# TEXT TO SPEECH
# USER'S GUIDE

Texas Instruments Inc.
Personal Computer Division
[redacted]


October 1980

# Introduction

This document provides the information a BASIC user needs to use the text-to-speech system under BASIC. It provides all the basic knowledge required to take full advantage of the options available in the text-to-speech (TTS) system.

This document is <u>not</u> intended for GPL programmers. They are referred to the Text to Speech Design Specification, which contains additional information on the internal operations of the TTS system and calling procedures in GPL.

The TTS system has been implemented as two independent devices, called SPEECH and ALPHON. The SPEECH device is the standard TTS output device. Standard english text can be printed to this device.

The ALPHON device has been added to allow for direct allophone access. The output to this device consists of strings of allophone data. For implementation reasons, the data format used for this device is INTERNAL.

Unlike the SPEECH device, the ALPHON device can also be used as input device for previously spoken speech. This speech may have been output either through the SPEECH device or through the ALPHON device itself. The allophone string given is always the <u>last</u> phrase spoken.

The TTS system can be subdivided into 2 distinct functional blocks, as shown in Figure 1.

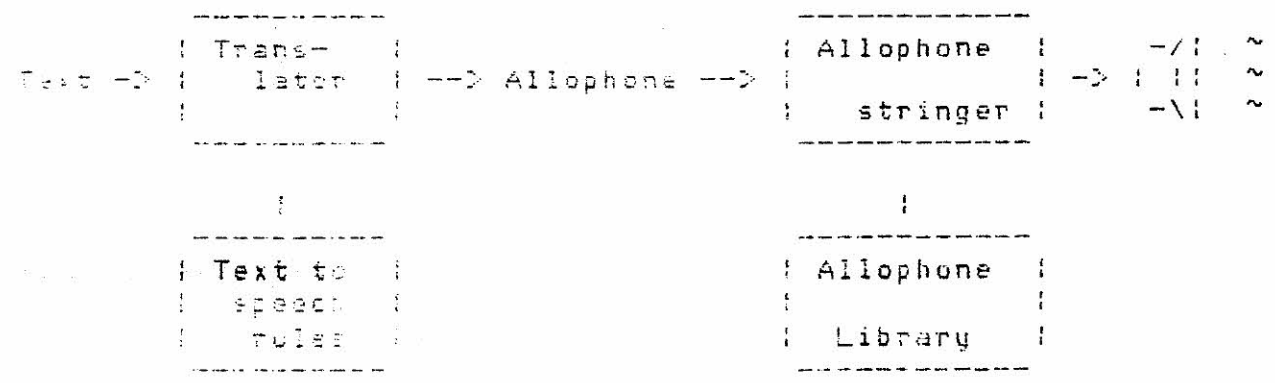


Figure 1 TTS Block Diagram

The first block is the text to allophone translator. This block is the main component of the TTS system. It uses speech rules, stored in a GROM rules library, to translate english text into it's allophonic equivalent. This also includes symbols for sentence contouring.

The second block converts allophone strings, which may include pitch modifiers, to LPC (Linear Predictive Coding) strings, that are directly output to the 0285 speech chip. Note that this means that the actual LPC data is never completely in memory, in order to save space.

The entire TTS is mainly optimized for speed. This means that the actual text translator and the allophone stringer have been completely programmed in 9900 code. For the allophone stringer this is a necessity, since GPL is not fast enough to support real time allophone stringing. The translator could be, and in fact at first has been, implemented in GPL code. Tests with this arrangement have shown however, that the translation process was to slow for pratical purposes. After recoding in 9900 assembly code, the translator showed a dramatic improvement in performance. It is virtually operating real-time now.

# SECTION 1

## Setting up the hardware

In order for the TTS system to function properly, a standard speech unit has to be connected to the TI 99/4. Instead of plugging in the Speech Editor module, the TTS module (or the TTS EGROM box) is plugged in to the cartridge port. If you are using an EGROM box, please don't forget to hook up the power to it.

After everything has been powered up, select BASIC and type the following statement :

```
OPEN #1:"SPEECH",OUTPUT
```

If the computer didn't protest against this, proceed to the next section. If somehow the computer didn't like it, chances are that you got the following message :

```
I/O ERROR 00
```

If this is the case, make sure that youve got :

A the cartridge or EGROM box hooked up properly.

B if youre using an EGROM box, the power connected to it.

If everything seems to be o.k., but BASIC still refuses to accept the OPEN statement, please contact me at (806)-741-2477.

# SECTION 2

## Using Text to Speech from BASIC

This section will provide a brief introduction in how to use the TTS system. At the end of the first paragraph you should be able to use TTS from normal BASIC programs. The remaining paragraphs explain how to use the more refined parts of TTS like phrase contouring and pitch control.

At the end of this section you will find some simple examples on how to incorporate TTS in a program.

### 2.1 The SPEECH device

The TTS system is implemented as a device under the TI 99/4 File Management System. This means that it is being accessed in exactly the same way as any other device. The limitations of the SPEECH device are simular to the limitations found in for instance a thermal printer, i.e. you can only output to the device, and the format used is the readable version (DISPLAY). More formal, the attributes (or parameters) given when the SPEECH device is opened, are :

DISPLAY - only human readable format

OUTPUT - were working with a write-only device

SEQUENTIAL- data is being spoken in the same sequence it is being output

VARIABLE - just accept this; there's no easy explanation for it

The easiest way to open the SPEECH device, which is recommended, is the following :

```
OPEN #1:"SPEECH",OUTPUT
```

Although the maximum record length can be selected by the user, it is recommended that you only modify the default value if your program is short on space. In that case you can add the ",VARIABLE xx" clause to the OPEN statement, with xx being a value between 1 and 255. This will however, limit the number of characters accepted by the device!

After the SPEECH device has been successfully opened for output, you can start printing to it. If you couldn't open the device, please refer to the last pragraph in this section for

more details on errors.

Getting the unit to actually speak, is now a fairly simple operation. Just type for instance :

```
PRINT #1:"HELLO THERE"
```

The speech unit should now give you its first greeting.

By now you will have noticed that the TTS system is perfectly capable of pronouncing strings, but how about numerics. Just type the following :

```
PRINT #1:4*ATN(1)
```

and the speech unit should give you the value of PI verbally. Notice that the period in PI is actually pronounced as "point". Some more special symbols that are being pronounced are given further on in this section.

This is about all the information you need to use the TTS system for normal operation. It is recommended that you play around with the TTS system some to get a feeling for the way it pronounces words, and how words sometimes have to be mis-pronounced to get the correct american pronounciation.

WARNING

> The current speech unit is not designed to handle speech strings containing only pauses. Typically, if you feed the TTS system an empty string, it will garbage the next phrase spoken. Don't be afraid of this, it doesn't damage the speech unit, nor does it have any permanent effects on the TTS system. The next output to the speech unit will sound normal again.

## 2.2 Pitch control

After the SPEECH device has been opened for access, standard english text can be output to the SPEECH device. To allow the user to set the default level for pitch and slope, a special record can be output, containing the following data :

```
//xxbyyy
```

in which xx is the pitch period, and yyy is the slope level indication. "b" is a single space character.

The pitch period indicator has to be in the range 0 through 63. The slope level indicator can be any value in the range 0

through 255. It actually indicates 32 times the slope increment used by the contouring algorithms. In general is should be selected in such a way that it is approximately 10% of the pitch period selected. For PITCH=43, the slope should be approximately 4 * 32 = 128 (These are the actual default values used by the SPEECH device). The multiplication factor of 32 has been selected to allow for fractional increments.

One restriction to the slope indicator is, that it should not exceed the following limits :

$$yyy < [\ xx - 1\ ] * 16$$
$$yyy < [\ 63 - xx\ ] * 16$$

The reason for this is that if the slope exceeds these limits, the contouring algorithms would cause the pitch parameter to cross either the high or the low end of the 0 - 63 range. The current SPEECH device automatically limits the slope parameter to a save value.

If the pitch period is selected to be 0, the speech will be unvoiced, i.e. it will sound like a whisper. Any non-zero value has to be within 1 (high pitched voice) and 63 (low pitched voice).

## 2.3 Special characters

The SPEECH device recognizes a number of special characters, which are being treated in a special way. These special characters can be subdivided into five major groups :

1. Alphabetical characters
2. Numerical characters
3. Pause and break characters
4. Inflection symbols
5. Special symbols

The following paragraphs will deal in detail with these five groups.

### 2.3.1 Alphabetical characters.

If the TTS system recognizes a single alphabetic character with no alphabetic neighbour, the standard pronounciation as a single character will be chosen. As an example, the following line

```
PRINT #1,"A B C"
```

would be pronounced as "AY BEE SEA".

### 2.3.2 Numerical characters.

The numerical character group consists of the numerical character "0" through "9", and the special symbols ",", "." , " " and "-". The characters "0" through "9" are always pronounced in the usual way. The special symbols however, are only pronounced if they directly preceed a numerical character. In every other case, they will be ignored, or treated as characters from one of the other groups, if applicable.

### 2.3.3 Pause and break characters.

This group consists of the characters ".b", ",b", "!", "?", ":" and ";". Note that the period and comma characters have to be followed by a space (here represented by a "b").

These codes generate pauses in a phrase, the length of which depends on the symbol used. The ",b" causes a short pause, all others a long pause. The shortpause is approximately .1 second; the long pause .45 seconds.

In addition to generating pauses, these characters also affect the inflection contour of the phrase if a primary stresspoint (see next paragraph) has been indicated. The ".b" and "?" both specify a rising contour; all other codes specify a falling contour. For more information please refer to the next paragraph and appendix A.

### 2.3.4 Inflection symbols.

This group consists of two main symbols, the "^" and the "_" symbols, and one additional symbol, the ">" symbol. The first two symbols are the actual inflection symbols, whereas the last symbol can be used to shift a stresspoints within a word.

Since every non-alphabetic character automatically causes a word-break, the inflection within a word has to be indicated before the word. The shift operator can be used the stresspoint to the desired vowel within the word.

As an example, please try the following

```
PRINT #1:"^SUPERB"
PRINT #1:"^>SUPERB"
PRINT #1:"_WHAT ^TIME IS _IT"
```

In these examples, note that the "^" symbol gives a primary stress point, of which there can only be one per line. The "_" symbol indicates a secondary stress point. The number of secondary stress point is basically unlimited.

If more than one primary stress point is used within the same phrase (i.e. not separated by break characters), the subsequent primary stress points will be demoted to secundary stress points.

For more information on the actual contouring algorithms, please refer to appendix A.

### 2.3.5 Special Symbols.

This group contain the symbols "@", "$", "%", "&", "*", "(", ")", "=" and "/". These symbols are always pronounced in the way indicated in Figure 2-1.

| Symbol | Pronounced as |
|---|---|
| @ | at |
| $ | dollar |
| % | percent |
| & | and |
| * | asterisk |
| ( | open |
| ) | close |
| = | equals |
| / | slash |

Figure 2-1 Special Symbol Pronounciation

## 2.4 DSR Error Codes

I/O errors detected by the TTS software are always indicated by BASIC in the following format :

```
* I/O ERROR xy [IN lll]
```

The digits "xy" indicate the type of error that has occurred. The first digit (x) indicates the I/O call at which the error occurred. The following I/O routine codes can be given :

0 error in OPEN routine

1 error in CLOSE routine

2 error in READ routine

3 error in WRITE routine

The other I/O codes are not supported by either SPEECH or ALPHON DSR. In addition the SPEECH DSR does not support the READ routine.

The second digit (y) indicates the type of I/O error that has occurred. Out of the 8 different possible codes, only the following are supported by the TTS DSRs :

0 The specified device could not be found.

2 BAD OPEN ATTRIBUTE - one or more OPEN attributes were illegal, or didnt match the devices characteristics.

3 ILLEGAL OPERATION - indicates that an unsupported I/O routine has been called (like for instance RESTORE).

4 OUT OF SPACE - the internal buffers werent big enough to hold the allophone string generated by the text to speech translation, or the generated string was bigger than 254 bytes, and a READ operation was attempted on the ALPHON device.

6 DEVICE ERROR - the TTS devices couldn't find their internal buffers. Usually indicates either hardware error, or a program that removes the internal buffers to obtain more VDP space.

# SECTION 3

# Direct Allophone Access

The TTS software also provides for direct access to the allophone part of the system, for those programmers wanting to experiment with new sounds, special words etc.

This direct allophone access is provided through a second device, called "ALPHON". Although this device can be used totally seperate from the "SPEECH" device, there are some cross links between the two devices. All the details required to operate the ALPHON device, are discussed in the following paragraphs.

## 3.1 ALPHON device

The ALPHON device can be used in combination with the SPEECH device to obtain an allophonic transscription of any standard english text. A second mode of operation is the "stand alone" mode, in which the user can directly experiment with the allophones.

In order to obtain a transcription of english text, the user has to output the text via the SPEECH device, and then input the allophonic transcription through the ALPHON device. For more details please refer to the paragraph about ALPHON INPUT.

### 3.1.1 OPEN command.

The ALPHON device can be opened in the following way

```
OPEN #x:"ALPHON",INTERNAL
```

All other parameters are default. The ALPHON device will use 255 byte records, unless otherwise instructed. Like the SPEECH device, all other default modes are required, i.e. the full OPEN statement would look like

```
OPEN #x:"ALPHON",INTERNAL,SEQUENTIAL,VARIABLE
```

The access mode specifier (UPDATE, OUTPUT, INPUT or APPEND) is not restricted to any specific mode. All modes are supported, with APPEND mode yielding the same result as OUTPUT mode.

The data-format has to be specified as INTERNAL, since the BASIC has no means of reading in binary data in DISPLAY format, without interpreting characters like "," and the quote mark itself.

### 3.1.2 PRINT command.

The PRINT command expects one string of allophone codes. Please note that this makes the construction

```
PRINT #1:CHR$(22);CHR$(56)
```

look like only one allophone (#22) to the ALPHON device. If the user wants to give more than one string argument to the ALPHON device, the individual arguments should be concatenated using the "&" operator. The construction

```
PRINT #1:CHR$(22)&CHR$(56)
```

would therefore give the expected result.

### 3.1.3 INPUT command.

The INPUT command can be used to input allophonic transcriptions of phrases that have been previously output, either through the SPEECH device, or through the ALPHON device itself. The INPUT command will always give the most recent allophone string spoken by the speech module. As an example, consider the following piece of code :

```
100 OPEN #1:"SPEECH",OUTPUT
110 OPEN #2:"ALPHON",INTERNAL
120 PRINT #1:"I AM THE T I HOME COMPUTER"
130 PRINT #1:"HELLO"
140 INPUT #2:A$
150 PRINT #2:A$
```

After the execution of line 140, the string variable A$ will contain the allophonic transcription of the word "HELLO", which was the most recent phrase spoken by the speech module. Line 150 will subsequently pronounce that phrase again.

The given allophone string will also include some special codes, which are used to get inflection in a phrase. These special codes are described in appendix B. In general, all codes above 240 are reserved for special functions.

As an example of what can be done when using the special function codes and direct allophone access, please try the following little program

```
100 OPEN #1:"ALPHON",INTERNAL
110 A$=CHR$(53)&CHR$(49)&CHR$(252)
120 A$=CHR$(252)&CHR$(30)&A$&CHR$(33)&A
130 A$=A&CHR$(36)&CHR$(67)
140 PRINT #1:A$
```

This was our first example of a singing computer, using permanent pitch modifiers and direct allophone input.

## 3.2 General restrictions for ALPHON

The ALPHON device has some restrictions in the codes it will accept. These restrictions are

1. The unit will stop speaking on any illegal allophone code, i.e. the codes between 128 and 247 inclusive.

2. Illegal pitch codes (codes > 63), will be masked off to 6 bits, i.e. code 64 is being mapped into code 0 again.

3. Slope parameters are always accepted, but are adjusted if they would cause pitch problems using the standard contouring algorithms.

4. If more then one primary stress point per phrase is given, or if the number of secundary stress points indicated in the "start phrase" code (250) is incorrect, the boundary checking algorithms may not work properly, in which case the pitch may cross 0 or 63. Both crossings will result in squeeking noises.

## 3.3 Allophone code description

This paragraph gives a short description of the allophone numbers currently assigned, and the sound associated with them.

| Allophone | Description | Allophone | Description |
|---|---|---|---|
| 1 - AE1 | as in ".A.ddition" | 41 - OOR2 | as in "P.oor.ly" |
| 2 - AE1N | as in ".A.nnuity" | 42 - OR2 | as in "H.or.se" |
| 3 - AH1 | as in "Delt.a." | 43 - OW2 | as in "B.oa.t" |
| 4 - AH1N | as in ".O.n time" | 44 - U2 | as in "Sh.oo.t" |
| 5 - AW1 | as in ".Au.tonomy" | 45 - UH2 | as in "H.u.t" |
| 6 - AW1N | as in "An.o.nimity" | 46 - UU2 | as in "B.oo.t" |
| 7 - E1 | as in ".E.liminate" | 47 - AE3 | as in "H.a.d" |
| 8 - E1N | as in ".E.nough" | 48 - AH3 | as in ".O.dd" |
| 9 - EH1 | as in "Cont.e.xt" | 49 - AI3 | as in "H.i.de" |
| 10 - EH1N | as in "Anci.e.nt" | 50 - AR3 | as in "C.a.rd" |
| 11 - ER1N | as in "West.e.rn" | 51 - AU3 | as in "L.ou.d" |
| 12 - I1 | as in "Synth.e.s.i.s" | 52 - AW3 | as in "S.a.w" |
| 13 - I1N | as in ".I.nane" | 53 - E3 | as in "S.ee.d" |
| 14 - OO1 | as in "T.oo.k on" | 54 - EELL | as in "H.eel." |
| 15 - OW1N | as in "D.o.nation" | 55 - EER3 | as in "H.ear." |
| 16 - U1 | as in "Ann.u.al" | 56 - EH3 | as in "S.ai.d" |
| 17 - U1N | as in ".U.nique" | 57 - EHR3 | as in "Th.ere." |
| 18 - UH1 | as in ".A.bove" | 58 - EI3 | as in "D.ay." |
| 19 - UH1M | as in "Instr.u.ments" | 59 - ER3 | as in "H.ear.d" |
| 20 - UH1N | as in "..U.nderneath" | 60 - I3 | as in "H.i.d" |
| 21 - Y1 | as in "Ros.e.s" | 61 - ILL | as in "H.ill." |
| 22 - Y1N | as in "Basem.e.nt" | 62 - ING2 | as in "Th.in.k" |
| 23 - ER1 | as in "Seek.e.r" | 63 - OI3 | as in "B.oy." |
| 24 - OW1 | as in "Rati.o." | 64 - OO3 | as in "C.ou.ld" |
| 25 - Y2 | as in "Funn.y." | 65 - OOR3 | as in "P.oor." |
| 26 - AE2 | as in "H.a.t" | 66 - OR3 | as in "C.ore." |
| 27 - AH2 | as in "H.o.t" | 67 - OW3 | as in "L.ow." |
| 28 - AI2 | as in "H.ei.ght" | 68 - U3 | as in "Sh.oe." |
| 29 - AR2 | as in "C.a.rt" | 69 - UH3 | as in "M.u.d" |
| 30 - AU2 | as in "H.ou.se" | 70 - ULL | as in "Sk.ull." |
| 31 - AW2 | as in "S.ou.ght" | 71 - UHL | as in "P.ull." |
| 32 - E2 | as in "H.ea.t" | 72 - UU3 | as in "M.oo.n" |
| 33 - EER2 | as in "P.ie.rce" | 73 - L | as in ".L.ike" |
| 34 - EH2 | as in "S.e.t" | 74 - L- | as in "Bow.l." |
| 35 - EHR2 | as in "Th.e.rapy" | 75 - LL | as in "Awf.ul." |
| 36 - EI2 | as in "T.a.ke" | 76 - M | as in ".M.ay" |
| 37 - ER2 | as in "H.u.rt" | 77 - MM | as in "Hu.m." |
| 38 - I2 | as in ".I.ssue" | 78 - N | as in ".N.ice" |
| 39 - OI2 | as in "Ch.oi.ce" | 79 - NN | as in "Sa.ne." |
| 40 - OO2 | as in "C.oo.k" | 80 - NG1 | as in "Thi.n.k" |

| Allophone | Description |
|---|---|
| 81 - NG2 | as in ".Thi.ng." |
| 82 - R | as in ".R.eal" |
| 83 - W | as in ".W.itch" |
| 84 - WH | as in ".Wh.ich" |
| 85 - Y | as in ".Y.ou" |
| 86 - B | as in ".B.ad" |
| 87 - BB | as in "Da.b." |
| 88 - D | as in ".D.ig" |
| 89 - DD | as in "Bi.d." |
| 90 - G1 | as in ".G.ive" |
| 91 - G2 | as in ".G.o" |
| 92 - GG | as in "Ba.g." |
| 93 - J | as in ".J.ug" |
| 94 - JJ | as in "Bu.dge." |
| 95 - THV | as in ".Th.is" |
| 96 - THV- | as in "Clo.the." |
| 97 - V | as in ".V.ine" |
| 98 - VV | as in "Li.ve." |
| 99 - Z | as in ".Z.oo" |
| 100 - ZZ | as in "Doe.s." |
| 101 - ZH | as in "A.z.ure" |
| 102 - ZH- | as in "Bei.ge." |
| 103 - K2 | as in "S.k.ate" |
| 104 - KH | as in ".C.ase" |
| 105 - KH- | as in "Ma.ke." |
| 106 - KH1 | as in ".K.ey" |
| 107 - KH2 | as in ".C.ough" |
| 108 - P | as in "S.p.ace" |
| 109 - PH | as in ".P.ie" |
| 110 - PH- | as in "Na.p." |

| Allophone | Description |
|---|---|
| 111 - T | as in "S.t.ake" |
| 112 - TH | as in ".T.ie" |
| 113 - TH- | as in "La.te." |
| 114 - CH | as in ".Ch.ur.ch." |
| 115 - F | as in ".F.at" |
| 116 - FF | as in "Lau.gh." |
| 117 - HI | as in ".H.it" |
| 118 - HO | as in ".H.ome" |
| 119 - HUH | as in ".H.ut" |
| 120 - S | as in ".S.eem" |
| 121 - SS | as in "Mi.ss." |
| 122 - SH | as in ".Sh.ine" |
| 123 - SH- | as in "Wa.sh." |
| 124 - THF | as in ".Th.ing" |
| 125 - THF- | as in "Wi.th." |
| 126 - Pause1 | <short pause> |
| 127 - Pause2 | <long pause> |

# APPENDIX A

## Speech Contouring Algorithms

The TTS system uses a pre-defined set of rules to translate secondary and primary stress points into pitch variations. This appendix will give an explanation of how those rules interpret the stress points, and what effects stress points have on the pitch in a phrase.

Sentence profiles can be subdivided into two major groups :

1. Falling phrase mode
2. Rising phrase mode

Typically a rising phrase mode occurs in sentences terminated by a "," or a "?". The falling phrase mode prevails in any other situation.

Stress points are only used for vowel allophones. These allophones are all grouped in the range 1 through 73. All the other allophones are not used for sentence profiling. Future profiling algorithms may start using these allophones too however.

### A.1 Falling Phrase Mode

A falling phrase centers around a falling pitch on the primary stress point. If the primary stress point is followed by one or more secondary stress points, the pitch on the primary stress point will fall from 20% above the average pitch level, back to the average pitch level (see Figure A-1).

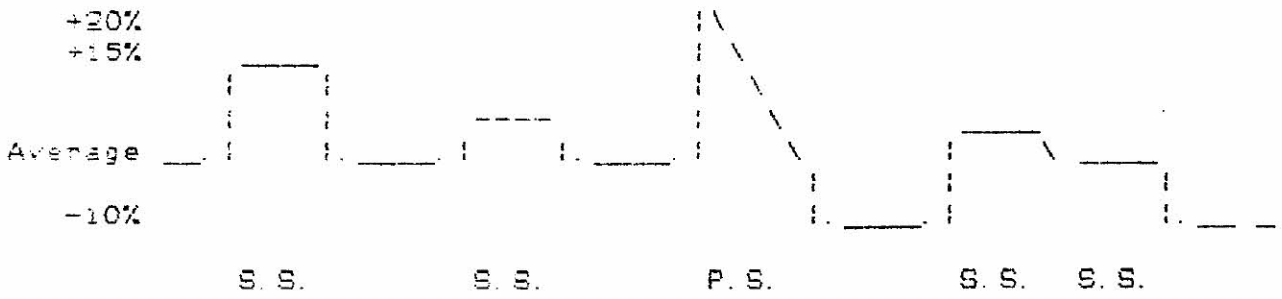


Figure A-1 Secondary Stress after Falling Primary Stress

If the primary stress point is the last stress point in the phrase, the pitch on this point will fall from 10% above the average pitch level to 15% below average pitch level (see Figure A-2).

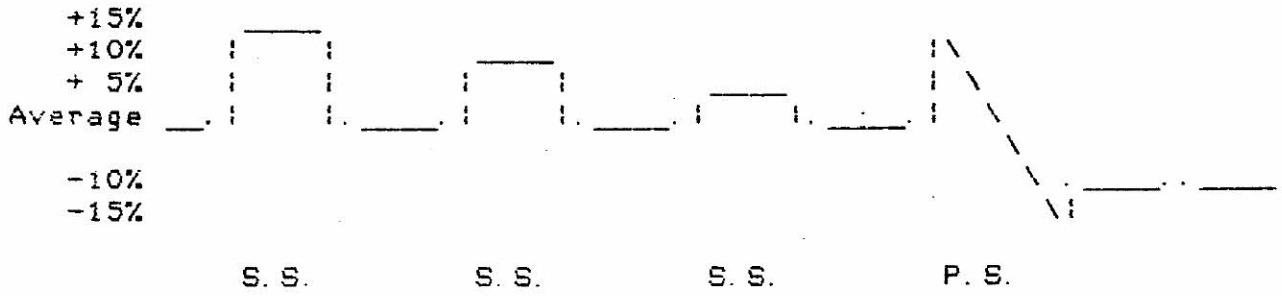


Figure A-2 No Secondary Stress after Falling Primary Stress

Notice that in both cases the default pitch level after the primary stress is lowered by 10%.

The secondary stress points before and after the primary stress point are treated in exactly the same way for falling phrase mode. The first secondary stress point is always positioned 15% above the default level. Any further secondary stress points are spread out evenly between a 15% raise and the default level. E.g. for two stress points the first one will be 15% above default level, and the second one 7.5% above default.

## A.2 Rising Phrase Mode

A rising phrase, like a falling phrase, is centered around the primary stress point. As the name already implies, the primary stress point will follow a rising contour. However, if the primary stress point is followed by one or more vowels, the rising contour is spread out over all the vowels following it (see Figure A-3), starting at the default level for the primary stress point itself, and ending at 15% above the default level for the last vowel.

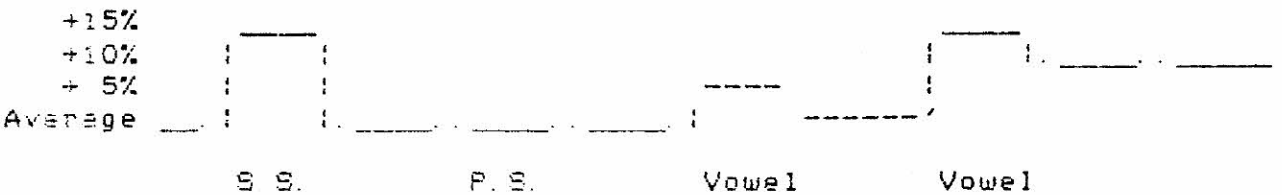


Figure A-3 One or more Vowels after Rising Primary Stress

If no vowels are following the primary stress point, the primary stress point has a gradual upswing, starting at a level 10% below the current default level, and rising to a level 15% above the current default level (see Figure A-4).

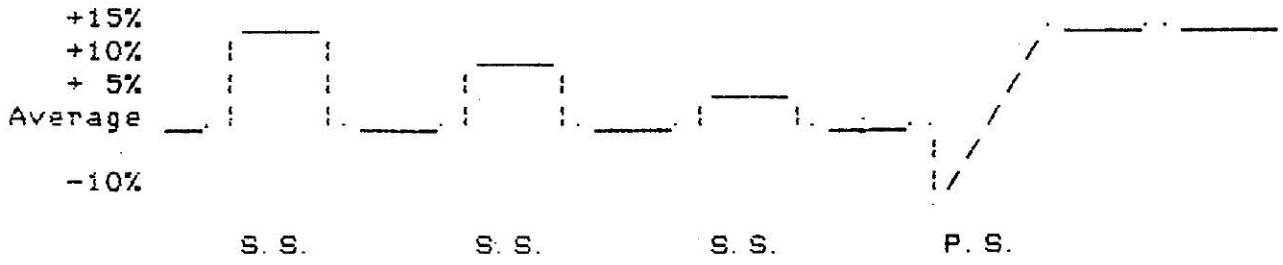


Figure A-4 No Vowels after Rising Primary Stress

# APPENDIX B

## Inflection Codes

In general, the allophone code space is subdivided into two areas. The first one, ranging from the numbers 1 through 240, is the actual allophone range. At the moment only 127 out of those 240 codes are actually used.

The second range, from 241 up to and including 255, is used for special function codes. Each special function has up to two additional bytes associated with it. The remainder of this appendix is used to describe the codes currently used.

249 Secondary stress point. Depending on the state the stringer is in, the stress point will either be given the current vowel pitch level, and the vowel pitch will be incremented by a computed delta amount, or, after an upgoing primary stress point, the stress point will be raised one level above the current vowel pitch level, and the vowel pitch will be decremented by the computed delta amount.

250 Sentence break code. Resets parameters to default values given by "//" record in SPEECH. SPEECH initially starts out with the default values PITCH = 43 and SLOPE = 128 (4*32).

This code is followed by two parameters: the number of secondary stress points before and after the primary stress point respectively. If the primary stress point is a rising (question mark) stress, the second parameter indicates the number of vowels after the primary stress point. If the first parameter has either of the values 254 or 255, the second parameter will indicate the total number of vowels in the phrase.

A value of 254 or 255 for the first parameter indicates that the entire phrase should have a rising or falling tendency respectively. This tendency is spread out over at least 8 vowels, and slopes over 15% as indicated by the slope parameter in the "//" record in SPEECH.

251 New default slope parameter. The byte following this code indicates the new slope parameter to be used. As usual, the slope indicates 32 times the actual slope value, to allow for fractional values of the slope. Normally the slope parameter should be selected to be approximately 10% of the pitch parameter.

252 New default pitch parameter. The byte following this parameter indicates the new default pitch level to be used. Notice that this pitch level is only valid for the current phrase. Permanent pitch level modifications have to be made using the "//" record option in SPEECH.

253 Rising pitch contour. Used for question type phrases at primary stress point. Will cause the next allophone to be gradually raised in pitch, starting at a -10% level, to a +15% level.

If no vowels are following the primary stress point, as indicated by the most recent 250 code, the allophone will be put at the standard pitch level, but the vowels following the stress point will be contoured along a gradually rising slope, starting at the current level (with the current stressed allophone), and rising to a +15% level.

254 Falling pitch contour. Used for standard type phrases at primary stress point. Will cause the next allophone to be gradually lowered in pitch, starting at a +15% level if no secondary stress points are following, or at a +20% level if secondary stress points are following the primary stress point, as indicated by the most recent 250 indicator. The contour will always fall over a total of 25%, which equals 2.5 times the indicated slope value.

255 Temporary pitch level modification. Will modify the pitch level of the allophone following it to the level indicated in the second byte. This is useful in special applications like singing computers etc. although most of the effects can also be accomplished by using the permanent modifier code 252.