Microvox

Ciarcia's Circuit Cellar
Published September/October 1982 Byte Publications Inc.
by Steven A. Ciarcia, POB 582, Glastonbury, CT 06033 USA.

Build the Microvox Text-to-Speech Synthesiser Part 1. Hardware

The 6502 microprocessor in this intelligent peripheral device translates plain English text to phonemes to control a Votrax SC-01A.

Build the Microvox Text-to-Speech Synthesiser Part 2. Text-to-Speech

Rules for conversion of English plain text to phonemes govern the operation of this SC-01A-based device

Special thanks to Dianna Visek for her work on the text-to-speech algorithm.
Type-'N-Talk and Votrax are trademarks of Federal Screw Works.

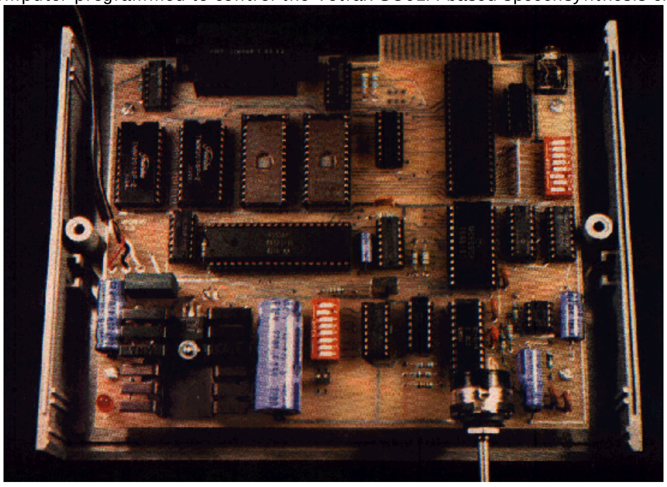
- Microvox

- This month's project may have a strange ring of familiarity to those of you who follow my activities in the Circuit Cellar. Twice before, in June and September of last year, I have written about peripheral devices that give personal computers the ability to speak with an imitation of a human voice.

- General-Purpose Computer

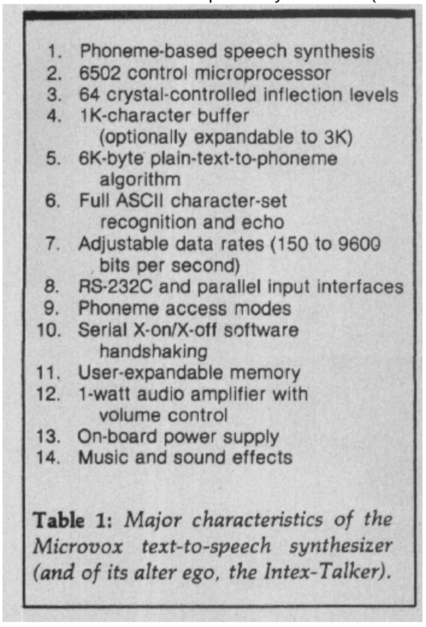
- The 6502-based microcomputer that forms an integral part of the Microvox is ideal for use in many other small-scale applications. Only the application sofiware and the interface to the SC-01A chip are specific to the micro-computer's use in the stand-alone text-to-speech voice synthesizer. If you are among the many readers who write to me asking for suggestions on how to put together a low-priced, general-purpose microcomputer system, you should consider building the computer part of the Microvox design.
- The computer section contains, among other things, a 1-MHz 8-bit 6502 microprocessor, a serial input port that can run at crystal-controlled data rates from 75 to 19,200 bps (bits per second) with full handshaking, 3 parallel input ports, provision for up to 4K bytes of RAM (random-access read/write memory) and 16K bytes of EPROM (erasable programmable read-only memory), and an on-board power supply. It is suitable for use as a learning tool for computer concepts, as a dedicated device controller, or as the center of an expanded microcomputer system (similar to systems that have been built around the MOS Technology KIM-1 or the Rockwell AIM-65).
- The Micromint will be supplying essential components of the microcomputer section of the Microvox for those who wish to experiment with it. And you may expect to see the same 6502-based control-computer design in future Circuit Cellar projects.
- The September article (see reference 5) described the Sweet Talker speech synthesizer, which has since become especially popular. The original Sweet Talker, a parallel-interfaced synthesizer module programmed by phoneme (speech sound) codes, was quickly joined by a version that could be plugged into an Apple II computer and operated using a text-to-speech algorithm stored on a floppy disk.
- But I wasn't satisfied. Neither the Sweet Talker nor my June project (see reference 4), the Micromouth, was flexible enough to fit the variety of applications I had envisioned. I could foresee applications requiring unlimited vocabulary (thus ruling out use of the Micromouth) that also need a smaller, more portable voice-synthesis system than could be made out of an Apple II. While I was content with the Sweet Talker's speech quality, I did not want to try converting the text-to-speech algorithm to run on my Z8-BASIC Microcomputer.

Photo 1: Prototype of the Microvox speech synthesiser, which can pronounce texts consisting of English words from their representation as ASCII characters according to fixed pronunciation rules. The Microvox contains a general-purpose 6502-based microcomputer programmed to control the Votrax SC01A-based speechsynthesis circuitry.



- I next considered using the Votrax Type-'N-Talk. As a stand-alone voice synthesizer with a built-in microprocessor and 4K-byte text-to-speech algorithm, it does quite well considering its moderate cost (see reference 12). However, its design is somewhat limited for commercial applications. Not finding any other suitable product on the market, I did what any red-blooded engineer would naturally do: I decided to design an improved text-to-speech voice synthesizer. You are reading the first of two articles on the design, construction, and operation of a text-to-speech voice synthesizer I call the Microvox. This new device, like the Sweet Talker, is based on the Votrax SC01A speechsynthesis integrated circuit, but it incorporates new functions (most notably pitch inflection) and a larger, more complex control program. A list of its features appears in table 1 on page 66.
- To support its various functions, the Microvox contains a general-purpose 6502-based microcomputer programmed to control the speech-synthesis circuitry. Program routines stored in ROM (read-only memory) activate various control options upon the user's command; the most complex of the routines performs the crucial task of translating the Microvox's input - a stream of text represented by ASCII (American Standard Code for Information Interchange) character codes into the special phoneme codes required by the SC-0IA chip. Incidentally, this 6502-based microcomputer is ideal for use in many other smallscale applications, as the text box explains.
- As with many Circuit Cellar projects, the Microvox design has been cast in printed circuit, and I have arranged for The Micromint to offer a kit of the parts needed to build it. Furthermore, an assembled, FCC- (Federal Communications Commission) approved version of the unit is being sold by Intex Micro Systems Corporation under the trade name Intex Talker. Information on availability of both products appears at the end of this article.
- I cannot thoroughly cover such a comprehensive topic in one article, so this month I shall present only the hardware and a brief overview of the system commands. Next month in Part 2, I'll discuss the design of the text-to-speech algorithm and the system software.
- Let's begin with an explanation of what we are trying to accomplish and a brief review of the Votrax SC01 chip and phonetic speech synthesis in general.

Table 1: Major characteristics of the Microvox text-to-speech synthesizer (and of its alter ego, the Intex-Talker).

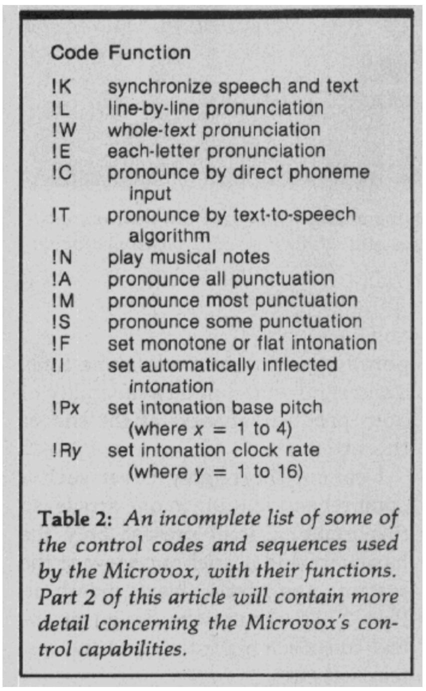


1. Phoneme-based speech synthesis
2. 6502 control microprocessor
3. 64 crystal-controlled inflection levels
4. 1K-character buffer (optionally expandable to 3K)
5. 6K-byte plain-text-to-phoneme algorithm
6. Full ASCII character-set recognition and echo
7. Adjustable data rates (150 to 9600 bits per second)
8. RS-232C and parallel input interfaces
9. Phoneme access modes
10. Serial X-on/X-off software handshaking
11. User-expandable memory
12. 1-watt audio amplifier with volume control
13. On-board power supply
14. Music and sound effects

Table 1: *Major characteristics of the Microvox text-to-speech synthesizer (and of its alter ego, the Intex-Talker).*

- Text-to-Speech Background
- Many articles in BYTE and other technical magazines have been devoted to the topic of computer speech synthesis. In general, they have dealt more with the production of the speech interface and the technology of specific synthesizers than with the applications to which speech synthesis may be put. Such treatment is similar to comparing computer systems by their processor instruction sets only instead of the highlevel-language software available for them. Today, far more computer users are concerned with applications than with construction of computers or peripheral devices. The Microvox is designed for easy use in a wide variety of applications.
- With the majority of lowcost speech-synthesizer interfaces, the user must arrange for conversion of the material to be spoken from textual characters to data that the speech synthesizer can work with (phonemes, linear-predictive-coding formants, word codes, etc.). The difficulty of conversion depends largely on the size of the required vocabulary. For small vocabularies, a table of words and their corresponding synthesizer codes can be compiled with reasonable effort. When the required vocabulary becomes very large, all-inclusive tables become prohibitively cumbersome, and a generalized text-to-speech algorithm is required instead.
- A text-to-speech algorithm is embodied in a program that accepts ASCII characters as input and performs a synthesis-by-rule analysis of character strings; that is, the algorithm interprets the characters as words or other elements of language and devises a scheme for pronouncing them according to a fixed set of rules that determine which characters are voiced, and in what way, and which characters are silent. The rules are based on how given combinations of characters are pronounced most of the time in English (or the language in use).
- Text-to-speech programs vary in length depending upon the degree of exactness required in pronunciation. Typical algorithms use from 4K to 8K bytes of object code for most processors, but some of the more sophisticated programs need up to 80K bytes. (Often, half of an 80K-byte synthesis-by-rule routine consists of tables of words that are exceptions to the rules.)
- The primary difference you can see between a 6K-byte and a 20K-byte program is how the input text must be spelled to obtain acceptable pronunciation; the final sound quality may be the same. Certain words may be spelled unusually to fit the prescribed pronunciation rules of the smaller algorithm. For instance, my name, Ciarcia, is properly pronounced by most synthesizers (and by a lot of people, come to think of it) only when it is spelled "see-are-see-ah." The only other major differences are features such as pronunciation of punctuation or inflected speech. (Both of these capabilities are supported by the Microvox.)

Table 2: An incomplete list of some of the control codes and sequences used by the Microvox, with their functions.
Part 2 of this article will contain more detail concerning the Microvox's control capabilities.



Code	Function
!K	synchronize speech and text
!L	line-by-line pronunciation
!W	whole-text pronunciation
!E	each-letter pronunciation
!C	pronounce by direct phoneme input
!T	pronounce by text-to-speech algorithm
!N	play musical notes
!A	pronounce all punctuation
!M	pronounce most punctuation
!S	pronounce some punctuation
!F	set monotone or flat intonation
!I	set automatically inflected intonation
!Px	set intonation base pitch (where $x = 1$ to 4)
!Ry	set intonation clock rate (where $y = 1$ to 16)

Table 2: *An incomplete list of some of the control codes and sequences used by the Microvox, with their functions. Part 2 of this article will contain more detail concerning the Microvox's control capabilities.*

- Strengths of Microvox

- While there are many speech-synthesizer interfaces designed to be used with a variety of personal computers, packaging the text-to-speech algorithm with its own dedicated processor greatly simplifies the integration of any system. By creating an intelligent peripheral device, we don't have to depend on operating systems and application programs to support speech synthesis.
- The Microvox text-to-speech synthesizer is just such a smart peripheral device. It speaks any ASCII character string directed to it through either its serial or parallel input ports. The ASCII text can come from PRINT statements in a BASIC program or from a previously prepared disk file. Microvox connects to the computer in the same manner as a printer or modem, and virtually anything that can be printed or viewed on the terminal screen can be spoken.
- The Microvox is controlled by the host computer through that same connection by means of special character sequences either transmitted before the text to be spoken or embedded in it. These control sequences are in the form:

- !letter, numeral

- The exclamation point is a signal to the Microvox that a control sequence follows. Operating modes and options can be changed at any time by sending the appropriate sequences. Table 2 lists some of the control sequences and their functions. I'll write about the intricacies of the Microvox text-to-speech algorithm and the control capabilities next month.

Figure 1: The general scheme to be followed in connecting the Votrax SC-01A to a microcomputer system.

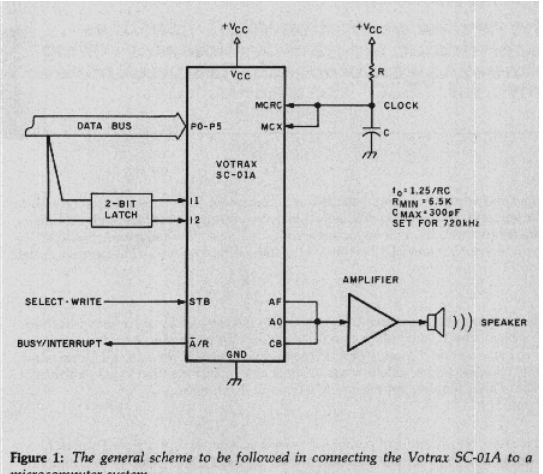


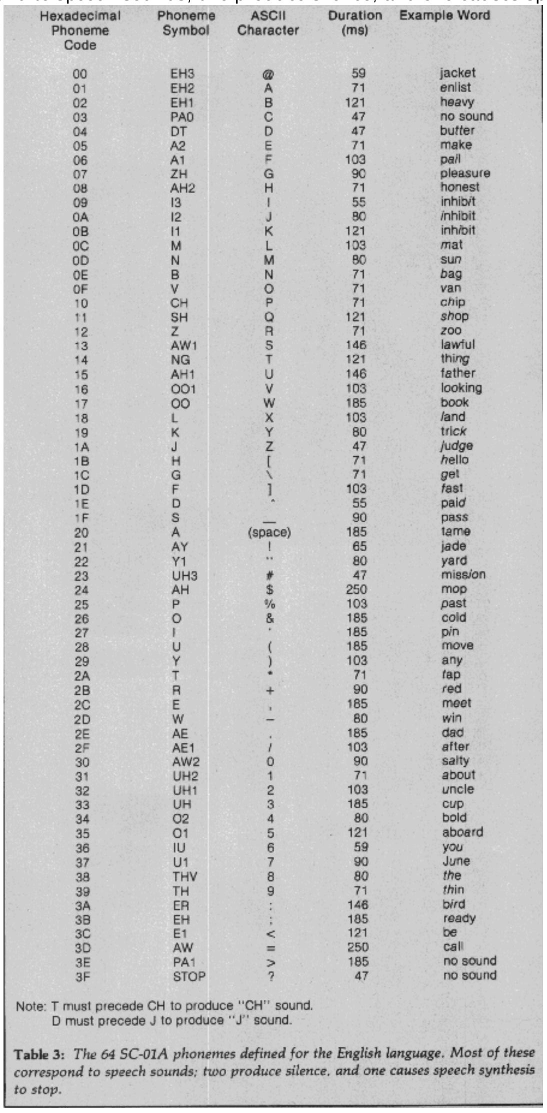
Figure 1: *The general scheme to be followed in connecting the Votrax SC-01A to a microcomputer system.*

- SC-01A Phoneme Synthesizer

- As I mentioned before, the Microvox is a combination of two major elements: a 6502-based control microcomputer and a Votrax SC-01A speech-synthesizer chip. I explained the SC-01A in detail in last September's Circuit Cellar article (reference 5), but for new readers I'll summarize the important facts.

- The SC-01A is a 22-pin integrated circuit which consists of a digital code translator and an electronic model of the human vocal tract. The internal phoneme controller translates a 6-bit phoneme code and 2-bit pitch code into a matrix of spectral parameters that adjust the vocaltract model to synthesize speech.
- The SC-01A is manufactured using CMOS (complementary metaloxide semiconductor) technology and operates within a range from + 7 to + 14 V. Handshaking with external control circuitry is accomplished through a strobe (STB) line and an acknowledge/request (A/R) line. A diagram of the generalized connection scheme appears as figure 1.
- The output pitch of the SC-01A's voice is controlled by the frequency of the clock signal, which can either be supplied from an external source or set internally with a resistor/capacitor combination. The clock frequency is nominally 720 kHz, but subtle variations of pitch can be induced to add inflection by varying this frequency. Such variations prevent the synthesized voice from sounding too monotonous or artificial. Two separate pitchcontrol lines, I1 and I2, are available for gross variations in pitch so that the chip can seem to speak with more than one voice. These socalled manual-inflection controls operate independently of clock-rate-induced inflection.
- The 64 SC-01A phonemes defined for the English language are listed in table 3 on page 72. Most of these correspond to speech sounds, but two produce silence and one causes speech synthesis to stop. The sound for each phoneme is generated when a 6-bit phoneme code is placed on the control-register input lines (P0 through P5) and latched by pulsing the strobe (STB) input. Each phoneme is internally timed and has a duration ranging from 47 to 250 ms (milliseconds) depending on the phoneme selected and the clock frequency. The A/R line goes from a logic 1 to a logic 0 while a phoneme is sounding.
- The usual method for using the SC-01A with a microprocessor sets up the hardware so that the computer system directly times the transmission of phoneme codes. This method sends phoneme codes to the synthesizer chip through a latched parallel output port and monitors the synthesizer's activities through the A/R line, which is connected to an input port or interrupt line.

Table 3: the 64 SC-01A phonemes defined for the English language.
Most of these correspond to speech sounds; two produce silence, and one causes speech synthesis to stop



Hexadecimal Phoneme Code	Phoneme Symbol	ASCII Character	Duration (ms)	Example Word
00	EH3	@	59	jacket
01	EH2	A	71	enlist
02	EH1	B	121	heavy
03	PA0	C	47	no sound
04	DT	D	47	butter
05	A2	E	71	make
06	A1	F	103	pail
07	ZH	G	90	pleasure
08	AH2	H	71	honest
09	I3	I	55	inhibit
0A	I2	J	80	inhibit
0B	I1	K	121	inhibit
0C	M	L	103	mat
0D	N	M	80	sun
0E	B	N	71	bag
0F	V	O	71	van
10	CH	P	71	chip
11	SH	Q	121	shop
12	Z	R	71	zoo
13	AW1	S	146	lawful
14	NG	T	121	thing
15	AH1	U	146	father
16	OO1	V	103	looking
17	OO	W	185	book
18	L	X	103	land
19	K	Y	80	trick
1A	J	Z	47	judge
1B	H	[	71	hello
1C	G	\	71	get
1D	F	]	103	fast
1E	D	^	55	paid
1F	S	_	90	pass
20	A	(space)	185	tame
21	AY	!	65	jade
22	Y1	"	80	yard
23	UH3	#	47	mission
24	AH	$	250	mop
25	P	%	103	past
26	O	&	185	cold
27	I	'	185	pin
28	U	(	185	move
29	Y	)	103	any
2A	T	*	71	tap
2B	R	+	90	red
2C	E	,	185	meet
2D	W	–	80	win
2E	AE	.	185	dad
2F	AE1	/	103	after
30	AW2	0	90	salty
31	UH2	1	71	about
32	UH1	2	103	uncle
33	UH	3	185	cup
34	O2	4	80	bold
35	O1	5	121	aboard
36	IU	6	59	you
37	U1	7	90	June
38	THV	8	80	the
39	TH	9	71	thin
3A	ER	:	146	bird
3B	EH	;	185	ready
3C	E1	<	121	be
3D	AW	=	250	call
3E	PA1	>	185	no sound
3F	STOP	?	47	no sound

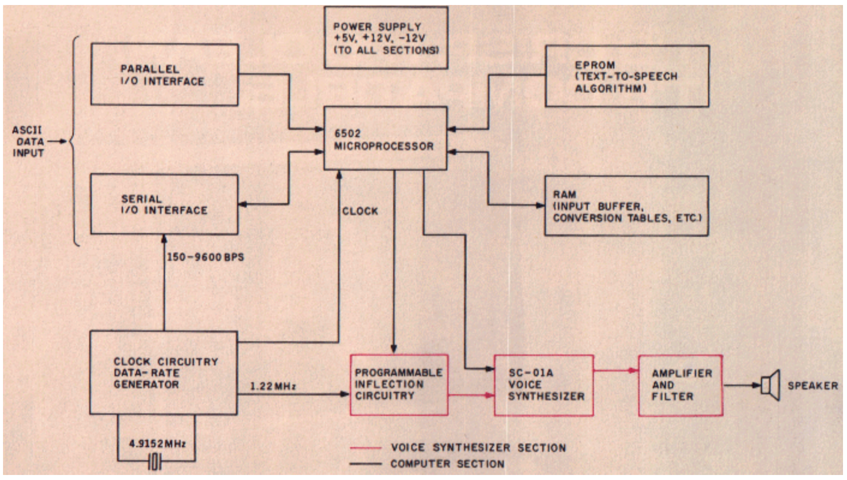
Note: T must precede CH to produce "CH" sound.
D must precede J to produce "J" sound.

Table 3: *The 64 SC-01A phonemes defined for the English language. Most of these correspond to speech sounds; two produce silence, and one causes speech synthesis to stop.*

-

The Microvox hardware can be viewed as a general-purpose 6502-based computer with a speech synthesizer attached using a memory-mapped I/O (input/output) port. The computer section (shown in black) uses 14 integrated circuits for serial and parallel I/O, address decoding, memory, and other processing functions. Five additional chips (outlined in red) constitute the phoneme synthesizer, inflection circuitry, and audio amplifier.

Figure 2: Block diagram of Microvox.


POWER SUPPLY +5V, +12V, -12V (TO ALL SECTIONS)
PARALLEL I/O INTERFACE
EPROM (TEXT-TO-SPEECH ALGORITHM)
ASCII DATA INPUT
6502 MICROPROCESSOR
SERIAL I/O INTERFACE
CLOCK
RAM (INPUT BUFFER, CONVERSION TABLES, ETC.)
150-9600 BPS
CLOCK CIRCUITRY DATA-RATE GENERATOR
1.22 MHz
PROGRAMMABLE INFLECTION CIRCUITRY
SC-01A VOICE SYNTHESIZER
AMPLIFIER AND FILTER
SPEAKER
4.9152 MHz
VOICE SYNTHESIZER SECTION
COMPUTER SECTION

- Microvox Hardware Overview

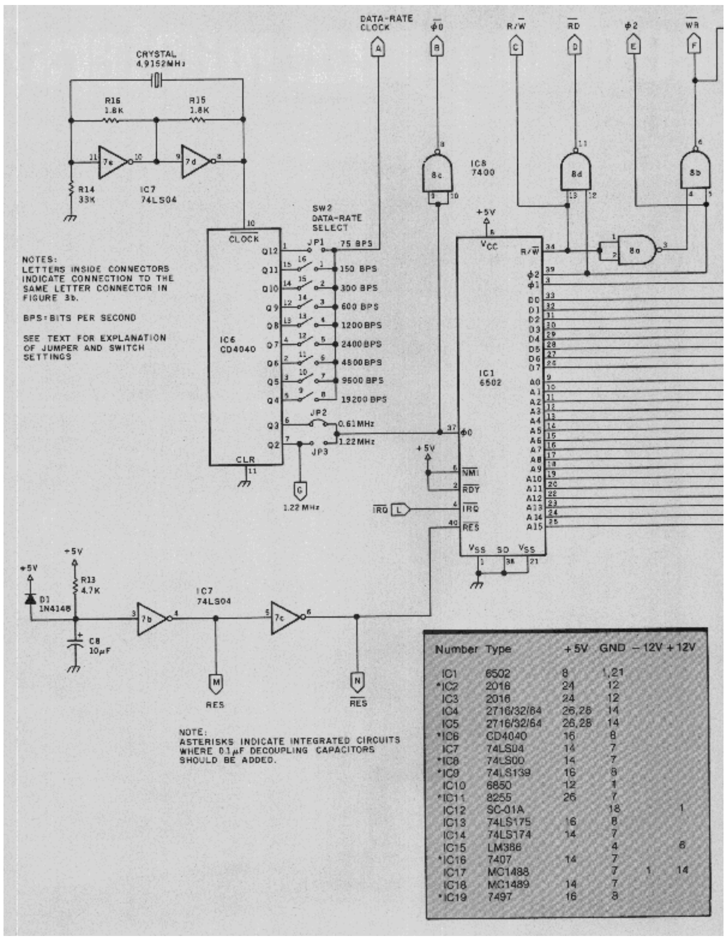
- Figure 2 is a basic block diagram of Microvox. As previously mentioned, the Microvox contains its own micro-computer that allows the unit to be configured to function as an intelligent peripheral device; therefore, the Microvox hardware can be viewed as a general-purpose 6502-based computer with a speech synthesizer attached using a memory-mapped I/O (input/output) port.
- The computer section uses 14 integrated circuits for serial and parallel I/O, address decoding, memory, and other processing functions. Five additional chips constitute the phoneme synthesizer, inflection circuitry, and audio amplifier (outlined in red).
- The Microvox is best explained by dividing the circuitry into four functional subsections: processor and timing, memory, serial and parallel I/O, and speech synthesizer. A complete schematic diagram of Microvox appears as figure 3a on pages 76 and 77 and figure 3b on pages 78 and 79.

Variations in pitch prevent the synthesized voice from sounding too monotonous or artificial.

- Processor and Data-Rate Clock

- The 1-MHz (megahertz) 6502 microprocessor, the same type used in the Apple II and Atari 800 computers, and the data-rate generator (shown by itself in figure 4 on page 80) obtain their clock signals from a circuit that divides down a 4.9152-MHz frequency from a crystal-controlled oscillator. You may find the rationale for using this lowcost clock divider interesting.
- Most data-rate-generator circuits are very costly because they use specialized data-rate generator chips such as the COM5016, which you must have if you really need to cover 134.5 and 110 bps (bits per second) as well as the other standard data rates from 75 to 19,200 bps. The former two data rates are the only ones that require oddball frequencies. If you can get along without them (and most people can nowadays), no special divider networks or integrated circuits are required. By using a 4.9152-MHz (75 x 2(16)) base frequency and a 12-stage binary divider (a CD4040, IC6 in figure 4), the nine remaining rates are derived directly.

Figure 3a: A section of the Microvox schematic diagram.
Shown here are the 6502 microprocessor and the timing section.
The schematic is continued in figure 3b on the next two pages.



Number	Type	+5V	GND	−12V	+12V
IC1	6502	8	1,21		
*IC2	2016	24	12		
IC3	2016	24	12		
IC4	2716/32/64	26,28	14		
IC5	2716/32/64	26,28	14		
*IC6	CD4040	16	8		
IC7	74LS04	14	7		
*IC8	74LS00	14	7		
*IC9	74LS139	16	8		
IC10	6850	12	1		
*IC11	8255	26	7		
IC12	SC-01A		18		1
IC13	74LS175	16	8		
IC14	74LS174	14	7		
IC15	LM386		4		6
*IC16	7407	14	7		
IC17	MC1488		7	1	14
IC18	MC1489	14	7		
*IC19	7497	16	8		

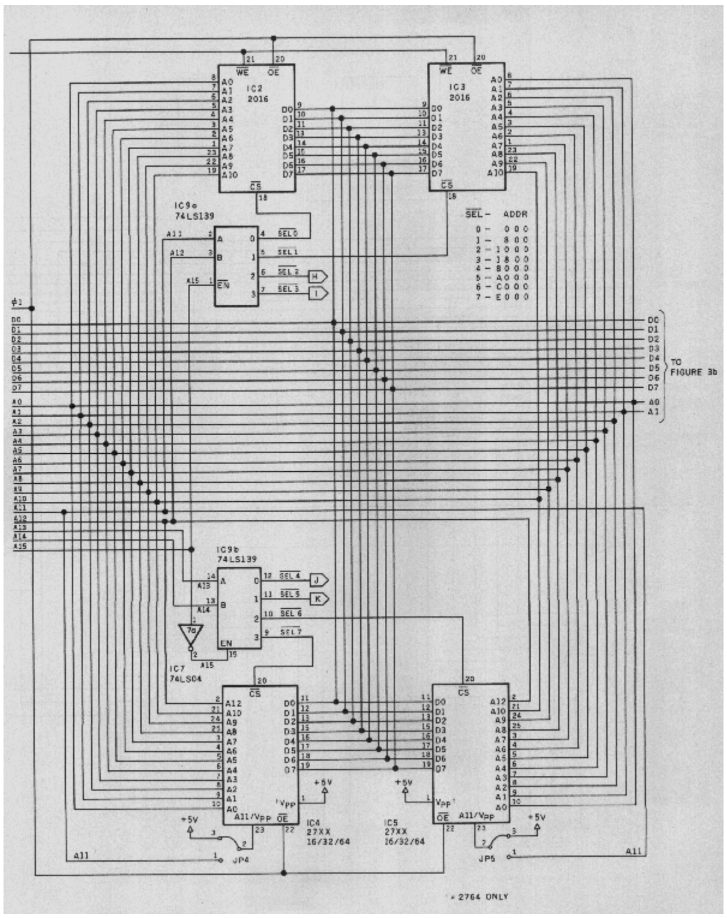
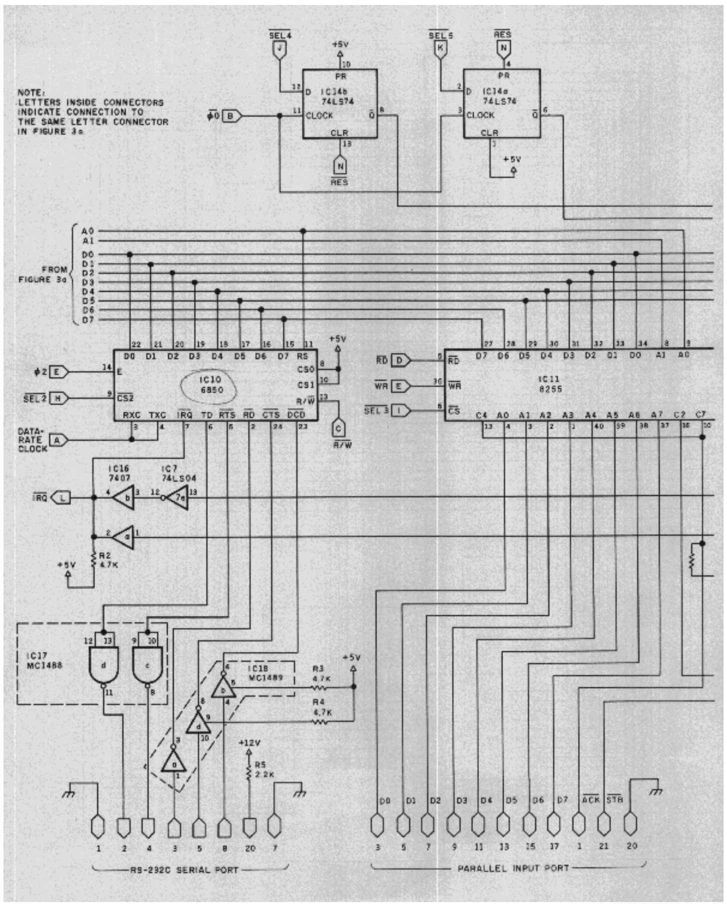
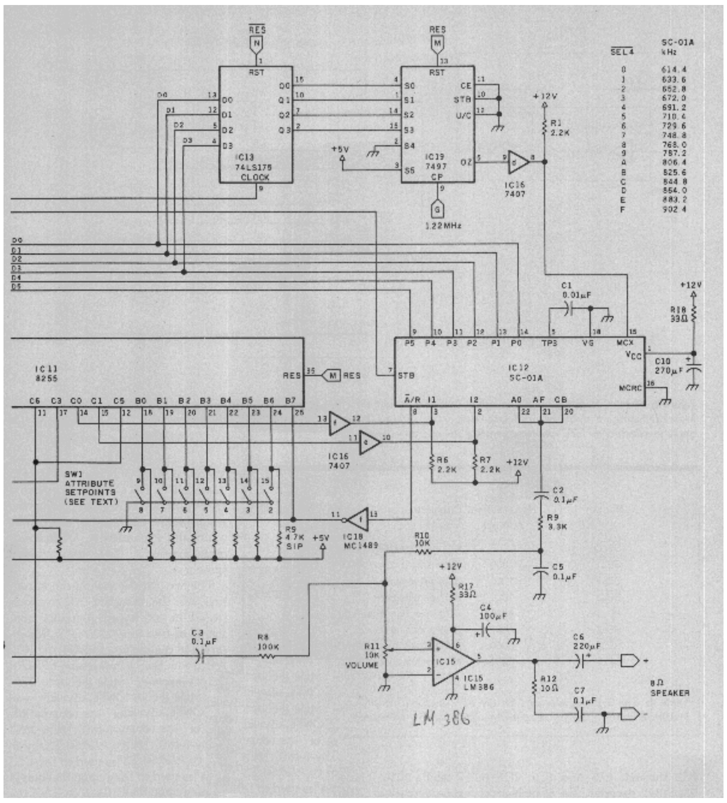


Figure 3b: A section of the Microvox schematic featuring the serial and parallel I/O and the SC-01A speech-synthesis intergrated circuit.





$\overline{SEL4}$	SC-01A kHz
0	614.4
1	633.6
2	652.8
3	672.0
4	691.2
5	710.4
6	729.6
7	748.8
8	768.0
9	787.2
A	806.4
B	825.6
C	844.8
D	864.0
E	883.2
F	902.4

- Processor and Data-Rate Clock, continued

- This approach does require one other design compromise. The 6502 processor is specified to operate at 1 MHz, but, using this crystal and divider circuit, only 611 kHz and 1.22 MHz are available as systemclock signals. The computer must run at either 61 percent or 122 percent of its rated speed.
- Practically speaking, this is not a problem. The 1-MHz specification is for worstcase conditions, which you probably will not have. I have personally run 1-MHz 6502s at 1.8 MHz with no trouble. Furthermore, in the Microvox application, we can note that the speech

synthesizer requires data at only about 200 bps to speak continuously. Processor speed is just not significant except when receiving and manipulating data at 19,200 bps. Just to be on the conservative side, while the hardware can produce rates from 75 to 19,200 bps, I have specified rates of 150 to 9600 bps for the Microvox.

Figure 4: A detail from the Microvox circuit, showing the circuitry that derives
the data-rate and clock frequencies. With this simple, lowcost arrangement,
all standard data rates, except 110 bps (bits per second) and 134.5 bps, are available.
Also, a trade-off must be made in selecting a clock rate for operating the microprocessor.

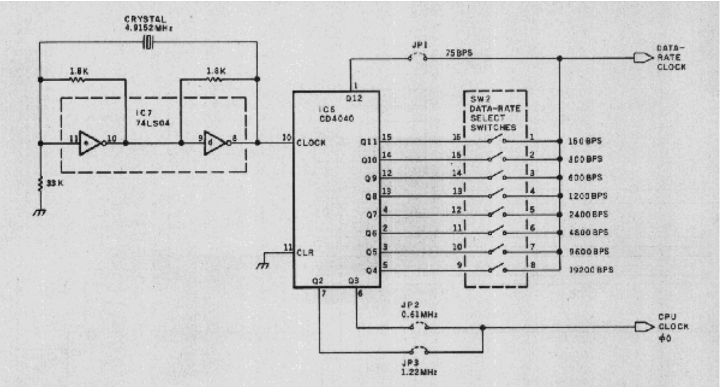
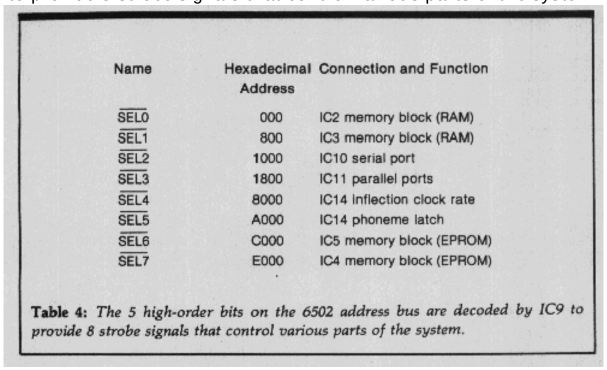


Figure 4: *A detail from the Microvox circuit, showing the circuitry that derives the data-rate and clock frequencies. With this simple, low-cost arrangement, all standard data rates, except 110 bps (bits per second) and 134.5 bps, are available. Also, a trade-off must be made in selecting a clock rate for operating the microprocessor.*

- Memory Section

- The address-decoding and memory section of the Microvox consists of IC2 through IC5 and IC9. IC9 (a 74LS139) decodes the 5 high-order bits on the address bus to provide 8 strobe signals, as listed in table 4. In the Microvox configuration, memory components IC2 and IC3 are intended to be RAM, while IC4 and IC5 are meant to be ROM or EPROM (erasable programmable read-only memory). The pin designations for IC2 and IC3 are for 2K-by-8-bit RAM chips, such as the Hitachi 6116 or Toshiba 2016. These components are pin-compatible with the type-2716 EPROM, so you could use 2716s in these sockets instead, if the computer were being used in some other application. The read/write memory (IC2 and IC3) is used for conversion tables and register stacks and as the ASCII input buffer.
- A buffer is required because the Microvox can receive data faster than it can speak it. The standard Microvox uses only one RAM chip (installed as IC2), which provides a 1K-byte input buffer; by adding the second RAM chip in IC3, this can be optionally expanded to 3K bytes of text memory (for long-winded speeches). The text-to-speech conversion routine for the standard Microvox is stored in 8K bytes, presently consisting of two type-2732 EPROMs inserted in the sockets for IC4 and IC5. As production increases or EPROM prices drop, a single 8K-byte 2764 EPROM (or its ROM equivalent) will be used. Any of the compatible type-2716 (2K-by-8-bit), type-2732 (4K-by-8-bit), or type-2764 (8K-by-8 bit) EPROMs can be used in these IC positions, depending upon the jumper selections JP4 and JP5.

Table 4: The 5 highorder bits on the 6502 address bus are decoded by IC9
to provide 8 strobe signals that control various parts of the system.



Name	Hexadecimal Address	Connection and Function
$\overline{SEL0}$	000	IC2 memory block (RAM)
$\overline{SEL1}$	800	IC3 memory block (RAM)
$\overline{SEL2}$	1000	IC10 serial port
$\overline{SEL3}$	1800	IC11 parallel ports
$\overline{SEL4}$	8000	IC14 inflection clock rate
$\overline{SEL5}$	A000	IC14 phoneme latch
$\overline{SEL6}$	C000	IC5 memory block (EPROM)
$\overline{SEL7}$	E000	IC4 memory block (EPROM)

Table 4: *The 5 high-order bits on the 6502 address bus are decoded by IC9 to provide 8 strobe signals that control various parts of the system.*

- Serial and Parallel I/O

- Microvox, unlike most other voice synthesizers, has both serial and parallel input ports to receive ASCII characters. The serial port uses a Motorola MC6850 ACIA (asynchronous communications interface adapter, IC10). During system initialization, the ACIA's functional configuration is set up; communication parameters such as character word length, clock division ratios, parity, and stop bits are selected by setting the proper bits in the ACIA's control register. The data rate is set by the system data-rate clock (from SW2 and IC6), and data is sent and received from the transmit- and receive-data registers, respectively. Framing errors, parity errors, buffer status, and handshaking status are determined by reading the ACIA's status register.
- On the Microvox, the serial port can be used with or without hardware handshaking, that is, with or without using the RS-232C Clear to Send, Data Carrier Detect, Ready to Send, and other lines. The Microvox software incorporates software hand-shaking, which is especially useful when communicating over a modem link or with terminals that do not use handshaking signals.
- When receiving ASCII text in the software-handshaking mode, the Microvox sends an "@" (at sign) to the host computer when its input buffer is almost full, signaling the host to stop sending data. The Microvox sends a "#" (number sign) when it is ready to receive data again. (The characters used for signaling can be changed to the X-on and X-off control characters if need be.)
- Obviously, this handshaking is not needed if the data comes from the host at a speed slower than the rate at which the buffer is emptied. The parallel-input section uses a programmable Intel 8255 PIA (peripheral interface adapter, IC11). As configured, 8 bits of the PIA are used to receive ASCII data in parallel format. By using two additional connections for data-available-strobe and acknowledge signals, the Microvox can be made to work with a Centronics-compatible parallel printer interface.

- Also attached to the PIA is the DIP (dual-inline pin) switch SW1, which can be used to select operating parameters as follows. Bit 0 selects hardware or software handshaking; bit 1 selects receipt of the ASCII input data through the serial or parallel port; bits 2 through 4 set the serial input word length, stop bits, and parity on the ACIA; and bits 5 through 7 are not used.

Figure 5: The business end of the Mircrovox, the circuitry that actually produces the artificial voice.
This design is similar to the Sweet Talker speech synthesizer; it is based also on the
Votrax SC-01A integrated circuit. The main improvement is provision for 64 levels
of pitch inflection, instead of the 4 levels available on the Sweet Talker.

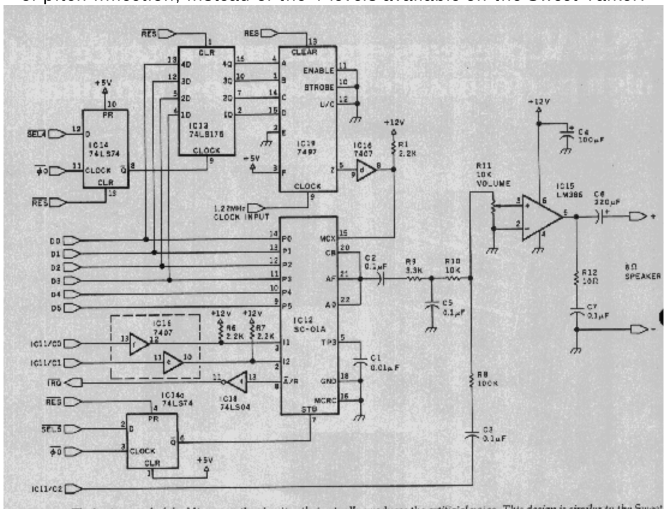


Figure 5: The business end of the Microvox, the circuitry that actually produces the artificial voice. This design is similar to the Sweet Talker speech synthesizer; it is based also on the Votrax SC-01A integrated circuit. The main improvement is provision for 64 levels of pitch inflection, instead of the 4 levels available on the Sweet Talker.

- Speech Inflection

- The business end of the Microvox, the circuitry that actually produces the artificial voice, is shown in the schematic diagram of figure 5. Regular followers of Circuit Cellar projects will recognize the Votrax SC-01A integrated circuit and notice that this design is similar to the Sweet Talker speech synthesizer from last September's article. This time, however, I have provided for 64 levels of pitch inflection, instead of the 4 levels previously available. The output pitch of the phonemes is fundamentally controlled by the frequency of the clock signal provided to the SC-01A. In general use, this frequency, set with a resistor/capacitor combination, is nominally 720 kHz. But as with any current-controlled analog circuit, the frequency may be susceptible to change from temperature variation and pickup of external noise.

Coarse variations in pitch are best used for simulating completely different speaking voices.

- In the Microvox, the analog clock circuitry is eliminated. Instead of using the SC-01A's internal timing circuit, the chip is configured for input of an external clock signal, derived from the crystal-controlled system clock. While the fundamental range of the output pitch is a function of the clock frequency, the two pitch-control lines I1 and 12 (the "manual-inflection" lines) can act independently to cause four coarse variations in pitch from the fundamental setting. I think that these coarse variations are best used for simulating completely different speaking voices rather than for vocal inflections. The frequency shift is simply too great.
- The preferred way to influence the output pitch is by changing the external clock frequency fed into the SC-0lA, although this takes more work. Subtle variations in output pitch can be obtained with reasonable effort, by shifting the clock frequency up or down by 20 or 40 kHz. And by applying a digital rate multiplier to the 1.22-MHz system dock, the signal input to the SC-01A can be made programmable to produce smaller and better defined pitch inflections. IC19 in the Microvox is a 6-bit binary rate multiplier. Its output frequency obeys the function:

FOUT = MxFIN
where
M = b5 X 32 + b4 X 16 + b3 X 8
+ b2 x 4 + bi X 2 + b0 X 1
(b5 through b0 being the six multiplier bits) and
FIN = 1.22 MHz

- When the SEL4 strobe is activated a 4-bit inflection code is latched into IC13 (a 74LS175 quad D flip-flop) and applied to the rate multiplier. The 4-bit combination (corresponding to a hexadecimal value of 0 to F loaded into IC13) selects one of 16 clock rates that range from 614.4 I+Iz to 902.4 kHz in 19.2-kHz increments. The frequency change of near 20 kHz creates a relatively small pitch change by itself (out of a 720-kHz nominal input frequency), but, used dynamically in a sentence, it is just what the doctor ordered for syllable inflection.
- Remember that the 2 manual-inflection bits are still available to the user; they are set by 2 bits on IC11 (SEL3). I refer to the level set by these bits as the "base pitch" and the 16 frequencies from the rate multiplier as the "clock rate." The combination of the 2 functions results in 64 pitch levels or inflections.
- The pitch at which individual phonemes are pronounced may be controlled automatically by the text-to-speech algorithm, kept fixed, or altered by user command. Some peopie prefer automatic inflection, because of the variety it gives to the speech. Others think a computer should sound like a computer and prefer flat speech to artificially intoned speech. Still others may wish to directly control the pitch to make the unit sing (pitch and rate codes may be mixed with phoneme codes to pro- duce singing) or to pronounce words with special emphasis.
- The user may control the base pitch setting independently of the clock rate by issuing a pitch-control command:

!Px
where
x is a digit from 1 through 4; x=1 selects the lowest pitch with pitch increasing according to the value of x.

The user may also control the clock rate with a command of the form

!Ry

where y can take on values from 1 to 16; y = 1 selects the lowest pitch; y=16 the highest.

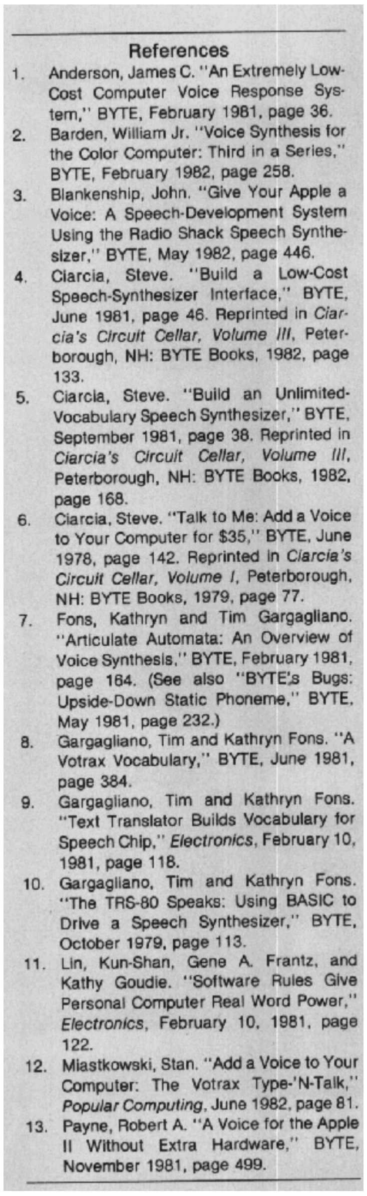
- Musical Abilities

- One final feature of the Microvox is the ability to play musical notes and produce sound effects by using a program routine to toggle one bit of the PIA (IC11) at a predetermined rate. This line is connected to the output audio amplifier along with the output from the speech synthesizer chip (IC12). The results are similar to the sound produced by the internal speaker of an Apple II computer using the same technique.
- While the Microvox is not exactly a virtuoso instrument, programming it to play a simple tune is not hard at all. The music mode is turned on by the command:
- !N
- Once the music mode has been activated, a different set of note-specifying commands is used.
- In the music mode, notes may be chosen from a range of 3 octaves centered on middle C, indicated by numbers from 1 to 3. Each octave contains notes identified as A, B, C, D, E, F, or G. Sharps are indicated by the suffix character " +", flats by "-". Time values are selected by reciprocal numeric arguments: the length of a note may vary from a whole note (length of 1) to a 128th note (length of 128). Rests are indicated by "R". When in the music mode, sending Microvox the character string "3F+4" causes it to play a quarter note at a pitch of F sharp in the third octave. "R16" causes a sixteenth-note rest.
- Notes of unconventional lengths may be used; for instance, the software supports "thirty-seventh" notes. Tempo may set from values of 50 to 128 beats per minute by a command of the type "Tx" with x in the proper range. The default tempo is 80.

- To Be Continued...

- I apologize if I am jumping ahead too quickly. It's just that I want you to be assured that these hardware features of music and programmable pitch are not an overcomplication; they are easily accommodated in the software.
- Obviously, there is no conclusion this month. I'll have a lot more to say next month. And keep in mind that while the main object of this project is an easy-to-use text-to-speech synthesizer, the computer section of the circuit has some special merit of its own. You may expect to see the same 6502-based control-computer design in future Circuit Cellar projects.

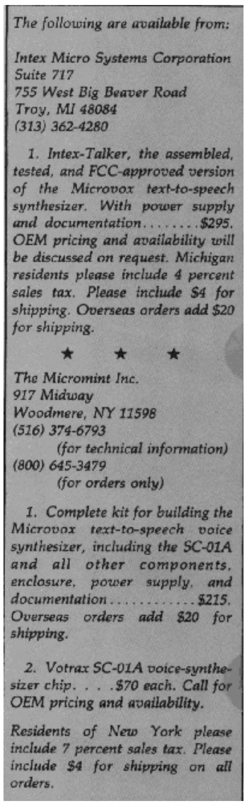
References



References

1. Anderson, James C. "An Extremely Low-Cost Computer Voice Response System," BYTE, February 1981, page 36.
2. Barden, William Jr. "Voice Synthesis for the Color Computer: Third in a Series," BYTE, February 1982, page 258.
3. Blankenship, John. "Give Your Apple a Voice: A Speech-Development System Using the Radio Shack Speech Synthesizer," BYTE, May 1982, page 446.
4. Ciarcia, Steve. "Build a Low-Cost Speech-Synthesizer Interface," BYTE, June 1981, page 46. Reprinted in *Ciarcia's Circuit Cellar, Volume III*, Peterborough, NH: BYTE Books, 1982, page 133.
5. Ciarcia, Steve. "Build an Unlimited-Vocabulary Speech Synthesizer," BYTE, September 1981, page 38. Reprinted in *Ciarcia's Circuit Cellar, Volume III*, Peterborough, NH: BYTE Books, 1982, page 168.
6. Ciarcia, Steve. "Talk to Me: Add a Voice to Your Computer for $35," BYTE, June 1978, page 142. Reprinted in *Ciarcia's Circuit Cellar, Volume I*, Peterborough, NH: BYTE Books, 1979, page 77.
7. Fons, Kathryn and Tim Gargagliano. "Articulate Automata: An Overview of Voice Synthesis," BYTE, February 1981, page 164. (See also "BYTE's Bugs: Upside-Down Static Phoneme," BYTE, May 1981, page 232.)
8. Gargagliano, Tim and Kathryn Fons. "A Votrax Vocabulary," BYTE, June 1981, page 384.
9. Gargagliano, Tim and Kathryn Fons. "Text Translator Builds Vocabulary for Speech Chip," *Electronics*, February 10, 1981, page 118.
10. Gargagliano, Tim and Kathryn Fons. "The TRS-80 Speaks: Using BASIC to Drive a Speech Synthesizer," BYTE, October 1979, page 113.
11. Lin, Kun-Shan, Gene A. Frantz, and Kathy Goudie. "Software Rules Give Personal Computer Real Word Power," *Electronics*, February 10, 1981, page 122.
12. Miastkowski, Stan. "Add a Voice to Your Computer: The Votrax Type-'N-Talk," *Popular Computing*, June 1982, page 81.
13. Payne, Robert A. "A Voice for the Apple II Without Extra Hardware," BYTE, November 1981, page 499.

The following are available from:



The following are available from:

Intex Micro Systems Corporation
Suite 717
755 West Big Beaver Road
Troy, MI 48084
(313) 362-4280

1. Intex-Talker, the assembled, tested, and FCC-approved version of the Microvox text-to-speech synthesizer. With power supply and documentation........$295. OEM pricing and availability will be discussed on request. Michigan residents please include 4 percent sales tax. Please include $4 for shipping. Overseas orders add $20 for shipping.

★ ★ ★

The Micromint Inc.
917 Midway
Woodmere, NY 11598
(516) 374-6793
(for technical information)
(800) 645-3479
(for orders only)

1. Complete kit for building the Microvox text-to-speech voice synthesizer, including the SC-01A and all other components, enclosure, power supply, and documentation............$215. Overseas orders add $20 for shipping.

2. Votrax SC-01A voice-synthesizer chip. . . .$70 each. Call for OEM pricing and availability.

Residents of New York please include 7 percent sales tax. Please include $4 for shipping on all orders.

Build the Microvox Text-to-Speech Synthesiser Part 2: Software

- This is the second of two articles on the design and construction of an advanced text-to-speech voice synthesizer that can be used as a peripheral device in most small computer systems. Its features (listed in table 1) include phonemebased speech synthesis, 64 inflection levels, software handshaking, and the ability to produce music and sound effects. In addition, the synthesizer recognizes and echoes the entire printable ASCII (American Standard Code for Information Interchange) character set, plus the control characters Return, Linefeed, Escape, and Backspace.
- The voice synthesizer is sold under two trade names: Microvox (from The Micromint Inc.) and Intex-Talker (from Intex Micro Systems Corporation). I'll call it the Microvox in this article. The hardware of the Microvox, described in detail last month and shown in photo 1, consists of a general-purpose 6502-based micro-computer with a voice-synthesizer output section. This month, I will concentrate on how text-to-speech algorithms work in general and on how the Microvox's program operates.

The Votrax SC-01 A chip allows the construction of English words and phrases from phonemes.

- Text-to-Speech Conversion

- By the end of the first or second grade, most people have the ability to convert written text in their native language into speech. This conversion has three basic steps:

1. the visual recognition of the characters in the printed text
2. the mental conversion of these characters into the appropriate commands to the mouth, tongue, larynx, and lungs
3. movement of the body parts to make the sounds

  We shall now look at how a computer can simulate the second and third of these tasks.
- The specific commands necessary to produce synthetic speech vary according to which speech synthesizer is being used. The Votrax SC-01A chip used in the Microvox is designed to allow the construction of English words and phrases from the phonemes (basic speech sounds) of the English language. (The phonemes used in the Votrax system are listed in table 2 on page 42.) Simulation of step 2 consists of converting a sequence of known characters into commands to voice-synthesis circuitry, which simulates the vocal cords and mouth.
- The basic task of the control program in the Microvox is to convert a string of characters making up an English-language phrase into the corresponding string of phonemes. In addition, as will be discussed, the computer should try to produce the appropriate intonation for each phoneme.
- Phrases can be converted to phonemes in three ways:

1. translating whole words to phonemes by looking the words up in a table, with one table entry for each word
2. breaking words into syntactically significant groups of letters (called morphs) and looking up the phonemes corresponding to each group of letters
3. applying a set of rules to letter patterns and individual letters in words

  Let's examine these in order.

Photo 1: An assembled Microvox speech synthesiser, which can produce texts consisting of English words from their representation as ASCII characters according to fixed pronunciation rules.

The Microvox contains a general-purpose 6502-based microcomputer programmed to control the Votrax-SC-01A-based speech-synthesis circuitry.

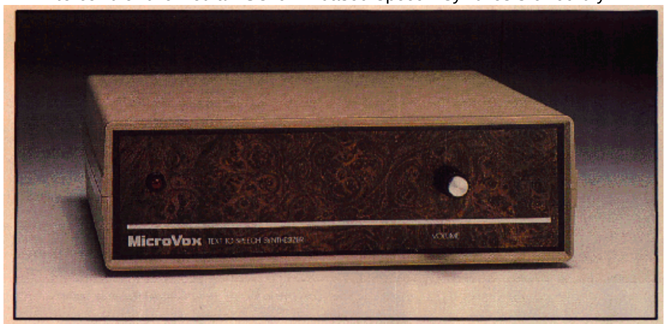
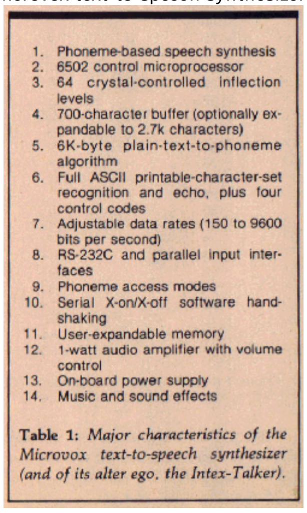


Photo 1: *An assembled Microvox speech synthesizer, which can pronounce texts consisting of English words from their representation as ASCII characters according to fixed pronunciation rules. The Microvox contains a general-purpose 6502-based microcomputer programmed to control the Votrax-SC-01A-based speech-synthesis circuitry.*

- Whole-Word Lookup

- Possessing the appropriate copyright license, you could store a standard pronouncing dictionary, such as A Pronouncing Dictionary of American English by Kenyon and Knott (reference 6), in computer memory. The input text could then be broken into its constituent words. After this, each word could be looked up in the dictionary and replaced with its corresponding pronunciation. This simple lookup program would contain no more than 1000 bytes.
- There are, however, two disadvantages with this method. First, because a lot of highspeed, randomly accessible storage would be needed to store a sufficiently large vocabulary, searching the list for each word might take too much run time. Second, whole-word lookup fails completely when given a word not in the dictionary; an unusual word, a newly coined term, or a proper name could cause failure. For the next few years anyway, whole-word lookup seems unpromising for most applications.

Table 1: Major characteristics of the Microvox text-to-speech synthesizer (and of its alter ego, the Intex-Talker).



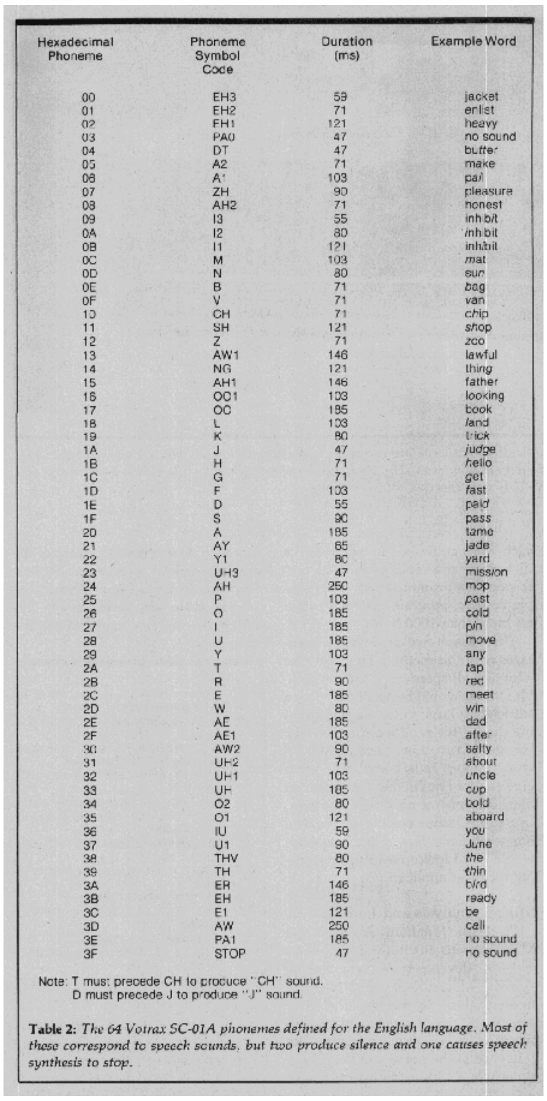
1. Phoneme-based speech synthesis
2. 6502 control microprocessor
3. 64 crystal-controlled inflection levels
4. 700-character buffer (optionally expandable to 2.7k characters)
5. 6K-byte plain-text-to-phoneme algorithm
6. Full ASCII printable-character-set recognition and echo, plus four control codes
7. Adjustable data rates (150 to 9600 bits per second)
8. RS-232C and parallel input interfaces
9. Phoneme access modes
10. Serial X-on/X-off software handshaking
11. User-expandable memory
12. 1-watt audio amplifier with volume control
13. On-board power supply
14. Music and sound effects

Table 1: *Major characteristics of the Microvox text-to-speech synthesizer (and of its alter ego, the Intex-Talker).*

- Morph Analysis and Lookup

- Professor Jonathan Allen of the Massachusetts Institute of Technology has developed a pronouncing system MITALK-79, that is based upon analysis of morphs the letter representations of constituent parts of words. In a recent article (see reference 1), he points out that a dictionary of 8000 morphs is sufficient to
- deal with more than 95 percent of the words in typical texts. Also, because new morphs are seldom formed, the morph dictionary rarely needs updating. In the few cases where the morph approach fails, MITALK-79 uses the letter-to-phoneme approach described later.
- Dr. Allen's system offers the best-quality output of any currently available text-to-speech system. Its processing and memory demands, however, stretch the limits of the present generation of microcomputers. If you allow 5 bytes for each morph and 5 bytes for its pronunciation, the morph dictionary will occupy 80,000 bytes. The algorithm for finding morphs considers all possible ways in which each word can be decomposed. Thus, it requires too much processing power to achieve real-time performance with a typical 8-bit microprocessor. Using a 16-bit computer would, of course, increase throug put, but the total cost of the system would be significantly higher.

Table 2: The 64 Votrax SC-01A phonemes defined for the English language. Most of these correspond to speech sounds, but two produce silence and one causes speech synthesis to stop.



Hexadecimal Phoneme	Phoneme Symbol Code	Duration (ms)	Example Word
00	EH3	59	jacket
01	EH2	71	enlist
02	EH1	121	heavy
03	PA0	47	no sound
04	DT	47	butter
05	A2	71	make
06	A1	103	pail
07	ZH	90	pleasure
08	AH2	71	honest
09	I3	55	inhibit
0A	I2	80	inhibit
0B	I1	121	inhibit
0C	M	103	mat
0D	N	80	sun
0E	B	71	bag
0F	V	71	van
10	CH	71	chip
11	SH	121	shop
12	Z	71	zoo
13	AW1	146	lawful
14	NG	121	thing
15	AH1	146	father
16	OO1	103	looking
17	OO	185	book
18	L	103	land
19	K	80	trick
1A	J	47	judge
1B	H	71	hello
1C	G	71	get
1D	F	103	fast
1E	D	55	paid
1F	S	90	pass
20	A	185	tame
21	AY	65	jade
22	Y1	80	yard
23	UH3	47	mission
24	AH	250	mop
25	P	103	past
26	O	185	cold
27	I	185	pin
28	U	185	move
29	Y	103	any
2A	T	71	tap
2B	R	90	red
2C	E	185	meet
2D	W	80	win
2E	AE	185	dad
2F	AE1	103	after
30	AW2	90	salty
31	UH2	71	about
32	UH1	103	uncle
33	UH	185	cup
34	O2	80	bold
35	O1	121	aboard
36	IU	59	you
37	U1	90	June
38	THV	80	the
39	TH	71	thin
3A	ER	146	bird
3B	EH	185	ready
3C	E1	121	be
3D	AW	250	call
3E	PA1	185	no sound
3F	STOP	47	no sound

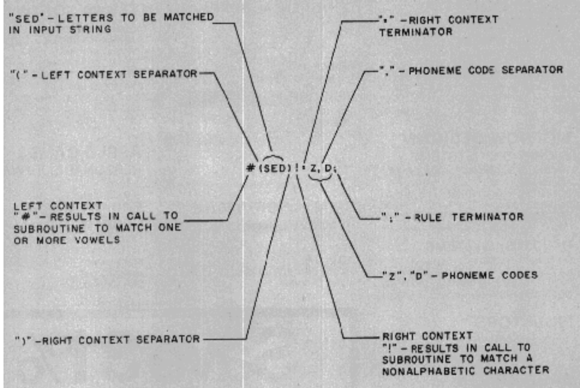
Note: T must precede CH to produce "CH" sound.
D must precede J to produce "J" sound.

Table 2: *The 64 Votrax SC-01A phonemes defined for the English language. Most of these correspond to speech sounds, but two produce silence and one causes speech synthesis to stop.*

- Letter-to-Phoneme Rules

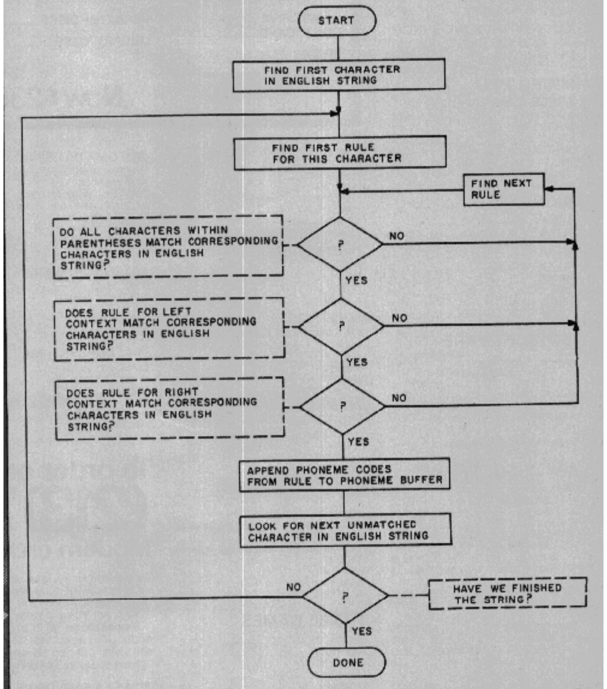
- Letter-to-phoneme rules are a necessary supplement to word or morph lookup because there will inevitably be words or morphs not found in the system's dictionary. By eliminating or at least greatly reducing the size of word and morph dictionaries, and relying mainly on letter-to-phoneme rules, it is possible to construct a text-to-speech program that will easily run in real time on an 8-bit microprocessor and will provide satisfactory performance with 4K to 8K bytes of memory.
- Probably the best of the publish' rulebased text-to-speech algorithm is that developed by a team at the Naval Research Laboratory (referred to as NRL; see reference 5). The text-to-speech algorithm embodied in the software of the Microvox is derived from the NRL algorithm, which combines word, morph, and letter rules in a single table of about 400 rules. This table contains subtables for each letter of the alphabet.

Figure 2: Flowchart of the text-to-speech algorithm used by the Microvox, which employs the rules of table 3.



"SED" - LETTERS TO BE MATCHED IN INPUT STRING
"(" - LEFT CONTEXT SEPARATOR
LEFT CONTEXT "#" - RESULTS IN CALL TO SUBROUTINE TO MATCH ONE OR MORE VOWELS
")" - RIGHT CONTEXT SEPARATOR
# (SED) ! = Z, D;
"=" - RIGHT CONTEXT TERMINATOR
"," - PHONEME CODE SEPARATOR
";" - RULE TERMINATOR
"Z", "D" - PHONEME CODES
RIGHT CONTEXT "!" - RESULTS IN CALL TO SUBROUTINE TO MATCH A NONALPHABETIC CHARACTER

Figure 1: *Interpretation of the format of the text-to-phoneme rules of table 3, on the next page.*



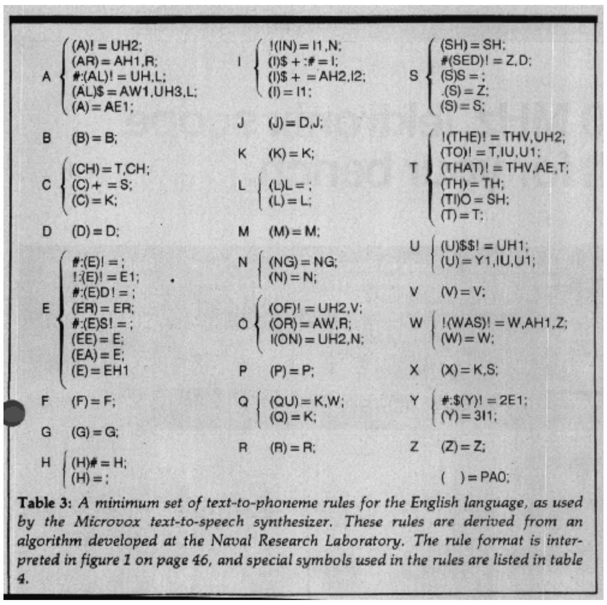
START
FIND FIRST CHARACTER IN ENGLISH STRING
FIND FIRST RULE FOR THIS CHARACTER
FIND NEXT RULE
DO ALL CHARACTERS WITHIN PARENTHESES MATCH CORRESPONDING CHARACTERS IN ENGLISH STRING?
?
NO
YES
DOES RULE FOR LEFT CONTEXT MATCH CORRESPONDING CHARACTERS IN ENGLISH STRING?
?
NO
YES
DOES RULE FOR RIGHT CONTEXT MATCH CORRESPONDING CHARACTERS IN ENGLISH STRING?
?
NO
YES
APPEND PHONEME CODES FROM RULE TO PHONEME BUFFER
LOOK FOR NEXT UNMATCHED CHARACTER IN ENGLISH STRING
NO
?
HAVE WE FINISHED THE STRING?
YES
DONE

Figure 2: *Flowchart of the text-to-speech algorithm used by the Microvox, which employs the rules of table 3.*

- Letter-to-Phoneme Rules, continued

- A minimum set of rules for English text-to-speech conversion is shown in table 3 on page 48. With these rules, it should be possible to achieve intelligible, albeit less than perfect, speech.Each rule in table 3 supplies a pronunciation for the character string enclosed in parentheses; each parenthetic string may also have a right context and a left context, as shown in figure 1 on page 46. The algorithm for interpreting the rule expressions is as follows.
- The processor recognizes the first character of the input plain-English text string; it skips down the list to the first applicable rule (one that contains the character in question as the first character in the parenthetic string) and attempts to match the rule's parenthetic string to the input text. If there is no match, the process is repeated with the next rule applicable for the letter. If there is a match on the parenthetic string, an attempt is made to match first the left and then the right context. If either context match fails, the processor proceeds to the next rule. The final rule for each letter contains a parenthetic string of just the letter with no left or right context, thus guaranteeing an eventual match for any letter in the input text. Once a match has been achieved, the phoneme codes invoked by the rule (shown to the right of the equal sign in the rule expression) are transferred to a phoneme buffer. Note that some rules invoke no phonemes.

Table 3: A minimum set of text-to-phoneme rules for the English language, as used by the Microvox text-to-speech synthesiser. These rules are derived from an algorithm developed at the Naval Research Laboratory. The rule format is interpreted in figure 1 on page 46 and special symbols used in the rules are listed in table 4.

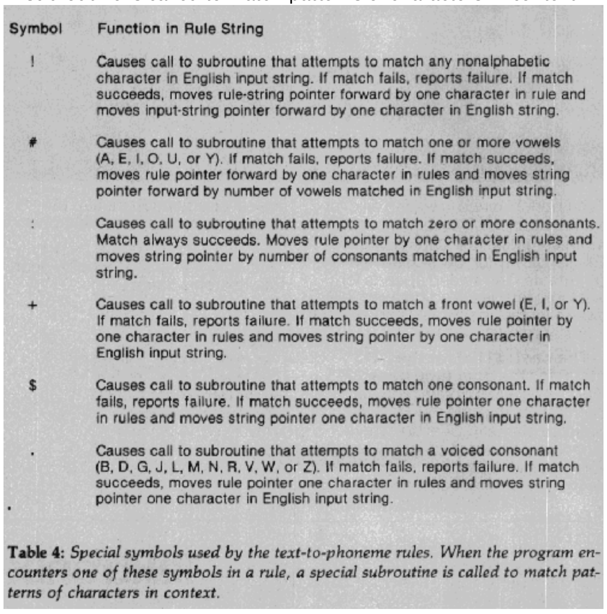


A	(A)! = UH2;
	(AR) = AH1,R;
	#:(AL)! = UH,L;
	(AL)$ = AW1,UH3,L;
	(A) = AE1;
B	(B) = B;
C	(CH) = T,CH;
	(C)+ = S;
	(C) = K;
D	(D) = D;
E	#:(E)! = ;
	!:(E)! = E1;
	#:(E)D! = ;
	(ER) = ER;
	#:(E)S! = ;
	(EE) = E;
	(EA) = E;
	(E) = EH1
F	(F) = F;
G	(G) = G;
H	(H)# = H;
	(H) = ;
I	!(IN) = I1,N;
	(I)$ + :# = I;
	(I)$ + = AH2,I2;
	(I) = I1;
J	(J) = D,J;
K	(K) = K;
L	(L)L = ;
	(L) = L;
M	(M) = M;
N	(NG) = NG;
	(N) = N;
O	(OF)! = UH2,V;
	(OR) = AW,R;
	!(ON) = UH2,N;
P	(P) = P;
Q	(QU) = K,W;
	(Q) = K;
R	(R) = R;
S	(SH) = SH;
	#(SED)! = Z,D;
	(S)S = ;
	.(S) = Z;
	(S) = S;
T	!(THE)! = THV,UH2;
	(TO)! = T,IU,U1;
	(THAT)! = THV,AE,T;
	(TH) = TH;
	(TI)O = SH;
	(T) = T;
U	(U)$$! = UH1;
	(U) = Y1,IU,U1;
V	(V) = V;
W	!(WAS)! = W,AH1,Z;
	(W) = W;
X	(X) = K,S;
Y	#:$(Y)! = 2E1;
	(Y) = 3I1;
Z	(Z) = Z;
	( ) = PA0;

Table 3: *A minimum set of text-to-phoneme rules for the English language, as used by the Microvox text-to-speech synthesizer. These rules are derived from an algorithm developed at the Naval Research Laboratory. The rule format is interpreted in figure 1 on page 46, and special symbols used in the rules are listed in table 4.*

- During the attempt to match a character string, the processor may encounter a special symbol (such as "#", ":", or "!") in the rule expression. In such a case, the symbol is looked up in a table in memory, and the corresponding subroutine, one of several listed in table 4, is called. For instance, the symbol "$" calls a subroutine that tries to match any single consonant. After a successful match of one consonant, the rule pointer moves to the next character in the rule, and the inputstring pointer moves to the next character in the input string. If the rule pointer encounters a "#", a similar matching subroutine for vowels is invoked. When a matching attempt of this type fails, the subroutine reports failure, and the processor skips to the next rule.

Table 4: Special symbols used by the text-to-phoneme rules. When the program encounters one of these symbols in a rule,a special subroutine is called to match patterns of characters in context.



Symbol	Function in Rule String
!	Causes call to subroutine that attempts to match any nonalphabetic character in English input string. If match fails, reports failure. If match succeeds, moves rule-string pointer forward by one character in rule and moves input-string pointer forward by one character in English string.
#	Causes call to subroutine that attempts to match one or more vowels (A, E, I, O, U, or Y). If match fails, reports failure. If match succeeds, moves rule pointer forward by one character in rules and moves string pointer forward by number of vowels matched in English input string.
:	Causes call to subroutine that attempts to match zero or more consonants. Match always succeeds. Moves rule pointer by one character in rules and moves string pointer by number of consonants matched in English input string.
+	Causes call to subroutine that attempts to match a front vowel (E, I, or Y). If match fails, reports failure. If match succeeds, moves rule pointer by one character in rules and moves string pointer by one character in English input string.
$	Causes call to subroutine that attempts to match one consonant. If match fails, reports failure. If match succeeds, moves rule pointer one character in rules and moves string pointer one character in English input string.
.	Causes call to subroutine that attempts to match a voiced consonant (B, D, G, J, L, M, N, R, V, W, or Z). If match fails, reports failure. If match succeeds, moves rule pointer one character in rules and moves string pointer one character in English input string.

Table 4: *Special symbols used by the text-to-phoneme rules. When the program encounters one of these symbols in a rule, a special subroutine is called to match patterns of characters in context.*

- How to Use the Algorithm

- The operation of the text-to-speech algorithm can best be illustrated by following the translation of a specific phrase into the Votrax phonemes listed in table 2. Our example will be the phrase" the national debt".
- Following the algorithm and its rules, matching uppercase and lowercase letters identically, we begin to find the pronunciation of this phrase by translating the initial blank to the short silent phoneme represented by the mnemonic PAO. Then the "t" of "the" leads us to the rules for the letter T. The first T rule's parenthetic string exactly matches "the", and the exclamation point ("!") symbols for right and left context match the spaces on each side of "the" in the English input text; therefore, we add the phoneme codes THV and UH2 to the list of phonemes to be spoken, which are stored in the phoneme buffer. The space after "the" in the English text becomes PA0.
- We then come to the "n" of "national", which sends us to the rules for the letter N. Although matching fails for the "NG" rule, it succeeds for "N". We then add the phoneme N to the output buffer. Next, we proceed to the letter-A rules. The first A rule is not matched because the 'a in national" is not followed by a nonalphabetic character, as demanded by the exclamation point. The next three rules also fail to match because a is not followed by "r" or "l". The final rule does match, and we add AE1 to the phoneme buffer.
- We now return to the T rules, matching "ti" with "o" as the right context and adding the phoneme SH to the buffer. We consult the O rules, matching "on" with "i" as the left context; we place the phonemes UH2 and N in the phoneme buffer.
- Now we are up to the second "a" in "national". We return to the rules for the letter A. The first rule fails because the letter we are trying to match is not followed by a nonalphabetic character. The second rule fails because the second "a" in "national" is not followed by an "r".

- However, the third rule succeeds. The "al in national" matches the "AL" in the rule. Checking the left context, moving from right to left, we first encounter a colon (":"), which means we must match zero or more consonants to the left of the "al". We match "n" and then proceed leftward to a number sign ("#"), which means we must match one or more vowels. This we do with "io". When we check the right context, we find that "al" is indeed followed by a nonalphabetic character, satisfying the rule's exclamation point. The rule thus succeeds and we transfer UH and L to the phoneme buffer. Last, we translate "debt", each character matching on its last rule.
- We end up with the following phonemes in the buffer: PA0, THV, UH2, PA0, N, AE1, SH, UH2, N, UH, L, PA0, D, EH1, B, T, and PA0. Except for the inclusion of the B phoneme for the normally silent "b" in "debt", this is a good translation. Only a much larger set of rules would contain a rule to handle the silent "b". If you have control over the input English text, you could change the spelling of "debt" to "det" and avoid the offending phoneme.

- intonation

- Providing realistic intonation is much more difficult than choosing the correct phonemes. Most intonation patterns are not represented in English spelling. Achieving the proper intonation may require grammatical parsing of a sentence or even knowing the writer's state of mind. Probably the best that can be done short of very detailed analysis is to use the algorithm developed by Bruce Sherwood (see reference 9), which involves raising the pitch on stressed syllables, raising it at the start of sentences and before commas, and lowering the pitch before the period at the end of a sentence. Before a question mark, the pitch is raised, unless the sentence begins with a question word (who, what, when, where, etc.), in which case it is lowered.

- Punctuation and Abbreviations

- Punctuation and abbreviations can also be converted into words and pronounced by the text-to-speech algorithm. A simple rule that works for many abbreviations is to pronounce the individual letters in an abbreviation consisting entirely of consonants and to pronounce abbreviations containing vowels as words. This rule works well for the names of some computer companies such as CDC and DEC. Unfortunately, it fails miserably for IBM.

- Operator Interaction

- The Microvox is a stand-alone intelligent peripheral device that converts ASCII-character text into spoken English. The Microvox is attached to the source of ASCII text (a computer, terminal, or modem) through either a serial or parallel communication link. Operation of the Microvox is similar to that of a printer except that the output consists of sounds instead of black marks on paper.
- The Microvox has many selectable function options that make possible a high level of intelligibility in many different applications. These options are activated by devicecontrol codes transmitted to the Microvox along with the text.
  In general, Microvox control codes are in the following form:
-

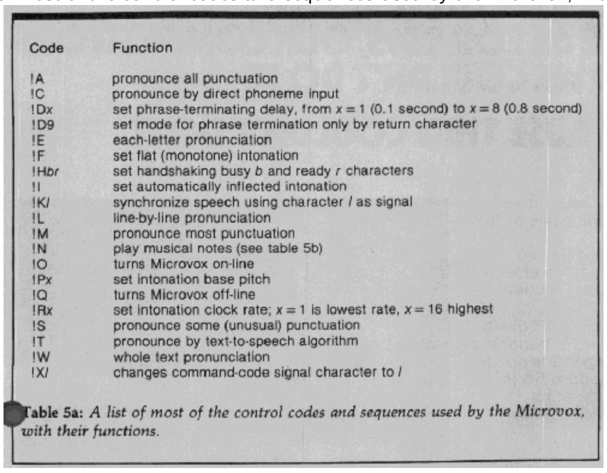
```
!(letter)(option)(option)
```

For example:

```
!D3
```

Most of the control codes are listed in table 5.

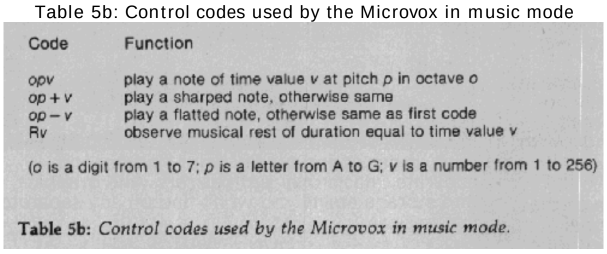
Table 5a: A list of most of the control codes and sequences used by the Microvox, with their functions



Code	Function
!A	pronounce all punctuation
!C	pronounce by direct phoneme input
!D*x*	set phrase-terminating delay, from $x = 1$ (0.1 second) to $x = 8$ (0.8 second)
!D9	set mode for phrase termination only by return character
!E	each-letter pronunciation
!F	set flat (monotone) intonation
!H*br*	set handshaking busy *b* and ready *r* characters
!I	set automatically inflected intonation
!K*l*	synchronize speech using character *l* as signal
!L	line-by-line pronunciation
!M	pronounce most punctuation
!N	play musical notes (see table 5b)
!O	turns Microvox on-line
!P*x*	set intonation base pitch
!Q	turns Microvox off-line
!R*x*	set intonation clock rate; $x = 1$ is lowest rate, $x = 16$ highest
!S	pronounce some (unusual) punctuation
!T	pronounce by text-to-speech algorithm
!W	whole text pronunciation
!X*l*	changes command-code signal character to *l*

Table 5a: *A list of most of the control codes and sequences used by the Microvox, with their functions.*

Table 5b: Control codes used by the Microvox in music mode



Code	Function
opv	play a note of time value *v* at pitch *p* in octave *o*
op + *v*	play a sharped note, otherwise same
op − *v*	play a flatted note, otherwise same as first code
R*v*	observe musical rest of duration equal to time value *v*

(*o* is a digit from 1 to 7; *p* is a letter from A to G; *v* is a number from 1 to 256)

Table 5b: *Control codes used by the Microvox in music mode.*

- The exclamation point is a signal to the Microvox that a control code follows. If you wish, you can set it up to use any other character as the control-code signal. This is done by giving the following instruction:
(old signal character)X(new signal character)
- For example:

  !X$

  which changes the control signal from an exclamation point to a dollar sign. From this state of affairs, the command

  $X!

  wil change the control signal from the dollar sign back to te exclamation point. Device-control codes can be embedded anywhere in the text transmission; they are not spoken.

- Device-Control: Handshaking

- If a standard parallel or an RS-232C serial connection is used, the sending computer hardware can detect and examine the ACK (Acknowledge) or RTS (Ready to Send) signal to determine whether the Microvox is ready to receive a character - However, many popular microcomputers lack the hardware to detect the RTS handshaking signal. Furthermore, the RTS signal cannot be used for this purpose if the communication path includes a modem / telephone link. In the Microvox, special software-handshaking signals, described below, are provided to control the flow of input text. (In general, hardware handshaking through RTS or ACK is preferable whenever possible, because it relieves the host computer's processor of the handshaking chore and allows use of higher data rates.)
- Software handshaking is activated by setting switch section 3 of DIP (dual-inline pin) switch SW1 on the Microvox's circuit board to the closed position. (The open position allows hardware handshaking.) The particular characters that the host computer should recognize may be selected by the command
- !H(busy character)(ready character)

  For example:

  !H@#
- After receiving this command, the Microvox will send the at-sign character to the computer when it is unable to receive more data; it will send the number sign to the computer when it is again ready to receive data. It is the responsibility of the host-computer programmer to write the software necessary to use software handshaking.
Finally, it is possible to use the Microvox with no handshaking by simply invoking the software-hand-shaking mode and ignoring the hand-shaking transmissions. In this case, you must insert timing delays in the text-transmitting program so that data will not be sent to the Microvox faster than it can handle.

- Text Synchronization

- For many applications, it is important to synchronize the output speech with other outputs from the computer, such as text or graphics appearing on the display screen. For instance, an instructional program may require placing a picture on the screen when certain speech output begins and placing a question mark on the screen when the speech ends. For synchronization, the following command may be used:
- !K(synchronization character)

  For example:

  !K#John!K%Marsha!K$
- After receiving this text string, the Microvox will send a "#" back to the computer just before starting to say "John"; it will send a "%" to the computer just after saying 'John" and just before starting to say "Marsha"; and it will send a "$" character to the screen just after saying 'Marsha". None of these special synchronization characters will be spoken. It is the programmer's responsibility to use the incoming synchronization characters to coordinate the screen display with the speech.

- Phrase Termination

- Many aspects of English pronunciation are controlled by the context in which a given letter or word is spoken. For this reason, the Microvox can wait to receive a complete phrase before translating from text to speech. If you don't specify otherwise, the Microvox will wait to translate a phrase until it has received one of the following phrase-terminating characters:
- 1. a period followed by two spaces or a return character
  2. a comma, semicolon, colon, exclamation point, or question mark followed by a space or return
  3. a return character
- For some types of output, such as computer programs or poems, you would want each line read as a separate phrase. For others, such as ordinary English narrative text, you may not want a return character to terminate a phrase. You have two options to deal with this situation.
- The command "!W" means "whole-text pronunciation." If this option is selected, a return character will not terminate a phrase unless one of the conditions of rule 1 or 2 above is fulfilled.
- The command "!L" means "line-by-line pronunciation." If this option is selected, a return character will always be treated by the Microvox as terminating a phrase. When the Microvox is first turned on, it is in the line-by-line mode.
- Rather than always send a special signal to terminate a phrase, you may wish to have the Microvox treat a phrase as terminated if a certain delay occurs without any phrase terminator being received. Possible applications of this option include situations where the user does not fully control the output. For instance, suppose the Microvox is passively connected to a transmitting device that doesn't send any of the terminating characters listed above (maybe it sends "STOP" instead). In such a case, there is no way to insert phrase-termination characters in the output stream. However, if the Microvox is set to treat a half-second delay without receipt of information as the end of a phrase, computer output will not be lost or ignored.
- The following option provides timed-delay phrase termination:

  !D(delay time)

- The delay parameter, which ranges from 1 to 8, varies the delay from 0.1 second to 0.8 second. (If too short a delay is used, a phrase may be translated in pieces, resulting in odd intonation or pronunciation, because the Microvox uses the context of letters and words to determine their pronunciation.)
- The command !D9 is a different case; it makes the Microvox wait for a phrase-terminating character even if it has to wait forever. (This is the default mode.) Generally speaking, !D9 should be used with slow data sources such as a keyboard.
- This selectable-delay feature is particularly useful for the visually disabled. It can allow a blind programmer to use a standard unintelligent terminal by connecting the Microvox to receive the output from both the user and the computer. If the delay is set to 0.1 second, keys pressed by the user will be echoed as spelled letters (because the slight delay between them will be treated as an end of phrase), but output generated by the computer will be spoken as complete lines because there generally will be no significant delay between characters. The delay can be varied to fit the particular application.

- Intonation

- The pitch at which individual phonemes are pronounced can be controlled automatically by the text-to-speech algorithm, be kept fixed, or be altered by user command. Some of you will prefer automatic inflection, because of the variety it gives to the speech, even though the inflection is often not accurate. Others think a computer should sound like a computer and will prefer flat speech. Still others may wish to experiment with controlling the pitch to optimize intelligibility. This control can extend to even make the Microvox sing.
- The hardware in the Microvox allows control of pitch in two different ways. The Votrax SC-01A speech-synthesis integrated circuit has four selectable pitch levels. In addition, the output pitch can be varied by selecting 1 of 16 different rates for the clock signal fed into the SC-01A. When the Microvox is first turned on, the synthesizer chip is set to pitch level 1 (low) and base speech rate 5 (defined below). The intonation is generated by an algorithm that selects an appropriate clock rate for each phoneme.
-

  To turn on or off the automatic clock-rate setting, you can send the command

  !F

  (which stands for flat intonation), and the output rate will stay at the base rate. To restore automatic clock-rate variation, you can send the command

  !I

  which stands for inflected intonation (by algorithm).

  The intonation algorithm adds to or subtracts from the base rate to derive the final voice pitch. Using the !I mode, however, limits output to only four base-rate pitch-level shifts. You may decide to operate without automatic inflection on all text-to-speech translation and yet desire to add certain pitch changes on specific words or phonemes. This can be easily done on the Microvox, because the base pitch and clock rate can be controlled independently and changed at any time.

  The control code is of the form

  !Px

  where x is a digit from 1 through 4; x = 1 selects the lowest pitch with pitch increasing according to the value of x.

  You may also decide to control the clock base rate with a command of the form

  !Rx

  where x =1 yields the slowest rate

  and lowest level for the given base pitch, and x = 16 yields the fastest base rate. The text examples to follow will demonstrate this function.

- Punctuation Modes

- The Microvox has three modes for pronouncing punctuation. The user options are:
-

  !A (all mode-all punctuation pronounced)

  !M (most mode-all punctuation pronounced except return, linefeed, and space)

  !S (some mode-only unusual punctuation pronounced)
- When the Microvox is turned on, it is in "some" mode.

- Major-Mode Options

- The Microvox can operate in four different major modes: text-to-speech, text-to-spelled-speech (pronouncing each letter), phoneme-code, and music. When the Microvox is turned on, it begins in text-to-speech mode.

- Text-to-Speech Mode

- In the text-to-speech mode, selected by the !T command, the Microvox uses the algorithm previously described to attempt correct pronunciation of all phrases sent to it. However, no program of reasonable size can possibly contain all the rules and exceptions for the pronunciation of English. Moreover, since the Microvox lacks understanding of the text, it cannot tell which of two homographs is intended. For instance, when the text contains the word "read", the Microvox cannot know if the present or the past tense is meant.
- When you must have the expected pronunciation, you can modify the spelling. By typing "red" or "reed" instead of "read", you can be sure to get the pronunciation you want. If "hic-cough" is pronounced strangely, try "hiccup". Often, it helps to break a word into syllables. Compare the pronunciation of "typewriter" and "type write er". Getting recognizable renditions of foreign words will require considerable ingenuity, because the Microvox works on the principles of English pronunciation. Compare "parlez vous" and "parlay voo".

- Spelled Speech

- The spelled-speech mode, activated by the !S command, is useful for abbreviations and words that a user might have difficulty in understanding. When this option is selected, every letter is pronounced separately. It is often useful to use the IA punctuation mode in conjunction with the spelled-speech mode, so that all punctuation is also pronounced.

- Phoneme-Code Mode

- The Microvox can also accept input in the form of Votrax phoneme codes (see table 2). A space must be lef t between the phoneme mnemonics. For example, the input string
- !C AE N D PA0 THV UH2 PA0 S E PA0 I Z PA0 B O1 AY I3 L I NG PA0 H AH T PA1
- will cause the Microvox to say "and the sea is boiling hot."
- Either the flat- or automatic intonation mode can be used with phoneme-code input. If the automatic intonation is off, the output pitch will correspond to the base rate. If it is on, intonation will be like that for the equivalent text. If there are erroneous phoneme codes, the erroneous mnemonies will be spoken as if they were text. Pitch and rate codes can be mixed with phoneme codes to produce singing.

- Music Mode

- Since last month's article was written, a few changes for increased overall capability have been made to the Microvox's controlling software. As a result, the music mode now works in the following manner.
- The music mode is turned on by the command !N. The notation shown in table 5b is used. The seven playable octaves centered about middle C are indicated by numbers from 1 to 7. Each octave contains notes identified as A, B, C, D, E, F, orG. Sharps are indicated by the suffix character "+", flats by "-". Time values are selected by numeric arguments used as multipliers to an internal time constant; the arguments' values may range from 1 to 256. Rests are indicated by 'R". When in the music mode, sending Microvox the character string "3F + 4" causes it to play a four-period-long note at a pitch of F sharp in the third octave. "R16" causes a 16-period-long rest. The separate tempo command had to be omitted.

- Default Options

- When the Microvox is turned on, certain default conditions are in force. They are equivalent to entering the following commands:
-

```
!P1 lowest pitch
!R5 clock base rate 5
!F flat intonation
!T text-to-speech mode
!S some punctuation
!L line-by-line pronunciation
!D9 wait for return
!O Microvox on-line
!H@# handshake with @ and #
```

- Any of these options can be changed simply by sending the proper control code to the Microvox, either transmitted separately or embedded in text. For example, if the Microvox is connected to a terminal, typing "This is a test." and hitting Return will result in that phrase being spoken with no intonation.To add automatic intonation, you can type !I This is a test. Return. From this point on, all spoken text will have automatic inflection unless you resume flat intonation by typing As previously mentioned, intonation can be added selectively or by the automatic algorithm.
- Let's look at four ways of commanding the Microvox to pronounce the same sentence:
-

```
1. (text-to-speech mode, no added inflection)
!T!F
Please enter your access number.

2. (automatic inflection in text-to-speech mode)
!T!I
Please enter your access number.

3. (selected inflection in text-to-speech mode)
!T!F!P1!R5
Please !R8en!R5ter !R7yor !R5access number.

4. (phoneme-input mode with selected intonation)
!F!C!P1!R5
P L E1 Y Z PAl PAl PAl PAl
!R9 EH1 EH3 N !R5 T ER PAl
Y !R8 O2 O2 O2 !R5 R PAl
!R7 AE1 !R5 K S EH1 EH3 S PAl
N UH1 M B ER
```

- These examples demonstrate various ways in which you can increase the intelligibility of the synthesized speech by programming the Microvox. You can use the text-to-speech mode with either selective or automatic intonation, or you can optimize pronunciation by choosing exactly the pitches and phonemes you wish.
- An exaggerated example of combined pitch and phoneme control can actually allow Microvox to sing, as demonstrated by a bar of happy Birthday":
-

```
!C !P3 !R3
H H H AEl AEl AEl AEl AEl AEl P P !P2!R5 Y Y Y
```

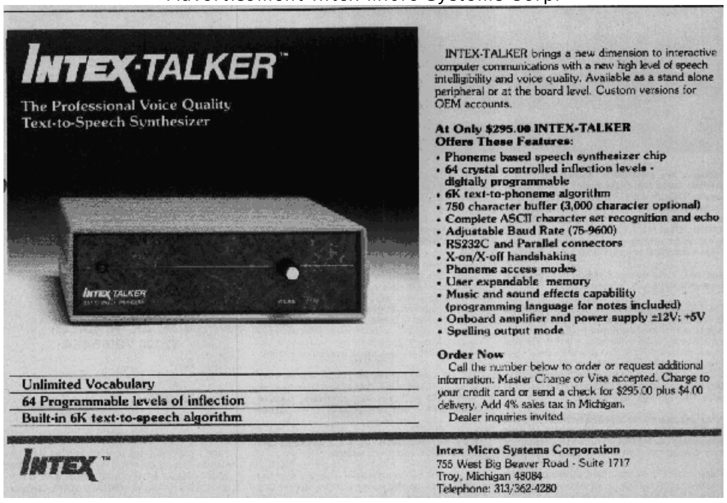
```
!P3!R5 B ER ER ER ER R TH TH TH TH !Rl D Al Al Al Al 13
!R9 T IU IU IU IU Ul Ul Ul Ul Ul !R7 Yl IU IU IU
Ul Ul Ul Ul Ul Ul
```

- and a scale of D through E:
-

```
!C
!P1 !Rl D D El El Y Y Y
!P1 !R5 El El El Y Y Y
!P1 !R11 EHl EHl EHl EH2 F F F
!P2 !R5 D J J El El Y Y Y
!P2 !R11 Al Al Al Al Al Y
!P2 !R14 B B El El Y Y Y
!P3 !R11 S S El El Y Y Y
!P3 !R15 D D El E Y Y Y
!P4 !R11 El El El Y Y Y
```

- Since there are only 64 pitch levels, which were set up for speaking rather than singing, the range of octaves is somewhat limited. Singing is probably the most dramatic example of programmable pitch control, but Pavarotti doesn't have to worry about his job.

Advertisement Intex Micro Systems Corp.



INTEX·TALKER™

The Professional Voice Quality Text-to-Speech Synthesizer

INTEX-TALKER brings a new dimension to interactive computer communications with a new high level of speech intelligibility and voice quality. Available as a stand alone peripheral or at the board level. Custom versions for OEM accounts.

At Only $295.00 INTEX-TALKER Offers These Features:

- **Phoneme based speech synthesizer chip**
- **64 crystal controlled inflection levels - digitally programmable**
- **6K text-to-phoneme algorithm**
- **750 character buffer (3,000 character optional)**
- **Complete ASCII character set recognition and echo**
- **Adjustable Baud Rate (75-9600)**
- **RS232C and Parallel connectors**
- **X-on/X-off handshaking**
- **Phoneme access modes**
- **User expandable memory**
- **Music and sound effects capability (programming language for notes included)**
- **Onboard amplifier and power supply ±12V; +5V**
- **Spelling output mode**

Order Now

Call the number below to order or request additional information. Master Charge or Visa accepted. Charge to your credit card or send a check for $295.00 plus $4.00 delivery. Add 4% sales tax in Michigan.

Dealer inquiries invited

Unlimited Vocabulary

64 Programmable levels of inflection

Built-in 6K text-to-speech algorithm

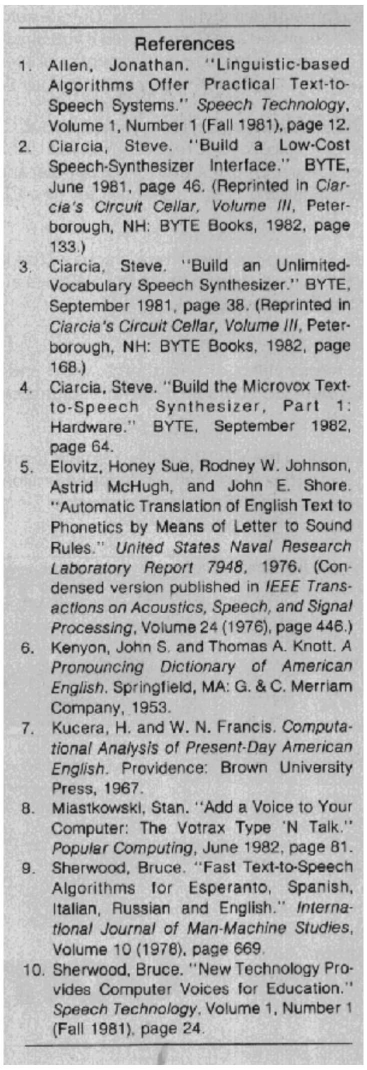
INTEX™

Intex Micro Systems Corporation
755 West Big Beaver Road - Suite 1717
Troy, Michigan 48084
Telephone: 313/362-4280

- In Conclusion

- Speech synthesizers raise the level of communication between man and machine. Today, they are regularly used in telephone-answering Systems, elevators, fire-alarm systems, annunciators, and nonvisual-commumcation aids. The price/performance ratio of voice synthesizers no longer limits their uses.
- The Microvox is a second-generation voice synthesizer with many professional features. Nothing is sealed from inspection, and the schematic diagram isn't kept in a vault some-place. Circuit Cellar projects are meant to be built and enjoyed.
- If you had asked me four years agc why anyone would spend money on a speech synthesizer, I wouldn't have had an easy answer. Today, how ever, after designing four speech synthesisers, and reading hundreds of readers' letters each month, I've come to regard speech synthesis as a new technology that's only begun to be used to its full potential.

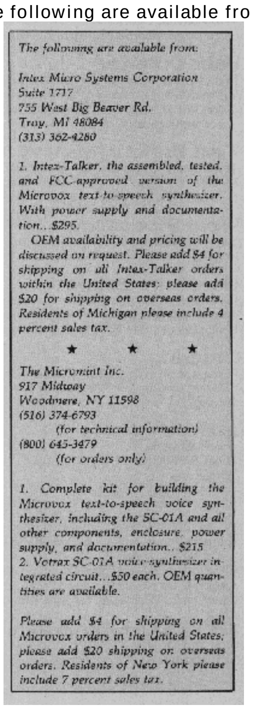
References



References

1. Allen, Jonathan. "Linguistic-based Algorithms Offer Practical Text-to-Speech Systems." *Speech Technology*, Volume 1, Number 1 (Fall 1981), page 12.
2. Ciarcia, Steve. "Build a Low-Cost Speech-Synthesizer Interface." BYTE, June 1981, page 46. (Reprinted in *Ciarcia's Circuit Cellar, Volume III*, Peterborough, NH: BYTE Books, 1982, page 133.)
3. Ciarcia, Steve. "Build an Unlimited-Vocabulary Speech Synthesizer." BYTE, September 1981, page 38. (Reprinted in *Ciarcia's Circuit Cellar, Volume III*, Peterborough, NH: BYTE Books, 1982, page 168.)
4. Ciarcia, Steve. "Build the Microvox Text-to-Speech Synthesizer, Part 1: Hardware." BYTE, September 1982, page 64.
5. Elovitz, Honey Sue, Rodney W. Johnson, Astrid McHugh, and John E. Shore. "Automatic Translation of English Text to Phonetics by Means of Letter to Sound Rules." *United States Naval Research Laboratory Report 7948*, 1976. (Condensed version published in *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Volume 24 (1976), page 446.)
6. Kenyon, John S. and Thomas A. Knott. *A Pronouncing Dictionary of American English*. Springfield, MA: G. & C. Merriam Company, 1953.
7. Kucera, H. and W. N. Francis. *Computational Analysis of Present-Day American English*. Providence: Brown University Press, 1967.
8. Miastkowski, Stan. "Add a Voice to Your Computer: The Votrax Type 'N Talk." *Popular Computing*, June 1982, page 81.
9. Sherwood, Bruce. "Fast Text-to-Speech Algorithms for Esperanto, Spanish, Italian, Russian and English." *International Journal of Man-Machine Studies*, Volume 10 (1978), page 669.
10. Sherwood, Bruce. "New Technology Provides Computer Voices for Education." *Speech Technology*, Volume 1, Number 1 (Fall 1981), page 24.

the following are available from:



The following are available from:

Intex Micro Systems Corporation
Suite 1717
755 West Big Beaver Rd.
Troy, MI 48084
(313) 362-4280

1. Intex-Talker, the assembled, tested, and FCC-approved version of the Microvox text-to-speech synthesizer. With power supply and documentation...$295.

OEM availability and pricing will be discussed on request. Please add $4 for shipping on all Intex-Talker orders within the United States; please add $20 for shipping on overseas orders. Residents of Michigan please include 4 percent sales tax.

★ ★ ★

The Micromint Inc.
917 Midway
Woodmere, NY 11598
(516) 374-6793
(for technical information)
(800) 645-3479
(for orders only)

1. Complete kit for building the Microvox text-to-speech voice synthesizer, including the SC-01A and all other components, enclosure, power supply, and documentation...$215
2. Votrax SC-01A voice synthesizer integrated circuit...$50 each. OEM quantities are available.

Please add $4 for shipping on all Microvox orders in the United States; please add $20 shipping on overseas orders. Residents of New York please include 7 percent sales tax.

- posted by RDapril2001