4-1-1995

The effects of chorded keyboards on portable computing devices

Tiki L. Suarez
Clark Atlanta University

ABSTRACT

COMPUTER SCIENCE

TIKI L. SUAREZ

THE EFFECTS OF CHORDED KEYBOARDS ON PORTABLE COMPUTING DEVICES

Advisor: Professor Byron Jeff

Thesis dated April, 1995

This study will examine if a chorded keyboard will allow us to build more compact portable computers with faster input rates than the present portable computing input devices. The two step approach includes designing and developing a chorded keyboard and an alphanumeric mini keypad along with the actual usability experiment. The usability experiment will compare the input rates of the chorded keyboard to the mini keypad in different applications. Hence, the goal of this research is to compare the input rates of chorded keyboards to the input rates of mini keyboards for portable computers.

THE EFFECTS OF CHORDED KEYBOARDS ON PORTABLE COMPUTING DEVICES

A THESIS

SUBMITTED TO THE FACULTY OF CLARK ATLANTA UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

BY

TIKI L. SUAREZ

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE

ATLANTA, GEORGIA

APRIL 1995

R= viii T= 86

ACKNOWLEDGEMENTS

First, I would like to thank the Lord for all of His many blessings. Without Him I would not be where I am today. I would also like to thank my parents, CSM(Ret) Louis F. and Clara L. Suarez for their untiring love and consistent encouragement. I love you both with all of my heart. This is dedicated to you. With all sincerity, I would like to thank my research advisor, Byron A. Jeff for his patience, guidance and support. It was a pleasure to work with you. I would also like to thank all of my colleagues who assisted me in my research. Special appreciation goes to Jerome Farquharson, Lundy King, Anderson Cross, Phillip Nelms, Cynthia Bryant, Phillippe Jean-Bart and Rahman Pounds for their time and assistance with my experiments. I would also like to thank the DIVAS and the ONR office for their support and advice. To my committee members, Dr. Kenneth Perry and Dr. David Kerven, I truly appreciate all of your contributions to my research, advice and letters of references. To Leslie Smart, thank you for your "psychological" insight. To my closest friends and sorors, Cynthia Bryant and Courtney Smith, I would like to especially thank both of you for your continued support, assistance, and friendship. May God continue to bless each one of you.

TABLE OF CONTENTS

Appendix

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ASCII	American Standard Code for Information Interchange
CMOS	Complementary Metal Oxide Semiconductor
EEPROM	Electrically Erasable Read Only Memory
RISC	Reduced Instruction Set Computer

CHAPTER 1

HISTORICAL

Today's society is filled with the latest technological devices. From the huge processors in the past that filled the entire room to the desktop and portable personal computers of the present day, advances in computer software and hardware have increased significantly. Information can be processed with tremendous speed, hard drives have dramatically increased in size and memory upgrades become less expensive. Unfortunately, little emphasis has been on the devices that actually input the data, programs, and applications. Input systems, the primary interface for humans, are a vital part of the information processors they serve. The research in the area has created several input devices that produce faster input rates than the standard keyboards, but they have gone largely unnoticed because of the dominance of the conventional keyboard.

Input devices for portable computers have considerable problems such as large number of keys for a small available space. Input technology for portables in open area of research have produced alternatives devices like voice automated, light pens, and the mouse, although each have shortcomings compared to the conventional keyboards. Additional research in this area is warranted. This particular study compares the performance of another two alternative input devices; the chorded keyboard and the alphanumeric keypad in a portable computing environment.

QWERTY Layout

There has basically been one standard input device layout for the computer and that is the QWERTY keyboard layout. This layout typically includes alphanumeric, function, auxiliary numeric and cursor control key groups. The greatest attention has been focused on the layout of the alphanumeric group, which is composed of the uppercase and lower case alphanumeric characters, ten numerals, punctuation marks, and special symbols such as ampersand and asterisk. The primary focus of the QWERTY keyboard has been on this alphanumeric group the most. Advantages of the QWERTY layout are as follows (Greenstein and Muto 1988, 134):

1. Widespread acceptance and use
2. Accepted as an American National Standard
3. Usually alternate between hands
4. Bottom row contains least frequent letters
5. Ring and little fingers key least frequent letters
6. Relatively few successive keystrokes by same finger

The QWERTY layout has been adopted as the basis for a standard alphanumeric keyboard arrangement (American National Standards Institute 1982, 134). Because of the widespread acceptance of the QWERTY keyboard, researchers and inventors use caution when deviating from this layout. In 1873 the Sholes brothers organized the keys of the QWERTY layout to minimize the jamming of type bars; hence, the performance of the input device was deliberately slowed (Norman and Fisher 1982, 509). Since jamming is not a major concern with modern type bars and typing elements, electric typewriters and computer terminals, the need for another, possibly faster input device has presented itself.

However, the strongest argument against any proposed changed is simply that the Sholes keyboard, universally known as the QWERTY keyboard, is so well known and used that it would not be practical to change.

Dvorak Layout

Of the devices designed to produce faster input rates than the QWERTY layout, the Dvorak layout has proven the most enduring. The arrangement emphasizes an efficient layout of the keys to minimize hand and finger motion. (Dvorak 1943, 58). A variant of this layout has in fact been accepted by the American National Standards Institution as an alternative (American National Standard Institute 1982, 140). The Dvorak was designed to the following criteria (Noyes 1983, 57):

1. The right hand was given more work (56%) than the left hand (44%).
2. The amount of typing assigned to different fingers was proportional to their skill and strength.
3. Seventy percent of typing was carried out on the home row - the most frequently used letters were arranged on this row. Only 22% and 8% of typing was carried out on the top and bottom rows, respectively.
4. Letters often occurring together were assigned positions so that alternate hands could strike them.
5. Finger motions from row to row and difficult, awkward reaches from the home row were minimized.
6. Thirty-five percent of the words typically used were typed exclusively on the home row.

One study of the Dvorak stated that its layout is easier to learn, is less fatiguing to use, and permits faster data entry with fewer errors (Yamada 1980, 183). Advantages of the Dvorak layout are as follows (Greenstein and Arnaut, 1987).

1. Increased efficiency
2. Accepted as an American National Standard
3. Increased use of alternate-hand keying
4. Increased use of home row
5. Amount of keying assigned to fingers is proportional to finger strength and skill
6. Minimal awkward finger movement
7. Increased use of right hand

Alphanumeric Keypad Layout

According to Greenstein and Muto as well as other past researchers, it seemed reasonable to assume that an alphabetically ordered keyboard would enhance speed and accuracy. The alphabetical layout should provide an easily understood structure that aids in the search for desired keys. Several studies of alphabetic layouts have been reported and their conclusions are consistent in that the alphabetical layout does not appear to offer any practical performance advantages relative to the QWERTY layout. However, an alphabetically ordered key layout does enhance the performance of unskilled users relative to a randomly structured layout (Greenstein and Muto 1988, 135). Alphabetical layouts may first involve a memory search to locate the letter's position in the alphabet, followed by a visual search for the key on the board. According to (Norman and Fisher 1982, 509) "Performance with some alphabetical layouts is quite slow, but with others, it is within 2% of the speed achieved when using the Sholes (QWERTY) keyboard."

Chorded Keyboards

As technology continues to evolve with faster and more powerful tools such as computers, so does the desire for a faster input device. The Chorded Keyboard is a handheld input device that fits in the palm of either hand. It is controlled by simultaneous patterned pressing of one or more keys to produce alphanumeric symbols, hence the term "chord". This is analogous to playing a chord on the piano. This layout requires fewer keys being needed on the input device compared to sequential layouts. Five keys allow a total of 31 (2^5-1) different chord combinations to be generated. The chorded keyboard will release data on the release of the switches unlike the pressing of buttons with ordinary keyboards.

D.C. Englebart developed a successful rudimentary one-hand chord keyboard as part of a terminal (Bequaert et al., 1978, 62). His objective was to make one hand free to operate a cursor that performs the function of the CRT display light pen. The US Post Office also developed and tested a rudimentary 10-key keyboard as a part of a system for controlling the flow of mail (Bequaert et al., 1978, 62). Feasibility of using a chorded keyboard as an input device was first seriously investigated in the mid - 1950's by the Canadian Post Office (Noyes 1983, 63). Studies in this area reached a climax in 1960 with IBM who studied two chorded keyboards to rival the typewriter. In the 1970's chorded keyboards became commercially available with the emphasis on the development of keyboards moving from a specific task to more general purpose applications.

The study conducted by (Levy 1955, 10) was one of the first to describe a 10-key 'binary' keyboard for use at the Toronto Post Office. The keyboard consisted of two 5-key keyboards and by pressing the correct combination of keys, the addresses could be condensed into a series of characters. The use of chord keying sorting resulted in higher throughput rates sorting than manual. Five years later, a study conducted by (Conrad 1960, 111) reported a letter sorting machine involving simultaneous depressions of two keys (one by each hand) in order to sort letters into one of 144 possible destinations. It was found that sorting rates improved from approximately 35 sorts/min to about 60 sorts/min using the keyboard over a practice time of 39 weeks.

One of the most comprehensive studies performed on sequential and chord keyboards were two experiments carried out in 1965. One study by (Conrad and Longman 1965, 111) compared the performance of two groups of postmen one on chorded and one on QWERTY keyboard. The chord keyboard had 10 'home' keys, which were depressed in pairs using the left and right hands. The time allowed to depress the two keys was 50ms, and if more keys were depressed or the 50 ms ran out, an error was registered. Both these situations did not arise with the standard keyboard. The subjects that participated in the experiment were between the 30-40 years age group. Twenty-four individuals were allocated at random to the typewriter group and Twenty-two to the chord group. No subjects had prior typing experience and they trained for 3½ hrs daily, five days a week for seven weeks. This seven week course took place on four occasions making the total length of the experiment about 10 months. Conrad and Logman discovered that the group using the chord keyboard became 'operational' about two weeks earlier than the standard typewriter group, that is, in two weeks instead of four and after that their improvement rates could be regarded as parallel. At all times during training the chorded group's rate exceeded the typewriter group (Conrad and Longman 1983, 112).

Another 5-key chorded keyboard was invented by (Bequaert, Rochester and Sharp 1978, 57). A study was performed to show that the keyboard prototype offers one-handed operation, small size, low cost of manufacture and permits the touch typing of large alphabets. An operator may press up to three finger dimples at once, producing a string of letters in one stroke. Typical examples of chords are "the" and "fro", which common strings in English.

A chorded keyboard can be considered a 'blind' method of typing, where each finger of a hand has a small number of keys or one key assigned to it. According to (Biegel 1967, 281) there are two chief advantages of 'blind' typing, which are:

1. It dispensed with the thousands of small controlling movements of eyes and head that are necessary in 'sight' typing, and therefore involves less physical and mental fatigue.
2. In the long run, it permits of greater speed and accuracy than 'sight' typing.

CHAPTER 2

HARDWARE/SOFTWARE DESIGN

The hardware/software design is achieved by a designing and developing the chorded keyboard development board and the mini keyboard-keypad development board. These development boards are the interface between the input device and the personal computer because neither the chorded keyboard nor the alphanumeric keypad have the capability to transmit or receive data. The prototype of each keyboard consists the following:

1. 16C84 Microcontroller
2. MAX 232 serial interface
3. 10MHz crystal oscillator.

The alphanumeric keypad has an additional 74LS138 Decoder chip on its development board along with a 9.6MHz crystal oscillator. Program coding in assembly and compilation is done on the PC and downloaded to the hardware - PIC microcontroller development board. The schematics for the Chorded Keyboard and the Alphanumeric Keypad are as follows:

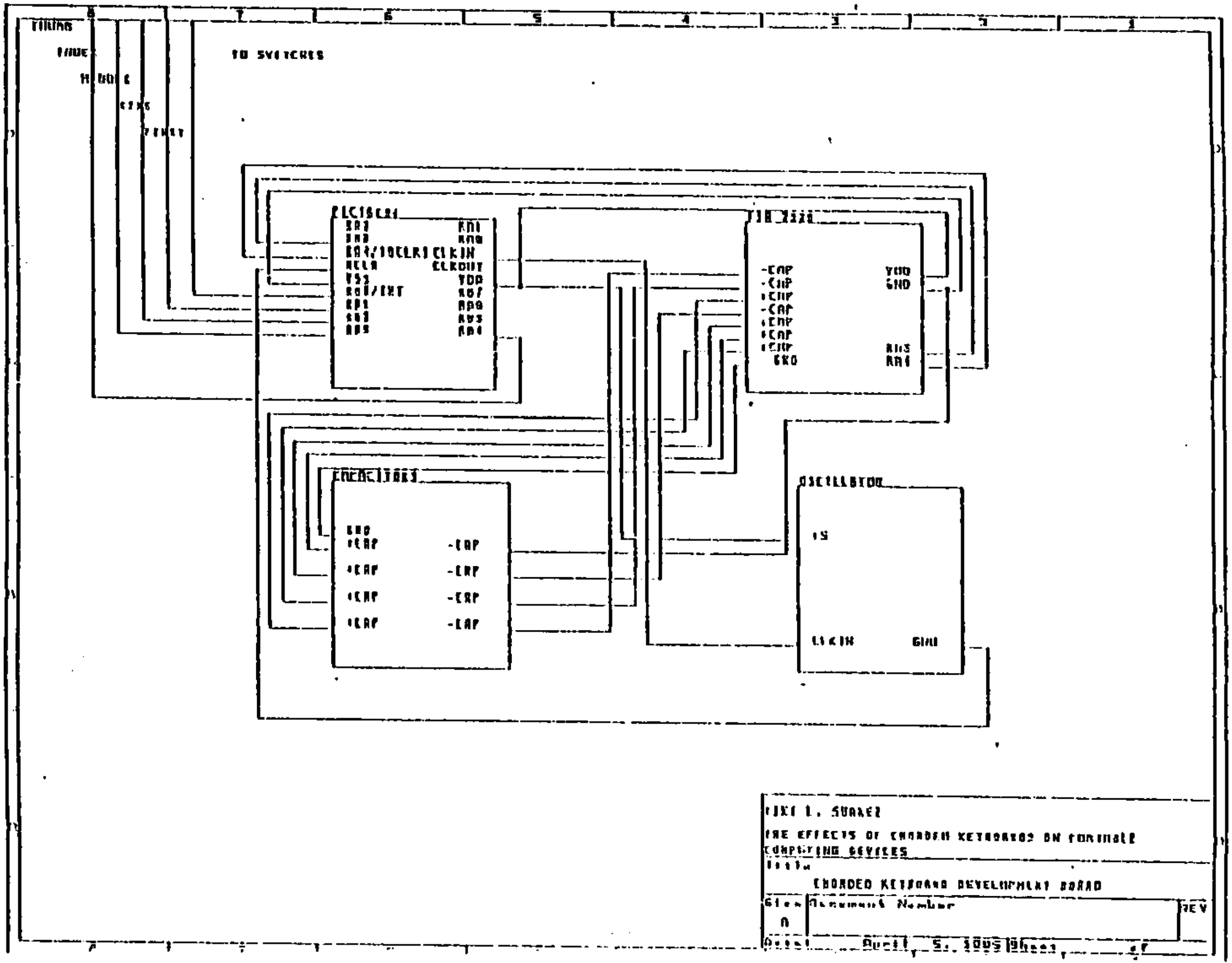


Fig. 1. Schematic of Chorded Keyboard

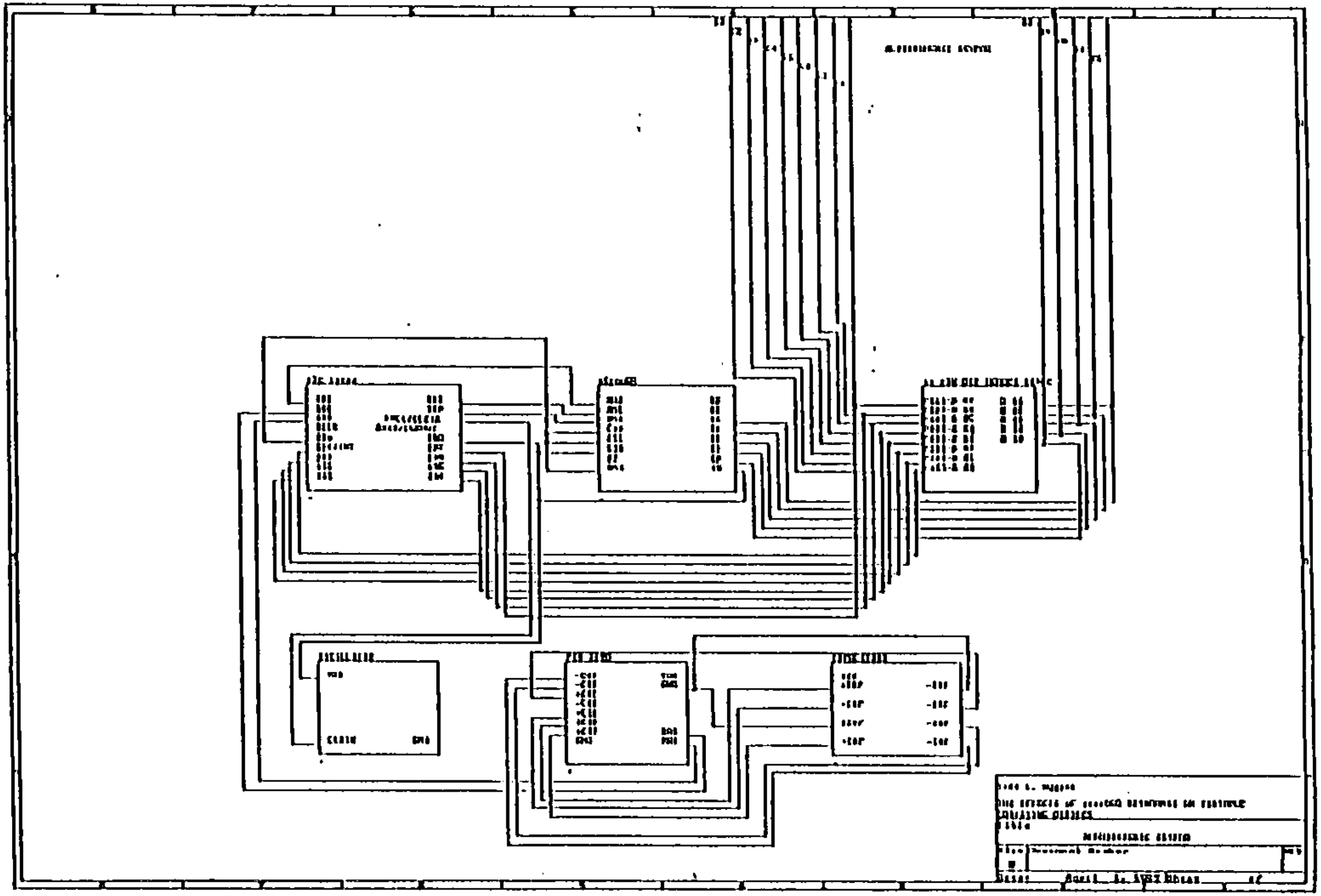


Fig. 2. Schematic of Alphanumeric Keypad

Hardware Design for Input Devices

The PIC16C84 is a high-performance, low-cost Complementary Metal Oxide Semiconductor (CMOS), fully-static 8 bit microcontroller with 1K x 14 Electrically Erasable Programmable Read Only Memory (EEPROM) program memory and 64 bytes of EEPROM data memory (Microchip 1994, 535). It is the second member of an enhanced family of PIC 16CXX microcontrollers. The PIC16C84 has the following features:

1. Two parallel ports:

   Port-A is a 5-bit wide port with pins RA<4:0>. Port pins RA<3:0> are bi-directional whereas RA4 has a open collector output.

   Port-B, is an 8-bit wide bi-directional port. Reading Port-B register reads the status of the pins whereas writing to it will write to the port latch. Port-B also possesses weak internal pull-ups.
2. EEPROM program memory for code development and One-Time-Programmable memory for full production
3. The EEPROM data memory (64-bytes) readable and writable during normal execution at full V_{DD} range (2.0V - 6.0V).
4. Employs an advanced RISC-like architecture. A reduced set of 35 instructions, single word instruction (14-bit wide), single cycle instructions (400ns at 10MHz clock) which take two cycles (880ns) instruction pipelining, large register set and separate instruction and data memory (Harvard architecture) schemes.
5. Achieves a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in its class.
6. Reduce external components and thus reduce cost, such as the High current drive (25mA max, sink, 20mA max source) of the I/O pins,

thus reducing external components and cost.

7. Enhance the system reliability and reduce power consumption.
8. Four interrupt sources and an eight-level hardware stack.
9. Peripherals include an 8-bit timer/counter with an 8-bit prescaler (effectively a 16-bit timer) and 13 bi-directional I/O pins.
10. Supported by an assembler, an in-circuit emulator and a production quality programmer which are supported on IBM PC® and compatible machines.

The MAX 232 allows serial transmission between the PIC 16C84 microcontroller and the terminal. The 10MHz crystal oscillator drives the system at an instruction cycle frequency of 2.5 MHz. A 9.6MHz crystal oscillator was used for the alphanumeric keypad development board. Capacitors were used to stabilize the oscillator.

Software Design for Input Devices

The Software development was performed in several stages. Because the PIC16C84 does not have a on-chip hardware asynchronous serial port, is was necessary to develop this particular element in software. SERIAL.ASM, found in Appendix 1, performs the asynchronous serial transmission/reception of data. This program is a major component of the software design implementation.

The following is a description and diagram of SERIAL.ASM:

1. Main loop consists of two main routines: getc and putc.

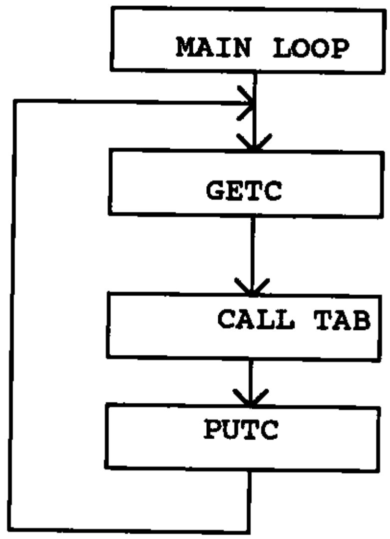


Fig. 3. Serial.asm

2. Getc gets one character from the input mode Port-A,4 and stores it in a temporary register. This is achieved by checking for a start bit (0) from Port-A,4. The test checks the center of the cell.
   a) When (0) is received, the COUNT register is set to 8, for eight data bits to be read in.
   b) Port-A,4 is tested for an incoming bit either (0) or (1).
   c) Bit is then rotated in the temporary register and COUNT is decremented.
   d) The routine then checks for a stop bit (1).
   e) Procedure is executed 8 times before returning to the main loop.

```
;the following is actual code of getc (In PIC16C84 Assembly)
     getc btfsc   Port_A,4
          goto    getc
          movlw   .53        ;check the middle

                             ;of the next bit

          call    delay
          btfsc   Port_A,4   ;check for start
                             ;bit (0)
          goto    getc
          movlw   .8
          movwf   COUNT      ;set number of
                             ;bytes to read as 8

     gtest movlw   .107
          call    delay
          btfsc   Port_A,4 ;read in 8 bits
                           ;of data
          goto    gone     ;if Port_A is a
                           ;'1' then put '1' in
                           ;XMIT
          bcf     STATUS,CARRY    ;if Port_A
                                  ;is a'0' then put
                                  ;'0' in XMIT
          goto    gnext
          gone   bsf      STATUS,CARRY

    gnext rrf    XMIT,1
        decfsz COUNT,1
        goto  gtest

        movlw  .107
        call   delay
        movf   XMIT,0

        return
```

3. Putc routine displays one character on the terminal by sending one bit to the output mode Port-A,3.

a) Newly stored string is placed in the working register.

b) Test for a start bit, (0).

c) When start bit is found the POS (position) register is set to 8, for the eight data bits.

d) The string is rotated because the least significant bit was stored first in the temporary register.

e) A test is performed to see if the current bit is a (0) or a (1) and is

f) placed in Port,3

g) The POS register is then decremented and the routine checks for the stop bit (1).

h) This is achieved by using the delay routine to check the status of the next bit in the middle of its cell.

i) Finally, the procedure is executed 8 times before returning to the main loop.

```
;the following is actual code of putc(in 16c84) Assembly
        putc    movwf   XMIT
                bcf     Port_A,3        ;start
                                        ;bit
                movlw   .115
                call    delay
                movlw   .8              ;loop 8
                                        ;times
                movwf   POS
        ptest   rrf     XMIT,1          ;rotates the
                                        ;bits in XMIT
                btfsc   STATUS,CARRY    ;tests
                                        ;to see if the
                                        ;bit is zero or
                goto    pone            ;one and puts
                                        ;it in Port_A,3
                bcf     Port_A,3
                goto    pnext
        pone    bsf     Port_A,3

        pnext   movlw   .107
                call    delay
                decfsz  POS,1
                goto    ptest
                bsf     Port_A,3        ;stop
                                        ;bit
                movlw   .115
                call    delay

                return
```

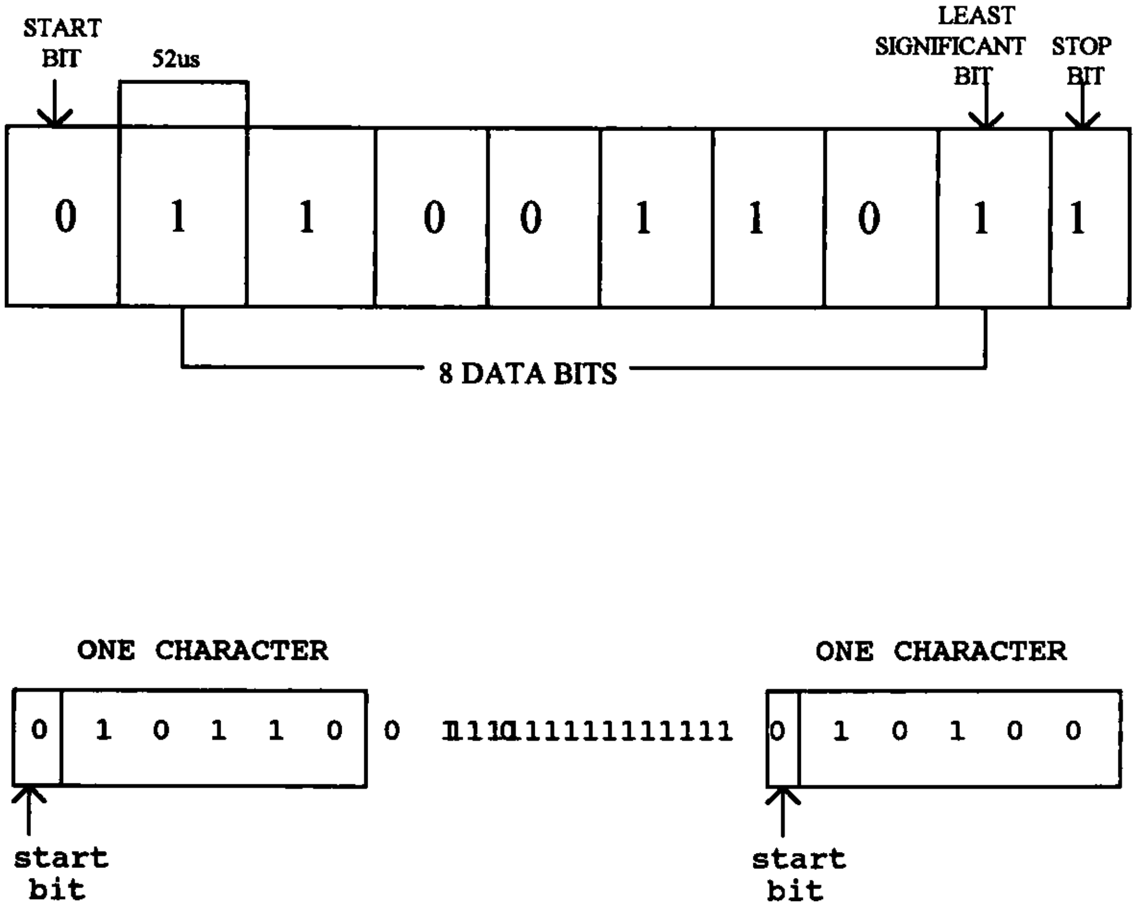
4. The delay subroutine permits both the putc and gets routines to wait for a predetermined amount of time before another action takes place. This delay is necessary to keep timing between cells of the asynchronous

character. The following is the code for the delay routine a and diagram for an asynchronous serial transmission.

```
;the following is actual code of delay routine
       delay
              clrf     TMR0
              movwf    TIME

       dloop  movf     TMR0,0           ;subtracts
                                        ;124 from TMR0,
                                        ;if the #
              subwf    TIME,0           ;is positive stay
                                        ;in the loop if
                                        ;the # is not
              btfsc    STATUS,CARRY ;then
                                        ;return
              goto     dloop
              return
```

(Text continued on the following page)



Bit rate 19200
8 data bits = one character

Fig. 4. Asynchronous Serial Transmission

NEWCHORD.ASM, found in Appendix 2, receives data from the five switches on the chorded keyboard to be read from Port-B's 0-4 pins of the PIC 16C84 Microcontroller. This input will then be translated onto the American Standard Code for Information Interchange (ASCII) character and will be transmitted to the terminal through the MAX 232. An actual character is formed when the buttons are pressed. The following is a description and diagram of NEWCHORD.ASM

1. Port-A is set to input, Port-A,3 is set serial out to idle and Port-B[0-4] are set to input and Port-B[5-7] are set to output.
2. Main loop consists of five main routines: stable, press, a call to table, putc, and release. The following is a diagram of the main loop.

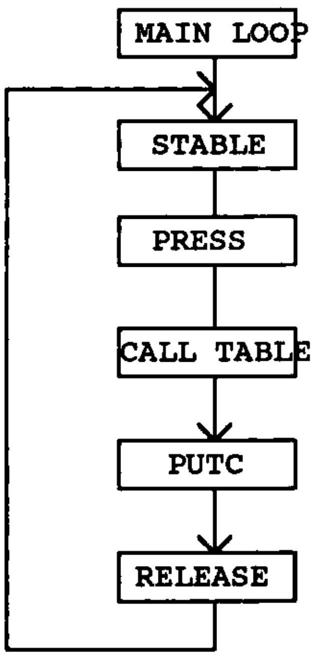


Fig. 5. Newchord.asm

3. Stable routine loops as long as receives all 1's: [11111] from Port-B. [11111] Represents no keys have been pressed. Hence stable will loop until a key is pressed.

```
;the following is actual code from stable routine
      stable movlw   .31
             xorwf   Port_B,0  ;see if
                               ;all buttons release
             btfsc   STATUS,Z  ;leave if button pressed
             goto   stable
             return
```

4. When the keypress is verified stable returns to main loop then the press routine is executed.
5. The press routine sets the COUNTHI register to .08, the COUNT register to 200 and stores the new string in the CHORD register.
   a) Performs delay for 20ms and moves the string new found in Port-B into the working memory.
   b) A test is executed to see if the original string in CHORD is the same as the string in working memory. If the test is false return to press, if true, COUNT is decremented and delay is called again.
   c) When COUNT is zero COUNTHI is decremented and press2 is called. When COUNTHI is zero the routine is exited and returned to the main loop.

```
;the following is actual code of press subroutine
      press  movf    Port_B,0
             movwf   CHORD
             movlw   .08        ;keys are held
                                ;solid for 1/12
                                ;sec
             movwf   COUNT_HI
      press2 movlw   .200
             movwf   COUNT
```

```
press1 movlw   .230
       call    delay

prloop movf    Port_B,0
       xorwf   CHORD,0
       btfss   STATUS,Z   ;test if
                          ;string changed
       goto    press      ;test false
       decfsz  COUNT,1    ;test true, decr if equal
       goto    press1
       decfsz  COUNT_HI,1;decr if
                          ;equal
       goto    press2
       return
```

6. The next routine is putc. This routine transmits the character through the serial line via the MAX 232

7. The last routine, release, is the opposite of the stable routine. It waits until all buttons have been released. Release returns to main loop when all buttons have been released and the main look is then repeated..

```
; the following is actual code of release routine
      release    movlw    .31
                 xorwf    Port_B,0   ;see if all buttons
                                     ;pressed
                 btfss    STATUS,Z   ;leave if all buttons released
                 goto     release    ;short delay to stabilize
                 call     wait
                 return
```

(Text continued on the following page)

ALPHA.ASM, found in Appendix 3, receives data from the alphanumeric keypad through Port-B 0-7 pins. The columns of the keypad are scanned until a button is pressed. The scanning stops and the column is noted. A delay of 1/12 sec is executed to make sure that a button was pressed. The particular row that the character is located in is found and the ASCII equivalent is then outputted to the screen. The program is executed as follows:

1. Port-A is set to input, Port-A,3 is set serial out to idle and Port-B[0-7] are set to input.
2. Main loop consists of five main routines: stable, press, getrow, a call to table putc, and release. The following diagram is constructed from the main loop.

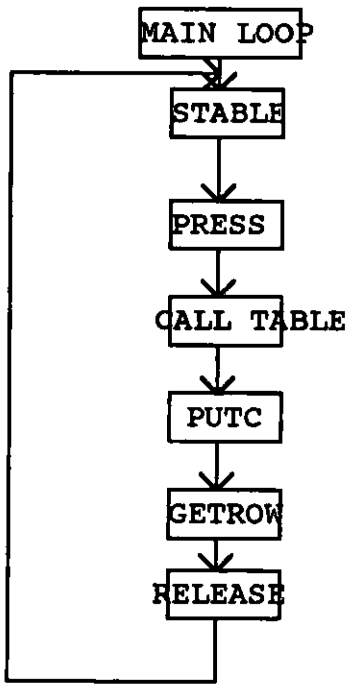


Fig. 6. Alpha.asm

3. Stable routine sets Port-A and scans the columns.

   When the stable string verified it returns to main loop and press routine is executed.

4. Press routine waits 1/12 second to make sure a button was pressed.

   This routine also takes care of debouncing.

   Getrow finds the bit position and stores it

```
;the following is actual code from getrow routine
       getrow clrf     ROW
              bcf     STATUS,CARRY    ;Make sure a 0 is detected.
       grloop rrf      PAD,1          ;rotates the its in PAD
             btfss   STATUS,CARRY     ;tests
                                      ;to see if the
                                      ;bit is zero
              return
              incf     ROW
              goto     grloop
```

5. The next routine is putc. This routine displays the character on the screen and returns to the main loop.

6. The last routine, release, is the opposite of the stable routine. It gets the new stable string and stores it. The main loop is then repeated.

CHAPTER 3

KEYBOARD LAYOUT DESIGN

The layout of an input device is guided by the importance, frequency-of-use, function, and sequence-of-use of the individual components. In (Hanes 1975, 191) it is stated that the following fundamental notions to provide a more detail set of guidelines for keyboard layout:

1. Determine the characters and numbers of keys required.
2. Arrange the keys according to their frequency of use and user characteristics. The most frequently used keys should be assigned to the stronger fingers. To enhance keying speed, the keys should also be arranged to maximize alternation of key presses between hands.
3. Follow historical precedent.
4. Follow established standards.
5. Group frequently used keys under the resting position of the hand where the user can determine their locations by touch.
6. Group related functions together.
7. Group logically and according to sequence of use.
8. Locate according to importance.
9. Code the keys so that the user can easily locate important or frequently used keys and key groups. In addition to key labels, keys can be coded by variations of shape, color, surface texture,

and space.

10. Consider all factors, including the intended applications, costs, and manufacturing requirements.

The two main reasons for the particular selection of keyboard layouts for this study are as follows:

1. The layout is constrained by the physical dimension of the unit. This due to the proposed usage in portable computing equipment.
2. The need to discount the familiarity of the average conventional keyboard was addressed by comparing the alphanumeric keyboard to the chorded keyboard.

The terminal displays a character after a "chord" has been pressed unlike other input devices that output a character only after the switches have been released. This is important because it provides feedback to the user while the switches are pressed. To determine which "chord" produces a character after being pressed, the following should be taken into account:

1. The relative frequency of the various letters in the most widely-used languages.
2. The sequences of these letters in the most frequently-used word-stems of these languages.

In order for the chorded keyboard to take these suggestions and requirements into account as well as become an user-friendly device, a survey on placement of switches was performed. Eight college students were asked to place their hand in the most comfortable position around the base (project box) of the chorded keyboard. Placement of the four finger switches and one thumb switch as well as the hand placement of the chorded keyboard itself were noted. Results of the survey are as follows:

TABLE 1

SURVEY ON SWITCH PLACEMENT

Name	Placement of keyboard and switches in hand
Subject 1	Thumb button at bottom, finger buttons on top of box.
Subject 2	Thumb button on side, finger buttons on top - 90° angle.
Subject 3	Thumb button on side, finger buttons on top - 90° angle.
Subject 4	Thumb button on side, finger buttons on top.
Subject 5	Thumb button on side, finger buttons on top and held like a gun.
Subject 6	Thumb button on side, finger buttons on top held like a gun.
Subject 7	Thumb button on side, Finger buttons on top, placement of chorded keyboard was resting on table top. Hand rested over top of chorded keyboard.

Conclusions from the survey are as follows:

The four finger switches were placed on the left side, vertically and the thumb switch was placed on the top horizontally. As for the placement of the input device itself, the chorded keyboard proved most comfortable while it was placed in the right hand with the four finger buttons placed away from the body. Another position that fared well with the subjects, was to rest the chorded keyboard on the table with the hand "curled" around the device, similar to the placement of the hand on a mouse. The current prototype of the chorded keyboard also permits the user to type with either the right or the left hand.

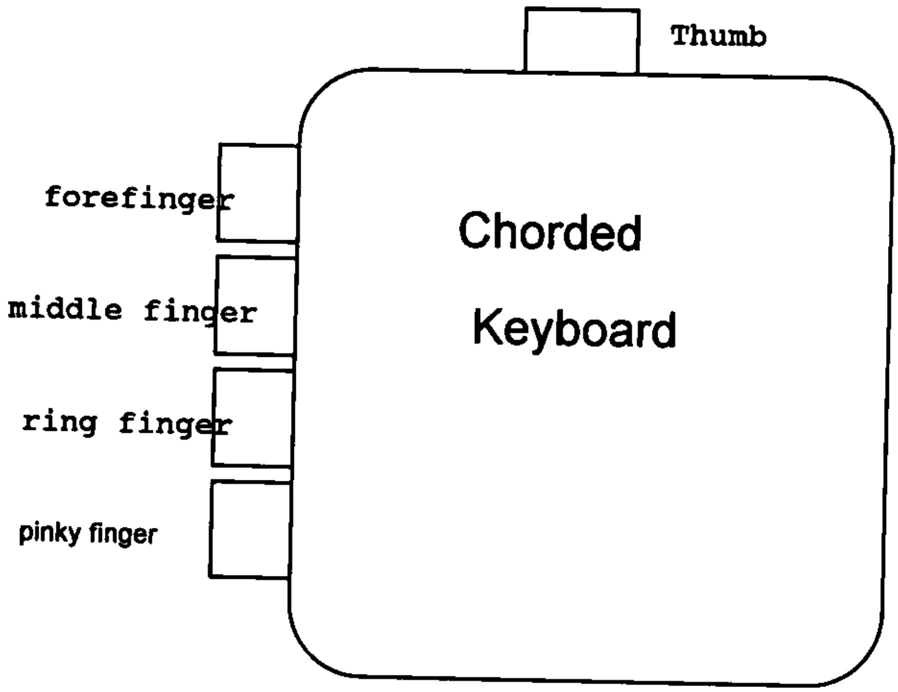


Fig. 7. Chorded Keyboard

The logic keying chart of the following chord keyboard is more extensive than the prototype which is being implemented for this particular study. Toggling provides a greater number of characters to be produced which can be used for chording special characters, numbers, and additional functionality. The following chart was used as a major reference in producing the prototype designed for this study. This mapping was designed to take into account the frequency of use of letters.

TABLE 2

CHORDIC KEYS

Character	Thumb	Forefinger	Middle finger	Ring finger	Pinky finger	Character	Thumb	Forefinger	Middle finger	Ring finger	Pinky finger
a	1	0	0	0	0	q	1	0	1	1	1
b	1	1	1	0	1	r	0	1	1	1	0
c	1	1	0	0	1	s	0	1	1	0	0
d	0	0	1	1	1	t	0	0	0	1	0
e	0	0	1	0	0	u	0	1	1	1	1
f	0	1	0	0	1	v	1	0	1	0	1
g	0	1	0	1	1	w	1	1	0	1	1
h	0	0	1	1	0	x	(2*q) (2 * 1 0 1 1 1)				
i	1	1	0	0	0	y	1	0	1	0	0
j	1	0	1	1	0	z	(2*j) (2 * 1 0 1 1 0)				
k	0	0	1	0	1	punctuation	1	0	0	1	1
l	1	1	1	0	0	esc	1	1	1	0	0
m	0	1	1	0	1	space	0	1	0	0	0
n	0	0	0	1	1		1	0	0	1	0
o	0	0	0	0	1	CR-LF	0	1	0	1	0
p	1	1	0	1	0	,	1	0	0	0	1

Numbers Mode (Toggled with ESC-n)

0	1	0	0	0	0	a	0	1	0	1	0
1	0	0	0	0	1	b	0	1	0	1	1
2	0	0	0	1	0	c	0	1	1	0	0
3	0	0	0	1	1	d	0	1	1	0	1
4	0	0	1	0	0	e	0	1	1	1	0
5	0	0	1	0	1	f	0	1	1	1	1
6	0	0	1	1	0						
7	0	0	1	1	1						
8	0	1	0	0	0						
9	0	1	0	0	1						

TABLE 3

ESCAPE CODES

ESC - b	Backspace destructive/non-destructive toggle.
ESC - i	Initialize (Clear) current page.
ESC - j	Jump to MONITOR-51.
ESC - 1#	Move line # to current window.
ESC - n	Numbers mode toggle.
ESC - o	Overnight/push-right toggle.
ESC - s	Shift - toggle upper/lower case.
ESC - t	Toggle serial port on or off (to save 10ma)
ESC - u	Upload current line to serial port.
ESC - w	Toggle windows (leave current line displayed).
ESC - y	Move cursor to beginning of current line.
ESC - .	Move cursor to end of current line.

The following table gives the actual keying logic of the prototype used for the experiment. This prototype attempts to map the most frequent used characters to the easiest chords.

TABLE 4

KEYING LOGIC OF CHORDED KEYS

Character	Thumb	Forefinger	Middle finger	Ring finger	Pinky finger	Character	Thumb	Forefinger	Middle finger	Ring finger	Pinky
A	0	1	0	0	0	Q	1	0	1	1	1
B	1	1	1	0	1	R	0	1	1	1	0
C	1	1	0	0	1	S	0	1	1	0	0
D	0	0	1	1	1	T	0	0	0	1	0
E	0	0	1	0	0	U	0	1	1	1	1
F	0	1	0	0	1	V	1	0	1	0	1
G	0	1	0	1	1	W	1	1	0	1	1
H	0	0	1	1	0	X	1	1	1	1	0
I	1	1	0	0	0	Y	0	1	0	1	0
J	1	0	1	1	0	Z	1	1	1	1	1
K	0	0	1	0	1	return	1	0	0	0	1
L	1	1	1	0	0	backspace	1	0	0	1	1
M	0	1	1	0	1	space	1	0	0	0	0
N	0	0	0	1	1	period	1	0	0	1	0
O	0	0	0	0	1	line feed	1	0	1	0	0
P	1	1	0	1	0						

An empirical experiment performed on the output timing of a character after a chord was pressed was conducted for the following reasons:

1. To handle to problem of debouncing
2. To address the non simultaneous pressing of switches to produce a character

Conclusions from the experiment returned a delay of 1/12 sec to give a good balance between making sure the right characters were being transmitted and handling the debouncing condition.

CHAPTER 4

USABILITY EXPERIMENTAL DESIGN

The usability experiment to compares and contrasts the input rates of the chorded keyboard versus the alphanumeric keypad. We must also take into consideration the required training for the use of chorded keyboards. Hence, some type of training on the chorded keyboard must take place. The actual experiment consisted of testing the responses of both keyboards performing similar tasks in the same environment. The data was gathered and the results displayed. Eight college students were used in the experiment. The eight students were divided into two groups, with four students being placed in the chorded keyboards group and the other four students were placed in the alphanumeric keypad group.

Experimental Design

The experimental design used for the study could be said to fit a repeated measures design. The study examined the performance measures of the chorded keyboard and the alphanumeric keypad on portable computing devices. Both groups were exposed to training, drills and tests that examined their accuracy and speed performances as well as immediate feedback to their performances. The independent variables included the chorded keyboard and alphanumeric keypads and the dependent variable, or effect to be studied, assessed the performances of both groups.

The Repeated Measures Design takes the form of:

$$R \quad O_1 \; X \quad O_2$$
$$R \quad O_1 \; X \quad O_2$$

Fig. 8. Repeated Measures Design

The *X* represents the exposure of a group to an experimental variable or event, the effects of which are to measured; *O* will refer to some process of observation or measurement; the symbol R indicates random assignment to separate treatment groups, (Campbell and Stanley 1967, 6). In this research the *X* variable is the chorded keyboard group and the alphanumeric keypad group. O_1 represents the exact same training performed by both groups. O_2 represents the final text paragraphs that both groups performed after the training period.

Apparatus

Stimuli were generated and responses collected using either the chorded keyboard or the alphanumeric keypad input device. A block diagram of both experimental apparatus are shown in Figure 2.1 and Figure 2.2. The chorded keyboard is a hand-held device that fits in the palm of either hand. Five keys or switches are mounted on the side and top surfaces and placed at the positions occupied by the fingertips with the fingers slightly curled. The pressed switches produce chords and the appropriate character is then displayed on the terminal. The alphanumeric keypad is also a hand-held device that sits on a flat surface while the a finger on the other hand presses a particular button to transmit a character to the terminal.

Subjects

Eight college students were asked to participate in the study. These individuals were volunteers and were not compensated for their participation. Age ranged from 19 to 29. There were 4 students in the chorded keyboard group and 4 students in the alphanumeric keypad group. The majority of users were defined as 'casual' users, that is, they generally access the conventional "QWERTY" keyboard on a infrequent, or casual, basis.

Procedure

The usability experiments were divided into two stages: Stage I was primarily responsible for the training of both groups. The subjects received training for a period of 60 minutes daily, for 5 days. During stage I the subjects were instructed in the basics of keying and chord formation while they learned the chord alphabet. Training procedures inevitably differed for the two groups. The alphanumeric keypad training procedures, which were very minimal, are found in Appendix 6. The chorded keyboard's training procedures are found in Appendix 5. In both cases the objective was to teach all keys and functions as quickly as possible, so that both groups could be practiced and tested on identical material at the earliest time possible.

Stage I

During Stage I the subjects were asked to give background information and identify their degree of familiarity with computers, calculators, typewriters and musical instruments. A sample of the Information Sheet is found in Appendix 6. The actual data is found in Table 5. In addition, subjects were asked to complete a questionnaire that required them to give an initial overall rating of which input device they believed would perform best. A sample of the Initial Questionnaire is found in Appendix 7. The actual data is found in Table 6. The subjects were given a choice of three charts to learn the chords of the keys: Alphabetical ordering of letters, the conventional "QWERTY" keyboard ordering of letters, or the ordering of the most-frequently used letters. Each of these charts are found in Appendices 8, 9, and 10 respectively. The training for each device included an introduction to new chords, entitled New Letter Drills. A sample of the New Letter Drills is found in Appendix 11. This gave the subjects time to familiarize themselves with the device and drills on text to improve basic typing speed. Another exercise, Drill Mode Exercise, gave the subjects the opportunity to improve on their performance measures. A sample of the Drill Mode Exercise is found in Appendix 12.

Stage II

During Stage II, the students were given two final text drills. One a large paragraph and the other a small paragraph. The same text was used throughout and was adapted from a creative timed writings text on Black Inventors (Axelrod 1975, 11) and is found in Appendicies 13 and 14. It is approximately 395 characters long and was presented as double-spaced, typewritten copy, printed in all uppercase characters. The instructor stressed speed (e.g. time to complete a particular task) without hindrance to accuracy (keypress errors). Subjects typed as much of the text as they could within the time allowed to them. After all the trials had been completed, they were asked for their impressions and preferences.

Subjects were also asked to complete a second questionnaire on keyboard logic and layout. Again, the subject was given the opportunity to voice their opinion about the design, development, and training of the input device. All of their comments and suggestions were considered and well documented.

Testing

The atmosphere of formal tests were minimized, therefore most practice took the form of timed trials from the beginning. During each session, at least 6 such timed trials were given. In the early stages these were of 2-5 minutes' duration. At the discretion of the instructor, the duration was increased until trials were regularly of 10 minutes' duration. The subjects were also informed at the end of each training session to "play" with the different characters learned. During each session the instructor gave the opportunity for the subjects to voice their opinion about the design, development, and of the input device. The subjects comments and suggestions were considered and well documented. Test material was designed to represent as closely as possible the types of code in practical use. Test and practice material accordingly used double spaced typed passages from different forms of text. Drill mode exercises presented words to familiarize the subjects with the new chords.

CHAPTER 5

RESULTS AND DISCUSSION

Each subject participated in the five day training session, which consisted of instruction, practice, and data collection. No subject had prior experience with one-handed alphanumeric keypads or chorded keyboards of the type studied. The subject used either the keyboard or keypad and was familiarized with the keying logic and keyboard layout. After familiarization, subjects trained to an accuracy and speed criteria. Accuracy was stressed over speed, and the subjects were instructed to correct errors. Each subject also participated in a test drill testing session, where they typed as much of the text as they could within the time allowed to them, where accuracy was still stressed over speed.

Stage I Results

Subjects listed their general information on the Background Information sheet found in Appendix 7. A summary of this information is found in the following Table 5. The questionnaire found in Appendix 8, given to all subjects at the beginning of training showed the subject's user preference ranking between the two input devices. As shown in Table 6, the chorded keyboard was initially preferred to perform faster than the alphanumeric keypad. For the most accurate input device, the subjects unanimously chose the alphanumeric keypad.

TABLE 5

SUBJECTS BACKGROUND INFORMATION

SUBJECT	Major	Age	Sex	QWERTY Typing Speed	Right /Left Handed	Musical Instrument
Chord 1	Computer Science	22	M	55 WPM	Right	Flute
Chord 2	Math	19	M	25 WPM	Right	NO
Chord 3	Chemistry	22	F	45 WPM	Right	Bass Clarinet
Chord 4	Computer Science	19	M	40 WPM	Right	NO
Alpha 1	Computer Science	29	M	40 WPM	Right	NO
Alpha 2	Comm. Health Education	22	M	50 WPM	Right	Piano
Alpha 3	Comm	22	M	45 WPM	Right	NO
Alpha 4	English	20	F	60 WPM	Right	NO

TABLE 6

INITIAL USER PREFERENCE RANKING

PERFORMANCE MEASURES	FASTEST SPEED	MOST ACCURATE
Chord 1	Chord	Alpha
Chord 2	Alpha	Alpha
Chord 3	Chord	Alpha
Chord 4	Chord	Alpha
Alpha 2	Chord	Alpha
Alpha 2	Chord	Alpha
Alpha 3	Chord	Alpha
Alpha 4	Chord	Alpha

The following list states which keying logic charts were available for the subjects to choose from:

1. Alphabetical Keying Logic Chart, where the character and corresponding chord were listed in alphabetical order, found in Appendix 9.
2. QWERTY Keying Logic Chart, where the character and corresponding chord were listed according to the QWERTY keyboard layout, found in Appendix 10.
3. Most Frequently-Used Keys Keying Logic Chart, where the character and corresponding chord were listed according the most frequently used keys, found in Appendix 11.

Table 7 displays the results of the baseline test that was given to each subject on Day 1. Both groups were asked to perform all letters of the alphabet in random order without any initial training. The time in which they finished was recorded. This test was used to show the initial performance of both groups before training and the following table displays the results:

TABLE 7

INITIAL BASELINE DRILL

SUBJECT/ INPUT DEVICE	# OF CHAR PRODUCED / # OF CHAR IN ALPHABET	COMPL. TIME
Chord 1	46 / 26	4:16
Chord 2	49 / 26	3:41
Chord 3	44 / 26	2:00
Chord 4	33 / 26	2:38
Alpha 1	26	1:03
Alpha 2	25	1:08
Alpha 3	22	:35
Alpha 4	26	1:30

During Stage I, all subjects were both groups received training on their respective keyboards, while performance measures and performance ranking was recorded. Speed and accuracy were documented as performance measures. According to the following Table 8, the first day of training the chorded keyboard group averaged 11 WPM. The alphanumeric keypad group averaged 8 WPM.

TABLE 8

SUBJECT TRAINING PROGRESS

SUBJECT/ INPUT DEVICE	DAY 1 NEW LETTER DRILLS PRT 1/ PRT 2	DAY 1 DRILL MODE PRT 1/ PRT 2	DAY 1 MEAN WPM
Chord 1	13 WPM / 7 WPM	18 WPM / 9 WPM	12 WPM
Chord 2	19 WPM / 14 WPM	12 WPM / 11 WPM	14 WPM
Chord 3	19 WPM / 13 WPM	12 WPM / 7 WPM	10 WPM
Chord 4	16 WPM / 10 WPM	13 WPM / 7 WPM	12 WPM
Alpha 1	7 WPM / 6 WPM	6 WPM / 5 WPM	6 WPM
Alpha 2	7 WPM / 8 WPM	5 WPM / 6 WPM	7 WPM
Alpha 3	13 WPM / 12 WPM	12 WPM / 11 WPM	12 WPM
Alpha 4	8 WPM / 8 WPM	7 WPM / 8 WPM	8 WPM

The chorded keyboard group dramatically increased their performance as the training continued. This is shown by comparing the Day 1 Subject Training Progress found in Table 8 with the Day 3 Subject Training Progress, which is found in the following Table 9. The alphanumeric keypad group averaged 8 WPM. The chorded keyboard group also averaged 8 WPM on the third day of training. These results showed that the training period was effective in providing time for the subjects to gain experience using the input devices.

TABLE 9

SUBJECT TRAINING PROGRESS (CONT)

SUBJECT/ INPUT DEVICE	DAY 3 NEW LETTER DRILLS PRT 1/ PRT 2	DAY 3 DRILL MODE PRT 1/ PRT 2	DAY 3 MEAN WPM
Chord 1	6 WPM / 10 WPM	5 WPM / 6 WPM	7 WPM
Chord 2	7 WPM / 7 WPM	7 WPM / 8 WPM	7 WPM
Chord 3	6 WPM / 11 WPM	7 WPM / 9 WPM	8 WPM
Chord 4	9 WPM / 11 WPM	7 WPM / 7 WPM	9 WPM
Alpha 1	5 WPM / 6 WPM	5 WPM / 6 WPM	6 WPM
Alpha 2	6 WPM / 7 WPM	5 WPM / 4 WPM	6 WPM
Alpha 3	12 WPM / 10 WPM	12 WPM / 11 WPM	11 WPM
Alpha 4	8 WPM / 9 WPM	9 WPM / 7 WPM	8 WPM

For the first day, subjects working on the chorded keyboard felt that if the chorded keyboard was tilted one could get a better "handle" of the input device. Two subjects who were intermediate typists, agreed that the timing for the actual character to be inputted to the screen was too long. The subjects were used to the slight touch of the button from the conventional keyboard to produce a letter. The chorded keyboard is based on the press and hold of the chord for 1/12 of a second. Subjects in the chorded keyboard group also noted at first that it was tedious having to look at the chart and then at the paper to produce a letter. The instructor advised them that the following situation was only occurring because of the lack of knowledge of the chords. The issue of fatigue was a great one with the subjects of the chorded keyboard. The instructor informed the subjects that they were using muscles that had not been exercised before in a capacity such as this.

Subjects working on the alphanumeric keypad for the first day, stressed that there was no comfortable position to hold the input device, thus making if difficult to use. They also commented on the consistent pressure needed to produce the desired output. One subject noted that the size of the keypad was too large and awkward for a portable computing device, while another subject stated that the size of his fingers restricted him from producing the correct character. Yet another subject commented that he felt at ease with the keypad. As for their performance, the trainees produced characters at the same or even slower rate than their counterparts on the chorded keyboard.

The second day of training all subjects in the chorded keyboard group had a problem outputting the letters M and B. The velcro strap had been added and all subjects responded positively towards the assistance of hold the chorded keyboard in position. Unfortunately, there was no significant increase of speed from Day 1, which is understood since it takes some time for the corresponding chords of the letters to be mastered. The errors by all subjects were kept to a minimum during this session. For the chorded keyboard the issue of fatigue on the arm, finger and thumb was also addressed.

The third and last day of training, subjects in the chorded keyboard group stated that they were tired, due to other events. However, all of the subjects performance on the sentence drills were higher than the other drills performed. All of the subjects also engaged in a competition to boost morale in which each subject drastically improved their performance. Again the subject commented on the pressure that is needed to produce a character. Chord 1 subject noted that the characters just learned were easier than the previous session the day before and commented that the more frequently used characters should be changed to easier chords. The errors produced during the training period were well documented but were not used in the data for computing the tables and figures.

Subjects in the Alphanumeric Keypad group, again unanimously agreed that the input device should not be used in a portable computing device. One subject stated the device was too small for finger to output the desired character. The subjects stated that the design of the keypad did not conform to the contour of the hand. One subject explained that it was hard to produce an output because of the pressure needed to press the button.

Stage II Results

The results of the Large Paragraph Final Drill and Small Paragraph Final Drill that were given to both groups are displayed in Table 10 and Table 11, respectively. For the Large paragraph, the chorded keyboard averaged 9 WPM and the alphanumeric keypad averaged 8 WPM. The results of the Large Paragraph Drill found in Table 10 are as follows:

TABLE 10

FINAL TEXT DRILL PERFORMANCE
LARGE PARAGRAPH

SUBJECT NUMBER /INPUT DEVICE	# OF COMPL. WORDS	GROSS WORDS	COMPL TIME	# OF ERRORS	NET WORDS G-E(10)=N
Chord 1	1103	11 WPM	20:28	5 10 WPM	10 WPM
Chord 2	1103	12 WPM	18:11	10 11 WPM	11 WPM
Chord 3	1103	10 WPM	22:18	20 8 WPM	8 WPM
Chord 4	1103	7 WPM	25:33	14 5 WPM	5 WPM
Alpha 1	1103	7 WPM	32:35	10 6 WPM	6 WPM
Alpha 2	1103	7 WPM	25:32	10 6 WPM	6 WPM
Alpha 3	1103	10 WPM	22:45	0 10 WPM	10 WPM
Alpha 4	1103	10 WPM	20:32	5 6 WPM	10 WPM

AVERAGE GROSS WORDS FOR CHORDED KEYBOARD = 9 WPM
AVERAGE NET WORDS FOR CHORDED KEYBOARD = 9 WPM

AVERAGE GROSS WORDS FOR ALPHANUMERIC KEYPAD = 8 WPM
AVERAGE NET WORDS FOR ALPHANUMERIC KEYPAD = 8 WPM

AVE COMPLETION TIME FOR CHORDED KEYBOARD = 21:23
AVE COMPLETION TIME FOR ALPHANUMERIC KEYPAD = 25:45

The results of the Small Paragraph Drill found in Table 11, are as follows:

TABLE 11

FINAL TEXT DRILL PERFORMANCE
SMALL PARAGRAPH

SUBJECT NUMBER/ INPUT DEVICE	# OF COMPL. WORDS	GROSS WORDS	COMPL. TIME	# OF ERRORS	NET WORDS G-E(10)=G
Chord 1	82	10 WPM	8:21	3	9 WPM
Chord 2	82	11 WPM	7:19	5	10 WPM
Chord 3	82	8 WPM	11:34	12	6 WPM
Chord 4	82	8 WPM	11:10	11	6 WPM
Alpha 1	82	7 WPM	12:32	0	7 WPM
Alpha 2	82	8 WPM	11:56	7	7 WPM
Alpha 3	82	7 WPM	11:05	2	7 WPM
Alpha 4	82	7 WPM	13:52	10	6 WPM

AVERAGE NET WORDS FOR CHORDED KEYBOARD = 8 WPM
AVERAGE GROSS WORDS FOR CHORDED KEYBOARD = 7 WPM

AVERAGE NET WORDS FOR ALPHANUMERIC KEYPAD = 7 WPM
AVERAGE GROSS WORDS FOR ALPHANUMERIC KEYPAD = 7 WPM

AVE COMPLETION TIME FOR CHORDED KEYBOARD = 9:43
AVE COMPLETION TIME FOR ALPHANUMERIC KEYPAD = 12:45

The Chorded Keyboard group average net words were greater than the Alphanumeric Keypad group average net words, for both the Large Paragraph Drill and the Small Paragraph Drill. It should also be noted that the average completion time for the Chorded Keyboard group is much faster that the average completion time for the Alphanumeric Keypad group in both cases as well. This shows that the Chorded Keyboard group produced more characters in a smaller amount of time. Hence the Chorded Keyboard, under this particular study, is a faster input device than the Alphanumeric Keypad.

At the end of the session the subjects were given the same questionnaire in which they recorded their preferences for fastest speed and most accurate input device. The following Table 12 shows the subjects final preference considering the two performance measures.

TABLE 12

FINAL USER PREFERENCE RANKING

PERFORMANCE MEASURES	FASTEST SPEED	MOST ACCURATE
Chord 1	Chord	Alpha
Chord 2	Chord	Alpha
Chord 3	Chord	Alpha
Chord 4	Chord	Alpha
Alpha 1	Chord	Alpha
Alpha 2	Chord	Alpha
Alpha 3	Chord	Alpha
Alpha 4	Chord	Alpha

As shown all subjects agreed in the Final User Preference Ranking. All eight preferred the Chorded Keyboard for fastest speed and all preferred the Alphanumeric Keyboard for best accuracy.

The following Table 13 gives an overall account of the mean number of tasks performed by each group as well as a summary of the subjects final preferences.

TABLE 13

PERFORMANCE MEASURES FOR TWO KEYBOARDS

INPUT DEVICES	MEAN NUMBER OF TASKS PERFORMED	ACCURACY FINAL PREFERENCE	SPEED FINAL PREFERENCE
Chord	27	3	8
Alpha	14	8	3

2 Subjects on the Alphanumeric Keypad Group
1 on the Chorded Keyboard group Chorded Keyboard Most Accurate
1 Subject on the Alphanumeric Keypad Group
2 on the Chorded Keyboard group Alphanumeric Keypad Faster

The following Table 14 shows the subjects overall performance ranking.

APPENDIX 14

PERFORMANCE RANKING

INPUT DEVICE	MEAN PREFERENCE RANKING
Chord	fastest speed
Alpha	first in accuracy

This study displayed that chorded keyboard offers an advantage over the alphanumeric keyboard in its one-handed operation, its small size, its low cost on portable computing devices. The usability experiment with both input devices concluded the following:

1. Subject exposed to the Chorded Keyboard experienced difficulties in initial training.
2. Subjects in the Chorded Keyboard Group average speed after training in higher than the Alphanumeric Keypad Group.
3. Accuracy for Alphanumeric Keypad is attributed by visual indicators.

CHAPTER 6

CONCLUSION

This study demonstrated that having a chorded keyboard allows us to build more compact portable computers with faster input rates than the present portable computing input devices. The chorded keyboard was designed with three aims portable computing devices in mind: to furnish a comfortable input device useful for portable computing devices, to provide a compact size for the computing device, and to operate the input device with one hand only. As shown in the design, the layout of the input device is guided by the importance, frequent-of-use, function, and sequence-of-use of the individual components. The following factors give support the research problem of this study.

1. Chorded keyboards cost less to manufacture. This is because there is no need for the amount of parts that are required to manufacture the conventional keyboards or mini-keyboards that portable computing devices possess at the present time.

2. Chorded keyboards permit smaller portable computing devices to be manufactured. The design of the chorded keyboard described has drastically reduced keyboard size while expanding the number of available functions. Although this particular experiment only dealt with 31 chords, functionality can be improved with the use of toggling. This reduction has

been achieved by cutting down the physical number of keys and the actual size of the input device itself on the portable computing device The consistent placement of the fingers and thumb on the switches of the chorded keyboard permits continuous, high-speed data entry and has a distinct advantage over the miniature buttons used on present devices. The keyboard's small size also permits a portable computing device to have more performance within the same size space.

3. The chorded keyboard provides a user with the capability to input data with one hand. This is very beneficial for a number of operations in which an user communicating with a computer requires a hand free to perform other operations. An operator may wish to communicate instructions to a system while keeping one hand on the machine controls. A user may wish to perform another task, such as turning pages of a book or leaf through papers while entering data into a computer. An operator of a vehicle would be able to use the chord keyboard for data entry while controlling the vehicle with the other hand. Data can even be entered with the chorded keyboard with conditions of complete darkness. Chorded keyboards will assist the use of portable computing devices in this capacity.

4. Handling large alphabets for handicapped people can be compensated with the chorded keyboard. One-handed people can use the chord keyboard to compensate for this disability. Blind people could read braille with one hand and type with the other. Mute people might use the keyboard as an input device to speech-synthesis device (Bequaert et al., 1978, 62).

1. The 'load' on the little fingers must be reduced by placing the keys that are heaviest to strike where they can be struck by the strongest fingers.
2. The shape of the keyboard must be such that the hands can take up a natural position, oblique to the body.
3. The keys must be arranged so that the groups of keys for the right and for the left hand are the inverse images of each other.
4. The keys must be arranged so that every finger, when moving from one row to another, always follows the same straight line. These lines must be parallel for the fingers of each hand.

It must be noted that past experiments related to this study were given extensively more time for both stages. Unfortunately due to time this study could not be performed for a longer amount of time. It is believed that, if given the time, performance for the chorded keyboard would drastically increase than what has been reported in this study Also in the past experiments, a very powerful automated teaching system was used to teach their chorded keyboards. This particular aspect is very important in the results of an experiment such as this and is noted because the system could not be simulated because of time allotted.

Future Work

Research with chorded keyboards as a faster input device for portable computing devices has several avenues that can be explored further. Studies that this particular research did not touch on are the following:

1. An ergonomic issue for placement of switches. This will allow the keyboard to assume a design that will fit the contour of a person's hand

just as a mouse does not.

2. The actual delay time that is used to transmit a character on the screen after the actual "chords" have been pressed can vary depending on if the subject is a novice or an expert in conventional typing or chorded typing. In this study the delay time was done empirically.
3. To train the chords of the new letters in a way such that the frequency of common phrases is accentuated, also called "grey coding". The actual training course would take into consideration characters that are frequently formed one right after the other.

APPENDIX 1

SERIAL.ASM

```
;This program will allow me to transmit data through ;asynchronous
;serial I/O.  Hardware involved are PIC
;16C84 and MAX232 chips.  A 10MHz crystal will be
;used, bit rate 19200, instruction cycle frequency
;is 2.5 MHz and the temporary register (?) is 8 bits
long.

        include "picreg.equ"

XMIT    equu    .12
TIME    equ     .13
POS     equ     .14
COUNT   equ     .15
TEMP    equ     .16


      clrf    XMIT

      movlw   17h
                                      ;set up free running TMR0 and
      tris    Port_A                  ;set RA3 to output mode
      bsf     Port_A,3
      movlw   .15
      OPTION
iloop call    getc
      call    putc
      goto    iloop

putc  movwf   XMIT
      bcf     Port_A,3                ;start bit

      movlw   .115
      call    delay

      movlw   .8                      ;loop 8 times (data bits)
      movwf   POS
ptest rrf     XMIT,1                  ;rotates the bits in XMIT
                                      ;need least significant bit
                                      ;first)
      btfsc   STATUS,CARRY            ;tests to see if the bit is
                                      zero or
      goto    pone                    ;one and puts it in Port_A,3
      bcf     Port_A,3
      goto    pnext
pone  bsf     Port_A,3
```

```
pnext movlw    .107
      call     delay
      decfsz   POS,1
      goto     ptest
      bsf       Port_A,3                  ;stop bit
      movlw    .115
      call     delay
      return

getc btfsc    Port_A,4
      goto     getc
      movlw    .53                       ;Check the middle of the next bit


      call     delay
      btfsc    Port_A,4                  ;check for start bit (0)
      goto     getc
      movlw    .8
      movwf    COUNT                     ;set number of bytes to read
                                         ;as 8

gtest movlw    .107
      call     delay
      btfsc    Port_A,4                  ;read in 8 bits of data
      goto     gone                      ;if Port_A is a '1' then put
                                         ;'1' in XMIT
      bcf      STATUS,CARRY              ;if Port_A is a '0' then put
                                         ;'0' in XMIT
      goto     gnext
gone  bsf      STATUS,CARRY

gnext rrf      XMIT,1
      decfsz   COUNT,1
      goto     gtest

      movlw    .107
      call     delay
      movf     XMIT,0
      return

delay
      clrf      TMR0
      movwf     TIME

dloop movf     TMR0,0           ;subtracts 124 (now 117) from
                                ;TMR0, if the  #
      subwf    TIME,0           ;is positive stay in the loop
                                ;if the # is not
      btfsc    STATUS,CARRY     ;then return
      goto     dloop
      return

      ;send the data
       end
```

APPENDIX 2

NEWCHORD.ASM

```
;This program will allow you to input data into Port-B, ;wait until
;Port-B has a stable press, send the character pressed, wait until Port-
;B has a stable release, and loop.

        include "picreg.equ"

XMIT    equ     .12
TIME    equ     .13
POS     equ     .14
COUNT   equ     .15
TEMP    equ     .16
STRPOS  equ     .17
RET     equ     .13
SPACE   equ     .32
BSPACE  equ     .8
LFEED   equ     .10
NULL    equ     .0
STRING  equ     .35
INDREG  equ     .0
CHORD   equ     .18
TMP     equ     .19
COUNT_HI equ    .20

      clrf    CHORD

      movlw   17h
                                  ;sets RA to input
      tris    Port_A
      movlw   .31                 ;RB0 - RB4 input, RB5
                                  ;through RB7 output
      tris    Port_B
      clrf    Port_B              ;set Port_B to input mode
                                  ;(default)
      bsf     Port_A,3            ;set serial out
                                  ;to idle
      movlw   .15
      OPTION

      movlw   .12                 ;loop 12 times
      movwf   STRPOS

loop  call    stable              ;gets stable string from
                                  ;switches
      call    press               ;stores changed character
```

```
        movf    CHORD,0
        xorlw   .31
look    call    table               ;W contains table offset value
                                    ;now has table value
        call    putc
        call    release             ;gets new stable string
        goto    loop


stable  movlw   .31
        xorwf   Port_B,0            ;see if all buttons release
        btfsc   STATUS,Z            ;leave if button pressed
        goto    stable
        return

press   movf    Port_B,0
        movwf   CHORD
        movlw   .08                 ;keys are held solid for 1/12
                                    ;sec
        movwf   COUNT_HI
press2  movlw   .200
        movwf   COUNT



press1  movlw   .230
        call    delay

prloop  movf    Port_B,0
        xorwf   CHORD,0
        btfss   STATUS,Z
        goto    press
        decfsz  COUNT,1
        goto    press1
        decfsz  COUNT_HI,1
        goto    press2
        return


release movlw   .31
        xorwf   Port_B,0            ;see if all buttons pressed
        btfss   STATUS,Z            ;leave if button released
        goto    release
        call    wait
        return

wait    movlw   .200                ;wait delay is about 20ms
        movwf   TEMP

wloop   movlw   .230                ;delay about 0.1ms
                call    delay
                decfsz  TEMP,1
                goto    wloop
                return


table
        addwf   PC      ;W = offset
        retlw   .0      ;begin table  '11111' '0'
```

```
        retlw   .79     ;begin table  '11110' 'O'
        retlw   .84     ;'11101' 'T'
        retlw   .78     ;'11100' 'N'
        retlw   .69     ;'11011'  'E'
        retlw   .75     ;'11010'  'K'
        retlw   .72     ;'11001'  'H'
        retlw   .68     ;'11000'  'D'
        retlw   .65     ;'10111'  'A'
        retlw   .70     ;'10110'  'F'
        retlw   .89     ;'10101'  'Y'
        retlw   .71     ;'10100'  'G'
        retlw   .83     ;'10011'  'S'
        retlw   .77     ;'10010'  'M'
        retlw   .82     ;'10001'  'R'
        retlw   .85     ;'10000'  'U'
        retlw   .32     ;'01111'  'SPACE'
        retlw   .13     ;'01110'  'RETURN'
        retlw   .46     ;'01101'  'PERIOD'
        retlw   .8      ;'01100'  'BACKSPACE'
        retlw   .10     ;'01011'  'LINE FEED'
        retlw   .86     ;'01010'  'V'
        retlw   .74     ;'01001'  'J'
        retlw   .81     ;'01000'  'Q'
        retlw   .73     ;'00111'  'I'
        retlw   .67     ;'00110'  'C'
        retlw   .80     ;'00101'  'P'
        retlw   .87     ;'00100'  'W'
        retlw   .76     ;'00011'  'L'
        retlw   .66     ;'00010'  'B'
        retlw   .88     ;'00001'  'X'
        retlw   .90     ;'00000'  'Z'


gets    movlw   STRING                  ; Point FSR to the beginning
                                        ;of the string
        movwf   FSR

gsloop          call    getc
        movwf   TEMP
        call    putc

        movlw   RET
        xorwf   TEMP,0                  ;test for return
        btfss   STATUS,Z                ;test flag bit to see if
                                        ;affected
        goto    strnext                 ;no return - go to next test
        movwf   INDREG                  ;return - end input, drop
                                        ;null in next
        movlw   LFEED
        call    putc
        return

strnext         movlw   LFEED
                xorwf   TEMP,0          ;test for linefeed
                btfss   STATUS,Z        ;test flag bit to see if
                                        ;affected
                goto    strntl          ;no linefeed - go to next test
                movwf   INDREG          ;linefeed - end input,
```

```
                                        ;drop null in next
          return

strntl    movlw     BSPACE              ;test for backspace
          xorwf     TEMP,0
          btfss     STATUS,Z            ;test flag bit to see if
                                        ;affected
          goto      savec               ;no backspace - go to
                                        ;next bit
          movlw     SPACE
          call      putc
          movlw     BSPACE
          call      putc
          decf      FSR,1
          goto      gsloop

savec movf   TEMP,0
          movwf     INDREG
          incf      FSR,1               ;increment file select register
          goto      gsloop


putc      movwf     XMIT
          bcf       Port_A,3            ;start bit

          movlw     .115
          call      delay

          movlw     .8                  ;loop 8 times (data bits)
          movwf     POS
ptest rrf    XMIT,1                     ;rotates the bits in XMIT

          btfsc     STATUS,CARRY             ;tests to see if the bit
                                             ;is zero or
          goto      pone                     ;one and puts it in
                                             ;Port_A,3
          bcf       Port_A,3
          goto      pnext
pone      bsf       Port_A,3

pnext movlw  .107
          call      delay
          decfsz    POS,1
          goto      ptest
          bsf       Port_A,3            ;stop bit
          movlw     .115
          call      delay
          return

getc      btfsc     Port_A,4
          goto      getc
          movlw     .53                 ;53us

          call      delay
          btfsc     Port_A,4            ;check for start bit (0)
          goto      getc
          movlw     .8
          movwf     COUNT               ;set number of bytes to
                                        ;read as 8
```

```
          gtest    movlw   .107
          call     delay
          btfsc    Port_A,4          ;read in 8 bits of data
          goto     gone              ;if Port_A is a '1' then
                                     ;put '1' in XMIT
          bcf      STATUS,CARRY      ;if Port_A is a '0'
                                     ;then put '0' in XMIT
          goto     gnext
gone      bsf      STATUS,CARRY

gnext rrf    XMIT,1
      decfsz COUNT,1
      goto   gtest

      movlw  .107
      call   delay
      movf   XMIT,0
      return

delay
      clrf    TMR0
      movwf    TIME

dloop movf    TMR0,0                 ;subtracts 124 (now 117) from
                                     ;TMR0, if the  #
          subwf    TIME,0            ;is positive stay in the loop
                                     ;if the # is not
          btfsc    STATUS,CARRY      ;then return
          goto     dloop
      return

          ;send the data
          end
```

APPENDIX 3

ALPHA.ASM

```
;This program will allow you to input data from the alphanumeric
;keyboard.  The low bits of Port-A will be used to scan and Port-B
;[RB0-RB7] will be used to read the actual data.


      include 'picreg.equu"

XMIT        equ   .12
TIME        equ   .13
POS         equ   .14
COUNT       equ   .15
TEMP        equ   .16
STRPOS      equ   .17
ROW         equ   .18
TMP         equ   .19
SCAN        equ   .20
COL         equ   .21
COUNT_HI    equ   .22
NUM         equ   .23
PAD         equ   .24


      movlw   10h             ;sets RA4 to input.  RA0-RA3: output
                              ;set up free running TMR0 and
                              ;set RA3 to output mode
      tris    Port_A
      bsf     Port_A,3        ;set serial out to idle
      movlw   .255            ;set Port-B to input mode (default)
      tris    Port_B
      clrf    Port_B

      movlw   .15
      OPTION

loop  call    stable          ;gets stable string from Port-B
                              ;scans the columns
      call    press           ;stores the column that has been pressed
      call    getrow          ;store the bit position
      btfsc   ROW,3           ; See if error.
      goto    loop            ; Continue if bit 3 not set.
      call    table           ;W contains table offset value
                              ;now has table value
      call    putc
      call    release
      goto    loop
```

```
rescol  clrf    COL
stable  movf    COL,0           ;store string in COL reg
        andlw   .7              ;clear all but the lowest 3 bits
        addlw   .8              ;be sure to make 232D output idle
        movwf   Port_A          ;write to Port-A
        movlw   .200
        call    delay           ;wait about .1ms
        movlw   .255
        xorwf   Port_B,0        ;see if all buttons released
        btfss   STATUS,Z        ;goto next column if no press
        return                  ;leave if button pressed
        incf    COL,1
        movlw   .5
        subwf   COL,0
        btfsc   STATUS,Z
        goto    rescol
        goto    stable

press   movf    Port_B,0
        movwf   PAD
        movlw   .08             ;keys are held solid for 1/12 sec
        movwf   COUNT_HI
press2  movlw   .200
        movwf   COUNT

press1  movlw   .230
        call    delay

prloop  movf    Port_B,0
        xorwf   PAD,0
        btfss   STATUS,Z
        goto    press
        decfsz  COUNT,1
        goto    press1
        decfsz  COUNT_HI,1
        goto    press2
        return

getrow  clrf    ROW
        bcf     STATUS,CARRY    ; Make sure a 0 is detected.
grloop  rrf     PAD,1           ;rotates the bits in PAD
        btfss   STATUS,CARRY    ;tests to see if the bit is zero
        return
        incf    ROW
        goto    grloop

release movlw   .255
        xorwf   Port_B,0        ;see if all buttons are pressed
        btfss   STATUS,Z        ;leave if button released
        goto    release
        call    wait
      return


wait     movlw   .200           ;wait delay is about 20ms
         movwf   TEMP

wloop    movlw   .230           ;delay about 0.1ms
```

```
        call    delay
        decfsz  TEMP,1
        goto    wloop
        return

table   bcf     STATUS,CARRY    ; Be sure that CARRY is 0
        rlf     COL,1
        rlf     COL,1
        rlf     COL,0       ; Put the column shifted 3 bits in W.
        iorwf   ROW,0       ; Merge the row in.
        addwf   PC          ;W = offset
        retlw   .65         ;'A'
        retlw   .70         ;'F'
        retlw   .75         ;'K'
        retlw   .80         ;'P'
        retlw   .55         ;'7'
        retlw   .52         ;'4'
        retlw   .49         ;'1'
        retlw   .32         ;SPACE
        retlw   .66         ;'B'
        retlw   .71         ;'G'
        retlw   .76         ;'L'
        retlw   .81         ;'Q'
        retlw   .56         ;'8'
        retlw   .53         ;'5'
        retlw   .50         ;'2'
        retlw   .8          ;BACKSPACE
        retlw   .67         ;'C'
        retlw   .72         ;'H'
        retlw   .77         ;'M'
        retlw   .82         ;'R'
        retlw   .57         ;'9'
        retlw   .54         ;'6'
        retlw   .51         ;'3'
        retlw   .46         ;PERIOD
        retlw   .68         ;'D'
        retlw   .73         ;'I'
        retlw   .78         ;'N'
        retlw   .83         ;'S'
        retlw   .85         ;'U'
        retlw   .87         ;'W'
        retlw   .89         ;'Y'
        retlw   .10         ;LINE FEED
        retlw   .69         ;'E'
        retlw   .74         ;'J'
        retlw   .79         ;'O'
        retlw   .84         ;'T'
        retlw   .86         ;'V'
        retlw   .88         ;'X'
        retlw   .90         ;'Z'
        retlw   .27         ;ESC

putc    movwf   XMIT
        bcf     Port_A,3        ;start bit
        movlw   .115
        call    delay

        movlw   .8              ;loop 8 times (data bits)
```

```
          movwf    POS
ptest     rrf      XMIT,1            ;rotates the bits in XMIT (need least
                                     ;significant bit first)
       btfsc    STATUS,CARRY    ;tests to see if the bit is zero or
       goto     pone            ;one and puts it in Port_A,3
       bcf      Port_A,3
       goto     pnext
pone   bsf      Port_A,3

pnext  movlw    .107
       call     delay
       decfsz   POS,1
       goto     ptest
       bsf      Port_A,3       ;stop bit
       movlw    .115
       call     delay
       return

delay
       clrf     TMR0
       movwf     TIME

dloop  movf     TMR0,0          ;subtracts 124 (now 117) from TMR0, if the
#
       subwf    TIME,0          ;is positive stay in the loop if the # is
not
       btfsc    STATUS,CARRY    ;then return
       goto     dloop
       return

  ;send the data
end
```

APPENDIX 4

ALPHANUMERIC KEYPAD TRAINING

The training for the alphanumeric keypad is based on standard typewriter tutors for novices. The goal for the training is to teach the subjects the alphabetical keys only within one week.

During the one week training course the instructor met with each subject for approximately 5 hours. A training session was conducted as follows:

Day One

Instructor asked subject which way he/she wished to hold the alphanumeric keypad.
Instructor gave subject baseline exam which tested subjects performance on keypad - noting speed and accuracy.
Instructor introduced the alphabetical keys SPACE, A, E, T, and O.
The instructor performs a guided keys and fingers exercise with subject
Subject is given drill mode exercise with these new letters.
Subject attempts to perform the keys and is given a lesson containing words based on these letters (drill mode exercise).

Instructor introduces alphabetical keys I, S, H, N, and RETURN.
The instructor performs a guided keys and fingers exercise with subject.
Subject is given drill mode exercise with these new letters.
Subject is given drill exercise with alphabetical keys SPACE, A, E, T, O, I, S, H, N, and RETURN. This gives the subject a chance to type real words as soon as possible (short real words lesson).
Subject is also informed to "play" with the different letters learned after each session.

Day Two

Instructor introduces alphabetical keys L, R, D, U, and LINE FEED.
The instructor performs a guided keys and fingers exercise with subject.

Subject is given drill mode exercise with these new letters.

Subject is given drill exercise with alphabetical keys SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, and LINE FEED. This gives the subject a chance to type real words as soon as possible (short real words lesson).

Instructor introduces alphabetical keys C, M, B, P, and BACKSPACE.

The instructor performs a guided keys and fingers exercise with subject.

Subject is given drill mode exercise with these new letters.

Subject is given drill exercise with alphabetical keys SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, LINE FEED, C, M, B, P, and, BACKSPACE. This gives the subject a chance to type real words as soon as possible (short real words lesson).

Subject is also informed to "play" with the different keys learned after each session.

Day Three

Instructor introduces alphabetical keys F, G, W, J, K, and PERIOD.

The instructor performs a guided keys and fingers exercise with subject.

Subject is given drill mode exercise with these new letters.

Subject is given drill exercise with alphabetical keys SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, LINE FEED, C, M, B, P, BACKSPACE, F, G, W, J K, and PERIOD. This gives the subject a chance to type real words as soon as possible (short real words lesson).

Instructor introduces alphabetical keys Q, V, X, Y, and Z .

The instructor performs a guided keys and fingers exercise with subject.

Subject is given drill mode exercise with these new letters.

Subject is given drill exercise with alphabetical keys SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, LINE FEED, C, M, B, P, BACKSPACE, F, G, W, J K, PERIOD, Q, V, X, Y and Z. This gives the subject a chance to type real words as soon as possible (short real words lesson).

Subject is also informed to "play" with the different keys learned after each session.

Day Four

Fifteen minute unstructured practice period

Instructor gives instructions on how to improve speed and accuracy.

Subject is drilled on text
The drills become more challenging.

Day Five

Fifteen minute unstructured practice period
Instructor continues operations like Day Four.
Instructor makes conclusion of Training period and records results.

APPENDIX 5

CHORDED KEYBOARD TRAINING

The training for the chorded keyboard is based on standard typewriter tutors for novices. The goal for the training is to teach the subjects the alphabetical chords within one week.

During the one week training course the instructor met with each subject for approximately 5 hours. A training session was conducted as follows:

Day One

Instructor asked subject which way he/she wished to hold the chorded keyboard.

Instructor gave subject baseline exam which tested subjects performance on chorded keyboard - noting speed and accuracy.

Instructor introduced the alphabetical chords SPACE, A, E, T, and O.

The instructor performs a guided keys and fingers exercise with subject.

Subject attempts to perform the chords and is given a lesson containing words based on these keys (drill mode exercise).

Instructor introduces alphabetical chords I, S, H, N, and RETURN.

The instructor performs a guided keys and fingers exercise with subject.

Subject is given drill mode exercise with these new chords.

Subject is given drill exercise with alphabetical chords SPACE, A, E, T, O, I, S, H, N, and RETURN. This gives the subject a chance to type real words as soon as possible (short real words lesson).

Subject is also informed to "play" with the different chords learned after each session.

Day Two

Instructor introduces alphabetical chords L, R, D, U, and LINE FEED.

The instructor performs a guided keys and fingers exercise with subject.

Subject is given drill mode exercise with these new chords.

Subject is given drill exercise with alphabetical chords SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, and LINE FEED. This gives the subject a chance to type real words as soon as possible (short real words lesson).

Instructor introduces alphabetical chords C, M, B, P, and BACKSPACE.
The instructor performs a guided keys and fingers exercise with subject.
Subject is given drill mode exercise with these new chords.
Subject is given drill exercise with alphabetical chords SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, LINE FEED, C, M, B, P, and, BACKSPACE. This gives the subject a chance to type real words as soon as possible (short real words lesson).
Subject is also informed to "play" with the different chords learned after each session.

Day Three

Instructor introduces alphabetical chords F, G, W, J, K, and PERIOD.
The instructor performs a guided keys and fingers exercise with subject.
Subject is given drill mode exercise with these new chords.
Subject is given drill exercise with alphabetical chords SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, LINE FEED, C, M, B, P, BACKSPACE, F, G, W, J K, and PERIOD. This gives the subject a chance to type real words as soon as possible (short real words lesson).

Instructor introduces alphabetical chords Q, V, X, Y, and Z .
The instructor performs a guided keys and fingers exercise with subject.
Subject is given drill mode exercise with these new chords.
Subject is given drill exercise with alphabetical chords SPACE, A, E, T, O, I, S, H, N, RETURN, L, D, R, U, LINE FEED, C, M, B, P, BACKSPACE, F, G, W, J K, PERIOD, Q, V, X, Y and Z. This gives the subject a chance to type real words as soon as possible (short real words lesson).
Subject is also informed to "play" with the different chords learned after each session.

Day Four

Fifteen minute unstructured practice period
Instructor gives instructions on how to improve speed and accuracy.
Subject is drilled on text

The drills become more challenging.

Day Five

Fifteen minute unstructured practice period

Instructor continues operations like Day Four.

Instructor makes conclusion of Training period and records results.

APPENDIX 6

INFORMATION SHEET

NAME

____

MAJOR

____

AGE ________ **SEX** ______

RIGHT HANDED OR LEFT HANDED

____

TYPING SKILL LEVEL (NOVICE, INTERMEDIATE, EXPERT)
PLAY MUSICAL INSTRUMENT

____

TYPING SPEED GOAL

____

DATE

____

SIGNATURE

____

APPENDIX 7

INITIAL QUESTIONNAIRE

SPEED FIRST PREFERENCE:____________________________

WHY:____________________________________

____

SPEED SECOND PREFERENCE:____________________________

WHY:____________________________________

____

ACCURACY FIRST PREFERENCE:____________________________

WHY:____________________________________

____

ACCURACY SECOND PREFERENCE:____________________________

WHY:____________________________________

____

MOST ERRORS FIRST PREFERENCE:____________________________

WHY:____________________________________

____

MOST ERRORS SECOND PREFERENCE:____________________________

APPENDIX 8

ALPHABETICAL KEYING LOGIC CHART

A - 01000	B - 11101	C - 11001	D - 00111	E - 00100	F - 10001
G - 10011	H - 00110	I - 11000	J - 10110	K - 00101	L - 11100
M - 01101	N - 00011	O - 00001	P - 11010	Q - 10111	R - 01110
S - 01100	T - 00010	U - 01111	V - 10101	W - 11011	X - 11110
Y - 01010	Z - 11111				

PUNCTUATION

SPACE - -10000 LINE FEED - 10100 BACKSPACE - 10010

RETURN - 10001 PERIOD - 10010

APPENDIX 9

QWERTY KEYING LOGIC CHART

Q - 10111 W - 11011 E - 00100 R - 01110 T - 00010 Y - 01010
U - 01111 I - 11000 O - 00001 P - 11010

A - 01000 S - 01100 D - 00111 F - 01001 G - 01011 H - 00110
J - 10110 K - 00101 L - 11100

Z - 11111 X - 11110 C - 11001 V - 10101 B - 11101 N - 00011
M - 01101

Punctuation

SPACE - 10000 RETURN - 10001 LINE FEED - 10100 PERIOD - 10010
BACKSPACE - 10011

APPENDIX 10

MOST FREQUENTLY USED CHARACTERS KEYING LOGIC

A - 10000 E - 00100 I - 11000 U - 01111 O - 00001

R - 01110 S - 01100 T - 00010 N - 00011 L - 11100

D - 00111 M - 01101 B - 11101 C - 11001 P - 11010

F - 01001 G - 01001 H - 00110 U - 01111 W - 11011

J - 10110 K - 00101 Q - 101111 V - 10101 X - 11110

Y - 01010 Z - 11111

Punctuation

SPACE - 01000 PERIOD - 10010 BACKSPACE - 10011 LINE FEED - 10100
RETURN - 10001

APPENDIX 11

NEW LETTER DRILLS

Day One (SPACE, A, E, T, and O) 31 WORDS

(sp)(sp)(sp)	AAA	EEE	TTT	OOO	(sp)(sp)(sp)	AAA	EEE	TTT	OOO		
TTA	AAT	TTE	EET	OOT	TOO	AAO	OAA	EOO	OEE	EAA	AEE
(sp)(sp)(sp)	AAA	EEE	TTT	OOO	(sp)(sp)(sp)	AAA	EEE	TTT	OOO		
TTA	AAT	TTE	EET	OOT	TOO	AAO	OAA	EOO	OEE	EAA	AEE

Day One (I, S, H, N, and RETURN) 31 WORDS

III	SSS	HHH	NNN	(rt)(rt)(rt)	III	SSS	HHH	NNN	(rt)(rt)(rt)		
ISS	SII	HII	IIH	SNN	NSS	INN	NII	HSS	SHH	HNN	NHH
III	SSS	HHH	NNN	(rt)(rt)(rt)	III	SSS	HHH	NNN	(rt)(rt)(rt)		
ISS	SII	HII	IIH	SNN	NSS	INN	NII	HSS	SHH	HNN	NHH

APPENDIX 12

DRILL MODE EXERCISE

DAY ONE 28 WORDS

TO	TE	TA	TEA	TOE	TAO
TEO	ETO	ATO	ATE	TOA	AT
OT	ATAE	AAEO	TTAE	OOATE	OAAT
AT	TOAE	ATOE	AATT	OOTT	OOEE
EOEO	ATAT	OTOT	AEAE	TO	OT
TO	AT	TA	AT	OT	ET

SENTENCE DRILL

TAT TAO TOE AT ATE EAT.

TAT TAO TOE AT ATE EAT.

APPENDIX 13

FINAL TEXT DRILL LARGE PARAGRAPH

A SMALL DRIP CUP TO HOLD OIL, ALONG WITH AN AUTOMATIC DEVICE TO CONTROL THE FLOW OF THE OIL, MIGHT NOT SEEM AT FIRST TO BE SUCH AN IMPORTANT INVENTION. BUT THE PRINCIPLE IT IS BASED ON IS USED FOR THE MOST ADVANCED LUBRICATING SYSTEMS IN INDUSTRIES TODAY. ELIJAH MCCOY A BLACK MAN WAS THE INVENTOR JUST ABOUT A HUNDRED YEARS AGO.

MOST OF US USE SUGAR IN ITS PRESENT WHITE FORM BUT MANY OF US KNOW THAT IN THE EIGHTEEN FORTIES A MAN NAMED NORMAN RILLIEUX DEVISED A WAY TO TURN CANE INTO A FINE WHITE PRODUCT AT ABOUT ONE HALF THE COST OF PRODUCING FINE TABLE SUGAR AT THE TIME. HIS METHOD THE VACUUM PAN PROCESS BECAME WELL KNOWN AND WAS SOON USED IN THE SUGAR INDUSTRY NOT ONLY IN THIS COUNTRY BUT IN CUBA AND MEXICO AS WELL. AT THE TIME IT WAS INVENTED HIS PROCESS WAS CALLED THE GREATEST IN AMERICAN CHEMICAL ENGINEERING HISTORY.

WITH THE AUTOMATIC SHOE LASTING MACHINE THAT JAN MATZELIGER INVENTED FINE SHOES COULD BE MADE MUCH FASTER AND MUCH MORE CHEAPLY THAN BY HAND LASTING. HIS NEW WAY

OF MAKING SHOES MADE THE TOWN OF LYNN MASSACHUSETTS THE SHOE CAPITAL OF THE WORLD.

APPENDIX 14

FINAL TEXT DRILL SMALL PARAGRAPH

LEWIS LATIMER BEGAN HIS CAREER BY MAKING THE PATENT DRAWINGS FOR THE FIRST TELEPHONE AND LATER HE BECAME CHIEF DRAFTSMAN FOR TWO LARGE FIRMS. HE STARTED TO INVENT ON HIS OWN AND MADE THE FIRST LIGHT BULB WITH A CARBON FILAMENT. HIS WORK IN LIGHTING WAS SO EXPERT THAT HE WAS CHOSEN TO TAKE ON THE GREAT TASK OF LIGHTING ALL OF NEW YORK CITY. HE WAS ALSO PLACED IN CHARGE OF THE LIGHTING SYSTEMS FOR PHILADELPHIA MONTREAL AND LONDON.

WASHINGTON CARVER MADE HIS NAME IN RESEARCH IN AGRICULTURE. HE WAS A CHEMIST WHO DEVELOPED NEW USES FOR FARM PRODUCTS. THE PRODUCTS HE DEVELOPED FROM THE PEANUT AND THE SOYBEAN HELPED THE ECONOMY OF THE SOUTH BY FREEING IT FROM TOO GREAT A DEPENDENCE ON THE COTTON CROP. CARVER DID NOT PATENT ANY OF HIS IDEAS. GOD GAVE THEM TO ME HOW CAN I SELL THEM TO SOMEONE ELSE HE IS QUOTED AS SAYING.

REFERENCE LIST

American National Standards Institute. 1982. American National Standard for Office Machines and Supplies - Alphanumeric Machines - Keyboard Arrangement. ANSI X4.23. New York.

Axelrod, M. 1978. Creative Timed Writings Academic Press, Chicago.

Bequaert, F.C. and Rochester, N., Teaching Typing on a Chord Keyboard. Poughkeepsie: IBM Laboratory Technical Report, 1977. TR 000.2918.

Biegel, R.A. 1934. An Improved Typewriter Keyboard. Human Factor, 8:280-285.

Butterbaugh, L. C. 1982. Evaluation of Alternative Alphanumeric Keying Logics. Human Factors, 24. 5: 521-533.

Campbell, D.T. and Stanley, J.C. 1967. Experimental and Quasi-Experiemental Design for Research. Chicago: Rand McNally College Publishing Company.

Conrad, R. 1967. Performance with different push button arrangements. Het PTT-Bedrijfdeel, 15: 110-113.

Devoe, D. B. 1967 Alternatives to handprinting in the manual entry of data. IEEE Transactions on Human Factors and Electronics, HFE-8: 21-32.

Dvorak, A.. 1943. There is a Better Typewriter Keyboard. National Business Education Quarterly, 11: 58-66.

Goodman, D., Dickinson, J., and Francas, M. J 1985. Human Factors Design Considerations For Public Videotex Input Devices. Behaviour and Information Technology, 4.3: 189-200.

Greenstein J. S. and Muto W. H. 1988. Introduction to Input Devices in Input Devices, ed. Sol Sherr, 128-148. New York: Academic Press, Inc.

Hanes, L. F. 1975. Human factors in international keyboard arrangement. In A Chapanis (Ed.), Ethnic Variables in Human Factors Engineering, 189-206. Baltimore: Johns Hopkins University Press.

Levy, M. 1955. An Electronic aspects of the Canadian sorting of Mail Systems. Proceedings of the National Electronic Conference, 10: 1-14.

Norman, D.A. and Fisher, D. 1982. Why Alphabetic Keyboards Are Not Easy to Use: Keyboard Layout Doesn't Much Matter. Human Factors, 24:5: 509-519.

Noyes, J. 1983. Chord keyboards. Applied Ergonomics, 14.1: 55-59.

Ratz, H.C. and Ritchie, D.K. 1961. Operator Performance on a Chord Keyboard. Journal of Applied Psychology, 45: 303-308.

Sarlanis, L. and Buckley, E.P. 1965. A Human Factors Comparison of Two Data Entry Keyboards. Paper presented at the Sixth Annual Symposium of the Professional Group on Human Factors in Electronics, IEEE.

Microchip Data Book. 1994. Microchip Technology Inc, 535.

Yamada, H. 1980. A Historical Study of Typewriters and Typing Methods: From the Position of Planning Japanese Parallels. Journal of Information Processing, 2: 175-202.