Design of a speech synthesis system

Instituut voor Perceptie Onderzoek
Postbus 513, 5600 MB Eindhoven

Rapport no. 565

Design of a speech synthesis system

Shi Hong

DESIGN OF A SPEECH SYNTHESIS SYSTEM

by SHI HONG

Department : Institute for Perception Research

Coach : Ir. R.Deliege

Chief : Ir. L.F.Willems

PII Supervisor : Dr. N.C.de Troye

Period : Jun. - Dec.1986

ACKNOWLEDGRMENTS

I would like to thank my coach, Ir. R. Deliege and his colleague Ir. R. P. Waterham, for their valuable suggestions and kind help on my work.

I would also like to thank Ir. L. F. Willems, chief of this project and Dr. N.C. de Troye, my PII supervisor for their support and encouragement.

SUMMARY

This report describes the design of a speech synthesis system.

The system is built around a 68000 micropressor which has the power and addressing capability needed for the large text-to-speech programs. Speech synthesis is done on basis of diphone concatenation and using a formant synthesizer (PCF 8200).

Chapter 1, and Chapter 2 are an introduction about the speech synthesis technique. Chapter 3 describes the designing of the system. The testing software is given in Chapter 4. Finaly, a program example is given in Chapter 5.

CONTENTS

Chapter 1

INTRODUCTION

1.1 General

Since speech is the simplest and most important way for human communication and since there is an increasing need for information exchange between man and machine. It is natural to ask whether speech might also be a suitable medium for man-machine communication. The rapid development of microelectronics and the development of speech synthesis technique made this a real possibility.

The bit rate and the memory required for a digital speech signal are dramatically reduced by using a speech synthesis technique. Besides of this, diphone speech synthesis is a flexible system which allows a user to compile his own utterances.

The purpose of a "speaking machine" is versatile, such as spoken instructions on how to use equipment, complex telephone systems that give a spoken output. It might also serve as an aid to handcapped who have lost their ability to speak, but are still able to operate a typewriter.

The Institute for Perception Research(IPO) has done a lot of work in this field where they have developed their own formant analysis-synthesis system. The synthesis part of this procedure is roughly explained in the 2nd Chapter of this report. A portable keyboard-to-speech system, using a 8031 microprocessor and hardware synthesizer MEA 8000, was developed in Philips application lab.

1.2 Objectives

In order to realize real text-to-speech conversion, we have to use a more powerful microprocessor. Because of this,the MC68000 is chosen which has the power and addressing capability needed for the large text-to-speech programs. Speech synthesis is done on basis of diphone concatenation and using a new generation formant synthesizer, the Philips PCF 8200.

The main application of the board will be in a speech communication aid, so the power consumption should be as low as possible. This means that only CMOS components (inclouding 68000) should be used.

CHAPTER 2

FUNDAMENTALS OF SPEECH SYNTHESIS TECHNIQUES

2.1 Model of the speech waveform

The basic technique for speech synthesis can be viewed in terms of a model of the speech waveform as the response of a slowly time varying system to either a periodic or a noiselike excitation.

The speech-production mechanism consists essentially of an acoustic tube. The vocal tract, excited by an appropriate source to generate the desired sound. In the case of voiced speech sounds, the excitation corresponds to a quasi-periodic pulse train representing the air flow through the vocal cords as they vibrate. (sounds like vowels A,E,I, etc.). Unvoiced sounds are generated by forcing air through a constriction in the vocal tract, thereby creating turbulence, which produces a source of noise to excite the vocal tract. (sounds like S, SH, F, etc.)

Fig 2-1. is a cross-sectional view of the vocal mechanism.

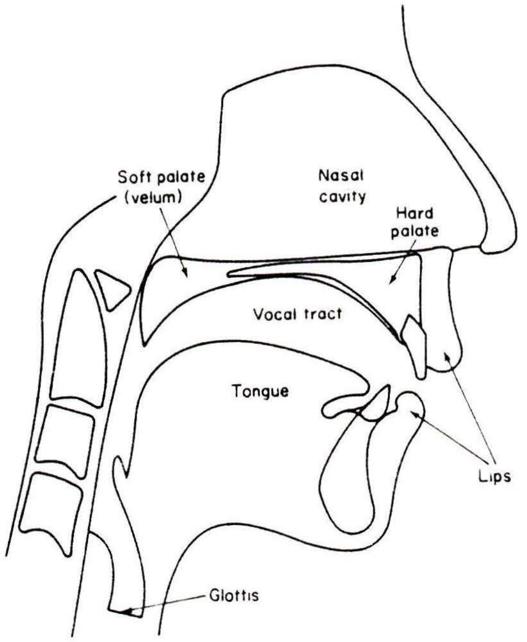


FIGURE 2-1. Cross-sectional view of the vocal mechanism

As sound, generated as discussed before, propagates down these tubes, the frequency spectrum is shaped by the frequency selectivity of the tube. This effect is very similar to the resonance effects observed with organ pipes or wind instruments. In the context of speech productions, the resonance frequencies depend upon the shape and dimensions of the vocal tract; each shape is characterized by a set of formant frequencies. Different sounds are formed by varying the shape of the vocal tract.

Thus the spectral properties of the speech signal vary with time as the vocal tract shape varies.

2.2 Formant synthesis principle

The general structure of a formant synthesizer is shown in Fig. 2-2. The basis for this class of synthesizer is the observation that, since the the vocal tract is an acoustic cavity, it is characterized by a set of modes or resonant frequencies. Thus its transfer function can be approximated by a cascade combination of resonant circuits. Each one representing one of the modes or resonances of the vocal tract. As the shape of the vocal tract changes, the resonances change. Consequently, the resonant circuits are provided with a set of time-varying parameters that control the center frequencies and bandwiths of the resonators. If the synthesizer is driven by an impulse train for voiced speech and white noise for unvoiced speech, a source-shaping filter is generally not time varying. In addtion, a filter that accounts for the effect of the coupling of the acoustic tube into space (i.e., a tube with infinite cross-sectional area) is required. This is also a fixed filter with a characteristic approximately corresponding to that of a differentiator.

In the overall effect of the vocal tract, glottal volume flow and lips radiation can be modelled as an all-pole discrete time-varying linear filter whose parameters are kept constant for the time the characteristics of the overall system may be considered constant. The effect of the nasal cavity is neglected which means no zero in the transfer function and that is quite acceptable. This all-pole model is in agreement with the well known fact that the value of the speech waveform at a given movement is closely correlated with its values at the previous movement. These have led to the sample speech productions model.

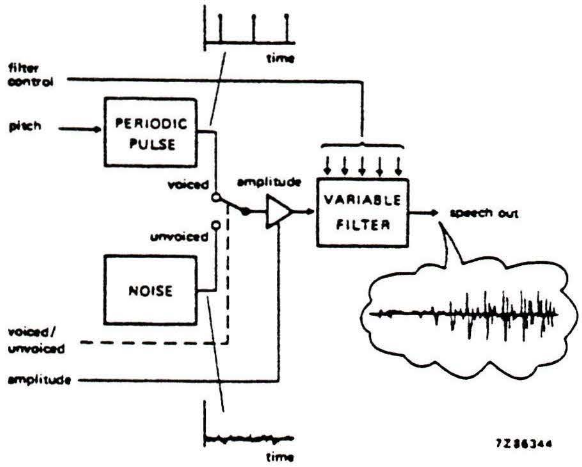


FIG. 2-2 Simple electronic model of the human speech production mechanism.

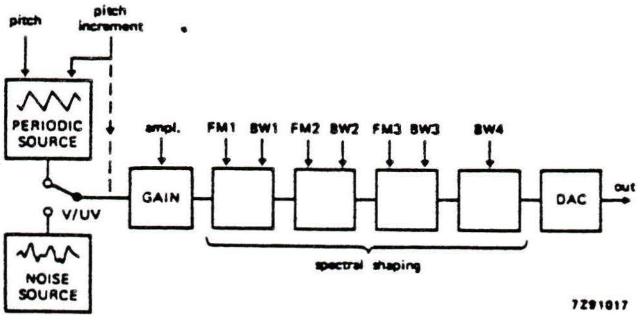


FIG 2-3 Formant synthesizer.

Fig. 2-3 shows a complete formant synthesizer. Either a periodic signal representing the pitch of the original speech, or an aperiodic signal, is fed to a variable filter comprising four resonators via an amplifier which controls the amplitude of the synthesized sound. The resonators model the sound in accordance with the formants in the original speech. Each resonator is controlled by two parameters, one for the resonant frequency and one for the bandwidth. The information required to control the synthesizer is:

-Pitch
-Amplitude
-Voiced/Unvoiced source selector
-Filter settings

In the synthesizer, each formant is simulated by a second order digital filter. To simulate four formants, four such filters are cascaded.

A good replica of the original speech is obtained by periodic update of this control information.

2.3 Diphone technique

There are several ways of coding speech. If we consider a fixed system with resynthesis speech, resident in external memory, we can have a fairly good quality. But it has a major drawback: the restricted number of utterances that can be made audible; only those which have been previously stored in memory.

Instead of storing complete utterances we might store smaller parts of speech to be used as building blocks in speech synthesis. By concatenating these units, the user can compile new sentences that fit his particular wishes.

Several units have been proposed to serve as building blocks. Such as words and isolated speech sounds. However, it has been demonstrated that neither method yields satisfactory results. The transitions between consecutive sounds found in natural speech appear to be very important for its quality. When this transitions are incorrect, speech becomes unnatural and difficult to understand. As this transitions are not easily computed by rule, other units have been selected. These units are called diphones. We can look at the speech sound as formed by units of sound called phonemes; then the diphones consist of the second half of a phoneme followed by the first half of the consecutive one. Once the user has these units at his disposal, they can be concatenated. Afterwards an intonation contour, which is needed for speech melody is superimposed on the utterance. Experiments show that during an utterance the pitch tends to fall gradually from a relatively high initial value to a lower one at the end. This is referred to as the declination phenomenon. Further, accented syllables often coincide with abrupt pitch movements.

CHAPTER 3

HARDWARE DESIGN

3.1 General

The system should be able to fulfill the following requirements:

- It should have the power and addressing capability needed for the large text-to-speech programs which are directly translated into 68000 assembly language from PASCAL programs written for VAX.

- The power consumption should be as low as possible, so that it could be used as a stand-alone speech communication aid.

To fulfill the hardware requirments. The 68000 microprocessor is chosen and all the components are CMOS due to the power consumption consideration.

The hardware consists of a microprocessor, a maximum of up to 128 kbytes of RAM, 1024 kbytes of EPROM, an asynchronous communications interface adapter (ACIA) 6350, a peripheral interface adapter (PIA) 6321, a hardware formant synthesizer-PCF 8200 and a programmable timer(PTM) 6340.

3.2 68000 brief description

The 68000 contains seventeen 32 bit registers, one 32 bit program counter and a 16 bit status register. The 68000 CPU register model is shown in Fig.3-1. The first eight registers (d0-d7) are used as data registers for byte (8bit) word (16bit), and longword (32bit) data operations. The second set of seven registers (a0-a6) and the system stack pointer may be used as software stack pointer and base address registers.In addition, these registers may be used for word and longword address operations. All seventeen registers may be used as an index register.

The input and output signals can be functionally organized into groups, which are shown in Fig.3-2.

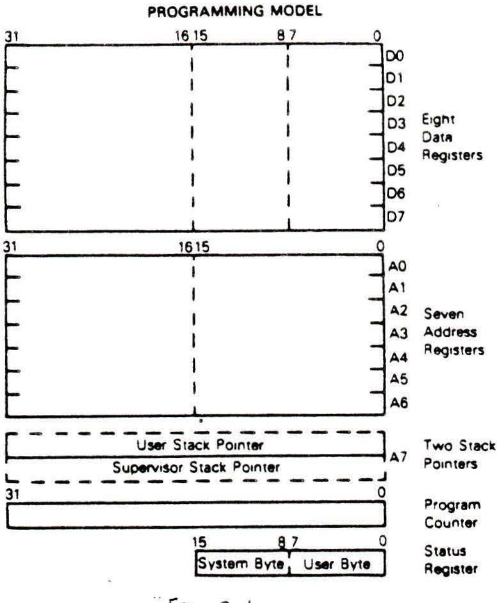


Fig. 3.1

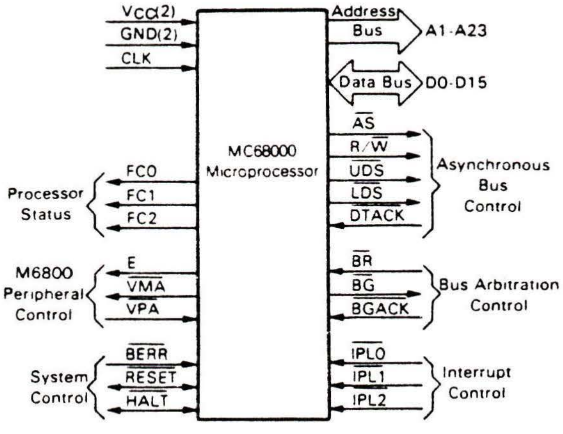


Fig. 3.2

3.3 Memory map

The contents that have to be mapped into RAM area are:

-Buffer of input sentences.

-Buffer of output information to be send to the synthesizer.

-Stack

The RAM area is mapped from $800000-83ffff.

The contents that have to be mapped into EPROM area are:

- phoneme table (fontab).
- diphone table.
- speech code table.
- program

Because the 68000 microprocessor uses memory mapped I/O, the addresses of peripheral devices are also in memory space.

All address of I/O devices, the RAM and the EPROM areas are listed in table-3.1

TABLE 3.1

$000000	EPROM
$07FFFF	
$800000	RAM
$83FFFF	
$840000	PCF 8200 SYNTHESIZER
$850000	PTM 6340
$860000	PIA 6321
$870000	ACIA 6350

3.4 Interface with memory

In the system, HM 6264 and HN 27C256 were chosen as RAM and EPROM. The access time is 150 ns and 200 ns respectively. The main clock frequency of the 68000 is 8MHZ, so T=125 ns. There is a problem. That is ,the external hardware is not electrically prepared to complete the communication started by the 68000 in one clock cycle. To solve this problem, the DTACKN input to the 68000 is used. We delay the DTACKN for a certain amount of time. In this way, the access time for each read or write cycle is controlled because the CPU waits for the DTACKN signal and inserts an unlimited number of wait states.

Fig. 3.3 shows the detailed hardware description of the time-out-counter and the jumper for the time delay.

The timing diagram is given in Fig. 3.4.

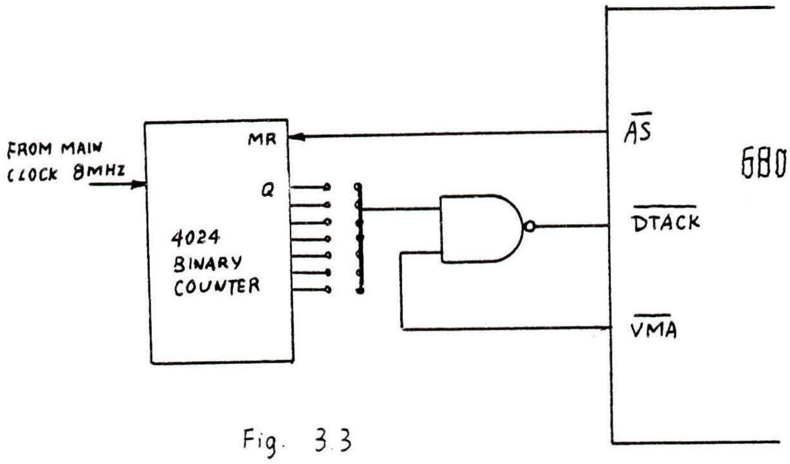


Fig. 3.3

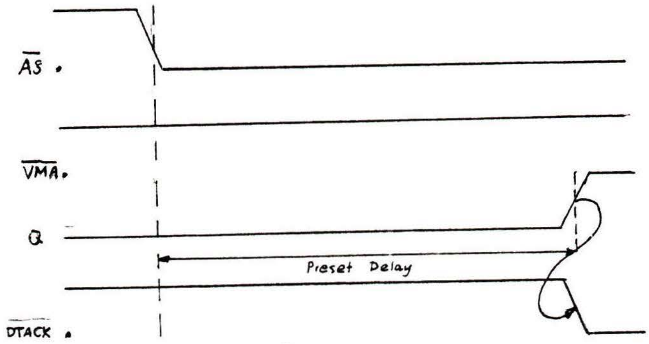


Fig. 3.4

3.5 Interface with ACIA and PIA

The 68000 CPU device is capable of operating at clock speeds up to 10 megahertz. The 6350 ACIA and 6321 PIA are capable of operating at speeds near 1 Mhz and slower. It seems difficult to match them. Fortunately, there is a signal line of the 68000 CPU (VPAN) special for the purpose of this situation. When this input is asserted in the logical'0' state, the 68000 knows electrically that the communication cycle is to be with a 6800-type peripheral device.

When the VPAN input is asserted the 68000 'synchronises' up to its output line labled 'E pin 20'. The E output is the input clock frequency of the 68000 on pin 15 divided by 10. When using 6800 (including 6350 ACIA and 6321 PIA) peripheral devices with the 68000, the 'E' output line takes on the same function as the phase 2 clock in a 6800-based system.

Fig. 3.5 illustrates the schematic diagram of how we will assert the VPAN input to the 68000. In Fig.3.5 the proper address space is decoded and must be qualified by the ASN output line from the 68000 and it is combined with the interrupt acknowledgement from the 68000 also. These will ensure that the 68000 receives the VPAN input at the proper time in the communication cycle.

Fig.3.6 is the timing diagram.

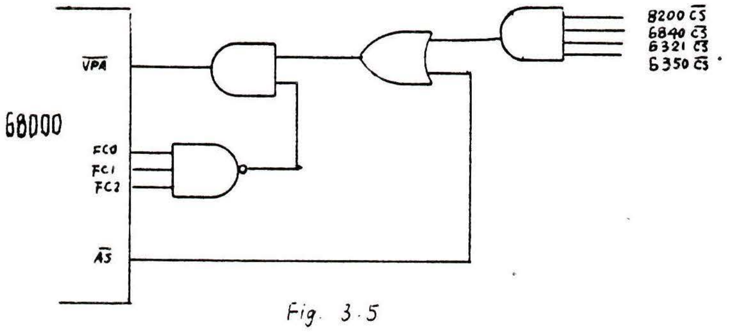


Fig. 3.5

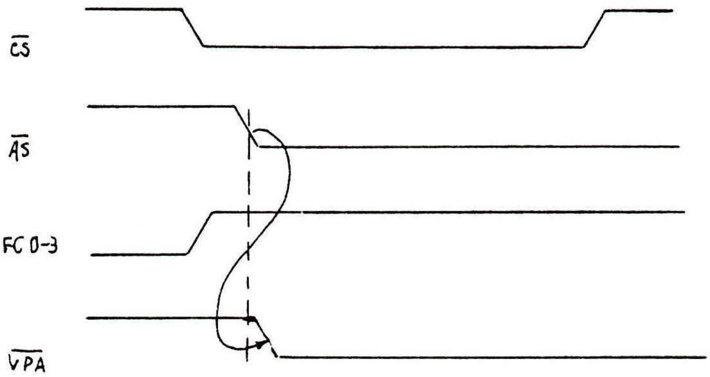


Fig. 3.6

3.6 Interface with synthesizer PCF 8200

The PCF 8200 is a hardware formant synthesizer. Its principle has been described in the section 2.3. Fig. 3.7 is the block diagram. The synthesizer receives data to be processed and pronounced. It requests data via the interrupt mechanism which can be enabled by setting a bit in the status register. Data can be written to the synthesizer when REQN=0 or when REQN=1 and BUSY=0. Fig.3.8 shows the interface protocal.

In addition the command register must be written once in order to set the operation mode of the synthesizer. A command write exists of two bytes and it may occur before a data block.

The timing and wiring diagram are shown in Fig.3.9, Fig. 3.10.

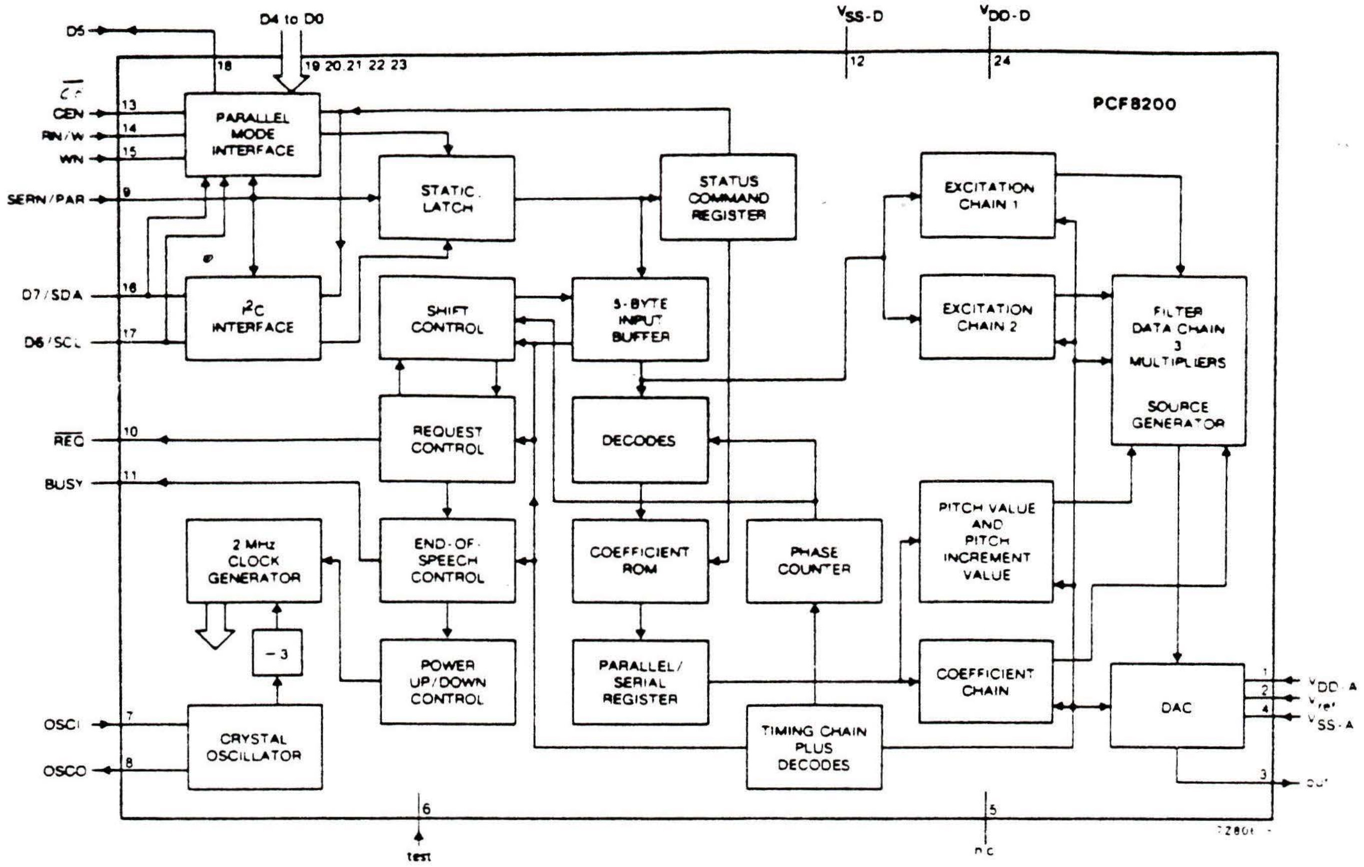


Fig.3.7 Block diagram.

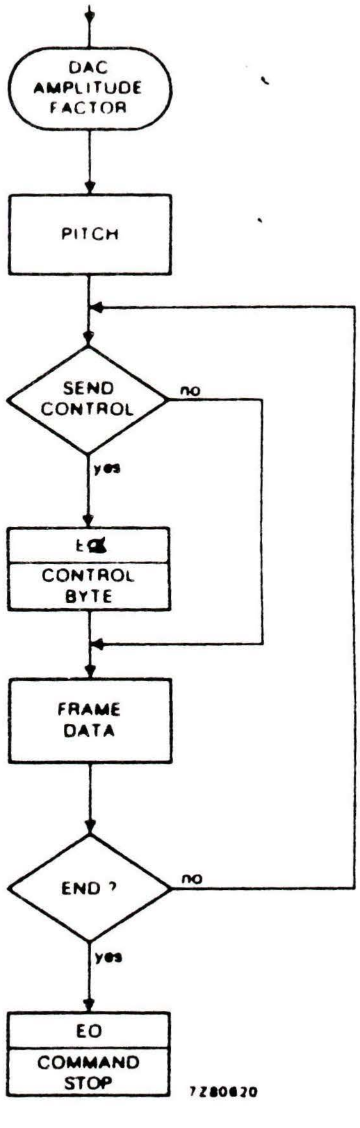
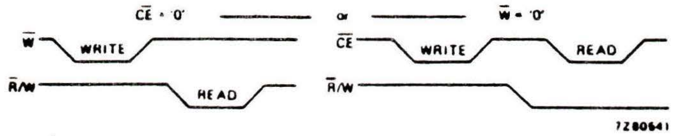
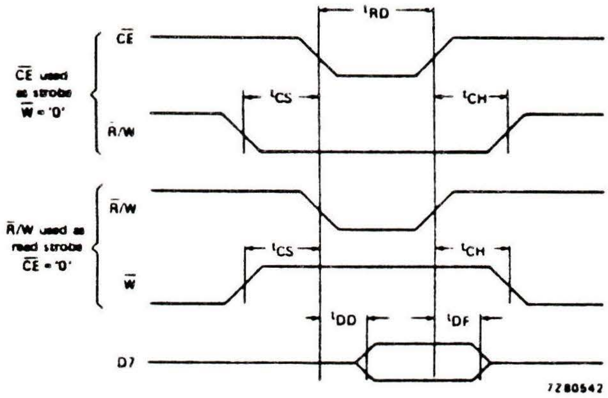


Fig. 3.8Interface protocol.



Typical connection of control signals.



Read timing.

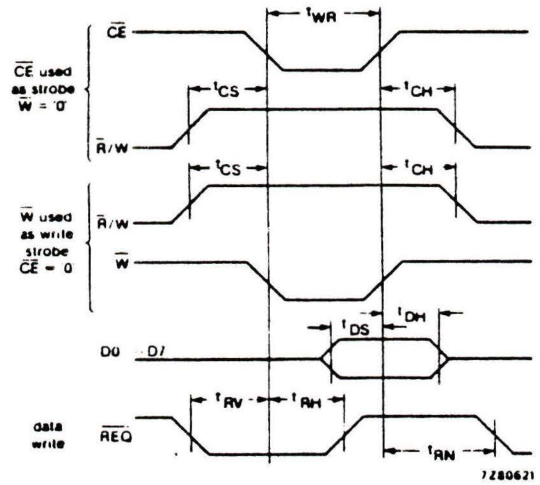


Fig.3.9 Write timing.

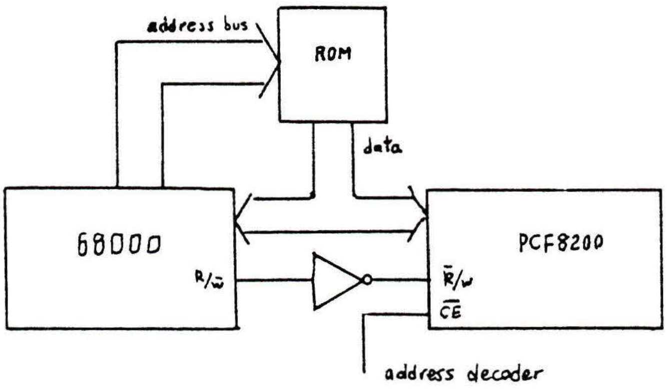


Fig 3.10

CHAPTER 4

TESTING OF THE SYSTEM

4.1 CPU Test

```
.sect .vect
      .data4  0x0
      .data4  0x400

.sect .text
.base 0x400
loop: nop
      jump    loop
```

4.2 ACIA Test

```
#
#define       ACIACR  0X870001
#define       ACIASR  0x870001
#define       ACIAT   0X870003
#define       data    0x15
#define       reset   0x3

.sect .vect
      .data4  0x0
      .data4  0x400

.sect .text
.base         0x400

              move.b  #reset,ACIACR
      move.b  #data,ACIACR
stat: move.l  #text,a0
loop: btst    #1,ACIASR
      beq     loop
      move.b  (a0)+,d0
      beq     end
      move.b  d0,ACIAT
      jmp     loop
end:  jmp     stat
text: .asciz  "hallo"
```

4.3 RAM Test

```
#
#define       ram     0x800000
#define       nr1     0xffff
#define       nr2     0xaaaa
#define       nr3     0x5555

.sect .vect
      .data4  0x0
      .data4  0x400

.sect .ramtest
.base 0x400

loop: move    #nr3,d0
      jsr     test
      move    #nr1,d0
      jsr     test
      move    #nr2,d0
      jsr     test
      move    #nr3,d0
      jsr     test
      jmp     loop

test: move.l  #ram,a0
      move    d0,d1
l1:   move    d0,(a0)
      not     d0
      move    (d0)+,d0
l2:   cmp     d0,d1
      bne     l2
      cmp.l   #0x804000,a0
      bne     l1
      rts
```

4.4 Interrupt Test

```
#
#define       aciacr  0x870001
#define       aciar   0x870003
#define       control 0xd5
#define       reset   0x3
#define       int     0x500
#define       flag    0x800040
#define       key     0x800041

.sect .vect
      .data4  0x800010
      .data4  0x400
.sect .auto
.base 0x7c
      .data4  int
.sect .inttest
.base 0x400

      clr.b   flag
      move.b  #reset,aciacr
      move.b  #control,aciacr
wait: tst.b   flag
      beq     wait
loop: move.b  key,d0
      jmp     loop

.sect .recint
.base int
      move.b  #1,flag
      move.b  aciar,key
      rte
```

4.5 Synthesizer Test

```
#
#define       pcf     0x840001
#define       dac     0x30
#define       status  0x840001

.sect .vect
      .data4  0x800010
      .data4  0x400

.sect .pcf8200test
.base 0x400
stat: move.l  #0x600,a0
      move    (a0),a1
      move.b  (a0)+,d0
      move.b  (a0)+,d0
      move.b  (a0)+,d1
      move.b  #dac,pcf
      jsr     wait
      move.b  d1,pcf
      jsr     wait
      move.b  #0xe0,pcf
      jsr     wait
      move.b  do,pcf
      sub     #5,a1
loop: jsr     wait
      move.b  (a0)+,pcf
      sub     #1,a1
      cmp.w   #0,a1
end:  beq     end
      jmp     loop

wait: btst    #7,status
      bne     wait
      rts
```

CHAPTER 5

A PROGRAM EXAMPLE

```
#
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
! DEMONSTRATION PROGRAM FOR 68000 BOARD
!
! Copyright november 1986
! Institute for perception research,IPO
! Eindhoven, Holland
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#define ACIACR  0x870001
#define ACIASR  0x870001
#define ACIADR  0x870003
#define control 0x95
#define reset   0x3
#define flag    0x800040
#define input   0x800044
#define PCF     0x840001
#define dac     0x30
#define INT     0x500
!
.sect   .vect
        .data4  0x800010        ! stack
        .data4  0x400           ! program
!
.sect   .auto
.base   0x7c                    ! level 7 autovector
        .data4  INT
!
.sect   .main
.base   0x400
! initialisation
        clr.b   flag            ! key pressed flag
        move.b  #reset,ACIACR   ! reset ACIA
        move.b  #control,ACIACR ! init ACIA
! send prompt
prompt: move.l  #text,a0
loop:   btst    #1,ACIASR       ! TR empty ?
        beq     loop
        move.b  (a0)+,d0
        beq     end                     ! last char ?
        move.b  d0,ACIADR               ! send char
        jmp     loop
end:    jmp     read
text:   .asciz  "\n\r zin nr. : "! prompt
! read input
read:   tst.b   flag                    ! key pressed ?
        beq     read
        clr.b   flag                    ! clear flag
        move.l  #0,d0
        move.b  input,d0
        move.b  d0,ACIADR               ! input echo
        sub     #'0',d0                 ! convert ASCII to digit
        lsl.l   #1,d0                   ! * 2
! speak sentence
        move.l  #0x600,a0               ! start speach data
        move.l  #0x600,a1
        add     0(a0,d0),a1             ! add offset in speach block
        move.l  #0,d0
        move.b  (a1)+,d0                ! length (upper)
        lsl.l   #8,d0
        move.b  (a1)+,d0
        move    d0,a2
        move.b  (a1)+,d0                ! skip 00
        move.b  (a1)+,d0                ! comnd
        move.b  (a1)+,d1                ! pitch
        move.b  #dac,PCF                ! dac factor
        jsr     wait
        move.b  d1,PCF                  ! pitch
        jsr     wait
        move.b  #0xe0,PCF               ! comnd start
        jsr     wait
        move.b  d0,PCF                  ! comnd
        jsr     wait
        sub     #5,a2                   ! update length
speak:  jsr     wait
        move.b  (a1)+,PCF               ! send speech data
        sub     #1,a2                   ! decrement length
        cmp.w   #0,a2
        beq     ready                   ! back to prompt
        jmp     speak
ready:  jmp     prompt
!
wait:   btst    #7,PCF                  ! test PCF status
        bne     wait
        rts
!
.sect   .aciaint
.base   INT
        move.b  #1,flag                 ! set flag
        move.b  ACIADR,input            ! read input
        rte
```

CHAPTER 6

REFERENCES

1. "Applications of digital signal processing"
   by A.V.Openheim

2. "Manipulation of speech sounds"
   by J.'t Hart, S.G.Nooteboom,L.L.M.Vogten and
   L.F.Willems, Philips tech. Rev.1982, No.5

3. "Towards a keyboard-to-speech system"
   by P.G.Menting, IPO Annual Progress Report 19

4. "Development of hardware and software for the diphone speechboard"
   by E.J.Falzoni, PII report

5. "SC68000 microprocessor user manual"
   Fourth Edition

6. "SYS68K/CPU-1 User's Manual"
   Force computer advanced systems

7. "Using and Troubleshooting the MC68000"
   by J.W.Coffron

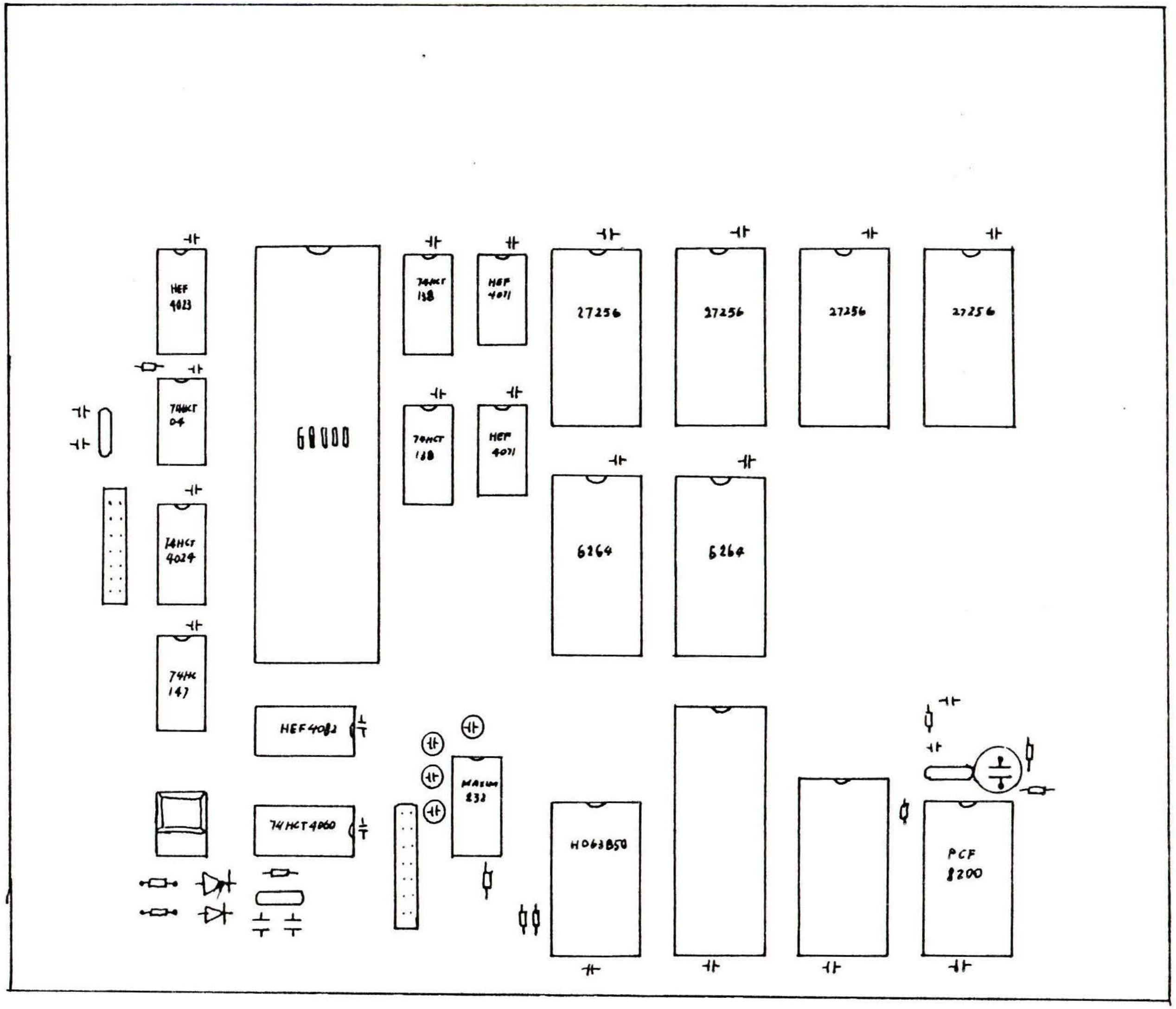
8. "68000 microprocessor handbook"
   by Gerry Kane

9. "System Secification voice synthesizer PCF8200"
   by M.ten Have, Philips CAB Elcoma Report

10. "Data sheet -68000 CPU
    -6350 ACIA
    -6321 PIA
    -6340 PTM"
    -HM6264 RAM
    -HN27C256 EPROM"
    High speed CMOS databook

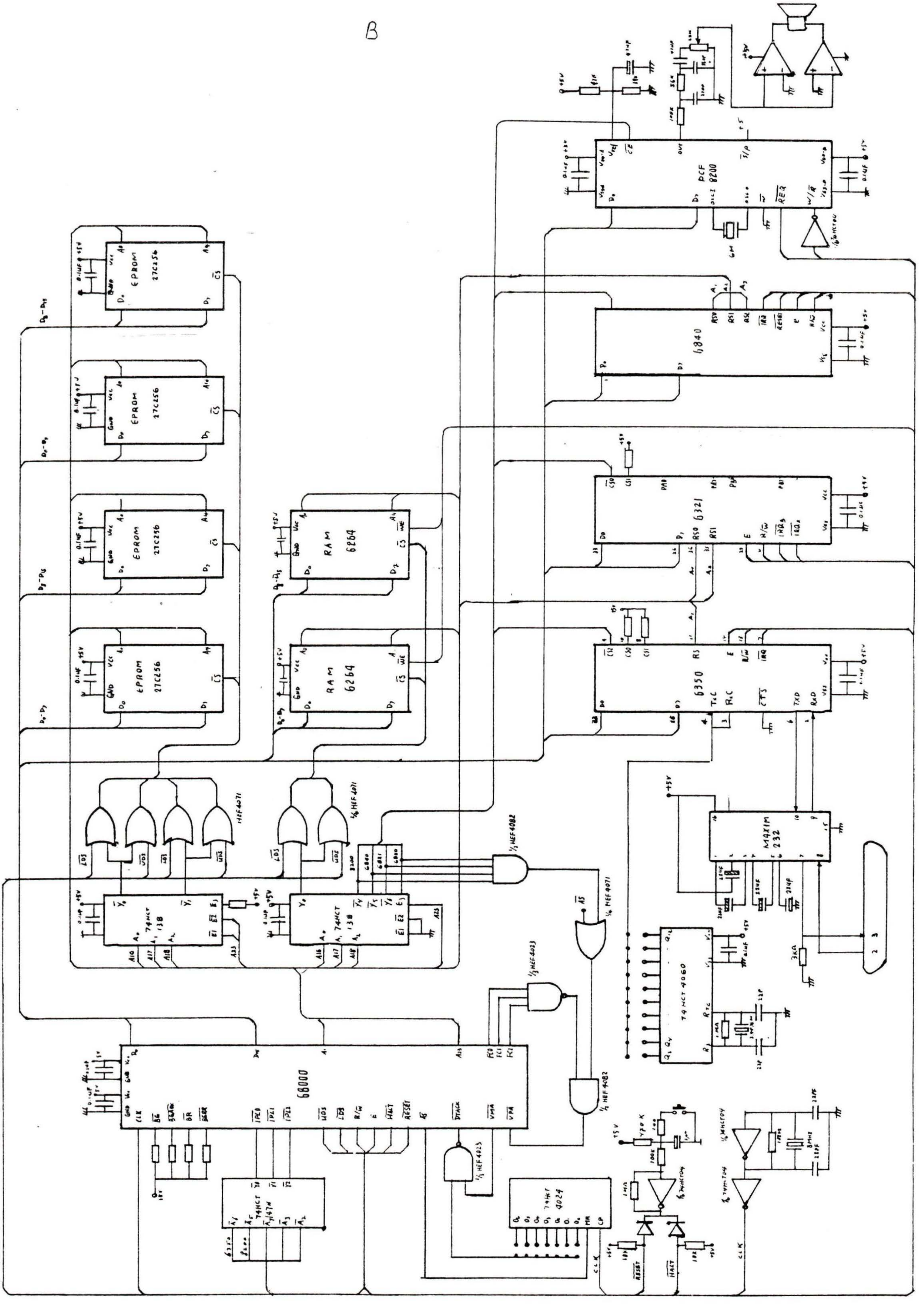
APPENDIX

A



Location diagram

B


PCF 8200
6840
6321
6350
MAX1M 232
74HCT 4060
EPROM 27C256
EPROM 27C256
EPROM 27C256
EPROM 27C256
RAM 6264
RAM 6264
HEF4071
¼ HEF4011
74HCT 138
74HCT 138
½ HEF4082
¼ HEF4071
⅓ HEF4023
½ HEF4082
68000
74HCT 147N
⅓ HEF4023
74HCT 4024
⅙ 74HCT04
8MHz
6M
3KΩ
D0-D7
D8-D15
RESET
HALT
CLK