Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

THIS SPEC IS OBSOLETE

Spec No: 001-32906

Spec Title: ANALOG - LOW CPU CONSUMPTION DTMF DETECTOR - AN2247

Sunset Owner: Pushek Madaan (PMAD)

Replaced by: None

Analog - Low CPU Consumption DTMF Detector

AN2247

Author: Ruslan Bachinskyy
Associated Project: Yes
Associated Part Family: CY8C27xxx, CY8C29xxx
Software Version: PSoC® Designer™ 5.1
Associated Application Notes: AN2122, AN2038, AN2027

OBSOLETE

Application Note Abstract

AN2247 demonstrates the ability of the PSoC® device to create frequency filters. This device was selected to create an efficient dual-tone multi-frequency (DTMF) detector with minimal CPU consumption.

Introduction

DTMF signaling is widely used in analog telephone dialing, data entry, voice mail systems, and remote control of various consumer electronics such as answering machines, home automation devices and bank information services.

A DTMF signal is the sum of two sinusoidal waveforms with predefined frequencies, which were selected according to the ITU Recommendations, Q23 and Q24. (See References) A user can interact with the DTMF dialer using a keypad with 16 characters. Figure 1 illustrates keypad layout and corresponding signal frequencies.

Conventional phones typically contain the keys '0' to '9', '*' and "#'. Keys 'A' to 'D' can be found in industrial controllers and are mainly used for remote control. There are eight different frequencies, grouped into rows and columns. Each character is associated with a pair of frequencies from row and column groups. The DTMF decoder must determine which frequencies are present in the incoming signal and correctly decode them according to the corresponding character. Table 1 details key decoder ITU specifications. The full DTMF signaling specifications can be found in Reference 1 at the end of this Application Note.

There are various approaches to build DTMF detectors. The typical detector consists of a front-end signal processing system and a back-end logic system to discern the presence or absence of a DTMF character.

The most popular algorithms for front-end signal processing are based on Discrete Fourier Transform (DFT) modifications. (See References) To reduce the number of calculations, the coefficients are calculated by using a modified Goertzel algorithm, which requires only *N+2* multiplication and *2N+2* additions to process data for every *N* sample. (See References).

The proposed DTMF detector is designed to maximize the PSoC analog blocks' potential, which results in low CPU usage and reduction in cost. In the device, two 4-pole band pass filters are used to separate signals into low and high frequency components. High and low frequency parts are then processed independently. To calculate the frequency for each part, a signal period duration detection scheme is used.

Figure 1. Touch Tone Phone Keypad

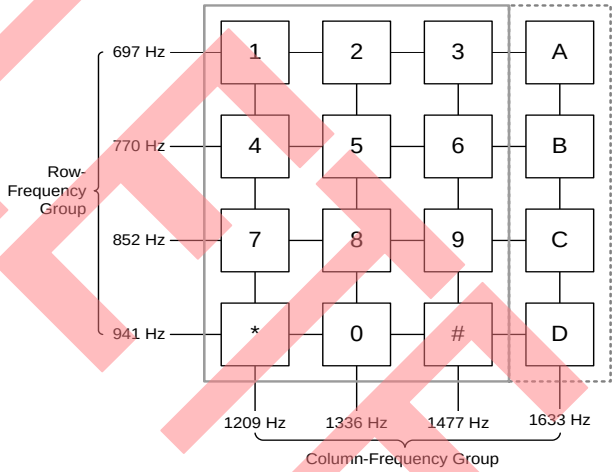
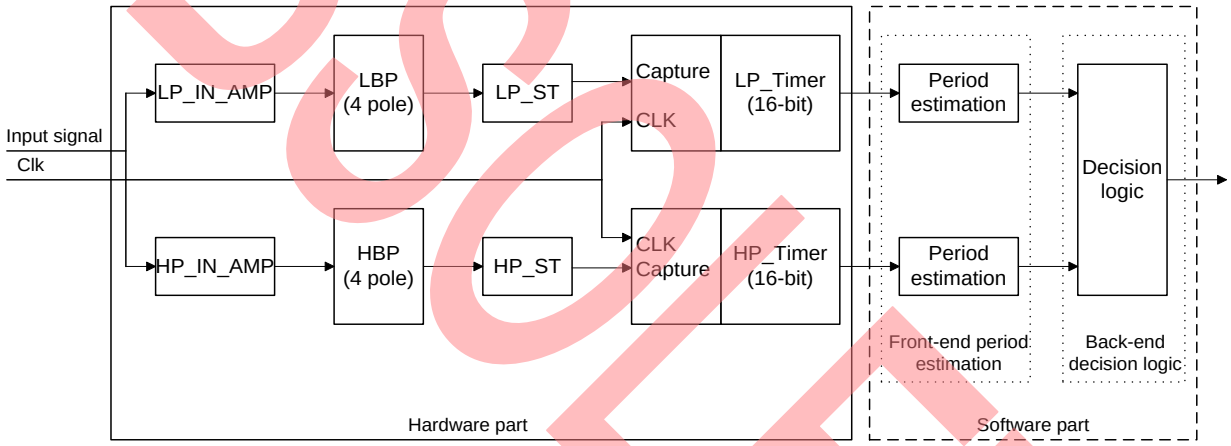


Table 1. DTMF Decoder ITU Specifications

Characteristic	Parameter
Frequency Tolerance	Frequency tolerance ±1.5 % is allowed for valid DTMF tone. Tones with offset ±3.5% must be rejected.
Signal Duration	A signal with duration of 40 ms must be considered valid. Tones with duration 23 ms or less must be rejected.
Signal Interruption	A valid DTMF signal interrupted for 10 ms or less should not be detected as two distinct tones.
Signal Pause	A valid DTMF signal separated by 40 ms pause or more must be detected as two distinct characters.
Signal-to-Noise Ratio (SNR)	The DTMF detector must correctly process signal with SNR at 15 dB.
Tones Twist	The detector must correctly decode DTMF codes when row frequency signal is 8 dB larger than column frequency signal. The detector must correctly operate when column frequency signal is 4 dB larger than row signal.
Talk-Off	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.

Figure 2. DTMF Detector - One Frequency Channel Functional Diagram



LP_IN_AMP/HP_IN_AMP – input amplifiers for row (low)/column (high) frequencies;
LPB/HPB – 4-pole band pass filters for passing row/column frequencies only;
LP_ST/HP_ST – Schmidt triggers of row/column frequencies group;
LP_Timer/HP_Timer – 16-bit timers for row/column frequencies:
CLK – timer clock input;
capture – timer capture input;

DTMF Detector Functioning

The function of the DTMF detector is based on filtering the input signal into row and column groups, calculating the sine wave period and independently comparing them with predefined values. First, two 4-pole band pass filters split the DTMF signal into two components – low frequency (697 Hz – 941 Hz) and high frequency (1209 Hz – 1633 Hz). Each filter output is routed to a Schmidt trigger input for converting a sine wave into a square wave. Refer to Figure 3 on page 3. This square signal is routed to the “capture” input of 16-bit timer through a comparator bus. When the signal is present, the detector’s two input timers (low and high) start. When the sine wave begins to increase, it is converted into a square wave front and this front initiates the capture mechanism of a digital timer block.

The captured value is stored in a circular RAM buffer for further processing. To minimize measurement errors, the analyzing procedure begins once eleven values are captured and stored in the RAM buffer. Then the frequency estimate is made for ten periods of the sine wave. The DTMF detector functional diagram is depicted in Figure 2.

The frequency channel consists of:

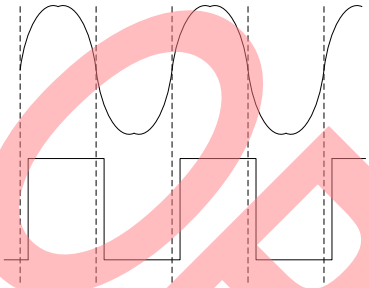
- Input amplifier (IN_AMP) used for amplification and routing the input signal to the band pass filters.
- Band pass filters tuned to pass low/high bands of DTMF frequencies. They only pass low/high DTMF frequency bands and block other frequencies.

OBSOLETE

- Schmidt trigger, which converts sine wave and increases and decreases into square wave rising and falling edges, respectively.
- 16-bit timer.

The sine wave conversion process is shown in Figure 3.

Figure 3. Sine Wave Conversion into Square Wave



Sample counting during sine wave period is used to estimate frequency. Two 16-bit timers are used (one for the low frequency component and the other for the high frequency component). When the signal starts to rise, it is converted into a rising edge of a square wave, which initiates the capture mechanism of the timer block. The capture mechanism writes the current timer value (number of clocks between current rising edge and previous rising edge) into an internal digital block register and continues counting. The timer's interrupt service routine, called by the timer capture, reads this value from the register and stores it in a RAM buffer for further processing. Each falling edge of the sine wave is transformed into the falling edge of the square wave, which causes the Schmidt trigger output value to reset to zero state. This process is shown Figure 3.

After eleven captured timer values are collected, the first value is subtracted from the last value and the number of samples between ten sine periods is obtained. When the first subtraction result is ready, the sine frequency estimation starts. Frequency estimation is made by comparing the number of samples between ten signal periods with random predefined values for each DTMF frequency. If the subtraction result is in a predefined range for one of the DTMF frequencies, the given DTMF frequency is detected. After eleven timer values are collected, the next timer value is stored in the buffer at the location in which the oldest timer value resides. The latest timer value is subtracted from the next oldest timer value. This process is repeated while the DTMF signal is present. When the DTMF signal pauses during a defined time interval, the analyzing procedure checks to see if the DTMF symbol corresponds to the valid DTMF symbol demands and determines if the symbol is valid. If the DTMF symbol is valid, it is transmitted to a host PC via a COM port.

The timer interrupt service routine (ISR) and analyzing procedure are discussed below.

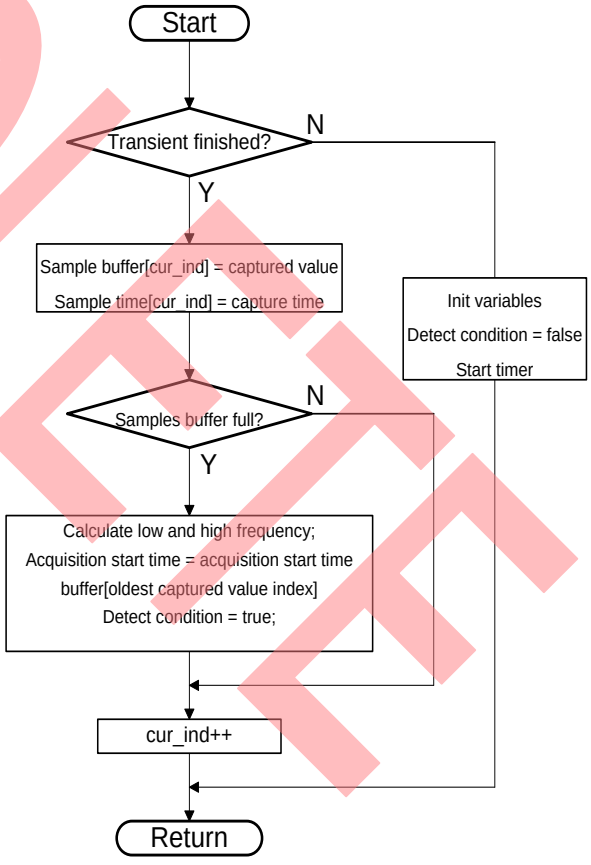
Detector Implementation

The DTMF detector consists of front-end period estimation and back-end decision logic. Period estimation consists of two interrupt service routines (one for the low frequency part and the other for the high frequency part), which provide data, generated and processed by the capture mechanism. The back-end decision logic checks to see if the DTMF symbol corresponds to the valid DTMF symbol conditions.

Front-End Period Estimation

The front-end processing consists of two identical timer capture ISRs. The flow of the timer capture ISR is shown in Figure 4. As can be seen from the figure, the period information starts collecting after the transient finishes. It is designed to collect only stable information. The transient is finished after two signal periods. Once it is finished, the timer values are written to a RAM buffer. After eleven signal periods are collected, the number of samples between ten signal periods is calculated by subtracting the oldest timer value from the latest timer value. The detect flag, for the analyzing procedure, is also set.

Figure 4. Timer Interrupt Service Routine Flow Diagram



OBSOLETE

Back-End Decision Logic

The back-end decision logic is very simple and consists of two functions. The first function checks to see if the detected symbol corresponds to the preset time conditions. The second function converts frequency information obtained from the timer interrupt service routine into DTMF symbols. The first function is shown in Figure 8 on page 6. If the DTMF symbol detected is valid, the DTMF analyzing procedure writes this symbol into a global variable sets the valid DTMF symbol and detects the condition for main function.

The main function checks if a valid DTMF symbol condition is set. If a valid condition is set, the function transmits the detected symbol to the host PC and resets the DTMF symbol condition.

DTMF Detector Internals

The proposed DTMF detector functions with low CPU consumption.

The DTMF detector uses the following digital blocks:

- DBB00, DBB01 (LP_Timer) 16-bit timer, calculates number of samples between two sine periods for the low frequency part of the DTMF symbol.
- DBB10, DBB11 (HP_Timer) 16-bit timer calculates the number of samples between two sine periods for the high frequency part of the DTMF symbol.
- DBB02 (LP_BPF_CLK) and DBB12 (HP_BPF_CLK) generates clock signal for the band pass filter on each low and high frequency part of the DTMF signal.
- DCB03 (Timer) measures DTMF signal and pausing times.
- DCB13 (Transmitter) is used to transmit detected DTMF symbols to the host PC.

The DTMF detector uses the following analog blocks:

- ACB00 (LP_IN_AMP) and ACB02 (HP_IN_AMP) amplify and route input signal to the BPF's inputs.
- ASC10, ASD11 (LPB_1) first pole of band pass filter for the low frequency part of DTMF signal transmission.
- ASD11, ASC21 (LPB_2) second pole of band pass filter for the low frequency part of DTMF signal transmission.
- ASC12, ASD22 (HPB_1) first pole of band pass filter for the high frequency part of DTMF signal transmission.
- ASD13, ASC23 (HPB_2) second pole of band pass filter for the high frequency part of DTMF signal transmission.
- ACB01 (LP_SCH_TRIG), ACB03 (HP_SCH_TRIG) respective Schmidt trigger for low and high frequency parts to convert sine waves into square waves and route to timer capture inputs via comparator buses.

DTMF Detector Scheme

The DTMF detector schematic is shown in Figure 10 on page 9. The DTMF detector consists of:

- PSoC device (U1) analyzes input signals and converts them into DTMF symbols.
- MAX232 voltage level shifter (U2) converts TTL voltage levels into EIA-232 (RS-232) voltage levels.
- DB9 connector connects the DTMF detector via P1 to the PC.

The PC is used to show the detected DTMF symbols.

Table 2 on page 5 illustrates the test results of the proposed decoder. The decoder was tested using waveforms generated with the sound editor Cool Edit Pro 2.0. You can download the full-featured evaluation copy shown in Reference 11 at the end of this Application Note. Note that the text marked in red in Table 2 illustrates the properties that are outside of ITU specifications.

Program Usage Estimates

The program consists of the following procedures:

- Main function
- DTMF analyze procedure, which is called from the main function
- Two, timer capture interrupt service routines (one for row frequency and other for column frequency of DTMF signal).

When a signal is not present on the detector input, the interrupt service routines are not triggered. The main function and DTMF analyze procedure are executed. In this case, CPU usage is near 1%.

When a signal is present on the detector input, the interrupt service routines are called. This increases the duration at which the main function executes and, as a result, CPU usage. In this case, CPU usage is near 7%.

The estimate of CPU usage is made for CPU core frequency, 24 MHz.

OBSOLETE

Table 2. The Decoder Test Results

Test Item	Parameter	Test Result
DTMF Symbols Tests		
Symbol	**Frequency Deviation**	**Result**
0-9,A-D,*,#	Row freq. - 1.5% ... Column freq.-1.5%	PASSED (All DTMF Symbols were Detected)
0-9,A-D,*,#	Row freq. - 1.5% ... Column freq.+1.5%	PASSED (All DTMF Symbols were Detected)
0-9,A-D,*,#	Row freq. + 1.5% ... Column freq.-1.5%	PASSED (All DTMF Symbols were Detected)
0-9,A-D,*,#	Row freq. + 1.5% ... Column freq.+1.5%	PASSED (All DTMF Symbols were Detected)
0-9,A-D,*,#	Row freq. - 3.5% ... Column freq.-3.5%	PASSED (All DTMF Symbols were Rejected)
0-9,A-D,*,#	Row freq. - 3.5% ... Column freq.+3.5%	PASSED (All DTMF Symbols were Rejected)
0-9,A-D,*,#	Row freq. + 3.5% ... Column freq.-3.5%	PASSED (All DTMF Symbols were Rejected)
1-9,A-C	Row freq. + 3.5% ... Column freq.+3.5%	PASSED (All DTMF Symbols were Rejected)
0,D,*#	Row freq. + 3.5% ... Column freq.+3.5%	NOT PASSED (DTMF Symbols were not Rejected) when frequency deviation is more than 3.5%
0-9,A-D,*,#	Row freq. + 3.6% ... Column freq.+3.6%	PASSED (All DTMF Symbols were Rejected)
Twist Tests		
Normal Twist, Limit	8,5 Db	Tests were performed for all digits
Reverse Twist, Limit	5,5 Db	
Time Tests		
Minimum Tone Time	40 ms	Test was performed for all digits
Minimum Pause Time	16 ms	
Dynamic Range Tests		
Dynamic Range	9 dB	Dynamic Range can be extended by using the automatic gain control loop
Talk-Off Resistance Tests		
Detection False Characters Rate	180-200 responses per hour when speech signal present	Talk-off test can be completed using the Bellcore reference tape
Break Resistance		
Breaks	DTMF signal is considered as valid DTMF symbol if break up to 10 ms is present	PASSED

OBSOLETE

Figure 5. High Frequency Amplitude 1.5 V

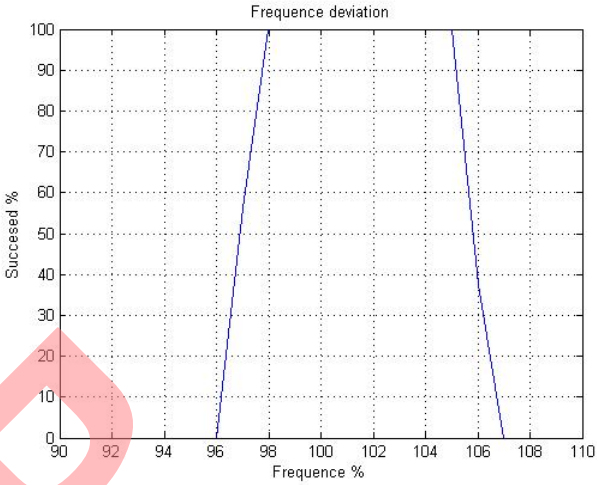


Figure 6. Frequency Amplitude 1.5 V

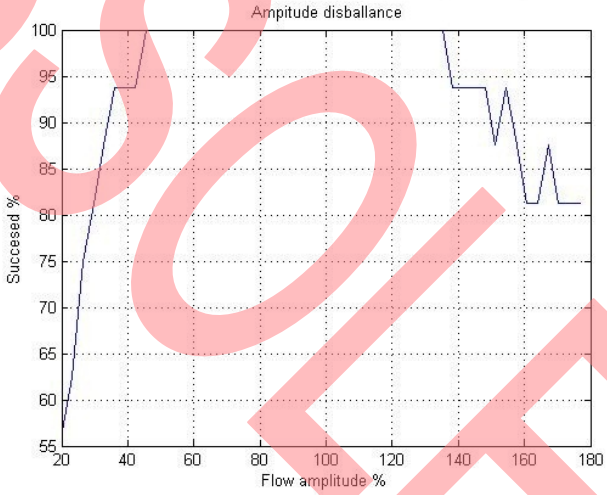
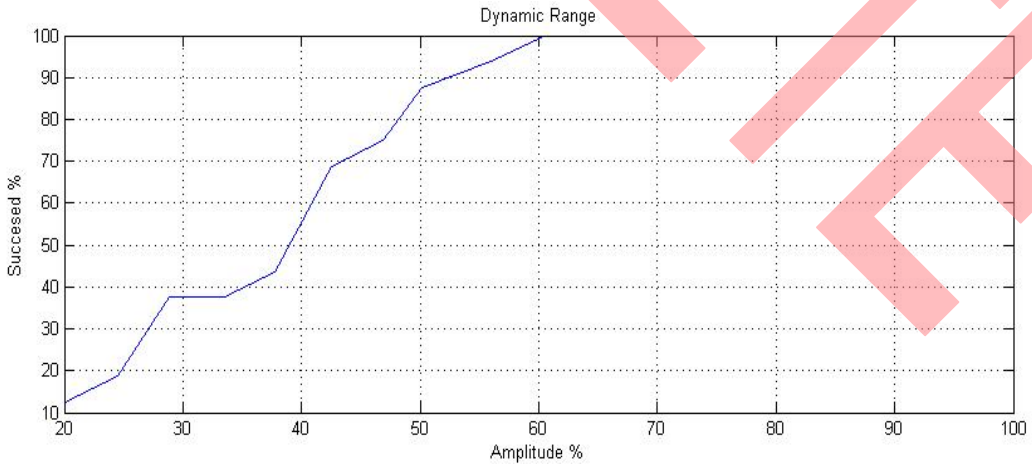
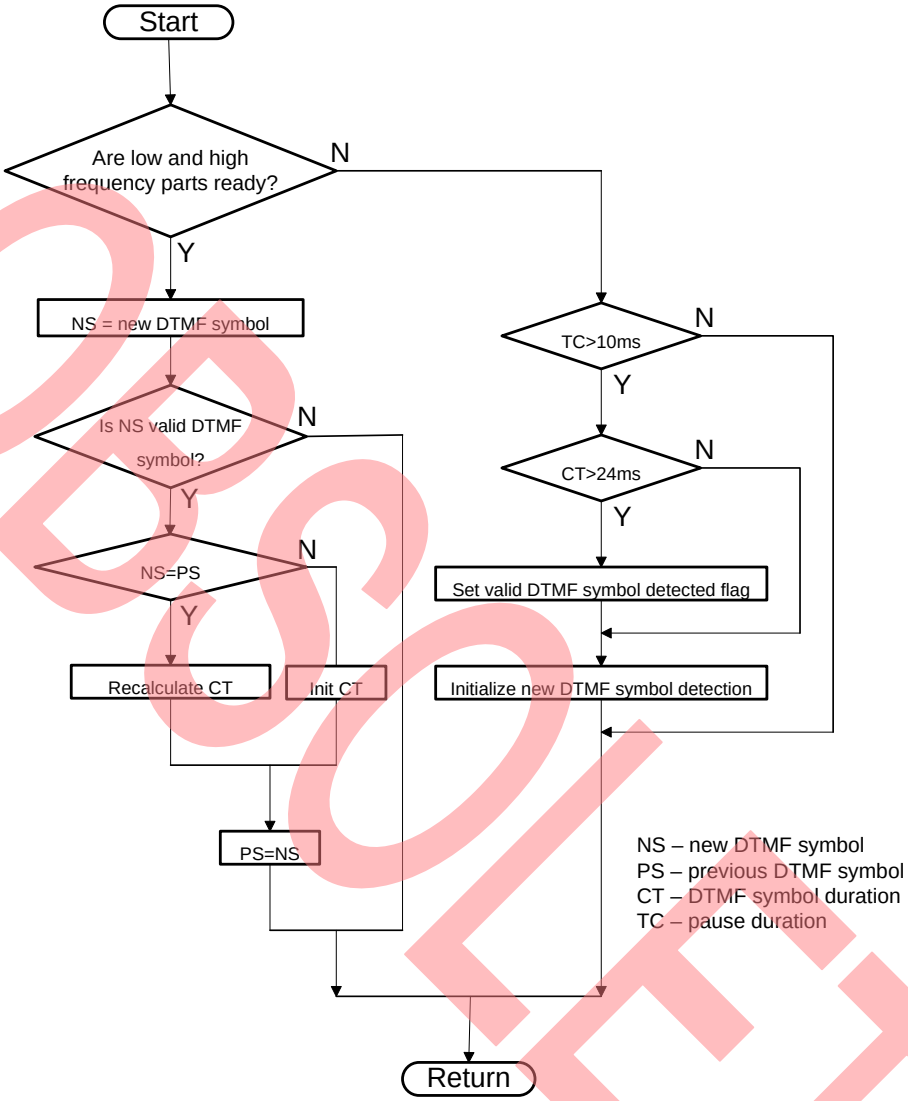


Figure 7. Maximum Amplitude 2.0 V



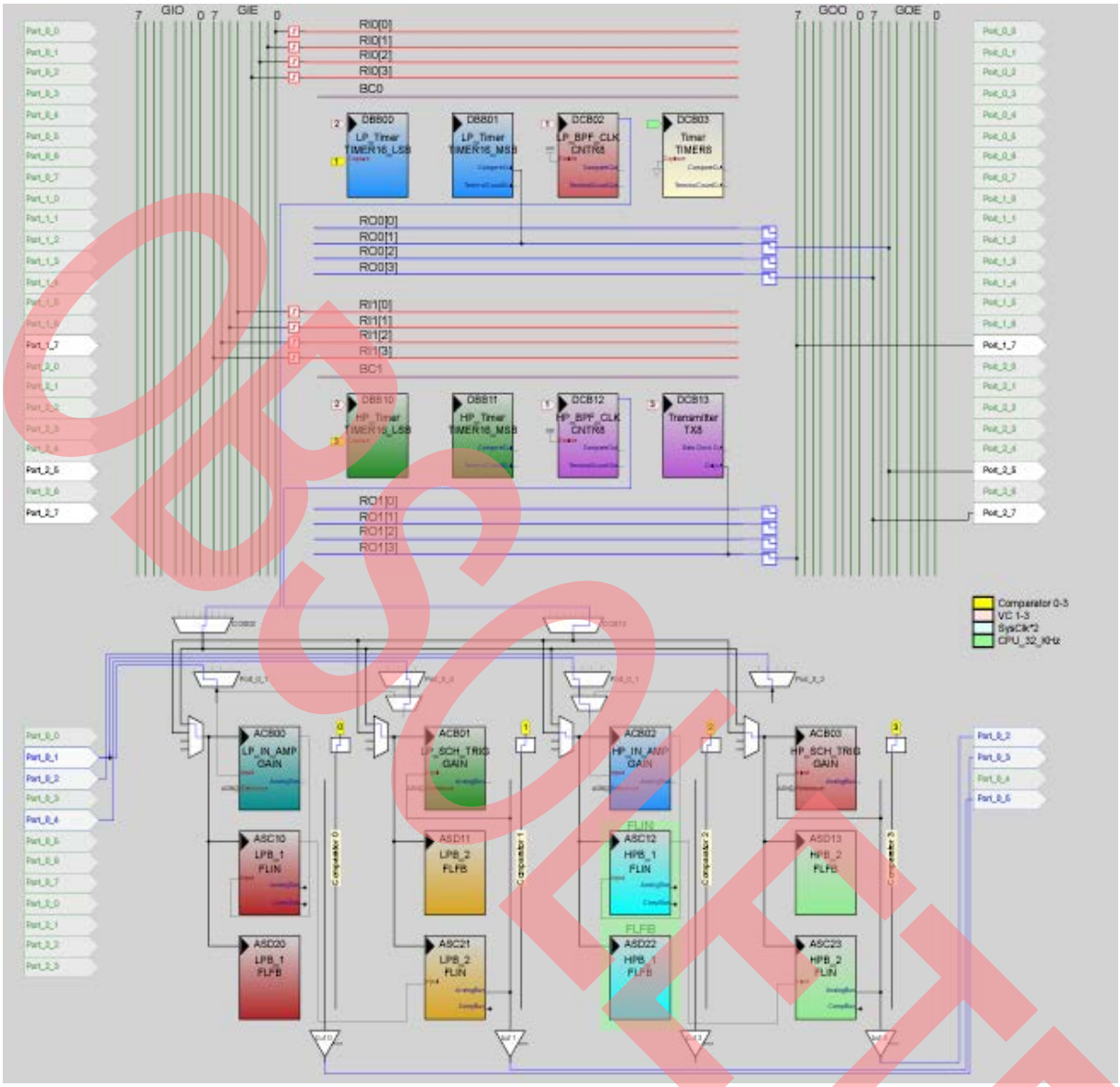
OBSOLETE

Figure 8. Analyzing Procedure Flow Diagram



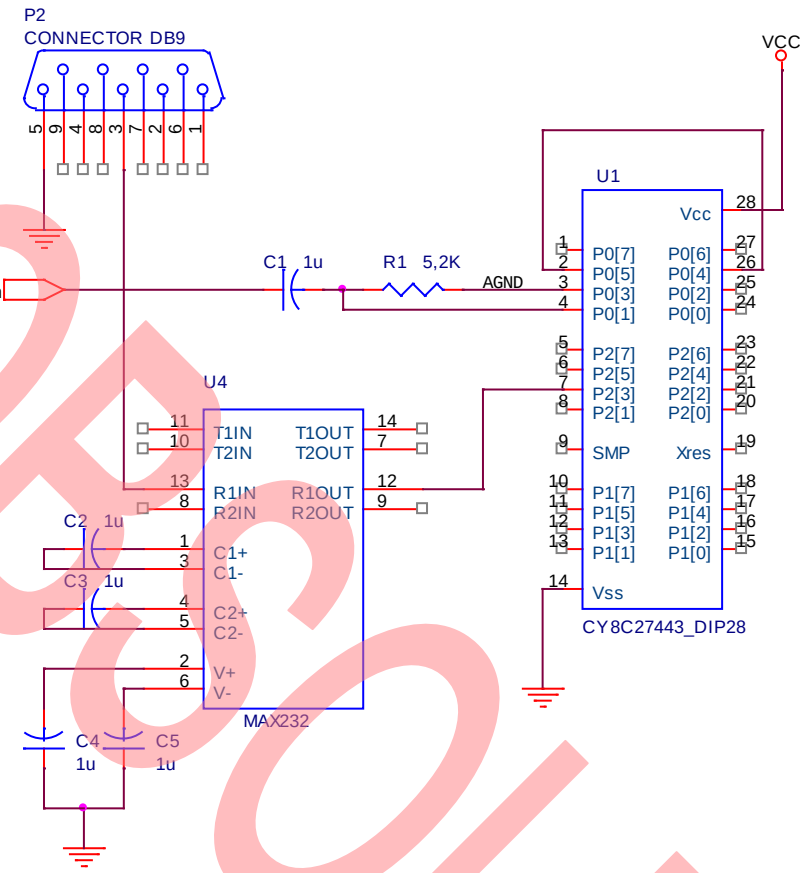
OBSOLETE

Figure 9. PSoC Internals for DTMF Detector



OBSOLETE

Figure 10. DTMF Detector Schematic



OBSOLETE

Summary

This Application Note demonstrates how PSoC hardware resources can be used to detect DTMF signals. Compared with software-based DTMF detectors, this application is characterized by low CPU consumption.

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, (Istambul, Turkey), pp. 884-887, Apr. 1996. The paper can be downloaded in electronic form at 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. 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 downloaded in electronic form at http://www.ece.utexas.edu/~bevans/papers/2000/dtmf/dtmf.pdf
6. Kaiser Ulrich, Digital Processor Architecture with Reduced Instruction Set for Wave Digital Filters, Dissertation, Bochum, 1991.
7. V. Oppenheim and R. W. Schafer, 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 for free at www.syntrillium.com/download/
12. Various DTMF detections; Term project report for ME780 Digital signal analysis for mechanical systems, Ta-young Gabriel Choi.

About the Author

Name: Ruslan Bachinskyy
Title: Postgraduate Student
Background: Ruslan earned his Master’s diploma in “Specialized Computer Systems” in 1999 from National University "Lvivska Polytechnika," and is presently working as an Application Engineer consultant for Cypress Semiconductor. His interests include the full cycle of embedded systems design together with various processors, operation systems and target applications.
Contact: bacr_ukr@cypress.com

OBSOLETE

Document History

Document Title: Analog - Low CPU Consumption DTMF Detector – AN2247

Document Number: 001-32906

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	1494923	YARD_UKR	09/21/2007	New Spec
*A	3255506	ANBI_UKR	06/09/2011	Updated document as per template. Software Version Updated.
*B	3315215	YARD_UKR	07/15/2011	Updated associated project.
*C	4422196	PMAD	06/27/2014	Obsolete document. Completing Sunset Review.

OBSOLETE

In March of 2007, Cypress recataloged all of its Application Notes using a new documentation number and revision code. This new documentation number and revision code (001-xxxxx, beginning with rev. **), located in the footer of the document, will be used in all subsequent revisions.

PSoC is a registered trademark of Cypress Semiconductor Corp. "Programmable System-on-Chip," and PSoC Designer, are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com/

© Cypress Semiconductor Corporation, 2007-2014. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.