Application development for TMS320C5416 DSK system

The laboratory presents two applications realized using Code Composer Studio: DTMF tones detection using Geortzel algorithm and creation of special effects sounds.
In the first part are presented the methods and the algorithms which are the base of their development, then are described the source programs. Both projects are done on TMS320C5416 DSK board.

1. Geortzel Algorithm

1.1 DTMF tones detection

The first phone with keyboard based on tones was installed in 1963. DTMF (Dual Tone Multi Frequency) signaling uses voice band tones for sending addresses signals but also other digital information. DTMF technique is often used for digital phones with bold buttons which are an alternative of phones with disk. This technique was used in electronic post or bank application through phone, the users are selecting options from the menu sending DTMF signals from phone.

In a DTMF signaling system, a combination of two frequency tones represents a specific number, a character (A, B, C or D) or a symbol (*, #), as we can see in the figure below. e.g. key tone 5 is generated simultaneously by the frequencies 770Hz and 1336Hz as in the figure 2.

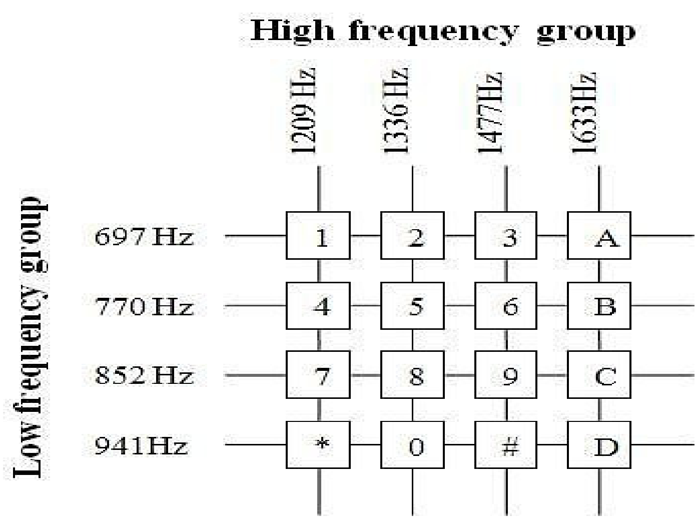

Fig. 1. DTMF frequencies and character assignment

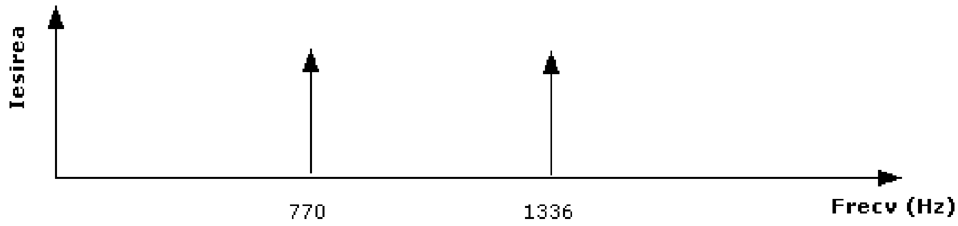

Fig. 2. Key tone 5

DTMF system encapsulates:

- one encoder which translates one key pressing or number's information in dual tone;
- one decoder which detects the presence and the content of DTMF tone signal information received.

Dual tone generation is realized connecting two parallel sinusoidal sources. One method used for implementation is polynomial approximation. DTMF signals must have the next characteristics: 8 kHz sampling rate, 10 digits/second transmitted, tone duration greater then 40ms.

The task to detect DTMF tones from a received signal and to convert them into numbers is more complicated then coding process. Decoding process is a continuous one, this means that continuously must be checked if DTMF tones are present in the data signal pending to be received.

Tone detection can be done using several filters or using Discrete Fourier Transform (DFT or FFT). Geortzel algorithm was proved to be more efficient for this type of application then the other methods remembered.

1.2 Geortzel algorithm description

Geortzel algorithm is the base of DTMF decoder. This represents a rapid and efficient method to extract spectral information from an input signal. Geortzel is derived from DFT and is showed the phase factor periodicity exp(-j*2πk/N) for reducing the DFT computation complexity.

With Geortzel algorithm are necessary only 16 DFT samples for 16 tones. The algorithm uses an IIR filter with two poles for calculating DFT's effective values. Is a recursive structure which operates always on a received sample while FFT needs a data block before starting processing.

For tone effective detection, DFT magnitude information (here is squared magnitude) is enough. After N samples (one DFT block), the output of Geortzel filter converge to a DFT pseudo-value $v_k(n)$, which can used to determine the squared magnitude. We present a short mathematic description and also the block schema of the algorithm.

1. DFT pseudo-value recursive calculus for $n = \overline{0, N}$

$$v_k(n) = \underbrace{2\cos\left(\frac{2\pi}{N}k\right)}_{q} \cdot v_k(n-1) - v_k(n-2) + x(n)$$

where: $v_k(-1) = 0;$ $v_k(-2) = 0;$ $x(n)$ - input samples;

2. Magnitude calculus

$$\left|X(k)\right|^2 = y_k(N)\cdot y_k^*(n)$$
$$= v_k^2(n) + v_k^2(N-1) - 2\cos(2\pi f_k / f_s) v_k^2(N) v_k^2(N-1)$$

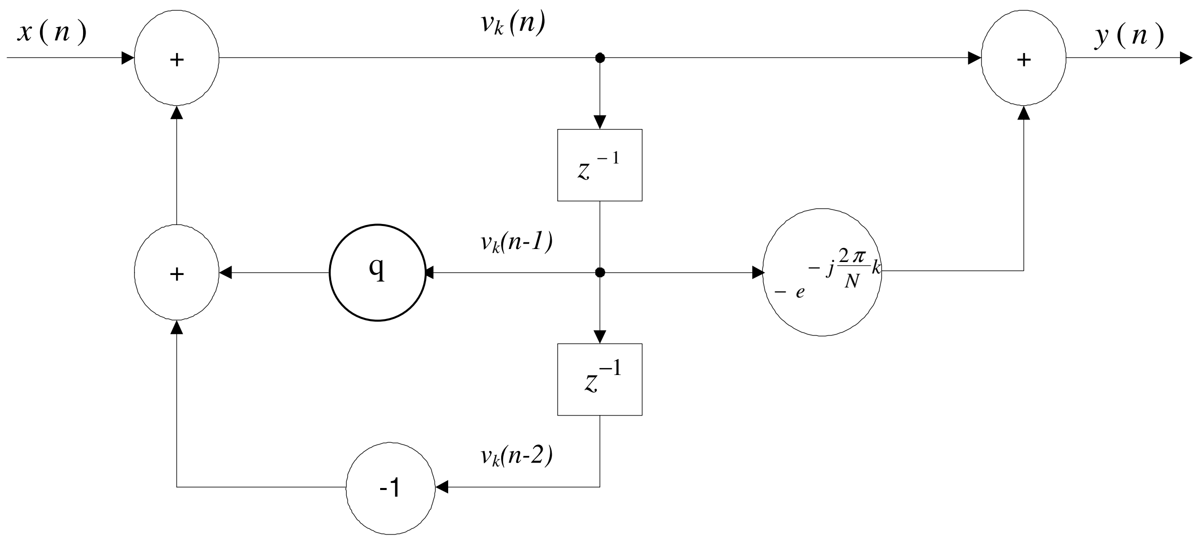

Fig. 3. Geortzel algorithm block scheme

Geortzel algorithm is a rapid method to calculate DFT values in certain conditions. It has two advantages:

- phase periodicity;
- just a particular number of DFT spectral values are necessary (in this application we have tones for 8 lines/rows)

Parameter N defines the number of recursive iterations and also provides the means to give the frequency resolution. K value determines the tone that we try to detect and is given by the expression:

$$k = Nx\frac{f_k}{f_s}$$

Where: - f_k = tone frequency
- f_s = sampling frequency
- N is fixed at 205.

Here we can calculate (cos(2πfk/fs)) coefficients assigned to used frequencies:

Frequency	**k**	**Coefficients (decimal)**	**Coefficients (Q15)**
697	18	1.703275	0x6D02*
770	20	1.635585	0x68AD*
852	22	1.562297	0x63FC*
941	24	1.482867	0x5EE7*
1209	31	1.163138	0x4A70*
1336	34	1.008835	0x4090*
1477	38	0.790074	0x6521
1633	42	0.559454	0x479C

Tabel 1. Goertzel algorithm coefficients

* decimal values are devised by 2 (for obtaining factionary values) then are represented in Q15 format.

1.3. Geortzel algorithm implementation

Geortzel.pjt project was realized using CSS. It is used a microphone for receiving sounds generated by phone's keyboard. Pressed buttons are identified using Geortzel algorithm and their values are displayed in CCS output window. In the figure bellow is presented the application organigram and then is described the method implemented. The geortzel.c application source can be followed in Annex 1.

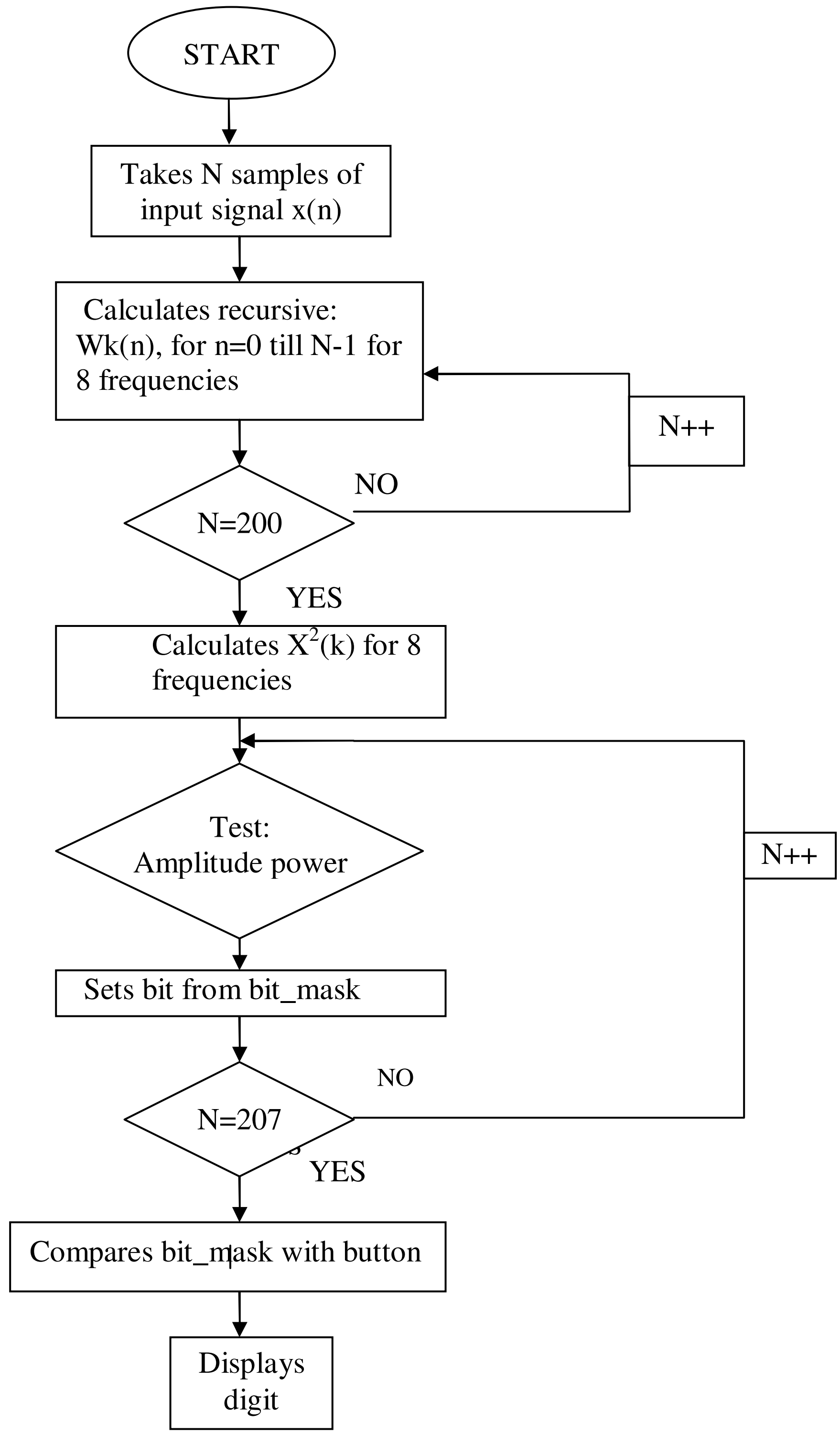

Fig. 4. Geortzel algorithm organigram

The main part of this DTMF decoder is implemented with UserTask() method. First we must initialized codec in line:

```
hCodec = DSK5416_PCM3002_openCodec(0, &setup);
```

then the program enter in a loop and in each iteration are read signals from input channels and then are send to the codec's output.

```
while (!DSK5416_PCM3002_read16(hCodec, &left_input));
      while (!DSK5416_PCM3002_write16(hCodec, left_output));
      while (!DSK5416_PCM3002_read16(hCodec, &right_input));
      while (!DSK5416_PCM3002_write16(hCodec, right_output));
```

The program reads on DSK switches and display their value in CCS output window; generates also one left and right mono input.

```
switch_value = switch_status_display();
mono_input = stereo_to_mono ( left_input, right_input);
```

It is checked switches state. e.g. if the switch is fixed on value 9, calculate recursive part Vk(n) for n=1..N.

```
     if ( 9 == switch_value) //
           {
             goertzel_filter(           &delay_697_Hz[0],           mono_input,
COEFFICIENT_697_Hz);
             .........
             goertzel_filter(           &delay_1633_Hz[0],          mono_input,
COEFFICIENT_1633_Hz);
             N++; //incrementeaza  N
```

Once that spectral information (magnitude squared) for each row and column frequency is collected, it is necessary a series of keys to be executed to determine tones and numbers results validation. A first examination is assured by the fact that the power spectrum of the possible tones and columns pairs don't overcome a fixed threshold. Then is verified for each frequency also that the calculated power is greater then the threshold. In case that the results are valid are set the corresponding bits in bit mask:

```
  if ( 199 == N) //First row of keyboard?
              {
                goertzel_output_697_Hz        =        calculate_goertzel_output(
&delay_697_Hz[0],//calculeaza   puterea   de   iesire   Goertzel   pentru   aceasta
frecventa.

COEFFICIENT_697_Hz);
                if   (   goertzel_output_697_Hz   >   GOERTZEL_THRESHOLD)//daca
rezultatul este mai mare decat pragul seteaza bitul 0 in mask
                  {
                    goertzel_bit_mask |= 0x0001; /* seteaza bitul 0 in bitmask
*/
                  }
                else
                  {
                    goertzel_bit_mask  &=  0xFFFE;  /*  reseteaza   bitul  0  in
bitmask */
                  }
```

When N reaches the value 207, the program compares bit mask with buttons value declared in geortzel.h and display a message with pressed button.

```
if  (  207  ==  N)//afiseaza  butonul  apasat  in   Stdout  si  apoi  se  reseteaza
bitmask
              {
               if ( BUTTON_0 == goertzel_bit_mask)         /*compara bitmask-ul
cu "butoanele" declarate in goertzel.h*/
                 {
                   puts("Button 0 pressed.\n");
                 }
               N = 0;
```

```
goertzel_bit_mask = 0;
}
```

Then N value and bit mask are reset for doing a new reading.

2. Creation of special sounds effects

2.1. Theoretical considerations

The project alienvoices.pjt presents the modulation effect on radio frequencies. The application shows how the fundamental generates sums or differences of frequencies. So are generated strange voices for aliens used in science fiction movies and in television.

Digital Modulations

Base Band signaling is used for data transmission on physic channels as telephone cables or radio frequency channels. Data source (bits or symbols) can be a folder or a digitized wave form. (voice, video…). The signal transmits information about data, and his characteristics are changed at the same time with data transmission rate.

When the signal sends information:

- is extended from 0 Hz upward and is used the term base band signaling.
- has the power centered around one central frequency *fc* and is used the term pass band signaling or modulation.

Modulation is used because:

- the channel doesn't include the 0 Hz frequency and base band signaling is impossible.
- the channel's band is divided in multiple channels for frequency multiplexing.
- for radio-communication, the antenna dimension decreases when frequency increases.

In simple modulation schemes the fundamental is modified with the same rate with which the data are transmitted. The fundamental is usually written: $A\cos(2\pi f_C + \Phi)$,

where: *fc* – fundamental frequency

A - fundamental amplitude

Φ – fundamental phase

Those principlas fundamental parameters : amplitude, phase and frequency can be modified so to transmit information leading to amplitude modulation, phase modulation, frequency modulation.

In the next exemple base band components Z_I and Z_Q are modulated in amplitude by two quadrature fundamentals.

$$x(t) = \cos(2\pi f_c t + \Phi(t)) = \cos(\Phi(t))\cos(2\pi f_c t) - \sin(\Phi(t))\sin(2\pi f_c t)$$
$$x(t) = Z_I(t)\cos(2\pi f_c t) - Z_Q(t)\sin(2\pi f_c t)$$

where : *fc* – fundamental frequency

$Z_I(t) = \cos(\Phi(t))$ and $Z_Q(t) = sin(\Phi(t))$ - base band components

$purtatoare_I = \cos(2\pi f_c t)$ and $purtatoare_Q = sin(2\pi f_c t)$ - fundamentals (are in general RF analogical signals, generated by analogical oscilators)

2. 2 Application presentation

In the application alienvoices.c the input signal is taken from a microphone or from a CD player, is multiplied with a sinusoidal signal for generating frequency sum or difference and then is sended to speakers output.

The program organigram is presented in figure below:

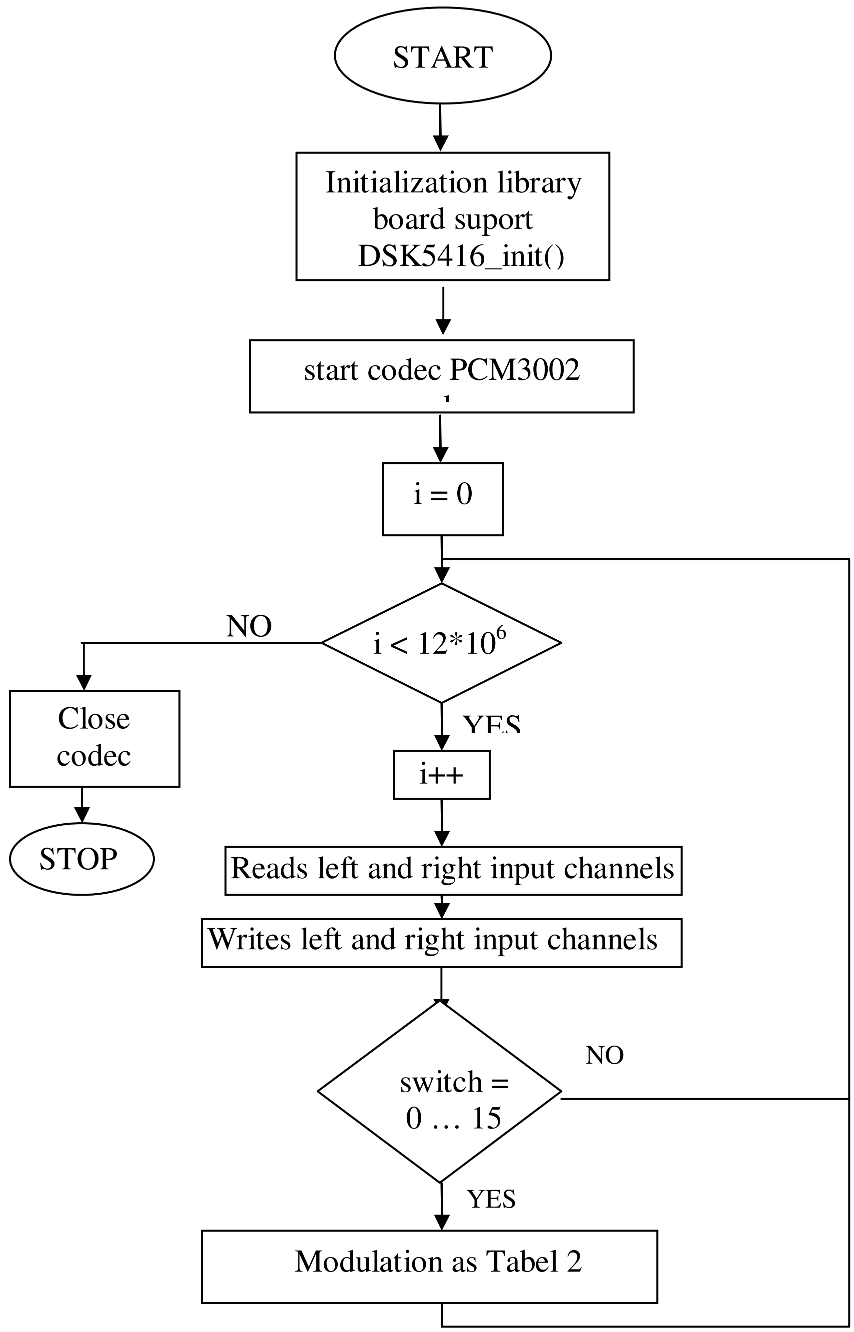

Fig. 5. alienvoices application organigram

Fundamental frequency modulation is simulated with *static short int* ***ring_modulation*** *(short int input1m short int input2) function*, defined in main file alienvoices.c (see Annex 2) This function realize a circular modulation by multiplying the input with a generated sinusoidal signal.

Sinusoidal signal generation is realized in file sinewave.c studied in previously laboratory. The file generates to sinusoidal signals with adjustable frequency using sine() function from dsplib.lib. Both functions:

- *signed int **generate_sinewave_1**(signed short int frequency, signed short int amplitude)*
- *signed int **generate_sinewave_2**(signed short int frequency, signed short int amplitude)*

receive as first parameter sinusoidal frequency signal (between 10 Hz and 16000 Hz), while the second parameter is the maximum amplitude of sinus (between 1 Hz and 32767). In this application all sinusoidal signals have amplitude of 32767.

Both channels (left and right) are modulated and then the result is send to the output to correspond channel.

For some switching values, circular modulation result is filtered with a four order high and/or low pas filter. The filtering process is implemented in file *IIR_filters_fourth_order.c* and is described in application *iir.c.*

For different switching value, we can obtain the next type of modulations:
(L/R- left/righ):

Switch Value	**Input**	**Output**	**Tipulfiltrării**
0	L/R	L/R	output = input
1	L/R	L/R	Modulation on a sinusoidal signal of 2Hz
2	L/R	L/R	Modulation on a sinusoidal signal of 20Hz
3	L/R	L/R	Modulation on a sinusoidal signal of 100Hz
4	L/R	L/R	Modulation on a sinusoidal signal of 200Hz
5	L/R	L/R	Modulation on a sinusoidal signal of 500Hz
6	L/R	L/R	Modulation on a sinusoidal signal of 1000Hz
7	L/R	L/R	Modulation on a sinusoidal signal of 2000Hz
8	L/R	L/R	Modulation on a sinusoidal signal of 600Hz + filtration
9	L/R	L/R	Modulation on a sinusoidal signal of 1000Hz + filtration
10	L/R	L/R	Modulation on a sinusoidal signal of 2000Hz + filtration
11	mono	L/R	Double modulation, at 2000Hz then at 2400Hz + FTS + FTJ
12	mono	L/R	Double modulation, at 2000Hz then at 2400Hz + FTJ + FTS
13	mono	L/R	Double modulation, at 2400Hz then at 2000Hz + FTS + FTJ
14	mono	L/R	Double modulation, at 2400Hz then at 2000Hz + FTJ + FTS
15	mono	L/R	Modulation on a sinusoidal signal of500Hz + reverberation

Tabel 2 – Modulation types