Dezvoltarea aplicaţiilor pentru sistemul DSK TMS320C5416

Lucrarea prezintă două aplicaţii realizate folosind mediul Code Composer Studio: detecţia tonurilor DTMF utilizând algoritmul Goertzel şi crearea unor efecte sonore speciale. In prima parte sunt prezentate metodele şi algoritmii care stau la baza implementării lor iar apoi sunt descrise programele sursă. Ambele proiecte sunt rulate pe placa DSK TMS320C5416.

1. Algoritmul Goertzel

1.1. Detecţia tonurilor DTMF

Primul telefon cu tastatură bazată pe tonuri a fost instalat in 1963. Semnalizarea DTMF (Dual Tone Multiple Frequency) foloseşte tonuri în banda de voce pentru a trimite semnalele de adrese precum şi alte informaţii digitale. Tehnica DTMF este predominant folosită pentru setarea telefoanelor digitale cu butoane reliefate care sunt o alternativă a setării telefoanelor rotative. A fost extinsă mai nou şi la poşta electronică sau la aplicaţiile bancare prin intermediul telefonului, în care utilizatorii selectează opţiuni din meniu trimiţând semnale DTMF de pe telefon.

Într-un sistem de semnalizare DTMF o combinaţie dintre două tonuri de frecvenţă reprezintă o cifră specifică, un caracter (A, B, C sau D) sau un simbol(* , #) , după cum se poate observa în urmatoarea figură. De exemplu, tonul tastei 5 este generat simultan de frecvenţele 770Hz şi 1336Hz, conform figurii 2.

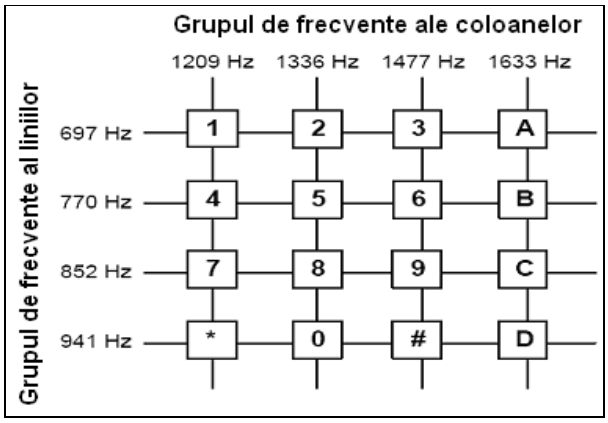

Fig. 1. Frecvenţele DTMF şi asignarea caracterelor

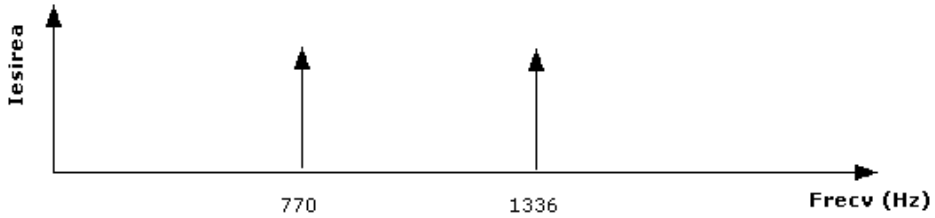

Fig. 2. Tonul Tastei 5

Sistemul DTMF încorporează:

- un codor care traduce apăsarea unei taste sau informaţia unei cifre în semnale de ton dual;
- un decodor care detectează prezenţa şi conţinul informaţiei semnalelor de ton DTMF primite.

Generarea tonurilor duale se realizează conectând două surse sinusoidale în paralel. Una din metodele folosite în implementare este aproximarea polinomială. Semnalele DTMF trebuie să aibă următoarele caracteristici: rata de eşantionare 8kHz, se transmit 10 digiţi pe secundă, durata unui ton mai mare de 40ms.

Sarcina de a detecta tonuri DTMF într-un semnal primit şi de a le converti în cifre este mult mai complexă decât procesul de codare. Procesul de decodare este prin natura sa un proces continuu, acest fapt însemnând că trebuie să se verifice în permanenţă dacă tonurile DTMF sunt prezente în semnalul de date în curs de primire.

Detectarea tonurilor se poate face utilizând mai multe filtre sau folosind Transformata Fourier Discreta (DFT sau FFT). Algoritmul Goertzel s-a dovedit a fi mult mai eficient pentru acest tip de aplicaţie decât celelalte metode amintite.

1. 2. Descrierea algoritmului Goertzel

Algoritmul Goertzel este baza detectorului DTMF. Acesta reprezintă o metodă foarte eficientă şi rapidă de extragere a informaţiei spectrale dintr-un semnal de intrare. Algoritmul Goertzel derivă din DFT şi evidenţiază periodicitatea factorului de fază exp(-j*2πk/N) pentru a reduce complexitatea calculului asociat DFT.

Cu algoritmul Goertzel sunt necesare doar 16 eşantioane DFT pentru 16 tonuri. Algoritmul foloseşte filtre de tip IIR cu doi poli pentru a calcula efectiv valorile DFT. Prin urmare, este o structură recursivă care operează întotdeauna doar pe un eşantion primit, în timp ce Transformata Fourier (sau FFT) are nevoie de un bloc de date înainte de a putea începe procesarea.

Pentru detecţia efectivă de ton, informaţia de magnitudine (aici pătratul magnitudinii) a DFT-ului este suficientă. După un anumit număr de eşantioane N (echivalent cu dimensiunea unui bloc DFT), ieşirea filtrului Goertzel converge spre o pseudo valoare DFT $v_k(n)$, care poate fi apoi folosită pentru a determina pătratul magnitudinii. În continuare se prezintă o scurtă descriere matematică precum si schema bloc a algoritmului:

1. Calculul recursiv a pseudo-valorilor DFT pentru $n = \overline{0, N}$

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

unde: $v_k(-1) = 0; \quad v_k(-2) = 0; \qquad x(n)$ - eşantioane intrare

2. Calculul magnitudinii:

$$\begin{aligned} |X(k)|^2 &= y_k(N) \cdot y_k^*(n) \\ &= v_k^2(n) + v_k^2(N-1) - 2\cos(2\pi f_k / f_s) v_k^2(N) v_k^2(N-1) \end{aligned}$$

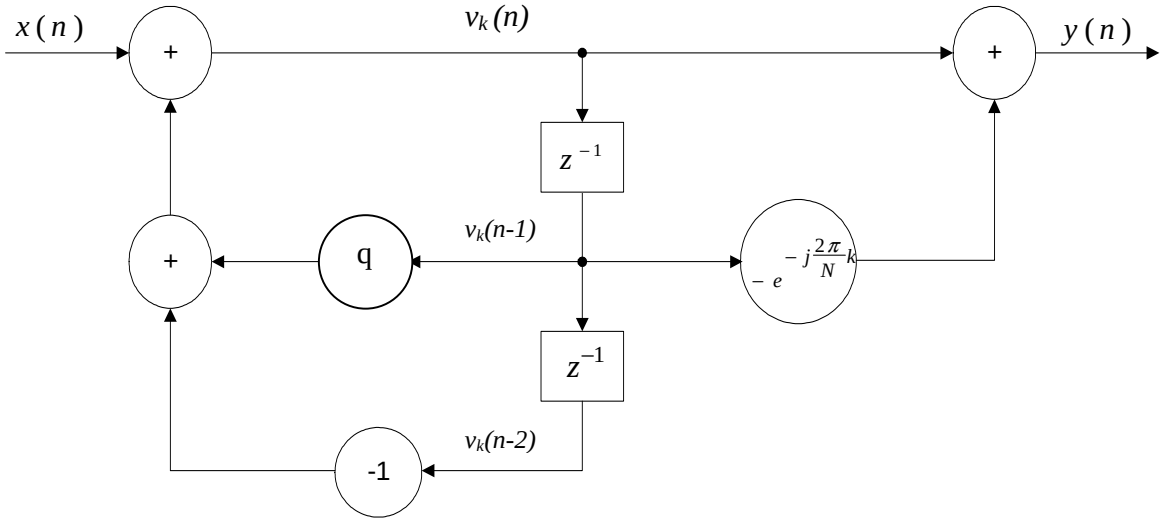

Fig 3. Schema bloc a algoritmului Goertzel

În esenţă algoritmul Goertzel este o metodă foarte rapidă de a calcula valorile DFT în anumite condiţii. Are două avantaje:

- periodicitatea fazei;
- doar un anumit set din valorile spectrale ale unui DFT sunt necesare (în această aplicaţie avem tonuri pentru 8 rânduri/coloane)

Parametrul N defineşte numărul de iteraţii recursive şi de asemenea furnizează mijloacele pentru a acorda rezoluţia frecvenţei. Valoarea lui k determină tonul pe care încercăm să-l detectăm şi este dată de expresia :

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

unde: - f_k = frecvenţa tonului

- f_s = frecvenţa de eşantionare.
- N este fixat la 205.

Apoi putem calcula coeficienţii (cos(2πfk/fs)) corespunzători frecvenţelor utilizate:

Frecvenţa	**k**	**Coeficienţii (zecimal)**	**Coeficienţii (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. Coeficienţii algoritmului Goertzel

* valorile zecimale sunt împărţite cu 2 (pentru a obţine valori subunitare) iar apoi sunt reprezentate în format Q15

1. 3. Implementarea algoritmului Goertzel

Proiectul *goertzel.pjt* a fost realizat folosind mediul CCS. Se foloseşte un microfon petru a recepta sunetele generate de tastele telefonului. Butoanele apăsate sunt identificare folosind algoritmul Goerztel şi valorile lor sunt afişate în fereastra de ieşire a CCS. În figura următoare este prezentată organigrama aplicaţiei iar apoi este descrisă metoda implementată. Sursa aplicaţiei *goertzel.c* poate fi urmărită în Anexa 1.

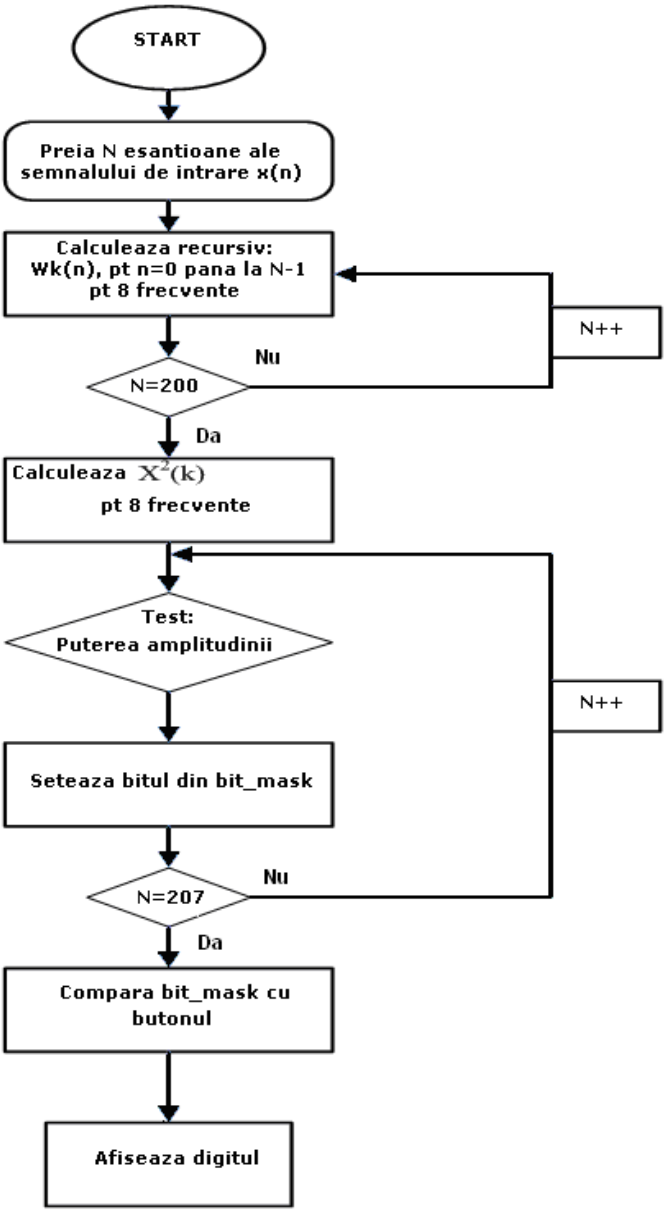

Fig. 4. Organigrama algoritmului Goertzel

Partea principală a acestui detector DTMF este implementată în metoda *UserTask()*. Mai întâi trebuie iniţializat codecul în linia:

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

apoi programul intră într-o buclă şi în fiecare iteraţie citeşte semnalele de la canalele de intrare şi apoi le trimite la ieşirea codec-ului.

```
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));
```

Programul citeşte apoi comutatoarele de pe DSK şi le afişează valoarea în fereastra de ieşire a CCS; generează de asemenea o intrare mono de la intrarea stânga şi dreapta.

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

Se verifică apoi starea comutatoarelor. De exemplu, dacă comutatorul este fixat pe valoarea 9, calculează partea recursivă Vk(n) pentru 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
```

Odată ce informaţia spectrală (pătratul magnitudinii) pentru fiecare frecvenţă de pe rând şi coloană este colectată, este necesar ca o serie de teste să fie executate pentru a determina validitatea rezultatelor tonului şi a cifrelor. O primă verificare asigură faptul că puterea semnalului posibilei perechi de tonuri DTMF este suficientă. Suma pătratelor magnitudinilor vârfurilor componentelor spectrale ale rândurilor, respectiv coloanelor trebuie să depăsească un anumit prag. Apoi se verifică pentru fiecare frevenţă şi dacă puterea calculată e mai mare decât pragul. În cazul în care rezultatele sunt valide se setează biţii corespunzători în bitmask:

```
            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 */
                }
```

Când N atinge valoarea 207 programul compară bitmask-ul cu valoarea “butoanelor” declarate în goertzel.h şi afişează un mesaj cu numărul butonului apăsat.

```
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;
             }
```

Apoi valoarea lui N şi bitmask-ul sunt resetate pentru a face o nouă citire.

2. Crearea unor efecte sonore speciale

2.1 Consideraţii teoretice

Proiectul *alienvoices.pjt* prezintă efectul modulaţiei asupra frecvenţelor radio. Aplicaţia arată cum purtătoarea generează sume şi diferenţe de frecvenţe. Astfel se generează vocile ciudate folosite pentru extratereştri în filmele science fiction şi în televiziune.

Modulatii digitale

Semnalizarea bandă de bază şi trece bandă este folosită pentru transmiterea datelor pe canalele fizice cum ar fi cablurile telefonice sau canalele de radio frecvenţă. Sursa de date (biţi sau simboluri) poate fi un fişier sau o formă de undă digitizată (voce, video…). Semnalul transmite informaţii despre date, iar caracteristicile lui se schimbă odată cu schimbarea ratei de transmisie a datelor.

Cand semnalul care transportă informaţia:

- se extinde de la 0 Hz în sus, este folosit termenul de *semnalizare banda de bază*
- are puterea centrata în jurul unei frecvenţe centrale *fc*, este folosit termenul de *semnalizare trece banda* sau *Modulaţie*

Modulatia este folosita deoarece:

- Canalul nu include frecvenţa de 0 Hz şi semnalizarea banda de bază este imposibilă
- Banda canalului este împărţită în mai multe canale pentru multiplexare în frecvenţa
- Pentru radio-comunicatii, dimensiunea antenei descreşte când frecvenţa transmisă creşte

În schemele simple de modulaţie purtătoarea este modificată cu aceeaşi rată cu care se transmit datele. Purtătoarea este de obicei scrisă în modul următor:

$$A\cos(2\pi f_C + \Phi),$$

unde: *fc*= frecvenţa purtătoarei
A = amplitudinea purtătoarei
Φ = faza purtătoarei

Cei trei parametrii principali ai purtătoarei: amplitudine, fază şi frecvenţă pot fi modificaţi astfel încât să transmită informaţii conducând la modulaţie în amplitudine, modulaţie de fază şi modulaţie de frecvenţă.

În exemplul următor componentele bandei de bază Z_I şi Z_Q sunt modulate în amplitudine de două purtătoare în quadratură.

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

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

unde: f_c = frecvenţa purtătoarei
$Z_I(t) = \cos(\Phi(t))$ şi $Z_Q(t) = sin(\Phi(t))$ - componentele bandei de bază

$purtatoare_{\mathrm{I}} = \cos(2\pi f_c t)$ şi $purtatoare_{\mathrm{Q}} = sin(2\pi f_c t)$ - purtătoarele (sunt în general semnale analogice RF, generate de oscilatoare analogice)

2.2. Prezentarea aplicatiei

In cadrul aplicaţiei *alienvoices.c* semnalul de intrare, preluat de la un microfon sau de la un CD player, este multiplicat cu un semnal sinusoidal pentru generarea sumei şi a diferenţei de frecvenţe şi apoi este trimis la ieşire spre difuzoare.

Organigrama programului este prezentată în figura următoare:

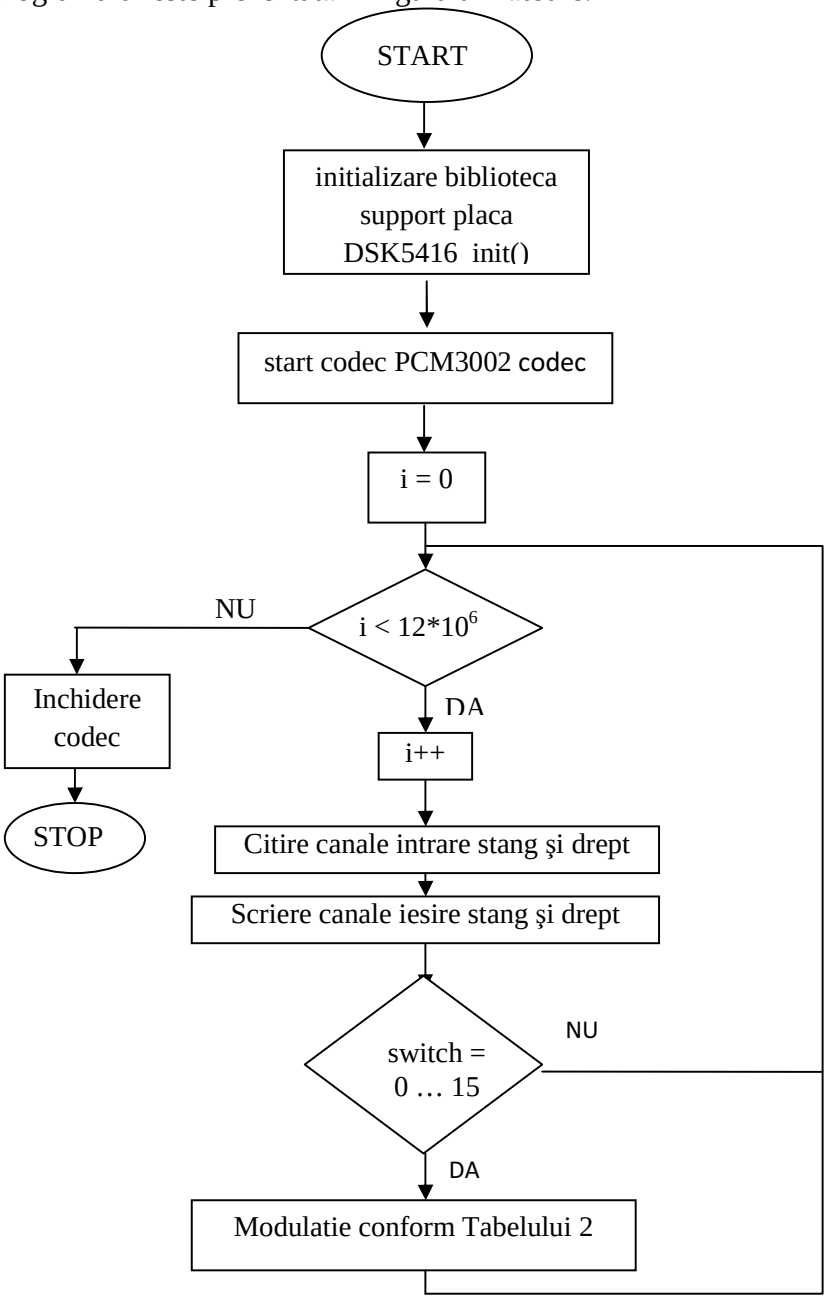

Fig. 5. Organigrama aplicaţiei alienvoices

Modularea frecvenţei purtătoare este simulată cu ajutorul funcţiei *static short int* ***ring_modulation*** *(short int input1, short int input2),* definită în fişierul principal *alienvoices.c* (vezi Anexa 2). Această funcţie realizează modulaţie circulară prin înmulţirea intrării cu semnalul sinusoidal generat.

Generarea semnalului sinusoidal este realizată în fişierul *sinewave.c* studiat în lucrarea anterioară. Fişierul generează două semnale sinusoidale cu frecvenţa reglabilă folosind funcţia *sine()* din *dsplib.lib.* Cele două funcţii:
- *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)*

primesc ca prim parametru frecvenţa semnalului sinusoidal (între 10 Hz şi 16000Hz), în timp ce al doilea parametru este amplitudinea maximă a sinusului (între 1 şi 32767). În această aplicaţie toate semnalele sinusoidale au o amplitudine de 32767.

Ambele canale (drept şi stâng) sunt modulate şi apoi rezultatul este trimis spre ieşire la canalul corespunzător.
Pentru unele valori de comutare, rezultatul modulaţiei circulare este apoi filtrat cu un filtru trece sus şi/sau trece jos de ordinul patru. Procesul de filtrare este implementat în fişierul *IIR_filters_fourth_order.c* şi este descris în aplicaţia *iir.c.*

Pentru diferite valori de comutare, se pot obţine următoarele tipuri de modulaţii:
(L/R – stânga/dreapta):

Valoare switch	**Intrare**	**Ieşire**	**Tipul filtrării**
0	L/R	L/R	Ieşire = intrare
1	L/R	L/R	Modulaţie pe un semnal sinusoidal de 2Hz
2	L/R	L/R	Modulaţie pe un semnal sinusoidal de 20Hz
3	L/R	L/R	Modulaţie pe un semnal sinusoidal de 100Hz
4	L/R	L/R	Modulaţie pe un semnal sinusoidal de 200Hz
5	L/R	L/R	Modulaţie pe un semnal sinusoidal de 500Hz
6	L/R	L/R	Modulaţie pe un semnal sinusoidal de 1000Hz
7	L/R	L/R	Modulaţie pe un semnal sinusoidal de 2000Hz
8	L/R	L/R	Modulaţie pe un semnal sinusoidal de 600Hz + filtrare
9	L/R	L/R	Modulaţie pe un semnal sinusoidal de 1000Hz + filtrare
10	L/R	L/R	Modulaţie pe un semnal sinusoidal de 2000Hz + filtrare
11	mono	L/R	Modulaţie dublă, la 2000Hz apoi la 2400Hz + FTS + FTJ
12	mono	L/R	Modulaţie dublă, la 2000Hz apoi la 2400Hz + FTJ + FTS
13	mono	L/R	Modulaţie dublă, la 2400Hz apoi la 2000Hz + FTS + FTJ
14	mono	L/R	Modulaţie dublă, la 2400Hz apoi la 2000Hz + FTJ + FTS
15	mono	L/R	Modulaţie pe semnal sinusoidal de 500Hz + reverberaţie

Tabel 2 – Tipuri de modulaţie