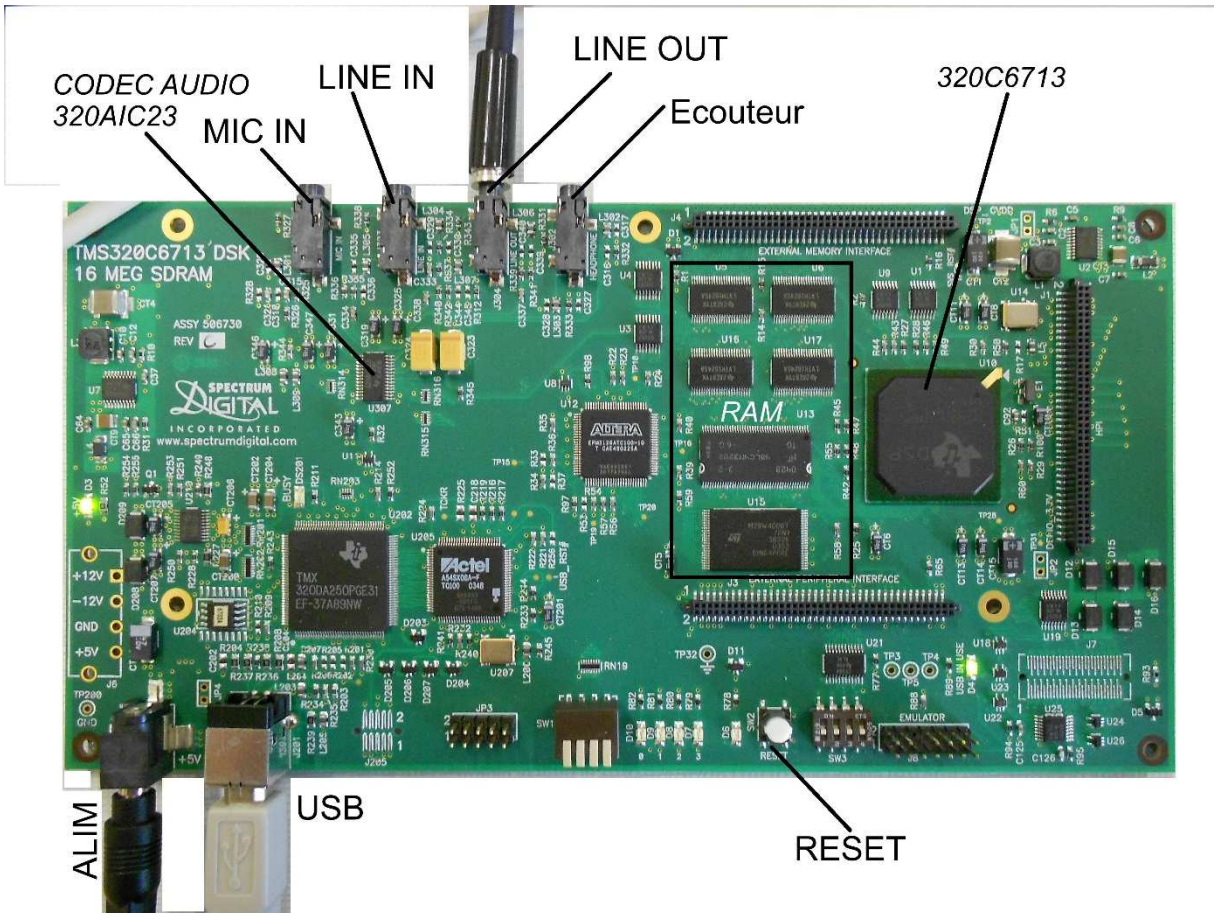
$$S(\tau) = \int_{-\infty}^{\infty} sbarker(\tau) \times sbarker(t-\tau) d\tau$$

On pourra associer les coefficients suivants pour chaque échantillon de phase
'0' coefficient de +1
'π ' coefficient de -1

1.1.2.7) Au vue des résultats obtenus pour s(τ), les échantillons d'information de phase rajoutés doivent-ils être indépendants ou dépendants entre eux, justifiez votre réponse.

1.1.3) Etude expérimentale

La modulation numérique sera réalisée grâce à un processeur spécialisé dans le traitement numérique du signal DSP qui est implanté sur une carte de développement de chez Spectrum Digital Incorporated. Le Processeur est du type TMS320C6713 qui un processeur de calcul à virgule flottante. D'un point de vue matériel la carte est représentée ci-dessous :



Le processeur TMS320C6713 dispose d'une liaison série MCBSP lui permettant de piloter un CODEC AUDIO le 320AIC23. D'un point de vue logiciel, Spectrum Digital Incorporated nous fournit des librairies et des fonctions de haut niveau afin de réaliser simplement des applications dans le traitement numérique audio. Les éléments logiciels variables, constantes, librairies, fichier d'en tête et fonctions de haut niveau sont donnés dans le dossier annexe 1. On souhaite établir un signal bpsk modulé par le code de Barker. Pour simplifier l'étude le signal modulé bpsk sera généré continuellement.

1.1.3.1) Branchez votre carte LINE OUT, USB, ALIM. La suite logicielle est appelée code composer studio. Lancez le logiciel code composer studio et ouvrir le projet bpskint.pjt. Complétez le programme document réponse N°3 ainsi que le fichier bpskint.c. Compilez, chargez et testez votre programme.
1.1.3.2) Relevez le signal bpsk modulé par la séquence de Barker (faire apparaitre cette séquence sur le chronogramme).
1.1.3.3) Relevez le débit binaire, le spectre du signal modulé BPSK (module FFT), la bande passante, l'éfficacité spectrale et interprétez les résultats obtenus.

La modulation QPSK est l'une des modulations numériques la plus utilisée de fait de sa robustesse.
Elle est très utilisée dans le domaine des télécommunications satellites

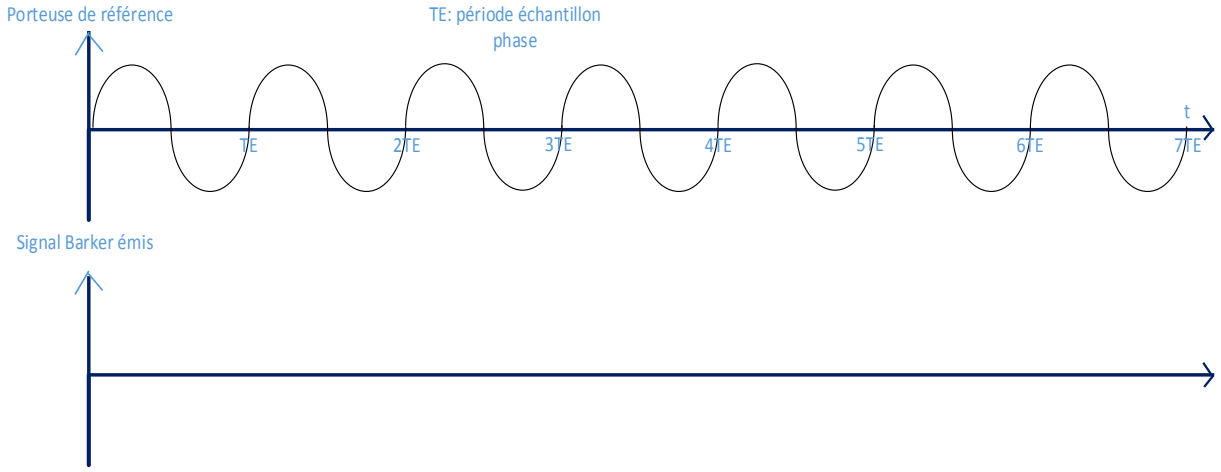
1.2) La modulation QPSK ou QAM4

On souhaite coder cette modulation QPSK par la séquence binaire suivante :
short bits[12]={0,0,0,1,1,0,1,1,0,0,0,1};
1.2.1) Ouvrir le projet qpskint.pjt et complétez la fonction interrupt void c_int11(), afin de générer cette modulation qpsk. Suivant la valeur du dibit de la séquence on aura un offset de phase. Cet offset de phase nous permettra au démarrage de chaque dibit de démarrer la conversion du premier échantillon à un indice du tableau sinetable égal à l'offset trouvé.
1.2.2) Relevez le signal qpsk modulé par la séquence suivante (faire apparaitre cette séquence sur le chronogramme).
1.2.3) Relevez le débit binaire, le spectre du signal modulé qpsk (module FFT), la bande passante, l'efficacité spectrale et interprétez les résultats obtenus.

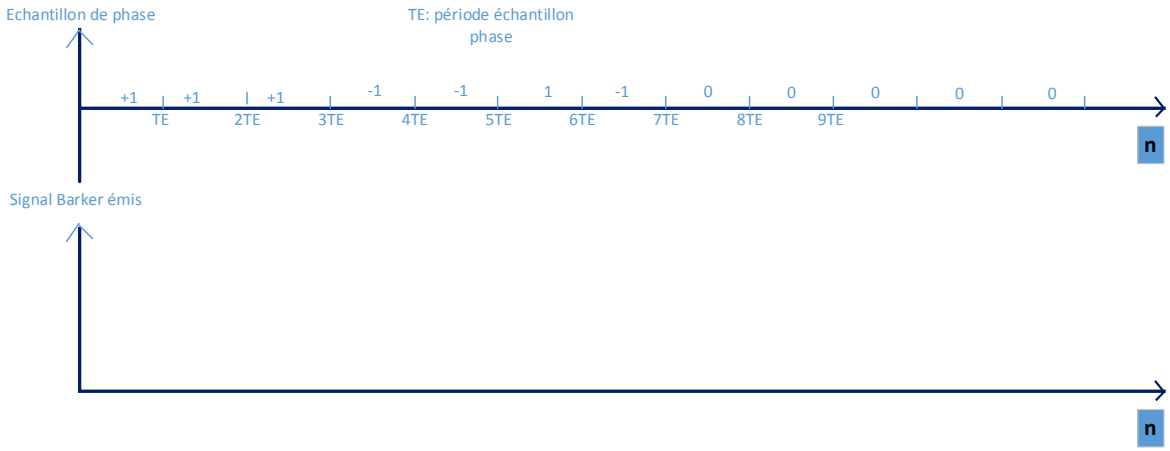
1.3) La modulation QAM

On donne en annexe 2 le programme qui nous permet de générer cette modulation
1.3.1) Quel type de modulation numérique obtient-on ?
1.3.2) Complétez la constellation de cette modulation document réponse N°4.
1.3.3) Ouvrir le projet qamint.pjt, compilez et chargez le programme dans la cible.
1.3.4) Relevez le signal qam modulé par la séquence binaire (faire apparaitre cette séquence sur le chronogramme).
1.2.3) Relevez le débit binaire, le spectre du signal modulé qam (module FFT), la bande passante, l'éfficacité spectrale et interprétez les résultats obtenus.

Document réponse N°1



Document réponse N°2



Document réponse N°3

```
/*
 *  Copyright 2002 by Spectrum Digital Incorporated.
 *  All rights reserved. Property of Spectrum Digital Incorporated.
 */

/*
 *  ======== tone.c ========
 *
 *  This example uses the AIC23 codec module of the 6713 DSK Board Support
 *  Library to generate a 1KHz sine wave on the audio outputs for 5 seconds.
 *  The sine wave data is pre-calculated in an array called sinetable.  The
 *  codec operates at 48KHz by default.  Since the sine wave table has 48
 *  entries per period, each pass through the inner loop takes 1 millisecond.
 *  5000 passes through the inner loop takes 5 seconds.
 *
 *  Please see the 6713 DSK help file under Software/Examples for more
 *  detailed information.
 */

/*
 *  DSP/BIOS is configured using the DSP/BIOS configuration tool.  Settings
 *  for this example are stored in a configuration file called tone.cdb.  At
 *  compile time, Code Composer will auto-generate DSP/BIOS related files
 *  based on these settings.  A header file called tonecfg.h contains the
 *  results of the autogeneration and must be included for proper operation.
 *  The name of the file is taken from tone.cdb and adding cfg.h.
 */
//#include "tonecfg.h"

/*
 *  The 6713 DSK Board Support Library is divided into several modules, each
 *  of which has its own include file.  The file dsk6713.h must be included
 *  in every program that uses the BSL.  This example also includes
 *  dsk6713_aic23.h because it uses the AIC23 codec module.
 */
#include "dsk6713.h"
#include "dsk6713_aic23.h"

/* Length of sine wave table */
#define SINE_TABLE_SIZE   48
#define MAX_DATA_LENGTH 10
short bits[7]={0,1,0,_____________};// sequence de Barker
short  encSeqNum=0; // indice de la sequence de Barker
short  encSymbolVal=0; // indice du tableau sinus
Uint32 fs=DSK6713_AIC23_FREQ_8KHZ; //set sampling rate

/* Pre-generated sine wave data, 16-bit signed samples */
Int16 sinetable[SINE_TABLE_SIZE] = {
   0x0000, 0x10b4, 0x2120, 0x30fb, 0x3fff, 0x4dea, 0x5a81, 0x658b,
```

```
   0x6ed8, 0x763f, 0x7ba1, 0x7ee5, 0x7ffd, 0x7ee5, 0x7ba1, 0x76ef,
   0x6ed8, 0x658b, 0x5a81, 0x4dea, 0x3fff, 0x30fb, 0x2120, 0x10b4,
   0x0000, 0xef4c, 0xdee0, 0xcf06, 0xc002, 0xb216, 0xa57f, 0x9a75,
   0x9128, 0x89c1, 0x845f, 0x811b, 0x8002, 0x811b, 0x845f, 0x89c1,
   0x9128, 0x9a76, 0xa57f, 0xb216, 0xc002, 0xcf06, 0xdee0, 0xef4c
};


/*
 *  main() - Main code routine, initializes BSL and generates tone
 */

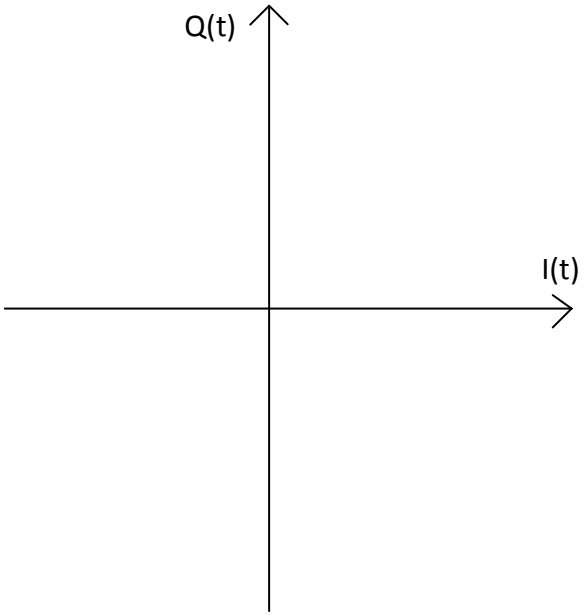
interrupt void c_int11()          //interrupt service routine
{
 int i;
 short j=0;

 Int16 outval=1;
 if (encSymbolVal<48) _____________________________________________
 else {  encSymbolVal=0;
                  encSeqNum++;
                  if (encSeqNum_____________________) encSeqNum=0;
  }
 if (bits[encSeqNum]==0)outval=1;
      else outval=-1;
 output_sample(_____________________________________________________________);//output next sample
}


void main()
{
   DSK6713_AIC23_CodecHandle hCodec;
   Int16 msec, sample=0,y[2],i;
   short outval=1;
    comm_intr();
   while(1);


   }
```

Document réponse N°4



Annexe N°1

/* Fonction permettant de configurer et d'initialiser les différents registres du DSP. Mise en place de l'interruption interne du vecteur INT11 du module série MCBSP suite à la conversion numérique analogique d'un échantillon */

```
void comm_intr() ;
```

/* Conversion numérique analogique de l'échantillon out_data et lancement de l'interruption. L'échantillon analogique est disponible sur la sortie ligne de la carte DSP. L'interruption sera généré à la fréquence fs */

```
void output_sample(int out_data)
```

/* Fonction d'interruption liée au vecteur d'interruption INT11. Cette fonction sera générée à chaque interruption de vecteur INT11, donc à chaque conversion d'un échantillon à la fréquence fs.

```
interrupt void c_int11()          //interrupt service routine
```

/*Fichier d'en tête et constantes à fournir pour toutes les déclarations utiles au CODEC et au module série du DSP*/

```
#include "dsk6713.h" // déclaration pour le TMS320C6713
#include "dsk6713_aic23.h"// déclaration pour le CODEC AUDIO aic23
fs=DSK6713_AIC23_FREQ_8KHZ; /* constante définissant la fréquence d'échantillonnage pour le CODEC*/
```

Faire calculez la résolution en portée et conclure sur l'intéret
Complétez les chronos modu du code de Barker
Faire la fonction d'autocrrélation et conclure

Annexe N°2

```
/*
 *  Copyright 2002 by Spectrum Digital Incorporated.
 *  All rights reserved. Property of Spectrum Digital Incorporated.
 */

/*
 *  ======== tone.c ========
 *
 *  This example uses the AIC23 codec module of the 6713 DSK Board Support
 *  Library to generate a 1KHz sine wave on the audio outputs for 5 seconds.
 *  The sine wave data is pre-calculated in an array called sinetable.  The
 *  codec operates at 48KHz by default.  Since the sine wave table has 48
 *  entries per period, each pass through the inner loop takes 1 millisecond.
 *  5000 passes through the inner loop takes 5 seconds.
 *
 *  Please see the 6713 DSK help file under Software/Examples for more
 *  detailed information.
 */

/*
 *  DSP/BIOS is configured using the DSP/BIOS configuration tool.  Settings
 *  for this example are stored in a configuration file called tone.cdb.  At
 *  compile time, Code Composer will auto-generate DSP/BIOS related files
 *  based on these settings.  A header file called tonecfg.h contains the
 *  results of the autogeneration and must be included for proper operation.
 *  The name of the file is taken from tone.cdb and adding cfg.h.
 */
//#include "tonecfg.h"

/*
 *  The 6713 DSK Board Support Library is divided into several modules, each
 *  of which has its own include file.  The file dsk6713.h must be included
 *  in every program that uses the BSL.  This example also includes
 *  dsk6713_aic23.h because it uses the AIC23 codec module.
 */
#include "dsk6713.h"
#include "dsk6713_aic23.h"

/* Length of sine wave table */
#define SINE_TABLE_SIZE   48
#define MAX_DATA_LENGTH 10
short bits[24]={0,0,0,0,0,1,0,1,0,0,1,1,1,0,0,1,0,1,1,1,0,1,1,1};
short encSeqNum=0,facteur=1;
short encSymbolVal=0,indexsin,offset;
Uint32 fs=DSK6713_AIC23_FREQ_8KHZ; //set sampling rate
/* Pre-generated sine wave data, 16-bit signed samples */
Int16 sinetable[SINE_TABLE_SIZE] = {
   0x0000, 0x10b4, 0x2120, 0x30fb, 0x3fff, 0x4dea, 0x5a81, 0x658b,
   0x6ed8, 0x763f, 0x7ba1, 0x7ee5, 0x7ffd, 0x7ee5, 0x7ba1, 0x76ef,
```

```
   0x6ed8, 0x658b, 0x5a81, 0x4dea, 0x3fff, 0x30fb, 0x2120, 0x10b4,
   0x0000, 0xef4c, 0xdee0, 0xcf06, 0xc002, 0xb216, 0xa57f, 0x9a75,
   0x9128, 0x89c1, 0x845f, 0x811b, 0x8002, 0x811b, 0x845f, 0x89c1,
   0x9128, 0x9a76, 0xa57f, 0xb216, 0xc002, 0xcf06, 0xdee0, 0xef4c
};


/*
 *  main() - Main code routine, initializes BSL and generates tone
 */

interrupt void c_int11()          //interrupt service routine
{
 int i;
 short j=0;
  Int16 outval=1;
  if ((bits[encSeqNum]==0)&&(bits[encSeqNum+1]==0)&&(bits[encSeqNum+2]==0)){ offset=0;facteur=2;}
 else if ((bits[encSeqNum]==0)&&(bits[encSeqNum+1]==0)&&(bits[encSeqNum+2]==1)){offset=0;facteur=1;}
 else if ((bits[encSeqNum]==0)&&(bits[encSeqNum+1]==1)&&(bits[encSeqNum+2]==0)){offset=12;facteur=2;}
 else if ((bits[encSeqNum]==0)&&(bits[encSeqNum+1]==1)&&(bits[encSeqNum+2]==1)){offset=12;facteur=1;}
 else if ((bits[encSeqNum]==1)&&(bits[encSeqNum+1]==0)&&(bits[encSeqNum+2]==0)){offset=24;facteur=2;}
else if ((bits[encSeqNum]==1)&&(bits[encSeqNum+1]==0)&&(bits[encSeqNum+2]==1)){offset=24;facteur=1;}
else if ((bits[encSeqNum]==1)&&(bits[encSeqNum+1]==1)&&(bits[encSeqNum+2]==0)){offset=36;facteur=2;}
else if ((bits[encSeqNum]==1)&&(bits[encSeqNum+1]==1)&&(bits[encSeqNum+2]==1)){offset=36;facteur=1;}

 indexsin=offset+encSymbolVal;
 if (indexsin>47) indexsin=((indexsin)%48);
   output_sample((sinetable[indexsin]/facteur));//output next sample
 if (encSymbolVal<48) {
                     output_sample((sinetable[indexsin]/facteur));//output next sample
                    encSymbolVal++;
                    }
 else    {
                    encSymbolVal=0;
                    encSeqNum=encSeqNum+3;// triibit = 3bits
                    output_sample(0);
                    if (encSeqNum>23) encSeqNum=0;
                    }


}

void main()
{
   DSK6713_AIC23_CodecHandle hCodec;
   Int16 msec, sample=0,y[2],i;
   short outval=1;
    comm_intr();
   while(1);
```

```
  }
//You have to define macro name CHIP_6424 for your project. Got to Project, Build Options, select Preprocessor,
then in Pre-defined symbol field type CHIP_6424. Voila!
// librairie rts6700.lib;dsk6713bsl.lib;csl6713.lib
```