

Add DTMF Generation and Decoding to DSP-mP Designs

APPLICATION REPORT: SPRA168

*Author: Pat Mock
Semiconductor Systems Engineering*

*Digital Signal Processing Solutions
1989*



IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Add DTMF Generation and Decoding to DSP- μ P Designs

Abstract

Because of the programmability of the digital signal processor, the TMS32010 can also be programmed to handle Dual-Tone MultiFrequency (DTMF) encoding and decoding over telephone lines. For a system already performing digital signal processing functions using the TMS320, this DTMF capability may be obtained at no additional hardware cost. This report is a reprinted article from Electronic Design News. The article details the DTMF implementation algorithm and provides TMS32010 program description and code.



Product Support on the World Wide Web

Our World Wide Web site at www.ti.com contains the most up to date product information, revisions, and additions. Users registering with TI&ME can build custom information pages and receive new product updates automatically via email.



Add DTMF generation and decoding to DSP- μ P designs

In a computer system that employs a digital-signal-processing μ P and that's equipped for phone-line communications, the DSP μ P can generate and decode DTMF dialing signals as well as handle typical DSP functions. Therefore, the system can both dial out to establish communications links and accept Touchtone inputs for remote control of its functions.

Patrick Mock, Texas Instruments

A digital-signal-processing (DSP) μ P can handle Touchtone (DTMF) dialing and decoding over telephone lines in addition to its customary signal-processing chores. As a consequence, if a computer system already has a DSP μ P and A/D and D/A converters in place, then the system can decode DTMF signals, and any Touchtone telephone can serve as a data-entry terminal or a remote-control console. The only cost for these DTMF enhancements is additional program space in the μ P's ROM.

This article outlines a DTMF generating scheme and describes in detail the implementation of DTMF decoding

in a specific DSP μ P, the TMS32010. Although the DTMF decoder functions as intended, it fails to meet AT&T specs exactly because it's designed to detect DTMF tones in the presence of speech and because it suits computer applications like voice-mail and electronic-mail systems, which are not pure telephone applications. DTMF tone decoders that *do* meet AT&T specs usually stop decoding tones if they detect speech. With a more exacting program, the TMS32010 could meet AT&T specs to the letter. One of the goals of this project, however, was to make the DTMF code as compact as possible to allow the DSP μ P to do other jobs. Some performance was sacrificed as a consequence.

Tone generation is easy

A DTMF tone generator (Ref 1) can consist of a pair of programmable, second-order harmonic oscillators (Fig 1). The sample-generation rate of the oscillators determines the total harmonic distortion of the output. The higher the sampling rate, the more nearly exact the signal will be. In all cases, you must choose a sample-generation rate greater than approximately 7k samples/sec to achieve an acceptable signal. (Fig 2 explains the DTMF tone-coding scheme.)

Because the telephone company's official digitizing rate is 8k samples/sec, most generating circuits run at this rate. According to the Nyquist criterion, which specifies that the sampling rate must be at least twice

If a computer can decode DTMF signals, then any Touchtone telephone can serve as a data-entry terminal.

the frequency of the highest-frequency signal being sampled, 8k samples/sec is more than adequate for generating any valid pair of tones using the TMS32010; the highest frequency involved is 1633 Hz. Because of a limitation in the system used to develop the chip's tone-generating and -decoding programs, the decoding-program version listed in Fig 3 runs at 9766 samples/sec, and all testing was done using this version. However, Table 1 presents coefficients for running at 8k samples/sec; Fig 4's listing shows the portion of the code that must be amended for 8k-sample/sec operation.

Fig 5 shows the flow chart for the DTMF tone-generating algorithm. (The DTMF tone-generating routine described in Ref 1 takes up 160 words in the program ROM.) The first step of the algorithm initializes the processor and the interfaces and performs all other required initialization. The next step retrieves the digit that's to be dialed (0 through 9, A through D, or "*" or "#") from a specified location in memory. The digit serves as a pointer within a table that contains the values required to initialize the resonators.

Because this design uses two oscillators for eight possible frequencies rather than eight oscillators, to provide the correct frequencies you must load the

Text continues on pg 212

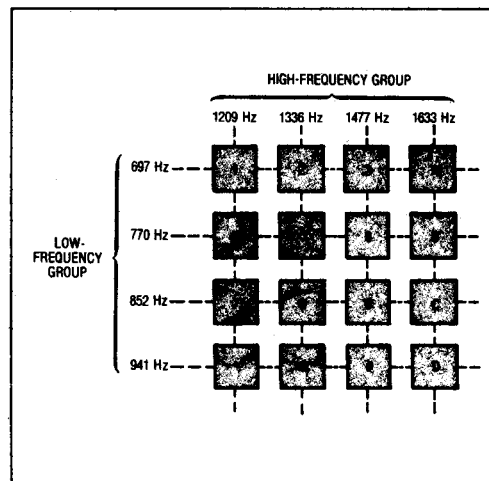


Fig 2—Pressing a button on a Touchtone telephone's 4×4 keypad generates DTMF signaling tones in pairs. For example, pressing "6" generates a 770-Hz tone from the low-frequency group and a 1477-Hz tone from the high-frequency group. Note that the keypad has four keys (A through D) that are not normally seen on most phones. They're available with some special instruments.

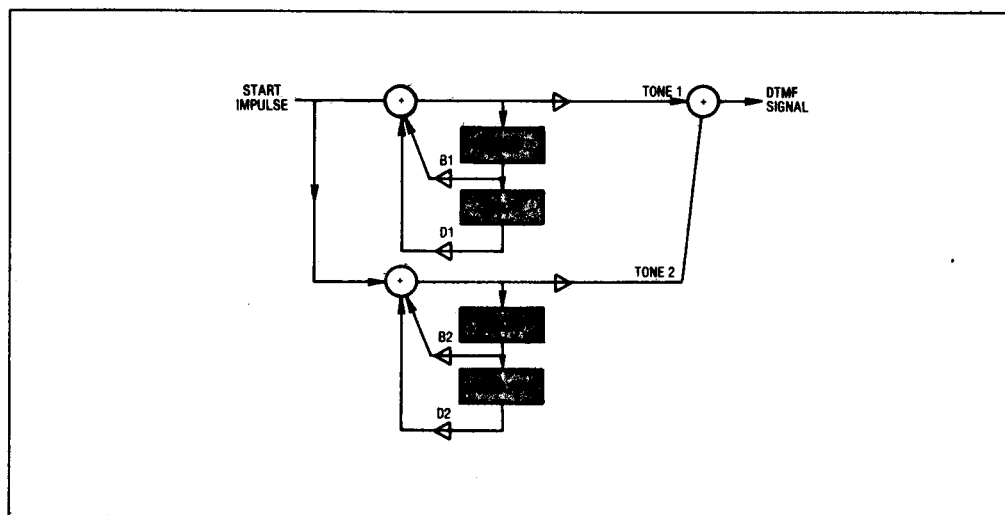


Fig 1—A pair of programmable, second-order harmonic oscillators make up the DTMF tone generator represented by this directed graph. The delay boxes temporarily hold samples for one iteration. The delayed samples are multiplied by coefficients B and D and summed to generate a tone sample. The tones, in turn, are summed and sent to a D/A converter.


```

v: 1.00      DTMF TONE DECODER                      TMS320 Assembler vers 1.3

0          *
1          *
2          *      DTMF TONE DECODER
3          *      c Copyright Texas Instruments, 1984
4          *      by Patrick C. Mock
5          *      Northeast Systems Engineering
6          *
7          *
8          *      TITL  / DTMF TONE DECODER /
9          *      IDT   /v: 1.00/
10         0000 f900      B      START      Go To The Beginning
11         0001 0017
12         *
13         *
14         0007 CS8      EQU 7      Define Variables
15         0008 CS9      EQU 8
16         000c CS13     EQU 12
17         000f CS16     EQU 15
18         0010 CLCK     EQU 16
19         0011 MODE     EQU 17
20         0010 ROWMX    EQU 16      10 Contains decoded row
21         0011 COLMX    EQU 17      11 " " column
22         0012 POSMAX   EQU 18
23         0013 NEGMAX   EQU 19
24         0014 ONE      EQU 20
25         0015 LAST     EQU 21      16 Contains last decode
26         0016 LAST2    EQU 22
27         0017 COUNT    EQU 23
28         0018 RC       EQU 24
29         0019 CC       EQU 25
30         001a ROWMAX   EQU 26
31         001b COLMAX   EQU 27
32         001c DAT11    EQU 28
33         0021 DAT23    EQU 33
34         0022 DAT14    EQU 34
35         0024 DAT15    EQU 36
36         0028 DAT17    EQU 40
37         0029 DAT27    EQU 41
38         002a DAT18    EQU 42
39         002b DAT28    EQU 43
40         002d DAT29    EQU 45
41         0035 DAT213   EQU 53
42         003b DAT216   EQU 59
43         003c DATIN    EQU 60
44         *
45         * BEGIN DATA TABLES
46         0002 738b      *      DATA 29579      Real Coeff N=226
47         0003 704e      DATA 28750
48         0004 6cb8      DATA 27832
49         0005 68cb      DATA 26827
50         0006 5b23      DATA 23331
51         0007 5355      DATA 21333
52         0008 4af3      DATA 19187
53         0009 3ef8      DATA 16120
54         *
55         000a 4eff      *      DATA 20223      Real Coeff N=222
56         000b 462b      DATA 17963      2nd harmonic
57         000c 39a0      DATA 14752
58         000d 2c58      DATA 11352
59         000e 01d0      DATA 464
60         000f ec28      DATA -5080
61         0010 d712      DATA -10478
62         0011 c000      DATA -16384
63         *
64         0012 0200      *      DATA 512      CLCK = Sample Frequency
65         0013 000a      DATA >000A      MODE
66         *
67         0014 7fff      *      DATA >7FFF      POSMAX = Mask for data in
68         0015 8000      DATA >8000      NEGMAX = Mask for data out
69         *

```

Program continues on pg 208

Fig 3—This tone-decoding program for the TMS32010 runs at 9766 samples/sec. However, the official digitizing rate specified by the phone company is 8k samples/sec; Fig 4 shows the section of code that adapts this program to 8k-sample/sec operation.

```

70      *
71      0016 0001  TABLE DATA 1      ONE
72      *      Start of Program      *****
73      *
74      0017 7f8b  START      SOVM
75      0018 6e00          LDPK 0
76      0019 6880          LARP 0      <Break>
77      001a 7014          LARK 0,ONE
78      001b 7e16          LACK TABLE
79      001c 6788  NEXT      TBLR *      Initialize Coefficients
80      001d 1014          SUB  ONE
81      001e f400          BANZ NEXT
82      0020 4811          OUT  MODE,0      Set AIB Mode
83      0021 4910          OUT  CLK,1      Set AIB Clock
84      *
85      *
86      * Load not recognized symbol
87      *
88      0022 7eff  NOT      LACK >FF
89      0023 6915          DMOV LAST
90      0024 5015          SACL LAST
91      *
92      0025 7f89  AGAIN     ZAC      Zero DFT Loop Variables
93      0026 701f          LARK 0,31
94      0027 711c          LARK 1,DAT11
95      0028 f500          BV      ZERO
96      0029 002a          *
97      002a 6881  ZERO     LARP 1
98      002b 50a0          SACL *+,0,0
99      002c f400          BANZ ZERO
100     002d 002a          *
101     *      Take data and calculate DFT loop
102     *
103     002e 7ee2          LACK 226      SET DFT LOOP VARIABLE
104     *
105     002f 5017  LOOP     SACL COUNT
106     0030 700f          LARK 0,CS16
107     0031 713b          LARK 1,DAT216
108     0032 1214          SUB  ONE,2
109     0033 fc00          BGZ  WAIT
110     0034 0037          *
111     0035 7007          LARK 0,CS8
112     0036 712b          LARK 1,DAT28
113     *
114     0037 f600  WAIT     BIOZ CALC      Wait for A/D
115     0038 003b          *
116     0039 f900          B      WAIT
117     003a 0037          *
118     *
119     003b 423c  CALC     IN      DATIN,2
120     003c 2012          LAC      POSMAX
121     003d 783c          XOR      DATIN
122     003e 503c          SACL      DATIN      Convert data to 320 format
123     *
124     * BEGIN DFT LOOPS
125     *
126     003f 6a81  FRPT     LT      *,1
127     0040 2c3c          LAC      DATIN,12      X(n)
128     0041 6298          SUBH     *-      X(n)-Y(n-2)
129     0042 6d88          MPY      *      cos(B*C)*Y(n-1)
130     0043 6b88          LTD      *      Y(n-1)->Y(n-2) and
131     0044 7f8f          APAC
132     0045 7f8f          APAC
133     0046 7f8f          APAC      X(n)+2cos(B*C)*Y(n-1)-Y(n-2)
134     0047 5890          SACH     *-,0,0      --> Y(n-1)
135     0048 f500          BV      CHECK
136     0049 0050          *
137     004a f400          BANZ FRPT
138     004b 003f          *
139     *
140     004c 2017          LAC      COUNT
141     004d 1014          SUB      ONE

```

```

137 004e fe00      BNZ  LOOP
    004f 002f
138 *
139 *   Calculate Energy at each frequency
140 *
141 *
142 0050 7007 CHECK  LARK  0,CS8
143 0051 712b      LARK  1,DAT28
144 *
145 0052 f800 MAGLP  CALL  ENERGY
    0053 00eb
146 0054 5990      SACH  *-,1,0
147 0055 f400      BANZ  MAGLP
    0056 0052
148 *
149 *   Compare Energies And Determine Decode Value
150 *
151 0057 7e03      LACK  3
152 0058 5010      SACL  ROWMX
153 0059 5011      SACL  COLMX
154 *
155 *   Find Row Peak
156 *
157 005a 7102 ROWS  LARK  1,2
158 005b 7021      LARK  0,DAT23
159 005c 2022      LAC   DAT14
160 005d 501a      SACL  ROWMAX
161 005e 6880 ROWL  LARP  0
162 005f 6898      MAR  *-,
163 0060 201a      LAC   ROWMAX
164 0061 1088      SUB  *
165 0062 fd00      BGEZ  ROWBR
    0063 0067
166 0064 3110      SAR   1,ROWMX
167 0065 2088      LAC   *
168 0066 501a      SACL  ROWMAX
169 0067 6891 ROWBR MAR  *-,1
170 0068 f400      BANZ  ROWL
    0069 005e
171 *
172 *   Find Column Peak
173 *
174 006a 7102 COLUMN LARK  1,2
175 006b 7029      LARK  0,DAT27
176 006c 202a      LAC   DAT18
177 006d 501b      SACL  COLMAX
178 006e 6880 COLL  LARP  0
179 006f 6898      MAR  *-,
180 0070 201b      LAC   COLMAX
181 0071 1088      SUB  *
182 0072 fd00      BGEZ  COLBR
    0073 0077
183 0074 3111      SAR   1,COLMX
184 0075 2088      LAC   *
185 0076 501b      SACL  COLMAX
186 0077 6891 COLBR MAR  *-,1
187 0078 f400      BANZ  COLL
    0079 006e
188 *
189 *   Check For Valid Signal Strength
190 *
191 007a 201b      LAC   COLMAX
192 007b 101a      SUB  ROWMAX
193 007c fd00      BGEZ  COLBIG
    007d 008a
194 007e 201b ROWBIG LAC   COLMAX      Reverse Twist
195 007f 1414      SUB  ONE,4
196 0080 fa00      BLZ   NOT
    0081 0022
197 0082 6a1b      LT    COLMAX
198 *
199 0083 800c      MPYK  12      Ideal Sdb = 6
200 *
201 0084 7f8e      PAC
202 0085 101a      SUB  ROWMAX

```

203	0086	fa00		BLZ	NOT	
	0087	0022				
204	0088	f900		B	VROW	
	0089	0094				
205	008a	201a	COLBIG	LAC	ROWMAX	Twist
206	008b	1414		SUB	ONE,4	
207	008c	fa00		BLZ	NOT	
	008d	0022				
208	008e	6a1a		LT	ROWMAX	
209			*			
210	008f	8003		MPYK	3	Ideal 4db = 3
211			*			
212	0090	7f8e		PAC		
213	0091	101b		SUB	COLMAX	
214	0092	fa00		BLZ	NOT	
	0093	0022				
215			*			
216			* Check for valid row tone			
217			*			
218	0094	6a1a		VROW	LT	ROWMAX
219			*			
220	0095	82ab		MPYK	683	683 = 1/6 = -8dB
221			*			
222	0096	7003		LARK	0,3	
223	0097	711c		LARK	1,DAT11	
224	0098	2014		LAC	ONE	
225	0099	5017		SACL	COUNT	
226	009a	6881	RVL	LARP	1	
227	009b	2ca8		LAC	++,12	
228	009c	7f90		SPAC		
229	009d	68a0		MAR	++,0	
230	009e	fb00		BLEZ	RCNT	
	009f	00a3				
231	00a0	2017		LAC	COUNT	
232	00a1	1014		SUB	ONE	
233	00a2	5017		SACL	COUNT	
234	00a3	f400	RCNT	BANZ	RVL	
	00a4	009a				
235	00a5	2017		LAC	COUNT	
236	00a6	fe00		BNZ	NOT	
	00a7	0022				
237			*			
238			* Check for valid column tone			
239			*			
240	00a8	6a1b		VCOL	LT	COLMAX
241			*			
242	00a9	82ab		MPYK	683	683 = 1/6 = -8dB
243			*			
244	00aa	7003		LARK	0,3	
245	00ab	7124		LARK	1,DAT15	
246	00ac	2014		LAC	ONE	
247	00ad	5017		SACL	COUNT	
248	00ae	6881	CVL	LARP	1	
249	00af	2ca8		LAC	++,12	
250	00b0	7f90		SPAC		
251	00b1	68a0		MAR	++,0	
252	00b2	fb00		BLEZ	CCNT	
	00b3	00b7				
253	00b4	2017		LAC	COUNT	
254	00b5	1014		SUB	ONE	
255	00b6	5017		SACL	COUNT	
256	00b7	f400	CCNT	BANZ	CVL	
	00b8	00ae				
257	00b9	2017		LAC	COUNT	
258	00ba	fe00		BNZ	NOT	
	00bb	0022				
259			*			
260			* Check 2ND Harmonic Energy Levels			
261			*			
262	00bc	7e2d		LACK	DAT29	Calculate address of
263	00bd	0110		ADD	ROWMX,1	row data locations
264	00be	5017		SACL	COUNT	
265	00bf	3917		LAR	1,COUNT	
266	00c0	7e08		LACK	CS9	
267	00c1	0010		ADD	ROWMX	

```

268 00c2 5017      SACL COUNT
269 00c3 3817      LAR 0,COUNT
270 00c4 f800      CALL ENERGY      Calculate energy level
    00c5 00eb
    00c6 6a1a
271 00c6 6a1a      LT ROWMAX
272
273 00c7 8fff      *      MPYK 4095      ROWMAX/8
274
275 00c8 7f90      *      SPAC
276 00c9 7f90      SPAC      ROWMAX/4 > 2nd Har
277 00ca fd00      BGEZ NOT
    00cb 0022
278
279 00cc 7e35      *      LACK DAT213      Calculate address of
280 00cd 0111      ADD COLMX,1      col data locations
281 00ce 5017      SACL COUNT
282 00cf 3917      LAR 1,COUNT
283 00d0 7e0c      LACK CS13
284 00d1 0011      ADD COLMX
285 00d2 5017      SACL COUNT
286 00d3 3817      LAR 0,COUNT
287 00d4 6880      LARP 0
288 00d5 f800      CALL ENERGY      Calculate energy level
    00d6 00eb
    00d7 6a1b      LT COLMAX      TEST CODE
289
290
291 00d8 8800      *      MPYK 2048      " "
292
293 00d9 7f90      *      SPAC      " "
294 00da fd00      BGEZ NOT      -12dB = 1/16.
    00db 0022
295
296
297
298 00dc 6916      *      *      Load recognized number and check that it is new
299 00dd 6915      *
300 00de 2210      *
301 00df 0011      *
302 00e0 5015      DMOV LAST2
303 00e1 1017      DMOV LAST
304 00e2 ff00      LAC ROWMX,2
    00e3 0022      ADD COLMX
    00e4 2015      SACL LAST
    00e5 1016      SUB COUNT      Return if same number
    00e6 fe00      BZ NOT
    00e7 0025
305
306 00e4 2015      LAC LAST
307 00e5 1016      SUB LAST2
    00e6 fe00      BNZ AGAIN      2 Passes to recognize
    00e7 0025
308
309 00e8 4a15      *      OUT LAST,2
310 00e9 f900      B AGAIN      <break>
    00ea 0025
311
312
313
314 00eb 2f13      *      ENERGY LAC NEGMAX,15      NEGMAX = >8000
315 00ec 0f81      ADD *,15,1
316 00ed 5817      SACH COUNT      -1/2 + CSn/2
317 00ee 6a98      LT *-
318 00ef 6d17      MPY COUNT
319 00f0 7f8e      PAC
320 00f1 5917      SACH COUNT,1      D2(CSn-1)/2
321 00f2 6aa8      LT **
322 00f3 6d17      MPY COUNT
323 00f4 7f8e      PAC
324 00f5 5917      SACH COUNT,1      D1*D2(CSn-1)/2
325 00f6 2f98      LAC *-,15
326 00f7 1f88      SUB *-,15
327 00f8 7f88      ABS
328 00f9 5888      SACH *      abs(D2-D1)/2
329 00fa 6a88      LT *
330 00fb 6d88      MPY *
331 00fc 7f8e      PAC      ((D2-D1)/2)^2
332 00fd 1f17      SUB COUNT,15      ((D2-D1)^2)/4-D1*D2(CSn-1)/2
333 00fe 7f8d      RET
334
335
    *      END

```

DTMF decoding doesn't necessarily require elaborate DSP routines; the routine presented here leaves room for several other DSP routines in the DSP μ P.

oscillators' coefficients (B1, D1, B2, and D2 in Fig 1) prior to the start of signaling. (The use of eight oscillators would decrease execution time but would require four times as much memory as two oscillators.) After initializing the resonators, the program loops repeatedly through the resonator code and generates samples of the appropriate high- and low-frequency tones. Then the program sums the pairs of tone samples. The DSP μ P then feeds this sum to an external D/A converter, and the resulting analog output is the DTMF signal.

Frequency specs aren't the only ones DTMF tones have to meet; duration specs apply also. According to AT&T specs, 10 digits/sec (or 100 msec/digit) is the maximum data rate for Touchtone signals. AT&T specifications state that within its allotted 100-msec interval, a tone must be present for at least 45 msec and no more than 55 msec. During the remainder of the 100-msec interval, the tone generator must be quiet to allow the receiver's DTMF decoder to settle. Therefore, a counter makes sure that the generated tone's duration meets the minimum time—approximately 45 msec—to minimize computing time. After the tone's been on for a sufficiently long time, the D/A converter is zeroed and maintained at the zero-output level so that the total on time and off time equals 100 msec.

Although DTMF tone-decoding schemes require con-

**TABLE 1—RECOMMENDED DFT LENGTHS
AT AN 8-KHz SAMPLING RATE**

1ST HARMONIC	N = 205 DUR = 25.6 mSEC = (18 20 22 24 31 34 38 42)	
2ND HARMONIC	N = 201 DUR = 25.1 mSEC = (35 38 43 47 61 67 74 82)	
COEFFICIENTS	N = 205	N = 201
	1ST HARMONIC	2ND HARMONIC
697	27906	15036
770	26802	11287
852	25587	7363
941	24295	3323
1209	19057	-10805
1336	16529	-16384
1477	12945	-22153
1633	9166	-27440

TABLE 2—DFT PROGRAM SPECIFICATIONS

DFT SIZE:	FIRST HARMONIC	N = 226
	SECOND HARMONIC	K = (16, 18, 20, 22, 28, 31, 34, 38)
		N = 222
		K = (32, 35, 39, 43, 55, 61, 67, 74)
PROGRAM WORDS	=	255 WORDS
DATA MEMORY WORDS	=	80 WORDS
SAMPLING FREQUENCY	=	9766 SAMPLES/SEC
SAMPLING INTERVAL	=	102.4 μ SEC
DFT LOOP TIME	=	45 μ SEC
TOTAL DFT TIME	=	23.2 mSEC
TIME REQUIRED BY THE DECISION LOGIC	=	150 μ SEC (ONE SAMPLE MISSED BETWEEN DFTs)

Glossary

Center frequency offset—the offset of the center of the recognition bandwidth from the nominal DTMF frequencies.

DFT—discrete Fourier transform.

DTMF—dual-tone multifrequency signaling system used by the telephone company for dialing.

Guard time—the duration of the shortest DTMF tone a detector will recognize.

IIR—infinite impulse response (a type of digital filter).

Log-Linear—transformation of logarithmically compressed data from a codec back to linear form.

Recognition bandwidth—the percent change from the nominal frequencies that a detector will tolerate.

Reverse twist—the condition that exists when a DTMF signal's row amplitude is greater than the column amplitude.

Standard twist—the condition that exists when a DTMF signal's column amplitude is greater than the row amplitude.

Talk-off—a measure of the detector's ability to ignore speech signals that look like DTMF signals.

Twist—the difference, in decibels, between the loudest row tone's amplitude and the loudest column tone's amplitude.

siderably more code than do generation schemes, the decoding program in Fig 3 takes less than twice as much code as the simpler generating program. Furthermore, both programs are much smaller than the total capacity of the DSP μ P. In this case, rather than being called as a result of a keystroke (as the generating algorithm is), the decoding algorithm continually processes signal samples and so must be interleaved with other DSP functions. The algorithm must run continually because, after all, it doesn't know whether or not DTMF tones are present until after it processes the input.

The discrete Fourier transform (DFT) algorithm employed in the program listing is known as Goertzel's algorithm (Ref 2). This algorithm is compact and needs only one real coefficient per frequency to determine magnitude (Fig 6); although extracting magnitude and phase requires complex coefficients and hence more complex programming, you can decode DTMF signals

```

*
*BEGIN DATA TABLES
*
      DATA      27906      Real Coeff N=205
      DATA      26802      697
      DATA      25597      770
      DATA      24295      851
      DATA      19057      941
      DATA      16527      1209
      DATA      12945      1336
      DATA      9166       1477
      DATA      1633       1633
*   2nd harmonic      Real Coeff N=201
      DATA      15036      1394
      DATA      11287      1540
      DATA      7363       1702
      DATA      3323       1882
      DATA      -10805      2418
      DATA      -16384      2672
      DATA      -22153      2954
      DATA      -27440      3266
*
      DATA      419        CLCK = Sample Frequency
      DATA      >000A      MODE
*
      DATA      >7FFF      POSMAX = Mask for data in
      DATA      >8000      NEGMAX = Mask for data out
*
*
TABLE   DATA      1        ONE
*   Start of Program      *****
*
START   SOVM
      LDPK      0
      LARP      0          <Break>
      LARK      0,ONE
      LACK      TABLE
NEXT    TBLR      *        Initialize Coefficients
      SUB       ONE
      BANZ      NEXT
      OUT       MODE,0     Set AIB Mode
      OUT       CLCK,1     Set AIB Clock
*
*
* Load not recognized symbol
*
NOT      LACK      >FF
      DMOV      LAST
      SACL      LAST
*
AGAIN    ZAC
      LARK      0,31      Zero DFT Loop Variables
      LARK      1,DAT11
      BV        ZERO
ZERO     LARP      1
      SACL      **+,0,0
      BANZ      ZERO
*
*   Take data and calculate DFT loop
*
      LACK      205      SET DFT LOOP VARIABLE
*
LOOP     SACL      COUNT
      LARK      0,CS16
      LARK      1,DAT216
*
      SUB       ONE,2
      BGZ       WAIT
      LARK      0,CS8
      LARK      1,DAT28

```

Fig 4—This amendment of Fig 3's listing adapts the tone-generating routine to 8k-sample/sec operation. It's a substitute routine for lines 45 to 122.

DTMF decoders that meet AT&T specs usually stop decoding tones if they detect the presence of speech.

simply by extracting the magnitude of a tone's frequency components and ignoring their phase. In addition, instead of waiting for a complete sample set to begin processing, Goertzel's algorithm processes each sample as it arrives.

Goertzel's algorithm takes the form of a series of second-order IIR (infinite-impulse-response) filters. Notice that, in Fig 6, you can divide the directed graph into two parts: a left-hand part that includes the two feedback elements (boxes marked "delay") and a right-hand portion leading to the output that has no feedback elements. For DTMF decoding, you are interested only in the last iteration (N-1) of the algorithm. Consequently, because the right-hand branches don't involve feedback, there's no need to execute these branches of

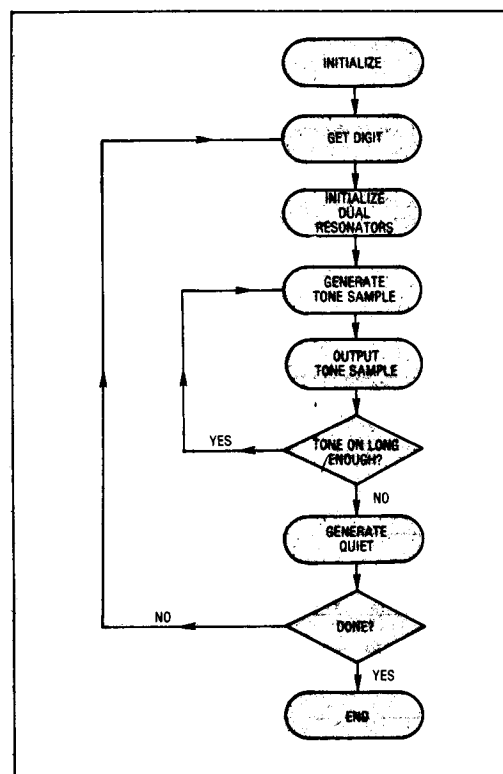


Fig 5—The directed graph in Fig 1 translates into a program that follows this flowchart. Note the step that produces a quiet period.

TABLE 3—TONE DECODER TEST RESULTS

BANDWIDTH TEST RESULTS:					
STONE	%HIGH	%LOW	STONE	%HIGH	%LOW
697	2.5	3.5	1209	2.4	3.0
770	3.7	2.3	1336	2.3	2.5
852	3.9	1.7	1477	1.3	2.9
941	3.3	1.7	1633	2.4	1.6
SPECIFICATIONS REQUIRE: MIN = 1.5% AND MAX = 3.5%					
AMPLITUDE RATIO TEST RESULTS					
	TWIST		REVERSE TWIST		
SPECIFICATIONS	>4.0 dB		>8.0 dB		
DIGIT 1	5.3		8.4		
DIGIT 5	5.7		8.0		
DIGIT 9	8.3		9.7		
DIGIT 16	5.4		9.5		
DYNAMIC RANGE: 25 dB (SPECIFICATION 25 dB)					
NOISE TEST: PASSES AT -24 dBV					
TALK-OFF IMMUNITY: ONE FALSE RECOGNITION PER 1000 CALLS (SPEC 1500)					

the algorithm until the last iteration of the algorithm.

What's not obvious from the directed graph of the algorithm is that the left-hand constant, $2\cos(2\pi k/N)$, is the same as the right-hand constant, W_N^k , for calculating the magnitude of DTMF signals. W_N^k is a complex number, and the left-hand constant is not. However, the program calculates the magnitude squared of the output of the algorithm. Squaring a complex number always yields a real number, and in this case, squaring the right-hand constant yields a real number that's the same as the left-hand constant. Therefore, Goertzel's algorithm, adapted to DTMF decoding, not only executes quickly because it has few steps, but it also takes up little memory space because it uses few constants.

Given a time-ordered sample set of size N, processing each sample means you'll do N iterations of the algorithm. If k is the frequency you're solving the transform for, then the values k and N determine the coefficients of each IIR filter. The values of k and N and the sampling rate also determine how accurately the transform discriminates between in-band and out-of-band frequencies.

Specifically, k is a discrete integer corresponding to the frequency you're solving for. It's defined as

$$k = N \times \text{frequency} / (\text{sample rate}).$$

(Note that you must round off the frequency of interest to an integer.) W_N^k is a frequently encountered constant in digital signal processing. It's defined as

$$W_N^k = \exp(-2\pi jk/N).$$

Because the sampling rate and the frequencies you're

The decoding algorithm processes signal samples continuously, so the DSP μP must interleave the decoding with other DSP functions.

extracting are fixed (by the phone company), the sample-set length (N) is the only parameter you can vary. In order to obtain the best performance from the transform, the length of the sample set must be optimized with respect to two conflicting criteria: The sample set must be small enough so that the decoder can accumulate a complete set in an interval that's short enough to keep up with the DTMF digit-transmission rate; conversely, the sample set must be long enough so that the transform discriminates between in-band and out-of-band signals. An exhaustive and inelegant computerized search of all possible combinations of k and N resulted in the sample lengths listed in Tables 1 and 2 and used in the Fig 3 and 4 program listings.

Companding not accounted for

This design assumes that the analog input is linearly encoded. This is often not the case, because many phone signals are compressed logarithmically by a codec. In such cases, you must first perform a log-to-linear expansion before submitting the samples to the DFT algorithm. (Ref 3 describes how to do companding with the TMS32010 if your system doesn't incorporate companding hardware.)

Fig 7 shows how samples are fed, in effect, into a

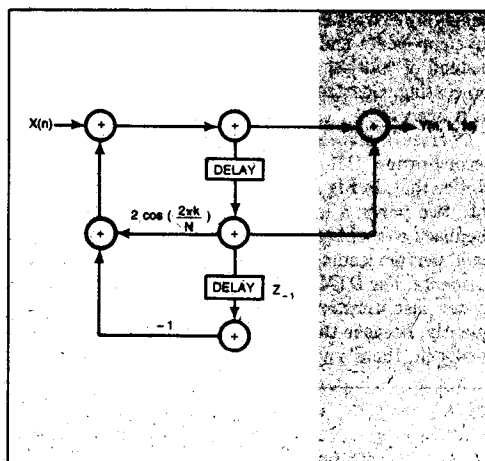


Fig 6—A simplified form of Goertzel's algorithm, represented by this directed graph, decodes DTMF tones. A program that implements it can save processing steps by performing the calculations illustrated by the right-hand portion of this graph for just the last iteration. Furthermore, for DTMF-decoding purposes, this compact algorithm requires only one constant per frequency because both the right-hand and left-hand constants have the same value.

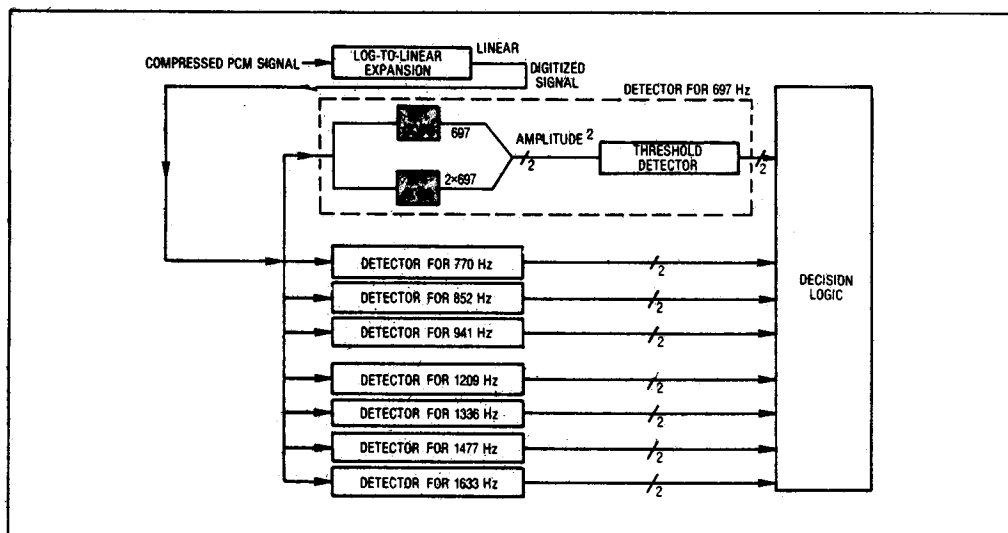


Fig 7—The tone decoder employs 16 of the transforms shown in Fig 6. They extract each of the eight Touchtone frequencies and their second harmonics.

Goertzel's algorithm, a discrete Fourier transform algorithm, operates as a series of second-order IIR filters.

parallel array of 16 DFT algorithms. There is one DFT for each of the eight frequencies and each of the second harmonics of the eight frequencies. You need the second harmonics as well as the fundamentals to discriminate between speech and DTMF tones. Of course, they execute serially because they are sections of code—not physical devices.

As Fig 8's flowchart shows, after initializing the processor, the program feeds the first sample to the IIR filters. After all the samples have been processed, each filter's current value is squared. This operation yields the magnitude of the strength of the signal at each of the eight DTMF frequencies and each second harmonic of the DTMF frequencies.

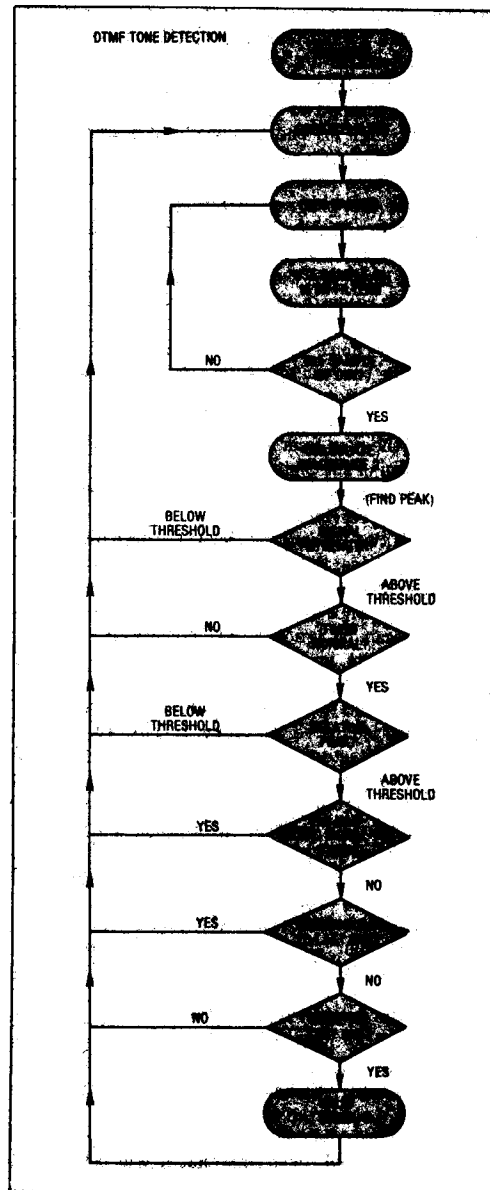
The program next compares the data against several thresholds. It performs four principal checks. First, after finding the strongest signals in the high- and low-frequency groups, it simply determines whether any valid DTMF tones are present at all. If the strongest signals are not above a minimal value (2^{-12} in Fig 3's program), the program does no more processing and begins collecting another sample set.

Second, if valid DTMF tones are present, the program checks the strongest signals in the low and high (row and column) groups for twist—the ratio of the row-tone amplitude to the column-tone amplitude. This ratio must be between certain values for the DTMF tones to be valid. (Because of the frequency response of telephone systems, the high tones are attenuated. Consequently, the phone company doesn't expect the high and low groups to have exactly the same amplitude at the receiver, even though they were transmitted at the same strength.)

Third, the program compares the amplitude of the strongest signal in each group to the amplitudes of the rest of the tones in its group. Again, the strongest tone must stand out from the other tones in its group by a certain ratio.

Finally, the program checks to see that the strongest signals are above one threshold while their corresponding second harmonics are below another threshold. Checking for strong harmonics insures that the DSP system won't confuse speech for DTMF signals. (Speech has significant even-order harmonics; DTMF signals don't.)

Fig 8—The tone-decoder program does far more than simply detect the presence of DTMF signals; it also performs an elaborate series of checks to ensure that the tones are within specifications and that a valid tone is new data that must be acted on.



The DTMF-decoding program checks the signal pair to establish tone validity, and then it determines whether the pair constitutes new information.

If the DTMF signal pair passes all these comparisons, then it's a valid tone pair that corresponds to a digit. Just because it's valid, however, doesn't mean that the corresponding digit is necessarily new information. The remaining two steps of the program compare the current digit to the two most recently derived digits. First, the program checks to see if the current digit is the same as the second-to-last digit. If they match, then the program assumes the tone hasn't changed lately. If they differ, it performs one final check to see if the current digit matches the last digit received. If these are the same, then the DTMF tone has changed recently and remained stable for two iterations. This means you finally have a valid new digit. If they don't match, it means the tone has changed since the last sample was acquired but hasn't remained stable long enough. Consequently, the program loops back without signaling that a new digit has arrived. If the new tone is really valid and stable, the next iteration of the algorithm will recognize the digit as valid because the new current digit will now match the previously received digit.

There are two reasons for checking three successive digits at each pass. First, the check eliminates the need to generate hits every time a tone is present; acknowledging it only once is enough. As long as the tone is present, it can be ignored until it changes. Second, comparing digits improves noise characteristics and speech immunity.

The implementation of the decoder algorithm follows the specification listed in Table 2. The TMS32010's Harvard architecture separates data and program memory. The data memory is on chip. The program keeps the tables required by the decoding algorithm in on-chip data memory. These tables take up more than half the available data-memory locations. Depending on your application, you might have to store the tables in program memory and move the tables onto the chip every time the decoding algorithm runs. This will free the on-chip memory for other uses, but it will obviously increase the decoding algorithm's execution time.

Checking the decoder's performance

Evaluating the performance of a DTMF decoder is more difficult than evaluating the performance of a DTMF generator. You can check the generator very simply with a spectrum analyzer. To test the decoder, on the other hand, you have to determine not only that it will decode valid tones, but that it will both reject invalid signals and operate properly in the presence of

noise. Testing the decoder using AT&T's published test method is an all-day affair and requires a specific instrumentation suite. Prerecorded tapes of various test tones speed things considerably. For example, Mitel's (San Diego, CA) \$90 CM7291 test cassette tape cuts the evaluation time of DTMF tone receivers to less than 90 minutes, according to the company (Ref 4).

The TMS32010's DTMF decoder was tested against the Mitel test tape. The test results given in Table 3 indicate that the receiver can detect all tones. And the receiver bandwidths conformed almost exactly to all AT&T specs. There were only three tones for which the decoder was slightly off. In two instances, results were 0.2% too large and, in one instance, 0.2% too small. The other AT&T specs were met perfectly, including the twist's dynamic range at 25 dB, the guard time at 20 msec, and the white-noise test at 24 dBV. **EDM**

References

1. Clark, N V, "DTMF Encoder Demonstration," Texas Instruments internal publication, February 1984.
 2. Oppenheim, A V, and Schafer, R W, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1975 (see Section 6.1).
 3. *Companding Routines for the TMS32010*, Application Note SPRA001, Texas Instruments, Dallas, TX.
 4. *Tone Receiver Test Cassette #CM7291*, Mitel Technical Data Manual, Mitel Semiconductor, 2321 Morena Blvd, Suite M, San Diego CA 92110. Phone (619) 276-3421.
 5. *Touch-Tone/RTM calling—Requirements for Central Office*, AT&T Compatibility Bulletin No 105, August 8, 1975.
-