3

Speech Synthesizer

In this chapter we will begin to apply what we learned in the last chapter with the practical application of synthesizing speech.

SPEECH SYNTHESIS

Speech synthesizers (or processors) appear in two main formats. One approach (format #1) uses digitally recorded speech stored in a ROM chip. The second approach (format #2) uses phonemes of English to construct words and sentences (a phoneme is a speech sound).

The main advantage in Format #1 is a excellent speech reproduction and fidelity. Its main disadvantage is a limited vocabulary of English that's been preprogrammed into the chip.

Format #2's strength is an unlimited user defined vocabulary. Its disadvantage is that the speech fidelity isn't as good as with the preprogrammed speech ROM. Even so, the speech fidelity of format #2 is quite acceptable in all but the most critical circumstances. We are taking this second approach to speech synthesis.

The speech synthesizer we will build, plugs into, and is powered by the user port. The cost is less than $25.00, and for that price it includes its own audio amplifier, filter, volume control, and speaker. Since it has an unlimited vocabulary, you can program any word you desire. You have the option to either modify existing programs to include speech, or, of course, to write new programs with speech.

THE SPEECH CHIP

General Instruments Company manufactures the 28-pin speech synthesizer chip (SPO256-A12) that is distributed by Radio Shack. This chip can generate 59 allophones (speech sounds) and five pauses (no sound) of various lengths (see allophones, Table 3-1). By adding (concatenating) allophones together, you can construct words and sentences. This may sound rather difficult at this point but it is not, the program does most of the work.

Table 3-1. Allophones.

Decimal Address	Allophone	Sample Word	Duration	Decimal Address	Allophone	Sample Word	Duration
0	PA1	Pause	10MS	32	AW	Out	370MS
1	PA2	Pause	30MS	33	DD2	Don't	160MS
2	PA3	Pause	50MS	34	GG3	Pig	140MS
3	PA4	Pause	100MS	35	VV	Venom	190MS
4	PA5	Pause	200MS	36	GG1	Gotten	80MS
5	OY	Toy	420MS	37	SH	Sharp	160MS
6	AY	Buy	260MS	38	ZH	Azure	190MS
7	EH	End	70MS	39	RR2	Train	120MS
8	KK3	Cat	120MS	40	FF	Forward	150MS
9	PP	Power	210MS	41	KK2	Sky	190MS
10	JH	Judge	140MS	42	KK1	Came	160MS
11	NN1	Pin	140MS	43	ZZ	Zolu	210MS
12	IH	Sit	70MS	44	NG	Anchor	220MS
13	TT2	To	140MS	45	LL	Lamb	110MS
14	RR1	Pural	170MS	46	WW	Wood	180MS
15	AX	Succeed	70MS	47	XR	Pair	360MS
16	MM	My	180MS	48	WH	Whine	200MS
17	TT1	Tart	100MS	49	YY1	Yes	130MS
18	DH1	They	290MS	50	CH	Chump	190MS
19	IY	Tee	250MS	51	ER1	Tire	160MS
20	EY	Beige	280MS	52	ER2	Tire	300MS
21	DD1	Should	70MS	53	OW	Beau	240MS
22	UW1	To	100MS	54	DH2	They	240MS
23	AO	Aught	100MS	55	SS	Best	90MS
24	AA	Home	100MS	56	NN2	Not	190MS
25	YY2	Yes	180MS	57	HH2	Noe	180MS
26	AE	Pat	120MS	58	OR	Pore	330MS
27	HH1	Him	130MS	59	AR	Arm	290MS
28	BB1	Boy	80MS	60	YR	Clear	350MS
29	TH	They	180MS	61	GG2	Guide	40MS
30	UH	Book	100MS	62	EL	Paddle	190MS
31	UW2	Food	260MS	63	BB2	Boy	50MS

A LITTLE ON LINGUISTICS

An allophone is the computer equivalent to English phonemes (speech sounds). There are two main points you should keep in mind when you are programming new words. First, in English there isn't a one to one correspondence between letters and sounds. This point is amply demonstrated by the younger members of our society who are learning to read and write. They are likely to spell *cat* as *Kat* and phone as *fone*, imitating in writing the way the words are pronounced. This is a very interesting point, because in order to program words to sound correct, you must spell the words phonetically. More about this later, let's continue on to the second point. Placement of a speech sound in a word can change the pronunciation. As an example, take a look at the two *D* letters in the word *d*epend. The *D*'s are pronounced differently. If we were to program this word using our table of allophones, the allophone DD2 would sound correct in the first position (*D*epend) and the allophone DD1 sounds correct in the second position (depend). We will return to programming technique later on. A booklet with more information on linguistics, allophones, and usage is included with the speech synthesizer chip.

CIRCUIT CONSTRUCTION

The circuit is comprised of two sections (see Fig. 3-1), separated by a dotted line. Section A on the left is the basic circuit. Section B contains the amplifier, low-pass filter, volume control and speaker added to the basic circuit.

The two sections A and B make up a stand-alone unit that only requires power and control signals from the user port to function. In contrast to section A of the circuit, which requires the use of the *Sound Interface Device* (SID) chip and a monitor or TV speaker.

By utilizing the SID chip in the C-64 or C-128 computer, you can eliminate section B, the audio amplifier, filter, volume control, and speaker (see Fig. 3-2), reducing the amount of parts required by more than half, simplifying the circuit, and saving a couple dollars. However, if you're using a Vic-20 computer you will have to build the entire circuit (see Fig. 3-3).

The C-64 and C-128 can use either section A, or the entire circuit. To use just section A, we eliminate section B and take the output of the circuit (at pin 24) and input the signal to the SID chip. We accomplish this with a wire to the "audio in" pin of the composite video connector (see pin 5 of the C-64 and C-128). Pin 24 is the digital output of the speech synthesizer chip. You can purchase the correct din plug for your computer or use a short wire pushed into the correct pin socket connected by a jumper wire to pin 24. You can use our experimenters breadboard for this circuit. See Fig. 3-2. Plug in your components as diagrammed and you're ready to begin programming.

For the Vic-20 I constructed the entire circuit on a modified experimenters card (see Fig. 3-3). The card is modified by cutting the end terminals on both sides leaving the center 12 positions. Use a 12/24 card connector and solder the lugs on the connector to the fingers on the board. If a 12/24 card connector isn't readily available you can modify a 22-position card connector into a 12-position connector by cutting 10 positions off as I have done. Only 10 connections are needed for this project. I did, however, solder all the connections to improve the mechanical strength of the unit. Pin 24 is connected into the B section circuit through a low-pass audio filter to a 10 k volume control pot. User either a trimmer pot that you can set once and forget about or eliminate the pot completely. The volume of sound with the pot removed isn't so great as to be objectionable. You'll probably use the speech synthesizer with the pot fully closed anyway.

Power for the entire unit is provided from the top side of the user port. The bottom side (Port B) accesses and controls the speech processor. If in wiring you get confused tracing the leads from the user port to the speech chip, I suggest holding the card connector (or experimenters board) to the diagram of the user port. This will help match where each wire connects. The diagram of the user port can be used this way because it shows how the user port appears when looking directly into it from the back. When completed the card connector plugs into the user port.

The manufacturer of the speech chip recommends using a 3.12 MHz crystal at pins 27 and 28. I recommend using a 3.57 MHz colorburst crystal instead. The reason is cost and availability. The 3.57 MHz colorburst crystal is approximately ¼ the cost of the 3.12 MHz crystal, and is more readily available. This change will increase the timbre of the speech slightly, but has no other effect on circuit operation.

THE PROGRAM

Type in the program (see Fig. 3-4). Assign a value to "PB" in line 60 depending on which computer you are using.

For the Vic-20: `PB=37136`
For the Com-64: `PB=56577`
For the C-128 : `PB=56577`

If you are using a C-64 or C-128 with section A only, type in this additional line:

```
55 S=54272:FORL=0to24:POKES+L,0:NEXT:POKES+24,15
```

When the program is run, the computer should say hello. Adjust the trimmer pot if you built the entire circuit, and have included the pot in the circuit.

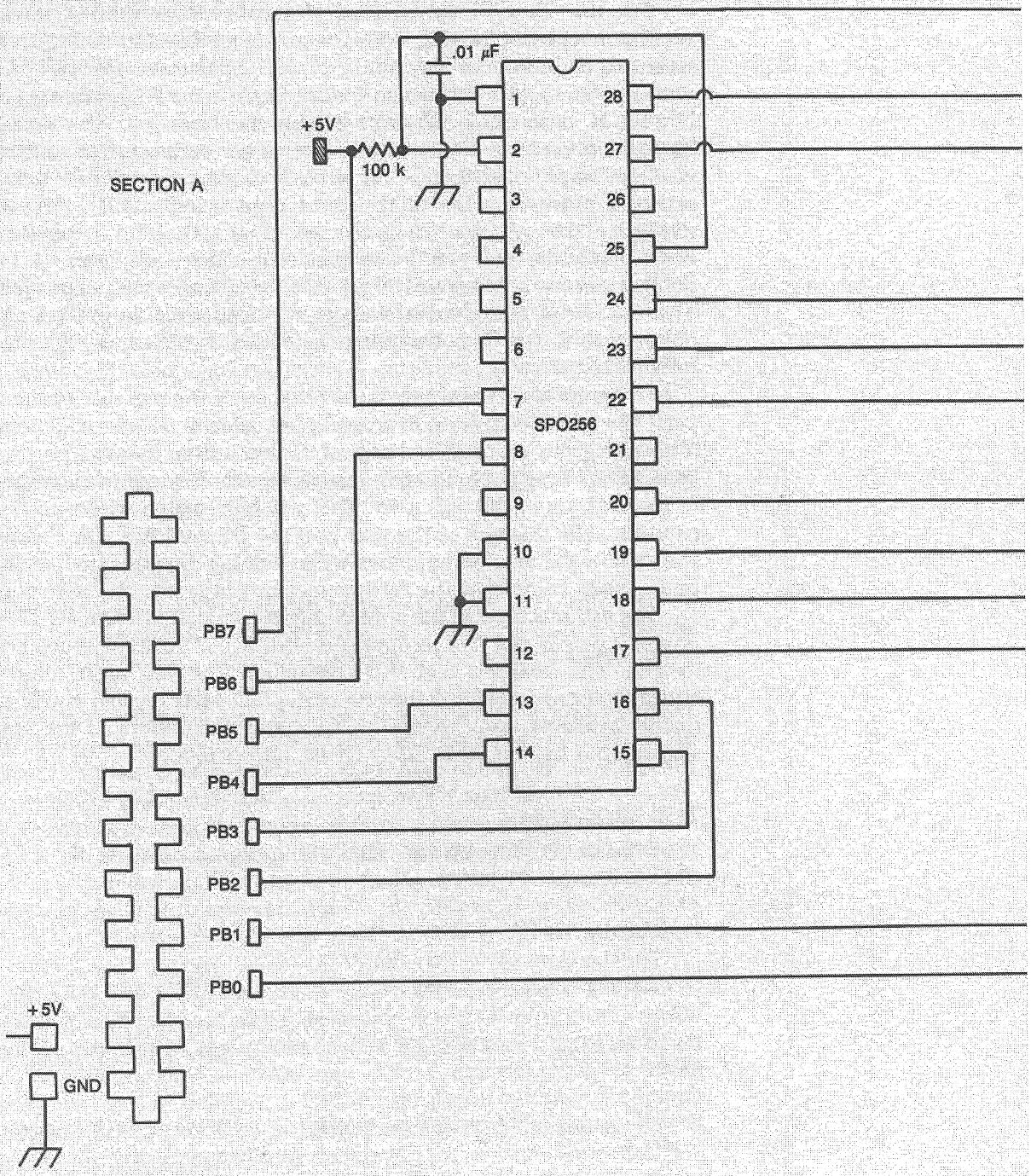
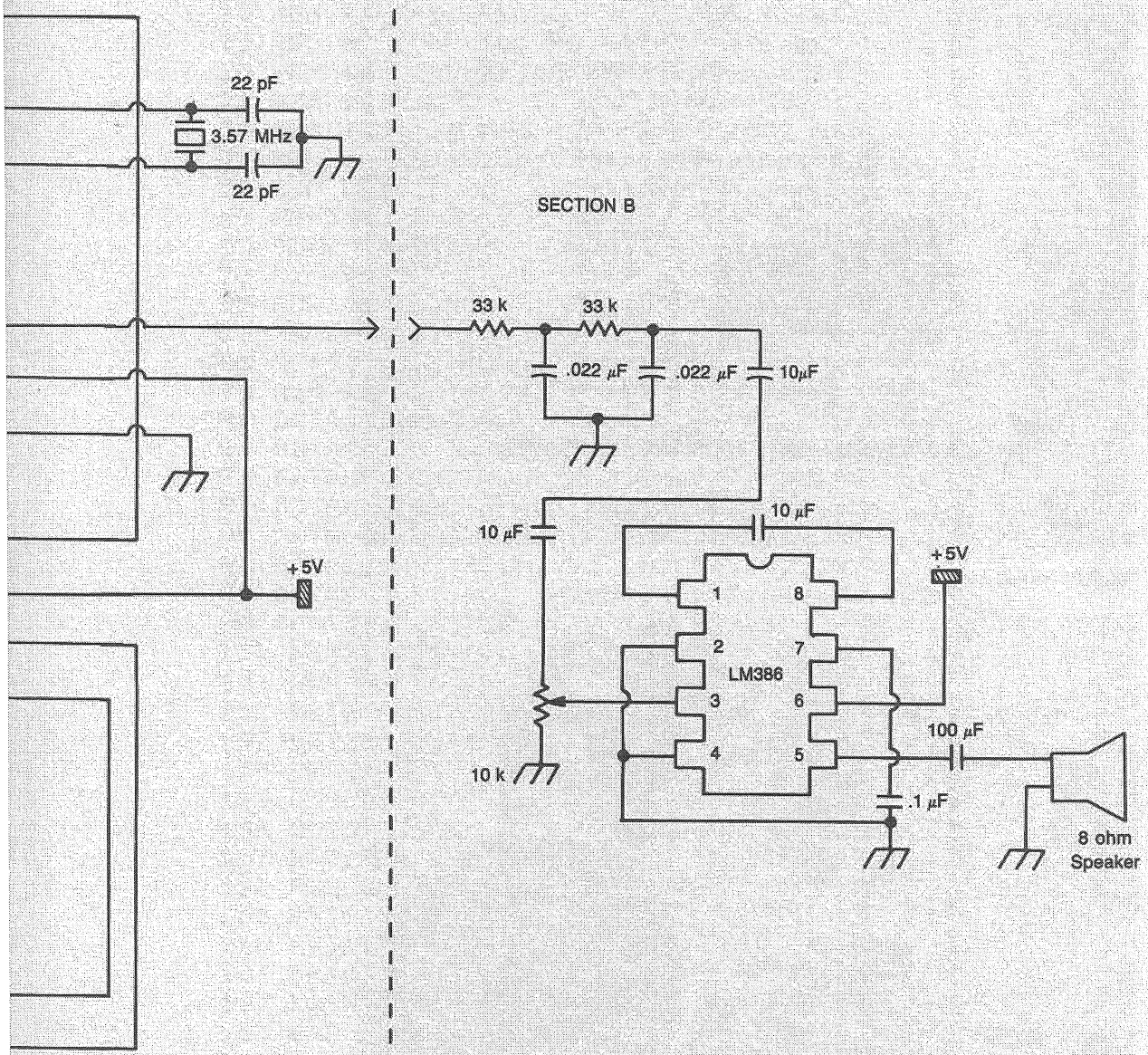
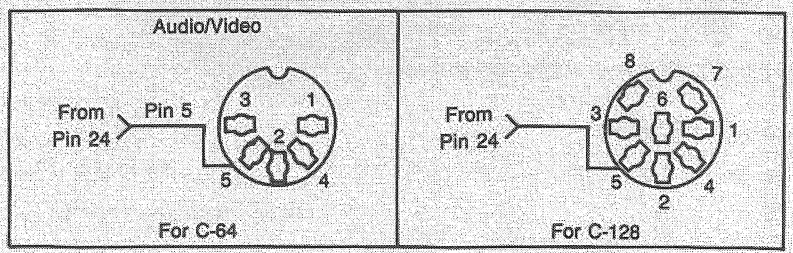


Fig. 3-1. Speech synthesizer sections A and B.


22 pF
3.57 MHz
22 pF
SECTION B
33 k
33 k
.022 μF
.022 μF
10μF
10 μF
10 μF
+5V
+5V
1
8
2
7
LM386
3
6
4
5
100 μF
10 k
.1 μF
8 ohm
Speaker


Audio/Video
From
Pin 24
Pin 5
3
1
2
5
4
For C-64
8
7
From
Pin 24
3
6
1
5
4
2
For C-128

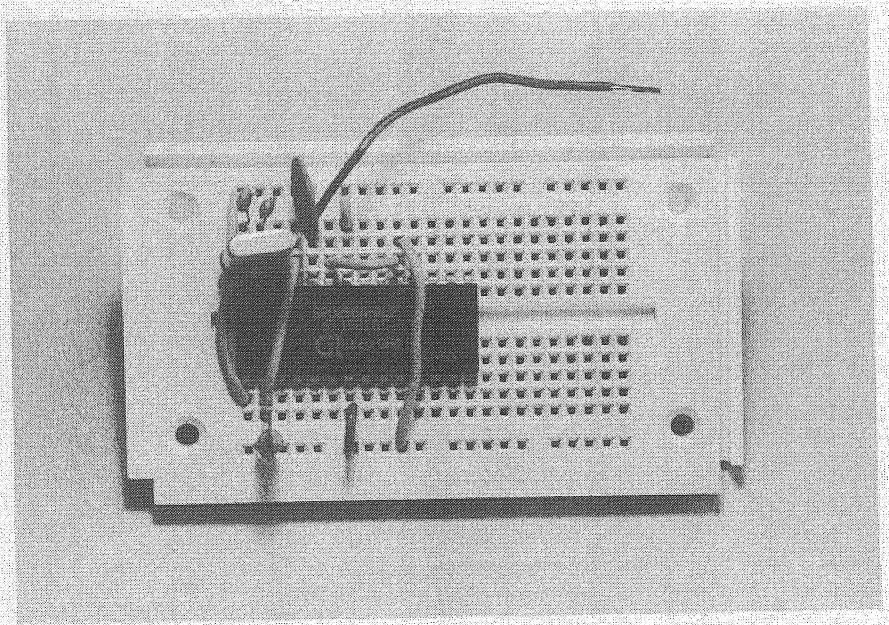


Fig. 3-2. Section A.

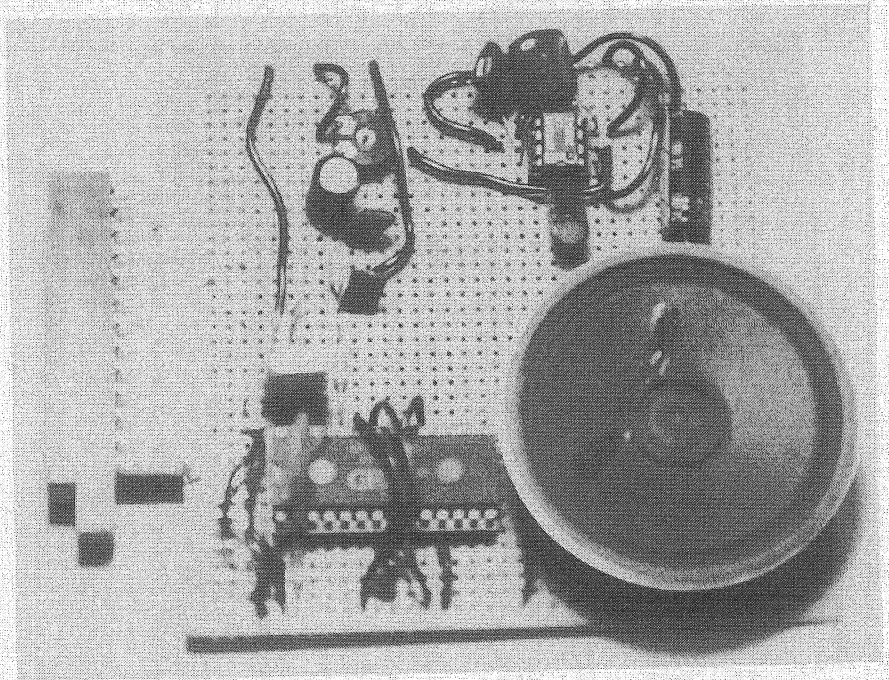


Fig. 3-3. Sections A and B on PC board.

```
10 REM ****** SPEECH PROCESSOR *****
15 REM BY JOHN IOVINE DATED 4-16-86
20 REM ******************************
25 REM ** VIC-20          PB=37136 **
30 REM ** C-64 & C-128    PB=56577 **
35 REM ******************************
50 REM SET UP DDR AND TABLE
60 PB=56577
65 POKE PB+2,191
70 DIMM$(63)
75 FORI=0TO63
80 READM$(I)
85 NEXTI
90 REM *** DATA TABLE ***
91 DATAPA1,PA2,PA3,PA4,PA5,OY,AY,EH,KK3
92 DATAPP,JH,NN1,IH,TT2,RR1,AX,MM,TT1
93 DATADH1,IY,EY,DD1,UW1,AO,AA,YY2,AE
94 DATAHH1,BB1,TH,UH,UW2,AW,DD2,GG3,VV
95 DATAGG1,SH,ZH,RR2,FF,KK2,KK1,ZZ,NG
96 DATALL,WW,XR,WH,YY1,CH,ER1,ER2,OW
97 DATADH2,SS,NN2,HH2,OR,AR,YR,GG2,EL,BB2
99 REM *** END OF DATA TABLE ***
150 REM *** SPEECH MODULE ***
151 REM HELLO
152 DATA HH1,EH,LL,AX,OW,PA1
153 LETG=G+1:IFG=7THEN157
154 READ A$
155 GOSUB10000
156 GOTO153:REM RETURN TO COUNTING LINE
157 G=0:REM RESET G
158 REM CONTINUE MAIN PROGRAM
159 PRINT"{CLR}"
160 PRINT"PROGRAM WOULD CONTINUE HERE"
161 PRINT"TYPE RUN THEN PRESS RETURN"
162 PRINT"TO HEAR AGAIN"
163 END
10000 FOR I=0TO63
10005 IF M$(I)=A$THEN10050
10010 NEXT I
10015 PRINT"ERROR IN DATA STATEMENT":POKE56577,0:END
10050 IF PEEK(PB)=128THEN10050
10055 POKE PB,I
10060 POKE PB,128
10065 RETURN
```

Fig. 3-4. Speech synthesizer program.

If the computer fails to speak, you have either a typing error in the program or a wiring error in the circuit. Check over the program to see that you entered it correctly. Recheck your wiring. If everything checks out, verify circuit operation by checking for clock pulses at pins 27, 28, and 24. If you show pulses the problem is in the audio section.

Although it isn't necessary to understand how the program operates to use it. Here is a brief description:

STEP 1 Lines 60 to 100:	Sets up Data Direction Register and allophone table
STEP 2 Lines 150 to 157:	Speech Module—reads speech in program
STEP 3 Lines 10000 to 10065:	Subroutine sends instructions to speech chip and returns

Until you gain some experience and feel comfortable designing your own speech program. STEP 2, lines 150 to 157 is the model to use to program speech in your basic programs. We will do a line by line analysis and an example will ensure a good understanding of the procedure.

Line 151	Is a REM statement labeling the word or phrase contained in the following data statements. This is useful in the event you wish to correct, change or eliminate words. By clear labeling you can locate the word quickly.
Line 152	Data statement; containing allophones for the word hello.
Line 153	Counting line; enabling the computer to read the proper number of allophones in the data statement then jump to the end of the speech module upon completion. G=(number of allophones)+1. Therefore if our word uses 6 allophones then G=7. If a sentence uses 31 allophones, then G=32.
Line 154	Reads allophones in data statement.
Line 155	Takes data read; jumps to subroutine line 10000. There computer compares to table, decodes and provides necessary electrical pulses to the speech chip; returns.
Line 156	Return program to counting; incrementing G; reading the next allophone. Process is repeated until G equals its assigned value.
Line 157	Line number called in line 153, <then 157>—Resets G to 0, enabling G to be used again in other modules of the program.

Example

The booklet provided with the chip has a dictionary with over 200 words, with their proper allophone data. These words can be put into your program at once. Some of the words included are:

- numbers 0 to 1 million
- days of the week

- months
- letters
- common words

For our example we will construct a sentence concatenating 4 words and entering it in our program.

```
200 Rem See You Next Tuesday
201 Data SS,SS,IY,PA1
202 Data YY1,UW1,PA1
203 Data NN1,EH,KK1,SS,TT2,PA1
204 Data PA1,TT2,UW2,ZZ,PA2,DD2,EY,PA1
```

The REM statement describes what is contained in the following data statements. You can use or start with any line number you'd like, just remember to be consistent.

```
205 Let G=G+1: If G=22 Then 225
```

Count the allophones in the above data statements. You should count 21 allophones. Since G=(# of allophones)+1 therefore G=22. Note line number 225 call out <Then 225> in this line, it marks the end of the speech module. You can easily predict this number since it is always 4 lines down from this line.

```
210 Read A$
```

Reads allophone in data statement.

```
215 GoSub 10000
```

Program goes to subroutine at line 10000.

```
220 GOTO 205
```

When program returns from subroutine this line returns program to line 205 the counting line. G is incremented, next allophone is read, until G equals its assigned value.

```
225 G=0
```

This is the line called when G reaches its assigned value. This line resets G to 0 so it can be used again for other speech modules.

The allophone table correlates each allophone with its approximate sound. This table is essential for programming words that aren't in the provided dictionary. Please be aware that there are a few typographical

errors in the dictionary. Such as in the following words:

```
Hello—HH,EH,LL,AX,OW,AW,ER1

AND

Computer—KK1,AX,MM,PP1,YY1,UW1,TT2,ER
```

In the word hello; the first allophone hh doesn't exist in the table. You should use HH1, or the word will sound like *ello*. In the word computer; the last allophone *er* doesn't exist. You must use ER1 or the word will sound like *compute* not computer.

If you use a word from the dictionary that doesn't sound correct, first check the allophones to see if there is a typo. Always end a word or phrase with one of the pauses PA1 to PA5. This is necessary to stop the computer from enunciating the last allophone.

BASIC CRUNCH

BASIC can run the speech processor, but it is a little slow. One of the easiest ways to bring BASIC up to speed is to use multiple statement lines and eliminate all unnecessary REMs. Effective programming has been known to help also. Experiment by crunching the program as much as possible. I would do this one step at a time or you stand a good chance of crashing.

Machine language is very quick and ideal to use with the circuit. If you're a machine language programmer here is a great opportunity to test your mettle. I advise running an ML wedge and implementing a new BASIC command such as "say" or "speak" that would completely eliminate all BASIC programming.

A good in between program could be implemented using a binary jump search routine for the data comparison after the data read, as this is the most time consuming portion of our program.

Conclusion

You now have the tools you need to program speech. To utilize the basic program into an existing program or to help organize a new program, think of the program as existing in three distinct modules; the data table, speech module, and speech routine. Set up the data table before it will be used by the program. Put the subroutine for speech near the end of the program. The speech modules are placed anywhere in between, where you want the computer to speak.

Examine the speech routine section of the program, with the knowledge and information given in the last chapter you should be able to figure out how this program is operating. If you have any problem you may want to place your LED interface at the user port and run the speech synthesizer program to observe the controlling bits, it'll definitely help.

What has been written in this installment is the bare essentials. Feel free to experiment and develop your own program.

For those of you who are more adventurous, Radio Shack sells a companion chip that's an ASCII text to speech converter. Practical applications for the chip are a text reader or verbal modem.

Parts List

Quantity	Item/Description	Part Number		Cost
1	SP0256-AL2	RS# 276-1784		$12.95
1	LM386	RS# 276-1731		1.09
1	100k ¼ watt	RS# 271-1311		.39
2	33k ¼ watt	RS# 271-1341		.39
3	.1 μF Cap	RS# 272-135	2 @	.49
2	.022 μF Cap	RS# 272-1066		4.69
2	22pF Cap			
2	100 μF Cap	RS# 272-1016	2 @	.79
1	10k Trimmer Pot	RS# 271-335		.49
1	8 ohm speaker	RS# 40-245		1.89
1	1 μF Cap	RS# 272-996		.79
1	10 μF Cap	RS# 272-999		.99
1	Experimenters Board	RS# 276-168		1.95
1	Card Connector 12/24 or Module	RS# 276-1551		2.99
1	3.57 MHz Crystal	RS# 272-1310		1.69

All parts are available from Radio Shack.