

## DTMF Detector

By: Victor Kremin  
 Associated Project: Yes  
 Associated Part Family: CY8C26443

### Summary

The proposed Application Note describes the implementation of a dual tone multiple frequency (DTMF) detector.

### Introduction

DTMF signalling is widely used in analog telephone dialing, data entry, voice mail systems, remote control of various consumer electronics (auto answering machines, home automation devices, bank information services, etc.).

The DTMF signal consists of sum of two sinusoidal waveforms with predefined frequencies, which were selected according to the ITU Recommendations Q.23 [1] and Q.24 [2]. A user can interact with DTMF dialer using keypad with sixteen characters. Figure 1. illustrates the keypad layout and corresponding signal frequencies.

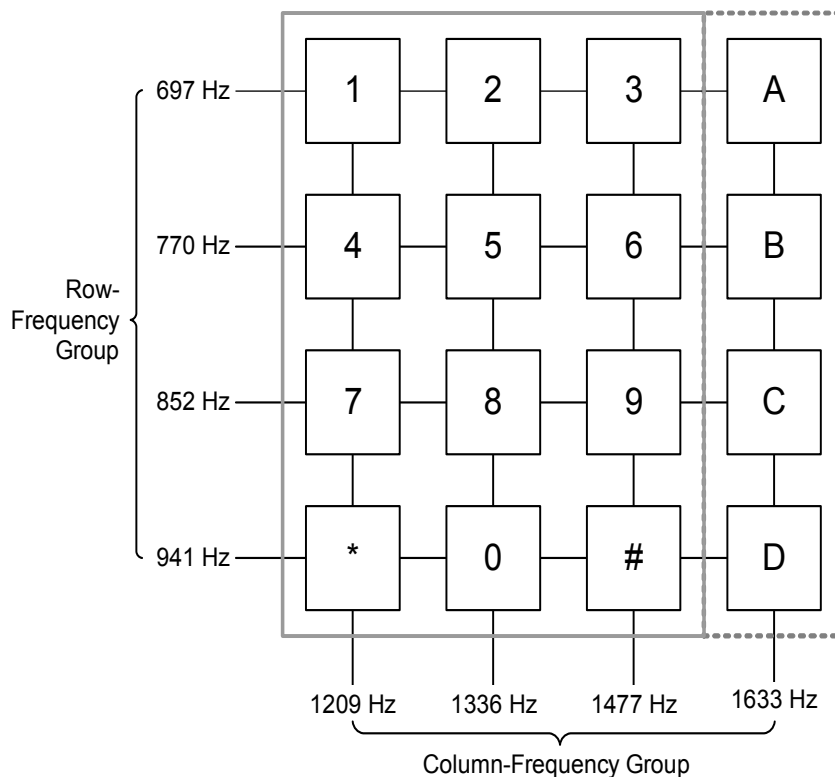


Figure 1. The Touch-Tone Phone Keypad

The conventional phones contain the keys '0'-'9', '\*', '#' typically, the keys 'A'-'D' can be found in the industrial controllers and are used especially for remote control. There are eight different frequencies, grouped into rows and columns. Each character is associated with pair frequencies from row and column groups.

The DTMF decoder must determine which frequencies are present in the incoming signal and correctly decode the corresponding character. Table 1 illustrates the key decoder ITU specifications. The full DTMF signalling specifications can be found in [1].

**Table 1. DTMF Decoder ITU Specifications**

<b>Frequency Tolerance</b>	The frequency tolerance about $\pm 1.5\%$ is allowed for valid DTMF tone. The tone with offset $\pm 3.5\%$ must be rejected.
<b>Signal Duration</b>	A valid signal with duration of 40 ms must be considered as valid. The tones with duration 23 ms or less must be rejected.
<b>Signal Interruption</b>	A valid DTMF signal interrupted for 10 ms or less should not be detected as two distinct tones .
<b>Signal Pause</b>	A valid DTMF signal separated by 40 ms pause or more must be detected as two distinct characters.
<b>Signal-to-Noise Ratio (SNR)</b>	The DTMF detector must correctly process signal with SNR 15 dB.
<b>Tones Twist</b>	The detector must correctly decode DTMF codes when row frequency signal is 8 dB larger than column frequency signal. The detector must also operate correctly when column frequency signal is 4 dB larger than row signal.
<b>Talk-Off</b>	The detector should operate in the presence of speech and music without incorrectly identifying the speech signal as valid DTMF signals. To evaluate the decoder speech resistance, the decoder can be tested using BellCore test tapes.

There are numerous DTMF decoder design approaches to build DTMF decoders. The typical decoder consists of front-end signal processing system and back-end logic system to make conclusions concerning presence or absence DTMF character.

The most popular algorithms for signal processing front-end are based on Discrete Fourier Transform (DFT) [7] modifications. To reduce the number of calculations, the coefficients are calculated by using modified Goertzel algorithm [8], which requires only  $N+2$  multiplications and  $2N+2$  additions to process data for every  $N$  samples. The modified Goertzel algorithm calculates the squared absolute value of DFT coefficients at extract row/column frequencies via second-order IIR filter:

$$s_k[n] = 2 \cos\left(2\pi \frac{f_k}{f_s}\right) \cdot s_k[n-1] - s_k[n-2] + x[n] \quad (1)$$

...where  $s_k[n]$  -  $k^{\text{th}}$  filter response for  $x[n]$  input sample,  $f_k$  and  $f_s$  -  $k^{\text{th}}$  filter corner and sample frequencies,  $s_k[-1] = s_k[-2] = 0$  and  $n = \overline{0..N}$ .

The squared absolute value of DFT coefficient  $|Y(k)^2|$  value can be calculated from two last filter responses:

$$|Y(k)|^2 = s_k^2[N] + s_k^2[N-1] - 2 \cos\left(2\pi \frac{f_k}{f_s}\right) s_k^2[N] \cdot s_k^2[N-1] \quad (2)$$

In spite of the fact that this algorithm is widely recommended by application notes from various DSP chipmakers, the work [3] shows the DFT-based detection algorithms are not strictly ITU compliant because the passband for row frequencies does not satisfy the ITU requirements. The algorithm can be improved by using non-uniform DFT and dedicated back-end logic [4]. This implementation is computation intensive and DSP orientated. To meet ITU specifications strongly, some DTMF detectors use DFT coefficients calculation for shifted row/column frequencies also and sophisticated decision logic.

To improve the speech-resistance, DTMF decoders are often equipped with second-harmonics estimators. Speech and music is characterized by rich even harmonics; DTMF signal is characterized by low level of these harmonics. The speech estimator analyzes the second harmonics level and compares results with fundamental frequency levels to detect speech/music and predict the false DTMF-signal detection.

The other effective decoder operation principles are based on resonant filtering using digital wave filters [5], which are characterized by low sensitivity to rounding filter coefficients, but each sample requires several 16-bit multiplications and additions with output normalization to provide reliable filter operation. So, this decoder is better oriented for 16-bit CPUs or DSPs.

A very attractive decoder oriented on low-cost 8-bit processors is suggested in a recent paper [6]. The decoder uses pair-adaptive notch filters to suppress one tone in the incoming signal and a direct frequency estimator to measure the frequency of other tone. This decoder meets all ITU specifications (both in time and frequency domains) and is characterized by very low speech resistance. The Bellcore talk-off test results are at the upper limit of allowable bounds and are much worse than a good DFT-based decoder with second harmonics analysis. The decoder consumes about 4.5 MIPS on an 8-bit CPU.

By author opinion, the optimal way to build a good DTMF decoder lies in using adaptive resonant/band-pass filters to suppress unwanted noise signals together with direct or indirect filter-output frequency estimation. The filter's (both for fundamental DTMF frequencies and second harmonics) output level analysis allows meeting all ITU signal level specifications together with reliable speech detection and direct frequency measurement provides satisfactory ITU frequency specifications. The decoder, which utilizes this principle, will be described in a separate Application Note.

### DTMF Decoder Operation

During the development of this decoder I wanted to build inexpensive DTMF decoder that will satisfy most ITU requirements and consume relatively low CPU resources to be operational in background. The decoder must be stable to speech signal and must incorporate second harmonics analysis to distinguish speech and music from a valid DTMF signal. The first time I planned to employ the modified Goertzel algorithm. The MATLAB simulations showed that 16\*16 bit multiplication with output normalization must be used to get stable IIR filter operation together with accurate filter corner frequency setting. Each Goertzel filter consumes about 400 CPU cycles or 17 us at 24 MHz for each sample processing using the macros for multi-byte multiplication, PSoC MAC [9]. To incorporate second-order harmonics analysis, 16 filter values must be calculated per sample and sample rate must not be lower than 7.8 kHz to satisfy the Nyquist criterion. A particular solution could be to use decimator for fundamental frequency filters or reject the second harmonics analysis but I decided this way is not very good here. So, the other approach must be used.

The proposed decoder incorporates the quadrature correlator for fundamental DTMF frequencies and their second harmonics. The correlator correlates the incoming signal with two quadrature signals, which correspond to the sign function of sine and cosine signals of one fundamental DTMF frequency or its second harmonic. The advantage of this correlator is the absence of multiplication operations; the correlator performs addition when the reference signal is positive and subtraction when the reference signal is negative.

After processing the N samples, the quadrature components are squared to eliminate incoming signal-phase dependence and ready to follow processing by back-end logic.

The following formulas illustrate the correlator operation:

$$vs_k[n] = vs_k[n-1] + \text{sign} \left[ \sin \left( 2\pi \frac{f_k}{f_s} n \right) \right] \cdot x[n] \quad (3)$$

$$vc_k[n] = vc_k[n-1] + \text{sign} \left[ \cos \left( 2\pi \frac{f_k}{f_s} n \right) \right] \cdot x[n]$$

$$Z_k^2 = vs_k^2[N] + vc_k^2[N] \quad (4)$$

...where  $vs_k[n]$  and  $vc_k[n]$  sine and cosine correlator components for  $k^{\text{th}}$  frequency,  $f_k$  the DTMF or second harmonic  $k^{\text{th}}$  frequency, and  $f_s$  is sampling frequency.  $vs_k[0] = vc_k[0] = 0$ ,  $n = \overline{1..N}$ , and  $k = \overline{0..15}$ .  $Z_k^2$  is squared correlator output.

A simulation model was constructed using MATLAB.

Figure 2 shows the simulated correlator response  $Z_k^2$  for DTMF signal also a function of correlation frequency  $f_k$  and sample length N. The correlator model uses 16-bit, fixed-point arithmetic and the output response is divided by  $2^{16} \cdot N^2$  for normalization and scaling. The DTMF signal consists of sum signals with frequencies 941 Hz and 1209 Hz that is corresponding '\*' character; the sample rate is 8518 Hz. The sample rate is larger than commonly used in telephone systems 8000 Hz, which was utilized to simplify the antialiasing filter.

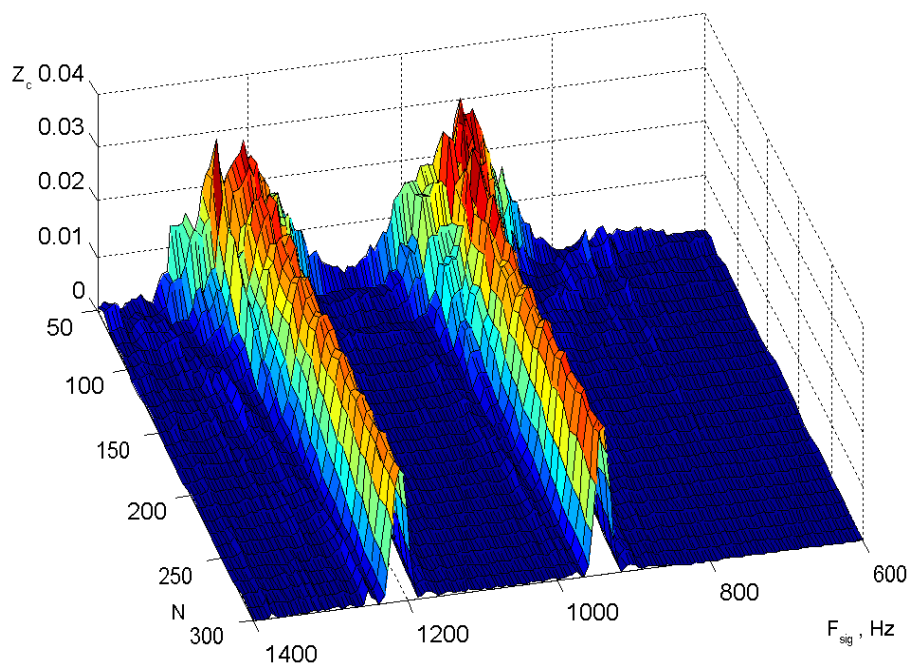
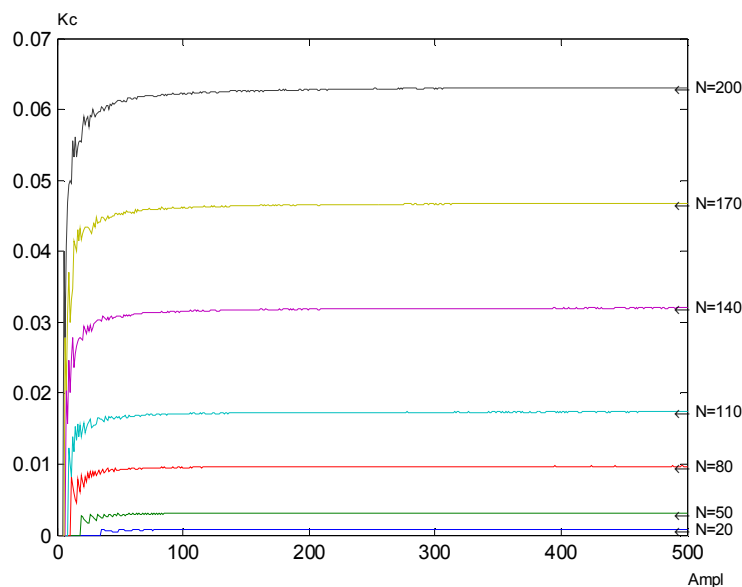


Figure 2. Correlator Response Simulation

The simulation results show the correlator will satisfy most ITU frequency specifications for column frequency group for  $N=127$  which corresponds accumulation time 15 ms and will satisfy the minimum DTMF signal-level detection demands.

For row frequencies, the allowable bound is larger. Because the correlator uses output result division on  $2^{16}$  for output data normalization, the low limit of the input signal must be evaluated as well. Figure 3. shows the simulated normalized correlator transfer characteristics as a function of signal amplitude and N.



**Figure 3. Correlator Transfer Characteristics**

Figure 3. shows that the transfer characteristic is virtually linear when DTMF signal code's amplitude is greater than 15. If an 8-bit ADC is used, the input signal should be multiplied with a gain of 4 to increase the dynamic range of the input signal. Under this condition, the estimated correlator dynamic range for input signal will be more 25 dB, which meets ITU specifications.

## Decoder Implementation

### Signal Processing Software Front-End

The DTMF decoder is implemented largely in software using both interrupt and main techniques. The signal processing front-end subsystem is located in an 8-bit sigma-delta ADC Interrupt Service Routine (ISR) and decision back-end logic is placed in the main program loop.

The ADC ISR performs the following tasks:

- correlation sums updating for fundamental DTMF frequencies and their second harmonics;
- incoming signal energy calculation using PSoC MAC;
- correlation sums squaring and normalization using PSoC MAC;
- input amplifier gain adjustment;
- synchronization with back-end decision logic.

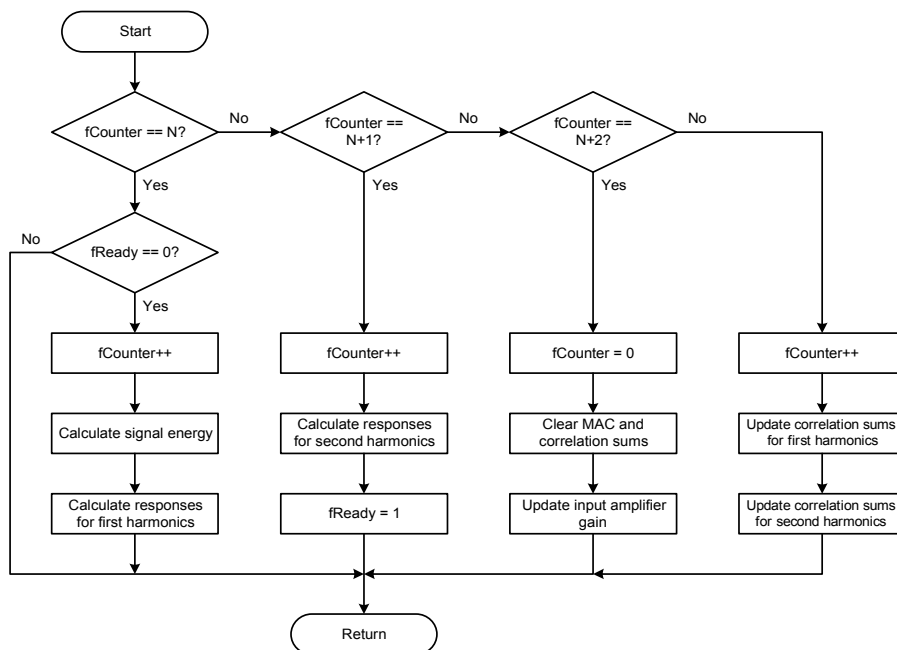


Figure 4. ADC ISR Structure

Figure 4 illustrates the ADC ISR structure. A simple state machine is implemented in the ADC ISR to process these tasks. When this routine is called, the *fCounter* variable is checked first. When the correlator has performed processing of *N* (where *N* is sample length) input samples, the *fCounter* is equal to *N*. Then, the *fReady* flag is checked. This flag is used to mark that previous calculation results have been read by the back-end logic and can be overwritten now.

If this flag was cleared, the *fCounter* is incremented and the correlation sums for fundamental DTMF frequencies are squared and normalized. Note that the correlation sums squaring and normalization code was placed in the ISR to save 32 bytes of RAM for user purposes. Moreover, because the ISR is using the PSoC MAC for incoming signal calculation, there is sense to use this MAC unit here to avoid resource-sharing problems. In addition, the correlator output calculation was distributed into several sequential interrupts to balance the interrupt latency.

The following interrupt performs the correlation results calculation for second harmonics and sets the *fReady* flag. Now the correlator data is completely ready and can be read by the back-end logic subsystem. This subsystem must read the correlation results and clear the *fReady* flag more often than the correlator accumulation period, or 15 ms in our case, to avoid the correlator stalls. The next ADC interrupt initializes the correlation sums, MAC, clears *fCounter* variable and updates the input PGA gain value if automatic gain-control loop is used. After, the correlator is ready to process the next *N* ADC samples.

#### Decision Logic Back-End

The back-end decision logic is more complicated and is implemented as a subroutine, which is periodically being called by the main program loop. The back-end logic routine implements the following tasks:

- performing the various validity checks;
- parsing the incoming data stream into separate DTMF characters;
- calling the automatic gain-loop routine, if needed.

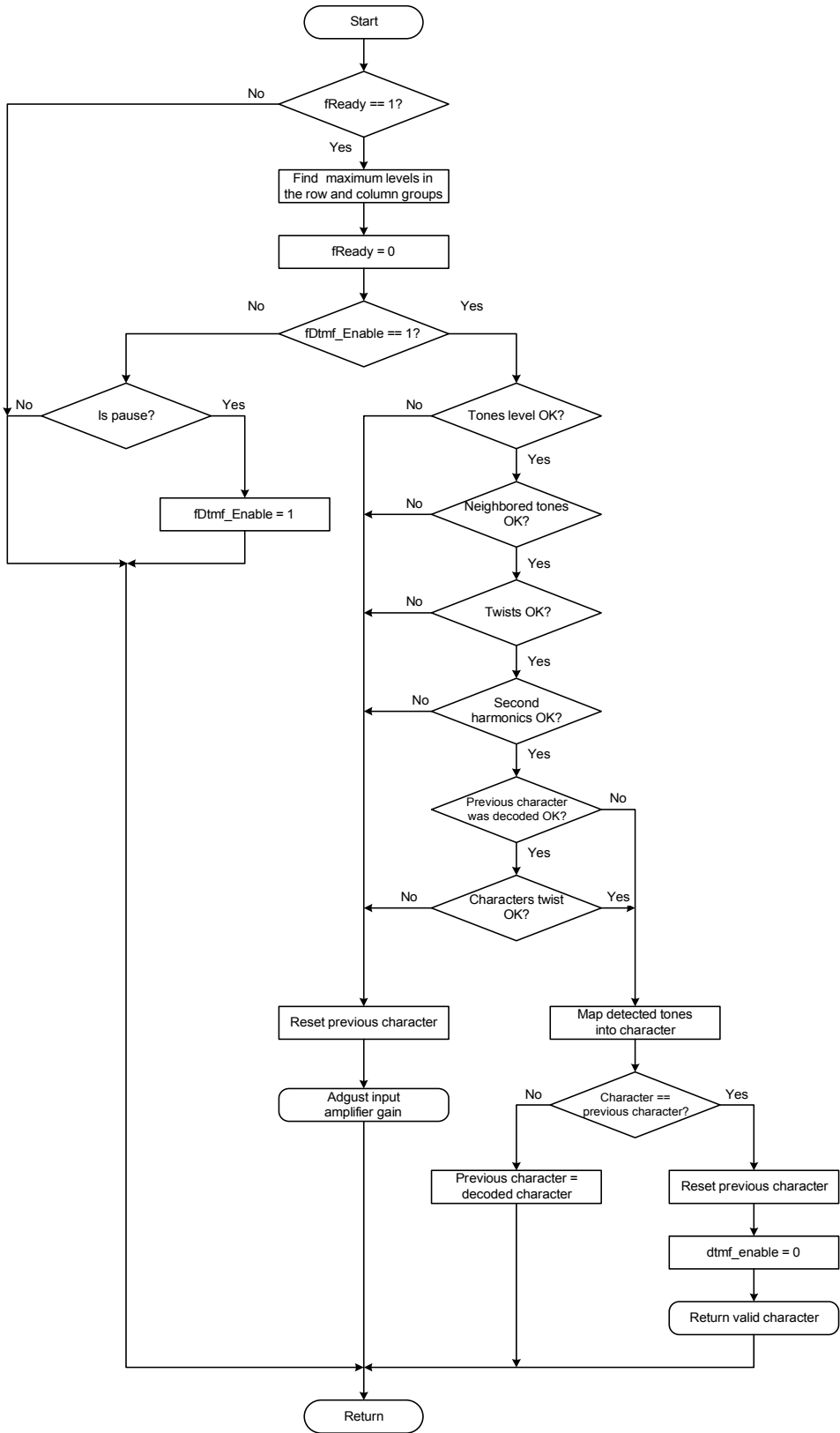


Figure 5. Decision Logic Algorithm

Figure 5. illustrates the decision logic operation. The *fReady* flag is checked initially. When the correlator data is ready, the tones with maximum level from row and column frequency groups are found. The *fReady* flag is cleared now to allow calculation of the next correlator data and avoid possible correlator stalls. The back-end logic subsystem can be set into two modes: awaiting valid DTMF tone signal and pending signal pause. The *fDtmf\_Enable* flag is used to switch between these modes. The *fDtmf\_Enable* flag is set to enable the waiting for DTMF tone mode.

To detect valid DTMF tone, a series of checks is performed. The first check is signal strength. The sum of squared amplitudes of the peak tones from row and column frequency groups must be greater than a pre-determined threshold. Because there are possible twists (differences in level between high and low tones) between row and column peak tones, the sum estimation works better than comparing the row and column tone levels separately.

The second check is adjacent tones strength analysis. The strongest row/column tone must stand out from its adjacent tones within its group by more than a pre-set threshold ratio. To avoid overflow, the peak tone levels are scaled and compared with its adjacent tones.

The third check is tones twist estimation. There are two possible twists types in the incoming DTMF signal. Assuming the telephone channel is low-pass filter, the high frequency tone is usually received with lower amplitude than low frequency tone, which is called "normal twist." To overcome this twist, the DTMF generators typically amplify the high frequency tone level. If telephone channel gain-frequency characteristic is near to linear, the high-frequency column tone can be larger than low frequency row tone. This phenomenon is called "reverse twist." The decoder must accept DTMF signals with 8 dB row/column amplitude relation for "normal twist" and 4 dB for "reverse twist." The decoder checks relation between peak row/column tones to detect the both "normal" and "reverse" twists.

The fourth check is the second harmonics strength check. It is assumed the DTMF signal contains only fundamental tones of its odd harmonics (if telephone channel amplifiers were saturated under incorrect signal levels). Speech and music have significant level of even harmonics added to fundamental frequencies. This check makes sure that ratios between peak row/column tones and their second harmonics are larger than a pre-defined threshold.

To provide reliable DTMF decoder operation, each character must be decoded twice, successively. If the previous character was decoded well, the summarized peak tone's level between two sequentially decoded characters must not differ substantially. This check also prevents false detection of short DTMF signals.

When all tone checks have been completed correctly, the peak tones are mapped in the corresponding character. If the previously detected character is same as the current character detected, the previous character is cleared, the decoder is switched in the pause-awaiting mode by clearing the *fDtmf\_Enable* flag, and valid character is returned. The pause-awaiting mode is required to prevent detecting one durable DTMF signal as several separate identical characters.

When any of checks fail, the previously detected character is cleared and a routine to calculate the input amplifier gain is called. If valid DTMF sequence detection is in progress, the gain adjusting system is blocked completely. Note that the gain is updated in the ADC ISR to synchronize the gain update with new measurement cycle and eliminate the unwanted correlator data fluctuations.

In the pause mode (*fDtmf\_Enable* is equal to 0) the strength of peak row/column tones is checked. The sum of squared row and column peak tones is used to detect the pause. If this sum is smaller than predefined threshold value then *fDtmf\_Enable* flag is set and decoder leaves pause mode.

#### **Automatic Gain-Control System**

The automatic gain control (AGC) subsystem is intended to expand the decoder input signal's dynamic range. The AGC adjusts the input PGA gain according to the estimated signal energy. Figure 6 illustrates the AGC operation:

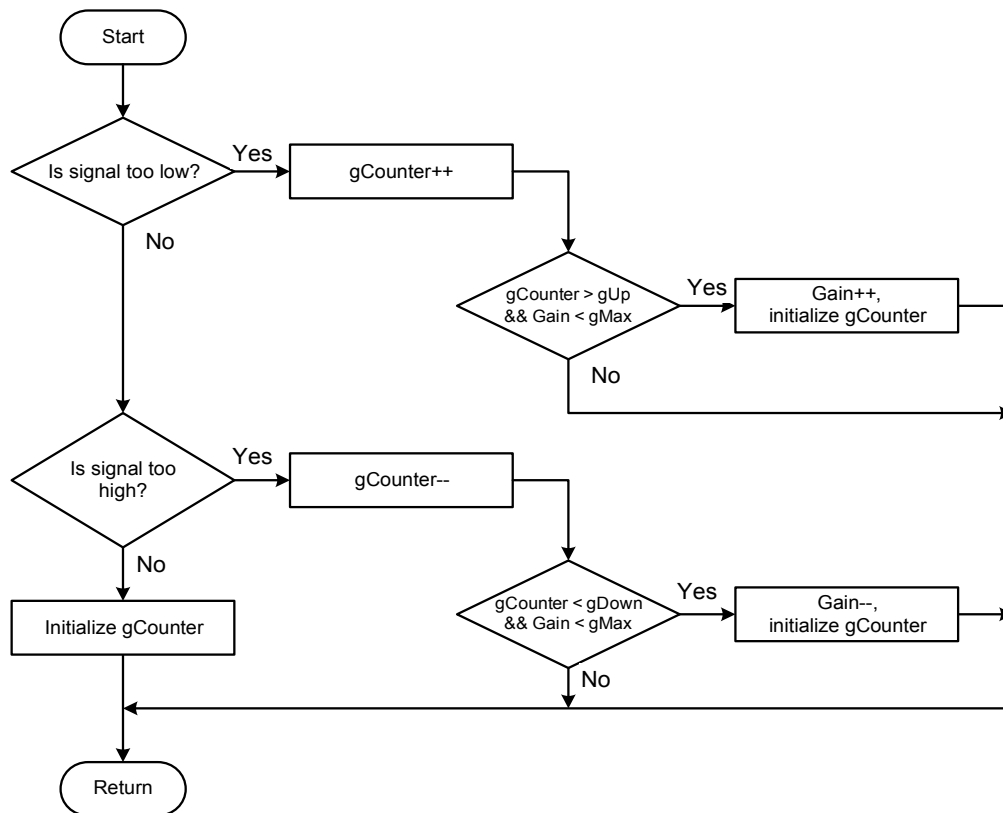


Figure 6. Automatic Gain-Control Algorithm

The AGC operates in the following way. When the signal energy is too low, the variable *gCounter* is incremented. If this variable reaches the upper predefined limit, the gain is increased and *gCounter* is reinitialized. If the incoming signal energy is too high, the *gCounter* is decremented. When this variable reaches the lower limit, the PGA gain is decreased. If the incoming signal energy falls in the allowed range, the gain counter variable *gCounter* is reinitialized to an immediate value. Upper and lower PGA gain level checks were implemented as well.

Note that the automatic gain subsystem has asymmetric transient characteristics; the gain rising time is much shorter than the gain falling time. This allows for quick recovery from over-range situations, but preserves the existing gain level during speech/signal pauses. These AGC transient characteristics were obtained by proper selection of gain counter *gCounter* upper, lower limits and intermediate initialization values. The AGC can be turned off at the time of compilation by setting *AGCUSE* condition variables in *dtmfcalc.asm* and *globdefs.h* to 0 if AGC is not used.

## Decoder Hardware

Figure 7. illustrates the internal PSoC connections and several external components that are needed to make the DTMF detector operational.

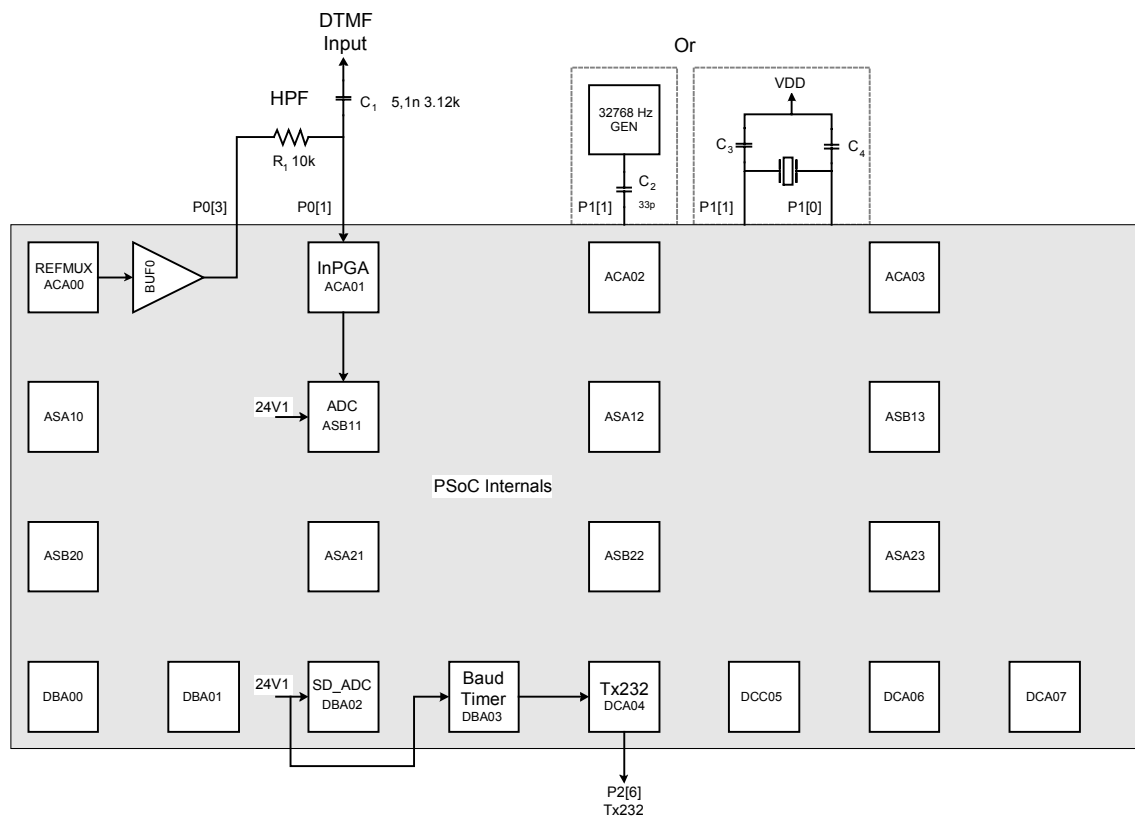


Figure 7. PSoC Internals

The high-pass filter  $R_1C_1$  cuts the low-frequency signals as the AC mains. The input programmable gain amplifier scales the incoming signal. The amplifier output is connected with input of the 8-bit sigma-delta ADC. The baud-rate timer forms the clock signal for serial transmitter, which can be connected via level shifter (MAX3221E for example) to a PC for analyzing the decoded characters. The default data-rate is 19200 bauds.

The crystal oscillator must clock the processor. The external oscillator (for example, DS32KHz from Maxim, SG-3030JF from Epson, etc.), connected with P1[1] port via small decoupling capacitor is recommended, although 32.768 kHz crystal, connected accordingly to [10] can be used as well.

If a watch crystal has been used, the high-frequency PSoC generator output must be analyzed to be sure the PLL is working properly and no visible jitter is present on generator signal when the serial transmitter is active. To do this, the baud-rate timer output can be passed to an external pin and observed via oscilloscope. The possible jitter can degrade the ADC signal-to-noise ratio and reduce the DTMF performance at whole. For debugging purposes, the external clock oscillator is strongly recommended. This is because the emulation board has dense traces and other digital signals can break/degrade the oscillation stability and the PLL. For end products the watch crystal is completely sufficient.

## Decoder Test Results

Table 2. illustrates test results for the proposed decoder.

The decoder was tested using generated waveforms from sound editor Cool Edit Pro 2.0; you can download the full-featured evaluation copy at [11]. Note that the text marked in *red italics* illustrates the properties that are outside of ITU specifications.

**Table 2. Decoder Test Results**

Test Item	Parameter	Comment
<b>Frequency Tests</b>		
Frequency	Allowed Deviation	The larger frequency tolerance for lower limit of 697 Hz, upper limit of 941 Hz, lower limit for 1209 Hz is caused by absence of proximity tones level checks for these frequencies. If you require a decoder strict in ITU-compliance, you can add two intermediate frequency bins. For example, take 660 Hz and 1075 Hz and add additional checks in the back-end logic routine.
697 Hz	<i>-5.5%</i> ... +3.1%	
770 Hz	-2.8%...+3.4%	
852 Hz	-3.4%...+3.5%	
941 Hz	<i>-3.7%</i> ... <i>+4.5%</i>	
1209 Hz	<i>-4.2%</i> ...+2.8%	
1336 Hz	-3.2%...+2.7%	
1447 Hz	-2.8%...+2.6%	
1633 Hz	-2.6%...+2.5%	
<b>Twist Tests</b>		
Normal Twist, Limit	9.1 Db	The tests were performed for digits 1, 5, 9, D.
Reverse Twist, Limit	4.7 Db	
<b>Time Tests</b>		
Minimum Tone Time	45 ms	The test was performed for digits 1, 2, 3, 4.
Min Pause Time	13 ms	
<b>Dynamic Range Tests</b>		
Dynamic Range (AGC Off)	<i>17 dB</i>	The dynamic range of 25 dB can be obtained by lowering DTMF signal-strength check threshold.
Dynamic Range (AGC On)	38 dB	The AGC must have time to stabilize, so if AGC was used, a continuous tone must precede the information tones.
<b>Talk-Off Resistance Tests</b>		
False Character Detection Rate	3-7 responses per hour when speech/music signal was present.	The talk-off test can be completed using the Bellcore reference tape.

## Possible Decoder Modifications

The proposed application provides the basic DTMF decoder, which can satisfy most designs. But many variations can be suggested. The decoder can be equipped with an anti-aliasing LPF for some applications. This LPF can be built using PSoC switched-capacitor filters or adjustable Sallen and Key low-pass filters, [AN2031](#).

If the user cannot dynamically allocate the sigma-delta ADC for detection, it can be done using PSoC dynamic re-configuration. A 6-bit SAR ADC or single-bit comparator can replace the ADC. The comparator-based DTMF detector has some performance issues, but is still suitable for many remote-control applications.

The correlation memory consumes  $4*N$  bytes of Flash memory. This is not suitable for some applications. In this case, the modified Goertzel algorithm can be implemented instead of a quadrature correlator and the second harmonics analysis can be omitted. Secondly, the user can adjust the sample length and sample frequency according to application demands. The command-line utility, *dtmf.exe* (included in the downloadable files) automatically generates the assembler include file *corrtables.inc* for given  $N$  and sample frequency.

The total time of execution for the ADC ISR during the correlation incoming signal is near 70 us with sample period 117 us and PSoC clock frequency 24 MHz. If the second harmonics analysis is not required, the ISR can be simplified by omitting the 16-quadrature sum calculation (saving 32 bytes of RAM) and the CPU clock can be decreased to 12 or 6 MHz. In this case, the sample rate can be reduced as well, and an anti-aliasing filter must be used to suppress possible speech/music signals.

If high accuracy in DTMF tone frequency analysis is required, the sample length can be increased by discarding the rule that each character must be detected twice consecutively.

Additional frequency bins with intermediate frequencies can be estimated and compared. To improve timing specifications and guarantee strong ITU compliance, the detector can be equipped with an incoming signal synchronizer to synchronize the correlation cycles with the DTMF start signal.

For debugging and testing purposes, a special debug mode is implemented when the detector sends, via serial port, the debug information concerning the incoming signal energy level, the correlator data, both fundamental DTMF frequencies and their second harmonics. You can use any terminal program to see this stream. The number at the far left is the scaled signal energy and the other sixteen numbers are correlator output data. The number at the far left corresponds to 697 Hz. The number at the far right corresponds to 3266 Hz. Figure 8. illustrates the terminal window during debug information transmission the when phone is dialing. Debug mode can be entered in the compile time by setting variable *DEBUG* to 1 in the *globdefs.h* file. In normal mode (*DEBUG* is set to 0), the decoder sends the properly decoded characters only, followed with new line symbols. Note that it is better to disconnect the ICE when the project is being debugged to decrease the possible influence of noise introduced by the ICE on analog part performance.

```

Mini Terminal
2620En 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
10En 1 1 1 0 0 0 0 0 0 1 0 3 0 0 1 1
43En 34 200 129 48 210 223 35 41 125 36 11 5 60 11 4 30
754En 8 10 4803 564 268 5181 67 79 4 111 20 17 6 20 15 85
3267En 68 102 5084 235 291 6981 32 84 6 204 397 59 6 54 17 77
2266En 183 180 82 10 56 71 111 46 125 161 41 15 40 19 23 32
1427En 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
275En 20 6 2 9 5 3 1 15 15 7 8 6 7 9 0 4
494En 6 99 2397 247 82 113 5629 110 44 103 54 24 61 37 32 34
3760En 20 80 3681 313 46 126 6089 49 2 107 12 15 18 55 22 10
2616En 203 42 2018 59 35 116 3139 39 179 44 171 9 55 38 64 6
2860En 2 1 1 1 2 1 1 1 2 1 5 1 1 2 2 1
75En 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
20En 76 52 111 2852 6194 61 24 54 182 21 163 23 144 27 2 2
3728En 0 43 203 4137 5956 83 55 70 6 12 133 6 129 10 4 11
3728En 168 165 493 4187 7514 133 29 43 233 11 182 38 274 19 2 19
4028En 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0
36En 1 1 1 0 1 1 1 1 1 1 4 1 1 1 1 1
59En 14 7 24 80 111 178 52 22 100 23 84 12 25 100 47 126
3546En 10 96 198 3541 307 6255 8 49 40 49 421 20 14 30 38 106
3546En 9 114 238 4486 75 6567 80 218 22 4 11 37 14 42 10 106
3905En 17 22 12 17 18 9 6 4 7 8 1 10 9 6 13 4
406En 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0
  
```

Figure 8. Terminal Window in Debug Mode

## References

1. ITU Blue Book, Recommendation Q.24, Multi-Frequency Push-Button Signal Reception, Geneva, 1989.
2. ITU Blue Book, Recommendation Q.23. Technical Features of Push-Button Telephone Sets, Geneva, 1989.
3. G. Arslan, B. Evans, F.A Sakarya and J. Pino, "Performance evaluation and real-time implementation of subspace, adaptive and DFT algorithms for multi-tone detection", in Proc. IEEE Int. Conf. Telecommunications, (Istanbul, Turkey), pp. 884-887, Apr. 1996. The paper can be downloaded in the electronic form at [http://ptolemy.eecs.berkeley.edu/papers/96/dtmf\\_ict/ict96.pdf](http://ptolemy.eecs.berkeley.edu/papers/96/dtmf_ict/ict96.pdf).
4. M. Felder, J. Mason, B. Evals, "Efficient dual-tone multi-frequency detection using the non-uniform discrete Fourier transform", IEEE Signal Processing Letters, vol. 5, pp.160-163, July 1998.
5. Kaiser Ulrich, Digital Processor Architecture with Reduced Instruction Set for Wave Digital Filters, Dissertation, Bochum, 1991.
6. Amey A. Deosthali, Shawn R. McCaslin, Brian L. Evans, A Low-Complexity ITU-Compliant Dual Tone Multiple Frequency Detector, IEEE Trans. on Signal Processing, vol. 48, no. 3, pp. 911-916, Mar. 2000, the article can be download in the electronic form at <http://www.ece.utexas.edu/~bevans/papers/2000/dtmf/dtmf.pdf>.
7. V. Oppenheim and R. W. Schaffer, Discrete-Time Signal Processing. Prentice-Hall, 1990.
8. J. Proakis, D.G. Manolakis, Digital Signal Processing Principles, Algorithms, and Applications. NJ, Prentice Halls, 1995.
9. Dave Van Ess, Signed Multi-Byte Multiplication, Application Note [AN2038](#), Cypress MicroSystems.
10. Jeff Dahlin, Using the PSoC Microcontroller External Crystal Oscillator, Application Note [AN2027](#), Cypress MicroSystems.
11. The sound editor Cool Edit Pro 2.0 can be downloaded freely at <http://www.syntrillium.com/download/>.

## About the Author

**Name:** Victor Kremin,

**Title:** Associate Professor

**Background:** Victor earned a radiophysics diploma in 1996 from Ivan Franko National Lviv University, PhD degree in Computer Aided Design Systems in 2000 and is presently working as an Associate Professor at National University "Lvivska Polytechnika" (Ukraine). His interests involve the full cycle of embedded systems design including various processors, operation systems and target applications. You may reach him at [vkremin@polynet.lviv.ua](mailto:vkremin@polynet.lviv.ua).

Cypress MicroSystems, Inc.  
2700 162<sup>nd</sup> St. SW, Building D  
Lynnwood, WA 98037  
Phone: 800.669.0557  
Fax: 425.787.4641

<http://www.cypressmicro.com/> / [http://www.cypress.com/aboutus/sales\\_locations.cfm](http://www.cypress.com/aboutus/sales_locations.cfm) / [support@cypressmicro.com](mailto:support@cypressmicro.com)

Copyright © 2003 Cypress MicroSystems, Inc. All rights reserved.

PSoC™ (Programmable System-on-Chip) and PSoC Designer are trademarks of Cypress MicroSystems, Inc. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

The information contained herein is subject to change without notice.