Sixth Quarterly Progress Report

December 26, 1984 through March 26, 1985

NIH Contract N01-NS-2356

Speech Processors for Auditory Prostheses

Prepared by

Blake S. Wilson, Charles C. Finley and Dewey T. Lawson

Neuroscience Program Office
Research Triangle Institute
Research Triangle Park, NC 27709

CONTENTS

I. Introduction

The purpose of this project is to design and evaluate speech processors for auditory prostheses. Ideally, the processors will extract (or preserve) from speech those parameters that are essential for intelligibility and then appropriately encode these parameters for electrical stimulation of the auditory nerve. Work in this quarter included the following:

1. Psychophysical stuides with an implant patient at the University of California at San Francisco (UCSF), mainly to obtain basic measures of performance with a patient fitted with the UCSF transcutaneous transmission system and to confirm that all hardware and software components of the present RTI testing facility work according to design;

2. Development of a portable, real-time speech processor appropriate for use with single-channel auditory prostheses;

3. Development of software for support of the RTI patient stimulator;

4. Further development of software for support of basic psychophysical studies and speech testing;

5. Continued preparation for the first implant patient at Duke University Medical Center (DUMC), including (a) coordination of parallel efforts at UCSF and Storz Instrument Company in St. Louis, (b) completing the construction of a laboratory at DUMC that is functionally identical to the laboratory we now have in place at UCSF, (c) presentation of the technical considerations in selecting auditory prostheses for implant candidates, for review by the board of North Carolina Blue Cross/Blue Shield, and (d) review of candidates for cochlear implants at Duke;

6. Establishment of a new collaboration with workers at the Washington University Medical Center and Central Institute for the Deaf, who plan to work with us and the UCSF team in the design and evaluation of speech processors for auditory prostheses.

In this report we will describe the activities indicated in points 1-4 above. Our ongoing collaboration with Duke (point 5) has been described in previous quarterly reports, and our new collaboration with the St. Louis group (point 6) will be described in future reports once the program there is fully under way.

II. Patient Tests

Several experiments were performed at UCSF in mid-March, 1985, with patient EHT, who is fitted with the four-channel, transcutaneous transmission system designed by the UCSF team. This patient had been the subject of intensive studies in the previous year using both the transcutaneous and percutaneous systems for transmission of stimuli to his implanted electrode array. The main objectives of our studies with EHT were the following:

1. Obtain basic psychophysical measures of EHT's performance with the transcutaneous system, including measures on all channels of thresholds to pulses of various waveforms and durations, measures of temporal discrimination on three channels, measures of the loudnesses of pulses of various amplitudes and durations for two channels, measures of loudness matches for suprathreshold pulses of various waveforms and durations for two channels, measures of "pitch" and "sharpness" comparisons for suprathreshold pulses of various waveforms and durations for three channels, measures on all channels of thresholds to bursts of filtered noise (filter break frequencies were 2.0 and 6.0 kHz, with 4th order skirts beyond each break frequency), measures of the time for "decay" or "extinction" of initial percepts when the filtered noise was presented continuously, and measures of loudness difference limens (DLs) for pulses delivered alone and for pulses superposed on a continuous background of "extinguished" bandpass noise;

2. Obtain measures on the repeatability of measurements made with the transcutaneous system, including measures of the effects on apparent pulse thresholds of manipulations in the positioning of the antenna array and in the connections between rf modulators and "unused" coils in the antenna array;

3. <u>Compare psychophysical measures of EHT's performance with the transcutaneous system to previous measures of his performance with the percutaneous system</u>, and have available a "baseline" of data for the transcutaneous system for comparisons with future implant subjects;

4. <u>Evaluate certain hypotheses that relate to the design of "stimulus primitives</u>," using some of the basic psychophysical measures listed above;

5. <u>Simulate the present UCSF speech processor with the block-diagram compiler to confirm that speech-testing results obtained with the block-diagram compiler and hardware interface are essentially identical to the results obtained with the analog processor</u>; and

6. <u>Confirm that all hardware and software components of the RTI testing facility work according to design and are "ready to go" for the next implant patient</u>.

All of these objectives were met with the exception of objective 5. This objective was not realized because we had only eight, 5-hour days for completing the tests with EHT and we simply ran out of time. However, we did meet the other objectives, including confirmation that all hardware and software components (with the exception of the block-diagram compiler simulation) of the RTI testing facility at UCSF work according to design and are ready for the next implant patient, to be intensively studied by the UCSF/RTI team this June and July. To conserve time during testing sessions with the next and subsequent patients, we have automated many of the psychophysical procedures used to obtain the data indicated in point 1 above. In addition, we are in the process of making side-by-side comparisons of outputs and intermediate waveforms produced with (1) the present analog UCSF speech processor and (2) the simulation of this processor using the block-diagram compiler system. So far, no discrepancies (other than very minor differences that would be expected between an analog system and a digital simulation of it) have been discovered between waveforms produced by the two processors, and we expect that the block-diagram compiler system will also be "ready to go" for this next patient.

Full presentation of the results obtained from the measurements listed under points 1 and 2 above is deferred for now but will appear in our next quarterly report. We expect to complete tests with the present patient at UCSF in the upcoming quarter (see section V and Appendix 2), and therefore plan to present comparisons of results obtained with the the transcutaneous transmission system (from patient EHT) with results obtained with the percutaneous transmission system (from the present patient and from EHT).

III. Development of Portable, Real-Time Hardware

We have developed a portable, real-time speech processor appropriate for use with single-channel auditory prostheses. The main objectives of this effort were to (1) demonstrate that the fundamental frequency (F_0) of voiced speech sounds could be reliably extracted with a low-power processor for both noisy and quiet acoustic environments; (2) demonstrate that this processor could reliably mark and code the boundaries between voiced, unvoiced and silent intervals in running speech, in the same acoustic environments; (3) provide a "building block" for multichannel speech processors in which signals representing excitation of the vocal tract are coded separately from signals representing the "short-time" configuration of the vocal tract; (4) provide a working hardware system for implementing other promising strategies in a portable unit; and (5) make a prototype processor to provide speech input that is largely complementary to the input provided by information available in lipreading, primarily for application in extracochlear prostheses for infants and young children (after full evaluation of this and competing coding strategies with adults).

To meet these objectives we designed a portable processor based on the CMOS ("Complementary Metal Oxide Semiconductor," a low-power technology for integrated circuits) version of the INTEL 8031 microcontroller. This microprocessor has a 1 microsecond instruction cycle and on-chip peripherals that facilitate its use in a low-cost, battery-powered processor for real-time analysis of speech.

A block diagram of the current configuration of the hardware is shown in Fig. III.1. The hardware consists of four main sections: the analog section for bringing speech from the environment to the input of an analog-to-digital converter (ADC); the ADC itself; the microcontroller section with memory; and a digital-to-analog converter (DAC) for output to the electrode driver(s). Under construction are two variations of this basic configuration, both to increase "processing throughput" with either the addition of another 8031 or a CMOS 12x12 bit multiplier (one of the ADSP-1000 series of multipliers made by Analog Devices, Inc.). These additional devices are not required for the present processing strategy, but may be required for more complex strategies such as those that might be used for multichannel prostheses. The power consuption of the present processor is about 70 mW for quiet environments, where not much current is drawn by the

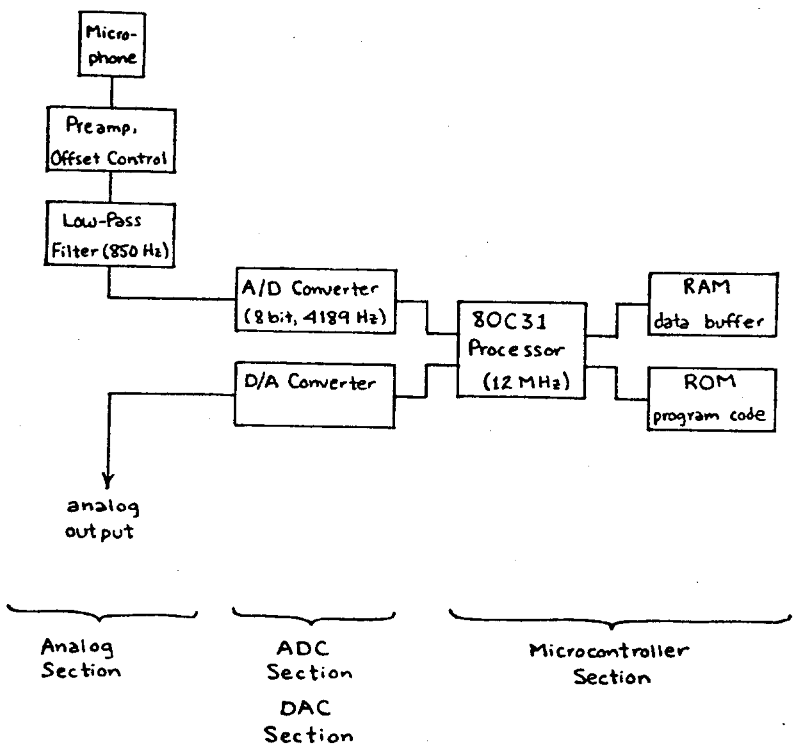


Fig. III.1. Block diagram of the 80C31-based processor. See text for details.

analog section, and about 74 mW when intense noise and speech are present at the microphone. At these power levels the processor will run continuously for several days on a 5-volt NiCad battery without recharging. [This estimate neglects, of course, the additional power required to drive prosthesis electrode(s).]

Also compatible with the objective of a portable unit is the small size of the instrument. A photograph of the prototype is presented in Fig. III.2. Even with the low density construction shown, the entire processor easily fits on a 12 x 9 cm board. Improved packaging could easily reduce the size of the processor to that of a pack of cigarettes.

In addition to portability, another important objective of our effort was to extract a reliable and accurate representation of F_0 for voiced speech sounds. We selected the "Average Magnitude Difference Function" (AMDF) algorithm (Ross et al., 1974; Sung and Un, 1980; Un and Yang, 1977) because its computational complexity is relatively modest and its performance is robust in noisy acoustic environments (Paliwal, 1983). In our implementation of this algorithm the AMDF output is further processed for median smoothing, detection of erronous indications of F_0, and detection and signalling of intervals that contain unvoiced speech sounds. Informal tests with inputs of sinusoids, noise and speech material indicate that the processor functions according to its design. Formal tests are now under way with synthesized speech tokens and with digitized natural speech whose F_0 contours and voice/unvoice boundaries are fully known. A technical description of our portable F_0 extractor using the AMDF algorithm is presented in Appendix 1.

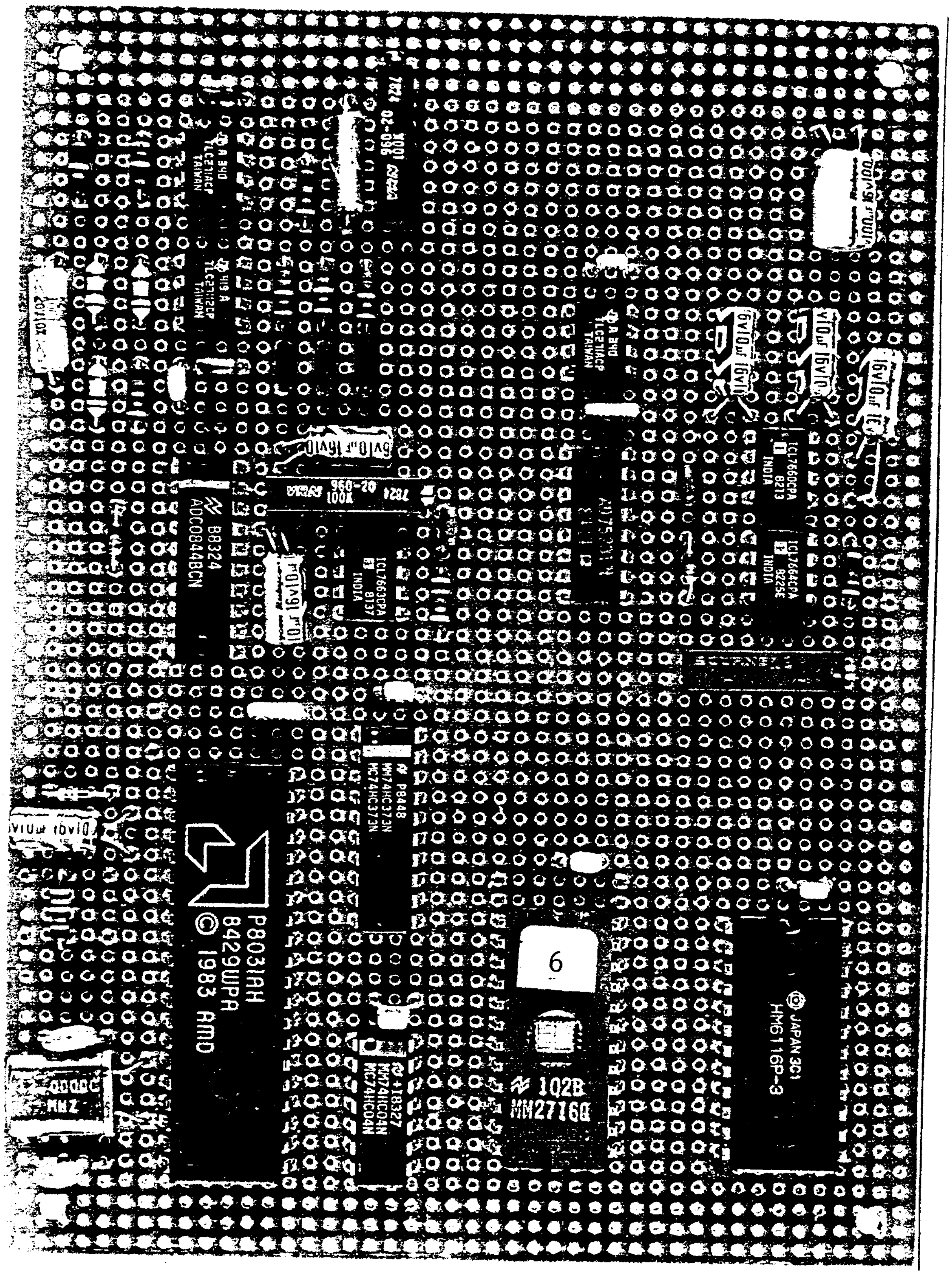


Fig. III.2. Photograph of the hardware prototype of a speech processor appropriate for use in a single-channel auditory prosthesis.

IV. Software for Support of the RTI Patient Stimulator

In designing the software handler that operates the RTI Patient Stimulator three clear priorities were observed:

1. to make available to a compiled FORTRAN program, running under Data General's Advanced Operating System (AOS), the full bandwidth capability of the Patient Stimulator, even for eight independent channels and a complex stimulus lasting several seconds;

2. to provide an array of convenient software features that simplify both automatic and special-purpose coding of various types of stimuli, while preserving access to the full flexibility of the Patient Stimulator hardware;

3. to minimize the size of stimulus code files.

The device handler we have developed for the Patient Stimulator is called TUBE. It is a self-contained program for execution by a digital control unit (DCU) operating under the control of a Data General Eclipse computer. TUBE reads code from buffers in the main Eclipse memory and, as often as every 50 microseconds, interprets a control word imbedded in that code and outputs an appropriate command string to the RTI Patient Stimulator. In keeping with the priorities listed above, TUBE is designed to output as quickly as possible the number of stimulator commands specified in the current control word. TUBE then will fulfill any "homework assignment" made by the same control word (such as resetting TUBE's buffer pointer, requesting buffer service from a task running on the Eclipse itself, reading the Patient Stimulator's ADC registers, or halting the DCU). Alternatively, the control word may instruct TUBE to spend a prescribed length of time in a "coasting" mode -- supplying the Patient Stimulator with appropriate continuation messages but not reading new control words.

TUBE is one of several device handlers that may be executed by the DCU. Others control analog-to-digital and digital-to-analog converters, for instance. Utility subroutines are available to load these handlers into the

DCU's dedicated memory and establish commmunications between them and the AOS tasks that will be using them. DCUSOLO is normally used to load TUBE, dedicating the full speed of the DCU to that single task.

When done in the context of Block-Diagram Compiler output, the whole process of converting a stimulus waveform to stimulator code and making it available to TUBE is automated and completely transparent to the user. In the discussion that follows, however, we will illustrate the process by referring to utilities that accomplish these tasks in stages, for program development and testing purposes.

BUILD is a FORTRAN program that defines a mnemonic language for creating special-purpose stimulator code files directly -- roughly the equivalent of an "assembly language" for the Patient Stimulator. This allows (and requires) the user to specify each control word and stimulator command of such a file.

Another program, CODER, generates stimulator code files that conform to a standard format also used by the block-diagram compiler. The user supplies CODER with a digitized waveform file for each channel to be driven. The program then verifies that the files are compatible and creates a single optimized stimulator code file. In addition, the user may specify an initial configuration for the Patient Stimulator or indicate that any of a number of optional stored configurations may be used with this stimulator code. Such configurations (any number of optional ones may be generated using a program called CONFIG) include specification of stimulator clock rate, relay settings, ground connections, and ADC control words; verification that the correct hardware configuration plug is installed; and optional specification of a custom sequence of initialization commands to the Patient Stimulator.

POUR is a FORTRAN main program (compiling to an AOS task) that (1) uses DCUSOLO to install TUBE in the DCU and establish communications between POUR and TUBE, (2) sets up a windowed memory system for double buffering, (3) loads the beginning of a user-specified standard-format stimulator code file (one produced by CODER, for instance) into the buffers, (4) determines whether this file contains its own configuration back and, if not, requests the name of a CONFIG-produced file from the user and loads it, (5) signals TUBE to begin execution, and (6) services the buffers and terminates as requested by TUBE.

V. Software for Support of Basic Psychophysical Studies and Speech Testing

Speech testing is achieved by presenting speech data files that have been prepared by the block-diagram compiler. These speech files are the synthesized outputs from a simlated speech processor. Inputs to the processor are speech tokens from the "MAC", "miniMAC", Klatt synthesizer, and/or confusion matrices. All speech data files are computed offline and are assembled into disk files for rapid access during testing. With speech data files available on disk at test time, full randomizations of test token presentations are possible. In preparing the speech data files the final step is to process the files using a specialized compiler called CODER (see section IV).

For most psychophysical studies the necessary stimulator code buffers must be generated by the very program coordinating the test sequence, rather than being read from a file prepared in advance. The design of TUBE (our Patient Stimulator control program, see section IV immediately above) makes such "real-time" synthesis easy to achieve. To illustrate this, we shall list some example psychophysical test stimuli, indicating how TUBE's features facilitate their rapid production.

Tests that utilize rectangular pulses in silence -- hearing threshold as a function of either pulse amplitude or duration, loudness comparisons varying pulse amplitude or duration, etc. -- are particularly easy to code. Varying the amplitude of a pulse of any length merely requires the altering of a single word in a buffer. (One buffer contains instructions for the current stimulus pulse(s) and is executed once by TUBE whenever the stimulus is to be presented. At other times TUBE repeatedly executes another buffer that generates only silence.) Altering pulse duration only involves moving a termination command from one location within the buffer to another. Using biphasic pulses merely doubles these modest computational loads.

To produce a continuous noise stimulus we prepare three buffers: one with a noise onset (cosine bell envelope, for instance), a second with continuous noise, and a third in which noise is smoothly terminated. Then a noise stimulus of arbitrary length (as required, say, for an extinction test) easily can be provided by having TUBE execute buffer 1 once, buffer 2 repeatedly for as long as necessary, and, finally, buffer 3. Noise pulses less than one buffer in duration can be calculated quickly by imposing an envelope on buffer 2. Similarly, rectangular pulses can be superimposed on continuous noise by adding a constant value to the appropriate subset of locations in a copy of buffer 2 and executing the copy once, preceeded and followed by repeated execution of buffer 2 itself. The amount of computing necessary between stimuli does increase significantly in the case, say, of frequency bursts superimposed on noise, but the buffer switching capabilities of TUBE still minimize the number of locations that must be recomputed.

VI. Plans for the Next Quarter

The most important activity for the next quarter will be the conduct of tests with an implant patient at UCSF. This patient was implanted on May 8, 1985, and will be available for testing with the percutaneous cable system until August 1, 1985. The UCSF and RTI teams will collaborate in the evaluation of various speech-processing strategies and "stimulus primitives" during this period. All elements of the RTI testing facility will be used in the tests, including the block-diagram compiler, patient stimulator, and software for support of psychophysical studies and speech testing. A detailed outline of studies the RTI team has proposed for the present patient is presented in Appendix 2.

Next, preparation for the first implant patient at Duke will continue. Several candidates have been identified for formal evaluation studies, and we expect that at least one of these selected candidates will qualify for an implant within one or two months. The equipment for the cochlear-implant laboratory at Duke has been completed or obtained; all that remains is installation and checkout. We therefore anticipate that we will be able to support fully the effort at Duke in parallel with meeting our major objectives and commitments at UCSF, as outlined above.

Finally, our plans for the next and upcoming quarters include presentations of various aspects of our work at major conferences on cochlear implants and related topics. These presentations are listed on the next page and include invited papers at the Gordon Research Conference on Implantable Auditory Prostheses, August 19-23, 1985; the International Cochlear Implant Symposium and Workshop in Melbourne, Australia, August 27-31, 1985; the special sessions on cochlear implants and on signal processing for the hearing impaired at the IEEE Bioengineering Conference, September 27-30, 1985; the special session on neurostimulation at the ACEMB, September 30- October 2, 1985; and the Conference on Speech Recognition with Cochlear Implants, April 17-19, 1986. In addition to preparation of platform presentations for these meetings, we will also be preparing several full-length papers for conference proceedings, as indicated below.

List of Upcoming Conference Presentations

Wilson, BS and CC Finley: Speech processors for auditory prostheses. To be presented at the International Cochlear Implant Symposium and Workshop, Melbourne, Australia, Aug. 27-31, 1985 (full-length paper to be published in the proceedings).

Finley, CC and BS Wilson: Field models of the Melbourne electrode array. Invited paper to be presented at the International Cochlear Implant Symposium and Workshop, Melbourne, Australia, Aug. 27-31, 1985 (full-length paper to be published in the proceedings).

Wilson, BS: Coding strategies for multichannel auditory prostheses. Invited paper to be presented at the Gordon Research Conference on Implantable Auditory Prostheses, Aug. 19-23, 1985.

Finley, CC: An integrated field-neuron model of intracochlear stimulation. Invited paper to be presented at the Gordon Research Conference on Implantable Auditory Prostheses, Aug. 19-23, 1985.

Wilson, BS: Discussion Leader, Gordon Research Conference on Implantable Auditory Prostheses, Aug. 19-23, 1985.

Wilson, BS: Comparison of strategies for coding speech with multichannel auditory prostheses. Invited paper to be presented at the Conference on Speech Recognition with Cochlear Implants, New York University, April 17-19, 1986.

Finley, CC and BS Wilson: Models of neural stimulation for electrically evoked hearing. Invited paper to be presented in the special session on neurostimulation, ACEMB Meeting, Sept. 30-Oct. 2, 1985.

Wilson, BS and CC Finley: Speech processors for auditory prostheses. Invited paper to be presented in the special session on signal processing for the hearing impaired, IEEE Bioengineering Conf., Sept. 27-30, 1985 (full-length paper to be published in the proceedings).

Finley, CC and BS Wilson: A simple finite-difference model of field patterns produced by bipolar electrodes of the UCSF array. Invited paper to be presented at the special session on cochlear implants, IEEE Bioengineering Conf., Sept. 27-30, 1985 (full-length paper to be published in the proceedings).

VII. Acknowlegment

We are pleased to acknowledge the many valuable contributions Dr. Mark White of UCSF made to the conduct and design of the experiments described in section II of this report.

VIII. References

Paliwal, K. K., Comparative performance evaluation of different pitch estimation algorithms for noisy speech, Acoustics Lett., 6 (1983) 164-166.

Ross, M. J., Shaffer, H. L., Cohen, A., Frendberg, R. and Manley, H. J., Average magnitude difference function pitch extractor, IEEE Trans. on Acoust., Speech & Signal Processing, ASSP-22 (1974) 353-362.

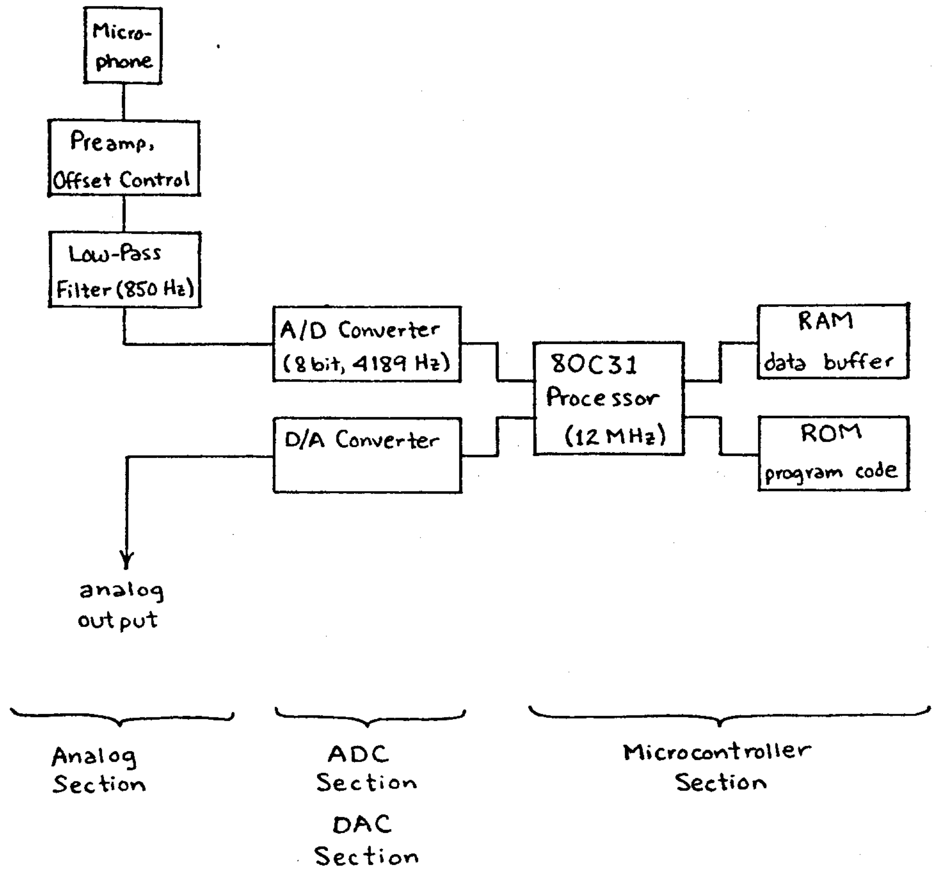
Sung, W. Y. and Un, C. K., A high-speed pitch extractor based on peak detection and AMDF, J. Korea Inst. Electr. Eng., 17 (1980) 38-44.

Un, C. K. and Yang, S.-C., A pitch extraction algorithm based on LPC inverse filtering and AMDF, IEEE Trans. on Acoust., Speech & Signal Processing, ASSP-25 (1977) 565-572.

Appendix 1

Circuit Diagrams and Software for a Portable F_0 Extractor Using the AMDF Algorithm

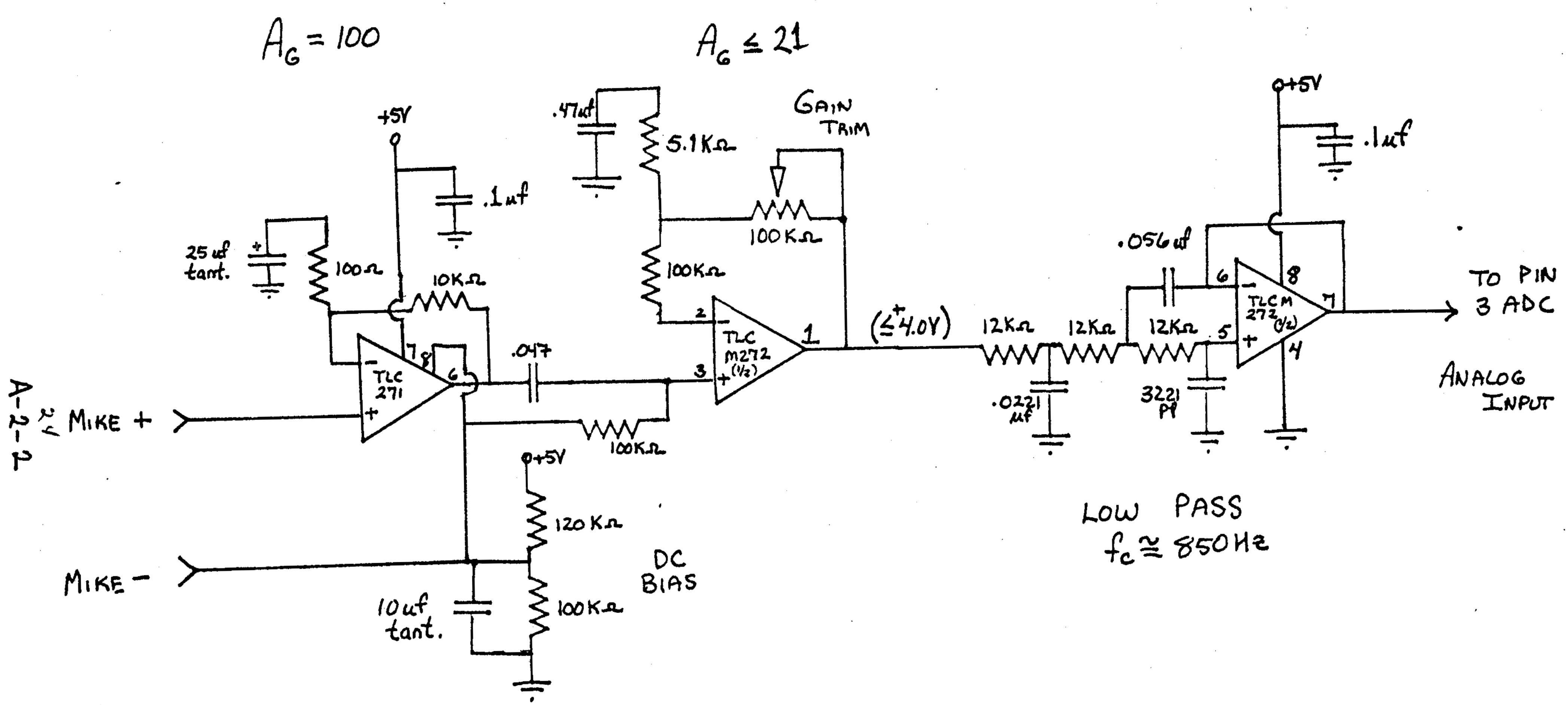
CONTENTS



Block Diagram of the Hardware

Keating

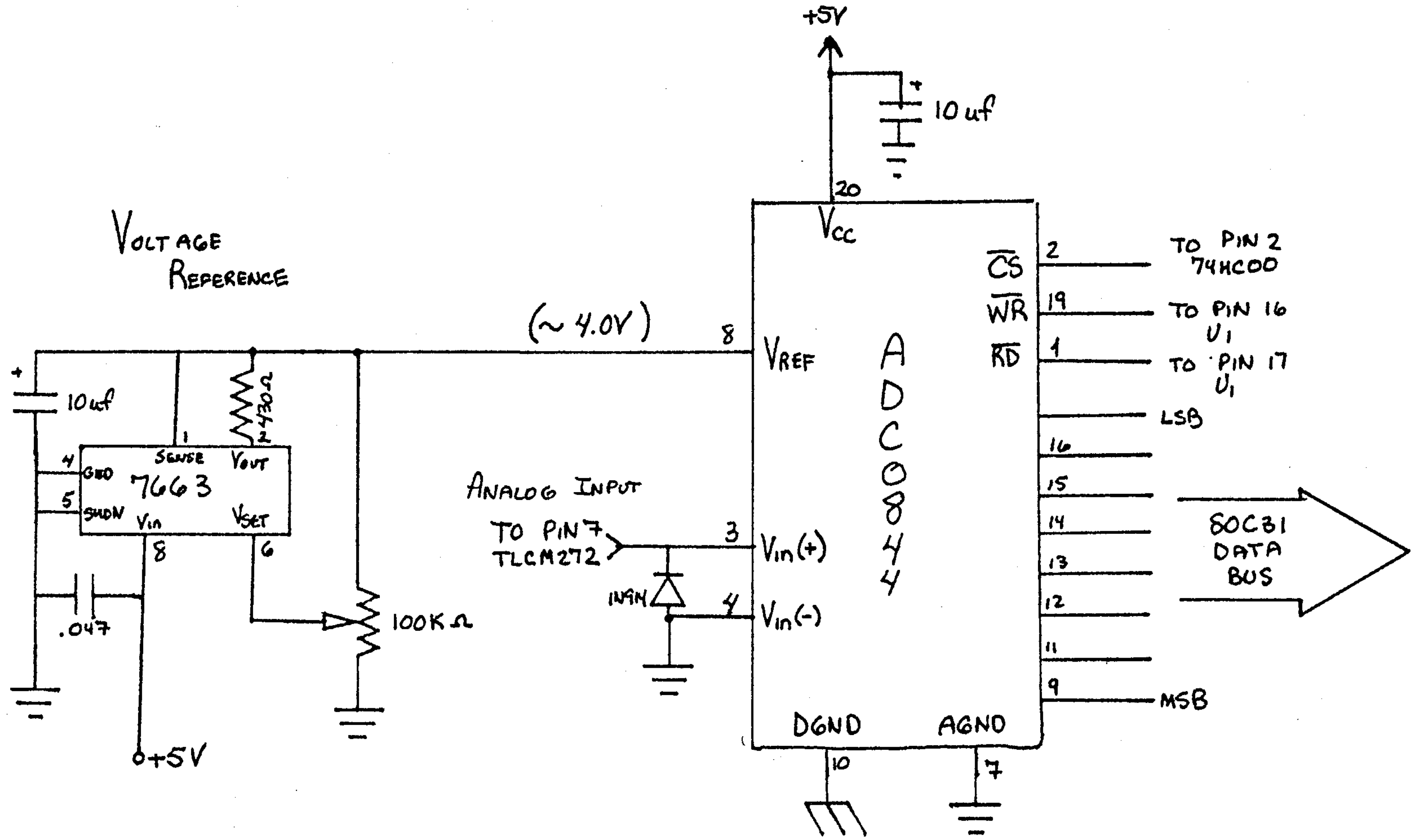
4/7/85



AUDIO SECTION

KEAT 6

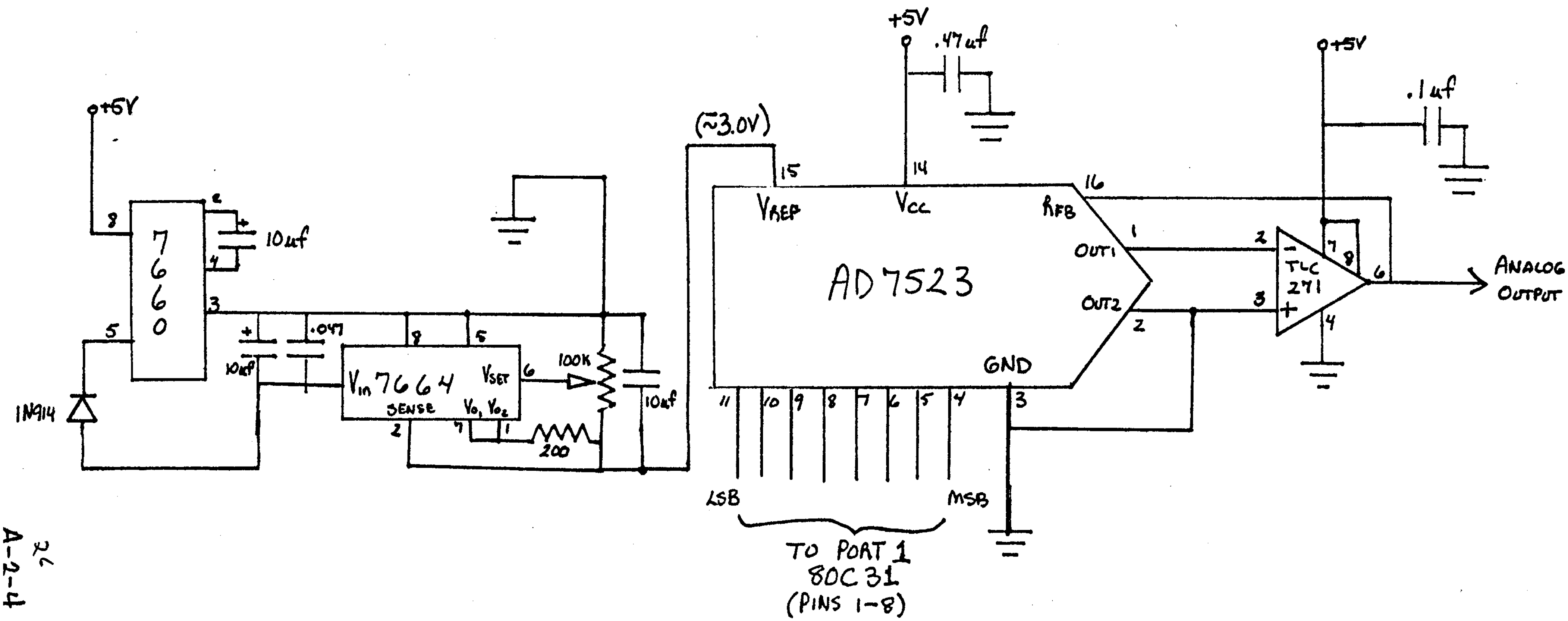
4/7/85



ANALOG/DIGITAL CONVERTER

KEATING

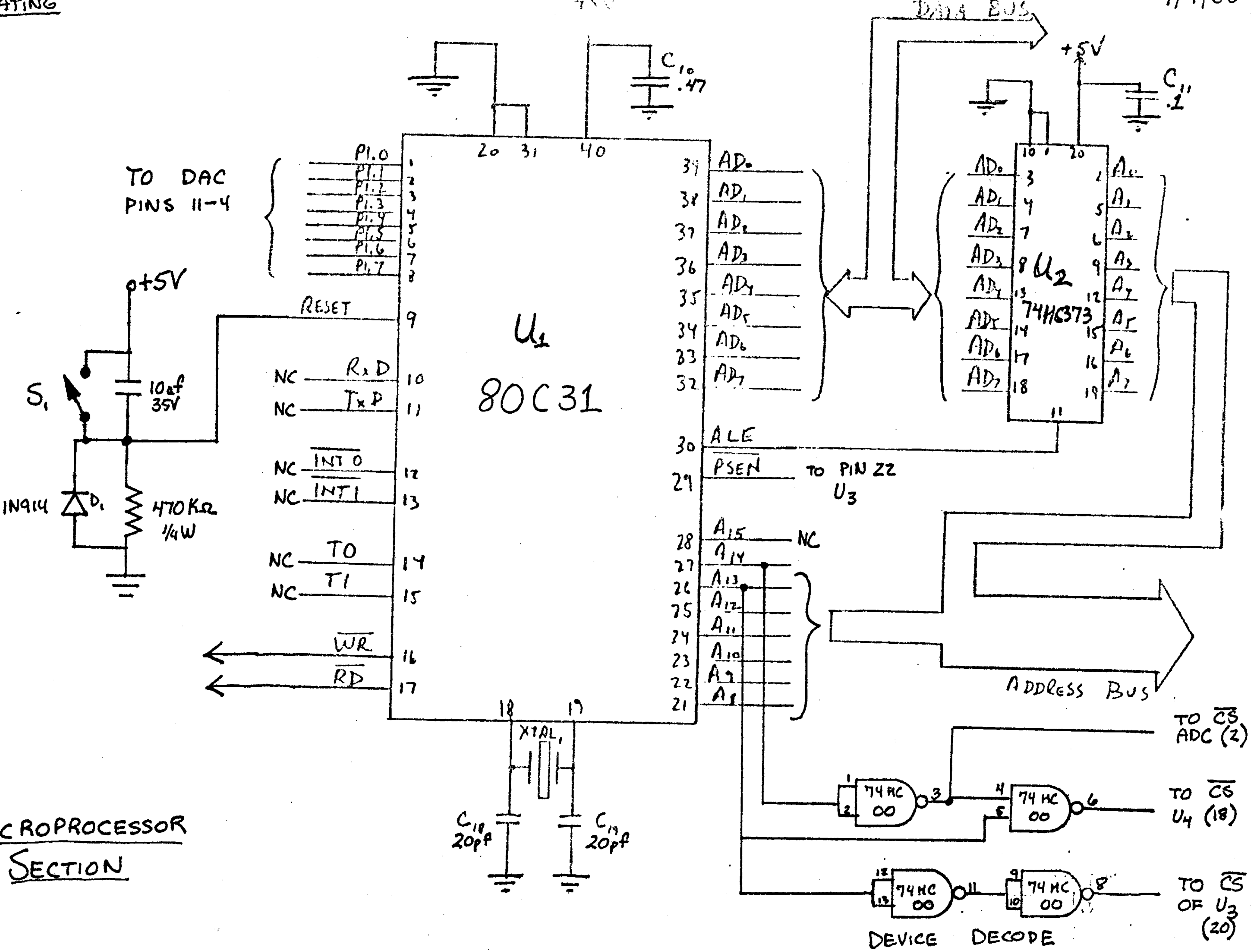
4/7/85



DIGITAL/ANALOG CONVERTER

KEATING

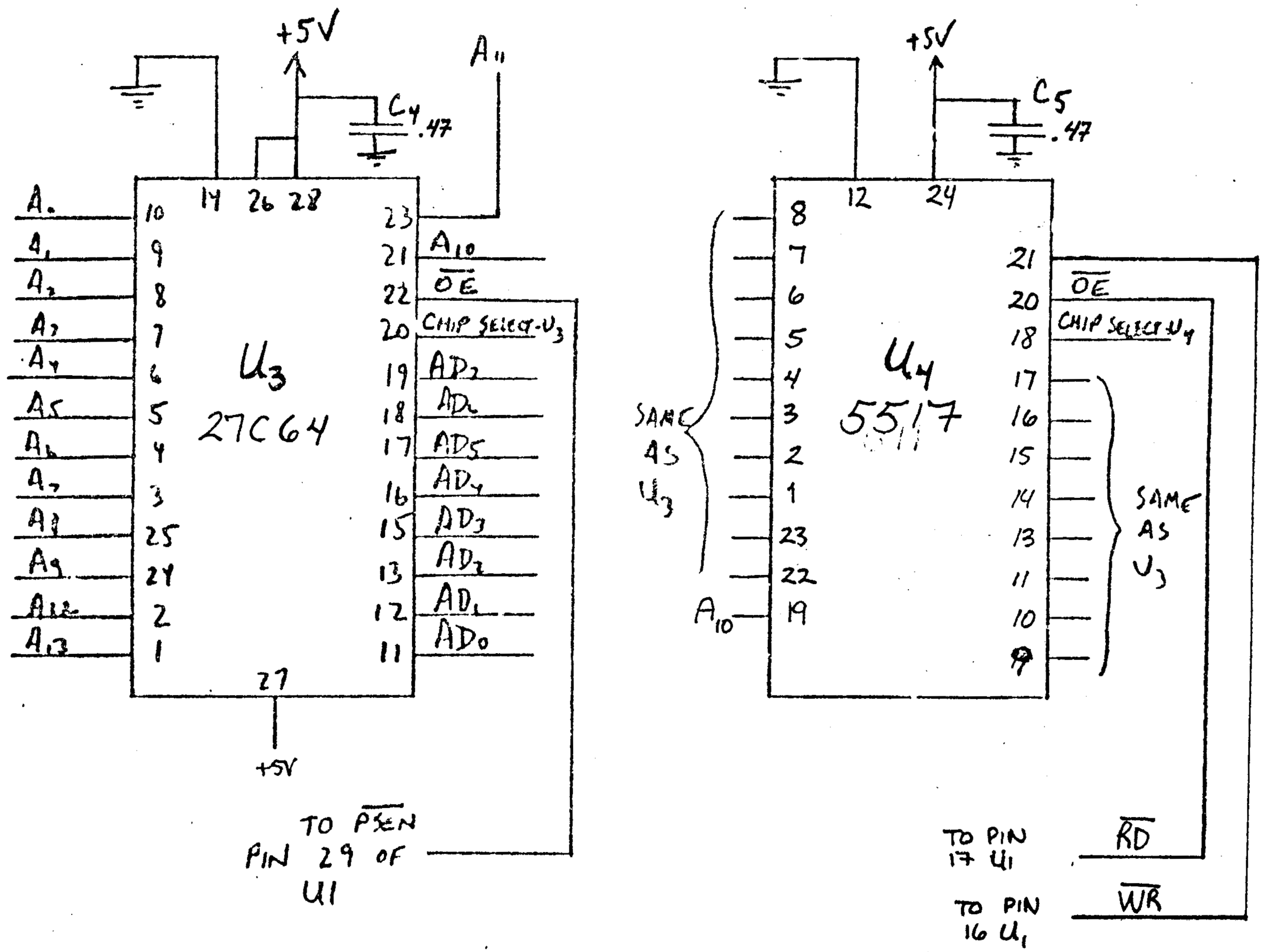
4/7/85



MICROPROCESSOR SECTION

KEATING

4/7/85



MEMORY SUBSECTION

```
;*************************************************************
;       EQUATE TABLES
;*************************************************************
;
USER    EQU     4030H
;
;
; *FOR PSW*
BANK1   EQU     08H                     ;REGISTER BANK 1
BANK2   EQU     10H                     ;REGISTER BANK 2
BANK3   EQU     18H                     ;REGISTER BANK 3
;
; *SETUP VARIABLES*
TSET    EQU     22H                     ;TIMERO =MODE2
TIME    EQU     -220                    ;SAMPLING INTERVAL
PAGE    EQU     47H                     ;PAGE ADDRESS OF RAM
STACK   EQU     100                     ;STACK ADDRESS
ADC     EQU     8000H                   ;ADDRESS OF A/D
POINTS  EQU     126                     ;SAMPLES PER FRAME
;
; *GENERAL ASSIGNMENTS*
RN6     EQU     0EH                     ;SAME AS R6 OF BANK 1
RN7     EQU     0FH                     ;SAME AS R7 OF BANK 1
RR0     EQU     10H                     ;SAME AS R0 OF BANK 2
RR2     EQU     12H                     ;SAME AS R2 OF BANK 2
RR5     EQU     15H                     ;SAME AS R5 OF BANK 2
RR6     EQU     16H                     ;SAME AS R6 OF BANK 2
RR7     EQU     17H                     ;SAME AS R7 OF BANK 2
;
; *USED BY AMDF ROUTINE*
AMDFL   EQU     13H                     ;SAME AS R3 OF BANK 2
AMDFH   EQU     14H                     ;SAME AS R4 OF BANK 2
STEP    EQU     6                       ;STEP FACTOR
NPTS    EQU     55                      ;NUMBER OF AMDF VALUES
STOR    EQU     55                      ;START AMDF STORAGE
STRT    EQU     18H                     ;SAME AS R0 OF BANK 3
ADMIN   EQU     19H                     ;SAME AS R1 OF BANK 3
MIN     EQU     1AH                     ;SAME AS R2 OF BANK 3
MAX     EQU     1BH                     ;SAME AS R3 OF BANK 3
LPRIME  EQU     10                      ;LENGTH OF WINDOW
SFT1    EQU     10
SFT2    EQU     (LOW BUF2) + SFT1
;
; *USED BY POSTPROCESSOR
PITCH   EQU     21H             ;PITCH LOGIC
THRES0  EQU     20              ;FOR MINIMUM SELECTION
THRES1  EQU     60              ;THE AMPLITUDE THRESHOLD
THRES2  EQU     2AH             ;THE PITCH THRESHOLD
;
PTCH0   EQU     22H             ;LAST PITCH ESTIMATE
PTCH1   EQU     23H             ;CURRENT PITCH ESTIMATE
PTCH2   EQU     24H             ;FUTURE PITCH ESTIMATE
;
; *BIT ADDRESSABLE BYTES*
FLAG    EQU     20H                     ;GENERAL FLAG REG.
; ***********************************************************
; ADSERV.ASM
; THE SERVICE ROUTINE FOR THE A/D CONVERTER.
; USES REGISTER BANK 1 AND ALSO SAVES IMPORTANT PARAMETERS
; FROM THE INTERRUPTED ROUTINE ON THE STACK.
; USES R0,R1,R6,R7 OF BANK 1
; ***********************************************************
        ORG     USER
        SJMP MAIN
;
        ORG TIMER0+USER
```

```
;
ADSERV: PUSH ACC                ;SAVE THESE ON THE STACK
        PUSH PSW
        MOV TH0,#TIME
        MOV PSW,#BANK1          ;SWITCH TO REGISTER BANK 1
        MOVX A,@DPTR            ;GET THE CURRENT VALUE
        MOVX @R0,A              ;NOW SAVE THE POINT
        INC R0                  ;AND INCREMENT THE POINTER
        DJNZ R1,NOTYET          ;FILL FRAME UNTIL DONE
        MOVX A,@R0
        MOV R0,A
        MOV R1,#POINTS          ;RESET INDEX
        CLR A
        MOVX @DPTR,A            ;RESTART THE A/D
        POP PSW
        CPL F0                  ;CMP FRAME FLAG
        POP ACC
        RETI
NOTYET: CLR A                   ;RESTART THE A/D
        MOVX @DPTR,A
        POP PSW                 ;RESTORE THE REGISTERS
        POP ACC
        RETI
; ***************************************************************
; THIS IS THE CODE FOR THE RESET INITIALIZATIONS
;  *IT SETS TIMER0 AS AN EIGHT BIT AUTORELOAD TIMER FOR THE A/D
; USES R0,R6,R7 OF BANK 1
;****************************************************************
; THE FOLLOWING ARE THE DEFINITIONS FOR THE DATA ARRAYS TO SET
; UP DOUBLE BUFFERING.
;
;
MAIN:   MOV PSW,#BANK1          ;SELECT BANK 1
        MOV SP,#STACK           ;FIRST STACK ADDRESS = 121 D !
        MOV P2,#PAGE            ;LATCH PORT 2 TO ENABLE PAGING
; THE FOLLOWING SETS UP THE DATA ARRAY FOR DOUBLE BUFFERING
        MOV R0,#(LOW NDX1)      ;ADDRESS OF THE POINTER
        MOV A,#(LOW BUF2)       ;ITS CONTENTS
        MOVX @R0,A              ;NOW ITS SET
        MOV R0,#(LOW NDX2)      ;THE OTHER POINTER
        MOV A,#(LOW BUF1)
        MOVX @R0,A
;
;
        MOV DPTR,#ADC           ;ADDRESS OF A/D
        CLR A                   ;INITIALIZE SOME REGISTERS
        ANL RN6,A
        ANL RN7,A
        ANL PITCH,A
        MOV R1,#POINTS
        MOV R0,#(LOW BUF1)
        MOV TMOD,#TSET          ;SET UP TIMER/COUNTERS
        MOV TH0,#TIME           ;THE SAMPLING INTERVAL
        SETB PT0                ;SET IMPORTANT BITS
        SETB ET0
        SETB EA
        SETB TR0
        MOVX @DPTR,A            ;START THE A/D
;
; SETTING OF THE FLAG BIT DETERMINES WHICH DATA BUFFER IS USED
; TO COMPUTE THE AMDF. IF SET USE SECOND BUFFER.
        MOV PSW,#BANK2          ;SELECT BANK 2
        SETB F0
WAIT:   JB F0,WAIT              ;WAIT FOR NEXT FRAME
;
;****************************************************************
```

```
; AMDF.ASM
; THIS IS THE ACTUAL ROUTINE THAT COMPUTES THE AMDF OF THE
; CURRENT FRAME AND STORES IT IN THE 8051 ON-CHIP MEMORY
;
; USES REGISTER BANK 2 AND DIRECTLY ADDRESSES R0,R1 OF BANK 3
;*****************************************************************
;
;
START:   JB F0,USETWO                   ;FINDS CURRENT FRAME
         MOV STRT,#(LOW BUF1)           ;USE BUFFER 1
         MOV RR5,#SFT1
         SJMP SKIP
USETWO:  MOV STRT,#(LOW BUF2)           ;ELSE USE BUFFER 2
         MOV RR5,#SFT2
SKIP:    MOV R7,#STOR                   ;STORAGE FOR AMDF
         MOV R6,#NPTS                   ;NO. OF AMDF POINTS
         CLR A
         ANL MAX,A                      ;CLR MAX
         MOV MIN,#0FFH                  ;RESET MIN
         SETB FLAG.6
AMDF:    MOV R2,#LPRIME                 ;THE INTEGRATING WINDOW
         CLR A                          ;INITIALIZE THESE
         ANL AMDFL,A                    ;SAME AS R3
         ANL AMDFH,A                    ;SAME AS R4
         MOV R0,STRT                    ;DATA POINTERS
         MOV R1,RR5                     ;
LOOP:    MOVX A,@R1                     ;THIS IS X(N+K)
         MOV B,A                        ;SAVE IT TEMPORARILY
         MOVX A,@R0                     ;THIS IS X(N)
         CLR C                          ;MAKE SURE THE CY IS OK
         SUBB A,B                       ;SUBTRACT THE TWO
         JNC NOOV                       ;ABSOLUTE VALUES ONLY!
         CPL A
         INC A
NOOV:    ADD A,R3                       ;ADD THE DIFFERENCE IN
         MOV R3,A                       ;SAVE IT
         JNC OVOK                       ;INC MSB IF APPROPRIATE
         INC R4
OVOK:    MOV A,R0                       ;BUMP POINTERS
         ADD A,#STEP
         MOV R0,A
         MOV A,R1
         ADD A,#STEP
         MOV R1,A
         DJNZ R2,LOOP                   ;KEEP GOING
;*****************************************************************
; AMDF VALUES ARE DIVIDED BY TWO AND STORED HERE
;  ALSO A RUNNING TAB OF THE MINIMUM AND MAX. IS MAINTAINED
;
         MOV B,AMDFH                    ;DIVIDE AMDF BY 2
         MOV A,R3
         MOV C,B.0
         RRC A
         MOV R0,RR7                     ;STORE RESULT IN OC RAM
         MOV @R0,A
         MOV P1,@R0                     ;OUTPUT TO DAC
;*****************************************************************
; THE MINIMUM SELECTION LOGIC FOLLOWS
;*****************************************************************
;
CHECK:   JNB FLAG.6,OKC                 ;IS THIS AMDF(1)
         CLR FLAG.6                     ;IF SO SKIP IT
         CLR C
         JMP NOPE
OKC:     DEC R0                         ;CHECK FOR MINIMUM
         CLR C
```

```
        SUBB A,@R0                          ;AMDF(N)-AMDF(N-1)
        JC NOPE                             ;CY => DECREASING
        JNB FLAG.0,NOPE                     ; => INCREASING
        MOV A,@R0                           ;AMDF(N-1) IS A MIN.
        CLR C
        SUBB A,MIN                          ;NEW MIN. < OLD MIN. ?
        JNC NOPE
        CPL A                               ;YUP !
        INC A                               ;BY HOW MUCH?
        CLR C
        SUBB A,#THRES0                      ;GREATER THAN THIS?
        JC MLTPL                            ;MUST BE MULPTIPLE
        MOV MIN,@R0                         ;ITS NEW MIN.
        MOV ADMIN,R0
MLTPL:  CLR C
NOPE:   MOV FLAG.0,C
        INC R0
;***************************************************************
; NOW SELECT A MAXIMUM AMDF VALUE
;***************************************************************
        MOV A,@R0                           ;CHECK FOR A MAX.
        CLR C
        SUBB A,MAX
        JC NOPE2
        MOV MAX,@R0
NOPE2:  INC RR7
        INC RR5                             ;INCREMENT SHIFT FACTOR
        DJNZ R6,AMDF
;
;       MOV P1,#0FFH
;**************************************************************
; ALL DONE FOR THIS FRAME, ON TO THE PITCH DETERMINATION
;       IS THE SELECTED MINIMUM CLEAR????
;
        SETB RS0                            ;SELECT BANK 3
DIFF:   MOV A,R3                            ;MAX !
        CLR C
        SUBB A,R2                           ;SUBTRACT MIN.
; THE VOICED/UNVOICED QUESTION IS ANSWERED HERE
;
        CLR C                   ;IS MIN. AMP >= THRES1 ?
        SUBB A,#THRES1
        JNC VOICED              ;IF SO => VOICED FRAME
        CLR PITCH.0             ;OTHERWISE UNVOICED
        CLR A
        ANL PTCH2,A
        SJMP TABLE
VOICED: SETB PITCH.0
        MOV A,ADMIN
        CLR C
        SUBB A,#45
        MOV PTCH2,A
;
;*******************************************************
; THE AMDF POSTPROCESSOR
;  AN ADAPTATION OF THE PITCH SMOOTHING ALGORITMS PRESENTED
;  BY BOTH SUNG AND UN (1980) AND ROSS ET. AL.(1974)
;
;*******************************************************
TABLE:  MOV A,#07H                  ;JMP TABLE USING PITCH
        ANL A,PITCH
        CJNE A,#00,NEXT1            ;UNVOICED
        CLR A
        JMP OUTPUT
NEXT1:  CJNE A,#01,NEXT2            ;UNVOICED
        CLR A
```

```
        JMP OUTPUT
NEXT2:  CJNE A,#02,NEXT3        ;UNVOICED
        CLR PITCH.1             ;ADJUST BIT
        CLR A
        ANL PTCH1,A
        JMP OUTPUT
NEXT3:  CJNE A,#03,NEXT4        ;POUT=1/3P(N) + 2/3P(N+1)
        MOV A,PTCH1
        MOV B,#3
        DIV AB
        MOV R4,A
        MOV A,PTCH2
        MOV B,#3
        DIV AB
        CLR C
        RLC A
        ADD A,R4
        SJMP OUTPUT
NEXT4:  CJNE A,#04,NEXT5        ;UNVOICED
        CLR A
        JMP OUTPUT
NEXT5:  CJNE A,#05,NEXT6        ;POUT=1/2P(N-1)+1/2P(N+1)
        SETB PITCH.1            ;ADJUST BIT
        MOV A,PTCH0
        ADD A,PTCH2
        RRC A
        MOV PTCH1,A
        SJMP OUTPUT
NEXT6:  CJNE A,#06,NEXT7        ;POUT=2/3P(N-1)+1/3P(N)
        MOV A,PTCH1
        MOV B,#3
        DIV AB
        MOV R4,A
        MOV A,PTCH0
        MOV B,#3
        DIV AB
        CLR C
        RLC A
        ADD A,R4
        SJMP OUTPUT
NEXT7:  MOV A,PTCH0             ;POUT= AVER. OF THE LEAST
        CLR C                   ;DIFFERENCE BETWEEN ANY
        SUBB A,PTCH1            ;TWO
        JNC NC1
        CPL A
        INC A
NC1:    MOV R3,A                ;R3=|P(N)-P(N-1)|
        MOV A,PTCH2
        CLR C
        SUBB A,PTCH1
        JNC NC2
        CPL A
        INC A
NC2:    MOV R4,A                ;R4=|P(N)-P(N+1)|
        CLR C
        SUBB A,R3
        JNC NC3
        MOV A,R4                ;R4 IS SMALLER
        MOV R3,A
        MOV R0,#PTCH2           ;SO POINT TO PTCH2
        SJMP COMP
NC3:    MOV R0,#PTCH0           ;R3 SMALLER,POINT TO PTCH0
COMP:   MOV A,PTCH0
        CLR C
        SUBB A,PTCH2
        JNC NC4
```

```
        CPL A
        INC A
NC4:    CLR C
        SUBB A,R3
        JNC NC5
        MOV A,PTCH0                 ;P(N-1)-P(N+1) IS SMALLEST
        ADD A,PTCH2
        RRC A
        SJMP OUTPUT
NC5:    MOV A,@R0                   ;P(N)-@R0 IS SMALLEST
        ADD A,PTCH1
        RRC A
;
OUTPUT: MOV B,#4
        MUL AB
        MOV P1,A
;
CLEAN:  MOV PTCH0,PTCH1                     ;SHIFT PITCH VALUES
        MOV PTCH1,PTCH2
        CLR A
        ANL PTCH2,A
;
        MOV A,PITCH                         ;SHIFT PITCH LOGIC
        CLR C
        RLC A
        MOV PITCH,A
;
        CLR  RS0                            ;SELECT BANK 2
;
; WAIT HERE TILL NEXT FRAME IS READY
;
SIT:    JNB F0,LP2
LP1:    JB  F0,LP1
        JMP START
LP2:    JNB F0,LP2
        JMP START
;***************************************************************
; THIS IS THE DATA ARRAY THAT RESIDES IN THE HIGHEST 256 BYTES
; OF THE 2K RAM.  THIS FACILITATES PAGING BY LATCHING 47H INTO
; THE PORT 2 OUTPUT BUFFER
        ORG 4700H
BUF1:   DS POINTS
NDX1:   DS 1
BUF2:   DS POINTS
NDX2:   DS 1
 END

A>IC
```

<u>Appendix 2</u>

Outline of Proposed Tests with the
Present Implant Patient at UCSF,
for the Months of June and July, 1985

Major Goals of San Francisco Trips, June, 1985

I. Conduct basic psychophysical studies, especially those related to "stimulus primitives"

a. thresholds to "monophasic" and "biphasic" pulses on all channels, for various durations/phase between 0.1 and 8.0 ms;

b. loudness matches for suprathreshold pulses, various durations, for selected channels;

c. measures of "pitch" and "sharpness comparisons for various waveforms and durations, selected channels and across channels;

d. thresholds to filtered noise bursts, break frequencies at 2.0 and 6.0 kHz;

e. thresholds to wideband noise;

f. measures of the time for "decay" or "extinction" of initital percepts when filtered and unfiltered noise are presented continuously;

g. measures of loudness DLs for pulses delivered alone and for pulses superposed on a continuous background of "extinguished" bandpass noise or of "extinguished" wideband noise;

h. measures of frequency DLs for the conditions of I.g above;

i. measures of "pitch" and "sharpness" comparisons for pulses delivered alone and for pulses superposed on a continous background of "extinguished" bandpass noise or of "extinguished" wideband noise, within and between channels.

II. Evaluate multichannel coding strategies

a. present, 4-channel UCSF processor;

b. 8-channel UCSF processor;

c. 4-channel processor that presents a "multipulse" excitation signal (from the linear-prediction residual) to the configured electrode channels according to the frequency and bandwidth of F2;

d. 8-channel version of the above.

III. Develop techniques to measure and interpret intracochlear evoked potentials

a. measure field patterns in the implanted ear using subthreshold stimuli, to evaluate various assumptions (e.g., homogeneous tissue properties) of the field-mapping model;

b. measure strength-duration curves on all channels, to determine the differences between these curves and the curves obtained in the psychophysical tests of I.a above;

c. obtain maps of dendrite survival in the implanted ear from the measurements of III.b above;

d. perhaps conduct channel-interaction studies with intracochlear EPs, to confirm predictions of the MELECSPRO model and to corroborate psychophysical measures of channel interactions.

IV. Evaluate single-channel coding strategies

a. learn if we can utilize the video tapes that have been made at UCSF, to present processed speech tokens in synchrony with information presented on the lips;

b. if not, test the strategy that codes F2/F1 and glottal excitation;

c. if so, also test the strategy that presents a "multipulse" excitation signal (from the linear-prediction residual, as in II.c above).