Texas Instruments Home Computer

Speech Editor

SOLID STATE SOFTWARE™ **COMMAND MODULE**

The Texas Instruments Solid State Speech™ Synthesizer must be attached to the computer while the Speech Editor Command Module is in use.

Discover the excitement of a "voice-on-a-chip"! Add speech to your Home Computer with this module and the Solid State Speech™ Synthesizer.

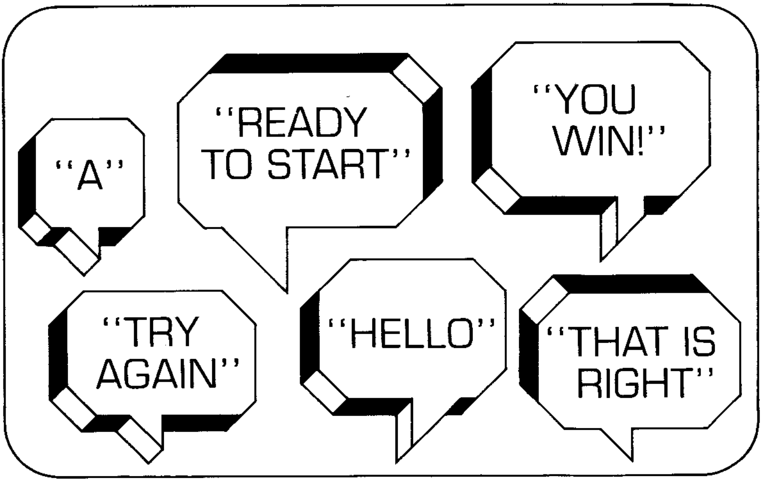
Quick Reference Guide

Note that the key sequences required to access special functions depend on the type of computer console you have. Important keystroke sequences are summarized here for your "quick reference."

TI-99/4	*TI-99/4A*	
SHIFT C **(CLEAR)**	**FCTN 4** **(CLEAR)**	Erases all the words typed in the Speech Editor option.
SHIFT T **(ERASE)**	**FCTN 3** **(ERASE)**	Erases the word or phrase on which the cursor is positioned.
SHIFT E **(Up arrow)**	**FCTN E** **(Up arrow)**	Moves the cursor on the screen in the direction indicated by the arrow. When you move the cursor off a word, the computer checks to make sure the word is in the Speech Synthesizer vocabulary.
SHIFT X **(Down arrow)**	**FCTN X** **(Down arrow)**	
SHIFT S **(Left arrow)**	**FCTN S** **(Left arrow)**	
SHIFT D **(Right arrow)**	**FCTN D** **(Right arrow)**	
SHIFT F **(DELETE)**	**FCTN 1** **(DELETE)**	Deletes the character covered by the cursor.
SHIFT G **(INSERT)**	**FCTN 2** **(INSERT)**	Lets you insert characters into the text on the screen. When finished inserting characters, press an arrow key or **ENTER**.
SHIFT Q **(QUIT)**	**FCTN =** **(QUIT)**	Leaves the Speech Editor and returns to the master title screen. All data you have entered will be erased.

TEXAS INSTRUMENTS
HOME COMPUTER

Speech Editor



This *Solid State Software*™ Command Module is designed to be used with the Texas Instruments Home Computer. Its preprogrammed solid-state memory expands the power, versatility, and capability of your Home Computer.

See important warranty information at back of book.

INTRODUCTION

The integration of human voice and machines holds an age-old fascination. The excitement over the invention of the telephone, radio, and television was due in part to the experience of hearing a human voice emerge from a machine. Now Texas Instruments has made innovative breakthroughs in *Solid State Speech*™ technology. With the Texas Instruments Speech Synthesizer in place and the *Solid State Software*™ Speech Editor Command Module inserted in the console, you can tap the speech capabilities of your Home Computer.

The Speech Editor Command Module provides two methods of utilizing the vocabulary of the Speech Synthesizer. You can select the Speech Editor feature, or you can choose to extend your TI BASIC programs to include speech. With either option you can easily create spoken words, phrases, and sentences.

The Speech Editor feature of the module requires no programming skills. You can immediately listen to and experiment with computer-generated speech by simply typing words, phrases, and sentences selected or formed from the resident vocabulary of the Speech Synthesizer.

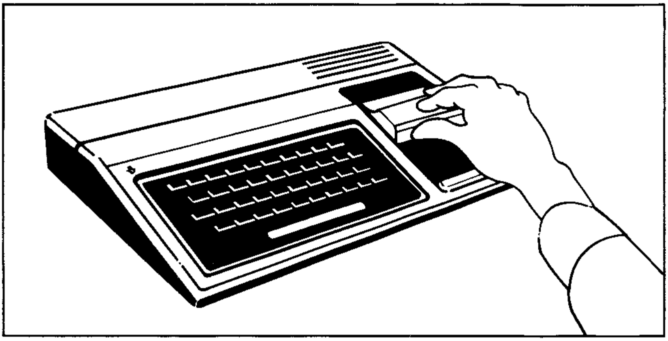
The Speech Editor Command Module also contains two subprograms that let you add speech to your TI BASIC programs. Customize your computer games! Reinforce important concepts in your educational applications! Experiment and discover the excitement of a "voice-on-a-chip"!

This manual can help you explore the features of the Speech Editor Command Module. It offers a "hands-on" guide to the Speech Editor option and to programming speech in TI BASIC. Be sure to try out the examples as you go through the manual and become familiar with the Speech Synthesizer's capabilities.

USING THE *SOLID STATE SOFTWARE*™ COMMAND MODULE

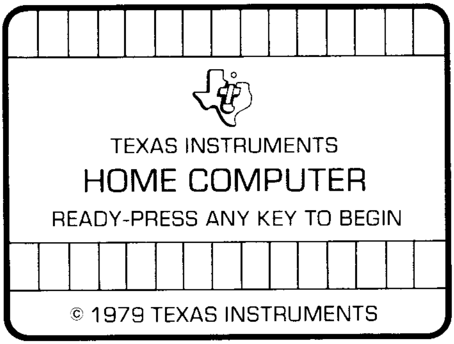
The Texas Instruments Speech Synthesizer must be attached to your Home Computer when you use the Speech Editor Command Module. (See the *Speech Synthesizer* owner's manual for complete information on handling, installing, and caring for the TI Speech Synthesizer.)

An automatic reset feature is built into the computer. When a module is inserted into the console, the computer returns to the master title screen. All data or program material you have entered will be erased.



Note: Be sure the module is free of static electricity before inserting it into the computer. (See page 38.)

1. Turn the computer ON and wait for the master title screen to appear. Then slide the module into the slot on the console.



TEXAS INSTRUMENTS

HOME COMPUTER

READY-PRESS ANY KEY TO BEGIN

© 1979 TEXAS INSTRUMENTS

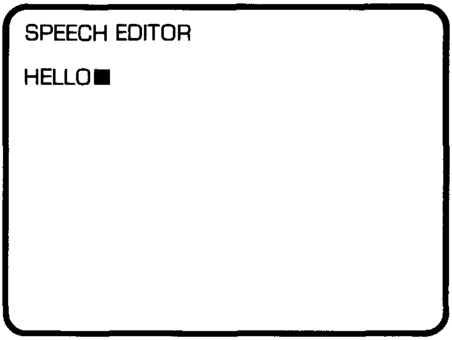
2. Press any key to make the master selection list appear. The title, SPEECH EDITOR, will be on the list.
3. To select SPEECH EDITOR, press the number key corresponding to the number shown next to SPEECH EDITOR. To select TI BASIC, press the **1** key.

(*Note:* To remove the module, *first* return the computer to the master title screen by pressing **SHIFT Q**. Then remove the module from the slot. If you have any problem inserting the module, or if it is removed from the slot while in use, please see "In Case of Difficulty" on page 39.)

USING SPEECH EDITOR

The Speech Editor option provides a convenient means of hearing the vocabulary of the Speech Synthesizer. You can also experiment with constructing compound words and complete sentences.

To select Speech Editor, press the appropriate key when the master selection list is on the screen. A new display then appears with the words SPEECH EDITOR at the top. On the next line, the cursor (a flashing square) indicates that the computer is waiting for you to type a word, phrase, or sentence. Type the word HELLO. This is how the screen looks:



```
SPEECH EDITOR

HELLO■
```

Now, press the **ENTER** key, and the Speech Synthesizer says "Hello." The flashing cursor disappears while a word is recited. Press **ENTER** again, and the Speech Synthesizer repeats the word

Continue by typing the words I AM A COMPUTER. Be sure to put a space between each of the words. Then press **ENTER** again. After the computer has spoken this sentence, hold down the **SHIFT** key and press **C (CLEAR)**. The sentence is erased, leaving the words SPEECH EDITOR and the cursor on the screen.

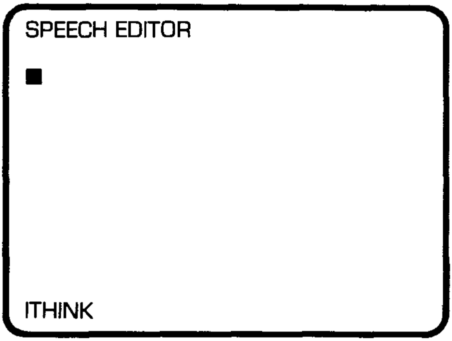
Speech Separator Characters

Punctuation marks and several other symbols listed in the table below are called *separator characters*. Separator characters serve a dual purpose in the Speech Editor Command Module. They are used to indicate the end of a word, phrase, or sentence. Each time the computer encounters a separator character, it checks the preceding word against the vocabulary. Also, these symbols specify the length of time the computer pauses between words. You can use these separator characters to give synthesized speech the cadence of natural conversation. The speech separator characters and the time pauses they represent are:

SPEECH SEPARATOR CHARACTERS

SYMBOL	*PAUSE* (seconds)
+	0
space	.1
–	.2
,	.3
;	.5
:	.8
.	1.0

Take time to experiment with the separator characters. First, see what happens when a separator character is not used between words. Type the words ITHINK without including a separator symbol, and press **ENTER**. Instead of hearing the computer say "I THINK," the letters "ITHINK" appear and flash at the bottom of the screen, indicating that no such word is in the vocabulary.



```
SPEECH EDITOR
■




ITHINK
```

Press **SHIFT C** to clear the screen.

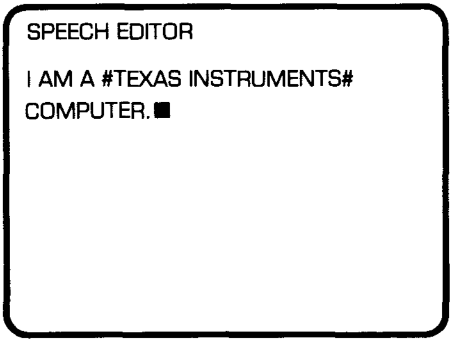
Now, try using different separator characters and listening to the pauses between words. Retype I THINK, putting a space between the words. Then type a period, a space, and the words I KNOW followed by a period. Press **ENTER**. Listen for the length of the pause. Now type these words with different separator symbols.

I THINK. I KNOW, I .... THINK – .

Press **ENTER**. (Notice that placing several separator characters together causes a multiple delay. The only exception to this is the space character. More than one space is ignored.) Press **SHIFT C** to clear the screen.

Speech Synthesizer Phrases

Some frequently used combinations of words are stored as single phrases in the Speech Synthesizer. These include phrases such as I WIN, TRY AGAIN, and TEXAS INSTRUMENTS. To designate the use of a phrase from the Speech Synthesizer vocabulary, place the number sign (#) at the beginning and end of the phrase. Try typing I AM A #TEXAS INSTRUMENTS# COMPUTER. Then, press **ENTER**.

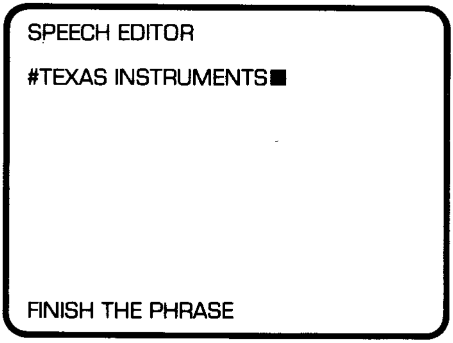


```
SPEECH EDITOR

I AM A #TEXAS INSTRUMENTS#
COMPUTER.■
```

After the phrase is spoken, clear the screen by pressing **SHIFT C**.

If you do not include the number sign (#) at the beginning and end of the phrase, the computer gives error messages. Try typing TEXAS INSTRUMENTS, leaving out both number signs. When you finish typing TEXAS and the space, the computer checks the word. Since TEXAS isn't in the vocabulary, the computer beeps and moves the word TEXAS to the lower left corner and flashes it to indicate an error has been made. (The flashing word is removed when you press a key to begin typing a new word.)

Next, type the phrase #TEXAS INSTRUMENTS and leave off the final number sign. Press **ENTER**. Without the closing sign (#), the computer will not search the vocabulary for the phrase. Instead, it displays the error message FINISH THE PHRASE.

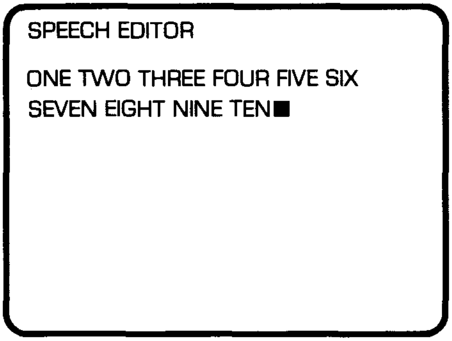


```
SPEECH EDITOR

#TEXAS INSTRUMENTS■







FINISH THE PHRASE
```

To erase the error message and correct the error, type the number sign at the end of the phrase. Now press **ENTER** and the computer says the phrase TEXAS INSTRUMENTS. Clear the screen by pressing **SHIFT C**.

Speaking Numbers

The Speech Synthesizer unit recognizes the names of whole numbers from 0 through 999 and the digits 0 through 9. Try entering numbers in both ways. First, type the names of the whole numbers from one to ten. Then press **ENTER**.



```
SPEECH EDITOR

ONE TWO THREE FOUR FIVE SIX
SEVEN EIGHT NINE TEN■
```

Now, erase the screen by pressing **SHIFT C**, and type: 0123456789.

(When you are typing the numerals 0 through 9, separator characters may be omitted.) Press **ENTER**.

The computer recognizes the plus sign (+) for positive numbers and the minus sign (–) for negative numbers. If you type a sign before a number, the computer says the word "positive" or "negative" before reciting the number.

You can also type fractions in decimal form. A period typed between numerals is considered a decimal point. Type 12.34 and press **ENTER**. The computer says "one-two-point-three-four."

Homonyms and Homographs

Many words in the English language are pronounced similarly but spelled differently (homonyms). Others are pronounced differently but spelled the same (homographs). The Speech Synthesizer has both homonyms and homographs in its vocabulary. An example of homonyms is TO, TWO, TOO, and 2. The Speech Synthesizer recognizes these different ways of spelling words that sound the same.

Homographs pose a different problem. Since both spellings are identical, the computer can't tell which pronunciation is wanted. To differentiate between two pronunciations a number is placed immediately after one of the homographs. For example, READ is pronounced "rēd" like reed. READ1 is pronounced "rĕd" like the color red. Check the vocabulary list in *Appendix A* for the designations for these and other homographs. To hear the different pronunciations, try the following example:

```
THE THE1 A A1 READ READ1
```

Press **ENTER** and listen to the computer say each word. When you are finished, press **SHIFT C** to clear the screen.

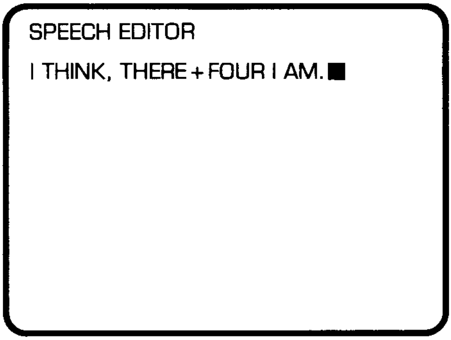
Forming Additional Words

At times you may try to type a word that isn't in the Speech Synthesizer vocabulary. For instance, start typing I THINK, THEREFORE I AM. The computer checks the preceding word each time it encounters a separator character. Thus, after you type THEREFORE and a space, the computer checks and finds THEREFORE is not in the vocabulary. It then flashes the word THEREFORE in the lower left corner of the display.

The word continues to flash until you press the next key.

Even though THEREFORE is not in the vocabulary, you can form it by using words that *are* in the vocabulary list. THERE and FOUR are a part of the vocabulary and can be connected with the plus sign (+) to form a word that sounds like THEREFORE.

The plus sign connects words without a pause between them and thus creates the desired effect. Complete the sentence using the constructed word THERE + FOUR.



```
SPEECH EDITOR

I THINK, THERE+FOUR I AM.■
```

Press **ENTER** to hear the computer say the sentence.

Other words such as SIX + TEEN and SOME + THING can be formed this way. Think of additional words that can be constructed from the Speech Synthesizer vocabulary.

Display Capacity

You can type ten screen lines of text before filling the screen. When the ten lines are filled, the computer displays the message OUT OF SPACE. To enter more words, currently displayed words must be deleted or the screen must be cleared.

Each screen line can contain up to twenty-eight characters. If you are typing a word and reach the end of the screen line, the entire word is automatically moved to the beginning of the next line. This is the "wrap-around" feature of the Speech Editor. Type the following words and observe this feature: TWO THREE FOUR FIVE SIX SEVEN EIGHT NINE. The word SEVEN started on the first line but was shifted to the second line because space wasn't available for the entire word on the first line. Press **ENTER** to hear the words.

Editing and Correcting

You can easily edit the displayed text. The following keys are used for erasing, correcting, and changing the words.

Try typing several lines of text, and then use the edit keys. Take your time and experiment with all the features of the Speech Editor option.

SHIFT C (CLEAR)

This key erases from the screen all words you have typed.

SHIFT ↑ (UP)
SHIFT ↓ (DOWN)
SHIFT ← (LEFT)
SHIFT → (RIGHT)

The arrow keys position the cursor on the screen. **SHIFT ↑ (UP)** moves the cursor up one line each time it is pressed; **SHIFT ↓ (DOWN)** moves the cursor down one line. **SHIFT ← (LEFT)** moves the cursor one space to the left each time it's pressed, and **SHIFT → (RIGHT)** moves the cursor one space to the right. When you move the cursor off a word using an arrow key, the computer checks to make sure the word is in the Speech Synthesizer vocabulary.

SHIFT F (DEL)

This key deletes the character covered by the cursor. Use the arrow keys to position the cursor over the character to be removed. Press SHIFT F. The character is deleted and all characters on the line to the right of that character are moved to the left. Press SHIFT F one time for each character you wish to delete. *Note*: Number signs (#) cannot be deleted with SHIFT F. The entire phrase must be erased using SHIFT T.

SHIFT G (INS)

This key lets you insert characters into the text. Use the arrow keys to position the cursor where you wish to insert one or more characters. Press SHIFT G, and type the characters to be inserted. The characters to the right of and under the cursor are shifted to the right as new characters are inserted. When you have finished inserting characters, press an arrow key or ENTER.

SHIFT T (ERASE)

This key is used to erase individual words or phrases. When the cursor is positioned over any letter of a word, pressing SHIFT T erases the entire word and any separator characters that follow the word. If the cursor is positioned over a separator character that follows a word, pressing SHIFT T erases both the separator character and the word immediately preceding it.

PROGRAMMING SPEECH IN TI BASIC

When you're ready to explore incorporating speech into your TI BASIC programs, press **SHIFT Q** to return to the master title screen. Then press any key to begin. When the master selection list appears, press **1** to select TI BASIC.

The Speech Editor Command Module contains two subprograms that enable you to include speech in TI BASIC programs. These two subprograms are labeled CALL SAY and CALL SPGET. CALL SAY instructs the computer to speak the word or phrase you type. CALL SPGET assigns the speech code for a word you designate to a string variable. You can apply this string variable in a variety of ways, including using it immediately with CALL SAY, viewing the speech code patterns, or preparing a master vocabulary file by storing words on a cassette tape or disk for later use with CALL SAY.

Both subprograms can be used in either the Immediate Mode or in TI BASIC programs. The Speech Editor Command Module must be plugged into the console, and the Speech Synthesizer must be attached to the computer. If not, the program stops, and an error message is displayed.

Using CALL SAY

It's easy to use the speech capabilities of your computer system through TI BASIC. Just type CALL SAY and the words you want the computer to say. Try it! Type

```
CALL SAY("HELLO, I LIKE YOU.")
```

Notice that the words you want the computer to speak are enclosed in parentheses and quotation marks. Now, press **ENTER** and listen to the computer speak.

Experiment with some of the words, phrases, or sentences from the vocabulary list. If you type in a word or phrase that isn't in the resident vocabulary, the computer says "UHOH."

Writing programs that include speech is just this simple.

The following program uses CALL SAY to give verbal prompts when it's time to enter certain information. Enter the program and RUN it to see how the verbal prompts and on-screen messages can work together.

```
NEW
100 CALL CLEAR
110 PRINT "TYPE A NUMBER."
120 CALL SAY["TYPE A NUMBER."]
130 INPUT FIRST
140 PRINT "ONE MORE TIME."
150 CALL SAY["ONE MORE TIME."]
160 INPUT SECOND
170 LET SUM=FIRST+SECOND
180 CALL SAY["HERE IS THE SUM"]
190 PRINT "THE SUM IS"; SUM
200 END
RUN
```

Lines 110 and 120 prompt you, both visually and audibly, to enter your first number. When the program stops at line 130 for your first entry, type a number and press **ENTER**. Lines 140 and 150 prompt you for the second number. When the program stops at line 160, type in the second entry and press **ENTER**. At line 170, the computer adds the numbers. Line 180 tells you the sum will be shown, and line 190 prints the answer.

Using Variables With CALL SAY

Occasionally, you may want to include the same verbal message several times in a program. For example, the phrase "PLEASE ENTER YOUR GUESS" may be said several times in a game program. You don't have to type the words each time. Instead, you can assign a string variable to the word or phrase, as follows:

```
NEW
10 A$="PLEASE ENTER YOUR GUESS"
20 CALL SAY(A$)
```

Now type RUN to hear the computer say the phrase. Each time you want the phrase spoken in the program, simply include the program line CALL SAY(A$).

You can join two or more string variables by using the symbol for *concatenation* (&). For example:

```
NEW
10 B$="#YOU WIN#"
20 C$="#I WIN#"
30 D$="ONE POINT"
40 CALL SAY(B$&D$)
50 CALL SAY(C$&D$)
```

Type this short program and RUN it. The computer says "You win one point" and "I win one point." (Notice that number signs (#) are included with the stored phrases "You win" and "I win.")

You can also join string variables by placing an empty (or *null*) string between the string variables. The null string is symbolized by "". Using the null string, you'd type line 40 as

```
40 CALL SAY(B$,"",D$)
```

(See page 24 for a more detailed discussion of the use of the null string.)

A Sample Application

Many of the procedures discussed in the Speech Editor section are also applicable with the CALL SAY subprogram. Let's briefly review some of the points again.

- Words must be separated by a valid separator character. Separator characters designate the end of words, phrases, and sentences and the length of pause between words. (See page 6 for a list of separator characters.) When using CALL SAY, numbers must be separated from other characters by a separator symbol.
- Combinations of words that are stored as a phrase must be preceded and followed by the number sign.
  Example: #GOOD WORK#
- New words may be constructed from words in the Speech Synthesizer vocabulary. Type the plus sign (+) between words to construct a new word.
  Example: SOME+THING,
  THERE+FOUR, SIX+TEEN

Now, try using speech in a game program called "Guess the Number." In this activity, you try to guess a number the computer has selected. The game begins after the computer generates a random whole number between 1 and 100. Enter a guess, and the computer tells you whether the answer is the same, higher or lower than your guess. Then enter another guess if necessary. The game continues until you guess the correct number, exhaust the allowed number of guesses, or give up.

Enter the following program into your computer.

```
NEW
90 RANDOMIZE
100 REM–GUESSING GAME
110 TRIES=0
120 ANSWER=INT[100*RND]+1
130 CALL CLEAR
140 CALL SAY["LET ME SEE.."]
150 A$="O YES,I HAVE IT NOW"
160 B$="TRY TO THINK OF THE NUMBER."
170 CALL SAY[A$,"",B$]
180 PRINT "TYPE YOUR GUESS. THEN PRESS
'ENTER': "
190 INPUT GUESS
200 TRIES=TRIES+1
210 IF GUESS<=ANSWER THEN 240
220 CALL SAY["LOWER"]
230 GOTO 260
240 IF GUESS=ANSWER THEN 310
250 CALL SAY["HIGHER"]
260 IF TRIES=8 THEN 330
270 CALL SAY["#TRY AGAIN#"]
280 CALL CLEAR
290 GOTO 180
300 REM-END ROUTINES
310 CALL SAY["#YOU WIN#..."]
320 GOTO 340
330 CALL SAY["UHOH, SO SORRY..."]
340 PRINT "THE ANSWER IS";ANSWER
350 PRINT "TRIES=";TRIES
360 END
```

Now RUN the program, and enjoy the new dimension speech adds to this game.

Notice each occurrence of CALL SAY. We'll list the program a few steps at a time and discuss it as we go along.

```
NEW
90 RANDOMIZE
100 REM–GUESSING GAME
110 TRIES=0
120 ANSWER=INT(100*RND)+1
130 CALL CLEAR
140 CALL SAY("LET ME SEE..")
```

Line 120 generates a random whole number from 1 through 100. Line 140 is the first occurrence of the CALL SAY subprogram. Note that the two periods cause a two second pause. When the program runs, the computer says this phrase.

```
150 A$="O YES,I HAVE IT NOW"
160 B$="TRY TO THINK OF THE NUMBER."
170 CALL SAY(A$,"",B$)
```

In lines 150 and 160, string variables are assigned to the phrases. Notice the null symbol and commas between A$ and B$ in line 170. The computer will say the words assigned to A$ and B$. You also could have concatenated in line 170: CALL SAY(A$&B$).

```
180 PRINT "TYPE YOUR GUESS. THEN PRESS
'ENTER': "
190 INPUT GUESS
200 TRIES=TRIES+1
```

TYPE YOUR GUESS AND PRESS ENTER prints on the screen. Line 200 "counts" the number of guesses you make.

```
210 IF GUESS<=ANSWER THEN 240
220 CALL SAY("LOWER")
230 GOTO 260
240 IF GUESS=ANSWER THEN 310
250 CALL SAY("HIGHER")
```

Line 210 checks to see if your guess is less than or equal to the answer. If your guess is greater than the answer, line 220 tells you to select a lower number. Then the program goes to line 260 to check your number of tries. If your guess is the correct answer, line 240 directs the program to go to and perform line 310. Through the tests in lines 210 and 240, the computer knows if your guess was too low. Line 250 tells you to guess a higher number.

```
260 IF TRIES=8 THEN 330
270 CALL SAY("#TRY AGAIN#")
280 CALL CLEAR
290 GOTO 180
300 REM-END ROUTINES
310 CALL SAY("#YOU WIN#...")
320 GOTO 340
330 CALL SAY("UHOH, SO SORRY...")
340 PRINT "THE ANSWER IS";ANSWER
350 PRINT "TRIES=";TRIES
360 END
```

Line 260 sets the maximum number of guesses at 8. The game is over after 8 guesses. If you have less than 8 guesses, the computer tells you to try again. Notice the number signs enclosing the resident phrase #TRY AGAIN#. The computer congratulates you when you win and sympathizes with you if you lose, and then shows you the answer and the number of tries.

For general speech programming in TI BASIC, CALL SAY is the only subprogram you need to use.

CALL SPGET requires more advanced programming techniques, as discussed in the following section.

Using CALL SPGET

The words and phrases are stored in the Speech Synthesizer unit as unique patterns of code. CALL SPGET actually calls the code pattern for a resident word and assigns it to a string variable. You can then apply this string variable in a variety of ways including using it with CALL SAY in the same program, storing the speech data on a mass-storage device, or viewing the actual speech data.

To assign the speech data for the word "HELLO" to the string variable R$, type

```
CALL SPGET("HELLO",R$)
```

Notice the use of parentheses, quotation marks, and the comma in the above example.

The computer searches for the code for HELLO and assigns that code to the string variable R$. When you want the computer to speak the word assigned to R$, use the string variable with CALL SAY. The format is

```
CALL SAY("",R$)
```

You must include the null string because the CALL SAY statement can utilize two kinds of string expressions. These string expressions must be listed in a specified order.

First, CALL SAY makes the computer speak words and phrases as discussed previously.

```
CALL SAY("HELLO")
```

or

```
10 A$="HELLO"
20 CALL SAY(A$)
```

These specified words or phrases are referred to as *word-strings.*

The second type of string expression is called a *direct-string.* A *direct-string* is the speech data assigned to a string variable through the CALL SPGET subprogram.

Using Word-Strings and Direct-Strings

If you are designating both kinds of string expressions in a CALL SAY statement, you must alternate between *word-strings* and *direct-strings.* The format is:

CALL SAY(*word-string*[,*direct-string, word-string. . .*]

The first specified word or phrase (*word-string*) is all that is required with CALL SAY. Any additional string expressions (*direct-strings* or *word-strings*) are optional.

If you use more than the first *word-string,* you must use the alternating sequence. Here's an example of the format:

```
CALL SPGET("AM",B$)
CALL SPGET("COMPUTER",C$)
CALL SAY("I",B$,"A",C$)
```

If you want to use two *word-strings* but no *direct-strings*, you must include the concatenation symbol (&) and the null string between the *word-strings*. Let's look again at lines 150, 160, and 170 in the "Guess the Number" program (page 20).

```
150 A$ = "O YES,I HAVE IT NOW"
160 B$ = "TRY TO THINK OF THE NUMBER."
170 CALL SAY(A$,"", B$)
```

Also, if you only use the code pattern assigned through CALL SPGET, you need to substitute the null string for the first *word-string*. For example,

```
10 CALL SPGET("HELLO",R$)
20 CALL SAY("",R$)
```

Using CALL SPGET in a Program

The most straightforward method of including speech in a TI BASIC program is with CALL SAY and a *word-string*. For example,

```
CALL SAY("HELLO. I AM A COMPUTER.")
```

However, more natural sounding speech is produced if you first use CALL SPGET to assign the speech code for each word to a string variable, and then use the string variables with CALL SAY to instruct the computer to speak the phrase. To produce the same words as above, you'd type

```
10 CALL SPGET("HELLO.",A$)
20 CALL SPGET("I",B$)
30 CALL SPGET("AM",C$)
40 CALL SPGET("A",D$)
50 CALL SPGET("COMPUTER",E$)
60 CALL SAY("",A$,"",B$,"",C$,"",D$,"",E$)
RUN
```

Listen for the slight difference in the timing of the phrase when you assign the speech data to variables with CALL SPGET and then use the variables with CALL SAY.

Appendix A

SPEECH SYNTHESIZER RESIDENT VOCABULARY

Note: Some of the words in the resident vocabulary have more than one pronunciation. In those cases, the word is listed twice with a "1" after the second listing. Each pronunciation is provided in parentheses after the word. When using one of these words, you must distinguish between them by including the "1" with the second listing if that is the pronunciation required.

-(NEGATIVE)
+(POSITIVE)
.(POINT)
0
1
2
3
4
5
6
7
8
9
A(a)
A1(ə)
ABOUT
AFTER
AGAIN
ALL
AM
AN
AND
ANSWER
ANY
ARE
AS
ASSUME
AT
B
BACK
BASE
BE
BETWEEN
BLACK
BLUE
BOTH
BOTTOM
BUT
BUY
BY
BYE
C
CAN
CASSETTE
CENTER
CHECK
CHOICE
CLEAR
COLOR
COME
COMES
COMMA
COMMAND
COMPLETE
COMPLETED
COMPUTER
CONNECTED
CONSOLE
CORRECT
COURSE
CYAN
D
DATA
DECIDE
DEVICE
DID
DIFFERENT
DISKETTE
DO
DOES
DOING
DONE
DOUBLE
DOWN
DRAW
DRAWING
E
EACH
EIGHT
EIGHTY
ELEVEN
ELSE
END
ENDS
ENTER
ERROR
EXACTLY
EYE
F
FIFTEEN
FIFTY
FIGURE
FIND
FINE
FINISH
FINISHED
FIRST
FIT
FIVE
FOR
FORTY
FOUR
FOURTEEN
FOURTH
FROM
FRONT
G
GAMES

GET
GETTING
GIVE
GIVES
GO
GOES
GOING
GOOD
GOOD WORK
GOODBYE
GOT
GRAY
GREEN
GUESS
H
HAD
HAND
HANDHELD UNIT
HAS
HAVE
HEAD
HEAR
HELLO
HELP
HERE
HIGHER
HIT
HOME
HOW
HUNDRED
HURRY
I
I WIN
IF
IN
INCH
INCHES
INSTRUCTION
INSTRUCTIONS
IS
IT
J
JOYSTICK
JUST
K
KEY
KEYBOARD
KNOW
L
LARGE
LARGER
LARGEST
LAST
LEARN
LEFT
LESS
LET
LIKE
LIKES
LINE
LOAD
LONG
LOOK
LOOKS
LOWER
M
MADE
MAGENTA
MAKE
ME
MEAN
MEMORY
MESSAGE
MESSAGES
MIDDLE
MIGHT
MODULE
MORE
MOST
MOVE
MUST
N
NAME
NEAR
NEED
NEGATIVE
NEXT
NICE TRY
NINE
NINETY
NO
NOT
NOW
NUMBER
O
OF
OFF
OH
ON
ONE
ONLY
OR
ORDER
OTHER
OUT
OVER
P
PART
PARTNER
PARTS
PERIOD
PLAY
PLAYS
PLEASE
POINT
POSITION
POSITIVE
PRESS
PRINT
PRINTER
PROBLEM
PROBLEMS
PROGRAM
PUT
PUTTING
Q
R
RANDOMLY
READ(rēd)
READ1(rĕd)
READY TO START
RECORDER
RED
REFER
REMEMBER
RETURN
REWIND
RIGHT
ROUND
S
SAID
SAVE
SAY
SAYS
SCREEN
SECOND
SEE
SEES
SET
SEVEN
SEVENTY
SHAPE
SHAPES
SHIFT
SHORT
SHORTER
SHOULD
SIDE
SIDES
SIX
SIXTY
SMALL
SMALLER
SMALLEST
SO
SOME
SORRY
SPACE
SPACES
SPELL
SQUARE
START
STEP
STOP
SUM
SUPPOSED
SUPPOSED TO
SURE
T
TAKE

TEEN
TELL
TEN
TEXAS INSTRUMENTS
THAN
THAT
THAT IS INCORRECT
THAT IS RIGHT
THE(thē)
THE1(thə)
THEIR
THEN
THERE
THESE
THEY
THING
THINGS
THINK
THIRD
THIRTEEN
THIRTY
THIS
THREE
THREW
THROUGH
TIME
TO
TOGETHER
TONE
TOO
TOP
TRY
TRY AGAIN
TURN
TWELVE
TWENTY
TWO
TYPE
U
UHOH
UNDER
UNDERSTAND
UNTIL
UP
UPPER
USE
V
VARY
VERY
W
WAIT
WANT
WANTS
WAY
WE
WEIGH
WEIGHT
WELL
WERE
WHAT
WHAT WAS THAT
WHEN
WHERE
WHICH
WHITE
WHO
WHY
WILL
WITH
WON
WORD
WORDS
WORK
WORKING
WRITE
X
Y
YELLOW
YES
YET
YOU
YOU WIN
YOUR
Z
ZERO

Appendix B

ERROR MESSAGES

I. Messages Displayed When Using Speech Editor

*OUT OF SPACE

1. All ten lines are filled with text.

*NO VOCABULARY PLUGGED IN

1. The Speech Synthesizer is not connected.

*FINISH THE PHRASE

1. The closing number sign(#) is omitted from a resident vocabulary phrase.

*CAN'T DO THAT

1. **SHIFT F** is used to delete a number sign(#).

Also, entered words that are not in the resident vocabulary are flashed at the bottom of the display.

II. Messages Displayed When Using CALL SAY or CALL SPGET

*ILLEGAL SPEECH FORMAT

1. Incorrect format is used in CALL SAY or CALL SPGET.

*SPEECH STRING TOO LONG

1. The length of a direct-string of speech data is greater than 252 characters.

*BAD NAME

1. The Speech Editor Command Module is not plugged into the console.

The computer says "UHOH" when words are entered that are not in the resident vocabulary.

Appendix C

CALL SAY SUBPROGRAM

CALL SAY (*word-string*[,*direct-string,word-string...*])

The CALL SAY subprogram activates the TI Speech Synthesizer from TI BASIC. Speech can be produced from specified words (*word-strings*) or through actual speech code patterns (*direct-strings*) generated by CALL SPGET.

The CALL SAY statement contains one or more parameters. The first and only required parameter is a *word-string*. All other odd-positioned parameters must also be *word-strings*. A *word-string* is a string expression (including string constants and variables) containing the words or phrases to be recited by the Speech Synthesizer. If a word or phrase is not contained in the Speech Synthesizer's vocabulary, the phrase UHOH is spoken.

The second and all other even positioned parameters must be *direct-strings*. A *direct-string* contains control information and speech code patterns that reproduce speech when processed by CALL SAY. A *direct-string* is generated by the CALL SPGET subprogram.

A comma must separate each parameter in CALL SAY. A *null string* (" ") is used to fill the position of any missing intermediate parameter in the format.

Appendix D

CALL SPGET SUBPROGRAM

CALL SPGET (*word-string,direct-string-variable*)

The CALL SPGET subprogram retrieves speech data from the Speech Synthesizer. The CALL SPGET subprogram can be used in Immediate Mode and in a TI BASIC program. The CALL SPGET statement always contains two parameters. The first is a string expression or *word-string* which specifies the word or phrase. The second is a string variable to which the speech code pattern for that word or phrase is to be assigned. The data stored in this variable is referred to as a *direct-string*.

If the word or phrase specified in CALL SPGET is not found in the Speech Synthesizer resident vocabulary, the code pattern for UHOH is stored in the string variable. If the word or phrase is in the vocabulary, the speech data is placed in the string variable preceded by three bytes (ASCII characters) of control information. The first byte is a command to the Speech Synthesizer. The next two bytes represent the number of bytes or characters of speech information following. The maximum number of bytes (characters) of speech data is 252. The total length of the direct string cannot exceed 255 characters.

APPENDIX E
DISPLAYING SPEECH DATA

You may have wondered what speech code patterns actually look like. You can use CALL SPGET to see speech code patterns on the screen. Type and RUN the following short program:

```
NEW
10 CALL SPGET("HELLO",R$)
20 PRINT "START"; R$; "STOP"
RUN
```

The words START and STOP are printed to show where the speech code begins and ends. As you can see, the output of this program by itself is interesting, but not very useful. Speech code patterns are made up of many characters that do not have a screen character representation. These data patterns can be translated into readable characters by using hexadecimal (HEX) representation.

The following program converts a string of direct speech code into HEX. Enter this program into your computer, and experiment with a variety of words from the resident vocabulary.

```
NEW
100 REM-HEX DUMP OF SPEECH DATA
110 CALL CLEAR
120 INPUT "TYPE WORD: ":WORD$
130 CALL SAY(WORD$)
140 CALL SPGET(WORD$,R$)
150 HEX$="0123456789ABCDEF"
160 L=LEN(R$)
170 PRINT "LENGTH="; L;"BYTES"::
180 FOR I=1 TO L
190 DEC=ASC(SEG$(R$,I,1))
200 HIGH=INT(DEC/16)
210 LOW=DEC-16*HIGH
220 HIGH=HIGH+1
230 LOW=LOW+1
```

```
240 PRINT SEG$[HEX$,HIGH,1 ];
250 PRINT SEG$[HEX$,LOW,1];
260 IF I/10<>INT[I/10] THEN 280
270 PRINT
280 NEXT I
290 PRINT : :
300 GO TO 120
```

Line 120 gives a prompt to type in a word selected from the resident vocabulary list. Then line 130 directs the computer to speak the word you typed. Line 140 tells the computer to retrieve the code pattern for the word you typed and assign the direct-string of speech data patterns to R$.

In line 160 the value of "L" tells you how long the pattern is. Line 190 picks a single character (byte) of speech code and expresses it as a decimal number between 0 and 225. Lines 200 and 210 return the HIGH and LOW order HEX digit of DEC. Lines 220 and 230 shift each digit by one to correspond with the proper symbol in HEX$.

Lines 240-290 direct the computer to print the speech data in HEX. The computer skips to the next line after every 10 bytes or 20 HEX characters. Line 300 sends the program back to line 120 for a new word input.

RUN the program and see the code patterns expressed in HEXADECIMAL. (Press **SHIFT C** to stop the program.)

Notice that the first two digits (60 HEX) are the same on all words. They represent a command to the Speech Synthesizer. The next four digits indicate the number of bytes of speech code to follow. The length of the speech code cannot exceed 252 bytes.

SERVICE INFORMATION

Caring for the Command Module

Command Modules are durable devices, but they should be handled with the same care you would give any other piece of electronic equipment. Keep the module and its recessed contacts clean and dry.

> CAUTION:
> The contents of a Command Module can be damaged by static electricity discharges.

Static electricity build-ups are more likely to occur when the natural humidity of the air is low (during winter, for example, or in very dry climates). To avoid possible damage of the Command Module, just touch any metal object (a doorknob, a desklamp, etc.) before handling the module. Always use this method to ensure that both you and the module are free of static electricity before you install the module on the console. If static electricity is a problem in your area, you may want to purchase a special carpet treatment that reduces static electricity build-up. These commercial preparations are usually available from local hardware and office supply stores.

In Case of Difficulty

If the module activities do not appear to be operating properly return to the master title screen by pressing **SHIFT Q**. Withdraw the module, align it with the module opening, and reinsert it carefully. Then press any key to make the master selection list appear. Repeat the selection process. (*Note*: In some instances, it may be necessary to turn the computer off, wait several seconds, and then turn it on again.)

If the module is accidentally removed from the slot while the module contents are being used, the computer may behave erratically. To restore the computer to normal operation, turn the computer console off, wait a few seconds, reinsert the module, and turn it on again.

If you have any difficulty with your computer or the Speech Editor Command Module, please contact the dealer from whom you purchased the unit and/or module for service directions.

Additional information concerning use and service can be found in your *User's Reference Guide* and in the *Solid State Speech*™ Synthesizer owner's manual.

A computer that speaks without tapes or other traditional recording devices! Through the advanced technology of *Solid State Speech*™ synthesis, Texas Instruments makes this futuristic dream a reality of today. With the Texas Instruments *Solid State Speech*™ Synthesizer attached and the Speech Editor Command Module plugged into the console, you can easily add speech to the capabilities of your Home Computer.

Speech Editor module features provide two methods of utilizing the vocabulary of the Speech Synthesizer:

- Select Speech Editor to listen to and experiment with computer-generated speech. No programming skills are required.
- Select TI Basic to include speech in your TI Basic programs.

Adds 6K bytes of active memory with stored program to your TI Home Computer.

TEXAS INSTRUMENTS
INCORPORATED

Printed in U.S.A.