DESIGN OF AN ELECTROCARDIOGRAM MACHINE
USING A PERSONAL COMPUTER

by

Luis Parra

B.S., Mechanical Electrical Engineering
National Autonomous University of Mexico, 1996

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the
Massachusetts Institute of Technology
June, 1998

©1998 Massachusetts Institute of Technology
All rights reserved

The author hereby grants to MIT permission to reproduce and distribute publicly paper and electronic
copies of this thesis document in whole or in part, and to grant others the right to do so.

Signature of Author

Department of Electrical Engineering and Computer Science
May 22, 1998

Certified by

Stephen K. Burns
Thesis Supervisor

Accepted by

Arthur C. Smith
Chairman, Department Committee on Graduate Students
DESIGN OF AN ELECTROCARDIOGRAM MACHINE
USING A PERSONAL COMPUTER

by
Luis Parra

Submitted to the DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE in partial fulfillment of the requirements for the degree of
Master of Science

Abstract

The goal of this work is to demonstrate the feasibility of using an inexpensive personal computer as the basis of a medical instrument which can be implemented and maintained in a developing country. Personal computers have become a commodity; they are inexpensive, available and are locally supported. A computer can be an instrument. This instrument can replace a traditional instrument; it can offer equal or better performance than the traditional dedicated instrument but have the advantages of a much larger user base of the personal computer hardware, software and support. I will develop and demonstrate a clinical electrocardiogram machine. I will use a PC and Visual-Basic to develop a "Virtual Electrocardiogram Machine". This will present a waveform in real-time allowing the operator to judge the quality and placement of the electrodes in the same manner as in the traditional machine. Leads can be selected, sensitivity and chart-speed can be selected with "controls" on the virtual EKG machine. In addition, the machine will allow selection and mounting of EKG to form an electrogram, annotated with date, time, names, etc. Transmission to a remote physician (Telemedicine) is an added possibility offered by the computer as is a diagnostic analysis program run on the local machine.

Thesis Supervisor: Stephen K. Burns

Title: Senior Research Scientist, HST.
ACKNOWLEDGMENTS

First of all, I want to thank my thesis advisor, Prof. Stephen Burns for his support, guidance, help and patience through all this work.

I want to thank to the National Autonomous University of Mexico, the school of Engineering, and in particular, I would like to thank Jose L. Perez Silva and Wilfredo Martinez for their support during my undergraduate thesis work.

I want to thank Impulmex, and in particular to Mario Villafaña for his help in obtaining valuable information of the Medical Device Industry in Mexico.

I was able to come to M.I.T. because of the support CONACYT, and the Electrical Engineering and Computer Science Department at M.I.T.

I thank all my friends at M.I.T. and Boston that made my life during this time much fun.

I want to thank my Parents, Jorge and Stella, thanks to your support during this time. I want to thank my brothers and sisters Estela, Teresa, Jorge, Pablo and my Nephew Vicente.
INDEX.

Abstract. .............................................................................. 2
Acknowledgments. .............................................................. 3
Index. ................................................................................... 4
List of Figures. ..................................................................... 6
List of Tables. ...................................................................... 8

Chapter 1. Introduction. ........................................................ 9
  1.1 Background. ................................................................ 9
  1.2 General overview. ..................................................... 10

Chapter 2. Electrocardiograph Machines. ............................. 12
  2.1 Background. .............................................................. 12
  2.2 Normal adult 12-lead EKG. ....................................... 15
  2.3 National Standards for Electrocardiograph machines. .......... 17
    2.3.1 EC11 Standard. ................................................. 18

Chapter 3. Entering An Emerging Medical Technology to the Market. ........ 25
  3.1 Overview Of The Medical Device Industry. .................... 25
  3.2 EKG Technology and Market Opportunities. .................. 27
  3.3 Introducing New Medical Technologies to Mexico. .......... 28

Chapter 4. Hardware description ........................................ 33
  4.1 Background. .............................................................. 33
  4.2 Electrodes and Lead Selector. ...................................... 34
    4.2.1 Lead Selector. ................................................... 35
    4.2.2 Test Signal. ....................................................... 40
Index

4.3 Amplification of the EKG signal. ........................................... 43
4.4 Noise and Antialiasing Filtering. ........................................... 44
4.5 Data Acquisition. ............................................................... 48

Chapter 5. Graphic User Interface. ........................................... 57
  5.1 Visual Basic Terminal. ....................................................... 57
  5.1.1 Serial Communication. ................................................... 58
  5.1.2 Display Text. ............................................................... 59
  5.1.3 Graphic Interface. ......................................................... 59
  5.1.4 Presenting and Storing EKG Histograms. ...................... 63
  5.1.5 Lead Selector. ............................................................. 65

Chapter 6. Conclusion ............................................................ 66

APPENDIX A. Assembler Code for ADC1251 Extensions to Basic. ........... 68

APPENDIX B. Visual Basic Code For the virtual EKG Machine. ............... 81

APPENDIX C. Complete Circuit Diagram ..................................... 97

References. ............................................................................. 98
List of Figures

Figure 2.1 Standard Bipolar and Unipolar Precordial Leads. ........................................ 13
Figure 2.2 Typical 12 leads electrogram. .......................................................... 14
Figure 2.3 Clinical EKG - Components and Intervals. ............................................. 15

Figure 3.1 Typical supply channel for Home Health Care Devices. ......................... 28
Figure 3.2 Age distribution in Mexico by 1996. .................................................. 30

Figure 4.1 EKG Monitor System Block Diagram. .................................................... 33
Figure 4.2 Interface Circuit. ................................................................................. 34
Figure 4.3 Lead Selector Buffer. .......................................................................... 35
Figure 4.4 Lead Selector Circuit. .......................................................................... 38
Figure 4.5 Switch Control Diagram. ..................................................................... 40
Figure 4.6 Triangular wave signal for Method D. ................................................. 41
Figure 4.7 Triangular wave generator. .................................................................. 41
Figure 4.8 Amplifier Circuit. ................................................................................ 43
Figure 4.9 Frequency response of an 8th order low-pass filter. ......................... 45
Figure 4.10 Circuit diagram of an 8th order high-pass filter. .............................. 46
Figure 4.11 Frequency response of an 8th order high-pass filter. ....................... 46
Figure 4.12 Noise Filters Block Diagram. ......................................................... 47
Figure 4.13 EKG signal before and after being filtered. .................................... 47
Figure 4.14 Antialiasing Filter. .......................................................................... 48
Figure 4.15 Timing diagram for the Calibration Cycle. ....................................... 49
Figure 4.16 Auto-calibration cycle Flow diagram. .............................................. 50
Figure 4.17 Timing diagram using S/H to start a conversion without Auto-zero. .... 51
Figure 4.18 ADC1251 Clock circuit. .................................................................. 51
Figure 4.19 ADC1251 Interface circuit. ............................................................... 53
Figure 4.20 Flow diagram for an A/D conversion. ............................................. 54
Figure 4.21 Simplified Flow diagram of the capture command. ......................... 55
List of Figures

Figure 5.1 Graphic user interface flow diagram. .............................................. 58
Figure 5.2 Data acquisition flow diagram. ...................................................... 60
Figure 5.3 VEKG. Selecting 3 seconds of signal to be stored. ........................... 62
Figure 5.4 VEKG. Presentation of an Electrogram. .......................................... 63
Figure 5.6 VEKG. Example of a Printed Electrogram. ..................................... 64
List of Tables

Table 2.1 Definition of Electrode Connection. ......................................................... 19
Table 2.2 Operating Conditions. .................................................................................. 20
Table 2.3 Definition of Leads (EC11). ....................................................................... 21
Table 2.4 Summary of Performance Requirements. .................................................... 22

Table 3.1 Medical Infrastructure in Mexico by 1996. .................................................... 29

Table 4.1 Switch Selector Values. ................................................................................ 39
Table 4.2 Frequency response. ..................................................................................... 40

Table 5.1 Lead Selector Controller. ............................................................................. 65
CHAPTER I

INTRODUCTION

1.1 Background

The electrocardiogram is an important and common medical procedure, providing an insight into the patient's cardiac function [1]. The electrocardiogram provides valuable information to the clinician [1]. It is particularly useful in defining cardiac rhythm, identifying chamber hypertrophy and documenting ischemia and infarction [2].

Cardiovascular diseases still cause 12 million deaths in the world each year, according to the third monitoring report of the World Health Organization, 1991-1993 [3, 4]. They cause half of all deaths in several developed countries, and are one of the main causes of death in many developing countries and the major cause in adults. In Mexico, data indicate approximately 65,000 deaths from cardiovascular diseases per year. This represent about 15% of all deaths.

In the hospital environment, EKG may be monitored continuously to alert medical staff of any sudden changes in patient status. EKG readings are also taken in an emergency outside the hospital by paramedics where the EKG machine is small and portable.

While the elite in developing countries have access to private hospitals with services equivalent to those in developed countries, the rest of the population must depend on public hospitals and services frequently lacking the most critical supplies and unable to modernize their technological infrastructure. In addition, shortages of parts and deficiencies in maintenance have paralyzed many installations.

The goal of this work is to develop an EKG machine that is cost effective to be used in clinics of developing countries like Mexico.

Accordingly to information from the government of Mexico [5], the number of operating rooms in Mexico is 2,568; from those 1,156 are in private hospitals and the rest is in a government hospitals. There are 133,711 hospital beds from which about 55% are in private hospitals. That same source states that there are 1,698 Electrocardiograph machines in private
Chapter I. Introduction

hospitals; from the information above we can estimate that there are about 1,389 EKG machines in public (government) hospitals and a total of 3,087 in the whole country. The total population in Mexico is about 95 Million.

The cost of an EKG machine in Mexico is about US$5,000.00 to $10,000.00. The estimated cost of the “Virtual EKG Machine” including a computer is estimated to be less than US$2,000.00. A computer can be repaired everywhere, reducing maintenance and supply costs considerably. For example, the EKG machines use a special paper that is quite expensive; by using a computer, we have the advantage of printing the electrogram as many times as we want, along with the patient name and some other information that can be useful on a common piece of bond paper that doesn’t fade away.

In addition to the economic advantage of this machine it can be used for other purposes like database, information storage and other applications usually done with a personal computer. Everyday more personal computers get connected to each other. The information that was obtained using one computer can be sent directly to the physician personal computer where he or she can evaluate it. By using a computer, the information about the patient heart condition may be evaluated automatically.

1.2 General Overview

The traditional electrocardiogram machine has 5 or more wires terminating in electrodes connected to the patient. The machine operator might be a doctor, nurse, or medical technician who can judge the quality of signals and re-prepare and re-apply electrodes in the case of poor signal quality. We want to preserve rather than automate this "judgment" process. So the instrument must be able to present a waveform adequate to make this judgment and have rapid-enough response to easily allow the operator to associate a change in the waveform with an intervention. We propose to allow substitution of a test waveform to verify function of the instrument but will rely on operator judgment to record and select appropriate waveform samples.

The Second Chapter presents a background on Electrocardiography and a summary of the National Standards for Electrocardiograph machines. The Third Chapter discusses some
Chapter I. Introduction

issues about introducing a Medical Device to the Market. The Fourth Chapter shows the
description of the proposed Hardware implementation from the patient leads, to the lead selector,
signal preparation, quantization, and signal communication devices. The Fifth Chapter explains
the proposed software implementation that presents the actual data on the computer monitor.
Finally, I present the Results, Conclusions and suggest improvements.
Chapter II.

Electrocardiograph Machines

2.1 Background.

Basically, an electrocardiograph machine records a graphic tracing of the electric current generated by the heart muscle during a heartbeat. Electrograms are sets of recording made by applying electrodes to various parts of the body. Twelve records constitute a typical clinical electrocardiogram. About 3 seconds of data are present for each of the 12 leads. Often a longer “rhythm strip” is recorded.

On a 12 Leads electrocardiograph, there are 12 different voltages differences that can be printed, divided on 3 groups:

- **Bipolar limb leads**: The electrodes are connected on the left arm, on the right arm and on the left leg. The Leads derived of this connection are:
  
  - I = LA-RA (Voltage of Left arm minus Voltage of Right arm).
  - II = LL-RA (Voltage of Left leg minus Voltage of Right arm).
  - III = LL-LA (Voltage of Left leg minus Voltage of Left arm).

- **Augmented Extremity Leads**: In this mode the electrodes are also connected to the left arm, right arm and left leg. The voltage difference printed is the voltage of one electrode minus the voltage average of the other two:
  
  - aVR = RA - 0.5 (LA + LL)
  - aVL = LA - 0.5 (LL + RA)
  - aVF = LL - 0.5 (LA + RA)

- **Unipolar Precordial Leads**: In this mode the electrodes are also connected to the left arm, right arm and left leg, and also there is a electrode on 6 different places of the chest. The voltage difference printed is the voltage on the chest electrode minus the average of the other three electrodes.
  
  V1 = V - 1/3(LA + RA + LL)
Chapter II. Electrocardiograph Machines

\[
\begin{align*}
V2 &= V - \frac{1}{3}(LA + RA + LL) \\
V3 &= V - \frac{1}{3}(LA + RA + LL) \\
V4 &= V - \frac{1}{3}(LA + RA + LL) \\
V5 &= V - \frac{1}{3}(LA + RA + LL) \\
V6 &= V - \frac{1}{3}(LA + RA + LL)
\end{align*}
\]

Figure 2.1 (a) Standard Bipolar Leads, and (b) Unipolar Precordial Leads

On Figure 2.2 we can see a typical 12 lead electrogram. As we can see, it show information about the patient (Age, Height, Weight, Sex, Race), Hospital Information (Physician name, Location, room), title of test, information about the machine settings (Strip speed, gain, bandwidths of signal), and 12 three second signals - one for each lead - and one 12 second strip of a particular lead.

---

Figure 2.2 Typical 12 leads electrocardiogram.
Chapter II. Electrocardiograph Machines

Any deviation from the norm in a particular electrocardiogram is indicative of a possible heart disorder [7]. Information that can be obtained from an electrocardiogram includes whether the heart is enlarged and where the enlargement occurs, whether the heart action is irregular and where the irregularity originates, and whether a slow rate is physiological or caused by heart block. The presence of high blood pressure, thyroid disease, and certain types of malnutrition may also be revealed by an electrocardiogram.

Generally the machine has a dial that the operator moves to select the type of lead used. Depending on the lead used, there will be a characteristic wave that a doctor can interpret, with the knowledge of the physical condition of the patient.

2.2 Normal adult 12-lead EKG

Figure 2.3 Clinical EKG- Components and Intervals
Chapter II. Electrocardiograph Machines

From [7] and [8] I found that the diagnosis of the normal electrocardiogram is made by excluding any recognized abnormality. It's description is therefore quite lengthy, and it is important to interpret each tracing in a standard fashion. A commonly followed sequence of analysis is as follows:

- normal sinus rhythm.
  Each P wave is followed by a QRS; P waves normal for the subject (upright in leads I, II, and III); P wave rate 60 - 100 bpm with <10% variation; rate <60 = sinus bradycardia; rate >100 = sinus tachycardia; variation >10% = sinus arrhythmia

- normal QRS axis

- normal P waves
  Height < 2.5 mm in lead II; width < 0.11 s in lead II; for abnormal P waves consider right atrial hypertrophy, left atrial hypertrophy, atrial premature beat, hyperkalaemia

- normal PR interval
  0.12 to 0.20 s (3 - 5 small squares); for short PR segment consider Wolff-Parkinson-White syndrome or Lown-Ganong-Levine syndrome (other causes - Duchenne muscular dystrophy, type II glycogen storage disease (Pompe's), HOCM) for long PR interval see first degree heart block

- normal QRS complex
  < 0.12 s duration (3 small squares); for abnormally wide QRS consider right or left bundle branch block, ventricular rhythm, hyperkalaemia, etc.; no pathological Q waves no evidence of left or right ventricular hypertrophy

- normal QT interval
  Calculate the corrected QT interval (QTc) by dividing the QT interval by the square root of the preceding R - R interval. Normal = 0.42 s.

1 A Normal adult 12-lead ECG. http://homepages.enterprise.net/djenkins/norm.html
Chapter II. Electrocardiograph Machines

Causes of long QT interval: myocardial infarction, myocarditis, diffuse myocardial disease; hypocalcaemia, hypothyroidism; subarachnoid haemorrhage, intracerebral haemorrhage; drugs (e.g. sotalol, amiodarone); Hereditary: Romano Ward syndrome (autosomal dominant); Jervill + Lange Nielson syndrome (autosomal recessive) associated with sensorineural deafness

- normal ST segment
  No elevation or depression; causes of elevation include acute MI (e.g. anterior, inferior), left bundle branch block, normal variants (e.g. athletic heart, Edeiken pattern, high-take off), acute pericarditis causes of depression include myocardial ischaemia, digoxin effect, ventricular hypertrophy, acute posterior MI, pulmonary embolus, left bundle branch block

- normal T wave
  Causes of tall T waves include hyperkalaemia, hyperacute myocardial infarction and left bundle branch block. Causes of small, flattened or inverted T waves are numerous and include ischemia, age, race, hyperventilation, anxiety, drinking iced water, LVH, drugs (e.g. digoxin), pericarditis, PE, intraventricular conduction delay (e.g. RBBB) and electrolyte disturbance.

- normal U wave

  This use requires that the wave is printed in a ruled paper with standardized scales, like the ones showed in figure 2.3.

2.3 National Standards for Electrocardiograph machines

There are standards developed by The Association for the Advancement of Medical Instrumentation (AAMI) and approved by the American National Standards Institute, Inc.(ANSI), that provides limits and measurement techniques for medical apparatus.

- ES1, is the “American National Standard, Safe Current Limits for Electromedical Apparatus.
  This standard provides limits and measuring techniques for risk currents of electromedical
apparatus as a function of frequency, the characteristics of the apparatus, and the nature of the intentional contact with the patient” [9].

- **EC11** is the “American National Standard for Diagnostic Electrocardiographic Devices. This standard establishes minimum safety and performance requirements for electrocardiographic (EKG) systems, with direct writing devices, which are intended for use in EKG contour analysis for diagnostic purposes”. This standard defines requirements for the electrocardiographic recording system, from the input electrodes to the output display [10].

### 2.3.1 EC11 Standard

The EKG system that we are proposing is included in this standard. Some of the requirements set by this standard are:

**Labeling Requirements.**

Diagnostic EKG devices shall be clearly and permanently marked with information like the manufacturer’s name, trademark, trade name; The catalogue, style, model, or other type designation; serial number; The range of supply (mains) voltage and the maximum operating current or power; the nominal supply (mains) frequency; etc.

All controls, switches, and connectors shall be clearly and concisely labeled to identify their function. Electrical safety labels, the location of fuse holders and the patient electrode connection nomenclature and colors shall be clearly marked.

The patient Electrode Connection Definitions and Color Code for the conventional system is summarized on table 2.1.
### Table 2.1 Definition of Electrode Connection.

<table>
<thead>
<tr>
<th>Patient Electrode Connection Identifier</th>
<th>Color Code</th>
<th>Position on Body Surface</th>
</tr>
</thead>
<tbody>
<tr>
<td>RA</td>
<td>White</td>
<td>Right arm</td>
</tr>
<tr>
<td>LA</td>
<td>Black</td>
<td>Left arm</td>
</tr>
<tr>
<td>LL</td>
<td>Red</td>
<td>Left leg</td>
</tr>
<tr>
<td>V</td>
<td>Brown</td>
<td>Single movable chest electrode</td>
</tr>
<tr>
<td>V1</td>
<td>Brown/Red</td>
<td>4&lt;sup&gt;th&lt;/sup&gt; intercostal (IC) space at right border of sternum</td>
</tr>
<tr>
<td>V2</td>
<td>Brown/Yellow</td>
<td>4&lt;sup&gt;th&lt;/sup&gt; IC space at left border of sternum</td>
</tr>
<tr>
<td>V3</td>
<td>Brown/Green</td>
<td>Midway between V2 and V4</td>
</tr>
<tr>
<td>V4</td>
<td>Brown/Blue</td>
<td>5&lt;sup&gt;th&lt;/sup&gt; IC space on left midclavicular line</td>
</tr>
<tr>
<td>V5</td>
<td>Brown/Orange</td>
<td>Left anterior axillary line at the horizontal level of V4</td>
</tr>
<tr>
<td>V6</td>
<td>Brown/Violet</td>
<td>Left midaxillary line at the horizontal level of V4</td>
</tr>
<tr>
<td>RL</td>
<td>Green</td>
<td>Right leg</td>
</tr>
<tr>
<td>I</td>
<td>Orange/Red</td>
<td>At the right midaxillary line&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>E</td>
<td>Orange/Yellow</td>
<td>At the front midline&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>C</td>
<td>Orange/Green</td>
<td>Between front midline and left midaxillary line at angle of 45 degrees</td>
</tr>
<tr>
<td>A</td>
<td>Orange/Brown</td>
<td>At the left midaxillary line&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>M</td>
<td>Orange/Black</td>
<td>At the back midline&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>H</td>
<td>Orange/Violet</td>
<td>On the back of the neck or on the forehead</td>
</tr>
<tr>
<td>F</td>
<td>Red</td>
<td>On the left leg</td>
</tr>
</tbody>
</table>

<sup>1</sup>Located at the transverse level of the ventricles.
Chapter II. Electrocardiograph Machines

An operator's manual, containing adequate instructions for the proper installation and the safe and effective operation of the device and identifying acceptable repair facilities, shall be provided with each unit. At least the following information shall be supplied:

- **Disclosure of Cautionary Information/Performance Characteristics**: Cautionary information regarding potential hazards/damage, including warnings on use of device in presence of electromagnetic interference or power overload caused by electrosurgical or diathermy instruments.

- **Battery-Powered Devices**: Minimum operating time; battery charge time; function of battery depletion indicator, if provided.

- **Accuracy of Input signal Reproduction**: description of methods used by manufacturer to establish overall system error and frequency response; description of modulating effects in digital systems.

Application notes: Description of device's intended applications and available functions; procedures for checking controls and functions; manufacturer's recommendations concerning electrodes. A service Manual, containing adequate care, preventive maintenance, and repair instructions; electrical specifications complete enough to allow reasonable field repair; identification of acceptable repair facilities; recommended frequency of preventive maintenance.

**Operating Requirements**: Unless otherwise stated, the performance requirements of this standard shall be met under the following ambient environmental conditions:

<table>
<thead>
<tr>
<th>Line Voltage:</th>
<th>104 to 127 Vrms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Frequency:</td>
<td>60 ± 1 Hz</td>
</tr>
<tr>
<td>Temperature:</td>
<td>25 ± 10 °C</td>
</tr>
<tr>
<td>Relative Humidity:</td>
<td>50 ± 20 %, noncondensing</td>
</tr>
<tr>
<td>Atmospheric pressure:</td>
<td>7 x10^4 to 10.6 x10^4 Pa</td>
</tr>
</tbody>
</table>

*Table 2.2 Operating Conditions*
Chapter II. Electrocardiograph Machines

The definition of lead sets employing the twelve conventional or orthogonal leads shall comply to Table 2.3.

The definition of the leads is given in terms of algebraic equation, assuming that the electrode identifier represents the voltage sensed by the electrode. For the unipolar chest leads, V represents the potential at each respective chest electrode location. By convention, X is oriented horizontally and towards the left arm of the patient, Y points towards the feet, and Z is horizontal and towards the back of the patient.

<table>
<thead>
<tr>
<th>Lead Nomenclature</th>
<th>Definition</th>
<th>Name of Lead</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>I = LA - RA</td>
<td>Bipolar</td>
</tr>
<tr>
<td>II</td>
<td>II = LL - RA</td>
<td>limb leads</td>
</tr>
<tr>
<td>III</td>
<td>III = LL - LA</td>
<td>(Einthoven)</td>
</tr>
<tr>
<td>aVR</td>
<td>aVR = RA - 0.5 (LA + LL)</td>
<td>Augmented leads</td>
</tr>
<tr>
<td>aVL</td>
<td>aVL = LA - 0.5 (LL + RA)</td>
<td>(Goldberger)</td>
</tr>
<tr>
<td>aVF</td>
<td>aVF = LL - 0.5 (LA + RA)</td>
<td>(Goldberger)</td>
</tr>
<tr>
<td>V1</td>
<td>V1 = V - 0.333(LA + RA + LL)</td>
<td>Unipolar Chest leads</td>
</tr>
<tr>
<td>V2</td>
<td>V2 = V - 0.333(LA + RA + LL)</td>
<td>(Wilson)</td>
</tr>
<tr>
<td>V3</td>
<td>V3 = V - 0.333(LA + RA + LL)</td>
<td>(Wilson)</td>
</tr>
<tr>
<td>V4</td>
<td>V4 = V - 0.333(LA + RA + LL)</td>
<td>(Wilson)</td>
</tr>
<tr>
<td>V5</td>
<td>V5 = V - 0.333(LA + RA + LL)</td>
<td>(Wilson)</td>
</tr>
<tr>
<td>V6</td>
<td>V6 = V - 0.333(LA + RA + LL)</td>
<td>(Wilson)</td>
</tr>
<tr>
<td>X</td>
<td>X = 0.610A + 0.171C - 0.781I</td>
<td>Orthogonal</td>
</tr>
<tr>
<td>Y</td>
<td>Y = 0.655F + 0.345M - 1.000H</td>
<td>Vector leads</td>
</tr>
<tr>
<td>Z</td>
<td>Z = 0.133A + 0.736M - 0.264I - 0.374E - 0.231C</td>
<td>(Frank)</td>
</tr>
</tbody>
</table>

Table 2.3. Definition of leads (EC11).

Table 2.4. Provides a summary of some other performance requirements of the EC11 standard.
### Chapter II. Electrocardiograph Machines

<table>
<thead>
<tr>
<th>Requirement Description</th>
<th>Min/Max</th>
<th>Units</th>
<th>Min/Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input Dynamic Range:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Range of linear operations of input signal</td>
<td>min</td>
<td>mV</td>
<td>±5</td>
</tr>
<tr>
<td>Slew rate change</td>
<td>max</td>
<td>mV/sec</td>
<td>320</td>
</tr>
<tr>
<td>DC offset voltage range</td>
<td>min</td>
<td>mV</td>
<td>±300</td>
</tr>
<tr>
<td>Allowed variation of amplitude with DC offset</td>
<td>max</td>
<td>%</td>
<td>±5</td>
</tr>
<tr>
<td><strong>Gain Control, Accuracy, and Stability</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gain selections</td>
<td>min</td>
<td>mm/mV</td>
<td>20, 10, 5</td>
</tr>
<tr>
<td>gain error</td>
<td>max</td>
<td>%</td>
<td>5</td>
</tr>
<tr>
<td>manual override of automatic gain control</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>gain change rate/min</td>
<td>max</td>
<td>%/min</td>
<td>±0.33</td>
</tr>
<tr>
<td>total gain change/hour</td>
<td>max</td>
<td>%</td>
<td>±3</td>
</tr>
<tr>
<td><strong>Time Base Selection and Accuracy:</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time base selections</td>
<td>min</td>
<td>mm/sec</td>
<td>25, 50</td>
</tr>
<tr>
<td>Time base error</td>
<td>max</td>
<td>%</td>
<td>±5</td>
</tr>
<tr>
<td><strong>Output Display</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Width of display</td>
<td>min</td>
<td>mm</td>
<td>40</td>
</tr>
<tr>
<td>trace visibility</td>
<td>max</td>
<td>mm/sec</td>
<td>1600</td>
</tr>
<tr>
<td>trace width</td>
<td>max</td>
<td>mm</td>
<td>1</td>
</tr>
<tr>
<td>departure from time axis alignment</td>
<td>max</td>
<td>mm</td>
<td>0.5</td>
</tr>
<tr>
<td>max</td>
<td>msec</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>preruled paper division</td>
<td>min</td>
<td>div/cm</td>
<td>10</td>
</tr>
<tr>
<td>error of rulings</td>
<td>max</td>
<td>%</td>
<td>±2</td>
</tr>
<tr>
<td>time marker error</td>
<td>max</td>
<td>%</td>
<td>±2</td>
</tr>
</tbody>
</table>
### Chapter II. Electrocardiograph Machines

<table>
<thead>
<tr>
<th>Requirement Description</th>
<th>Min/Max</th>
<th>Units</th>
<th>Min/Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy of Input Signal Reproduction</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall error for signals</td>
<td>max</td>
<td>%</td>
<td>± 5</td>
</tr>
<tr>
<td>up to ±5 mV &amp; 125 mV/sec</td>
<td>max</td>
<td>µ V</td>
<td>± 40</td>
</tr>
<tr>
<td>upper cut-off frequency (3 dB)</td>
<td>min</td>
<td>Hz</td>
<td>150</td>
</tr>
<tr>
<td>Response to 20 ms, 1.5 mV triangular input</td>
<td>min</td>
<td>mm</td>
<td>13.5</td>
</tr>
<tr>
<td>Response to 0.3 mV·s impulse</td>
<td>max</td>
<td>mV</td>
<td>0.1</td>
</tr>
<tr>
<td>Displacement slope</td>
<td>max</td>
<td>mV/s</td>
<td>0.30</td>
</tr>
<tr>
<td>error in lead weighting factors</td>
<td>max</td>
<td>%</td>
<td>5</td>
</tr>
<tr>
<td>Deflection from baseline</td>
<td>max</td>
<td>mm</td>
<td>0.5</td>
</tr>
<tr>
<td><strong>Standardizing Voltage</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nominal Value</td>
<td>NA</td>
<td>mV</td>
<td>1.0</td>
</tr>
<tr>
<td>Rise Time</td>
<td>max</td>
<td>msec</td>
<td>1</td>
</tr>
<tr>
<td>Decay Time</td>
<td>min</td>
<td>sec</td>
<td>100</td>
</tr>
<tr>
<td>Amplitude error</td>
<td>max</td>
<td>%</td>
<td>± 5</td>
</tr>
<tr>
<td><strong>Input Impedance at 10 HZ (Each lead)</strong></td>
<td>min</td>
<td>megohms</td>
<td>2.5</td>
</tr>
<tr>
<td><strong>DC Current</strong></td>
<td>max</td>
<td>µ A</td>
<td>0.1</td>
</tr>
<tr>
<td>(any input lead)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(any other patient electrode)</td>
<td>max</td>
<td>µ A</td>
<td>1.0</td>
</tr>
<tr>
<td><strong>Common Mode Rejection</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>allowable Noise with 20 V, 60 Hz &amp; ± 300 mV dc.</td>
<td>max</td>
<td>mm</td>
<td>10</td>
</tr>
<tr>
<td>&amp; 51 - kilohm imbalance</td>
<td>max</td>
<td>mV</td>
<td>1</td>
</tr>
<tr>
<td><strong>System Noise</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTI, p-p</td>
<td>max</td>
<td>µ V</td>
<td>30</td>
</tr>
<tr>
<td>Multichannel crosstalk</td>
<td>max</td>
<td>%</td>
<td>2</td>
</tr>
</tbody>
</table>
### Table 2.4 Summary of Performance Requirements

Section 4 of the EC11 standard provides referee test methods and procedures by which compliance of the device with the requirements of Section 3 can be verified.
Chapter III.

Entering an Emerging Medical Technology to the Market.

We have seen many examples of innovative companies and individuals that failed to capture significant returns from their creations. This is despite the fact that the innovations often became successful and generated substantial wealth for other parties [10]. This chapter examines an emerging medical technology and speculate on how we can avoid similar fate.

This technology can be applicable in a number of markets including hospital monitoring, ambulatory monitoring, and home care monitoring. Given the limited set of resources, the market decision is an important one.

3.1 Overview of the medical device industry

The medical device industry has been dominated by the United States for many decades. American medical devices are recognized world wide for quality and innovation 1.

Rapid advances in information technology are transforming many industries, and that is evident in the medical device industry. The combination of computers, sensors, and imaging systems are replacing invasive procedures and "exploratory" surgery.

In sharp contrast with the pharmaceutical industry, the medical device industry is composed of a very large number of small companies. One reason device companies are small is that the market itself is fragmented. For example, the total US market for anesthesia machines of all types is only about 3,000 units a year with a total value of $150 million. In contrast, the US market for systemic antibiotics is $5 billion per year [11].

---

Chapter III. Entering an Emerging Medical Technology to the market

The medical device companies conduct substantial research and development. Because of the complexity and high risk of the innovation, a very large proportion of significant innovation comes from the smaller companies with the least sales. Venture capital is generally attracted to these companies in the expectation that the product or the company itself will be sold to a larger company in 5 to 7 years.

While radically new technologies attract public attention, much of the long term improvement in medical devices comes from many small incremental innovations which cumulatively, over time, have a great clinical relevance. Much of the impetus for such product improvements comes from physicians in the field. Unimpeded communication between physicians and manufacturers is a requirement for much of this activity.

It is also important to have available different sites for testing the devices in a clinical environment, generally in academic health centers. The small size of the market for any medical device and the need for specialized material for many of these devices results in device companies being dependent on suppliers outside the medical device industry. The medical industry is dependent on multiple factors beyond its direct control:

- **Time.** Small companies have little reserves and are dependent on fresh infusions of capital or on current sales to finance innovation. Delays, fear of delays, or even the unpredictability of delays can lead to financial disaster and frighten capital investors. Such delays can be due to FDA approval, difficulty or hazard’s to test the device among other.

- **Uncertainty.** Research can't predict in advance what is going to work, how well and when. It's hard to know how the conditions in the market will be or how much competition will be when the product is brought to the market. Also there is uncertainty in knowing if the product will be obsolete when it gets to the market. The companies won't know if they will have the same talented people through all the research process.

- **Liability costs.** Liability costs are very high in these industries because their products are intrinsically involved in life and death situations.

- **FDA approval.** On top of these significant scientific and market risks, the FDA approval process creates its own major uncertainties, with respect both to ultimate approval and the
Chapter III. Entering an Emerging Medical Technology to the market

time and expense required to get there. FDA negativity at any stage of the process can reduce the value of a firm and raise its effective cost of capital overnight, especially for small companies without diversified (or even marketed) product lines. These regulatory uncertainties reduce the returns from research and development in the pharmaceutical and medical device sectors and, therefore, the number of new therapeutic products that can be developed.

3.2 EKG Technology and Market Opportunities

Electrocardiography is one of the pioneers in the medical device industry. Frank B. Sanborn, a civil engineering professor at Tufts, was on the pioneering edge of medical technology in the 1920s when his company invented among other equipment the first Table Model Electrocardiograph.

In 1928, the table model was converted to a portable EKG. Powered by a 6-volt automobile battery, the portable EKG weighted 50 pounds. This era also marked the beginning of many working partnerships with members of the medical community.

In 1931, the first research model EKG was installed in Mass General Hospital and Sanborn's relationship with the hospital continued through the years. During the 1940s and 1950s Sanborn saw a five-fold increase in sales and productivity reached an all-time high.

By 1960, Sanborn was faced with excessively high inventory, some uncertainty in industrial sales, and an increase in product prices. In 1961 Sanborn Company was merged with Hewlett-Packard Company.

In the 1970s EKG companies started to explicitly consider issues of patient safety. By the 1980's EKG analysis was introduced; the EKG industry shifted from product development to process development. By this time there is a dominant design, and the competition shifted to price and away from design.

In the last decade we have seen the creation of the market for home health care devices with the introduction of personal blood pressure monitors and glucometers among others. These
Chapter III. Entering an Emerging Medical Technology to the market

are necessarily simple devices sold through retail channels such as pharmacies (i.e. CVS, Wallgreens) and mass merchandisers (i.e. Kmart and Target).

The channel used for this market is very different from normal medical device channels. Home health care devices is the only segment sold through retail stores. Therefore, the market more closely resembles that of a consumer good than a traditional device. The typical supply for Home health care devices is structured as shown in Figure 3.1:

![Figure 3.1. Typical supply channel for Home Health Care devices](image)

### 3.3 Introducing New Medical Technologies to Mexico

The market that we are trying to focus is clinical electrocardiography in health provider facilities, in particular poor clinics in the developing countries. Although this is a problematic market because of resource limitation, there is a big need for health care devices. In 1996, 436,321 were registered dead. From those, 65,603 (about 15%) died directly from heart related causes. If for every dead person, there are 10 sick people that need to take an electrogram weekly, about 100,000 daily electrograms are needed to be taken. 10 electrograms per day is the normal use of this machines, and for one active machine, there is another with practically no work load (it may be in a physician office). About 20,000 electrocardiograph machines are required, minus 3,086 estimated actual machines, about 17,000 machines are needed.

The EKG machine that we are developing is built within a computer, therefore will be much less expensive than a stand-alone machine. There are a limited number of clients in this market, but a client can be a government health care department that buy many equipment for government hospitals that are common in some developing countries like Mexico.

In general, the hospital disposition in Mexico is divided into Private Hospitals and Clinics, and Public Hospitals and clinics. In general, the private hospitals are independent to each other. The Public ones are divided into different groups:
Chapter III. Entering an Emerging Medical Technology to the market

- **SSA.** Directly dependent on the Ministry of Health.
- **DDF.** Directly dependent on the Government of Mexico City.
- **IMSS-SOL.** Maintained by Social Security payments of the workers.
- **STATE HOSPITALS.** Directly dependent on the Government of each state of the federation.
- **ISSSTE.** For the use of the State dependent workers.
- **PEMEX.** For the use of the PEMEX (Petroleos Mexicanos) workers.
- **SDN.** Military Hospitals.
- **SM.** Marine Hospitals

Accordingly to information from the government of Mexico [5], the number of operating rooms in Mexico is 2,568; from those 1,156 are in private hospitals and the rest is in government hospitals. There are 133,711 hospital beds from which about 55% are in private hospitals. That same source states that there are 1,698 Electrocardiograph machines in private hospitals; from the information above we can estimate that there are about 1,389 EKG machines in public (government) hospitals and a total of 3,087 in the whole country. The total population in Mexico is about 95 Million. This information is shown in table 3.1:

<table>
<thead>
<tr>
<th>Hospital Type</th>
<th>Pop. Covered in thousands total: 94,732</th>
<th>Number of Clinics</th>
<th>Operating Rooms</th>
<th>Doctors</th>
<th>EKG machines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public</td>
<td>34,423</td>
<td>17,872</td>
<td>1,412</td>
<td>37,620</td>
<td>1,389*</td>
</tr>
<tr>
<td>Private</td>
<td>53,447*</td>
<td>12,928</td>
<td>1,156</td>
<td>58,411</td>
<td>1,698</td>
</tr>
<tr>
<td>Total</td>
<td>87,870*</td>
<td>30,800</td>
<td>2,568</td>
<td>96,031</td>
<td>3,086*</td>
</tr>
</tbody>
</table>

*Table 3.1. Medical infrastructure in Mexico by 1996. (* Estimated).*

As people gets older the Health problems increases. In the next figure, I show the Age distribution of the population of Mexico.
Chapter III. Entering an Emerging Medical Technology to the market

The cost of an EKG machine in Mexico is about US$5,000.00 to $10,000.00. The estimated cost of the “Virtual EKG Machine” including a computer is estimated to be less than US$2,000.00. A computer can be repaired everywhere, reducing maintenance and supply costs considerably. For example, the EKG machines use a special paper that is quite expensive; by using a computer, we have the advantage of printing the electrograms as many times as we want, along with the patient name and some other information that can be useful on a common piece of bond paper that doesn’t fade away.

One problem of the inexpensive EKG Machines now available is that they go out of calibration very often. By having fewer mechanical components, this will less likely in the Virtual Machine.

Being a market entrant has some advantages and disadvantages versus market incumbents in architectural innovation and in particular in the medical device industry. Some advantages are that entrants innovate at lower cost, entrants will chase small profit markets. A disadvantage of the entrants is that they have less Complementary assets than the incumbents. In the next paragraphs, I show the complementary Assets that a medical supply company should have.
Chapter III. Entering an Emerging Medical Technology to the market

- **Appropriability of technology.** The technology of developing an EKG machine is widely used and hard to patent.

- **Switching costs.** There are some switching cost related to changing from stand alone machines to computer based systems. This costs are of two types; first is the obvious, the cost of buying the new machine. The second cost is the technology learning cost; the cost related on instructing the physicians or users how to use the product. These costs can be justified by the lower cost of the machines and most important of the supplies (while a HP EKG paper costs $20 per roll, the regular printer paper costs less than $5 per 100 pages). The computer software can have "virtual" control switches, so the physicians can use it the same way they use a stand alone machine (the less disruptive possible to the user).

- **Access to retail distribution.** For the Home Health Care Market, the product is intended to be sold directly to the customer. Therefore, customers channels like drug stores (i.e. CVS, Wallgreens) and general mass merchandisers (i.e. K-mart, Target) are very important. For the Electrogram market, access to regional distributors is important.

- **Brand name recognition / Reputation in the market.** For the Home Health Care, brand name recognition is important. For the Electrogram market, good service and quality provides good reputation which encourages more Hospitals to buy the product.

- **Manufacturing capabilities.** Price is very important for this product, since we claim that the price of this machine is going to be a lot less expensive than the stand-alone Machines. Most of the machine is the personal computer, which is available everywhere. Access to low cost manufacturing is available through contract manufacturers.

When developing or sourcing EKG devices, the manufacturers or distributors need to be aware of regulatory issues. In the United States, all medical devices must pass through an FDA approval process known as 510(k). For EKG systems the key standard is one developed by the Association for the Advancement of Medical Instrumentation (AAMI) and approved by the American National Standard Institute (ANSI). This standard establishes minimum safety and performance requirements for electrocardiographic (EKG) systems, which are intended for use in EKG contour analysis for diagnostic purposes. This standard defines requirements for the electrocardiographic recording system, from the input electrodes to the output display.
Chapter III. Entering an Emerging Medical Technology to the market

Approval times for these devices have to be taken into account in the developing/marketing plan. Beyond the FDA, there are no further regulatory issues in the US. The strategy is to develop an instrument which meets US standards and could be marketed in the U.S. but to focus the market in Mexico. If we plan to commercialize the product in some other countries we have to study their safety standard for this type of devices, and plan ahead to get the regulatory permissions on those countries; there is not an specific standard for EKG machines in Mexico.
Chapter IV

Hardware Description.

4.1 Background

The traditional electrocardiogram machine has 5 or more wires terminating in electrodes connected to the patient. The machine operator might be a doctor, nurse, or medical technician who can judge the quality of signals and re-prepare and re-apply electrodes in the case of poor signal quality. We want to preserve rather than automate this "judgment" process. So the instrument must be able to present a waveform adequate to make this judgment and have rapid-enough response to easily allow the operator to associate a change in the waveform with an intervention. We propose to allow substitution of a test waveform to verify function of the instrument but will rely on operator judgment to record and select appropriate waveform samples.

The proposed EKG Machine can be described as a collection of several subsystems from the Lead Selector to the Optical Isolated Serial Communication to the Personal Computer,
Chapter IV. Hardware Description

including a test signal generator, Amplifier Circuit, Noise and Quantization Filters, and data capturing Microprocessor. Figure 4.2 shows a block-diagram of this circuit.

Figure 4.2 Interface circuit.

4.2 Electrodes and Lead Selector.

For economic reasons we are just amplifying one lead at a time instead of the twelve leads discussed above. We can generate the twelve leads from the Right Arm, Left Arm, Left Leg and Chest Electrodes, plus the right leg electrode used as voltage reference. In the next figure I show the buffer circuits used to generate the required electrode combinations for the leads.
4.2.1 Lead Selector

Figure 4.3 Lead Selector Buffer
Chapter IV. Hardware Description

The resistors R1- R5 in combination with the diodes D1-D8 are to protect the patient from currents above the specified by EC11 while connected to the machine. The maximum forward voltage of the diodes is 0.6 Volts divided over 50 μA, we get that the resistors should be approximately 12 KΩ or larger.

Amplifier U5 implements an active ground that should be connected to a reference point in the patient body. The active ground will help lower the common-mode; R23, R24 and C1 provides compensation to minimize oscillation.

The resistors R14-R26 are used to generate the different lead combinations, keeping the same output resistance. The different leads combinations are:

\[ I = LA - RA \]
\[ II = LL - RA \]
\[ III = LL - LA \]
\[ aVR = RA - 0.5 (LA + LL) \]
\[ aVL = LA - 0.5 (LL + RA) \]
\[ aVF = LL - 0.5 (LA + RA) \]
\[ V = V - 1/3 (LA + RA + LL) \]

Where LA is the Voltage of the left arm electrode,
RA is the Voltage of the right arm electrode,
LL is the Voltage of the left leg electrode,
V is the Voltage of the chest electrode,

Thus, the extra combinations needed are RA+ LL, RA+LA, LA+LL and RA+LA+LL. Since precision resistors are expensive, we use matched resistors. A matched resistor provides the correct combination even though the values are not precisely determined.

A test signal is introduced in order to verify the performance and response of the EKG machine. The description of the function generation system is explained in the next section.

The combined signals are connected to controlled analog switches which select the desired signals to be amplified. For example, for lead I we need to subtract the voltage value of
the right arm from the left arm. We need to send the RA signal to the positive terminal of the differential amplifier and the LA to the negative terminal, and so on as shown in figure 4.4.

The circuit draws only the amplifier input current through the switches minimizing the effort of shunted resistors. There are two Analog Switches that are good for this purpose, the LF1331 (Normally Open) and LF13332 (Normally closed). The main features of these switches are constant “ON” resistance for signals up to \( \pm 10 \) V and 100 kHz (EKG signal is \( \pm 1 \) mV and less than 100Hz); It can manage small signals, break before make action \( t_{\text{off}} < t_{\text{on}} \); It has a high open switch isolation (about -50dB at 1.0 MHz) and it is compatible with TTL levels which are going to be used for controlling them.

Since the logic of the Multiplexer that is going to be used to select the switches is inverted (i.e. An output of the Multiplexer is Low when selected and High when not selected), I decided that the use of the LF13332 Analog switches is a better choice, since they close with a low level signal and open with a high level signal.
Since there are twelve switches, twelve lines are needed to control the switches. I use some logic in order to reduce the number of control ports used. Since there are 9 different states, a 3 to 8 multiplexer can be used plus an extra port.

Since the output of the multiplexer is low for the selected output and high for the non-selected we are going to use and gates and the LF13332N that are open with a high and closed with a low. In the next table, I show the value needed for each switch for every state.
Chapter IV. Hardware Description

<table>
<thead>
<tr>
<th>Lead DCBA</th>
<th>Port 2 output</th>
<th>Mux</th>
<th>S1a</th>
<th>S1b</th>
<th>S1c</th>
<th>S1d</th>
<th>S2a</th>
<th>S2b</th>
<th>S2c</th>
<th>S2d</th>
<th>S3a</th>
<th>S3b</th>
<th>S3c</th>
<th>S3d</th>
</tr>
</thead>
<tbody>
<tr>
<td>I 1000</td>
<td>Y0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>II 1001</td>
<td>Y1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>III 1010</td>
<td>Y2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>aVR 1011</td>
<td>Y3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>aVL 1100</td>
<td>Y4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>aVF 1101</td>
<td>Y5</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>VN 1110</td>
<td>Y6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Short 1111</td>
<td>Y7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Test 0000</td>
<td>P2.3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

*Table 4.1 Switch Selector Values*

From this table we can see that:

\[
S_{1A} = \overline{Y}_3 \oplus \overline{Y}_7 \quad S_{2A} = \overline{Y}_0 \oplus \overline{Y}_1 \oplus \overline{Y}_7 \quad S_{3A} = \overline{Y}_3 \\
S_{1B} = \overline{Y}_0 \oplus \overline{Y}_4 \quad S_{2B} = \overline{Y}_4 \quad S_{3B} = \overline{Y}_6 \\
S_{1C} = \overline{Y}_1 \oplus \overline{Y}_2 \oplus \overline{Y}_5 \quad S_{2C} = \overline{Y}_5 \quad S_{3C} = P2.3 \\
S_{1D} = \overline{Y}_6 \quad S_{2D} = \overline{Y}_6 \quad S_{3D} = P2.3
\]
Chapter IV. Hardware Description

Figure 4.5 Switch Control Diagram

4.2.2 TEST SIGNAL

The EC11 Standard states that the device shall exhibit a frequency response conforming to the specifications of Table 4.2, at a gain setting of 10 mm/mV.

<table>
<thead>
<tr>
<th>Method</th>
<th>Nominal Input Amplitude (mVpp)</th>
<th>Input Frequency and Waveform</th>
<th>Relative Output Response (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.0</td>
<td>0.67 to 40 Hz, sinusoidal</td>
<td>± 10%¹</td>
</tr>
<tr>
<td>B</td>
<td>0.5</td>
<td>40 to 100 Hz, sinusoidal</td>
<td>+ 10%, -30%¹</td>
</tr>
<tr>
<td></td>
<td>0.25</td>
<td>100 to 150 Hz, sinusoidal</td>
<td>+ 10%, -30%¹</td>
</tr>
<tr>
<td>C</td>
<td>0.5</td>
<td>100 to 500 Hz, sinusoidal</td>
<td>+ 10%, -100%¹</td>
</tr>
<tr>
<td>D</td>
<td>1.5</td>
<td>0.5 to 40 Hz, sinusoidal</td>
<td>±0%, -20%²</td>
</tr>
</tbody>
</table>

¹ Relative to 10-Hz output
² Relative to 200-ms output

Table 4.2 Frequency Response
Chapter IV. Hardware Description

The instrument must meet the requirements of Methods A and D, or alternately, the requirements of all of Methods A, B, and C of Table 4.2. The manufacturer must disclose which of the two sets of requirements (or both) are met by the instrument. For method D, I designed a triangular function generator as shown in the next figure:

![Triangular Wave Signal for Method D](image)

*Figure 4.6 Triangular wave signal for Method D.*

I decided to use a DS5000 microprocessor to generate the signal to ensure that the specifications of EC-11 are followed when testing the performance of the equipment. I generate a step signal with the width of 20 ms, and then I integrate it to get the triangular wave. The circuit used to generate the triangular wave is shown in figure 4.7:

![Triangular Wave Generator Circuit](image)

*Figure 4.7 Triangular wave generator*
Chapter IV. Hardware Description

The advantage of this circuit is that several test signals can be programmed to be used to test the equipment. As shown, the signal has to be attenuated to obtain a 1.5 mV level required by EC-11. The program that I developed in Basic to generate the triangular wave is shown below:

```
1  REM  ***********************************************
2  REM  *  TRIANGULAR WAVE GENERATOR  *
3  REM  ***********************************************
10  CLOCK 1 :REM Initialize clock
20  ZERO=128  :REM Value for zero output
30  HIGH=ZERO+100 :REM Value for high output
40  LOW=ZERO-100 :REM Value for low input
50  PORT2=ZERO :REM Zero for 1 second
60  A=TIME
70  DO
80  B=TIME
90  WHILE B<A+1
100 PORT0=1 :REM Initiate integration for 0.01 seconds
110  A=TIME
120  PORT2=ZERO
130  DO
140  PORT2=HIGH
150  B=TIME
160  WHILE B<A+0.01
170  A=TIME :REM Integrate to zero for 0.01 seconds
180  DO
190  PORT2=LOW
200  B=TIME
210  WHILE B<A+0.01
220  PORT2=ZERO
230  PORT0=0
240  GOTO 20 :REM start again
```
Chapter IV. Hardware Description

4.3 Amplification of the EKG signal.

As noted on the EC-11, the dynamic range of the apparatus shall be ± 5 mV. The signal level that we want is ± 5V, so it can be processed using a ± 5 V power supply. From the previous line we can see that the signal shall be amplified 1000 times. Because of electrode offset (300mV), it is necessary to amplify with two or more steps, first an instrumentation amplifier with fixed gain, and then a fixed single-ended gain amplifier. The Instrumentation Amplifier has a low drift and offset, but the electrode offset requirement (300 mV) makes it necessary to use a high pass filter to get rid of the DC offset. EC-11 sets the minimum frequency to be 0.5 Hz; a filter with a cutoff at 0.2 Hz eliminate DC offset without disturbing the signal.

The amplifying circuit is shown on figure 4.8

Where,

U9 is an Instrumentation operational Amplifier like the AD621 with a gain of 10.

U10 is a rail-to-rail operational amplifier like the LMC6484 with a gain of 100.

C2 and R33 are the capacitor and resistor used as a High-pass filter respectively (To eliminate the DC offset).
4.4 Noise and Anti-aliasing Filters

Once the signal is amplified, unwanted components of the signal shall be filtered, in particular the 60 Hz noise from the power lines. EC-11 specifies that the device shall exhibit a frequency response conforming to the specifications of Table 4.2, at a gain setting of 10 mm/mV.

From table 4.2 of section 4.2.3 we can see that the relative output response is limited to ±10% for method A, +10%, -30% for method B, +10%, -100% for method C, and +0%, -20% for method D.

A filter at 60 Hz may be required to remove the power line noise. We don’t want the signal to be attenuated more than 10 percent at 40Hz, and around 40 dB at 60 Hz. A high order low pass filter is required like the MAX29X. I recommend to use the Bessel filter, since it has a better performance in the time domain [16].

The amplitude response (asymptotic behavior) of the low pass filter is given by:

\[
Gain[\text{dB}] = -20n \times \log_{10}\left(\frac{\omega}{\omega_p}\right)
\]

where,

n is the filter order,

\(\omega\) is the desired frequency, and

\(\omega_p\) is the cutoff frequency.

In order to comply with EC11 specifications, we set \(\omega_p\) to be 40 Hz. In the MAX29X, the internal clock determine the cutoff frequency; the value of the capacitor can be calculated by the equation:
Chapter IV. Hardware Description

\[ f_{osc}(KHz) = \frac{10^5}{3 \times Cosc(pF)} , \]

where,

\( f_{osc} \) is the frequency of oscillation in Khz, and it is 100 times \( op \), and

\( Cosc \) is the capacitor value in PicoFaradays.

![Frequency response of the low-pass filter](image)

**Figure 4.9 Frequency response of the low-pass filter**

Method B of table 4.2 specifies that for a sinusoidal input signal of frequency 40 to 100 Hz, the relative output signal doesn’t vary more than +10% and -30%. Although, this method is not necessary to comply with EC-11 if Methods A, C and D are met, some of the signal in this range may be relevant for a physician. I recommend the use of a high pass filter with -40 dB at 60 Hz of Bessel type since it has better response in the time domain than Butterworth or Chebyshev filters. Figure 4.10 presents us the circuit diagram, while Figure 4.11 shows the frequency response of a high-pass filter adequate for this task.
These two filters effectively realize a notch filter at 60 Hz. The disadvantage of using a notch filter is that the components have to be very precise and stable. For this reason, I recommend the use of the two filters discussed above. A block diagram of the filters is shown on figure 4.12.
Chapter IV. Hardware Description

![Block diagram of noise filters](image)

**Figure 4.12 Noise Filters Block diagram**

The signals coming out of the low pass filter and high pass filter are added together to get the original signal without the 60 Hz noise. The next figure shows us two EKG signals before and after being filtered.

![EKG signals before and after filtering](image)

**Figure 4.13 EKG signal before and after being filtered.**
Chapter IV. Hardware Description

The next step is to sample the data using an analog to digital converter. Since we are going to sample at a rate of 200 samples per second, an antialiasing filter with a cutoff at 100 Hz is needed to avoid signal-aliasing. We can use the uncommitted operational amplifier of the MAX29X, to make a second order low-pass filter with a cutoff at 100Hz. The next figure shows us the diagram of the antialiasing filter.

![Antialiasing filter diagram](image)

*Figure 4.14 Antialiasing filter*

4.5 Data Acquisition

After the signal is filtered it gets sampled using an Analog-to-Digital (AD) converter. In general the quantization error produced at the AD conversion is equivalent to the value of $\frac{1}{2}$ Least-Significant-Bit (LSB). For an 8 bit converter is $\frac{1}{2^{8}}$ of the operating range. The EC11 sets an operating range of $\pm 5$ mV; that gives us $\frac{10}{(2^{8})}$ mV = 39 $\mu$V. From table 2.4 we can see that the maximum error shall be $\pm 50$ $\mu$V. In order to be able to see the P-wave, the resolution has to be increased. A 12 bit AD converter + sign give us a quantization error of $\frac{10}{(2^{13})}$ = 1.22 $\mu$V. This quantization noise is considerably smaller than the specified in the EC11 standards.

The ADC1251 is a CMOS 12-bit plus sign successive approximation analog-to-digital converter. On request, the ADC1251 goes through a self-calibration cycle that adjusts for any
zero, full scale, or linearity errors. The ADC1251 also has the ability to go through an Auto-Zero cycle that corrects the zero error during every conversion.

The analog input to the ADC1251 is tracked and held by the internal circuitry, so an external sample-and-hold is not required. The ADC1251 has an S/H control input which directly controls the track-and-hold state of the A/D. A unipolar analog input voltage range (0 to +5V) or a bipolar range (-5V to +5V) can be accommodated with ±5V supplies.

The 13-bit data result is available on the eight outputs of the ADC1251 in two bytes, high-byte first and sign extended. The digital inputs and outputs are compatible with TTL or CMOS logic levels.

In order to generate a simple program in the DS5000 to sample data, I decided to add and modify some functions to the DS5000 basic interpreter instruction set. The first function calibrates the ADC1251 for any zero, full scale or linearity errors. The timing diagram for the Calibration cycle is shown on figure 4.15.

![Timing Diagram for the Calibration Cycle](image)

*Figure 4.15 Timing Diagram for the Calibration Cycle*

Where,

- **CLOCK**. The typical clock frequency range is 500 kHz to 6.0 MHz.
Chapter IV. Hardware Description

- **CAL.** Auto-Calibration control input. When CAL is low the ADC1251 is reset and a calibration cycle is initiated.

- **CS.** The Chip Select control input. This input is active low and enables the WR, RD and S/H functions. Since the AD1251 is the only device connected to Port 0 of the DS5000 we can leave CS selected.

- **EOC.** Output signal from the ADC1251 indicating the termination of the calibration cycle.

- **tw (cal).** Calibration Pulse Width.

- **t CAL.** Calibration time.

The duration of the auto-calibration cycle is about 1400 cycles; if we use a clock of 1 MHz, the time required for a cycle is about 14 ms. Although, it is not possible to do a calibration cycle every time a conversion is made, it is recommended to calibrate the ADC1251 before the first conversion. The flow diagram of the AUTOCAL function is shown in figure 4.16.

![Auto-calibration cycle Flow diagram](image)

**Figure 4.16 Auto-calibration cycle Flow diagram**

**AD1251**

I decided to develop a function that captures a single value from the ADC1251 storing the 13 bit result on the argument stack. The figure 4.17 shows us the timing diagram for a conversion cycle using S/H to start.
Chapter IV. Hardware Description

Figure 4.17 Timing diagram using S/H to start a conversion without Auto-zero

Where,

- **CLOCK.** The typical clock frequency range is 500 kHz to 6.0 MHz. I decided to use the timer LMC555, with the frequency set to 500 kHz. The diagram of the clock circuit is shown on figure 4.18.

Figure 4.18 Clock circuit.
Chapter IV. Hardware Description

\[ f_{\text{clock}} = \frac{1}{2 \times \pi \times R_c \times C_c} \]

- **CS.** The Chip Select control input. This input is active low and enables the WR, RD and S/H functions. Since the AD1251 is the only device connected to Port 0 of the DS5000 we can leave CS selected.

- **S/H.** The Sample and Hold control input. This control input is used to start a conversion.

- **WR.** The write control input. This control input may be used to start a conversion without sample and hold. Not used for this application.

- **RD.** The Read control input. With both CS and RD low, the tri-state output buffers are enabled and the INT output is reset high.

- **AZ.** The Auto-Zero control input. With the AZ pin held low during a conversion, the ADC1251 goes into an auto-zero cycle before the actual A/D conversion is started. This Auto-Zero cycle corrects for the comparator offset voltage. The total conversion time (tc) is increased by 26 clock periods when Auto-Zero is used.

- **EOC.** The End-of-Conversion control output. This output is low during a conversion.

- **INT.** The Interrupt control output. This output goes low when a conversion has been completed and indicates that the conversion result is available in the output latches. Reading the result or starting a conversion or calibration cycle will reset this output high.

- **DB0-DB7/DB8-DB12.** The TRI-STATE output pins. Twelve bit plus sign output data access is accomplished using two successive RDs of one byte each, high byte first (DB8-DB12). The data format used is two's complement sign bit extended with DB12 the sign bit, DB11 the MSB and DB0 the LSB.

- **tAQ** Acquisition time.

- **tOL.** Delay from Hold Command to Falling Edge of EOC.
Chapter IV. Hardware Description

- **tconv.** Conversion Time Using S/H to Start a Conversion.

- **tAcc.** Maximum Access Time. Delay from Falling Edge of RD to Output Data Valid.

- **tpd.** Maximum Delay from Falling Edge of RD or WR to Reset of INT.

- **tRR.** Delay Between Successive RD Pulses.

- **t1H+t 0L.** TRI-STATE Control. Delay from Rising Edge of RD to Output Data Valid.

Since we just have the data port of the ADC1251 connected to Port0 of the DS5000 and nothing else, CS can be selected for all time. The pins AZ and WR are connected to a high value (5 V) in order to do the conversion without Auto-zero calibration and start the conversion with the sample and hold pin respectively. The circuit diagram is shown on figure 4.18.

![Circuit Diagram](image)

*Figure 4.19 ADC1251 Interface circuit.*

From this timing diagram we can set a flow diagram to get a conversion as shown on figure 4.20:
Chapter IV. Hardware Description

Figure 4.20 Flow diagram for an A/D conversion

The Assembler code for the AD1251 conversion is shown in Appendix A.2. The result from the A/D converter comes in as a 16 bit word, where the sign is followed by the 12 bits result. Below I present a program that executes an A/D conversion presenting the result to the user.

REM Auto-calibration cycle
AUTOCAL

REM Execute an A/D conversion
AD1251

REM Get the result from the stack
POP A

REM Get LSB
LSB=A.AND.0FFH

REM Get MSB
MSB=(A.AND.0FF00H)

REM Scale MSB
MSB1=MSB/256

REM Get rid of the sign
MSB2=MSB1.AND.0FH

REM Just get the sign
SIGN=MSB1.AND.80H

REM sign flag to a 0-1 value
SIGN=SIGN/80H

REM Get the value
VALU=MSB2*256+LSB
Chapter IV. Hardware Description

110 IF SIGN=1 THEN VALU=\((-1)\)*(0FFFH-VALU) : REM inc. the sign
120 VALU=5*VALU/0FFFH : REM Scale to the -5 to +5 Range
130 PRINT "VALUE",VALU : REM Present the Result.

Then I decided to modify the capture command\(^1\) to be able to use a 12 bit Analog to Digital converter such as the National Semiconductor ADC1251. The capture command activates a background Analog-Digital conversion and event recording routine. Data is sampled every 5 milliseconds and stored in a circular queue buffer. The queue address and queue buffer size can be specified as arguments in the command line. The next figure shows us a flow diagram of the modified capture command.

![Flow Diagram of the capture command]

\textit{Figure 4.21 Simplified Flow diagram of the capture command}

\(^1\) MCS® BASIC-52 USER MANUAL, Intel Corporation, 1985
Chapter IV. Hardware Description

In order to identify the High-byte from the Low-byte an identifying byte is added to the circular buffer. Every sample three bytes are stored on the circular buffer as follow: Identifying byte- High byte - Low byte; Identifying byte- High byte - Low byte; and so on. Although the use of 24 bits to represent 13 seems inefficient, it gives us the advantage of reducing the computation time spent to identify the sample. The identifying byte may be eliminated when computation time is not an issue. The identifying byte is: 1010 1010 in binary or 170 in decimal. Then the High byte is a number smaller then 16 if is a positive number or bigger than 239 if it is a negative number. The Low byte can be any number between 0 and 255. In order to be able to plot a queue structure using a Tektronics 4010 format, I modified the TEK command. The code can be found on Appendix A.

Once the data is captured and stored in the circular buffer, it is sent to the personal computer through the serial port by using the dump command. The dump command dumps in raw binary format the contents of a queue buffer to the serial port of the DS5000. Below is a program that Calibrates the Analog to Digital converter, and then gets captures blocks of data from the Analog to Digital converter and dumps it through the serial port.

10 CLOCK 1 : REM Initialize the DS5000 internal clock
20 AUTOCAL : REM Auto-calibration cycle
30 CAPTURE 5000H,600 : REM Capture 600 samples in location5000H
40 FOR B=1 TO 4000 : REM Repeat 400 times
50 A=TIME
60 IF TIME<A+0.19 THEN GOTO 60 :REM Wait until all data captured
70 DUMP 5000H,300 : REM Dump samples through the serial port
80 NEXT B
CHAPTER V.

GRAPHIC USER INTERFACE

5.1 Visual Basic Terminal.

The user of this equipment is going to be a Physician or a nurse familiar with a regular electrocardiograph machine. The Graphic User Interface (GUI) should have Virtual knobs (controls) that perform an equivalent function to the one performed by physical knobs on mechanical electrocardiograms.

The primary function of the graphic user interface is to control and monitor the instrument status and to present data to the user. It grabs data from the serial port, process the information into graphic coordinates and to plot it on the display. The next figure shows us a general flow diagram of the GUI.

I decided to develop the Virtual Electrocardiogram machine (VEKG) under Microsoft Windows operating system (Windows 3.1, 95, NT, etc.) since is the operating system most used, and the programming tools widely available. I decided to use Visual Basic to develop the Graphic User Interface because of the applications development tools.

A feature of using Visual Basic is that you can develop a form with objects like buttons, text or graphic boxes, etc. With just a click you can modify the properties of the object like size, location, text, color, etc. This objects also respond to the mouse or keyboard, executing a program when the object is selected.

The VEKG will present an image of a traditional EKG machine with VB "controls" to show status and allow control of the machine function along with a "virtual" chart recorder which display from 1 to 5 seconds of the selected "lead". The operator will judge the quality of the data and use it to adjust the electrodes, leads, and will select representative data for inclusion in an "electrogram". The electrogram is a montage of 2 or 3 second recordings of each of the 12 standard "leads" along with a "rhythm strip" which is the basis of diagnostic analysis. The VEKG
Chapter V. Graphic User Interface

machine is superior to the traditional machine in that it can present a complete annotated electrogram on plain-paper. Figure 5.1 illustrates some of the major tasks controlled by the GUI.

As you can see on figure 5.1, the VEKG that I developed has a main form with several objects and a menu to modify some objects. The principal objects in the program are:

5.1.1 Serial Communication.

This object receives and sends data through the selected serial port with the selected settings. Depending on the computer the serial communication settings can be modified. The information that gets transferred through the serial port is Control data from the computer to the DS5000 (i.e. start capturing, change leads, modify DS5000 code), and captured data from the DS5000 to the computer.
Chapter V. Graphic User Interface

The DS5000 runs and interpretive BASIC language. The GUI sends text strings which are interpreted by BASIC. The text strings may be a BASIC program followed by the command “RUN”.

5.1.2 Display Text.

This object, if selected will show all the data that is coming from the serial port. This function is used to modify the DS5000 program or to test some functions in it. When capturing data from the DS5000, we are going to receive many data points per second, this function should be unselected in order just to see the graph corresponding the sample. By pressing the Hide button, the text form will appear and disappear from the screen.

5.1.3 Graphic Interface

The area where the signal is plotted, can be considered as another object, in which a virtual oscilloscope grid, background and foreground colors can be selected. Only a single graph can be captured at one time. For this prototype, I limit the number of graphs to 4 because of the limited space on the screen. The number and location of graphs can be modified easily in the Visual Basic software in order to accommodate the necessity of the physicians.

There are a couple of objects (virtual knobs), that modify the scale of the plot. For the vertical scale, the following selections can be made: 0.1, 0.5, 1 and 2 mV per vertical division. For the horizontal scale you always have 0.5 seconds per division, but you can select to see from 1 to 5 seconds.

The Patient Name, date and time will appear on the screen, they can be modified manually. The following diagram shows us the flow diagram of this module.
Chapter V. Graphic User Interface

Figure 5.2 Data acquisition flow diagram.
Chapter V. Graphic User Interface

The DS5000 is sampling every 5 ms, the data is stored in a circular buffer and then it is sent to the computer through the serial port. Since we are using a 13 bit analog to digital converter, two bytes are needed for each sample. I added a third one that consists of a binary number (10101010), equivalent to 170 in decimal. Each sample will have three bytes of information. The first byte is the identification byte, the second is as follows: the four most significant bits are the sign of the Analog signal (1 for negative values and 0 for positive), and the four less significant bits are the four most significant bits of the 12 bit result. The third byte corresponds to the 8 least significant bytes of the 12 bits result. In order to know which byte is which, I run the following algorithm: If the present byte value is 170 and the value of the next one is either greater than 240 or less than 16, the present byte is the identifying byte with no chance of missing. If the identifying byte is not used, more comparations have to be made in order to identify each two bytes for every sample, which means more computation time, with the possibility of failing.

The next step is to give a value for every sample. Since we know that the present value is the identifying byte, we can compare the second byte. If the second byte is greater than 240, the value corresponds to a negative value and it is given in complement 2. In this case, first we add the four most significant bits times 256 plus the 8 less significant bits. This number will be between 0 and 4096, and by subtracting this number from 4095 and multiplying it by (-1), we get a value from 0 to -4096 that corresponds to an equivalent voltage from 0 to -5. If the second byte is smaller than 16, the value corresponds to a positive number. In this case, first we just have to add the four most significant bits times 256 plus the 8 less significant bits getting a number from 0 to 4086 that corresponds a voltage from 0 to 5 volts. Finally this sample value is stored in an array that includes previous samples that is used to plot the graph and store the values in a file if requested.

The previous algorithm captures and plots up to the 5 previous seconds, without the possibility of selecting a portion of it to be transferred to the electrogram. I developed a new version of the software. This new version, presents the last 10 seconds of data on the screen. When paused, a horizontal scroll is used to select three seconds of data to be presented later on the electrogram. Several three second segments may be saved in to separate variables (in RAM memory) to be later presented on a electrogram. The lead number is also saved in order to associate a wave with the lead type. Figure 5.3 shows us an example:
Chapter V. Graphic User Interface

Figure 5.3 VEKG. Selecting 3 seconds of signal to be stored.

The gap shown in figure 5.3 is due to a change of leads. By pressing the store button the highlighted 3 seconds and lead name are stored in temporal variables (RAM memory), the number of segments that can be stored I limited to 14 (enough space to store the 12 leads plus the test and short circuit wave), but that can be increased by defining more space in memory for the variables. The highlighted bar can be moved inside the 10 seconds window by using the scroll below the screen.
5.1.4 Presenting and Storing Histograms

By selecting a wave and pressing the Electrogram button, the screen will change to present four waves in the actual graph slot selected from 1 to 4. Figure 5.4 shows us an example of an electrocardiogram as the user will see it:

Figure 5.4 VEKG. Presentation of an Electrogram.

Save

The electrogram can be saved in a file containing the name of the patient, the date and time of the recordings plus the four lead names and values for the 3 recorded seconds.
Chapter V. Graphic User Interface

Load

A stored electrogram can be loaded by pressing the load button, the histogram that was presented on the screen is erased and the new one is presented.

Print

By pressing the Print button, the wave is printed in the screen by permanent lines\(^1\). This type of lines cannot be used when presenting the wave in real time, since it takes a lot of time to draw them. Not all the information that is shown on the screen we want to print. By hiding the undesired objects while printing and unhiding them after printing, we can present an electrogram with just the required information. The next figure shows us an example of a printed electrogram:

![Electrogram Example](image)

Figure 5.5 VEKG. Example of a Printed Electrogram.

\(^1\) The lines generated by the line instruction are not printed when using the print form command. The lines, generated by assigning each line segment to a permanent object, don’t erase when printing or using the CLS command.
Chapter V. Graphic User Interface

The resolution of the printed wave is limited to the resolution of the screen, in this case 6804 by 5670 display units. Because of this, the resolution of the printed electrogram doesn't meet EC-11 standards.

Reset

By pressing the reset button, the permanent waves are erased from the screen, the number of graphs presented on the screen changes to 1 and the time to 10 seconds. This button is very helpful if the user wants to capture EKG waveforms after having showed or printed a histogram.

5.1.5 Lead Selector

The 12 leads can be amplified and be plotted, but only one at a time. As presented previously, the signal is selected using analog switches controlled through port 2 of the DS5000. In order to change of lead, the serial port should be open and the communication from the computer and microprocessor established. This communication can be verified using the text display window. Every time a lead is selected, a binary number is sent through port 2 as shown below:

<table>
<thead>
<tr>
<th>LEAD</th>
<th>PORT2 BINARY VALUE</th>
<th>PORT2 VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1000</td>
<td>8</td>
</tr>
<tr>
<td>II</td>
<td>1001</td>
<td>9</td>
</tr>
<tr>
<td>III</td>
<td>1010</td>
<td>10</td>
</tr>
<tr>
<td>aVR</td>
<td>1011</td>
<td>11</td>
</tr>
<tr>
<td>aVL</td>
<td>1100</td>
<td>12</td>
</tr>
<tr>
<td>aVF</td>
<td>1101</td>
<td>13</td>
</tr>
<tr>
<td>Vn</td>
<td>1110</td>
<td>14</td>
</tr>
<tr>
<td>TEST SIGNAL</td>
<td>1111</td>
<td>15</td>
</tr>
<tr>
<td>SHORT</td>
<td>0000</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.1 Lead Selector Controller
CHAPTER VI

CONCLUSION

In conclusion, this thesis has introduced and followed through a methodology for the design of a Virtual Electrocardiograph Machine. This design was based on a personal computer, that is in nature a lot less expensive than an equivalent acquisition and display system, a computer can be repaired everywhere, reducing maintenance and supply costs considerably.

The designed Virtual EKG machine presents a rapid-enough response (delayed about a second) to easily allow the operator to associate a change in the waveform with an intervention.

In the second chapter, I presented a background on EKG machines and the National Standards for Electrocardiographic Devices. It is very important to meet all safety standards performance requirements. In order to meet EC-11 standards, all the tests described on section 4 of this standard should be performed in a satisfactory way.

Chapter III presented us an insight of introducing a Medical Device to the Market, in particular an Electrocardiograph machine. Approval by the FDA or a similar agency in other countries, and Liability costs seems to be important factors to consider when developing and commercializing Medical equipment. Because of this, it is very important to follow the minimum safety and performance requirements for electrocardiographic (EKG) systems; and the approval time has to be taken in account in the developing/marketing plan.

Chapter IV presented us the proposed hardware form the electrodes to the computer interface. A lead selector is presented that from 5 electrodes calculate the bipolar limb leads, Augmented leads, and a unipolar chest lead. A EC11-specified triangular test signal is also incorporated, which helps us to evaluate the performance of the equipment. Some other signals may be programmed on the DS5000 ROM memory if more test signals are required. Noise and antialiasing filters were included in to order improve the signal and to prepare it to the quantization stage. A 12 bit plus sign Analog to digital converter was introduced in order to have an adequate quantization noise. Two new instructions were incorporated to the BASIC Interpreter (AUTOCAL and AD1251), that calibrates the A/D converter and executes a conversion respectively; while another two instructions were modified in order to be able to
Chapter VI. Conclusion

capture and plot samples from the ADC1251 A/D converter into a specified circular buffer (CAPTURE and TEK). There are some limitations on using the DS5000, like the communication is set to be 9600 kbps, and can’t be increased to allow more data transfer; the 5 millisecond resolution of the timer of the DS5000 is also a limitation, because the maximum sampling rate we can obtain is 200 samples per second. EC-11 standard recommends to sample at a rate higher than 200 samples per seconds, in order to not violate the 150 Hz bandwidth requirement.

Chapter V. Presented a Virtual Terminal developed using Microsoft® Visual Basic. Basically this program uses a toolbox to communicate through the serial port to the DS5000, it gets the data that is coming from the circular buffer, converts it to sample values, and plots the last 10 seconds on the screen. Three seconds of data can be selected and stored to be presented later with other stored waves in the form of electrogram. The number and position of waves conforming the electrogram can be modified as required by physicians or standard codes, by changing the software code. The display and printing of the Electrogram waves are limited to the resolution of the graphic display.

Future work can be directed to construct a power supply with limited noise, to construct a commercial model, and to incorporate different measurements to the same instrument. Increasing the resolution of the printed wave by instead of printing the screen form, printing a generated “print file” independent to the screen resolution is also necessary to be done following this work. A lead detector, which automatically determines the correct connection of the electrodes, and an algorithm to determine Heart problems by studying the waves can also be done in future work. A collaboration with a Research Laboratory in Mexico\(^1\) has been made to continue working on one of this topics or other relevant topics that are relevant to both research centers.

\(^1\) Laboratorio de Electrónica. Centro de Instrumentos. Universidad Nacional Autónoma de México.
APENDIX A.

Assembler Code for ADC1251 Extensions to Basic.

;-------------------------------------------------------;
; Define Analog-to-Digital subsystem parameters          ;
;-------------------------------------------------------;
AD_PORT EQU P0 ; Input port for A/D conversion data
AD_READ BIT P1.0 ; ADC1251 read pin (23z)
AD_BUSY BIT P1.1 ; ADC1251 status busy pin (21)
AD_WRITE BIT P1.2 ; ADC1251 S/H pin (11)
AD_EOC BIT P1.3 ; ADC1251 EOC pin (22)
AD_CAL  BIT P1.4 ; ADC1251 CAL pin (9)

;-------------------------------------------------------;
; Define tables                                        ;
;-------------------------------------------------------;
ORG 2900h ; Base token table above vector table
DB 11H,'CAPTURE',0 ; Start/Stop data capture mode
DB 1EH,'AD1251',0 ; Capture AD670 sample (single)
DB 1FH,'AUTOCAL',0 ; Auto-calibration cycle

;-------------------------------------------------------;
; Define the action routine table                      ;
;-------------------------------------------------------;
DW CAPTURE ; Data capturing start/stop
DW AD1251 ; Capture ADC1251 sample (single)
DW AUTOCAL ; Auto-calibration cycle

$TITLE(AUTOCAL A/D Calibration)
$EJECT

;-------------------------------------------------------;
; AUTOCAL A/D Calibration                              ;
;-------------------------------------------------------;
Syntax: AUTOCAL                                       ;
; This function adjust the ADC1251 analog to            ;
; digital converter for any zero, full scale,          ;
; or linearity errors                                   ;
Appendix A. Assembler Code for ADC1251 Extensions to Basic

;---------------------------------------------------------------;
; Start the calibration                                        ;
;---------------------------------------------------------------;
AUTOCAL: CLR AD_CAL ; Initiate calibration
SETB AD_CAL
;---------------------------------------------------------------;
; Save the time associated with the capture event              ;
;---------------------------------------------------------------;
MOV C,EA ; Save interrupt status
CLR EA ; Inhibit timer updates...
MOV TIME_CMS, IDR?M_TIMER ; Save MS timer value
MOV TIME_CLO, IDR?L_TIMER ; Save LO timer value
MOV TIME_CHI, IDR?H_TIMER ; Save HI timer value
MOV EA,C ; Restore timer update status
;---------------------------------------------------------------;
; Wait for the calibration to complete (1399 clocks)           ;
;---------------------------------------------------------------;
JB AD_EOC,$ ; Wait for A/D calibration completion
;---------------------------------------------------------------;
RET ; Return to caller
;---------------------------------------------------------------;
; AD1251 Data Capture Command                                 ;
;---------------------------------------------------------------;
; Syntax: AD1251 ; This function samples the ADC1251 analog to ;
; digital converter and places the resulting sample value on the argument stack. ;
;---------------------------------------------------------------;
AD1251: SETB AD_WRITE ; Initiate conversion
CLR AD_WRITE ; Setup for read of resulting data
SETB AD_READ
;---------------------------------------------------------------;
; Save the time associated with the capture event              ;
;---------------------------------------------------------------;
MOV C,EA ; Save interrupt status
CLR EA ; Inhibit timer updates...
MOV TIME_CMS, IDR?M_TIMER ; Save MS timer value
MOV TIME_CLO, IDR?L_TIMER ; Save LO timer value
MOV TIME_CHI, IDR?H_TIMER ; Save HI timer value
MOV EA,C ; Restore timer update status
;---------------------------------------------------------------;
; Wait for the conversion to complete (~10 microseconds)       ;
;---------------------------------------------------------------;
SETB AD_WRITE
Appendix A. Assembler Code for ADC1251 Extensions to Basic

```assembly
JB AD_BUSY,$ ; Wait for A/D conversion completion

; Sample the digitized result

CLR AD_READ
MOV A,AD_PORT ; Get the digitized result
SETB AD_READ
MOV R2,A
CLR AD_READ

MOV A,AD_PORT ; Get the digitized result
SETB AD_READ
MOV R0,A

; Convert the binary result to floating point
; and place on stack

%BASIC (OPC?PUSH_FIXED) ; Push R2:R0 onto the argument stack

; Return to caller

; Return to caller

$TITLE(Capture Queue Initialization Command)
$EJECT

Capture Queue Initialization Command

This action routine sets up a circular queue in data memory and intercepts the Timer 0 interrupt service routine to force an A/D conversion every 5 milliseconds. This provides approximately 200 samples per second.

Syntax: CAPTURE [queue_address], [count]

Determine if there are any parameters on the command line

CAPTURE:SETB IDR?B_CLOCK ; Inhibit user CLOCK ISR service
SETB IDR?B_INT_SET ; Inhibit user INT1 processing
CLR EX1 ; Disable interrupts on EX1

%BASIC (OPC?CUR_CHAR) ; Load function - current character
CJNE A,#0Dh,CAP?010 ; Skip if parameters are present
SJMP CAP?090 ; Return to caller
```

70
Appendix A. Assembler Code for ADC1251 Extensions to Basic

; Fetch the queue address parameter

;-------------------------------------------------------------;
CAP?010:*BASIC (OPC?EVALUATE); Load function - get argument
%BASIC (OPC?POP_FIXED) ; Load function - pop stack to R3:R1
%BASIC (OPC?GET CHAR) ; Load function - get character
;-------------------------------------------------------------;
; Initialize the current queue pointer
;-------------------------------------------------------------;
MOV QUEUE_DPL,R1 ; Load queue pointer (lo)
MOV QUEUE_DPH,R3 ; Load queue pointer (hi)
;-------------------------------------------------------------;
; Fetch the queue byte count parameter
;-------------------------------------------------------------;
%BASIC (OPC?EVALUATE) ; Load function - get argument
%BASIC (OPC?POP_FIXED) ; Load function - pop stack to R3:R1
$EJECT
;-------------------------------------------------------------;
; Clear the circular queue buffer
;-------------------------------------------------------------;
MOV R0B0,R1 ; Load queue length (lo)
MOV R2B0,R3 ; Load queue length (hi)
CJNE R2,#0,CAP?020 ; Skip if queue size is valid (hi)
CJNE R0,#0,CAP?020 ; Skip if queue size is valid (lo)
SJMP CAP?090 ; Invalid queue size - do nothing
CAP?020:MOV DPL,QUEUE_DPL ; Get the queue base address (lo)
MOV DPH,QUEUE_DPH ; Get the queue base address (hi)
CLR A ; Get a zero
INC R2 ; Bias page counter
CAP?030:MOVX @DPTR,A ; Clear the queue location
INC DPTR ; ... and point to next location
DJNZ R0,CAP?030 ; Continue if byte count not exhausted
DJNZ R2,CAP?030 ; Continue if page count not exhausted
;-------------------------------------------------------------;
; Load the first two bytes with the queue size
;-------------------------------------------------------------;
MOV DPL,QUEUE_DPL ; Load the queue header value (lo)
MOV DPH,QUEUE_DPH ; Load the queue header value (hi)
MOV A,R1 ; Get queue length (lo)
MOVX @DPTR,A ; Save the queue length (lo)
INC DPTR ; Point to the next control byte
MOV A,R3 ; Get queue length (hi)
MOVX @DPTR,A ; Save the queue length (hi)
INC DPTR ; Point to the next control byte
;-------------------------------------------------------------;
; Load the next two bytes with the initial queue pointer
;-------------------------------------------------------------;
MOV A,#4 ; Load initial queue index (lo)
MOV QUEUE_SPL,A ; Mark the queue index (lo)
MOVX @DPTR,A ; Save the queue index (lo)
Appendix A. Assembler Code for ADC1251 Extensions to Basic

INC DPTR ; Point to the next control byte
CLR A ; Load initial queue index (hi)
MOV QUEUE_SPH,A ; Mark the queue index (hi)
MO VX @DPTR,A ; Save the queue index (hi)
INC DPTR ; Point to the next control byte

; Enable user-defined CLOCK service
CLR IDR?B_INT_SET ; Enable user INT1 processing
SETB EX1 ; Enable interrupts on EX1
CLR IDR?B_CLOCK ; Toggle user CLOCK ISR service
ANL TMOD,#OFOh ; Set up the mode
ORL IE,#82h ; Enable ET0 and EA
SETB TR0 ; Turn on the timer

; Return to caller
CAP?090:RET ; Return to caller

$TITLE(Timer 0 interrupt service routine)
$EJECT

; Timer 0 interrupt service routine
;
; This routine services the timer 0 interrupt and performs
; an A/D conversion, placing the resulting data value in
; the current queue entry pointed to by the circular buffer
;
; The queue structure used by this routine starts at the
; address specified by the user which is stored in ram
; location QUEUE_DPx. This points to a queue head
; which contains 4 bytes. The first two bytes are a queue
; pointer to the current byte in the queue (the next one to
; be written, or the first one to be read). The next two
; bytes point to one byte beyond the end of the queue.
; These two bytes are used to determine when queue
; wraparound occurs. The user specifies the queue size in
; bytes. This number includes the four byte queue header
; mentioned here! This means that for a given queue
; size of N bytes, there is actually N-4 real data bytes.
;
; Start the conversion and put the A/D converter in
; READ mode.
;
T0_ISR: SETB AD_WRITE ; Initiate conversion
CLR AD_WRITE ; Setup for read of resulting data
SETB AD_WRITE

; Do some housekeeping during the A/D conversion


Appendix A. Assembler Code for ADC1251 Extensions to Basic

; Since the A/D conversion may consume about 10us, we
; have time to do some housekeeping chores, so that this
; overhead can be overlapped with the external A/D
; conversion process.
;
PUSH ACC          ; Save the accumulator
PUSH B            ; Save the B register
PUSH DPL          ; Save the data pointer
PUSH DPH
MOV A,R0          ; Get current R0
PUSH ACC          ; ... and save it
MOV A,R1          ; Get current R1
PUSH ACC          ; ... and save it

$EJECT

; Calculate the end-of-queue marker and put save it in
; R1:R0
;
MOV DPL,QUEUE_DPL ; Restore queue header pointer (lo)
MOV DPH,QUEUE_DPH ; Restore queue header pointer (hi)
MOVX A,@DPTR ; Get the end queue pointer (lo)
INC DPTR ; Point to the next byte
ADD A,QUEUE_DPL ; Calculate end-of-queue pointer (lo)
MOV R0,A ; ... and save it locally
MOVX A,@DPTR ; Get the end queue pointer (hi)
INC DPTR ; Point to the next byte
ADDC A,QUEUE_DPH ; Calculate end-of-queue pointer (hi)
MOV R1,A ; ... and save it locally

; Fetch the queue pointer and put it on the stack
;
MOVX A,@DPTR ; Get the current queue pointer (lo)
INC DPTR ; Point to the next byte
ADD A,QUEUE_DPL ; Point to the current byte (lo)
PUSH ACC ; ... and save it locally
MOVX A,@DPTR ; Get the current queue pointer (hi)
INC DPTR ; Point to the next byte
ADDC A,QUEUE_DPH ; Point to the current byte (hi)

; Point to the current queue byte
;
MOV DPH,A ; Restore queue pointer (hi)
POP DPL ; Restore queue pointer (lo)

; Save the time associated with the capture event
;
MOV C,EA          ; Save interrupt status
CLR EA            ; Inhibit timer updates...
MOV TIME_CMS,IDR?M_TIMER ; Save MS timer value
MOV TIME_CLO,IDR?L_TIMER ; Save LO timer value
MOV TIME_CHI,IDR?H_TIMER ; Save HI timer value
MOV EA,C          ; Restore timer update status
Appendix A. Assembler Code for ADC1251 Extensions to Basic

; Store the index to indicate Next is MSB
;
MOV A, #0AAh ; Load 10101010 word
MOVX @DPTR, A ; ... and save it in RAM
;
; Update the circular buffer pointer
;
INC DPTR ; Update the pointer
MOV A, DPL ; Get low queue pointer
MOV B, R0 ; Get value for testing
CJNE A, B, TO?010 ; Skip if not at queue end
MOV A, DPH ; Get hi queue pointer
MOV B, R1 ; Get value for testing
CJNE A, B, TO?010 ; Skip if not at queue end
;
; Reset the queue index to 4
;
MOV DPL, QUEUEDPL ; Get queue header pointer (lo)
MOV DPH, QUEUEDPH ; Get queue header pointer (hi)
INC DPTR ; Point past the control area
INC DPTR ;
MOV A, #4 ; Set queue index to start (lo)
MOVX @DPTR, A ; Write queue index (lo)
INC DPTR ; Point to the next byte
CLR A ; Set queue index to start (hi)
MOVX @DPTR, A ; Write queue index (hi)
SJMP TO?020 ; Return to caller
;
; Simply increment the queue index
;
T0?010: MOV DPL, QUEUEDPL ; Restore queue header pointer (lo)
MOV DPH, QUEUEDPH ; Restore queue header pointer (hi)
INC DPTR ; Point to the queue index
INC DPTR ;
MOVX A, @DPTR ; Fetch the queue index (lo)
ADD A, #1 ; Update the queue index (lo)
MOVX @DPTR, A ; Restore queue index value (lo)
INC DPTR ; Point to the next byte
MOVX A, @DPTR ; Fetch the queue index (hi)
ADDC A, #0 ; Propagate the carry
MOVX @DPTR, A ; Restore queue index value (hi)
SJMP T0?020

$EJECT

; Calculate the end-of-queue marker and put save it in
;
R1: R0
;
T0?020: MOV DPL, QUEUEDPL ; Restore queue header pointer (lo)
MOV DPH, QUEUEDPH ; Restore queue header pointer (hi)
MOVX A, @DPTR ; Get the end queue pointer (lo)
INC DPTR ; Point to the next byte
Appendix A. Assembler Code for ADC1251 Extensions to Basic

ADD A, QUEUE_DPL ; Calculate end-of-queue pointer (lo)
MOV R0, A ; ... and save it locally
MOVX A, @DPTR ; Get the end queue pointer (hi)
INC DPTR ; Point to the next byte
ADDC A, QUEUE_DPH ; Calculate end-of-queue pointer (hi)
MOV R1, A ; ... and save it locally

; Fetch the queue pointer and put it on the stack ;
MOVX A, @DPTR ; Get the end queue pointer (hi)
INC DPTR ; Point to the next byte
ADDC A, QUEUE_DPH ; Calculate end-of-queue pointer (hi)
MOV R1, A ; ... and save it locally

; Point to the current queue byte ;
MOV DPH, A ; Restore queue pointer (hi)
POP DPL ; Restore queue pointer (lo)

; Save the time associated with the capture event ;
MOV C, EA ; Save interrupt status
CLR EA ; Inhibit timer updates...
MOV TIME_CMS, IDR?M_TIMER ; Save MS timer value
MOV TIME_CLO, IDR?L_TIMER ; Save LO timer value
MOV TIME_CHI, IDR?H_TIMER ; Save HI timer value
MOV EA, C ; Restore timer update status

; Wait for the A/D conversion to complete ; (~10 microseconds) ;
SETB AD_WRITE
JB AD_BUSY, $ ; Wait for A/D conversion completion

; Store the MSB ;
CLR AD_READ
MOV A, AD_PORT ; Get the digitized result
SETB AD_READ
MOVX @DPTR, A ; ... and save it in RAM

; Update the circular buffer pointer ;
INC DPTR ; Update the pointer
MOV A, DPL ; Get low queue pointer
MOV B, R0 ; Get value for testing
CJNE A, B, TO?050 ; Skip if not at queue end
MOV A, DPH ; Get hi queue pointer

75
Appendix A. Assembler Code for ADC1251 Extensions to Basic

MOV B,R1 ; Get value for testing
CJNE A,B,T0?050 ; Skip if not at queue end

; Reset the queue index to 4
MOV DPL,QUEUE_DPL ; Get queue header pointer (lo)
MOV DPH,QUEUE_DPH ; Get queue header pointer (hi)
INC DPTR ; Point past the control area
INC DPTR
MOV A,#4 ; Set queue index to start (lo)
MO VX @DPTR,A ; Write queue index (lo)
INC DPTR ; Point to the next byte
CLR A ; Set queue index to start (hi)
MOVX @DPTR,A ; Write queue index (hi)
SJMP T0?060 ; Return to caller

; Simply increment the queue index

T0?050: MOV DPL,QUEUE_DPL ; Restore queue header pointer (lo)
MOV DPH,QUEUE_DPH ; Restore queue header pointer (hi)
INC DPTR ; Point to the queue index
INC DPTR
MO VX A,@DPTR ; Fetch the queue index (lo)
ADD A,#1 ; Update the queue index (lo)
MO VX @DPTR,A ; Restore queue index value (lo)
INC DPTR ; Point to the next byte
MO VX A,@DPTR ; Fetch the queue index (hi)
ADD C A,#0 ; Propagate the carry
MO VX @DPTR,A ; Restore queue index value (hi)
SJMP T0?060

$EJECT

; Calculate the end-of-queue marker and put save it in R1:R0

T0?060: MOV DPL,QUEUE_DPL ; Restore queue header pointer (lo)
MOV DPH,QUEUE_DPH ; Restore queue header pointer (hi)
MOVX A,@DPTR ; Get the end queue pointer (lo)
INC DPTR ; Point to the next byte
ADD A,QUEUE_DPL ; Calculate end-of-queue pointer (lo)
MOV R0,A ; ... and save it locally
MOVX A,@DPTR ; Get the end queue pointer (hi)
INC DPTR ; Point to the next byte
ADDC A,QUEUE_DPH ; Calculate end-of-queue pointer (hi)
MOV R1,A ; ... and save it locally

; Fetch the queue pointer and put it on the stack

MO VX A,@DPTR ; Get the current queue pointer (lo)
INC DPTR ; Point to the next byte
ADD A,QUEUE_DPL ; Point to the current byte (lo)
PUSH ACC ; ... and save it locally
Appendix A. Assembler Code for ADC1251 Extensions to Basic

MOVX A,@DPTR ; Get the current queue pointer (hi)
INC DPTR ; Point to the next byte
ADDC A,QUEUE_DPH ; Point to the current byte (hi)

MOV DPH,A ; Restore queue pointer (hi)
POP DPL ; Restore queue pointer (lo)

MOV C,EA ; Save interrupt status
CLR EA ; Inhibit timer updates...
MOV TIME_CMS,?M_TIMER ; Save MS timer value
MOV TIME_CLO,?L_TIMER ; Save LO timer value
MOV TIME_CHI,?H_TIMER ; Save HI timer value
MOV EA,C ; Restore timer update status

CLR AD_READ ; Inhibit timer updates...
MOV A,AD_PORT ; Get the digitized result
SETB AD_READ ; Restore queue header pointer (hi)
MOVX @DPTR,A ; ... and save it in RAM

MOV A,#4 ; Set queue index to start (lo)
MOVX @DPTR,A ; Write queue index (lo)
INC DPTR ; Point to the next byte
INC DPTR ; Point past the control area
MOV A,#4 ; Set queue index to start (lo)
MOVX @DPTR,A ; Write queue index (hi)
CLR DPTR ; Return to caller

INC DPTR ; Update the pointer
MOV A,DPL ; Get low queue pointer
MOV B,R0 ; Get value for testing
CJNE A,B,T0?070 ; Skip if not at queue end
MOV A,DPH ; Get hi queue pointer
MOV B,R1 ; Get value for testing
CJNE A,B,T0?070 ; Skip if not at queue end

MOV DPL,QUEUE_DPL ; Get queue header pointer (lo)
MOV DPH,QUEUE_DPH ; Get queue header pointer (hi)
INC DPTR ; Point past the control area
INC DPTR ;
MOV A,#4 ; Set queue index to start (lo)
MOVX @DPTR,A ; Write queue index (lo)
INC DPTR ; Point to the next byte
CLR A ; Set queue index to start (hi)
MOVX @DPTR,A ; Write queue index (hi)
SJMP T0?080 ; Return to caller

Simply increment the queue index

T0?070: MOV DPL,QUEUE_DPL ; Restore queue header pointer (lo)
Appendix A. Assembler Code for ADC1251 Extensions to Basic

```
MOV DPH, QUEUE_DPH ; Restore queue header pointer (hi)
INC DPTR ; Point to the queue index
INC DPTR ;
MOVX A, @DPTR ; Fetch the queue index (lo)
ADD A, #1 ; Update the queue index (lo)
MOVX @DPTR, A ; Restore queue index value (lo)
INC DPTR ; Point to the next byte
MOVX A, @DPTR ; Fetch the queue index (hi)
ADDC A, #0 ; Propagate the carry
MOVX @DPTR, A ; Restore queue index value (hi)
SJMP TO?080
```

$EJECT

```
; Restore registers
;
;
TO?080: POP ACC ; Restore R1
MOV R1, A ; ... and load it
POP ACC ; Restore R0
MOV R0, A ; ... and load it
POP DPH ; Restore the data pointer
POP DPL ;
POP B ; Restore B
POP ACC ; Restore the accumulator
;
;
LJMP ISR?INT_TIMER ; Continue with BASIC timer support
```

$TITLE(TEKtronix Plot Statement)
$EJECT

```
; Tektronix Plot Statement
;
;
Syntax: TEK [Q_address], [count]
;
This command routine takes the data contained in the queue structure and plots it using Tektronix 4010 primitives. The 8-bit queue data is transmitted as the Y-coordinate (unscaled), and the X-coordinate is derived from the queue index into the queue structure.
;
Initialize the plotting interface
;
TEK: ACALL TEK?INI ; Initialize plot
```
Appendix A. Assembler Code for ADC1251 Extensions to Basic

; Fetch the queue address and byte count parameters ;

%BASIC (OPC?EVALUATE) ; Load function - get argument
%BASIC (OPC?POPFIXED) ; Load function - pop stack to R3:R1
PUSH R1B0 ; Save queue (lo)
PUSH R3B0 ; Save queue (hi)
%BASIC (OPC?GET CHAR) ; Load function - get character
%BASIC (OPC?EVALUATE) ; Load function - get argument
%BASIC (OPC?POPFIXED) ; Load function - pop stack to R3:R1

; Load the queue end count index pointer ;

POP R2B0 ; Get head of queue pointer (hi)
POP R0B0 ; Get head of queue pointer (lo)
MOV DPL,R0 ; Set DPTR to queue head (lo)
MOV DPH,R2 ; Set DPTR to queue head (hi)
MOVX A,@DPTR ; Fetch queue end pointer (lo)
INC DPTR ; Point to the next byte
ADD A,R0 ; Add base address of queue (lo)
MOV R6,A ; Save the queue end pointer
MOVX A,@DPTR ; Fetch queue end pointer (hi)
INC DPTR ; Point to the next byte
ADDC A,R2 ; Add base address of queue (hi)
MOV R7,A ; Save the queue end pointer

$EJECT

; Load the current queue pointer ;

MOVX A,@DPTR ; Fetch queue pointer (lo)
INC DPTR ; Point to the next byte
ADD A,R0 ; Add queue address offset
MOV R4,A ; ... and save it locally
MOVX A,@DPTR ; Fetch queue pointer (hi)
INC DPTR ; Point to the next byte
ADDC A,R2 ; Add queue address offset
MOV R5,A ; ... and save it locally

; Save a pointer to the first byte in the queue data area ;

MOV R0,DPL ; Save first data byte pointer (lo)
MOV R2,DPH ; Save first data byte pointer (hi)

$EJECT

; Transfer data from the queue to the serial port ;

CJNE R3,#O,TEK?010 ; Check for nonzero byte count (hi)
CJNE R1,#O,TEK?010 ; Check for nonzero byte count (lo)
SJMP TEK?050 ; Zero byte transfer - abort
TEK?010:MOV DPL,R4 ; Get source address (lo)
MOV DPH,R5 ; Get source address (hi)
MOV R4,#0 ; Clear X-coordinate (lo)
MOV R5,#0 ; Clear X-coordinate (hi)
Appendix A. Assembler Code for ADC1251 Extensions to Basic

; Plot the data point
;---------------------------------------------------------------------
INC R3 ; Bias page count

TEK?020: MOVX A,@DPTR ; Get source byte ('start byte')
INC DPTR
MOVX A,@DPTR ; Get source byte (high)
ANL A,#0Fh ; Keep only low order 4 bits
MOV B,A
INC DPTR
MOVX A,@DPTR ; Get the source byte (low)
PUSH DPL ; Save the current data pointer (lo)
PUSH DPH ; Save the current data pointer (hi)
MOV DPL,R4 ; Load Y-coordinate (lo)
MOV DPH,R5 ; Load Y-coordinate (hi)
RR A ; Eliminate low order 2 bits
RR A
ANL A,#3FH ; Move from High order byte the
PUSH R1B0 ; two low order bits
MOV C,B.0 ; to get a 10 bit word from 12
MOV ACC.6,C
MOV C,B.1
MOV ACC.7,C
ORL A,ACC
MOV R1,A
MOV A,B
RR A
RR A
MOV B,A
MOV A,R1
POP R1B0 ; Restore R1
ACALL TEK?ORG ; Plot the data point
INC DPTR ; Increment the Y-coordinate
MOV R4,DPL ; Update Y-coordinate (lo)
MOV R5,DPH ; Update Y-coordinate (hi)
POP DPH ; Restore current data pointer (hi)
POP DPL ; Restore current data pointer (lo)
INC DPTR ; Update source address
MOV A,R7 ; Get the upper queue bound (hi)
CJNE A,DPH,TEK?040 ; Skip if bound not reached
MOV A,R6 ; Get the upper queue bound (lo)
CJNE A,DPL,TEK?040 ; Skip if bound not reached
MOV DPL,R0 ; Reload the queue pointer (lo)
MOV DPH,R2 ; Reload the queue pointer (hi)
DEC B ; Decrement byte count
TEK?040:DJNZ R1,TEK?020 ; Continue if byte count not exhausted
DJNZ R3,TEK?020 ; Continue if page count not exhausted
;---------------------------------------------------------------------
; Return to caller
;---------------------------------------------------------------------
TEK?050:RET ; Return to caller
APENDIX B.

Visual Basic Code for the Virtual EKG Machine.

'--------------------------------------------------------
' VBTerm - This is a demonstration program for the Virtual EKG machine.
' Copyright (c) 1998, Massachusetts Institute of Technology. 
' by Luis Parra.
'--------------------------------------------------------
DefInt A-Z
Option Explicit
Dim Ret ' Scratch integer.
Dim temp$ ' Scratch string.
Dim hLogFile ' Handle of open log file.
Dim Startxx, StartX, StartY,
Func, Angle, pi, Xcoord(3001),
Xcoord_ant, Ycoord(3001),
Ycoord_ant, yorigin, xorigin,
xmax, ymax, ygraph, xgraph, J,
JUMP, k, colr, Value(3000),
GraphValue(3000),
DrawGraphValue(5, 3000), Tempo$,
Dta2$, Dta3$, Prevcount, Xscale,
PauseTime, Start, Count_graph,
Graph_pause, samp_cm, Lead_num,
Stored_data(15, 1000),
Stored_Lead$(15)

' The next routine changes the graph to be plotted in the electrogram.
Private Sub ActualGraph_Change()
  ActualGraph = Int(ActualGraph)
  If ActualGraph < 1 Then
    ActualGraph = 1
  If ActualGraph > Numgraph Then
    ActualGraph = Numgraph
  End Sub

' The next routine shows and hides the text display window
Private Sub Command1_Click()
  If (Term.Visible = True) Then
    Term.Visible = False
  Else
    Term.Visible = True
  End If
End Sub

' The next routine decrements the selector of leads, and sends the command to the DS5000 to do so.
Private Sub Command10_Click()
  Select Case LeadLabel
  Case "II"
    LeadLabel = "I"
    Lead_num = 1
    If MSComm1.PortOpen Then
      MSComm1.Output = "Port2=8"
      MSComm1.Output = Chr$(13)
      End If
  Case "III"
    LeadLabel = "II"
    Lead_num = 2
    If MSComm1.PortOpen Then
      MSComm1.Output = "Port2=9"
      MSComm1.Output = Chr$(13)
      End If
  Case "aVR"
    LeadLabel = "III"
    Lead_num = 3
    If MSComm1.PortOpen Then
      MSComm1.Output = "Port2=10"
      MSComm1.Output = Chr$(13)
      End If
  Case "aVL"
    LeadLabel = "aVR"
    Lead_num = 4
    If MSComm1.PortOpen Then
      MSComm1.Output = "Port2=11"
      MSComm1.Output = Chr$(13)
      End If
  Case "aVF"
    LeadLabel = "aVL"

Form Unload Ret
End Sub
Appendix B. Visual Basic Code for the Virtual EKG Machine

Lead num = 5
If MSComm1.PortOpen Then
    MSComm1.Output = "Port2=12"
End If
Case "Vn"
    LeadLabel = "aVF"
    Lead num = 6
    If MSComm1.PortOpen Then
        MSComm1.Output = "Port2=13"
    End If
End If
Case "Test"
    LeadLabel = "Vn"
    Lead num = 7
    If MSComm1.PortOpen Then
        MSComm1.Output = "Port2=14"
    End If
End If
Case "Short"
    LeadLabel = "Test"
    Lead num = 8
    If MSComm1.PortOpen Then
        MSComm1.Output = "Port2=0"
    End If
End If
Case "I"
    LeadLabel = "Short"
    Lead num = 9
    If MSComm1.PortOpen Then
        MSComm1.Output = "Port2=15"
    End If
End If
End Select
End Sub

Private Sub Command11_Click()
    Dim I, J, k
    Numgraph = 1
    ActualGraph = 1
    mm_sec = 10
    Day = Date
    Time now = Time
    StartX = 120
    StartY = 120
    DrawWidth = 1
    xmax = 6804 '10 cm
    ymax = 5670 / 2 '5 cm
    yorigin = ymax + StartY / 2
    xorigin = StartX
    For I = 0 To 200 * mm_sec Step 1
        Xcoord(I) = xorigin + (xmax / ((mm_sec) * 200)) * (I)
    Next I
    For I = 0 To 8
        Line2(I).Visible = True
    Next I
    'Vertical grid
    If mm_sec > 1 Then
        For I = 0 To mm_sec - 1
            Line3(I).Visible = True
            Line3(I).X1 = 120 + (6804 / (mm_sec)) * (I + 1)
            Line3(I).X2 = 120 + (6804 / (mm_sec)) * (I + 1)
            Line3(I).Y1 = 120
            Line3(I).Y2 = 120 + (5670)
        Next I
    End If
    'Horizontal grid
    For k = 1 To 4
        Line_x_w(k).Visible = False
        For I = 0 To 10 Step 1
            Line_x(10 * (k - 1) + I).Visible = False
        Next I
    Next k
    Next I
    Next k
    For I = 2 To (600) Step 1
        Line_plot_1(I).Visible = False
        Line_plot_2(I).Visible = False
        Line_plot_3(I).Visible = False
        Line_plot_4(I).Visible = False
    Next I
Appendix B. Visual Basic Code for the Virtual EKG Machine

Next I
For k = 1 To 4
    Lead_num_text_1(k).Visible = False
Next k
Cls
'HORIZONTAL GRID
For k = 1 To Numgraph
    For I = 0 To 10 Step 1
        If I = 0 Then colr = 4: DrawWidth = 3 Else: If I = 5 Then colr = 14: DrawWidth = 1 Else: colr = 4: DrawWidth = 1
        Line (xorigin, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph) - (xorigin + xmax, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph), QBColor(colr)
    Next I
Next k
End Sub

' The next routine opens the comm port for communication
Private Sub Command3_Click()
    If MSComml.PortOpen Then
        MSComml.Output = Chr$(3)
    End If
End Sub

Increment the number of graphs to plot in the electrogram, up to 4
Private Sub Command4_Click()
    Dim I, k
    Cls
    StartX = 120
    StartY = 120
    DrawWidth = 1
    xmax = 6804 '10 cm
    ymax = 5670 / 2 '5 cm
    yorigin = ymax + StartY / 2
    xorigin = StartX
    Numgraph = Numgraph + 1
    If Numgraph > 4 Then Numgraph = 4
    If Numgraph > 1 Then
        For I = 0 To 8
            Line2(I).Visible = False
        Next I
    End If
    'HORIZONTAL GRID
    For k = 1 To Numgraph
        For I = 0 To 10 Step 1
            If I = 0 Then colr = 4: DrawWidth = 3 Else: If I = 5 Then colr = 14: DrawWidth = 1 Else: colr = 4: DrawWidth = 1
            Line (xorigin, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph) - (xorigin + xmax, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph), QBColor(colr)
        Next I
    Next k
End Sub

Next I
For k = 1 To 4
    Lead_num_text_1(k).Visible = False
Next k
Cls
'Horizontal grid
For k = 1 To Numgraph
    For I = 0 To 10 Step 1
        If I = 0 Then colr = 4: DrawWidth = 3 Else: If I = 5 Then colr = 14: DrawWidth = 1 Else: colr = 4: DrawWidth = 1
        Line (xorigin, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph) - (xorigin + xmax, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph), QBColor(colr)
    Next I
Next k
End Sub

If Command2.Caption = "Run" Then
    If MSComml.PortOpen Then
        MSComml.Output = Chr$(3)
    End If
    Command2.Caption = "Pause"
Else
    If MSComml.PortOpen Then
        MSComml.Output = Chr$(13)
    End If
    Command2.Caption = "Run"
End If
End Sub

Private Sub Command2_Click()
    Dim I
    If Command2.Caption = "Run" Then
        If MSComml.PortOpen Then
            Day = Date
            Time_now = Time
            StartX = 120
            StartY = 120
            DrawWidth = 1
            xmax = 6804 '10 cm
            ymax = 5670 / 2 '5 cm
            yorigin = ymax + StartY / 2
            xorigin = StartX
            Numgraph = Numgraph + 1
            If Numgraph > 4 Then Numgraph = 4
            If Numgraph > 1 Then
                For I = 0 To 8
                    Line2(I).Visible = False
                Next I
            End If
            'Horizontal grid
            For k = 1 To Numgraph
                For I = 0 To 10 Step 1
                    If I = 0 Then colr = 4: DrawWidth = 3 Else: If I = 5 Then colr = 14: DrawWidth = 1 Else: colr = 4: DrawWidth = 1
                    Line (xorigin, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph) - (xorigin + xmax, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph), QBColor(colr)
                Next I
            Next k
        End If
    End If
Else
    If MSComml.PortOpen Then
        MSComml.Output = Chr$(3)
    End If
    Command2.Caption = "Run"
End If
End Sub
Appendix B. Visual Basic Code for the Virtual EKG Machine

For J = 1 To 1000
    Ycoord(J) = StartY + (2 * ymax * (ActualGraph / Numgraph)) - (2 * (ymax) * (128 / (255 * Numgraph)))
Next J
End Sub

' Decrements the number of graphs to plot in the electrogram
Private Sub Command5_Click()
    Dim I, k
   Cls
   StartX = 120
   StartY = 120
   DrawWidth = 1
   xmax = 6804 '10 cm
   ymax = 5670 / 2 '5 cm
   yorigin = ymax + StartY / 2
   xorigin = StartX
    'Horizontal grid
    For k = 1 To Numgraph
        For I = 0 To 10 Step 1
            If I = 0 Then colr = 4: DrawWidth = 3 Else: If I = 5 Then colr = 14: DrawWidth = 1 Else: colr = 4: DrawWidth = 1
            Line (xorigin, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph)-(xorigin + xmax, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph), QBColor(colr)
        Next I
    Next k
    Numgraph = Numgraph - 1
    If Numgraph = 0 Then Numgraph = 1
    If ActualGraph > Numgraph Then
        ActualGraph = Numgraph
        For J = 1 To 1000
            Ycoord(J) = StartY + (2 * ymax * (ActualGraph / Numgraph)) - (2 * (ymax) * (128 / (255 * Numgraph)))
        Next J
    End Sub

If ActualGraph < 1 Then
    ActualGraph = 1
    For J = 1 To 1000
        Ycoord(J) = StartY + (2 * ymax * (ActualGraph / Numgraph)) - (2 * (ymax) * (128 / (255 * Numgraph)))
    Next J
End Sub

' Increment the position of the actual graph to be plotted
Private Sub Command7_Click()
    Dim J
    ActualGraph = ActualGraph + 1
    If ActualGraph > Numgraph Then
        ActualGraph = Numgraph
        For J = 1 To 1000
            Ycoord(J) = StartY + (2 * ymax * (ActualGraph / Numgraph)) - (2 * (ymax) * (128 / (255 * Numgraph)))
        Next J
    End Sub

' The next routine decrements the selector of leads, and sends the command to the DS5000 to do so.
Private Sub Command9_Click()
    Select Case LeadLabel
        Case "I"
            LeadLabel = "II"
            Lead_num = 2
            If MSComml.PortOpen Then
                MSComml.Output = "Port2=9"
                MSComml.Output = Chr$(13)
            End If
        Case "II"
            LeadLabel = "III"
            Lead_num = 3
            If MSComml.PortOpen Then
                MSComml.Output = "Port2=10"
                MSComml.Output = Chr$(13)
            End If
        Case "III"
            LeadLabel = "aVR"
            Lead_num = 4
            If MSComml.PortOpen Then
                MSComml.Output = "Port2=11"
                MSComml.Output = Chr$(13)
            End If
    End Select
End Sub
Appendix B. Visual Basic Code for the Virtual EKG Machine

Case "aVR"
  LeadLabel = "aVL"
  Lead_num = 5
  If MSComm1.PortOpen Then
    MSComm1.Output = "Port2=12"
    MSComm1.Output = Chr$(13)
  End If
End Select

Case "aVL"
  LeadLabel = "aVF"
  Lead_num = 6
  If MSComm1.PortOpen Then
    MSComm1.Output = "Port2=13"
    MSComm1.Output = Chr$(13)
  End If
End Select

Case "aVF"
  LeadLabel = "Vn"
  Lead_num = 7
  If MSComm1.PortOpen Then
    MSComm1.Output = "Port2=14"
    MSComm1.Output = Chr$(13)
  End If
End Select

Case "Vn"
  LeadLabel = "Test"
  Lead_num = 8
  If MSComm1.PortOpen Then
    MSComm1.Output = "Port2=0"
    MSComm1.Output = Chr$(13)
  End If
End Select

Case "Test"
  LeadLabel = "Short"
  Lead_num = 9
  If MSComm1.PortOpen Then
    MSComm1.Output = "Port2=15"
    MSComm1.Output = Chr$(13)
  End If
End Select

End Sub

' Increments the mm/div scale
Private Sub compxaxis_Click()
  Select Case mv_div
    Case 0.1
      mv_div = 0.5
    Case 0.5
      mv_div = 1
    Case Else
      mv_div = 2
    End Select
  End Sub

' Increments the mm/sec scale
Private Sub compyaxis_Click()
  Dim I
  For I = 0 To 10
    Line3(I).Visible = False
  Next I
  mm_sec = mm_sec + 1
  If mm_sec > 10 Then mm_sec = 10
  samp_cm = 200 * mm_sec
  For I = 0 To 200 * mm_sec Step 1
    Xcoord(I) = xorigin + (xmax / ((mm_sec) * 200)) * (I)
  Next I
  End Sub

' Decrements the mv/ division scale
Private Sub comnxaxis_Click()
  Select Case my div
    Case 0.1
      my div = 0.5
    Case 0.5
      my div = 1
    Case Else
      my div = 2
    End Select
  End Sub

' decrements the mm_sec scale
Private Sub commYaxis_Click()
  Dim I
  For I = 0 To mm_sec - 1
    Line3(I).Visible = True
    Line3(I).X1 = 120 + (6804 / (mm_sec)) * (I + 1)
    Line3(I).X2 = 120 + (6804 / (mm_sec)) * (I + 1)
    Line3(I).Y1 = 120
    Line3(I).Y2 = 120 + (5670)
  Next I
  For I = 0 To 200 * mm_sec Step 1
    Xcoord(I) = xorigin + (xmax / ((mm_sec) * 200)) * (I)
  Next I
  End Sub
Appendix B. Visual Basic Code for the Virtual EKG Machine

' Vertical grid
If mm_sec > 1 Then
    For I = 0 To mm_sec - 1
        Line3(I).Visible = True
        Line3(I).X1 = 120 + (6804 / (mm_sec) * (I + 1))
        Line3(I).X2 = 120 + (6804 / (mm_sec) * (I + 1))
        Line3(I).Y1 = 120
        Line3(I).Y2 = 120 + (5670)
    Next I
    End If
    For I = 0 To 200 * mm_sec Step 1
        Xcoord(I) = xorigin + (xmax / ((mm_sec) * 200)) * (I)
        Ycoord(I) = DrawGraphValue(ActualGraph, I) = Stored_data(Label_store, J)
    Next I
Next J
Next k
End Sub

' Make an electrogram from the stored waves
Private Sub Electrogram_Click()
    Dim I, k
    Cls
    StartX = 120
    StartY = 120
    DrawWidth = 1
    xmax = 6804 '10 cm
    ymax = 5670 / 2 '5 cm
    yorigin = ymax + StartY / 2
    xorigin = StartX
    ' Vertical grid
    For I = 3 To 10
        Line3(I).Visible = False
    Next I
    For I = 0 To 2
        Line3(I).Visible = True
        Line3(I).X1 = 120 + (6804 / (3)) * (I + 1)
        Line3(I).X2 = 120 + (6804 / (3)) * (I + 1)
        Line3(I).Y1 = 120
        Line3(I).Y2 = 120 + (5670)
    Next I
    Numgraph = 4
    mm_sec = 3
    For I = 0 To 600 Step 1
        Xcoord(I) = xorigin + (xmax / ((mm_sec) * 200)) * (I)
        Ycoord(I) = DrawGraphValue(J, 1)
    Next I
Next J
Next k
End Sub

' Horizontal grid
Private Sub Form_Resize()
    ' Resize the Term (display)
    ' control and status bar.
    MSComml.RThreshold = 1
    Term.Move 0, 15, ScaleWidth / 2, ScaleHeight / 2 + 15
End Sub

' Serial communication toolbox
Private Sub Form_Unload(Cancel As Integer)
    Dim T&
    If MSComml.PortOpen Then
        ' Wait 10 seconds for data to be transmitted.
        If I = 0 Then colr = 4:
            DrawWidth = 3
            If I = 5 Then
color = 14:
        DrawWidth = 1
    Else:
        color = 4:
        DrawWidth = 1
        Line (xorigin, StartY + (k - 1) * 2 * ymax / Numgraph) - (xorigin + xmax, StartY + (k - 1) * 2 * ymax / Numgraph) + (I * 0.2) * ymax / Numgraph), QBColor(colr)
    Next I
Next k
End Sub
Appendix B. Visual Basic Code for the Virtual EKG Machine

T& = Timer + 10
Do While MSComm1.OutBufferCount
    Ret = DoEvents()
    If Timer > T& Then
        Select Case MsgBox("Data cannot be sent", 34)
            ' Cancel.
            Case 3
                Cancel = True
            Exit Sub
            ' Retry.
            Case 4
                T& = Timer + 10
            ' Ignore.
            Case 5
                Exit Do
            End Select
            End If
        End Loop
    MSComm1.PortOpen = 0
    End If
    ' If the log file is open, flush and close it.
    If hLogFile Then MCloseLog_Click
End End Sub

' Print the form
Private Sub GoPrint_Click()
    Dim Msg ' Declare variable.
    On Error GoTo ErrorHandler
    ' Set up error handler.
    Command1.Visible = False
    'Hide Buttons
    Command2.Visible = False
    Command4.Visible = False
    Command5.Visible = False
    Command6.Visible = False
    Command7.Visible = False
    Command8.Visible = False
    Command9.Visible = False
    Command10.Visible = False
    Command11.Visible = False
    Command12.Visible = False
    Command13.Visible = False
    Label8.Visible = False
    LeadLabel.Visible = False
    Store.Visible = False
    Electrogram.Visible = False
    Label_store.Visible = False
    HScroll1.Visible = False
    PrintForm ' Print form.
    Command1.Visible = True
    'Unhide objects
    Command2.Visible = True
    Command4.Visible = True
    Command5.Visible = True
    Command6.Visible = True
    Command7.Visible = True
    Command8.Visible = True
    Command9.Visible = True
    Command10.Visible = True
    Command11.Visible = True
    Command12.Visible = True
    Command13.Visible = True
    Label7.Visible = True
    ActualGraph.Visible = True
    NumGraph.Visible = True
    GoPrint.Visible = True
    Save.Visible = True
    Load.Visible = True
    cmdQuit.Visible = True
    FileName.Visible = False
    Label8.Visible = False
    LeadLabel.Visible = False
    Store.Visible = False
    Electrogram.Visible = False
    Label_store.Visible = False
    HScroll1.Visible = False
    PrintForm ' Print form.
    Command1.Visible = True
    'Unhide objects
    Command2.Visible = True
    Command4.Visible = True
    Command5.Visible = True
    Command6.Visible = True
    Command7.Visible = True
    Command8.Visible = True
    Command9.Visible = True
    Command10.Visible = True
    Command11.Visible = True
    Command12.Visible = True
    Command13.Visible = True
    Label7.Visible = True
    ActualGraph.Visible = True
    NumGraph.Visible = True
    GoPrint.Visible = True
    Save.Visible = True
    Load.Visible = True
    cmdQuit.Visible = True
    FileName.Visible = False
    Label8.Visible = False
    LeadLabel.Visible = False
    Store.Visible = False
    Electrogram.Visible = False
    Label_store.Visible = False
    HScroll1.Visible = False
    End Sub

ErrorHandler:
    MsgBox Msg ' Display message.
    Resume Next
End Sub

' Change the selected 3 seconds of data by moving the scroll
Private Sub HScroll1_Change()
    Dim I
    If HScroll1.Value < 3 Then
        HScroll1.Value = 3
        Textout = HScroll1.Value
        'Box for the copy selection
        DrawWidth = 4
Appendix B. Visual Basic Code for the Virtual EKG Machine

Cls
QBColor (6)
Line (120 + 6810 * (Textout - 3) / mm_sec, 120)-(120 + 6810 * (TextOut) / mm_sec, 5670 + 120), QBColor(4), BF
DrawWidth = 1
CurrentX = Xcoord(l)
CurrentY = Ycoord(l)
For I = 1 To 200 * mm_sec - 20
Step 4
If I > 200 * (Textout - 3) And I < 200 * (Textout) Then
Line -(Xcoord(I), Ycoord(I)), QBColor(15)
Else
Line -(Xcoord(I), Ycoord(I)), QBColor(4)
End If
Next I
End Sub

The next routine loads a saved ' electrogram and plot it on the ' screen
Private Sub Load_Click()
Dim I, J, filename, temp1$(8), lead_num_temp, lead_text_graph(5) As String
Dim Ycoord_ant, Xcoord_ant
StartX = 120
StartY = 120
DrawWidth = 1
xmax = 6804 '10 cm
ymax = 5670 / 2 '5 cm
yorigin = ymax + StartY / 2
xorigin = StartX
Numgraph = 4
mm_sec = 3
Cls
Open File_name For Input As #1
' Open file.
Pat_name = " "
Do
Input #1, temp1$(1)
Pat_name = Pat_name + temp1$(1)
Loop Until temp1$(1) <> Chr(13)
Time_now = " "
Do
Input #1, temp1$(1)
Time_now = Time_now + temp1$(1)
Loop Until temp1$(1) <> Chr(13)
For k = 1 To 4
Lead_num_text_1(k).Visible = True
End Sub

Day = Day + temp1$(1)
Loop Until temp1$(1) <>
Chr(13)
Time_now = " "
Do
Input #1, temp1$(1)
Time_now = Time_now + temp1$(1)
Loop Until temp1$(1) <> Chr(13)
For k = 1 To 4
Lead_num_text_1(k).Visible = True
End Sub

DrawGraphValue(k, J) ' Write string to file.
Value(J)
Next J
Next k
Close ' Close all open files.
For I = 0 To 600 Step 1
Xcoord(I) = xorigin + (xmax / ((mm_sec) * 200)) * (I)
Next I
' Vertical grid
For I = 0 To 10
Line3(I).Visible = False
Next I
For I = 0 To mm_sec - 1
Line3(I).Visible = True
Line3(I).X1 = 120 + (6804 / (mm_sec)) * (I + 1)
Line3(I).X2 = 120 + (6804 / (mm_sec)) * (I + 1)
Line3(I).Y1 = 120
Line3(I).Y2 = 120 + (5670)
Next I
' Horizontal grid
For k = 1 To 4
Line_x_w(k).Visible = True
Line_x_w(k).X1 = xorigin
Line_x_w(k).Y1 = StartY + (k - 1) * 2 * ymax / Numgraph
Line_x_w(k).X2 = xorigin + xmax
Line_x_w(k).Y2 = StartY + (k - 1) * 2 * ymax / Numgraph
Next k
Next k
Close
For I = 0 To 600 Step 1
Xcoord(I) = xorigin + (xmax / ((mm_sec) * 200)) * (I)
Next I
' Vertical grid
For I = 0 To 10
Line3(I).Visible = False
Next I
For I = 0 To mm_sec - 1
Line3(I).Visible = True
Line3(I).X1 = 120 + (6804 / (mm_sec)) * (I + 1)
Line3(I).X2 = 120 + (6804 / (mm_sec)) * (I + 1)
Line3(I).Y1 = 120
Line3(I).Y2 = 120 + (5670)
Next I
' Horizontal grid
For k = 1 To 4
Line_x_w(k).Visible = True
Line_x_w(k).X1 = xorigin
Line_x_w(k).Y1 = StartY + (k - 1) * 2 * ymax / Numgraph
Line_x_w(k).X2 = xorigin + xmax
Line_x_w(k).Y2 = StartY + (k - 1) * 2 * ymax / Numgraph
Next k
Next k
Close
Appendix B. Visual Basic Code for the Virtual EKG Machine

For I = 0 To 10 Step 1
    If I = 0 Then colr = 4: DrawWidth = 3 Else: If I = 5 Then colr = 14: DrawWidth = 1 Else: colr = 4: DrawWidth = 1
    Line x(10 * (k - 1) + I).Visible = True
    Line x(10 * (k - 1) + I).X1 = xorigin
    Line x(10 * (k - 1) + I).Y1 = StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph
    Next I
Next k
For J = 1 To Numgraph
    CurrentX = Xcoord(I)
    CurrentY = DrawGraphValue(J, 1)
    Ycoord_ant = CurrentY
    Xcoord_ant = Xcoord(I)
    Case 3
    Line_plot_3(I).Visible = True
    Line_plot_3(I).X1 = Xcoord_ant
    Line_plot_3(I).Y1 = Ycoord_ant
    Xcoord_ant = Xcoord(I)
    Line_plot_3(I).X2 = Xcoord(I)
    Line_plot_3(I).Y2 = Ycoord(J)
    Ycoord_ant = Ycoord(J)
    Xcoord_ant = Xcoord(I)
    Case 4
    Line_plot_4(I).Visible = True
    Line_plot_4(I).X1 = Xcoord_ant
    Line_plot_4(I).Y1 = Ycoord_ant
    Xcoord_ant = Xcoord(I)
    Line_plot_4(I).X2 = Xcoord(I)
    Line_plot_4(I).Y2 = Ycoord(J)
    Ycoord_ant = Ycoord(J)
    Xcoord_ant = Xcoord(I)
End Select
Next I
Next J
End Sub

' Close the log file.
Private Sub MCloseLog_Click()
    Close hLogFile
    hLogFile = 0
    MOpenLog.Enabled = True
    MCloseLog.Enabled = False
    Form1.Caption = "VEKG Terminal"
End Sub

' Toggle the DTREnable property.
Private Sub MDTREnable_Click()
    MSComm1.DTREnable = Not MSComm1.DTREnable
    MSComm1.DTREnable.Checked = MSComm1.DTREnable
End Sub

' Close the program from the file menu.
Private Sub MFileExit_Click()
    ' Use Form Unload since it has code to check for unsent data
    ' and an open log file.
    Form_Unload Ret
End Sub
Appendix B. Visual Basic Code for the Virtual EKG Machine

' Toggle the DTREnable property
to hang up the line.
Private Sub MHangup_Click()
    Ret = MSComm1.DTREnable
    Save the current setting.
    MSComm1.DTREnable = True
    Turn DTR on.
    MSComm1.DTREnable = False
    Turn DTR off.
    MSComm1.DTREnable = Ret
    Restore the old setting.
End Sub

' Setting InputLen to 0 specifies
'that the entire contents of the
'buffer should be read.
Private Sub MInputLen_Click()
    On Error Resume Next
    temp$ = InputBox$("Enter New InputLen:", "InputLen",
    Str$(MSComm1.InputLen))
    If Len(temp$) Then
        MSComm1.InputLen = Val(temp$)
        If Err Then MsgBox Error$, 48
    End If
End Sub

' Toggles the state of the port
'(open or closed).
Private Sub MOpen_Click()
    On Error Resume Next
    Dim OpenFlag
    MSComm1.PortOpen = Not
    MSComm1.PortOpen
    If Err Then MsgBox Error$, 48
    OpenFlag = MSComm1.PortOpen
    MOpen.Enabled = OpenFlag
    MHangup.Enabled = OpenFlag
End Sub

' The next routine opens the log
'file
Private Sub MOpenLog_Click()
    Dim replace
    On Error Resume Next
    ' Get the log filename from the
'user.
    OpenLog.DialogTitle = "Open Communications Log File"
    OpenLog.Filter = "Log Files (*.LOG)|*.log|All Files (*.*)|*.*"
    Do
        temp$ = OpenLog.filename
        If the file already exists,
        ask if the user wants to
        overwrite the file or add to
        it.
        Ret = Len(Dir$(temp$))
        If Err Then

Appendix B. Visual Basic Code for the Virtual EKG Machine

MsgBox Error$, 48
Exit Sub
End If
If Ret Then
    replace = MsgBox("Replace existing file - " + temp$ + "?", 35)
Else
    replace = 0
End If
Loop While replace = 2
' User clicked the Yes button, so delete the file.
If replace = 6 Then
    Kill temp$
If Err Then
    MsgBox Error$, 48
    Exit Sub
End If
End If
' User clicked the No button, so don't delete the file.
If replace = 0 Then
    replace = MsgBox("Replace existing file - " + temp$ + "?", 35)
Else
    replace = 0
End If
Loop While replace = 2
' User clicked the Yes button, so delete the file.
If replace = 6 Then
    Kill temp$
If Err Then
    MsgBox Error$, 48
    Exit Sub
End If
End If
' Open the log file.
hLogFile = FreeFile
Open temp$ For Binary Access As hLogFile
If Err Then
    MsgBox Error$, 48
    Close hLogFile
    hLogFile = 0
    Exit Sub
End If
' Go to the end of the file so that new data can be appended.
Seek hLogFile, LOF(hLogFile) + 1
End If
Form1.Caption = "MSComm Terminal - " + OpenLog.FileTitle
MOpenLog.Enabled = False
MCloseLog.Enabled = True
End Sub

' This procedure sets the ParityReplace property, which holds the character that will replace any incorrect characters that are received because of a parity error.
Private Sub MParRep_Click()
    On Error Resume Next
    temp$ = InputBox("Enter Replace Character", "ParityReplace", Form1.MSComm1.ParityReplace)
    Form1.MSComm1.ParityReplace = Left$(temp$, 1)
    If Err Then MsgBox Error$, 48
    End Sub

' This procedure sets the RThreshold property, which determines how many bytes can arrive at the receive buffer before the OnComm event is triggered and the CommEvent property is set to vbMSCommEvReceive.
Private Sub MRThreshold_Click()
    On Error Resume Next
    temp$ = InputBox("Enter New RThreshold:", "RThreshold", Str$(MSComml.RThreshold))
    If Len(temp$) Then
        MSComml.RThreshold = Val(temp$)
        If Err Then MsgBox Error$, 48
    End If
    End Sub

' The OnComm event is used for trapping communications events and errors.
Private Static Sub MSComml_OnComm()
    Dim EVMsg$
    Dim ERMsg$
    ' Branch according to the CommEvent property.
    Select Case MSComml.CommEvent
        Case vbMSCommEvReceive
            ShowData Term, (MSComml.Input)
        Case vbMSCommEvSend
        Case vbMSCommEvCTS
            EVMsg$ = "Change in CTS Detected"
        Case vbMSCommEvDSR
            EVMsg$ = "Change in DSR Detected"
        Case vbMSCommEvCD
            EVMsg$ = "Change in CD Detected"
        Case vbMSCommEvRing
Appendix B. Visual Basic Code for the Virtual EKG Machine

EVMsg$ = "The Phone is Ringing"
Case vbMSCommEvEOF
 EVMsg$ = "End of File Detected"
' Error messages.
Case vbMSCommErrBreak
 EVMsg$ = "Break Received"
Case vbMSCommErrCSTSTO
 ERMsg$ = "CTS Timeout"
Case vbMSCommErrDSRSTO
 ERMsg$ = "DSR Timeout"
Case vbMSCommErrFrame
 EVMsg$ = "Framing Error"
Case vbMSCommErrOverrun
 ERMsg$ = "Overrun Error"
Case vbMSCommErrCDTO
 ERMsg$ = "Carrier Detect Timeout"
Case vbMSCommErrRxOverlap
 ERMsg$ = "Receive Buffer Overflow"
Case vbMSCommErrRxParity
 EVMsg$ = "Parity Error"
Case vbMSCommErrTxFull
 ERMsg$ = "Transmit Buffer Full"
Case Else
 ERMsg$ = "Unknown error or event"
End Select
If Len(EVMsg$) Then
' Display event messages in the label control.
' Labell1.Caption = EVMsg$
 EVMsg$ = ""
ElseIf Len(ERMsg$) Then
' Display error messages in an alert message box.
 Beep
 Ret = MsgBox(ERMsg$, 1, "Click to quit, OK to ignore.")
 ERMsg$ = ""
' If the user clicks Cancel (2)...
 If Ret = 2 Then
 MSComm1.PortOpen = 0
' Close the port and quit.
 End If
 End If
End Sub

Private Sub MSendText_Click()
 On Error Resume Next
 Dim hSend, BSize, LF&
 MSendText.Enabled = False
' Get the text filename from the user.
 OpenLog.DialogTitle = "Send Text File"
 OpenLog.Filter = "Text Files (*.TXT)\n *.txt|All Files (*)\n *.=*"
 Do
 OpenLog.filename = ""
 OpenLog.ShowOpen
 If Err = cdlCancel Then Exit
 Sub
 temp$ = OpenLog.filename
' If the file doesn't exist, go back.
 Ret = Len(Dir$(temp$))
 If Err Then
 MsgBox Error$, 48
 MSendText.Enabled = True
 Exit Sub
 End If
 If Ret Then
 Exit Do
 Else
 MsgBox temp$ + " not found!", 48
 End If
 Loop
 Open the log file.
 hSend = FreeFile
 Open temp$ For Binary Access Read As hSend
 If Err Then
 MsgBox Error$, 48
 Else
' Display the Cancel dialog box.
 CancelSend = False
 Form2.Label1.Caption = "Transmitting Text File - " + temp$
 Form2.Show
' Read the file in blocks the size of the transmit buffer.
 BSize = MSComm1.OutBufferSize
 LF& = LOF(hSend)
 Do Until EOF(hSend) Or CancelSend
' Don't read too much at the end.

Appendix B. Visual Basic Code for the Virtual EKG Machine

If LF& - Loc(hSend) <= BSize Then
    BSize = LF& - Loc(hSend) + 1
End If
' Read a block of data.
temp$ = Space$(BSize)
Get hSend, , temp$
' Transmit the block.
MSComml.Output = temp$
If Err Then
    MsgBox Error$, 48
    Exit Do
End If
' Wait for all the data to be sent.
Do
    Ret = DoEvents()
Loop Until MSComml.OutBufferCount = 0 Or CancelSend
Loop
Close hSend
MSendText.Enabled = True
CancelSend = True
Form2.Hide
End Sub

' This procedure adds data to the Term control's Text property. It also filters control characters, such as BACKSPACE, carriage return, and line feeds, and writes data to an open log file. BACKSPACE characters delete the character to the left, either in the Text property, or the passed string. Line feed characters are appended to all carriage returns. The data is converted in to samples and then plotted on the screen.
Private Static Sub ShowData(Term As Control, Dta$)
    On Error Resume Next
    Dim Nd, I, countJ, Value12, LSB, MSB, L, Start_count, End_count
    ' Make sure the existing text doesn't get too large.
    Nd = LenB(Term.Text)
    If Nd >= 16384 Then
        Term.Text = Mid$(Term.Text, 4097)
        Nd = LenB(Term.Text)
    End If
    If (Term.Visible = True) Then
        ' Point to the end of Term's data.
        Term.SelStart = Nd
        ' Filter/handle BACKSPACE characters.
        Do
            I = InStr(Dta$, Chr$(8))
            If I Then
                If I = 1 Then
                    Term.SelStart = Nd - 1
                    Term.SelLength = 1
                Else
                    Dta$ = Left$(Dta$, I - 2) + Mid$(Dta$, I + 1)
                End If
            End If
        Loop While I
        ' Eliminate line feeds.
        Do
            I = InStr(Dta$, Chr$(10))
            If I Then
                Dta$ = Mid$(Dta$, I + 1)
            End If
        Loop While I
Appendix B. Visual Basic Code for the Virtual EKG Machine

If I Then
    Dta$ = Left$(Dta$, I - 1) + Mid$(Dta$, I + 1)
End If
Loop While I
' Make sure all carriage returns have a line feed.
I = 1
Do
    I = InStr(I, Dta$, Chr$(13))
    If I Then
        Dta$ = Left$(Dta$, I) + Chr$(10) + Mid$(Dta$, I + 1)
        Textin = Left$(Dta$, I)
        I = I + 1
    End If
Loop While I

End If
' Get values for later graphing
For I = 1 To Len(Dta$)
    Value(Count_graph) = Asc(Mid(Dta$, I, 1))
    If (Term.Visible = True) Then
        Dta2$ = Chr(Value(Count_graph))
        ' Add the filtered data to the Text property.
        Term.SetSelText = Dta2$
    End If
    If hLogFile And (Count_graph Mod 3 = 0) And (Count_graph > 3) Then
        Dta3$ = Dta3$ + Str(Value(Count_graph)) + "," + Str(Value(Count_graph - 1)) + "," + Str(Value(Count_graph - 2)) + Chr(13)
    End If
    If Count_graph = 902 Then
        Count_graph = 0
        JUMP = 1
        For J = 1 To 200 * mm_sec - 300
            Ycoord(J) = Ycoord(J + 297)
        Next J
        If (Value(301) = 170 And Value(304) = 170) Then
            Start_count = 2
            End_count = -3
            ElseIf (Value(302) = 170 And Value(305) = 170) Then
                Start_count = 3
                End_count = -2
                ElseIf (Value(303) = 170 And Value(306) = 170) Then
                    Start_count = 4
                    End_count = -1
                    End If
                    For J = Start_count To (896 + End_count) Step 3
                        If Value(J) < 16 Then
                            MSB = Value(J)
                            LSB = Value(J + 1)
                            Value(J) = MSB * 256 + LSB
                        Else
                            MSB = Value(J) And 15
                            LSB = Value(J + 1)
                            Value(J) = -1 * (4095 - (MSB * 256 + LSB))
                        End If
                        Ycoord(200 * mm_sec - 303 + Int(J / 3)) = 120 + (2835 * ((2 * ActualGraph - 1) / (Numgraph))) - ((ymax) * ((Value(J)) / (4096 * Numgraph * mv_div)))
                    Next J
                    End If
                    Count_graph = Count_graph + 1
                    Next I
                    ' Log data to file if requested.
                    If hLogFile Then
                        I = 2
                        Do
                            Err = 0
                            Put hLogFile, , Dta3$
                            Dta3$ = ""
                            If Err Then
                                I = MsgBox(Error$, 21)
                            If I = 2 Then
                                MCloseLog_Click
                                End If
                                End If
                                End If
                                Loop While I <> 2
                                End If
                                ' Graphic interface
                                If JUMP Then
                                   Cls
                                    ' Horizontal grid
                                    For k = 1 To Numgraph
                                        For I = 0 To 10 Step 1
                                            If I = 0 Then colr = 4:
                                                DrawWidth = 3 Else: If I = 5 Then colr = 14: DrawWidth = 1 Else:
                                                colr = 4: DrawWidth = 1
Appendix B. Visual Basic Code for the Virtual EKG Machine

Line (xorigin, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph) - (xorigin + xmax, StartY + (k - 1) * 2 * ymax / Numgraph + (I * 0.2) * ymax / Numgraph), QBColor(colr)

Next I
Next k
CurrentX = xorigin
CurrentY = Ycoord(1)
For I = 1 To 200 * mm sec - 20 Step 4
Line -(Xcoord(I), Ycoord(I)), QBColor(1)
Next I
JUMP = 0
End If
End Sub

' The next routine stores the actual electrogram in a file, including the patient name, and the present date and time
Private Sub Save_Click()
Dim I, J, filename
Open File_name For Output As #1
' Open file.
Print #1, Pat name
Print #1, Day
Print #1, Time_now
For k = 1 To 4
Print #1, Lead num_text_1(k).Text
For J = 1 To 600
Print #1, DrawGraphValue(k, J)
Next J
Next k
Close ' Close all open files.
End Sub

' The next routine stores the selected 3 seconds of data into a variable in order to make later an electrogram
Private Sub Store_Click()
For J = 1 To 600
    Stored data(Label store, J) = (((2835 * (2 * ActualGraph - 1) / Numgraph) + 120 - Ycoord(200 * (Textout - 3) + J)) * Numgraph * mv_div / 22.23) + 128
Next J
Stored Lead(Label store) = LeadLabel
Label_store.Caption = Label_store.Caption + 1
End Sub

' Keystrokes trapped here are sent to the MSComm control
Appendix B. Visual Basic Code for the Virtual EKG Machine

' where they are echoed back via
' the OnComm (vbMSCommEvReceive)
' event, and displayed with the
' ShowData procedure.
Private Sub
Term_KeyPress(KeyAscii As
Integer)
' If the port is opened...
If MSComm1.PortOpen Then
' Send the keystroke to the
port.
    MSComm1.Output = Chr$(KeyAscii)
' Unless Echo is on, there is
no need to let the text control
display the key.
    If Not Echo Then KeyAscii = 0
End If
End Sub
Appendix C.

Complete Circuit Diagram
References:


References