SYNTHETIC SPEECH

FOR PERSONAL COMPUTERS

by
Joseph E. Hébert

date
5 August, 1991

project supervisor
Mr. Jack Burson

faculty committee
Mr. Jack Burson
Mr. Wilbur H. Gruver
Dr. Russell L. Heiserman

Abstract

This document accompanies the project outlined within. It was prepared for the course TECHNOLOGY 420, a required course for the completion of a Bachelor of Science degree in the Department of Electronics Technology, College of Engineering and Technology, at McNeese State University.

The text of this report is organized into chapters as prescribed by the course requirements. Some chapters are subdivided into Sections and/or Paragraphs as deemed necessary by the author. Note that, as it applies to the organizational structure of this document, the term 'Paragraph' refers to the lessermost subdivision of the textual content. This is similar to, but distinct from, the traditional definition of the term.

The INTRODUCTION chapter discusses the development of computer technology as it relates to the individual user. A historical review and a projection of near-future developments lend consideration to the potential impact of synthetic speech on the computer user.

The DESIGN OBJECTIVES chapter prescribes the author's intentions pertinent to this project.

The RESEARCH OF LITERATURE chapter discusses the considerations necessary for the implementation of synthetic speech, as well as the details of the chip-set chosen for this project and how they address the necessary considerations.

The CIRCUIT DESIGN chapter discusses the considerations made in choosing the circuitry components and configurations implemented in this project.

The THEORY OF OPERATION chapter discusses the operation of the unit built for this project. An overview of the unit as a system is given prior to discussing the details of the various sub-circuits.

The TESTING AND RESULTS chapter discusses each of the design problems encountered in the development of this project and, when applicable, how they were corrected.

The FINAL ASSEMBLY chapter details the finished unit in terms of how it is housed, as well as how interface connections are provided.

The CONCLUSIONS AND RECOMMENDATIONS chapter discusses the performance of the finished unit relative to considerations which should be made when configuring the unit for a given

application. Also discussed are techniques for constructing double sided printed circuit boards and silkscreen printing. This chapter also details the cost of the project.

The APPENDICES contain a bibliographical listing of the technical literature referred to in this report, a complete set of schematics, all trace patterns and component overlays, and a source code listing of the special OrCad® component library compiled for this project.

Registered Trademarks

Apple® and MacIntosh® are registered trademarks of Apple Computers, Inc.

AutoCad® is a registered trademark of AutoDesk, Inc.

AutoTrax® is a registered trademark of Protel, Inc.

CSAM® is a registered trademark of Computer Support Corporation, Inc.

General Instruments® is a registered trademark of General Instruments, Inc.

IBM®, AS/400®, and PS/2® are registered trademarks of International Business Machines, Inc.

Intel® is a registered trademark of Intel, Inc.

Microchip® is a registered trademark of Microchip, Inc.

Motorola® is a registered trademark of Motorola, Inc.

MS-DOS® is a registered trademark of Microsoft Corporation.

OrCad® is a registered trademark of OrCad Systems Corporation.

QAPlus® is a registered trademark of DiagSoft, Inc.

Tandy® and Radio Shack® are registered trademarks of Tandy, Inc.

Texas Instruments® and TI99-A® are registered trademarks of Texas Instruments, Inc.

Toshiba® is a registered trademark of Toshiba American, Inc.

Ulano® and Blue Poly® are registered trademarks of Ulano Corporation.

WordPerfect® is a registered trademark of WordPerfect Corporation.

Zilog® is a registered trademark of Zilog, Inc.

Acknowledgments

Though I did not have any collaborators in this project, I do feel compelled to extend thanks to several people who made it possible for me to be where I am today.

First, I owe a debt of thanks to Mr. Charles Prejean. Four years ago, I walked into Mr. Prejean's office and announced that I wanted to earn two degrees, one in physics and one in electronics. As if to redefine the term *audacity*, I told Mr. Prejean that I had only a ninth-grade education and a GED, that my ultimate goal was to earn my doctorate in physics, (this despite having never had a physics course), and the only way I could afford this dream was with financial assistance. I then asked Mr. Prejean if he could arrange a scholarship, and in the course of a single phone call it was done. I want to thank Mr. Prejean for abandoning reason and making that phone call.

This brings up my second debt of thanks. Were it not for the philanthropy of H.C. Drew, and the scholarship that bears his name, I truly would not have been afforded the opportunity to pursue my dream.

But the road from there to here was not as easy as the one phone call that made it possible. It has taken much work, not only on my part but on the parts of many. I am referring to part of that group of individuals who have devoted their lives to helping those of us who have chosen to pursue the dream of an education; I am referring to my instructors. During the past three and a half years I have benefitted from the efforts of many instructors, and I would like to thank them all.

Finally, I must thank God for bringing me to the point I have reached. Were it not His will, I could never have come this far, nor would I have had reason to.

Table of Contents

Table of Illustrations

Table of Tables

Introduction

The focus of this research project is the implementation of synthetic speech for personal computers. Though the focus here is on MS-DOS® based computers, it is important to understand that the same principles discussed in this report will apply to any computer. While a MacIntosh® would require different hardware specifications, and a mainframe would require the use of EBCDIC text encoding instead of ASCII, the general principles will remain constant.

In addition to understanding the principles of synthetic speech, the reader would do well to understand the broader ramifications of the same. Toward that end, this report will begin by considering how the development of personal computers has led to the advent of synthetic speech in a *natural* way. Though the need for, and benefits of, synthetic speech may be intuitively obvious, a more deliberate consideration of the matter will be given here.

§ A Historical Overview §

Not many years ago there was a clear distinction between mainframe computers and personal, or desktop, computers. Size notwithstanding, personal computers were little more than toys. The TI99-A® came equipped with 8 kilobytes of system RAM, a keyboard, and little else. A standard television was required to serve as the CRT, and saving a program required the addition of a special cable and a cassette player/recorder. Though these *toys* were little more than novelties, they did serve to introduce the individual to computers.

But technology is never static, and computer technology is by no means an exception. Developments have made the distinction between mainframes and desktops much less significant, even blurred. With the advent of ULSI microchips, desktop computers are now afforded microprocessors capable of addressing RAM in the gigabyte range and memory chips capable of providing it. Fixed magnetic storage disks with capacities of hundreds of megabytes are common. Advanced operating systems allow this new-found capability to be exploited with true multitasking and networking.

Even the distinction of size has been blurred. IBM® now sells a mainframe, the AS/400®, which is only slightly larger than their PS/2® tower models.

Of course, this new-found power would be of no use without software, and companies have sprung up to meet the need. Packages are readily available that allow people to solve advanced mathematical equations, draw architectural designs, or manage anything from a rental property to a check book.

As is evidenced by this report, instead of typing a term paper a student can now publish one, complete with artwork and illustrations. This report was published with WordPerfect®, version 5.1, and the artwork was produced with AutoCad® and CSAM®. The schematics at the back of this report were drawn with OrCad®, and the trace pattern that eventually became the circuit board which is the subject of this project was produced with AutoTrax®.

§ People and Computers §

But what good is all of this capability if people can't use it? This has been a driving question throughout the development of computers. The industry coined a phrase in answer to that question; They call it making computers *user-friendly*. A secretary doesn't have to understand ASCII encoding to write a letter, a programmer doesn't have to understand two's-compliment arithmetic to make a computer do subtraction, and a systems analyst doesn't have to understand flip flops or transistors in order to address memory.

Instead of being concerned with the true nature of whatever aspect of the computer an individual is working with, it is often sufficient to think, or understand, in terms of an analogy. A perfect example is the programmer thinking of five volts as a one-bit, and zero volts as a zero-bit. Many programmers don't even know what a volt is.

One consequence of this is the perception high-end users have of the computer as a whole. Most have come to think of the computer as being analogous to intelligent. The majority of people who work with computers have even lost sight of the distinction between the hardware and software. A clerical worker key-entering a formula into a spreadsheet cell perceives that the computer simply *knows* how to perform the calculation. Of course, everyone knows that "computers are stupid," but that statement is becoming less and less obvious.

§ Artificial Intelligence §

For years now computer scientists have been researching artificial intelligence. The concept has become so commonplace that it is known simply by its initials, AI. And the reasoning behind the quest is sound. The less expertise that is required to use a computer, the more useful the computer will be. AI is the ultimate *user-friendliness*. But how far into the future is AI? The answer depends on how AI is defined. Consider the following definitions:

Intelligence: *1: The ability to learn or understand.*
2: Information.

Ignorance: *1: Lacking knowledge.*

Knowledge: *1: Understanding gained by experience.*
2: Range of information.

Learn: *1: Gain knowledge by study or experience.*

If intelligence is simply information, then computers are intelligent by design. Their purpose is to process and store information.

But that isn't what is meant by AI. At the software level, AI has existed for some time now, though it is still not commonplace. Database managers have been written that use NQL, or *natural query language*, (NL for short). Instead of utilizing a well-defined syntax, natural language software packages are programmed to pick out key terms in a command. As defined above, these packages learn. A given individual will tend to habitually ask the same questions in the same way. These habits are retained in and by the software for reference so that in time it will become more efficient.

In that some programs have the ability to gain knowledge through experience, AI exists today. Of course, it's still a matter of the software that has this quality, not the computer. Nonetheless, one key word that appears in the above definitions remains unsatisfied. Neither the computer, nor the software, has *understanding*. Understanding implies consciousness, and computers are simply machines. It is imperative to remember that the key word in *artificial intelligence* is *artificial*, not *intelligence*.

That being understood, one question remains. How closely can machine intelligence approximate human intelligence?

§ SYNTHETIC SPEECH §

Anyone who has ever seen an episode of Star Trek has seen the vision of an intelligent computer. The ship's computer has no keyboard or CRT. It is accessed by simply addressing it verbally. Output from the computer is channeled through the ship's intercom. So common is this image to Star Trek devotees that one of the movie sequels contained a comedic scene in which Scottie, (the Chief Engineer), attempted to access a twentieth century Macintosh by using the mouse as a microphone. Will four centuries make that much difference in our perceptions? How far into the future is verbal interaction between users and computers?

Directory assistance gives out phone numbers via a computerized voice. Many children have listened to Teddy Ruxpin's digital voice. Voice mail and automated telephone services use digitally recreated voices. These are all applications utilizing computerized voices as a means of outputting data to the user that are employed today.

But if these examples are considered examples of AI, then it is also necessary to include tape recorders and phonographs as examples of AI. Clearly that is not the case, nor are these examples of true AI. Each of these use a member of the Toshiba® TC88xxx family of voice recording devices. The speech output is computer controlled, but it first had to be recorded into the computer by a human voice. In order to consider computer generated speech as even an early development of AI, that speech should be completely computer generated. The vocal tract, (the word or phrase), should be constructed by the computer, not just replayed.

One measure that is used to judge the intelligence of humans is literacy[1], at least to the extent that the ability to read affords people a greater ability to learn. And this is the subject of this research project, to explore the current limitations of enabling personal computers to *read* written text. If a computer could construct and *speak* a vocal tract from a text file, with some degree of accuracy, that would be true synthetic speech as well as a much closer approximation of intelligence at the hardware level.

The technology for true synthetic speech exists today, as does the technology for speech recognition, at least in the early developmental stages. True speech synthesis involves dissecting verbal language into fundamental components, and then establishing a relationship with the textual, (or written), language. This allows the computer to generate vocal tracts which were never uttered by a human voice.

Far from being nothing more than the stuff of science fiction, there are real markets for such technology. It isn't necessary to develop the kind of verbal interaction that exists on the star ship Enterprise, or even full vocal interaction, to begin realizing benefits. Simply enabling a computer to output data verbally would prove very beneficial.

§ APPLICATIONS §

Perhaps the most obvious benefactors of such technology would be the visually impaired. With a high resolution scanner, a refined optical character recognition program, (OCR), and a

[1] The distinction between education and intelligence is fully acknowledged by the author at this point.

computer capable of synthetic speech, blind people could read their own mail, books, or magazines. There would be no need for special audio recordings of books, and braille could become a lost art.

But the blind certainly wouldn't be the only ones to benefit from such technology. Industry is just beginning to realize some of the hazards associated with computers and computer terminals. Apple® has developed a special CRT that limits the electromagnetic radiation it emits. Some cities are beginning to legislate the number of hours workers can spend at a computer terminal, and revising health codes to prescribe required lighting conditions. Other companies are trying to develop more ergonomically suitable keyboards and mice. This because of a noted increase in the incidence of carpal tunnel syndrome[2] in computer operators.

Consider a device such as the one outlined in this report. It is capable of accepting input in the form of ASCII text strings and converting it into verbal output. If output to the CRT was simultaneously sent to this device, typists and data entry clerks would no longer be dependent on their CRTs to detect their errors. This would enable them to simply turn off their CRTs for much, if not most, of their daily routine. Considering that such a device could be developed with hardware that is currently available and inexpensive, it would certainly be preferable to rewiring the office lighting to meet new health codes. It could also eliminate the need for additional employees to make up for reduced work hours.

From a more futuristic perspective, accommodating verbal input could all but eliminate the need for a keyboard. Could this be accomplished today? Perhaps not, but Toshiba® has introduced voice recognition devices with limited capabilities. The TC8861F/TC8862F/TMP80C50AU chip-set allows for up to 78 words, (a word can be any utterance limited to some maximum time duration), of recognition. This can be either speaker independent, or for the security conscious, speaker dependent. While this is not sufficient to eliminate the keyboard for data input, it would certainly be sufficient for verbal control of the operating system. Considering that the keyboard is the cause of the increase in carpal tunnel syndrome, and considering the expense of corrective surgeries and increased insurance premiums, it would likely be easy to justify financing the research needed to further this technology.

[2] Carpal Tunnel Syndrome: A disorder in which the nerve passing through the carpal tunnel, (the wrist), is inflamed by friction with the bone. It can result in varying degrees of disability, depending on the severity of the inflammation. Carpal tunnel syndrome is not self healing since the result of the stimulus, the inflammation, aggravates the stimulus which caused it. For this reason, carpal tunnel syndrome normally requires orthopedic surgery for correction.

Design Objectives

The project outlined in this report is designed around the General Instrument® chip-set, CTS256A-AL2 and SPO256-AL2. Both devices are sold at the retail level by Radio Shack®. This chip-set is manufactured for the purpose of accepting input in the form of ASCII text strings and converting it to verbal output.

The intended purpose of this project is primarily experimental in nature. Instead of designing the device to serve a single purpose, maximum flexibility, as specified in the provided technical literature, is being sought. If successful, the completed unit will be limited in function only by the limitations designed into the components. This will allow the unit to be utilized in a variety of ways, depending on the software developed for it.

The finished unit should allow for data transmission through either a standard parallel port, or through a standard RS232 serial port. UART parameters for serial communications will be switch selectable.

A 2 kilobyte RAM buffer will be incorporated into the unit's design to lessen the demand of the unit on the computer's microprocessor time. This is desirable since speech is inherently much slower than normal microprocessor operations.

Finally, 4 kilobytes of EPROM space will be provided. This will allow for the storage of *exception words*, or words which will not be spoken properly otherwise. A good example of this is the author's last name, Hébert.

Upon completion, the author intends to use the unit primarily for reading the READ.ME documentation that often accompanies software. This can be accomplished with existing MS-DOS® utilities, therefore it is not intended to develop any specialized software to accompany this project.

Research of Literature

§ THE NATURE OF SPEECH §

To understand the problems of generating synthetic speech, it is necessary to give a more detailed consideration of natural speech. Think of the numbers of people in our nation who are illiterate, despite being fluent in conversational English. Consider too that the Acadian French language, that variety of French indigenous to south Louisiana, had no written form until the mid 1970's. Noting these facts should make clear that a written language is an entirely different beast than its spoken counterpart.

In stark contrast to being identical to its spoken form, a written language is a *mapping* of one set onto another, not unlike what is done in the study of mathematics. Just as Des Cartes mapped algebra onto geometry by introducing the cartesian coordinate system, so has mankind mapped its spoken languages into a written form. Unfortunately or not, those governing the literary arts have not taken the same care to keep everything as *well-defined* as have the mathematicians. While this makes for some wonderfully moving literary works, and adds untold subtleties and nuances to the language, it also means there is no one-to-one correspondence between a written word and the sounds that comprise that same word in spoken form, (SPO256-AL2 Technical Data, 7).

Consider the character 'T'. It has one sound associated with it in the word 'teach', a different sound in the word 'bathe', and yet an entirely different sound in the word 'faction'. Conversely, there is no one-to-one correspondence between a given sound and the letter associated with it. As an example, consider the sound 'SH'. In the word 'shovel' it corresponds to the letters 'sh', but in the word 'faction' it corresponds to the letters 'ti'. If a one-to-one correspondence existed for transforming in either direction, it might be possible to define the transformation as a function and, at worse, have to deal with the elements of a preimage. Unfortunately, this is not the case.

Still, the mechanical transformation from text to speech can be accomplished. It is intuitive to anyone who can read this report that some words *sound just like they look.* Perhaps the foremost problem in generating synthetic speech is to document, in detail, every rule that determines just what constitutes *sounding just like it looks.*

But before that task can be done, it is necessary to decompose the spoken language into its fundamental components. Just as the written language has a finite number of components, (the twenty six characters of the alphabet and a handful of punctuation marks), so too has the spoken language. Those components are called phonemes. The English language is comprised of forty four phonemes, twenty five consonant phonemes and nineteen vowel phonemes, (SPO256-AL2 Technical Data, 8).

Everyone is familiar with the consonants and vowels of the written language. The letters 'a', 'e', 'i', 'o', 'u', and sometimes 'y' are the vowels, all others being consonants.[3] As was pointed out earlier, so too are the phonemes grouped. A phoneme is consonant if it is generated by an occlusion or obstruction of the breath channel, and it is a vowel phoneme if it is generated with an unobstructed or open breath channel, (SPO256-AL2 Technical Data, 8).

Each group of phonemes is further broken down into sub-groups. Vowel phonemes are classified as either high, mid, or low, and front, central, or back, depending on their general pitch and where they originate within the mouth. Table 1 shows each of the vowel phonemes as they are classified, (SPO256-AL2, 15).

Vowel Phonemes	Front	Central	Back
High	YR, IY, IH		UW, UH
Mid	EY, EH, XR	ER, AX	OW, OY
Low	AE	AW, AY, AR, AA	AO, OR

Table 1

The consonant phonemes are not so simply classified. First, the consonants are grouped by the point of occlusion which generates the sound. Those classifications are, (see figure 1), Labial, Labio-Dental, Inter-Dental, Alveolar, Palatal, Velar, and Glottal. Labial sounds are produced by bringing the lips together, either in or near contact. Labio-Dental sounds are the result of bringing the upper teeth in contact with the lower lip. Inter-Dental sounds occur when the tongue is placed between the teeth. Alveolar sounds are produced by touching, or nearly touching, the alveolar ridge with the tongue, (the

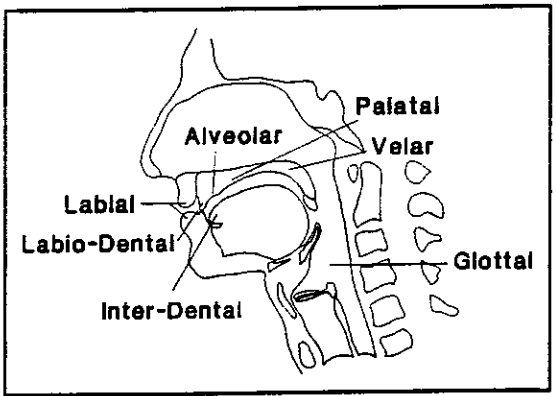


Figure 1: Points of occlusion for consonant phonemes.

[3] As an aside note, though everyone knows that 'y' is sometimes a vowel, many do not realize the circumstances that make this so. The letter 'y' is a consonant if and only if it appears in the initial position of a word. If it appears in any position after the initial position, it is a vowel.

Also, it is apparently no longer taught that 'w' can be a vowel. One example of this is the word 'cwm', meaning a kind of valley forming a natural amphitheater. A more contemporary example would be the word 'vacuum', except in this case the letter 'w' has been decomposed into its original components, double 'u's.

alveolar ridge is just behind the upper teeth). Palatal sounds are the result of nearly touching the palate with the body of the tongue, as opposed to Velar sounds wherein the body of the tongue touches the velum, or the fleshy rear part of the roof of the mouth. Finally, Glottal sounds are produced by constricting the opening between the vocal cords, (SPO256-AL2 Technical Data, 14).

In addition to these classifications, consonant phonemes are also classified as either a Stop, a Fricative, an Affricate, a Nasal, or a Resonant. A Stop phoneme is one in which the sound abruptly terminates. A Fricative is generated by continually passing air through the associated occlusion. An Affricate is a combination of both a Stop and a Fricative, wherein the passing of air is terminated fairly quickly. Finally, Resonants and Nasals are both generated by reverberating a pitch in the breath channel. The difference is that Resonants are reverberated in the mouth, and Nasals are reverberated in the nasal cavity.

Lastly, each consonant phoneme is either voiced or voiceless. If the sound includes vibration of the vocal cords, it is voiced. If air is passed through the vocal cords without vibrating them, the phoneme is voiceless. Table 2 shows each of the consonant phonemes by classification, (SPO256-AL2 Technical Data, 14).

		LABIAL	LABIODENTAL	INTERDENTAL	ALVEOLAR	PALATAL	VELAR	GLOTTAL
STOPS	VOICELESS VOICED	PP BB			TT DD		KK GG	
FRICATIVES	VOICELESS VOICED	WH	FF VV	TH DH	SS ZZ	SH ZH		HH
AFFRICATES	VOICELESS VOICED					CH JH		
NASALS	VOICED	MM			NN		NG	
RESONANTS	VOICED	WW			RR,LL	YY		

Table 2

Every word in the English language is spoken as some combination of the phonemes listed in tables 1 and 2. However, choosing the correct combination for a given word can be tricky at best. One phenomenon that is quickly discovered is that a given phoneme will sound acoustically different, depending upon its placement within a word, and within the context of those phonemes around it, (SPO256-AL2 Tech. Data, 8). This is due to the mechanics of the human anatomy generating the sounds. The result is that the synthetic generation of speech requires additional sound units, since a machine will produce the same sound from the same phoneme every time regardless of placement or context.

§ The SPO256-AL2 §

To distinguish between the phoneme set which makes up a language and the set of sounds required to generate a language mechanically, the latter is called an allophone set. In the special case of the English language, an expanded set of fifty nine allophones is required. These are contained in the internal ROM of the SPO256-AL2. To simplify timing requirements in constructing words from allophones, the SPO256-AL2 also contains five pauses of various durations. This provides a total of sixty four allophone addresses.

Decimal Address	Hex Address	Allophone	Sample Word	Duration	Decimal Address	Hex Address	Allophone	Sample Word	Duration
0	00	PA1	*PAUSE*	10 ms	32	20	/AW/	Out	370 ms
1	01	PA2	*PAUSE*	30 ms	33	21	/DD2/	Do	160 ms
2	02	PA3	*PAUSE*	50 ms	34	22	/GG3/	Wig	140 ms
3	03	PA4	*PAUSE*	100 ms	35	23	/VV/	Vest	190 ms
4	04	PA5	*PAUSE*	200 ms	36	24	/GG1/	Got	80 ms
5	05	/OY/	Boy	420 ms	37	25	/SH/	Ship	160 ms
6	06	/AY/	Sky	260 ms	38	26	/ZH/	Azure	190 ms
7	07	/EH/	End	70 ms	39	27	/RR2/	Brain	120 ms
8	08	/KK3/	Comb	120 ms	40	28	/FF/	Food	150 ms
9	09	/PP/	Pow	210 ms	41	29	/KK2/	Sky	190 ms
10	0A	/JH/	Dodge	140 ms	42	2A	/KK1/	Can't	160 ms
11	0B	/NN1/	Thin	140 ms	43	2B	/ZZ/	Zoo	210 ms
12	0C	/IH/	Sit	70 ms	44	2C	/NG/	Anchor	220 ms
13	0D	/TT2/	To	140 ms	45	2D	/LL/	Lake	110 ms
14	0E	/RR1/	Rural	170 ms	46	2E	/WW/	Wool	180 ms
15	0F	/AX/	Succeed	70 ms	47	2F	/XR/	Repair	360 ms
16	10	/MM/	Milk	180 ms	48	30	/WH/	Whig	200 ms
17	11	/TT1/	Part	100 ms	49	31	/YY1/	Yes	130 ms
18	12	/DH1/	They	290 ms	50	32	/CH/	Church	190 ms
19	13	/IY/	See	250 ms	51	33	/ER1/	Fir	160 ms
20	14	/EY/	Beige	280 ms	52	34	/ER2/	Fir	300 ms
21	15	/DD1/	Could	70 ms	53	35	/OW/	Beau	240 ms
22	16	/UW1/	To	100 ms	54	36	/DH2/	They	240 ms
23	17	/AO/	Aught	100 ms	55	37	/SS/	Vest	90 ms
24	18	/AA/	Hot	100 ms	56	38	/NN2/	No	190 ms
25	19	/YY2/	Yes	180 ms	57	39	/HH2/	Hoe	180 ms
26	1A	/AE/	Hat	120 ms	58	3A	/OR/	Store	330 ms
27	1B	/HH1/	He	130 ms	59	3B	/AR/	Alarm	290 ms
28	1C	/BB1/	Business	80 ms	60	3C	/YR/	Clear	350 ms
29	1D	/TH/	Thin	180 ms	61	3D	/GG2/	Guest	40 ms
30	1E	/UH/	Book	100 ms	62	3E	/EL/	Saddle	190 ms
31	1F	/UW2/	Food	260 ms	63	3F	/BB2/	Business	50 ms

Table 3

Table 3 lists all 64 allophones contained in the speech chip, (SPO256-AL2 Technical Data, 18). Constructing a vocal tract, (a spoken word or phrase), is simply a matter of addressing the appropriate allophones in the proper sequence. Since speech is significantly slower than normal microprocessor operations, each allophone can be output, (or *spoken*), as it is addressed. Terminating a word requires the addressing of a pause of appropriate length. Failure to address pauses between words will result in a phrase being spoken as a single word.

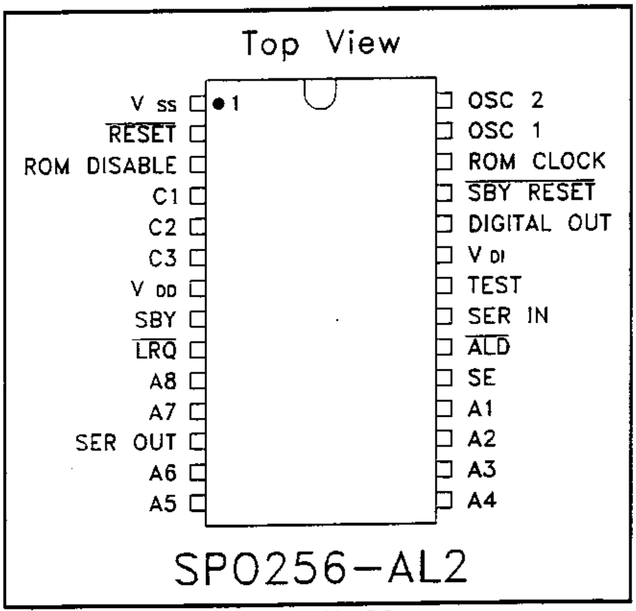


Figure 2: Pinouts of the SPO256-AL2

Two modes are available for addressing allophones in the SPO256-AL2. The addressing mode is selectable by pin 19, Strobe Enable, (SE).

Mode 0, (SE = 0), latches an address when any address pin(s), (A_1 - A_8)[4], make(s) a positive transition. There are two disadvantages to using this mode. First, timing is critical since latching occurs on the PGT edge of any one of the address lines. The second disadvantage is that PA1, the 10 ms pause, cannot be loaded.

Mode 1, (SE = 1), latches whatever address is setup on the address lines at the NGT edge of pin 20, ALD, (Address Load). Timing requirements are less critical in mode 1. The ALD pulse time should be no less than 200 ns, and no greater than 1100 ns. Address line transitions should occur no less than 80 ns before the ALD pulse, and should hold for at least 80 ns after the ALD pulse, (SPO256-AL2 Technical Data, 4).

Two pins are available for interfacing the SPO256-AL2 to a microprocessor. They are Load Request, (LRQ, pin 9), and Standby, (SBY, pin 8). LRQ is an active low pin. LRQ remains high during allophone output operations, and returns low upon completion indicating the appropriate time to load the next address. SBY is also active low. SBY assumes a logical high state until an address is loaded and goes low during allophone output processing.

[4] The documentation accompanying the SPO256-AL2 specifies the address pins as A_1 - A_8. The author does not know why the manufacturer did not use the accepted standard exponential A_0 - A_7 notation. In any case, to avoid confusion between this report and the manufacturers data sheets, the same notation is used here.

Even though both pins are active low by definition, they operate inversely in that LRQ provides a logical low signal at the appropriate time for a new address and SBY provides a logical high. Either pin may be used depending on the signal level required.

When an address is loaded, the allophone associated with that address is output serially over pin 24, (Digital Out), as a digital bit pattern. This digital sequence can then be channeled through a low pass filter. Since the digital signal is a square wave, and a square wave can be defined as a sum of odd harmonics, (the well known Fourier series):

$$f(x)=\frac{2}{\pi}\sum_{n=1}^{\infty}\frac{[1-(-1)^{n}]}{n}Sin(nx)$$

filtering out the higher harmonics, (n » 1), will decompose the square wave into a sinusoidal waveform. That sinusoidal waveform can then be amplified as an audio signal and output over a speaker.

§ THE CTS256A-AL2 §

Though the spoken language has been subdivided into fundamental components, and those components have been encoded into a specialized microprocessor, it remains to address them appropriately to construct meaningful vocal tracts. Toward that end, Naval Research Laboratories developed an algorithm to convert text strings into an appropriate string of allophone addresses. That algorithm is the basis for the op-code contained in the CTS256A-AL2, (CTS256A-AL2 Technical Data, 7).

The CTS256A-AL2 is actually a General Instruments® PIC7041 8-bit microprocessor, (a licensed second source of the Texas Instrument® TMS7041). The PIC7041 has 4 kilobytes of internal ROM, and 128 bytes of internal RAM. In the CTS256A-AL2, the internal ROM has been masked with op-code which enables it to convert ASCII text strings into allophone addresses compatible with the SPO256-AL2. The internal op-code also allows the user to run additional op-code stored in external ROM, but with somewhat less flexibility than the standard PIC7041.

The PIC7041 has four memory expansion modes available to the user. They are *single chip*, *peripheral*, *full*, and *microprocessor* expansion modes. The op-code masked into the CTS256A-AL2 sets the chip to the full expansion mode upon power up. This allows the 128 bytes of internal RAM and the 4 kilobytes of internal ROM to be utilized, and it allows for external addresses from 0200H to EFFFH, (Microchip® DS33001A, 15). Additionally, the built-in op-code allows for the following optional configuration features, (CTS256A-AL2 Technical Data, various pages).

¶ External ROM ¶

The CTS256A-AL2 allows for external ROM which can be used to store either exception words, (words which would not be pronounced properly), or user op-code. There is no default address for the external ROM, but its beginning address, (X000H), must fall on a 4K boundary such that 1000H ≤ X000H ≤ E000H. The existence of external ROM is specified by the five byte sequence 80H 48H 28H 58H 85H located at addresses X000H - X004H. Additionally, other segments of external ROM are reserved for special op-code to set it up as either an exception word ROM bank, or a user code ROM bank. Further discussion of the use of External ROM can be found in the chapter entitled Testing and Results, paragraph heading **External ROM**, (General Instrument® Product Data, 78).[5]

¶ External RAM ¶

An external RAM buffer can be added to the CTS256A-AL2. RAM must be added in 256 byte blocks, it must be contiguous, and a minimum of 512 bytes is required. External RAM must have an access time not greater than 250 ns. The default address of the external RAM buffer is 3000H and is redefinable in external ROM, (General Instrument® Product Data, 78).

¶ Parallel Data Port ¶

The CTS256A-AL2 contains a built in UART, (Universal Asynchronous Receiver Transmitter), which can accept data, or it can accept data via a parallel port. The default address for the parallel port is 0200H, but is redefinable in external ROM, (General Instrument® Product Data, 79).

[5] The reference material listed as General Instrument Product Data is actually a section of a reference book provided by Tandy, Inc. Since only those pages pertaining to the CTS256A-AL2 were provided, (pp 77 - 95), the actual title of the book is unknown.

¶ Switch Selectable UART Parameters ¶

By default, the CTS256A-AL2 will use UART parameters of asynchronous communication, 7 data bits, 2 stop bits, and no parity. Switch selected parameters can be specified via a 74LS373 buffer. The default address of this buffer is 1000H, and is redefinable in external ROM, (General Instrument® Product Data, 80).

¶ Pinout Information ¶

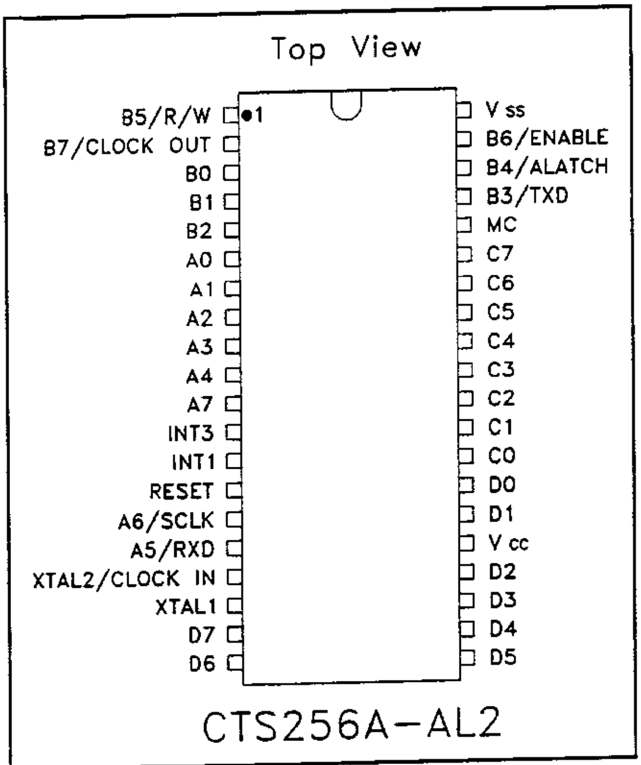


Figure 3: Pinouts of the CTS256A-AL2/PIC7041

Figure 3 shows the pinouts of the CTS256A-AL2. In this section, the pin functions will be described as they pertain to the PIC7041. These descriptions will then be qualified to relate their operation within the context of the CTS256A-AL2.

Pin 36, (MC, Mode Control or Memory Control), is an input pin which is used to specify the operation, (or memory expansion), mode of the PIC7041. If MC is held at logical hi it will force the microprocessor into the *microprocessor expansion mode* of operation, disabling the microprocessor's ability to access either the internal RAM or ROM and forcing all 64K addresses to be external. Holding MC at logical lo allows internal address space to be accessed, and the expansion mode is set by software, (General Instrument® DS33001A, 16). Since the op-code contained in the CTS256A-AL2 requires *full expansion mode*, MC must be held at logic level 0.

Of the microprocessor's 40 pins, 32 are I/O ports labeled A_0 - A_7, B_0 - B_7, C_0 - C_7, and D_0 - D_7, (ports A, B, C, and D).

Port A lines A_0 - A_4 and A_7 are bi-directional lines, and A_5 - A_6 are input only, (General Instrument® DS33001A, 16). In the case of the CTS256A-AL2, lines A_0 - A_4 and A_7 are used to specify configuration parameters, (see table 4 on the next page). Line A_5 is RXD, (Receive Data), for the built-in UART, and line A_6 is not used.

Port B is an output only port, and is utilized as follows in the CTS256A-AL2. Line B_0 is used for handshaking purposes for both the serial and parallel port. When the input data buffer, (internal or external RAM), reaches 86.5% full, B_0 goes to logic level 0. When the buffer's contents are reduced to 50%, B_0 returns to logic level hi, (CTS256A-AL2 Technical Data, 2). Lines B_1 - B_3 and B_7 are not used, though B_3 is the TXD, (Transmit Data), for the built-in UART. Line B_4 is used as an address latch control line when port C is used to output the least significant byte of an address. Line B_5 is used as read/write control for external RAM, and line B_6 is used to enable the chip-select device, 74LS138.

Pins	6 (A_0)	7 (A_1)	8 (A_2)	Mode
	0	0	0	Parallel Port Input
	0	0	1	Serial Port Input; 50 Baud
	0	1	0	Serial Port Input; 110 Baud
	0	1	1	Serial Port Input; 300 Baud
	1	0	0	Serial Port Input; 1200 Baud
	1	0	1	Serial Port Input; 2400 Baud
	1	1	0	Serial Port Input; 4800 Baud
	1	1	1	Serial Port Input; 9600 Baud
Pin	9 (A_3)			Mode
	0			Default UART Parameters: Asynchronous, 7, 2, None
	1			Switch Selected Uart Parameters
Pin	10 (A_4)			Mode
	0			Internal RAM Buffer
	1			External RAM Buffer
Pin	11 (A_7)			Mode
	0			Carriage Return Only Delimiter
	1			Any Delimiter

Table 4

Port C is a bi-directional I/O port, and in the full expansion mode, (i.e. for CTS256A-AL2 purposes), it serves as a multiplexed data/address bus. Lines C_0 - C_7 serve as the data bus, or in conjunction with B_4 and a 74LS373 data buffer, they can output the least significant byte of an external address, (General Instrument® DS33001A, 16).

Port D can be a bi-directional data bus, but as used for the CTS256A-AL2, it outputs the most significant byte of the address bus, (General Instrument® DS33001A, 16).

Other pins involved in CTS256A-AL2 operation are pins 12, 13, 14, 17, and 18. Pin 12 is Interrupt 3, and is dedicated to handshaking operations for the parallel port. Pin 13 is Interrupt 1, and is dedicated to handshaking with the SPO256-AL2. Pin 14 is a hardware reset pin. A negative pulse of not less than 500 μs will cause the microprocessor to go through its power-up initialization. Finally, pins 17 and 18 are inputs for the external 10.00 MHz crystal, (CTS256A-AL2 Technical Data & General Instrument® Product Data, various pages).

Circuit Design

It is difficult for the author to take credit for designing the circuitry involved in this project. For the most part the circuitry is precisely as recommended in the technical literature that accompanies the CTS256A-AL2. However, certain considerations were made before the decision to use the suggested configuration was made.

Recall from the <u>Design Objectives</u> chapter that the primary intent of this project is experimental in nature. For maximum flexibility, it was decided to incorporate into the unit all options pertinent to the processing of text-to-speech. Consideration was given to the possibility of reorganizing these options within the external decodable address space. According to the literature, this could be accomplished by re-defining the addresses from their program defaults via external ROM. The only supporting argument for doing so was to lend some degree of originality to the final design. No other reasons were evident to lend preference for one address over another for any external devices.

But deliberation did yield one reason for **not** redefining any default addresses. If the circuit configuration is defined in ROM, then the entire unit is dependent on that one microchip to work. By leaving all addresses at the program defaults, the unit can remain functional even if one or more of the optional devices become defective. For this reason, it was decided to leave all optional features at their default address locations.

§ CTS256A-AL2 §

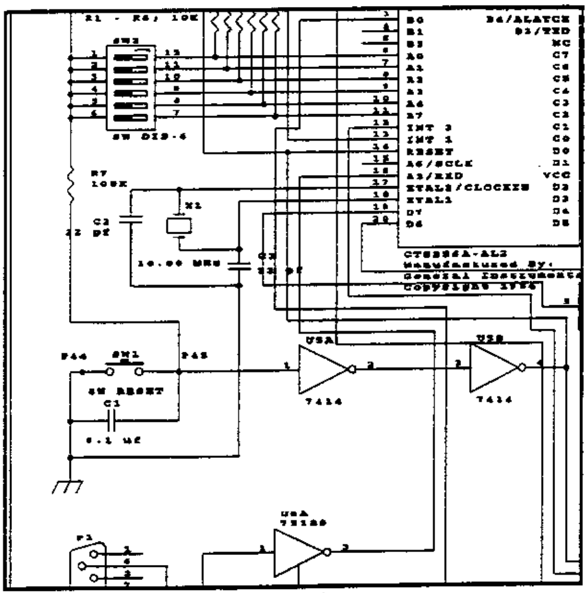


Figure 4: Hardware reset circuitry

Original or not, the reasons for certain component choices are obvious and noteworthy. The first is the choice of C_1 and R_7, (0.1 µf and 100 kΩ respectively). These components set the time constant that determine the pulse width of the hardware reset. Recall that a minimum of 500 µs is required for the reset to function properly, but no maximum pulse width is specified. Since:

$$T_c = RC = (0.1 \times 10^{-6})(1.0 \times 10^{5}) = 0.01 \text{ s}$$

and

$$0.01 \text{ s} \div 500 \times 10^{-6} = 20$$

C_1 and R_7 provide a time constant which is 20 times greater than the minimum requirement. Note that even though it will take 5 time constants for the voltage at pin 14 to reach V_{CC}, it will take significantly less than 1 time constant to reach a minimum logic level 1. Recall that logical 1 is defined as:

$$1 \equiv x \mid 2.5\text{ V} \le x \le 5\text{V}$$

therefore, a safety factor of 20 is not as excessive as it may seem at first glance.

This leads to a second component choice which bears noting. U_5 is specified as a 7414, (a Schmitt trigger invertor), instead of the more common 7404, (a standard high speed invertor), precisely because it is used for a mechanical pushbutton input. If a 7404 were substituted, the hardware reset would not function reliably, if at all.

The final component selections which warrant consideration are the choices for U_3 and U_4, a 74LS373 and 74LS138 respectively. To understand the choices of these components, it is necessary to consider some details of the CTS256A-AL2 address space.

Though the PIC7041 is only an 8-bit microprocessor, it can access addresses of up to 16 bits by multiplexing the data bus, (Port C; C_0 - C_7; CTS256A-AL2), into the address bus. Hence the need for U_3.

Addressing is accomplished in two steps. The least significant byte of an address is written onto the data bus, and latched into U_3. This fixes lines A_0 - A_7 and holds them through the next step. Then the most significant byte of the address is written to Port D. The least significant nibble of the most significant byte, (pins D_0 - D_3; CTS256A-AL2), is directly part of the address bus. Therefore, lines A_0 - A_{11} are set. The most significant nibble of the most significant byte, (pins D_4 - D_7; CTS256A-AL2), are connected to U_4 to select the appropriate chip(s).

At this point it is necessary to understand that the internal ROM of the CTS256A-AL2 is at address space F000H - FFFFH, for a total of 4096 bytes. Recall from the **Research of Literature** chapter that the CTS256A-AL2 is limited to run in the PIC7041 *full memory expansion* mode, or it must be able to address this space internally. This is the reason for connecting the most significant bit of Port D, (pin D_7; CTS256A-AL2), to the G_1 enable pin of U_4. Doing so prevents any possible conflict of address space.

Note, however, that a hidden ramification lies in this design feature. Although the microprocessor is capable of addressing 16 bits, or 2^{16} = 65,536 locations, disabling all external address space with $D_7 = 1$ reduces available external address space by half. This is significantly more than the 4 kilobytes of internal ROM. Also, using the rest of the most significant nibble to select the appropriate chip(s) further limits the amount of external address space available. Effectively, each device is allocated 2^{12} = 4096 bytes of address space, regardless of what it will actually use.

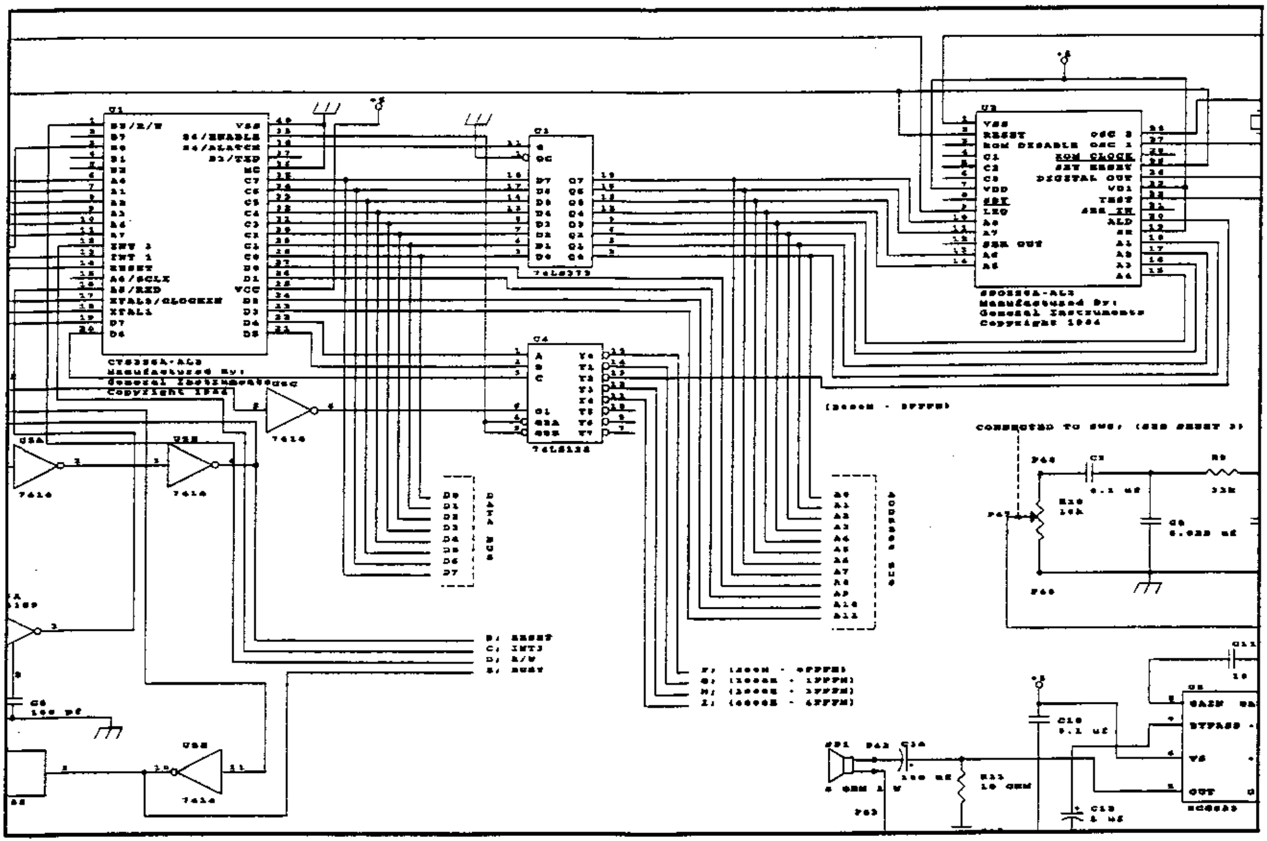


Figure 5: Data/Address bus circuitry

These design ramifications are mentioned here because the technical literature is misleading on this point. For example, the data sheet accompanying the CTS256A-AL2 states that external ROM can be configured anywhere between 1000H and E000H, yet the suggested circuit design allows nothing above 7FFFH, ($32,767_{10}$), to be addressed.

As it turns out, this is perfectly acceptable for this unit, since only 0200H - 4FFFH is being utilized. And, as long as these limitations are understood, the single 74LS138 is more than sufficient for the job. In fact, it leaves 3 address blocks of 4 kilobytes each not used.

§ Power Supply §

Digital circuitry transmits data in the form of a square wave signal. For this reason, a very *stiff* power supply, or one that will not load down easily, is required. To understand the reasoning behind this statement, it is necessary to consider the nature of a square wave, as well as the nature of AC circuitry.

Mathematically, an ideal square wave would be defined discontinuously such that f(x) was either 1 or -1 at any given point in its domain. There are several techniques that would allow this, (i.e. $\delta_{ij} \pm C$), but they are irrelevant since they are not physically realizable.

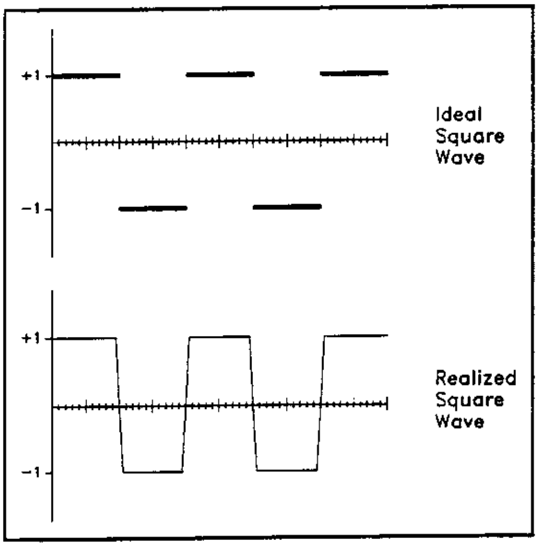


Figure 6: Square waves

Fortunately, realizable square waves are continuous over their domain. This is fortunate because it means they are always at a finite frequency.

Consider figure 6. Note that the frequency of an electrical signal is defined as dv/dt. The case of the ideal square wave is discontinuous, therefore dv/dt is undefined at the points of transition. Nonetheless, in the limit it can be shown to approach infinity. In the case of the realizable square wave, dv/dt is defined and finite. Of course, it is still quite large, but it is finite.

It is important not to confuse the frequency of the square wave with the frequency of the signal. The square wave may have a frequency of 5 MHz, (or a period of 200 ns), but the signal has a dynamic frequency. From edge to edge, the signal frequency is zero, while at the transition points its frequency approaches infinity as the transition period approaches zero.

Consider now the reactance of a capacitive circuit. It is defined mathematically as:

$$X_C = [2\pi fC]^{-1}$$

As the frequency of a signal, (f), approaches infinity, the reactance approaches zero. Recall that any two conductors separated by a dielectric have some capacitance. Therefore the traces on the circuit board form capacitors. Even though it is a very small capacitance, during the square wave transition period it produces relatively low reactance.

During the relatively long period of the square wave, the reactance of the load circuitry is much greater than that of the power supply, and most of the voltage is developed across the load. But, during the transition period of the signal, the reactance of the load becomes very low, (it approaches zero, though only for a very short time). At this time the voltage will tend to develop across the internal resistance of the power supply. This is the common loading effect.

In digital circuitry, this results in noise spikes which can cause erroneous circuit switching. The most obvious solution for this problem is to simply make the power supply much stronger than would seem necessary. This approach was chosen in this project.

The unit requires three voltage levels for proper operation, +5 V_{DC}, +12 V_{DC}, and -12 V_{DC}. The ±12 V_{DC} supplies are needed only for the RS232 interface line driver chip. For this purpose 15 W of power and 1 A of current is more than sufficient, so ECG966 and ECG967 fixed voltage regulators were selected.

For the 5 V_{DC} supply, 30 W of power and 3 A of current are sufficient, so the ECG970 adjustable voltage regulator was chosen, (Note: The transformer will provide a maximum of 2 A). An ECG931, (a fixed voltage regulator), would have been the author's first choice, but an ECG970 was already on hand.

A 5:1 transformer was chosen to provide 24 V_{rms} to the bridge rectifier. This provides a rectified and filtered voltage of approximately:

$$(24\ V) \div (0.707) \approx 34\ V_{DC}$$

Since the ECG966 and ECG967 regulators require a minimum input voltage of ±14 V:

$$34\ V_{DC} \div 2 = \pm 17\ V_{DC}$$

is sufficient.

The ECG5318 Bridge was chosen to rectify the output from the transformer because it will handle twice the maximum current of the transformer. A 1000 μf filter capacitor was chosen to filter the rectified signal because it was on-hand and it was *large*. Because ripple filtration at this point in the circuit is necessary but not critical, more careful consideration of the capacitor's rating was unwarranted. Final ripple filtration will be accomplished within the regulators.

To divide the voltage between the ±12 V_{DC} regulators, (the 5 V_{DC} regulator can take its input voltage from the +12 V_{DC} regulator's input voltage), two zener diodes are used. Under no load, they must be able to dissipate the power associated with maximum current, 2A. With 17 V_{DC} across each, a power rating of 34 W is required. Hence the choice of 17 V @ 50 W diodes. Measurements revealed that approximately 6 V_{DC} would develop across the bleeder resistance, which would call for 3 Ω @ 6W for the required 2 A. Because they were on hand, 3 resistors rated for 1 Ω @ 3 W each were chosen. This also provided a 50% safety factor on the expected power dissipation.

It was decided that the power supply should be external to the main unit, to prevent unnecessary heat build-up. A cable of 6 conductors is required to connect the power supply to the main unit. The conductors required are +5 V, +12 V, -12 V, GND, and 2 for the on/off switch, (an integral part of R_{10}, the volume control).

One design consideration that may not be obvious here is that the power supply unit will not remain powered if the cable is disconnected. This is due to SW5 being dependent upon the cable connection to provide continuity.

For a more detailed examination, the full schematics of this project are included in the appendices. They are comprised of 3 fold-out B-size plots.

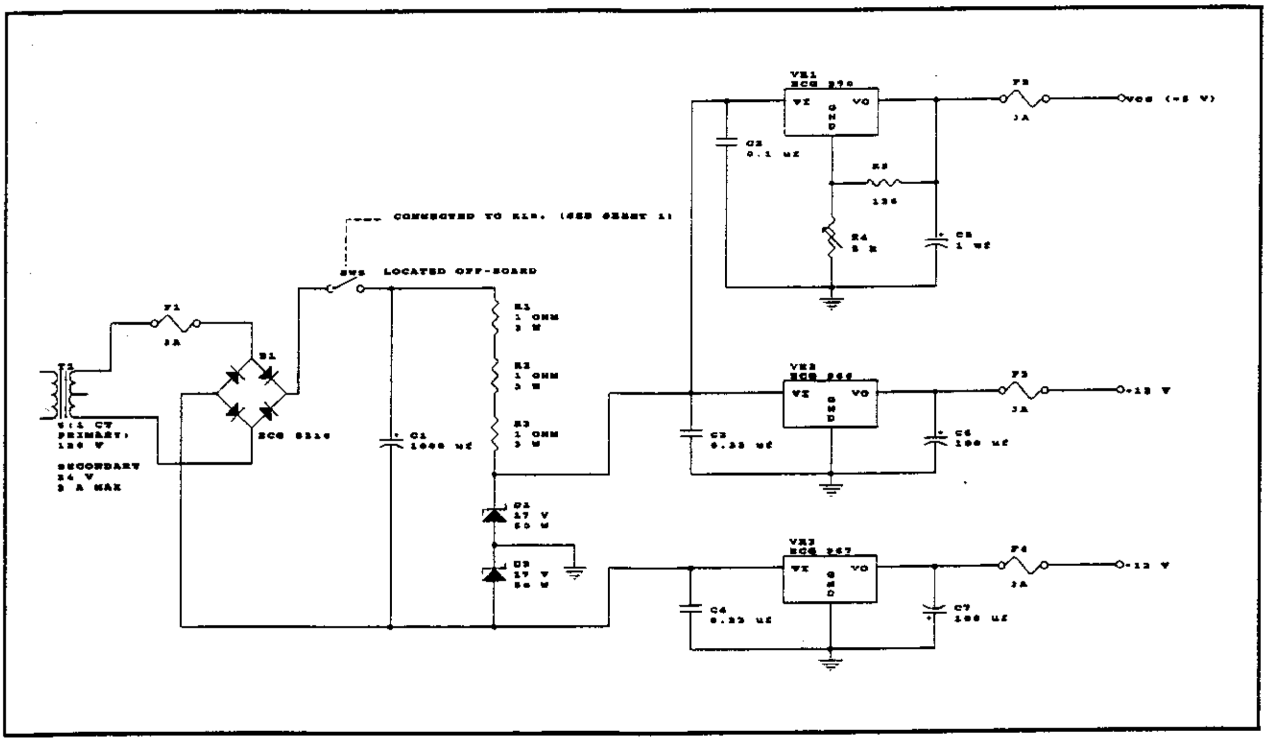


Figure 7: Power supply circuitry

Theory of Operation

Upon completion, the unit built for this project will be a complete single board computer, (SBC). The main circuit board will contain two microprocessors, 2 kilobytes of external RAM, 4 kilobytes of external ROM, both serial and parallel data ports, (1 each), a multiplexed data/address bus, and a control bus. Furthermore, maximum flexibility will be achieved by making optional configuration parameters fully switch selectable.

The function of the SBC will be to *read* text files from the disk(ette) of a host computer, (or text strings echoed from the CRT). Before addressing each section of the unit in detail, an overview of the system will be considered.

§ System Overview §

Whether displayed on the CRT or stored in a disk(ette) file, MS-DOS® based computers handle text in the form of ASCII encoded characters. The unit that is the subject of this report will

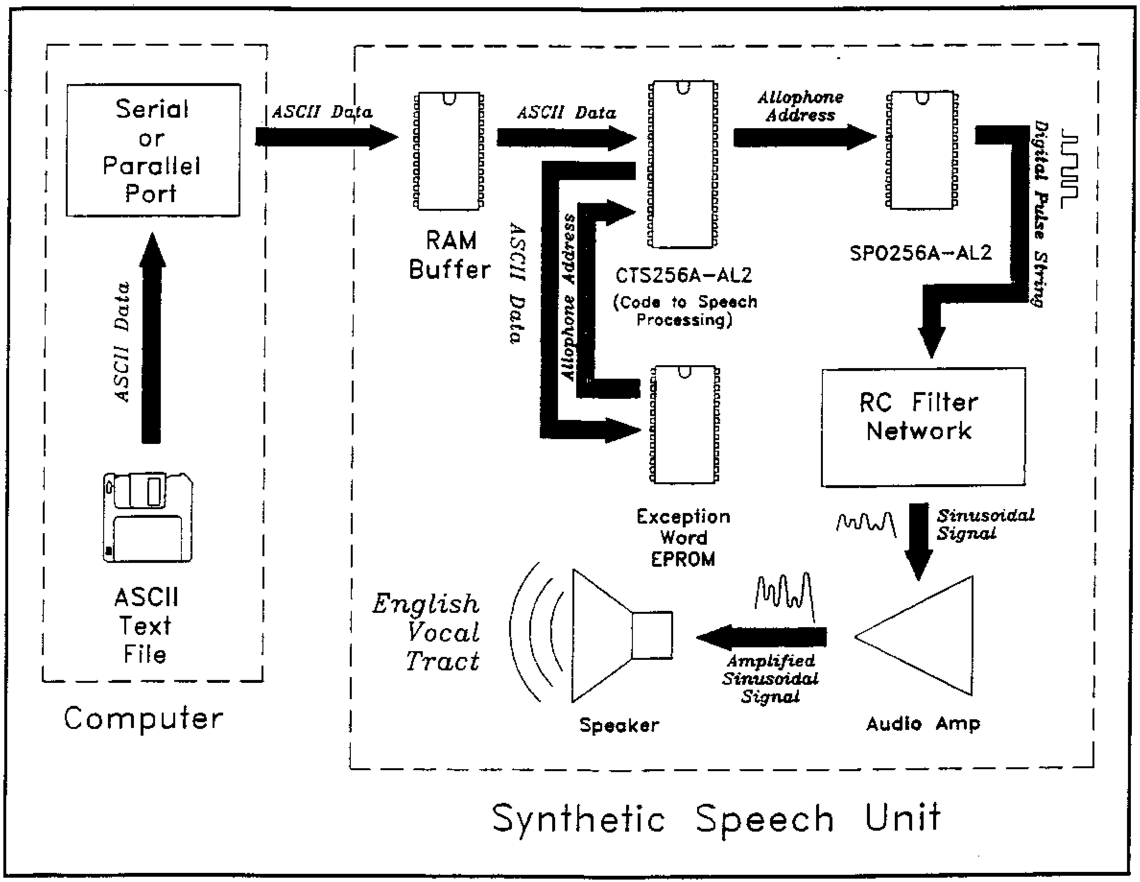


Figure 8: System flow chart

accept ASCII data via either a standard serial or parallel port. The data is then stored in a RAM buffer to await processing. The primary microprocessor, (CTS256A-AL2), reads in each character until it has an entire word. The CTS256A-AL2 distinguishes the beginning and end of each word in much the same way as a human, by the delimiting characters. Such delimiters are spaces, commas, periods, etc.

Once the CTS256A-AL2 has an entire word to work with, it checks the contents of external ROM for an exact match to the sequence. If it finds an exact match, it loads the allophone address sequence associated with that word for transmission to the speech microprocessor, (SPO256-AL2). Otherwise, it processes the text string that makes up the word through an algorithm. This algorithm converts the text into a sequence of allophone addresses and buffers them in memory for transmission to the SPO256-AL2.

Once a word has been processed, the resultant sequence of allophone addresses are transmitted to the SPO256-AL2. The SPO256-AL2 then outputs the allophones associated with each address in the form of a sequence of square wave pulses. That signal is then channeled through a low pass RC filter network which decomposes the signal into its lowermost harmonics, (recall that a square wave is the summation of odd harmonics). This decomposition results in the signal being transformed into a sinusoidal waveform. That waveform is then amplified and output through a speaker as a spoken vocal tract.

§ System Details §

Refer to the schematics in the appendices to aid in following the operation descriptions.

Upon power up, or upon hardware reset, the CTS256A-AL2 goes through an initialization routine. The CTS256A-AL2 is set to operate in the *full memory expansion* mode and optional parameters are established. The exact sequence of events are unknown, but each of the following tasks is performed. They are organized here in the order that seems most reasonable to the author.

A search is performed over all external decodable address space, in 4 kilobyte increments, to find the 5 byte sequence 80H, 48H, 28H, 58H, 85H. If found, this is assumed to be the beginning address of external ROM, and the op-code contained in the reserved areas establish it as containing exception words. No optional parameter addresses are redefined here, but if they were those changes would be set up.

Pins 6 - 11 are read into the CTS256A-AL2, (Port A; Lines A_0 - A_4 & A_7; These bits are set by DIP switch 2).

Pins 6 - 8 determine if data is to be input through the parallel port, (See paragraph heading PARALLEL PORT in this section), or the serial port, and if the serial port then at which baud rate, (See Table 4, p. 18).

If pin 9 is at logical lo, default UART parameters of asynchronous communication, 7 data bits, 2 stop bits, and no parity are assumed. If pin 9 is at logical hi, UART parameters are loaded from hardware, (See paragraph heading SWITCH SELECTABLE UART PARAMETERS in this section).

If pin 10 is at logical lo the 128 bytes of internal RAM are used as the input and output buffers. If pin 10 is at logical hi the 2 kilobytes of external RAM are used as the input and output buffers, (See paragraph heading EXTERNAL RAM in this section).

If pin 11 is at logical lo, allophone addresses are retained in RAM until a carriage return, (CR), is detected. Then and only then are they sent to the SPO256-AL2 for speech processing. This allows sentences to be spoken with greater fluency. If pin 11 is at logical hi, each word is spoken as it is processed.

Once all parameters have been set up, the CTS256A-AL2 will address an allophone sequence in the SPO256-AL2 causing it to speak "OK." Since no user code is contained in external ROM, the CTS256A-AL2 will idle continuously within its algorithm until it receives data through whichever port is currently configured.

¶ PARALLEL PORT ¶

If pins 6 - 8 are all at logical lo, the CTS256A-AL2 will accept data through the parallel port at address 200H, (See schematic sheets 1 of 3 and 2 of 3 in the appendices as an aid to follow the sequence of events).

Once the host computer has set up a data word on pins 2 - 9 of the standard DB25 port connector, (P2), it will strobe pin 1, (DS; Data Strobe), to a logical lo. This triggers three reactions. First it clears pin 5 of U9, (a D type flip flop), which is connected to pin 10 of P2, (ACK; Acknowledge). The NGT edge on ACK indicates to the host computer that the data has been received. Second, it strobes the data into U10, a 74LS374 data latch, though it must first be inverted. Finally, it triggers INT3 on the CTS256A-AL2, indicating a data word is waiting at address 200H. Even though these three events are listed sequentially, it should be noted that they occur simultaneously.

Upon receipt of the NGT pulse at INT3, the CTS256A-AL2 accesses address 200H. This sets pin 15 of U4, (the 74LS138), in the logical lo state. The NGT edge on pin 15 of U4 enables the output pins of U10, thus placing the data word on the data bus, where the CTS256A-AL2 then reads it. The subsequent PGT edge of the same signal then clocks a logical high back onto pin 5 of U9. Recall that pin 5 of U9 is connected to ACK on P2. That PGT edge indicates to the host computer that the speech unit is ready for the next character.

It will be covered in detail in the paragraph heading EXTERNAL RAM, but it should be noted that pin 3 of the CTS256A-AL2 is connected to pin 11 of P2, (Busy). This is to protect against overwriting data in the input buffer. At this point, it is sufficient to note that if pin 11 of P2 is not at logical lo, the host computer will not transmit any data. Leaving pin 11 of P2 open will not suffice.

¶ SERIAL PORT ¶

If pins 6 - 8 on the CTS256A-AL2 have any combination other than:

$$\text{Pin } 6 = \text{Pin } 7 = \text{Pin } 8 = 0$$

the combination sets the baud rate for the built in UART, (See table 4, p. 18). The CTS256A-AL2, upon initialization, will wait for data to be transmitted over pin 16, (A_5; Port A). No interrupt is required to signal data transmission through the serial port. However, an MC1488 line driver, (ECG75188; U7), and an MC1489 line receiver, (ECG75189; U6), are required for conversion of the TTL signals to and from the standard RS232 $\pm 12\ V_{DC}$ signals respectively.

The host computer will transmit data over pin 3 of a standard DB9 connector, (P1). This signal is then converted from ±12 V by U6, and input over pin 16 of the CTS256A-AL2.

No connection from the CTS256A-AL2 TXD to the host computer is required. If the speech unit were intended to be placed at a significant distance from the host computer, this may not be the case. Since, however, cable length will not be sufficient to allow significant interference, two way communication is not required.

Pin 3 of the CTS256A-AL2 is connected to pin 8 of P1, (CTS; Clear To Send). A logical lo on this line indicates to the host computer that the unit is ready to receive data, and a logical high indicates that the unit is busy. It should be noted here that a logical lo RS232 voltage is defined as:

$$\text{logic } 0 \equiv +3\text{ V} < \text{lo} < +12\text{ V}$$

and a logical high is defined as:

$$\text{logic } 1 \equiv -3\text{ V} > \text{hi} > -12\text{ V}$$

Any voltage X such that:

$$-3\text{ V} < X < +3\text{ V}$$

is undefined. This may seem opposite to what is logical, but note that the line driver and receiver both invert the signal. Therefore, the seemingly backward definition is simply a device to prevent confusion.

Finally, pin 4, (DTR; Data Terminal Ready), and pin 6, (DSR; Data Set Ready), of P1 must be shorted together. Pin 4 is set hi by the host computer before transmission begins, and Pin 6 is tested by the host computer to verify the unit is ready to accept data. Since the unit is only operated as a dedicated device, two way communication is not required and simply shorting these pins together will suffice.

¶ SWITCH SELECTABLE UART PARAMETERS ¶

If pin 9 of the CTS256A-AL2 is at logical lo upon initialization, default UART parameters of asynchronous communication, 7 data bits, 2 stop bits, and no parity are assumed. If, however, pin 9 is at logical hi, UART parameters are loaded in a one byte data word from address 1000H, a 74LS373 designated U13. DIP switches SW3 and SW4 are used to set a bit pattern on the inputs of U13. In all cases, logical lo is equivalent to *off* or *open*, and logical hi is equivalent to *on* or *closed.*

When the CTS256A-AL2 loads 10H in Port D, (the MSB of address 1000H), pin 14 of U4 pulls pin 1 of U13 lo enabling the output bits. The CTS256A-AL2 then reads that word from the data bus and sets the UART parameters as follows.

Bit B_0 is set by DIP switch SW3:1. If B_0 is lo, Motorola® protocol is used. Motorola® protocol places a longer idle time between blocks than between frames within a block. In Motorola® protocol, an idle time of 10 bits indicates the start of a new frame. For more details, refer to General Instruments® document DS33001A.

WARNING!

This option, **Motorola® Protocol**, does **NOT** make this unit compatible with Apple® or MacIntosh® computers. These computers use an RS232 interface which requires that each data bit be accompanied by its compliment on a separate line. Attempting to use this unit with either of these computers, or with any non-IBM® compatible computer, **may result in damage to either this unit, the host computer, or both.**

If B_0 is at logical hi, Intel® protocol is used. Intel® protocol distinguishes block separation from frame separation by an extra bit, an address bit, in each frame. If the address bit is 1, the frame is the first in a new block. For more information, refer to General Instruments® document DS33001A.

Bit B_1 is set by DIP switch SW3:2. If B_1 is at logical lo, isosynchronous communication is used. Isosynchronous communication is suitable for high speed data transfers since each bit of data is equal to one clock cycle.

If B_1 is at logical hi, asynchronous communication is used. Asynchronous communication differs from isosynchronous communication in that one data bit equals 16 clock cycles. For more information on either isosynchronous or asynchronous communications, refer to General Instrument® document DS33001A.

Bits B_2 and B_3 are set by DIP switches SW3:3 and SW3:4 respectively. Bits B_2 and B_3 determine the number of bits per character as defined in table 5.

B_2	B_3	Bits/Character
0	0	5
0	1	6
1	0	7
1	1	8

Table 5

Bit B_4 is set by DIP switch SW3:5. If B_4 is at logical lo, parity is disabled. This is equivalent to a *no parity* setting. If B_4 is at logical hi, parity checking is enabled.

Bit B_5 is set by DIP switch SW4:1. If parity is enabled, bit B_5 at logical lo sets the unit to odd parity checking. If B_5 is at logical hi, the unit is set to even parity checking.

Bit B_6 is set by DIP switch SW4:2. If B_6 is at logical lo the unit is set to *serial I/O* mode. If B_6 is at logical hi the unit is set to the *Multiprocessor Communication* mode. Multiprocessor communication is necessary for connecting to multiple microprocessors, and will not be discussed further in this report. For more information on multiprocessor communication mode, refer to General Instrument® document DS33001A.

Bit B_7 is set by DIP switch SW4:3. Setting B_7 at logical lo specifies 1 stop bit per character. Setting B_7 at logical hi specifies 2 stop bits per character.

¶ External RAM ¶

Upon initialization, if pin 10 of the CTS256A-AL2 is at logical lo an internal 128 byte RAM buffer will be used for storing input data from the host computer and output data for the SPO256-AL2. If pin 10 is at logical hi, the 2 kilobytes of RAM at address 3000H will be used instead.

When initialization procedures determine that pin 10 is at logic level 1, the CTS256A-AL2 begins searching for the end address of RAM by looking for the first non-RAM location. This search is performed in 256 byte blocks. Since no addresses are redefined in ROM, this search will not proceed past 2 kilobytes.

Whether internal or external RAM is employed, it is divided into an input and output buffer. If the 2 kilobytes of external RAM is in use, 256 bytes will be allocated as an output buffer and the remaining locations will be allocated as an input buffer. If the internal RAM is used, RAM allocation will be made in similar proportion. RAM allocation cannot be redefined.

RAM management is accomplished with pin 3 of the CTS256A-AL2. Normally held at logical hi, pin 3 will toggle lo when the input buffer reaches 87.5% input capacity, (Recall that pin 3 is used in both serial and parallel data transmission). If data transmission continues resulting in the input buffer reaching 100% capacity, INT3 and the internal serial port interrupt are both disabled to prevent overwriting. Once the input buffer contents are reduced to 50% capacity, pin 3 returns to a logical hi state and, if disabled, all interrupts are reenabled.

¶ Addressing the SPO256-AL2 ¶

Once a sequence of allophone addresses are buffered in RAM and ready to be spoken, they must be sent to the SPO256-AL2. It is imperative to note that the CTS256A-AL2 is **not** writing data to the SPO256-AL2, but is instead addressing ROM locations within it. The actual workings of the allophone processing within the SPO256-AL2 are essentially unknown to the author, but some things can be safely deduced from the information available.

If each allophone were simply the 8-bit pattern contained at its address, all allophones would be of identical duration. But table 3 on page 13 shows this not to be the case. Instead, each address vectors the SPO256-AL2 to some internal routine which produces the associated allophone.

Like the CTS256A-AL2, the SPO256-AL2 has an internal ROM bank masked with op-code, some of which is allophone information. However, instead of 4 kilobytes the SPO256-AL2 has 16 kilobytes of internal ROM. It also contains special circuitry for concatenating allophones into vocal tracts. Though the SPO256-AL2 is a microprocessor, its function is highly specialized and considerably more complex than the CTS256A-AL2's. Therefore, the inner functions will be considered transparent for the purpose of this report and concerns will be limited to its external operation within the circuitry of this project.

The SPO256-AL2 is allocated locations 2000H - 2FFFH within the external decodable address space of the unit. Nonetheless, it only utilizes locations 2000H - 203FH, (see table 3, p. 13).

These addresses are accessed as any other, by writing the LSB to Port C, and then writing the MSB to Port D. Note that the MSB of 20H causes pin 13 of U4, (the chip-select device), to pull pin 20 of the SPO256-AL2 lo. This is the active lo ALD, (address load), which causes the SPO256-AL2 to accept the address and process the allophone associated with it.

In this case however, the CTS256A-AL2 does not read anything from the data bus subsequent to addressing the location. Communication from the SPO256-AL2 to the CTS256A-AL2 is limited to the LRQ line. When the SPO256-AL2 is busy, LRQ is set to logical lo. This activates INT1 of the CTS256A-AL2. The CTS256A-AL2 then waits for INT1 to clear before continuing.

As a note of interest, pins 10 and 11 of the SPO256-AL2 could be left unconnected from the rest of the address bus. Since only 64 locations are used, and since:

$$2^6 = 64_{10} = 111111_2 = 3FH$$

these connections are unnecessary. If trace density were a problem in some later design, this may be worth noting.

¶ SPO256-AL2 OUTPUT ¶

The allophones associated with each address are output from the SPO256-AL2 as soon as they are addressed. Buffering output is unnecessary since verbal speech is inherently slower than normal microprocessor operations. The output is serial and in the form of a square wave.

This digital sequence must be converted to a sinusoidal waveform before it will approximate speech. Instead of using any standard form of D/A conversion, a low pass RC filter network is used. Recall that a square wave is the summation of odd harmonics. By filtering out the higher harmonics of the square wave, a more sinusoidal waveform is produced. The RC filter network consists of capacitors C7 - C9, and resistors R8 - R10, (see schematic 1 of 3 in the appendices). Resistor R10 is mounted on the front housing panel as a volume control.

The portion of the sinusoidal signal that is developed at the rotor tap of R10 is fed into U8, an ECG823 1 W audio amplifier. The output from U8 is *spoken* over a standard 8 Ω speaker.

Testing and Results

Several problems were encountered during the developmental stages of this project. Not the least of these was the nature of the circuitry itself. Because the microprocessor has a clock speed of 5 MHz, circuit construction on a breadboard was not even attempted. The stray capacitance associated with the friction connections would have rendered the unit inoperable.

Instead the unit was prototyped on printed circuit boards. This proved to be of secondary benefit in that it allowed the author to develop certain techniques for fabricating double sided circuit boards which will be discussed in the chapter entitled CONCLUSIONS AND RECOMMENDATIONS.

The first prototype was nearly completely unsuccessful. No speech was generated by the first prototype, since neither data port was operable. Among other problems, it was discovered that a number of the via connections had poor continuity. Nonetheless, the first prototype did initialize as long as no optional features were in place. When power was applied, or when the reset button was depressed, the unit would *speak* "OK." This served to verify that the interface between the CTS256A-AL2 and the SPO256-AL2 was functional, as well as the low pass filter and audio amplifier.

The second prototype was successful, after some testing and modification. Faults in the circuit design and other difficulties were discovered relating to seven general areas. Those areas are product support, product design, the address bus, the parallel port, the serial port, external ROM, and MS-DOS®. Each of these areas will be discussed under separate paragraph headings.

¶ PRODUCT SUPPORT ¶

The General Instrument® chip-set this unit is designed around was purchased by the author approximately two years ago for the purpose of this project. At that time the author began familiarizing himself with the products, the recommended circuitry, etc. Research and development work has been performed between semesters since. Unfortunately, during the interim General Instruments® sold the division which produces this chip-set to Microchip® Inc., and they have made the decision to discontinue this product.

Both microprocessors, the CTS256A-AL2 and the SPO256-AL2, are still available at Radio Shack®, but technical information is quite limited. Because they are discontinued products, no one at Microchip® is thoroughly familiar with this chip-set. Though they are eager to be of assistance, the Microchip® personnel are limited to providing only the documentation they have.

¶ PRODUCT DESIGN ¶

The most severe limitation of this unit is a flaw in the internal op-code of the CTS256A-AL2. As pointed out earlier, RAM is allocated to input and output buffers in predefined proportions. Unfortunately, the input buffer is afforded almost all of the available RAM. This is a problem since each character of input will likely produce more that one allophone address of output, and no means of protecting the output buffer from overwrite is available. Since only the input buffer is protected, and since memory allocation cannot be redefined, downloading entire text files must be done slowly enough that the output buffer does not back up. Even then, it is possible that portions of text will be lost.

¶ THE ADDRESS BUS ¶

The recommended circuitry specifies connecting bit D_3 of Port D, (CTS256A-AL2), to pin 5 of U4, the chip select device, (74LS138). Pin 5 of U4 is G2B, which must be at logical lo to enable the chip. This connection disables the chip select device when addressing any address KH such that:

$$(X)7FFH < KH < (X+1)000H$$

thus reducing the available address space within each 4 kilobyte block to 2 kilobytes, (note that D_3 is the MS bit of the LS nibble of the MS byte of the address bus). Note that in this unit, only the external ROM would be affected. This was corrected by simply deleting the connection, and shorting pin 5 to pin 4 on U4.

¶ THE PARALLEL PORT ¶

The recommended circuitry for the parallel port option specified two connections to pin 11 of the DB25 connector, (P2). One from pin 3 of the CTS256A-AL2, and one from pin 5 of U9, (the D-Type flip flop). The author first noticed this problem because it called for direct connection between two TTL outputs. Initially, an OR-gate was added to the circuit to correct the problem. This was in error. Pin 5 of U9 must be connected to pin 10 of P2, (ACK). Otherwise, any attempt to transmit data over the parallel port from the host computer will result in an error message of:

Errors indicate list device may be off-line

Correction of this problem is reflected in the schematic sheet 2 of 3 in the appendices.

¶ The Serial Port ¶

Two design problems were discovered in the serial port circuitry. The first was a lacking invertor between pin 3 of the CTS256A-AL2 and pin 2 of U7, (the line driver). Pin 3 of the CTS256A-AL2 is at logical hi under normal conditions, but the host computer interprets this condition as busy. The signal from pin 3 was already being inverted for the parallel port, so correcting this problem was simply a matter of moving the nodal point of inversion.

It also proved necessary to short pins 4 and 6 on P1 together. When transmitting data from the host computer over a serial port, MS-DOS® sets pin 4 high and tests for a similar condition on pin 6. This fact was not pointed out in the technical literature accompanying the microprocessor.

¶ External ROM ¶

In addition to the problem with the address bus discussed earlier, a problem was detected in the object code required for external ROM. With encoded external ROM in-circuit the unit will not initialize. However, if the op-code is removed from ROM, initialization occurs normally.

General Instrument® supplied a document containing all specifications for their PIC7041 microprocessor, (General Instrument® document DS33001A), including the instruction set. Disassembly of the required object code was attempted in an effort to find the cause of the problem. Due to certain instructions being obscurely defined in the documentation, and due to the interdependence between the external op-code and that already internal to the microprocessor, disassembly was unsuccessful.

When additional documentation on the use of external ROM was requested, it was discovered that three conflicting versions of the op-code were available. The conflicts were at various locations, and all could be attributable to transcription errors, (i.e. one document specifies a B, while another specifies an 8). Unless the appropriate combination of codes can be determined before this report is turned in, no further information on external ROM will be included. If corrections are made, the corrected op-code will be listed in the appendices.

¶ An MS-DOS® Bug ¶

While attempting to get the serial port to function properly, an obscure bug was discovered in the MS-DOS®, (version 4.01), MODE command. Specifically, when specifying UART parameters for a serial port, if 2 stop bits are specified, MS-DOS® will report that it has set the port as such but in fact will have only set it for 1 stop bit. Extensive testing was conducted and this occurred consistently, and was found to be independent of the other parameters specified.

The author discovered the problem with the aid of QAPlus®, which has a feature that allows bit manipulation of the serial port registers. Since there are a variety of such utilities available, and since most of them probably have their own vernacular, a detailed explanation of how to correct for this problem would likely be of little benefit here.

Final Assembly

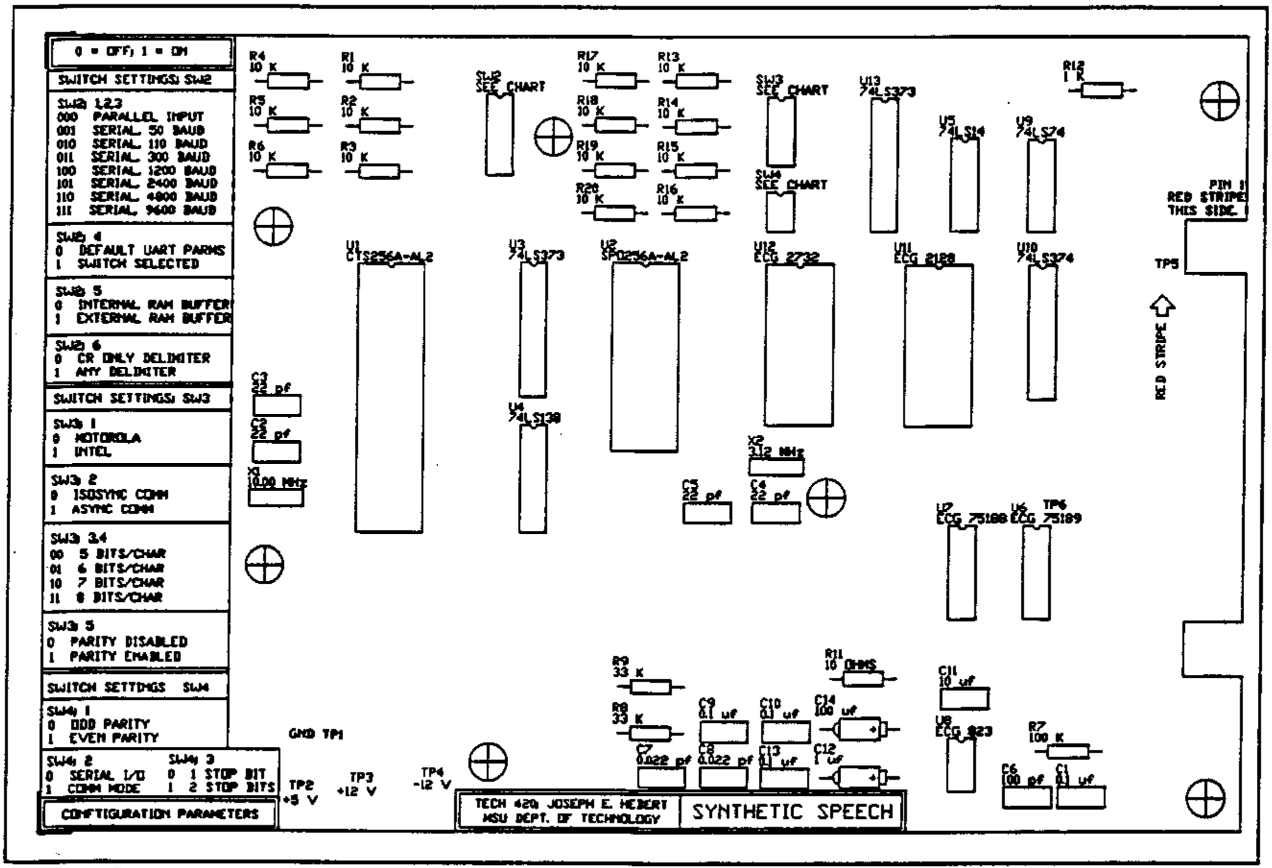


Figure 9: Component layout of main circuit board

Upon completion, the finished circuit board will appear similar to figure 9. The board dimensions are 6" x 9". The wiring harness shown in figure 10 provides all connections to the main circuit board via a 50 pin edge connector. The wiring harness is constructed from a 50 conductor flat ribbon cable.

The main unit will be housed in a standard aluminum BUD box. Figure 11 on the next page shows the general configuration of the main unit housing, and figure 12 shows the critical dimensions to scale.

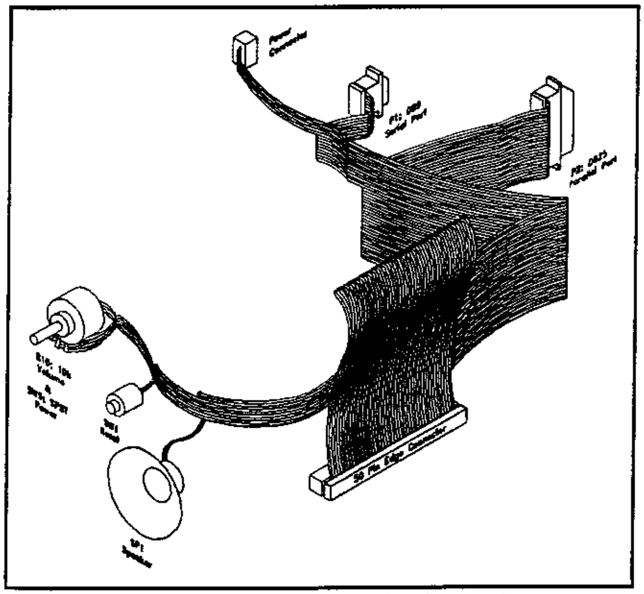
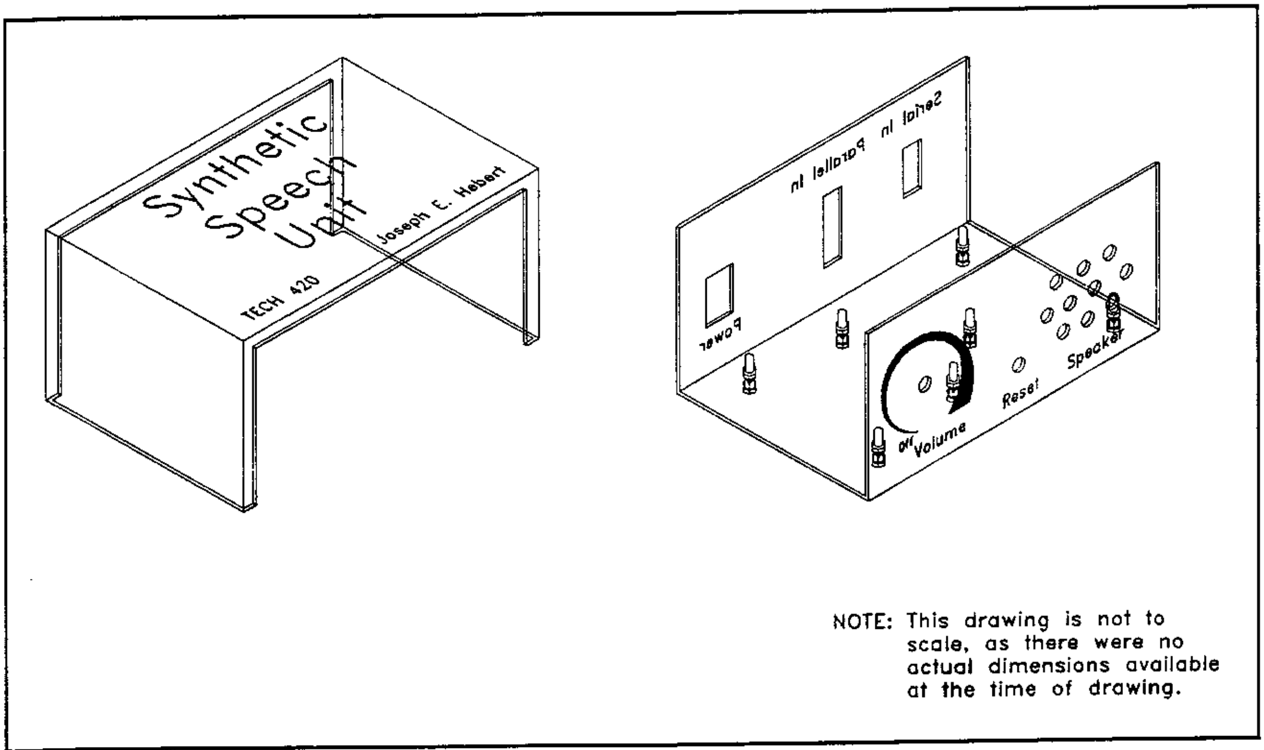


Figure 10: Wiring harness for main circuit board



NOTE: This drawing is not to scale, as there were no actual dimensions available at the time of drawing.

Figure 11: Housing for main circuit board

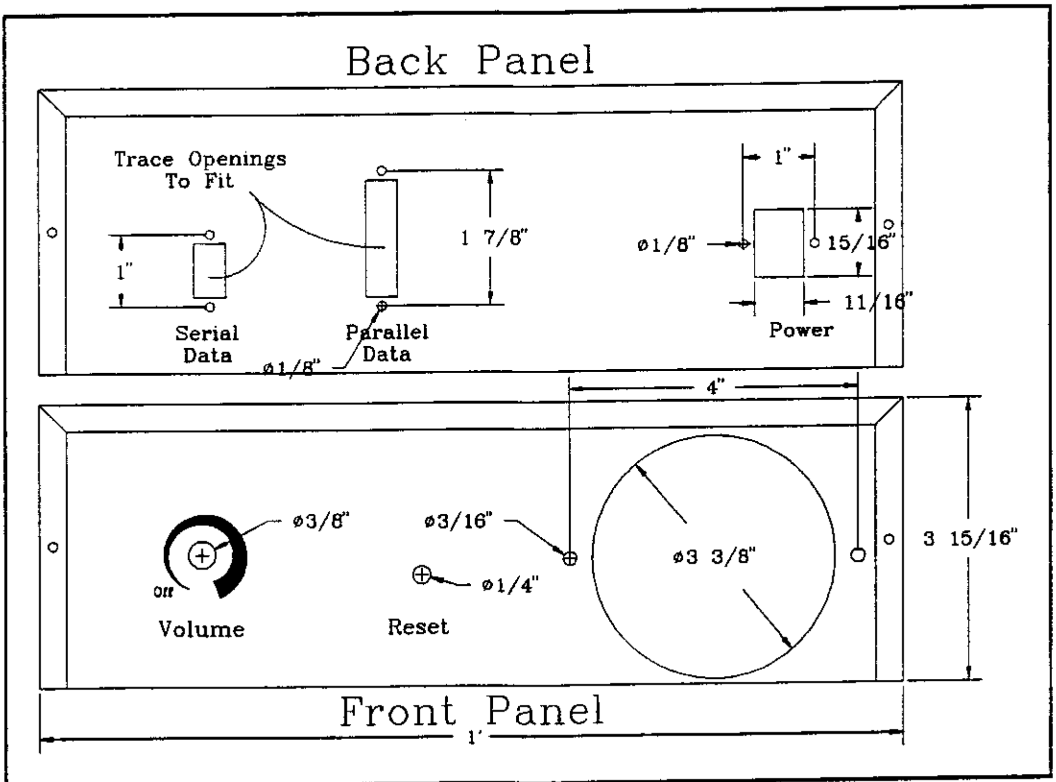


Figure 12: Critical dimensions of housing unit panels, (front and rear)

The main housing, (as well as the power supply housing), will be finished with paint. Since the housings are made of aluminum, they will require scoring before paint can be applied. This scoring can be accomplished by bathing the aluminum surfaces with common household vinegar. Necessary lettering will be applied to the housings by silkscreen process.

Also as a finishing touch, the component layouts will be silkscreened onto both the main circuit board and the power supply circuit board. For additional information on the silkscreening process used in this project, see the chapter entitled CONCLUSIONS AND RECOMMENDATIONS.

Nodal point connections for the wiring harness components are specified in the schematics, which can be found in the appendices.

Conclusions and Recommendations

§ Unit Performance §

This project has been overall successful, despite the inability to make the external ROM function as desired. In general the unit does in fact read text files from the host computer disk(ette), and surprisingly with reasonable fluency. It should be understood that the synthetic speech sounds mechanical, but surprisingly understandable. Nonetheless, for optimum performance certain considerations should be made when configuring the unit for any given application.

If the unit is going to be used to read text files, (as is intended), then serial data transfer should be used. Since overwriting the output buffer is a problem, the slower baud rate available with the serial port is desirable. Also, to prevent buffering long strings of allophone addresses, the *any delimiter* mode is recommended. This leads to the following suggested configuration for reading entire text files:

Serial data transfer

110 Baud Rate

Any delimiter speech mode

External RAM buffer

Using this configuration, the author successfully transmitted several text files of lengths varying from 2 kilobytes to 20 kilobytes, using the MS-DOS® MODE and PRINT commands.

Suppose, on the other hand, a TSR program were written to echo all text sent to the CRT out to the speech unit simultaneously. This would allow the typist to monitor his/her document as it was being typed without relying on the CRT. In such an application, the limiting speed of the data transfer would be the typing speed of the individual, not the data transmission rate. In this case it would be desirable that all data be transmitted as fast as possible, thus freeing the host computer's microprocessor for it's own work. Therefore, such an application as this would call for configuring the unit to accept data through the parallel port.

Such considerations as were mentioned in the preceding paragraphs should be made for any given application of this unit. And to the extent that the configuration parameters are flexible enough to accommodate the same, the primary objective of maximum flexibility has been attained.

Of course, some words are not pronounced accurately due to the absence of external ROM, but surprisingly few. Note however, that if a file is being prepared specifically for speech output, words can be spelled phonetically to improve pronunciation.

It should be said that if this chip-set is used for a specific function, designing in the degree of flexibility found in this unit is unjustifiable. For example, if this unit had been designed specifically for reading text files, it would have been better to design it as an add-in card. This would completely eliminate the need for the extra work of designing and building a special power supply, since all necessary voltages are supplied in each expansion slot. Furthermore, the above recommended configuration could be fixed by the trace pattern, thus eliminating several components. Of course, other requirements would have to be met, but overall the finished board would be much less dense, (i.e. less expensive).

Still, the final conclusion that should be drawn from this project is that, if it were desired to design a synthetic speech unit for any purpose, this chip-set should not be used. This conclusion is made necessary by the fact that the chip-set is no longer in production.

It would require more effort, but developing the op-code to convert text to speech is doable. The basic algorithm could be acquired from the Naval Research Laboratories, and all that would remain to be done would be to apply it to whichever microprocessor the designer prefers, (If following this route, the author would build this unit around a Zilog® microprocessor for cost considerations).

The most difficult development would be developing the actual speech processor. This would require special equipment to excise the needed allophones from digitally stored words. But once done, these allophones could be stored in ROM and output in a manner similar to that of this unit. An added benefit would be the opportunity to use more advanced technology to result in a more *natural* speech output. The Toshiba® family of digital voice products mentioned in the INTRODUCTION chapter of this report has a couple of members which would work well for such an application.

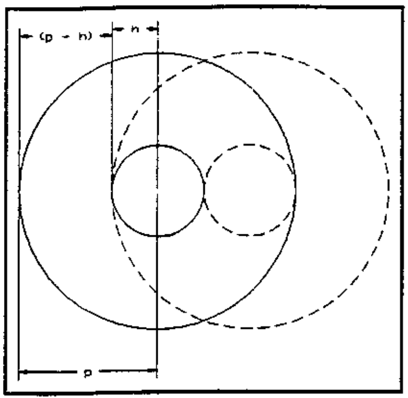


Figure 13: Registration limit

§ DOUBLE SIDED BOARD CONSTRUCTION §

¶ COMPUTER GENERATED TRACE PATTERNS ¶

One problem encountered in constructing this project was trace density. When traces become too dense, it is necessary to divide them into layers. In this case, only two layers were sufficient, but even dividing the traces between two layers presents challenges.

The first challenge to overcome is that of registration, (or alignment). But how critical is registration? Ideally, each pad should line up with its counterpart closely enough that the hole drilled between them is completely surrounded by copper on both sides. Figure 13 shows a one dimensional limit of such a case. If *p* is the radius of the pad, and *h* is the radius of the hole, the maximum offset should be *o* such that:

$$o < (p - h)$$

It must be understood that this is an overly simplified representation, since registration must be made in two dimensions. A more acurate statement of the registration limits would involve two vector displacements;

$$\mathbf{x} = x' - x \text{ and } \mathbf{y} = y' - y$$

such that;

$$o = [\mathbf{x}^2 + \mathbf{y}^2]^{1/2} < (p - h)$$

Nonetheless, in a less quantified statement, this leads to the conclusion that the pads should be as large as is feasible while the holes should be as small as feasible.

There is, however, a catch 22. If the trace pattern is so dense that two sides are necessary, it is highly probable that many traces will need to pass between pads, (this is especially true with DIP, or Dual Inline Package, components). This means that the pads must be kept relatively small. Therefore, it is critical that all holes be as small as possible.

Specialized drill bits can be purchased from a variety of specialty outlets. The author found drill bits through a company named START International[6], in sizes as small as 0.0135" diameter. However, their catalog is $20.00 and they require a minimum order of $50.00.

Even with the smallest holes possible, proper registration cannot be practically achieved without the aid of a computer. Generating a layered trace pattern will require software capable of generating output in layers. One option would be AutoCad®, but a great deal of time would be spent calculating appropriate dimensions, defining blocks, getting trace width's set properly, etc. Overall, it is far better to use software written specifically for electronic fabrication.

The schematic package used in this project was OrCAD®. Most components are predefined in libraries. If, as in the case of this project, specialized components are being used which are not in any of the libraries, OrCAD® has the facility to generate custom libraries. This involves writing source code to define the components and compiling it into a library file. Details of how to write the source code for such a library are left to the OrCAD® manual, but the source file written for this project is contained in the appendices as an example.

[6] START International; 3361 Boyington, Suite 120; Carrollton, TX 75006; 1(800) 527-1809

With the components predefined, drawing a schematic is simply a matter of placing the components and making the appropriate connections. As an added convenience, data pertinent to the schematic can be readily extracted in the form of ASCII text files. For example, generating a parts list requires one command, and it can be executed from a pull down menu via either a mouse or keystroke.

Another type of data file that can be generated from most schematic packages is called a *netlist*. A netlist is a listing of all nodal connections within the schematic. Such information makes testing a finished circuit board for continuity easier by serving as a checklist. However, the true benefit of a netlist is realized when used in conjunction with a trace pattern generating program.

Most of the more sophisticated trace pattern generating programs have a feature called *auto-routing*. One such package is AutoTrax®. The user must first define the outer limits of the circuit board, and then a netlist file, thus enabling AutoTrax® to place the required components on the board and route as many of the traces as possible under the constrictions specified by the operator. Though AutoTrax® often cannot complete very complex boards, it can save many hours of design work.

The drawback is that each software package requires its own format for the netlist. Though netlists can be easily edited with any text editor, doing so may require as much time as placing the traces manually, especially with a well designed package, and will certainly introduce more opportunity for error. AutoTrax® has a module available for drawing schematics and producing netlists in the appropriate format, but this module was not available to the author. For this reason, the author routed the circuit board for this project manually.

The resultant trace pattern consists of three layers, each of which is in the appendices of this report. The top and bottom layers contain the trace patterns for the circuitry, and the top overlay layer provides the component pattern to be silkscreened onto the circuit board.

Registration was accomplished by placing trace lines outside the edge of the circuit board on all layers of the trace pattern, and aligning these with the edges of the photosensitive copper-clad board before exposure. Though this technique does not produce exact registration, it is more than sufficient to meet the tolerance requirements specified previously.

¶ DEVELOPING AND ETCHING DOUBLE SIDED BOARDS ¶

Once the trace pattern has been generated, and photographically transferred to a copper-clad board, it remains to develop and etch the pattern into physical traces. It may seem necessary to have the freshest possible developer and etchant for this task, but such is not the case.

No quantified measurements were made, but figure 14 is a good representation of what was observed qualitatively. The etch resist will develop in a basically linear fashion, and the slope of each line represents a rate of development. Of course, fresh developer will produce the fastest development time.

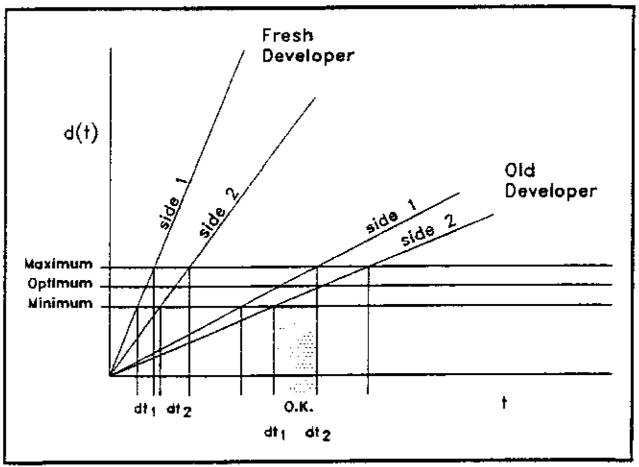


Figure 14: Development of etch-resist as a function of time

Unfortunately, both sides of the board will not develop at the same rate, (represented in figure 14 as the angular displacement between the slopes of the two sides). Fresh developer produces not only the fastest development rate, but also the greatest rate difference between the two sides. As represented in figure 14, this results in one side being overdeveloped before the other side is sufficiently developed.

By using old developer, the rate of development is slowed, allowing a greater margin of error. But more significantly, the rate difference between the two sides is decreased. This allows for adequate development anywhere within the shaded area of figure 14.

If only fresh developer is available, it is best to simply develop the etch-resist away from a few smaller, (i.e. less expensive), boards before attempting to develop a double sided trace pattern. Even if unintentional, the same is going to happen in failed attempts. Purposely weakening the developer with cheaper boards can not only save money, but untold frustrations as well.

Balanced development of the etch-resist is critical, especially when thin traces are involved. The etch-resist coating will not guard against etching indefinitely. The etchant will react with the edges of the protected copper resulting in an undercutting effect. Since the side that is most developed etches fastest, a time differential in etching occurs. If the imbalance is too great, the traces on one side will be etched through. Imbalanced etching can be prevented by achieving balanced development. It is not, however, possible to correct for imbalances in development during etching.

¶ VIAS AND THRU-CONTINUITY ¶

Proper registration of the trace patterns was discussed earlier, but no mention was made as to the means of providing thru-continuity, (or continuity from one side to the other). Clearly this must be accomplished, but doing so proved a challenge.

With components such as resistors and capacitors this is no real problem. It requires simply soldering the component leads on both sides of the board. On the other hand, integrated circuits pose a problem. Soldering integrated circuits requires expensive, temperature regulated, soldering stations, and is tricky even then. To prevent damaging ICs, the author prefers to mount DIP sockets. But low profile DIP sockets prevent soldering on the top side.

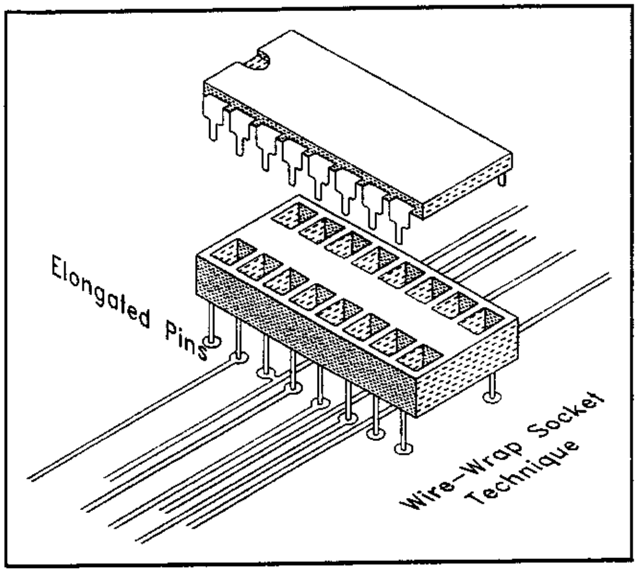


Figure 15: Wire-wrap socket technique

One solution to this problem is shown in figure 15. Wire-wrap sockets have elongated pins which can accommodate soldering on both sides of the circuit board. But they also introduce a new problem. The elongated pins, after soldering, are effective antennae inviting RFI and EMI into the circuitry. In many circuits this may not be a problem, but a microprocessor circuit with a clock speed of 5 MHz would be extremely sensitive to such interference. Also, such a technique is unsightly, and that is a valid consideration.

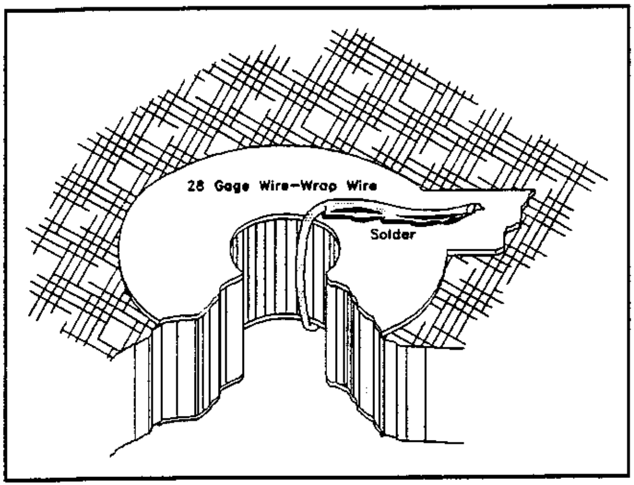


Figure 16: Wire-wrap wire technique

As an alternative, the author turned again to wire-wrap technology, but this time to the wire used with a wire-wrap socket.

Wire-wrap wire is very thin, (30 gage), and has a very high tensile strength. Furthermore, low profile DIP sockets do not have round, wire shaped leads. Instead, they have flat leads. By inserting a strand of wire-wrap wire through the hole and spot-soldering it to the top side of the board, ample room is left for the socket leads to be inserted through the hole and soldered to the back. The wire is strong enough that it can be pulled tight, removing slack from inside the hole, without fracture. This is the technique used in the construction of this project, and the author found it to be quite successful.

Another application of this wire-wrap wire technique involves vias. Vias are the pads that are only for thru-continuity. No components are soldered to vias. Of necessity, vias are smaller than component pads. This is due to the density of the traces. Recall the limiting factors in the trace pattern registration. Because vias are necessarily smaller than component pads, even smaller holes are required.

Using normal size holes can be done, and connection can be made with small pieces of component leads. But this results in an unsightly circuit board with stubs protruding on both sides. Also, the additional heat required to solder to the lead stubs causes unnecessary discoloration of the circuit board. By using smaller holes, and wire-wrap wire to make the connection, the protruding stubs are eliminated. The solder simply builds up into a small dome shape covering the wire. And, since the wire-wrap wire has less mass to heat, board discoloration is kept to a minimum.

§ SILKSCREEN PRINTING §

As a finishing touch to the circuit board, the component layout can be silkscreened onto the component side. This not only gives the circuit board a better appearance, it affords the designer an opportunity to provide information in a convenient location, (i.e. The switch setting information for this project).

Furthermore, silkscreened control labels can provide a better finished appearance for the unit housing. And, as is the case for this project, silkscreening can even make a report cover more impressive. But for all the benefits to be derived from silkscreening, few people seem to know just how easily it can be realized.

There are many products available for a variety of silkscreening processes, but the simplest seems to involve the following items:

Screen mesh

Ulano® Blue Poly® presensitized emulsion film

Ulano® A&B developer

Ink

Also, a frame for the mesh and a squeegee for the ink are required, but can probably be constructed with materials on hand.

For production runs of hundreds of items, a commercial grade frame is a necessity. But for a one or two item run, a homemade wooden frame will suffice. It simply needs to maintain uniform tension on the screen in all directions. The squeegee must have a smooth edge, free of nicks, to assure uniform ink application. For best results, it should be long enough to span the width of the image being printed. Commercially available squeegees are inexpensive enough that if the materials needed to construct one are not on hand, purchasing them would be just as expensive.

The ink must be purchased for the intended purpose. Various inks are available for various surfaces. A special ink is required for printing on circuit boards, and still another for a vinyl report cover. There are even special etch-resist inks available for silkscreening trace patterns onto copper-clad boards. Such an alternative may be quite useful for producing many identical boards.

There are many suppliers of silkscreen equipment, and it may even be possible to purchase the needed supplies from a local shop which does silkscreening. Nonetheless, The author purchased all items needed for silkscreening from Texas Screen Process Supply, Inc[7].

The process involving the items listed above is quite simple. First, the designer produces a positive transparency image of the artwork to be printed, just as (s)he would for a trace pattern. A piece of Blue Poly® is removed from the roll and the transparency is placed over it. A plate of glass placed over the film/transparency is needed to provide good contact. The film is then exposed to bright light, a pair of flood lamps is ideal, for approximately two minutes. If the transparency is opaque enough, a longer time will not hurt.

During exposure, any film not protected by the transparency pattern cures, or hardens. The unexposed areas can then be washed away in developer. The developer comes in component form, and must be prepared as needed. If, however, the prepared developer can be stored in an opaque container and protected from light, it will keep for some time.

Once the film is developed, the cellophane backing can be removed and the film can be affixed to the screen, (or mesh). The screen must already be stretched in a frame. Printing is simply a matter of placing the frame in contact with the surface to be printed on, and then applying the ink by smearing it over the image with a squeegee. The image is comprised of those areas of the film that were washed away during developing. Since no film blocks the passage of ink through the screen in these areas, the ink is deposited on the printing surface in the form of the image. The screen

[7] Texas Screen Process Supply, Inc.; 304 N. Walton Street; Dallas, TX 75226; 1(800) 366-1776; FAX: 1(214) 741-6527

should be chosen according to the desired resolution. A greater number of holes per inch in the screen is analogous to a greater number of dots per inch, (DPI), for a computer printout.

§ PROJECT COST §

No.	Item	Quantity	Unit Price	Cost
1	CTS256A-AL2	1	$ 16.95	$ 16.95
2	SPO256-AL2	1	12.95	12.95
3	74LS373	2	1.19	2.38
4	74LS138	1	1.19	1.19
5	74LS14	1	0.99	0.99
6	74LS74	1	0.99	0.99
7	74LS374	1	1.19	1.19
8	ECG 2128	1	13.10	13.10
9	ECG 2732	1	20.00	20.00
10	ECG 823	1	2.10	2.10
11	ECG 75188	1	2.03	2.03
12	ECG 75189	1	2.14	2.14
13	ECG 966	1	5.15	5.15
14	ECG 967	1	5.15	5.15
15	ECG 970	1	15.20	15.20
16	ECG 5318	1	3.06	3.06
17	ECG 5259A (Zener diodes)	2	12.69	25.38
18	10.000 MHz Crystal	1	1.45	1.45
19	3.120 MHz Crystal	1	4.95	4.95
20	6 Station DIP Switch	1	2.20	2.20
21	5 Station DIP Switch	1	2.12	2.12
22	3 Station DIP Switch	1	1.53	1.53
23	Resistors 10 kΩ	14	0.25	3.50
24	Resistors 33 kΩ	2	0.25	0.50
25	Resistors 100 kΩ	1	0.25	0.25
26	Resistors 1 kΩ	1	0.25	0.25
27	Resistors 100 Ω	1	0.25	0.25
28	Resistors 10 Ω	1	0.25	0.25
29	Resistors 1 Ω/3 W	3	0.35	1.05
30	Potentiometer 10 kΩ (w/SPST switch)	1	5.25	5.25
31	Potentiometer 5 kΩ (PC Mount)	1	2.15	2.15
32	Capacitors 0.022 pf	2	0.25	0.50
33	Capacitors 22 pf	4	0.29	1.16
34	Capacitors 100 pf	1	0.25	0.25
35	Capacitors 0.1 µf	4	0.25	1.00
36	Capacitors 1 µf	1	0.25	0.25
37	Capacitors 10 µf	1	0.25	0.25
38	Capacitors 100 µf	4	0.25	1.00
39	Capacitors 0.33 µf	3	0.25	0.75
40	Capacitors 1000 µf	1	0.50	0.50
41	PicoFuses 125 V, 3 A	4	N/A	3.55
42	BUD Box 12" x 7" x 4"	1	19.00	19.00
43	BUD Box 6" x 5" x 4"	1	8.40	8.40
44	Plugs	2	2.19	4.38
45	Sockets	2	1.28	2.56
46	Power Cord	1	4.80	4.80
47	Transformer	1	17.55	17.55
48	Ribbon Cable	1	5.75	5.75
49	50 Pin Edge Connector	1	4.09	4.09
50	DB9 Connector	1	3.95	3.95
51	DB25 Connector	1	3.95	3.95
52	Speaker 8Ω	1	5.50	5.50
53	Pushbutton, momentary	1	2.69	2.69
54	Assorted IC Sockets	Varies	Varies	10.00
55	GC Pre-sensitized Copper Clad Board 7" x 11"	1	11.95	11.95
56	15% Mark up for approximation errors and miscellaneous items			40.41
			Total Cost, (less tax):	$ 309.84

Table 6

NOTE: Items that were on-hand are priced by approximation only.

Appendices

APPENDIX A:

References

General Instruments, Inc. General Instrument® Product Data. published in-house, *unknown.*

Microchip Inc. Microchip® DS33001A. Also cited as General Instrument® DS33001A. published in-house, *unknown.*

Tandy, Inc. CTS256A-AL2 Technical Data. published in-house, 1984.

Tandy, Inc. SPO256-AL2 Technical Data. published in-house, 1984.

APPENDIX B:

Schematics

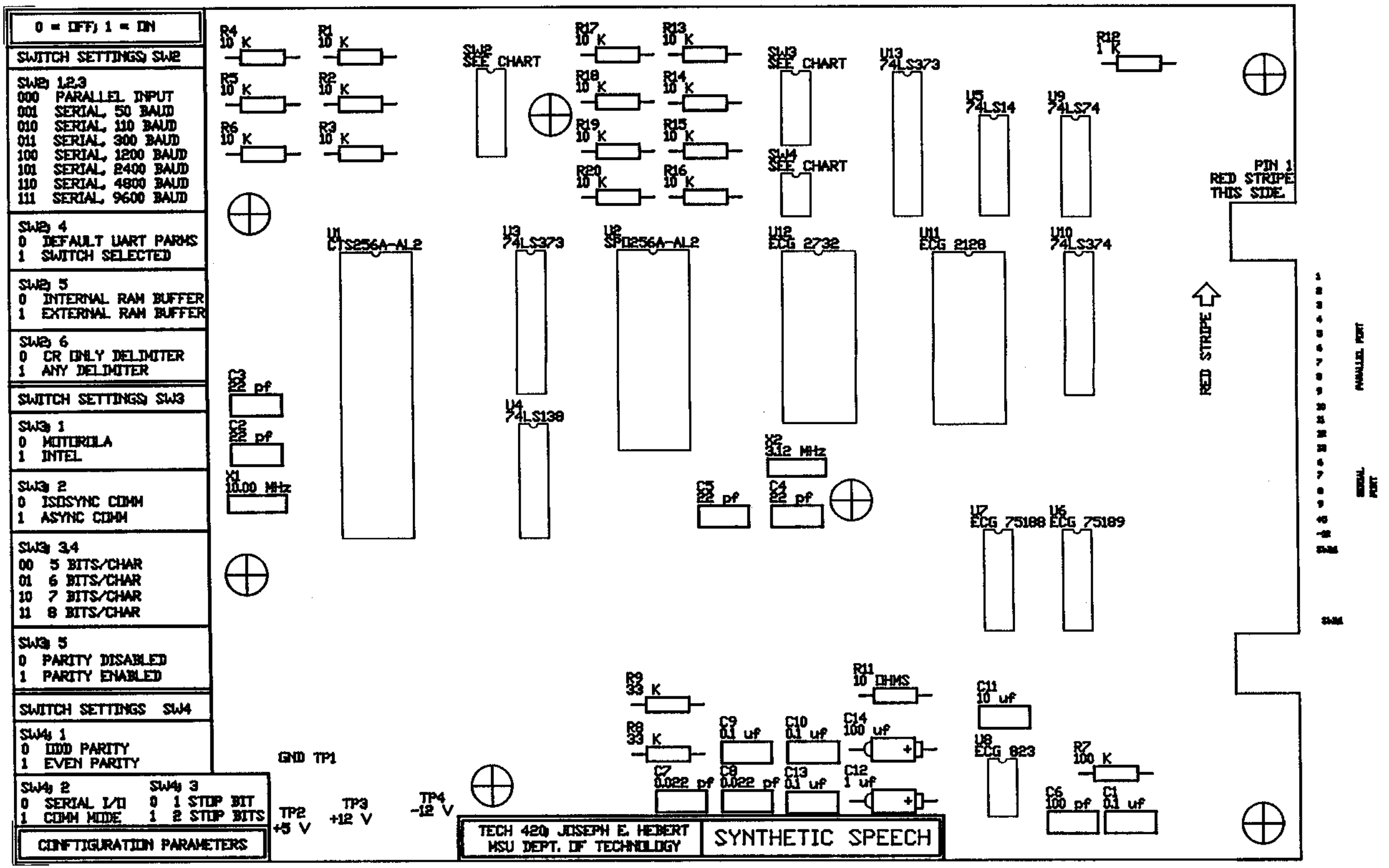
This appendix contains the full set of schematics which comprise the unit outlined in this report. They consist of three fold-out B-size plots.

Pages 58, 59 & 60 are not included in this digital copy of this report. They are schematics which were plotted on B-size (11" x 17") paper stock and were included in the hard bound copies of this report as foldout pages. Because of their dimensions they can not be scanned with any of my equipment, and are thus not included. Relevant portions of the schematics are embedded in the text of this report where applicable.

APPENDIX C:

Trace Patterns and Component Overlays

This appendix contains all trace patterns and overlays used in this project.



0 = OFF; 1 = ON

SWITCH SETTINGS; SW2

SW2; 1,2,3
000 PARALLEL INPUT
001 SERIAL, 50 BAUD
010 SERIAL, 110 BAUD
011 SERIAL, 300 BAUD
100 SERIAL, 1200 BAUD
101 SERIAL, 2400 BAUD
110 SERIAL, 4800 BAUD
111 SERIAL, 9600 BAUD

SW2; 4
0 DEFAULT UART PARMS
1 SWITCH SELECTED

SW2; 5
0 INTERNAL RAM BUFFER
1 EXTERNAL RAM BUFFER

SW2; 6
0 CR ONLY DELIMITER
1 ANY DELIMITER

SWITCH SETTINGS; SW3

SW3; 1
0 MOTOROLA
1 INTEL

SW3; 2
0 ISOSYNC COMM
1 ASYNC COMM

SW3; 3,4
00 5 BITS/CHAR
01 6 BITS/CHAR
10 7 BITS/CHAR
11 8 BITS/CHAR

SW3; 5
0 PARITY DISABLED
1 PARITY ENABLED

SWITCH SETTINGS SW4

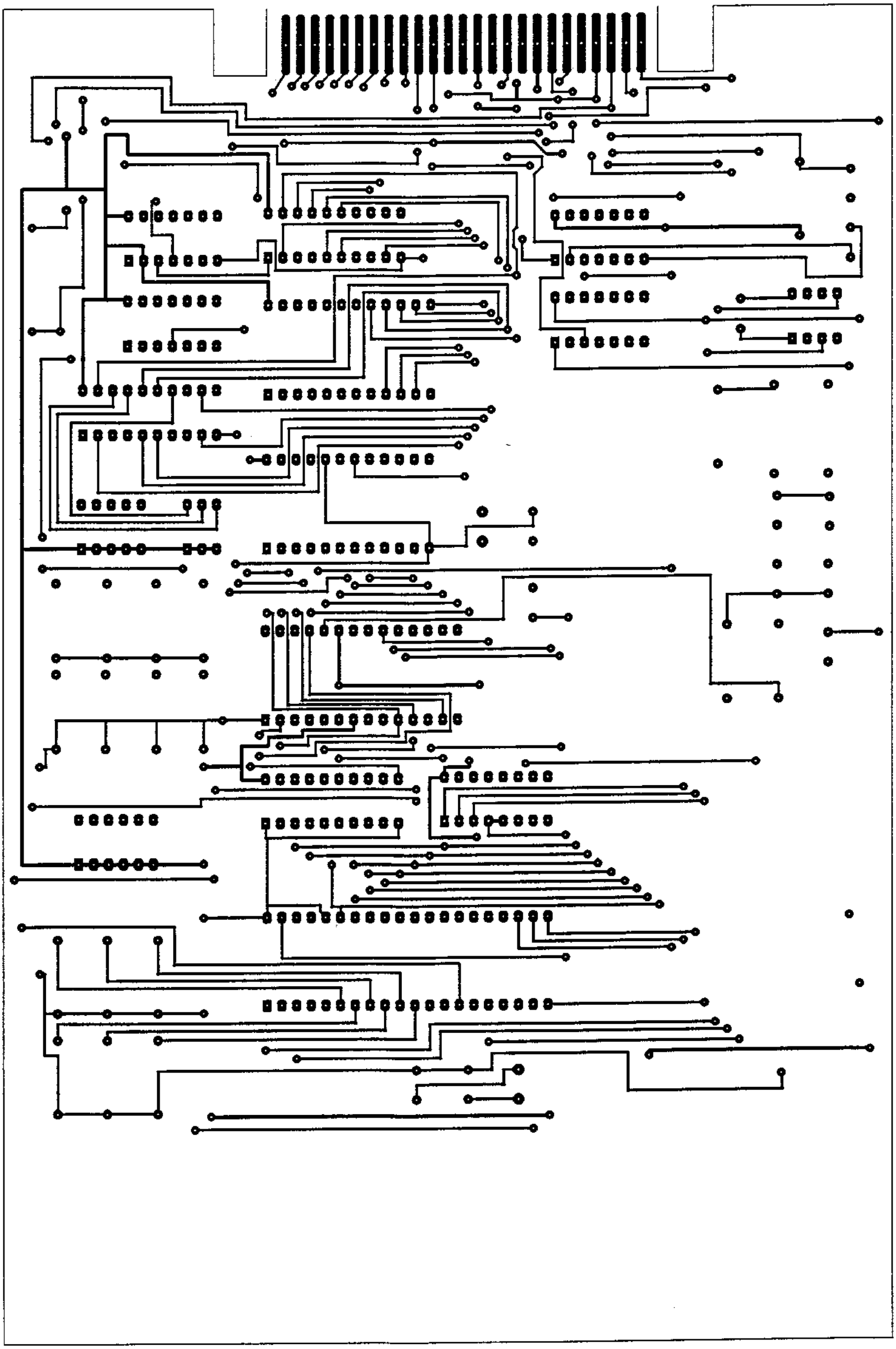
SW4; 1
0 ODD PARITY
1 EVEN PARITY

SW4; 2
0 SERIAL I/O
1 COMM MODE

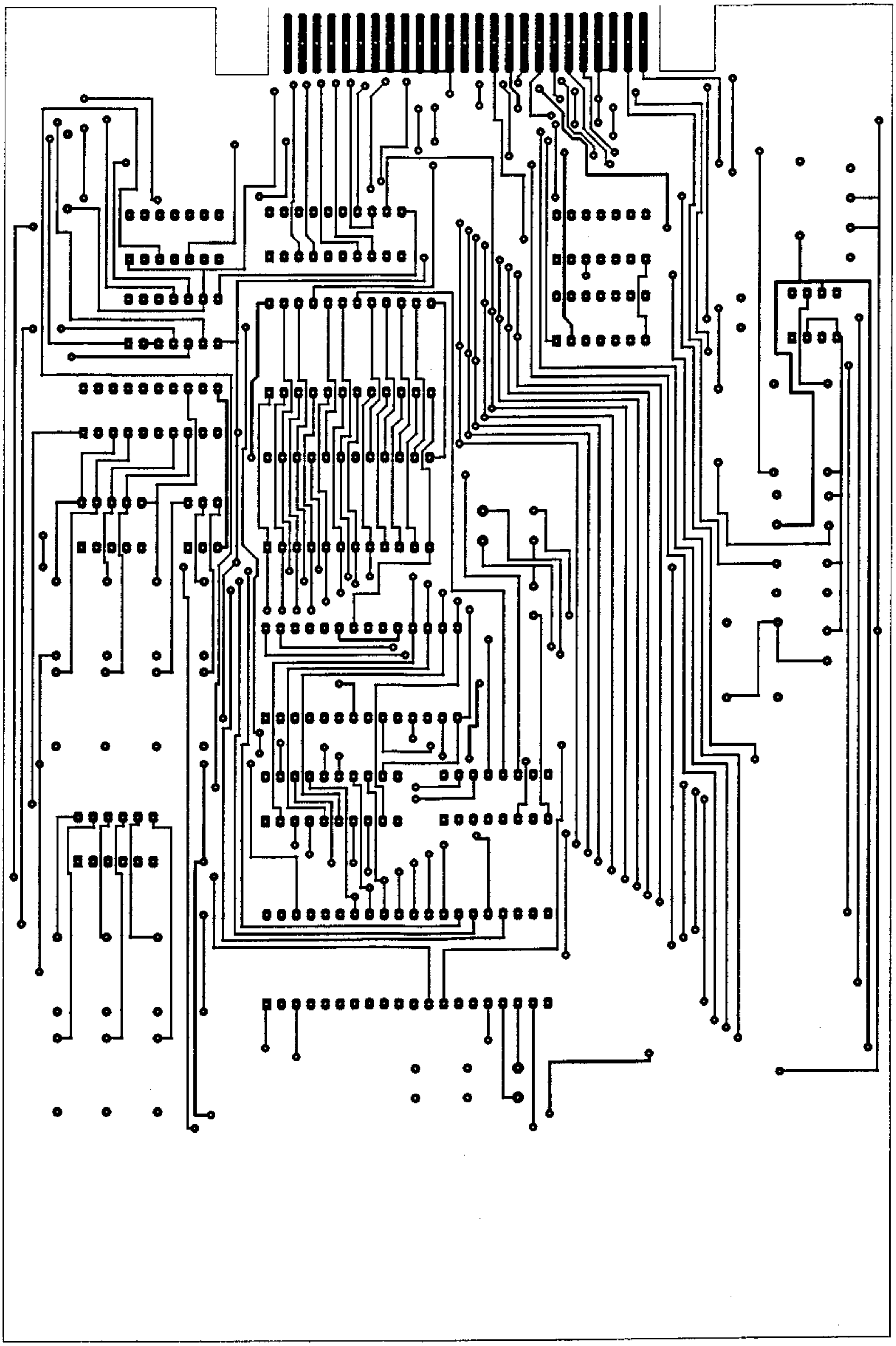
SW4; 3
0 1 STOP BIT
1 2 STOP BITS

CONFIGURATION PARAMETERS

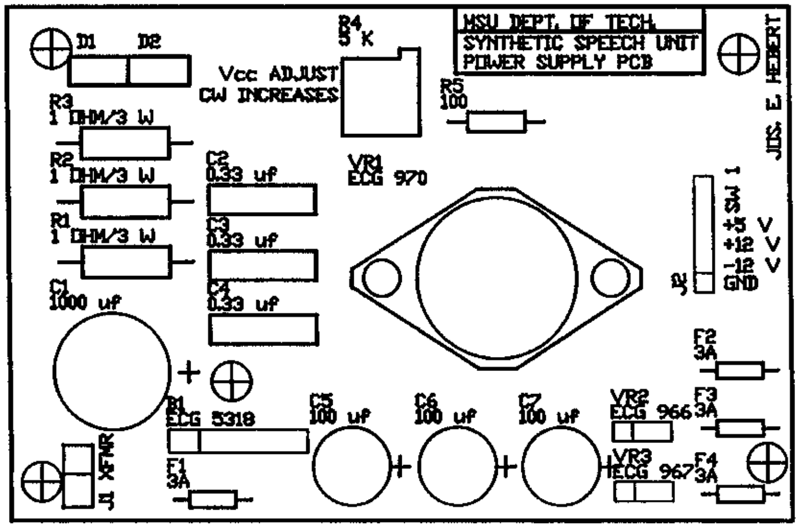
PROJECT Top Overlay



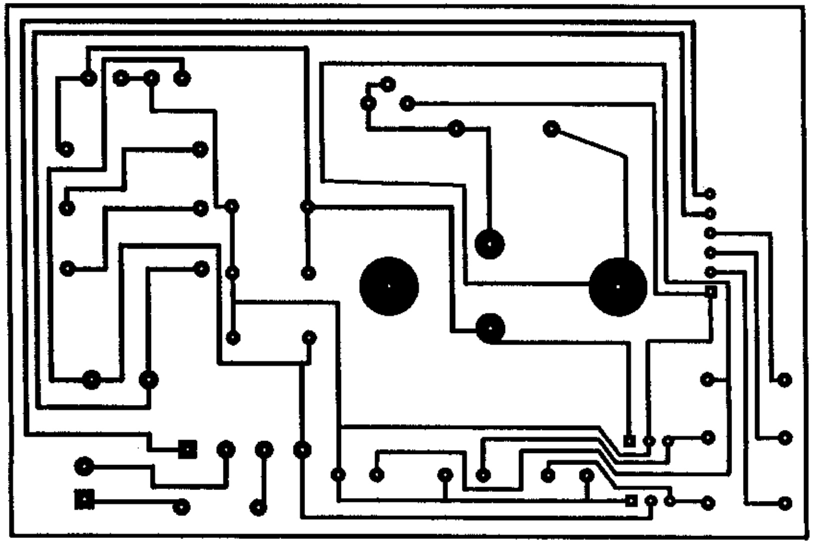
PROJECT Top Layer



PROJECT Bottom Layer



420PROJC Top Overlay



420PROJC Bottom Layer

APPENDIX D:

Source Code for OrCad® Library

The following is a listing of the source code needed to create the specialized OrCad® component library necessary for this project. Details of how to write OrCad® library code are left to the OrCad® manual, and are not offered in this report. The code contained in this appendix is sufficient for duplication of this project, and can be compiled as is.

```
{This library is for special components needed in TECH 420.}
PREFIX
END
'CTS256A-AL2’ {This is a text to allophone address converter}
15   22   1
L1    1   I/O   'B5/R/W’
L2    2   I/O   'B7’
L3    3   I/O   'B0’
L4    4   I/O   'B1’
L5    5   I/O   'B2’
L6    6   I/O   'A0’
L7    7   I/O   'A1’
L8    8   I/O   'A2’
L9    9   I/O   'A3’
L10  10   I/O   'A4’
L11  11   I/O   'A7’
L12  12   I/O   'INT 3’
L13  13   I/O   'INT 1’
L14  14   I/O   'RESET’
L15  15   I/O   'A6/SCLK’
L16  16   I/O   'A5/RXD’
L17  17   I/O   'XTAL2/CLOCKIN’
L18  18   I/O   'XTAL1’
L19  19   I/O   'D7’
L20  20   I/O   'D6’
R1   40   IN    'Vss’
R2   39   I/O   'B6/ENABLE’
R3   38   I/O   'B4/ALATCH’
R4   37   I/O   'B3/TXD’
R5   36   I/O   'MC’
R6   35   I/O   'C7’
R7   34   I/O   'C6’
R8   33   I/O   'C5’
R9   32   I/O   'C4’
R10  31   I/O   'C3’
R11  30   I/O   'C2’
R12  29   I/O   'C1’
R13  28   I/O   'C0’
R14  27   I/O   'D0’
R15  26   I/O   'D1’
R16  25   IN    'Vcc’
R17  24   I/O   'D2’
R18  23   I/O   'D3’
R19  22   I/O   'D4’
R20  21   I/O   'D5’
{Comments: The above microprocessor is a text to allophone address converter.
It is Radio Shack Cat. No. 276-1786. It utilizes a built-in algorithm
that converts ASCII text into allophone addresses compatible with the
SPO256A-AL2 Speech Processor to be defined next.}
’SPO256A-AL2’
15   16  1
L1   1   IN     ’Vss’
L2   2   I/O    ’RESET’
L3   3   I/O    ’ROM DISABLE’
L4   4   I/O    ’C1’
L5   5   I/O    ’C2’
L6   6   I/O    ’C3’
L7   7   IN     ’Vdd’
L8   8   I/O    ’SBY’
L9   9   I/O    ’L\R\Q\’
L10  10  I/O     ’A8’
L11  11  I/O     ’A7’
L12  12  OUT       ’SER OUT’
L13  13  I/O     ’A6’
L14  14  I/O     ’A5’
R2   28  OUT       ’OSC 2’
R3   27  OUT       ’OSC 1’
R4   26  I/O     ’ROM CLOCK’
R5   25  OUT       ’S\B\Y\ \R\E\S\E\T\’
R6   24  OUT       ’DIGITAL OUT’
R7   23  IN      ’Vd1’
R8   22  IN      ’TEST’
R9   21  IN      ’SER IN’
R10  20  I/O     ’A\L\D\’
R11  19  I/O     ’SE’
R12  18  I/O     ’A1’
R13  17  I/O     ’A2’
R14  16  I/O     ’A3’
R15  15  I/O     ’A4’
{Comments: The preceding microprocessor is a Speech Processor, Radio
Shack Cat. No. 276-1784. By addressing built-in allophones, a digital
output is generated which can be converted to analog by an external
RC circuit.}
{The next IC defined is a 2 kilobyte RAM chip.}
’TMS4016-25’
8    13  1
T2   18  I/O    ’ ’
T6   24  IN     ’ ’
L2   21  I/O    ’ ’
L4   9   I/O    ’D0’
L5   10  I/O    ’D1’
L6   11  I/O    ’D2’
L7   13  I/O    ’D3’
L8   14  I/O    ’D4’
L9   15  I/O    ’D5’
L10  16  I/O    ’D6’
```

```
L11  17  I/O   'D7’
B2   12  IN    ' '
B6   20  IN    ' '
R1   8   I/O   'A0’
R2   7   I/O   'A1’
R3   6   I/O   'A2’
R4   5   I/O   'A3’
R5   4   I/O   'A4’
R6   3   I/O   'A5’
R7   2   I/O   'A6’
R8   1   I/O   'A7’
R9   23  I/O   'A8’
R10  22  I/O   'A9’
R11  19  I/O   'A10’
{The following IC is an audio power amplifier. It is ECG’s part no.
ECG823. It is an 8 pin ic chip.}
'ECG823’
9     10      1
L1    1       I/O     'GAIN’
L3    2       IN      '-IN’
L6    3       IN      '+IN’
L9    4       IN      'GND’
R1    8       IN      'GAIN’
R3    7       IN      'BYPASS’
R6    6       IN      'VS’
R9    5       OUT     'OUT’
```