EFFICIENT TONE DETECTION SOLUTIONS USING PROGRAMMABLE LOGIC DEVICES

Guillermo A. JAQUENOD
Fac. Ingeniería, UNCPBA,
ARGENTINA.
chipi@netverk.com.ar

Horacio A. VILLAGARCÍA
CICPBA – Fac. Informática,
UNLP, ARGENTINA.
hvw@info.unlp.edu.ar

Marisa R. DE GIUSTI
CICPBA – Fac. Informática,
UNLP, ARGENTINA.
marisadg@volta.ing.unlp.edu.ar

ABSTRACT

Digital solutions have been used for many years in central office and private automated branch exchange (PABX) designs. Although many functional blocks are available (ASICs or DSP solutions), only a few programmable logic solutions are known.

Within a PABX and over E1 or T1 trunks, signals are digitized samples of audio, both for voice and signaling purposes. Dual Tone Multi Frequency (DTMF) is a standard where keystrokes from the telephone keypad are translated into dual tone signals over the audio link. Additionally, On/Off modulation of a 425 Hz signal, is used to identify the Call Progress Status (CPS): Dialing, Ringing, etc..

Special algorithms must be run to decode these signals, not only to detect the presence of tones, but also to ensure that voice signals can't be misdetected as DTMF.

This paper describes a DTMF/CPS numeric coprocessor for detection task implemented using programmable logic devices (PLD). The same architecture, with different parameters, can be used to process other tone standards, such as ITU-R2 signaling.

RESUMEN

Desde hace años las centrales telefónicas automáticas públicas y privadas (PABX) han sido resueltas con diseños digitales. Aunque muchos bloques funcionales están disponibles (ASIC´s y para DSP) pocas soluciones en lógica programable se conocen.

Dentro de un PABX y sobre los troncales digitales E1 o T1, las señales son muestras digitalizadas de audio, tanto para voz como para señalamiento. Dual Tone Multi Frequency (DTMF) es un estándar en donde cada tecla oprimida de un teclado telefónico se traduce en señales de tonos dobles sobre el enlace de audio. Además, modulación On/Off de una señal de 425 Hz permite identificar el estado de una llamada (Call Progress Status -CPS): Discando, Llamando, etc..

Algoritmos especiales deben ser ejecutados, no sólo para detectar la presencia de tonos DTMF sino tambien para asegurar que las señales de voz no sean consideradas tonos DTMF.

El presente artículo describe un coprocesador DTMF/CPS numérico para la tarea de detección, implementado usando dispositivos lógicos programables (PLD). La misma arquitectura, con diferentes parámetros, puede ser usada para procesar otros estándares con tonos como el señalamiento ITU-R2.

EFFICIENT TONE DETECTION SOLUTIONS USING PROGRAMMABLE LOGIC DEVICES

Guillermo A. JAQUENOD
Fac. Ingeniería, UNCPBA,
ARGENTINA.
chipi@netverk.com.ar

Horacio A. VILLAGARCÍA
CICPBA – Fac. Informática,
UNLP, ARGENTINA.
hvw@info.unlp.edu.ar

Marisa R. DE GIUSTI
CICPBA – Fac. Informática,
UNLP, ARGENTINA.
marisadg@volta.ing.unlp.edu.ar

ABSTRACT

Digital solutions have been used for many years in central office and private automated branch exchange (PABX) designs. Although many functional blocks are available (ASICs or DSP solutions), only a few programmable logic solutions are known.

Within a PABX and over E1 or T1 trunks, signals are digitized samples of audio, both for voice and signaling purposes. Dual Tone Multi Frequency (DTMF) is a standard where keystrokes from the telephone keypad are translated into dual tone signals over the audio link. Additionally, On/Off modulation of a 425 Hz signal, is used to identify the Call Progress Status (CPS): Dialing, Ringing, etc..

Special algorithms must be run to decode these signals, not only to detect the presence of tones, but also to ensure that voice signals can't be misdetected as DTMF.

This paper describes a DTMF/CPS numeric co-processor for detection task implemented using programmable logic devices (PLD). The same architecture, with different parameters, can be used to process other tone standards, such as ITU-R2 signaling.

1. INTRODUCTION

Dual Tone Multi Frequency (DTMF) signaling is a standard in telecommunications systems [1], that translates key strokes from the telephone keypad into dual tone signals over the audio link. Each key is uniquely encoded as the sum of its row and column frequencies as shown in Table 1; those keys usually found in a telephone keypad are shown remarked.

Table 1: DTMF encoding

	Columns 1-4			
Rows 1-4	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	**Key 1**	**Key 2**	**Key 3**	Key A
770 Hz	**Key 4**	**Key 5**	**Key 6**	Key B
852 Hz	**Key 7**	**Key 8**	**Key 9**	Key C
941 Hz	**Key ***	**Key 0**	**Key #**	Key D

DTMF (also known as Touch-Tone) was developed by Bell Labs and used by AT&T as an in-band signaling system. The Bellcore standard was later refined by the International Telecommunication Union (ITU) [2] with more stringent constraints in frequency tolerances, power, twist, and talk-off , shown in Table 2.

Table 2: ITU recommendations

Signal Frequencies	Low Group High Group	697,770,852,941 1209,1336,1477,1633
Frequency Tolerance	Operation Non-operation	<= 1.5 % >= 3.5 %
Signal Duration	Operation Non-operation	40 ms minimum 23 ms maximum
Signal Exceptions	Pause Duration Signal Interruption	40 ms maximum 10 ms minimum
Twist	Forward Reverse	8 dB 4 dB
Signal Strength	SNR Power	15 dB minimum -26 dBm minimum
Talk-Off: 1st to 2nd harmonic relation		> 30 dB

Classic hardware solutions for DTMF detectors have been based on individual detectors built as banks of 8 analog filters/detectors followed by a logic section. In the last years, the availability of faster and cheaper DSPs has lead to multi-channel numeric solutions that can easily be handled by a DSP concurrently with other tasks. A conventional implementation of a DTMF detector with a DSP requires approximately 1 MIPS on a 16 fixed-point DSP, thus the 30 voice channels within a complete E1 trunk can be processed by a single device [3][6][7][8][9].

The recent emergence of System On a Programmable Chip (SOPC) solutions [11] gave new alternatives to PABX designers, since most of the digital building blocks (Codec interfaces, A-law/mu-law companding, Time Slot Interchange circuits, Frame Synchronization, Framing/ Deframing, DTMF/Call Progress detection and generation, Automatic Gain Control, etc) can be embedded together with the control processor in a single chip, even including an operating system with file management abilities. This leads to very compact PABX solutions, built around a single PLD, a FLASH memory for firmware and configuration, and the analog chips: CODECs and SLICs for the analog lines, and LIUs for the digital trunks (CODEC: Analog to PCM Coder/Decoder, SLIC: Suscriber Line Interface Circuit, LIU: Line Interface Unit).

This paper describes an efficient DTMF/Call Progress Detection front-end, implemented using programmable logic devices.

2. TONE DETECTION BACKGROUND

Tone detection is based on the energy measurement of the spectral contents of a very narrow frequency bin.

To measure the spectral contents of a signal, both the Discrete Fourier Transform (DFT) and the Fast Fourier Transform (FFT) can be used to transform real discrete-time domain signals into their discrete, complex, frequency-domain components.

FFT is very efficient when the whole spectrum must be computed, but it has some drawbacks for DTMF detection:

- it requires the buffering of many samples
- the number of samples N must be a power of 2
- the center frequency of the spectral lines is an integer multiple of f_s/N, and the constraint imposed over the possible N values limits the matching between these lines and DTMF frequencies.

If less than $\log_2 N$ spectral lines must be computed, the DFT solution is more efficient [8][9][10]c,k,e], since no unnecessary information is generated.

The definition of an N-point DFT is as follows:

$$X(k) = \sum_{n=0}^{N-1} x(n).W_N^{nk} \qquad (1)$$

where:

$$k = 0,1,..,N-1$$
$$W_N^{nk} = e^{-j(2\pi k/N).n} \qquad (2)$$

The W_N^{nk} term represents a complex rotating vector, where $k=N.f_k/f_s$, and therefore the X(k) computation (basically, a correlation between the input signal x(n) and the complex sinusoid W_N^{nk}) involves many complex multiplications and additions.

When using the DFT, the number of samples analyzed doesn't need to be a power of 2, thus N may be chosen so that the center frequency of the analyzed spectral lines (defined by $f_s.k/N$) matches as tightly as possible to the normalized DTMF frequencies.

Another positive feature of the DFT is that the X(k) values can be computed as soon as each sample arrives, making buffering unnecessary. Although this approach still evaluates fixed blocks of data, some new solutions have been proposed for moving-window DFTs [4].

From a mathematical point of view, the DFT theory [10] assumes that the window of samples under analysis is repeated from minus infinity to plus infinity, therefore k must be an integer to ensure that an integral number of cycles of W_N^{nk} is enclosed in the window. This constraint may be relaxed, as discussed later.

3. THE GOERTZEL ALGORITHM

The Goertzel algorithm exploits the periodicity of the rotating vectors W_N^{nk}, allowing to evaluate the DFT as a linear filtering operation (for a complete mathematical demonstration see [10]).

This solution replaces the DFT by the computation of a resonant recursive IIR filter (Eq.3), with two poles on the unit circle:

$$X(k) = \frac{1 - W_N^k.z^{-1}}{1 - 2.\cos(2\pi k/N).z^{-1} + z^{-2}} \qquad (3)$$

The direct form II realization of this filter leads to the following two difference equations:

$$v_k(n) = x(n) + 2.\cos(\frac{2\pi k}{N}).v_k(n-1) - v_k(n-2) \qquad (4)$$

$$y_k(n) = v_k(n) - v_k(n-1).e^{-j.2\pi k/N} \qquad (5)$$

Where two important points may be noted:

- the coefficient of $v_k(n-1)$ is real, hence no complex products are required
- $v_k(n)$ doesn't depend of $y_k(n)$ values, consequently $y_k(n)$ only needs to be computed once, when n=N.

From the point of view of the hardware implementation, (Figure 1) each Goertzel step (Eq.4) only involves a 3-terms addition plus one real multiplication; buffering requirements are also small, since only $v_k(n-1)$ and $v_k(n-2)$ must be saved.

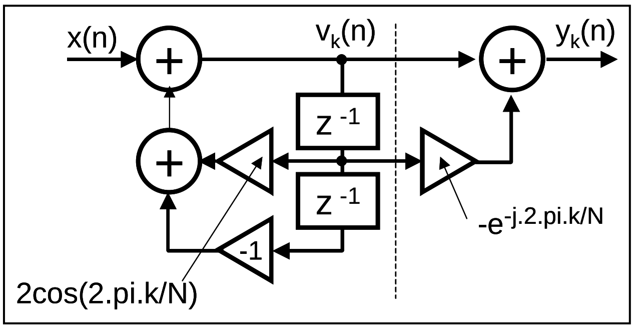

Figure 1: Hardware implementation

If only the tone presence must be detected, the phase component of $y_k(n)$ becomes irrelevant, and it squared amplitude may be computed instead. This leads to the "Modified Goertzel Algorithm" [5][6][7][8], where:

$$|X(k)|^2 = y_k(N).y_k^*(N) =$$
$$v_k^2(N) + v_k^2(N-1) - 2\cos(\frac{2\pi k}{N})v_k(N).v_k(N-1) \qquad (6)$$

The final computation of Eq.6 involves 4 products and 3 additions, and may be evaluated outside the programmable device, since the required data-throughput is very low, only once after N samples.

4. DESIGN CRITERIA

This section analyzes the design criteria for the N and k values, and the numeric resolution needed for the numeric computations.

4.1. About N

As a result of the limited number of samples used, the spectrum of the incoming sampled signal is convoluted with the spectrum of the rectangular sampling window. This convolution spreads the energy of each incoming tone X_{TONE} around the original frequency f with a sinc() distribution, as follows:

$$X_{SPREAD}(f \pm \Delta f) = X_{TONE}(f).\frac{\sin(\frac{\pi.\Delta f.N}{f_S})}{\frac{\pi.\Delta f.N}{f_S}} \qquad (7)$$

This scattering is small for larger values of N, since the first zero of Eq.7 are found when $\Delta F = f_{SAMP}/N$, and outside this region all other lobes are attenuated more than –14dB (Figure 2). This behavior must be considered to satisfy the ITU constraints for frequency tolerance.

The other restriction is related to the window size. The ITU specifies that the minimum tone duration may be 40ms (320 samples @8000samples/sec); as a consequence the maximum window should be smaller than 20ms (160 samples) to guarantee that the complete window is enclosed within this tone, and the shorter DTMF tones are correctly detected. An improved solution may require two consecutive windows of 13.3ms (106 samples) to detect valid tones, giving a way to discard tones shorter than 23ms.

Although the usual DSP algorithms don’t use this feature, a different N may also be used for each frequency. In this case, since the Goertzel output v(n-1) and v(n-2) increases with N, and $|X(k)^2|$ with N^2, these facts must be considered when amplitude and twist are evaluated or the hardware is sized.

4.2. About k

Although the DFT states that k must be integer, this constraint may be relaxed using a fractional value $k = f_{TONE}/f_{SAMP}$. In this case the meaning of Eq.1 is that of a correlation between the input tone samples f_{IN} and those of a reference complex sinusoid with frequency f_{TONE}; this correlation has a shape $\frac{1}{2}(1+\cos(\Delta F.N/f_{SAMP}))$ with a first zero when $\Delta F = \pi.f_{SAMP}/N$.

In figure 2 the combined N and k effects are evaluated together, using $\Delta F.N/f_{SAMP}$ units for the X axis. The shape of the main lobe is very similar to this of the sinc function, with the first zeros for $\Delta F = f_{SAMP}/N$; however, due to –4 dB added by the 1+cos function, secondary lobes are now atenuated more than -18 dB,.

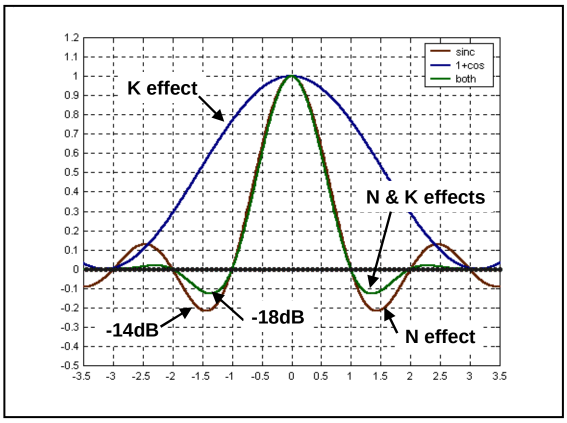

Figure 2: N and k effects

If only integer K_i values are used the maximum peak of the 1+cos function becoming displaced relative to the f_{TONE} position, adding undesirable attenuation.

Table 3: Fractional k values

Tone	k	Tone	k
697	1.7077	1209	1.1641
770	1.6453	1336	0.9964
852	1.5687	1477	0.7986
941	1.4782	1633	0.5685

Table 3 shows the fractional k values required for exact detection of the DTMF tones.

4.3. About the numeric resolution

The Goertzel values v(n-1) and v(n-2) are a function of N, the matching of f_{IN} with f_{TONE}, and the amplitude of f_{IN}.

By discrete simulation, the values shown in Table 4 were computed for N ranging from 100 to 215, and f_{IN} matched to the filter with amplitude A=0.5.

The analysis of Table 4 and the ITU constraints defines how data should be managed:

- The absolute maximum value for v(n) is proportional to N, with a range smaller than ±63 for N=100 and smaller than ±127 for N=215. These maximum are found for the lower and upper f_{IN} frequencies
- $|X(k)^2|$ increases proportionally to N^2. If the f_{IN} amplitude is held constant, $|X(k)^2|$ is not affected by changes of the input frequency.
- Signal to Noise ratio: the minimum 15 dB SNR is achievable using 8 fractional bits. If a two-tone signal with twist of 8 dB is considered, the resulting 23 dB means a 14:1 relation (less than 4 bits).
- Talk-off (30dB) means that the 2nd harmonic amplitude must be lower than 1/32 (5 bits) of the fundamental.

- Signal strength (-26dB minimum) is usually managed inserting an AGC before entering to the detection stage.

Table 4: Numeric simulations

f_{IN}	$v_{MAX}\|_{N=100}$	$v_{MAX}\|_{N=215}$	$\|X(k)^2\|_{N=100}$	$\|X(k)^2\|_{N=215}$
697	45	100	608	2857
770	42	93	609	2852
852	38	83	609	2895
941	35	75	613	2870
1209	29	66	615	2889
1336	27	55	611	2887
1477	26	58	625	2861
1633	25	55	623	2881
1394	25	58	625	2890
1540	25	56	612	2882
1704	23	53	614	2866
1882	24	51	624	2881
2418	25	56	613	2887
2672	26	61	617	2860
2954	31	72	617	2865
3266	44	94	607	2856

As a consequence of the previous points, an AGC controlled f_{IN} may be processed using 16 bits arithmetic, with 8 bits at the left of the radix point and 8 bits as the fractional value. Anyway, 24 bits (8 bits and 16 fractional bits) could be preferable is strict conformance to the talk-off constraint is desired.

The bibliography shows very different approaches to satisfy all the constraints:

- In [3] N=212 is used for low-group filters and N=106 for high group filters. Since N=212 defines a window duration larger than desired, each low-group frequency is evaluated by two overlapped-in-time Goertzel filters, one offset from the other by 106 samples. It uses 14 filters, because 2nd harmonics are only evaluated for the two stronger DTMF frequencies detected in the previous slice.
- In [6] a single value of N=105 is used
- In [7] a single value of N=136 is used, but dual bins are used for the high-group frequencies to satisfy the Bellcore's accept constraints, using 20 filters.
- In [8], N=205 value is chosen for detection of the DTMF frequencies, and N=201 for the 2nd harmonics. This solution uses 16 filters.

Assuming fractional k and N=106, the Table 5 shows what levels should be used to accept or reject a given Goertzel filter output related to the peak value detected within the low or high band, and the relative difference between the accept/reject levels (ARR: Accept/Reject Relation).

If a 15dB SNR must be tolerated (15dB means that the noise level is 18% of the signal level, or 5.16:1 relationship), the detection or rejection of the low group tones becomes marginal, justifying the solution described in [d]. In this solution, the use of N=212 for low group tones enhances the ARR value from 18% to 117% for the 697Hz tone, with better values in the case of the other low group tones.

Table 5: Accept/Reject levels for N=106

	N=106		
Freq.	Accept	Reject	ARR
697	0.965	0.816	18%
770	0.957	0.779	23%
852	0.947	0.735	29%
941	0.936	0.683	37%
1209	0.896	0.516	74%
1336	0.874	0.435	101%
1477	0.847	0.345	146%
1633	0.815	0.252	223%

Figure 3 represents the signal levels for accept/reject High Group tones, using N=106, for frequency offsets of 1.5% and 3.5%.

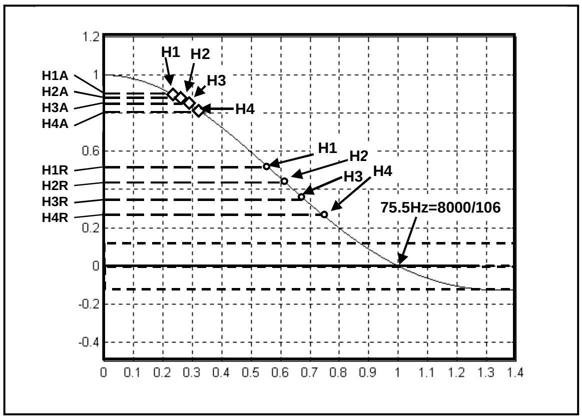

Figure 3: Accept/Reject levels for High Group Tones

Although a single threshold level may be used, different thresholds could be defined for the different tones to maximize the Accept/Reject confidence.

5. BAND SPLIT FILTER & DETECTOR

To verify that the incoming signal satisfies the strength, twist and frequency tolerance constraints, it is neccesary to split the signal spectrum to detect the signal level in the lower and higher bands.

The LPF filter must have a cut frequency greater than 941 Hz, and high-band tones with the worst twist should be attenuated to a level lower than noise (15+4=19 dB). In the HPF, the cut frequency must be lower than 1209 Hz, and attenuation of low-band tones should be better than 23 dB (15+8). The simplest solution is the use of two elliptic filters of order 4 (determined using `ellipord()`), with coefficients computed using the following MATLAB script, and response shown in Figure 4:

```
n=4; % filter order
[b,a]=ellip(n,0.5,23,0.235);        %LPF
[d,c]=ellip(n,0.5,23,0.302,'high'); %HPF
[H,q]=freqz (b,a,512); magl=abs(H);
```

```
[G,p]=freqz (d,c,512); magh=abs(G);
plot (q,magl,'k',p,magh,'b'); % plotting
```

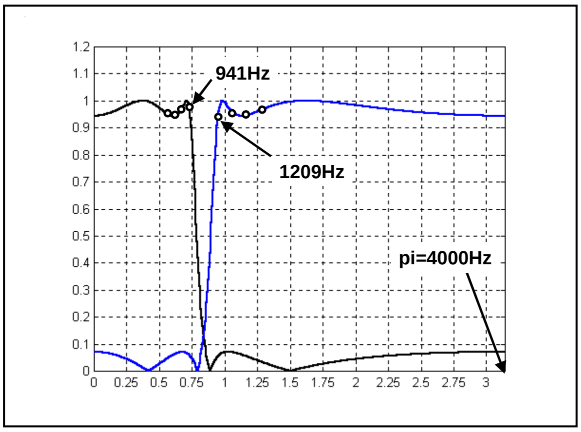

Figure 4: LPF and HPF response

This type of filter has ripple in the pass band. However, since DTMF tones are fixed, the passband attenuation for each frequency may be exactly computed (Table 6).

Table 6: Passband attenuation for DTMF tones

Low Pass Filter		High Pass Filter	
Frequency	Attenuation	Frequency	Attenuation
697	0.9592	1209	0.9274
770	0.9442	1334	0.9666
852	0.9643	1477	0.9449
941	0.9753	1633	0.9654

Zeros and poles of the filters can be found through the `roots()` function of MATLAB, using the 4th order coefficients of the numerator and denominator of each filter as arguments:

- The LPF has 4 zeros on 0.6306+/-0.7761i and 0.0676+/-0.9977i, and 4 poles on 0.6886+/-0.6469i and 0.6152+/-0.3278i.
- The HPF has 4 zeros on 0.7031+/-0.7111i and 0.9134+/-0.4070i, and 4 poles on 0.5621+/-0.7482i and 0.2619+/-0.4595i.

To guarantee numeric stability, each filter may be implemented as two cascaded biquad sections (see Table 7), computed with MATLAB using `tf2sos()`.

The detection of the LPF2 and HPF2 outputs can be implemented using an Absolute-value rectifier stage followed by Low-Pass filter. The worst case is for the LowGroup filter when the 425 Hz tone is evaluated, since the first AC component of the rectifier output that must be filtered is in 850Hz. In this case a simple moving average IIR, with Eq.8 is enough to obtain attenuation greater than 20db in 850Hz, with good numeric stability and a fast step response (36 samples to rise to 90% of the final value).

$$y(n)= (15/16)*y(n-1) + (1/16)*x(n) \qquad (8)$$

Table 7: Biquad sections of band splitter

LPF1	Numerator	$1.0 - 1.2616*z^{-1} + z^{-2}$
	Denominator	$1.0 - 1.3774*z^{-1} + 0.8925*z^{-2}$
LPF2	Numerator	$0.0903 - 0.0122*z^{-1} +0.0903*z^{-2}$
	Denominator	$1.0 - 1.2302*z^{-1} + 0.4859*z^{-2}$
HPF1	Numerator	$1.0 - 1.4062*z^{-1} + z^{-2}$
	Denominator	$1.0 - 1.1242*z^{-1} + 0.8758*z^{-2}$
HPF2	Numerator	$0.3919 – 0.716*z^{-1} + 0.3919*z^{-2}$
	Denominator	$1.0 - 0.5239*z^{-1} + 0.2797*z^{-2}$

6. AGC

The 26db variation range that ITU imposes for acceptable signal levels corresponds to more than 4 bits of resolution in the incoming signal. When fixed point hardware is used, a simple barrel shifter controlled by the most significant bits of the splitter detector may be used to reduce this range to only 6dB (1 bit). This AGC is different to any AGC used to manage voice levels, since the attack/decay times must agree with DTMF characteristics.

7. A COMPLETE DTMF DETECTOR

The suggested structure for a complete DTMF detector shown in Figure 5 closely resembles the solution described in [3]:

- The incoming signal is level adjusted by the AGC stage, feeding the LPF and HPF band splitter filters.
- The output of both filters is rectified and smoothed to measure the signal levels in Low and High bands.
- The maximum level of both meters is also used to control the AGC gain.
- The Low band signal feeds 8 Goertzel filters used to detect the Low Group DTMF tones. In this case N=212, and each tone is evaluated by two overlapped-in-time Goertzel filters, one offset from the other by 106 samples.
- The High band signal feeds 6 Goertzel filters, with N=106; these filters are used to evaluate the High-Group tones plus the 2nd harmonics of the two stronger DTMF tones detected in the previous slice. The 1633Hz filter may be avoided if only keyboard tones must be detected.
- One additional Goertzel filter with N=106 is also fed by the Low band signal to measure the Call Progress tone.

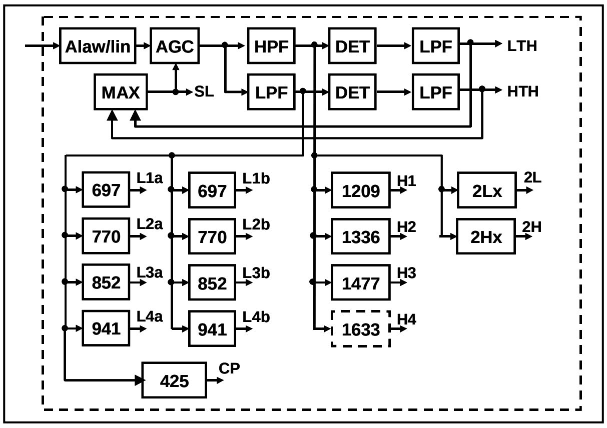

Figure 5: Complete DTMF detector

8. DETECTION ALGORITHM

The DTMF detection process involves a hierarchical set of tasks, solved in different moments:

1. During each frame
   - Compute the Band-Splitter and Detector filters
   - Adjust AGC level
   - Compute the 15 Goertzel filter
2. Once after 106 frames:
   - Check SL: Is the signal level > -26dB?
   - Compare HTH with LTH: Is the difference in the acceptable twist range? +4dB..-8dB
   - Select the 4 low-group filters that end this slice:
     - Select the strongest tone. Is this tone within the Accept level, when compared with LTH?
     - Select the second-in-strength tone. Does it satisfy the talk-off relationship?
   - Select the 4 high-group filters:
     - Select the strongest tone. Is this tone within the Accept level, when compared with HTH?
     - Select the second-in-strength tone. Does it satisfy the talk-off relationship?
   - Initialize the 4 low-group filters that ended in this slice.
   - Initialize the 4 high-group filters.
   - Initialize and select the two 2nd harmonics filters, based on the strongest low and high tones detected.
3. Once after two successive Low Group and High Group valid tones:
   - Does the strongest Low-Group satisfy the 30dB talk-off relationship with it 2nd harmonic?
   - Does the strongest High-Group satisfy the 30dB talk-off relationship with it 2nd harmonic?

9. MAC ARCHITECTURES BASED ON CPLDS

Distributed arithmetic can be used whenever an efficient MAC (Multiply and Accumulate) operation that uses constant coefficients is needed [5]. The MAC operation is defined with Eq.9 where $x_0..x_N$ is the set of N+1 samples and $A_0..A_N$ the N+1 constants coefficients.

$$MAC(x,A) = \sum_{i=0}^{N} x_i A_i = x_0 A_0 + x_1 A_1 + .. + x_N A_N \quad (9)$$

If samples and coefficients are represented with M+1 bits, and calling x_{ij} to the bit j of the sample i, Eq.9 may be decomposed at the bit level, as follows:

$$\begin{aligned} &MAC\,(x,A) = \\ &x_{00}.A_0.2^0 + x_{01}.A_0.2^1 + ... - x_{0M}.A_0.2^M + \\ &x_{10}.A_1.2^0 + x_{11}.A_1.2^1 + ... - x_{1M}.A_1.2^M + ... + \\ &x_{N0}.A_N.2^0 + x_{N1}.A_N.2^1 + ... - x_{NM}.A_N.2^M \end{aligned} \quad (10)$$

Which can be reordered as:

$$\begin{aligned} &MAC\,(x,A) = \\ &2^0.(x_{00}.A_0 + x_{10}.A_1 + .. + x_{N0}.A_N) \\ &+ 2^1.(x_{01}.A_0 + x_{11}.A_1 + .. + x_{N1}.A_N) + ... \\ &- 2^M.(x_{0M}.A_0 + x_{1M}.A_1 + .. + x_{NM}.A_N) \end{aligned} \quad (11)$$

If the Eq.11 is solved with serial arithmetic, the following architecture is straightforward:

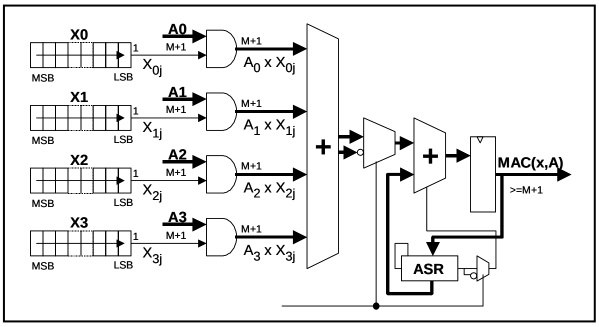

Figure 6: Serial MAC (picture based on [5])

Some details must be noted in Figure 6:

- The bit-width of the first adder could be up to 2 bits wider than M+1, because 4 terms are added.
- The bit–width of the MAC should add some bits to accommodate the complete addition of N+1 terms.
- Since input bits x_{ij} come serially, from LSB to MSB, and each new bit has twice the weight of the previous, each new product/sum must be added to the previous MAC value sign-shifted to the right. This is a hardware shifting.
- When the last input bit is computed (the sign bit) the output from the first stage and the Carry-input to the second stage must be negated.

If this solution is implemented with ALTERA FLEX10KE, ACEX 1K, APEX or STRATIX devices, many architectural features may be used for a more efficient design:

- An Embedded Array Block (EAB) of memory, with 5 inputs and M+3 outputs, can be used to compute the AND+ADD+INVERT function (BLOCK A).

- Logic Elements (LE) in Arithmetic mode can be used to implement a very fast ADDER+REGISTER (BLOCK B), using only one LE per bit.

The resulting solution is shown in Figure 7:

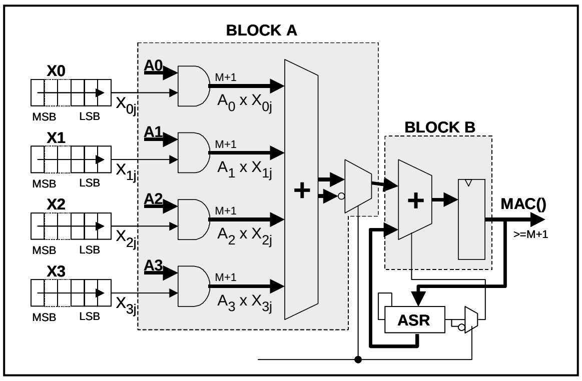

Figure 7: Serial MAC using ALTERA (based on [5])

This is only a general solution, and many architectural features may be greatly improved when a specific solution must be solved.

10. CPLD BASED GOERTZEL COPROCESSOR FOR DTMF DETECTION ON E1 TRUNKS

An E1 trunk is used to transport up to 32 simultaneous voice channels over a serial link (in fact, only 30 channels are used for voice and the two others for synchronization and signaling [1]). Each voice channel is allotted a time-slot and requires 8 companded bits, thus the 32 channels are arranged as 32 consecutive bytes, forming a 256 bits frame. Each frame is repeated 8,000 times/sec, so the whole bit-rate is 2.048 Mbit/sec.

For DTMF detection, the most intensive task is the computation of the Goertzel filters: if 15 filters must be evaluated for each of the 30 voice channels during each frame of 125µs, that leaves only 277ns for each of the 450 filters. This constraint precludes any microcontroller based solution, and even in the case of a 80 MIPS DSP, there is time for no more than 22 single-cycle instructions to compute each filter.

This intensive task is highly repetitive, and a CPLD based approach may be used to do it.

If fixed-point, 16 bits serial arithmetic is used, and assuming that 4 additional clocks are needed for registers load/unload, only 13.8ns remain available between clocks; these assumptions imply a computing clock of 72MHz, achievable even with the slower ACEX 1K devices (e.g.: EP1K30TC144-3).

The proposed solution (Figure 8) is built around 8 functional blocks:

- **S(n)**: a Parallel input/Serial Output re-circulating shift register where the input sample is stored. It is re-circulating because the same sample is used 15 times to evaluate the different filters.
- **V(n-1)** (and **V(n-2)**) Parallel input/Serial Output shift registers where v(n-1) (and v(n-2)) values are temporarily loaded for processing.
- **LUT**: Lookup Table based on a single EAB, configured as a 256x16 registered ROM, that stores the different partial product/sum for each combination of the bits of v(n-1), v(n-2) and s(n), for each filter. The LAST signal is used to negate this value, as required in the last step.
- **SHIFT/ADD/STORE (SAS)**: as described in the previous section, this is a 24-bit registered adder implemented using one LE per bit. Although only the upper 16 bits are used for input/output of data, the additional 8 lower bits avoid the propagation of truncation errors.
- **SELECT (SEL)**: it is a 16-stage 2:1 multiplexer, used to chose between the value stored in the V(n-1) register, or the SAS output.
- **RAM**: a 1024x16 RAM memory, built using 4 EABs, where two 16 bit words (v(n-1) and v(n-2)) are stored for each of the 450 filters.
- **CONTROL**: global timing and control block.

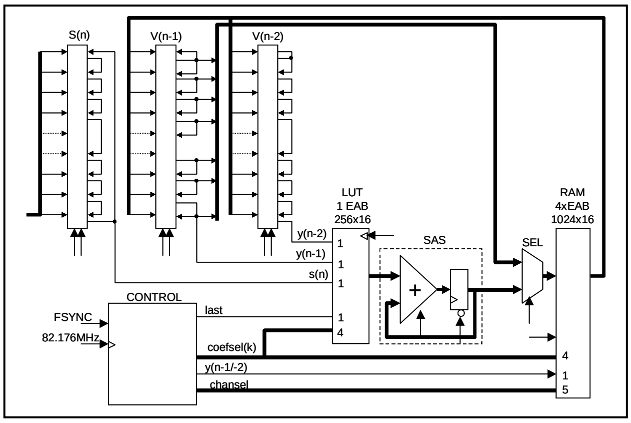

Figure 8:. Goertzel Coprocessor for DTMF detection

These building blocks operate as follows:

For each time-slot, S(n) is loaded with a linearized value of the PCM sample.

For each of the 15 filters:

cycle 1: RAM is addressed for reading v(k,n-1,ch(i))
cycle 2: V(n-1) is written with v(k,n-1,ch(i)).
RAM is addressed for v(k,n-2,ch(i))
SEL selects V(n-1) as the RAM data source
cycle 3: V(n-2) is written with v(k,n-2,ch(i)).
RAM address v(k,n-2,ch(i)) is written with the V(n-1) value.
The SAS register is cleared
cycles 4..19: the LUT output is registered
The new v(k,n-1,ch(i)) is computed
S(n), V(n-1) and V(n-2) are shifted.
SEL selects SAS as the RAM data source
RAM is addressed for writing v(k,n-1,ch(i))
cycle 20: RAM address v(k,n-1,ch(i)) is written with the SAS output value.

11. CONCLUSIONS

The last section describes a CPLD based solution for the GOERTZEL filters. However, a complete DTMF detector as shown in Figure 5, needs other functional blocks. These blocks are also CPLD suitable and its hardware requirements can be resume as:

- **A-law to linear conversion.** Fully combinatorial block, only requires 23 macrocells. [12].
- **AGC.** Fully combinatorial block, if the AGC range is limited to 4 bits (24dB) the decision logic may be solved with only 6 macrocells, adding 24 macrocells for a two-stage barrel-shifter.
- **Band splitter.** LPF and HPF filters share the use of the two delayed inputs, having each one two different delayed outputs, and can be computed concurrently using a serial architecture similar to the Goertzel case. In this case, data coming from 7 shift registers (112 macrocells) feed two 5-input table (2x3x16=96 macrocells) followed by two accumulators / shifters (other 32 macrocells). Each detector/filter needs an ABS (absolute value) stage, a register for the previous value and an actual value accumulator (2x48=96 macrocells). Each channel needs 8 words (2 for the LPF/HPF delayed inputs, 2 for the LPF delayed outputs, 2 for the HPF delayed outputs, and 2 for the detector filter), which can be stored in a simple EAB configured as 256x16. The required number of read/write cycles (10) plus the computing cycle (16) for each channel mean a processing frequency of only 6 MHz.

A complete CPLD based solution to DTMF detection is feasible and comparable with known solutions.

11. REFERENCES

[1]. Ericsson, *Telecommunications: Telephone Networks 1*, ISBN 91-44-24521-1, Sweden, 1986

[2]. ITU Blue Book, *Recomendations Q.24: Multi-Frequency Push-Button Signal Reception*, Geneva, 1989.

[3]. Felder M.D., Mason J.C. and Evans B.L. “Efficient Dual-Tone Multi Frequency Detection using the Non-Uniform Discrete Fourier Transform”, IEEE Signal Processing Letters, vol. 5, no. 7, pp. 160-163, July 1998.

[4]. Rosendo Macías J.A. and Gómez Expósito A. “Efficient Moving-Window DFT Algorithms”, IEEE Transactions on Circuits and Systems – II Analog and Digital Signal Processing”, vol.45, n0.2, pp. 256-260, February 1998.

[5]. Dulik T. “An FPGA Implementation of Goertzel Algorithm”, MOSIS'99 proceedings, vol. 2, pp. 35-42, ISBN 80-85988-33-X, 1999.

[6]. Chen C.J. “Modified Goertzel Algorithm in DTMF detection using the TMS320C80”, Application Report SPRA066, Texas Instruments, USA, June 1996.

[7]. Schmer G. “DTMF Tone Generation and Detection: An implementation using the TMS320C54x”, Application Report SPRA096A, Texas Instruments, USA, May 2000.

[8]. Analog Devices, *Digital Signal Processing Applications using the ADSP-2100 Family, volume 1*, Prentice Hall, pp. 441-500, ISBN 0-13-219726-X, USA, 1992.

[9]. Analog Devices, *Digital Signal Processing Applications using the ADSP-2100 Family, volume 2*, Prentice Hall, pp. 481-502, ISBN 0-13-178567-2, USA, 1995.

[10]. Proakis J.G. and Manolakis D.G. *Digital Signal Processing, 3rd. Edition*, Prentice Hall, pp. 480-481, ISBN 0-13-373762-4, 1996.

[11] Wilton S., Saleh R. “Programmable Logic IP Cores in SoC Design: Opportunities and Challenges”, Proc. IEEE Custom Integrated Circuit Conference, San Diego, CA., pp. 63-66, May 2001.

[12] Jaquenod G. “Building blocks for telephony applications: A-law conversion”, Unpublished paper, 2002.