Instrumentation for the reproduction and measurement of dynamic blood pressure and flow characteristics in the *Limulus polyphemus* aorta

by

Dale Eugene Crist

A Thesis Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Departments: Biomedical Engineering
Electrical Engineering and Computer Engineering
Co-majors: Biomedical Engineering
Electrical Engineering

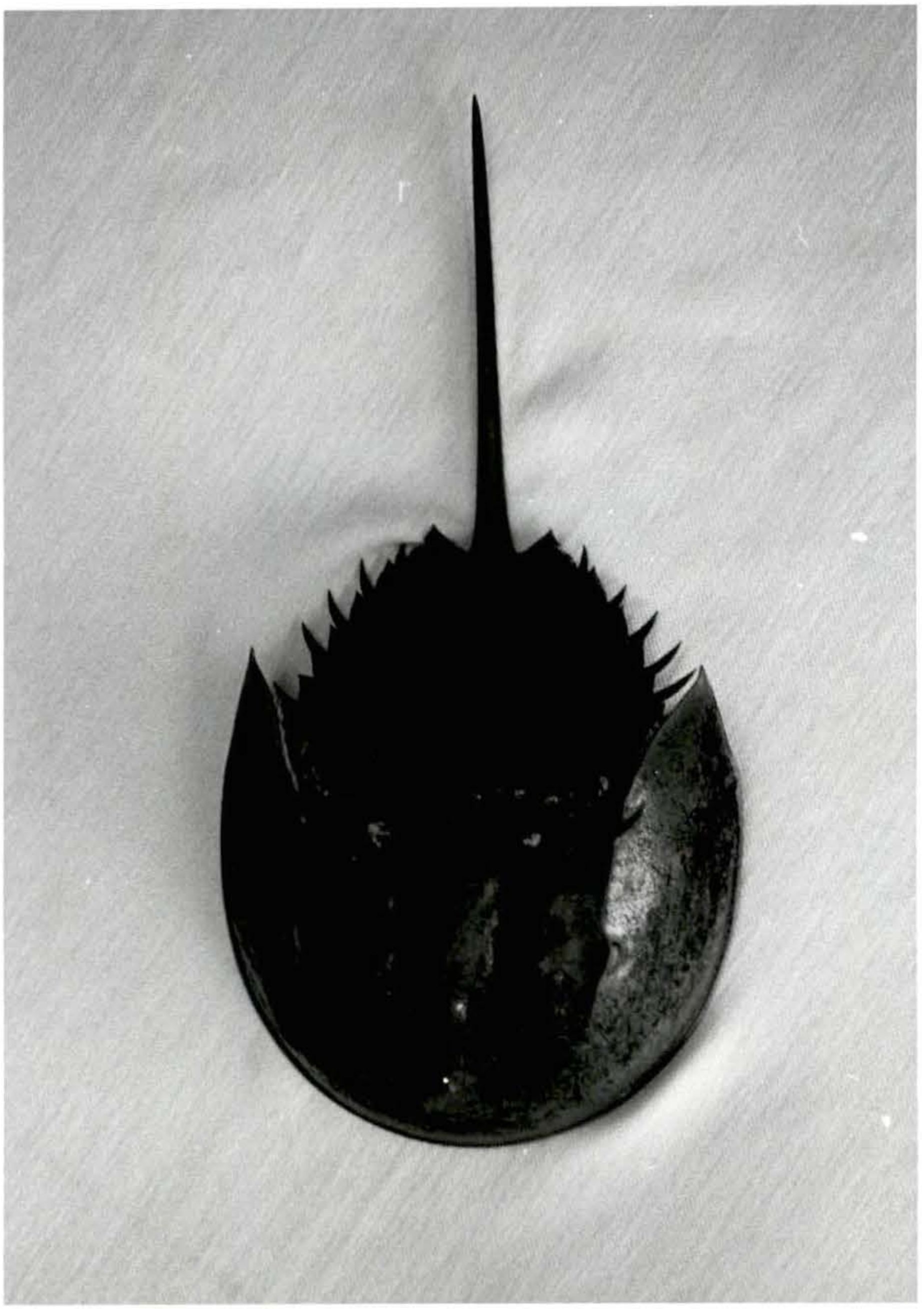
Approved:

For the Graduate College

Iowa State University
Ames, Iowa
1988

TABLE OF CONTENTS

LIST OF FIGURES



INTRODUCTION

The physiological study of animals involves a variety of different techniques many of which are applied to more-or-less intact organisms. Other techniques use living organs or tissues which have been removed from the body. Still, others are concerned with nonliving chemical or physical characteristics of the animal. Mammals are, by far, the group that has been most intensively studied. Therefore, most instrumentation for physiological study is oriented towards this group. To analyze other organisms such as invertebrates, it is often necessary to modify equipment used for mammalian research or to design totally new equipment which is suitable for use on smaller and anatomically different organisms.

Mollusca, Arthropoda, and Chordata are the three most highly developed of the animal phyla. Of these, the Chordata includes the vertebrates. With some exceptions, members of these groups have a very well developed circulatory system. Vertebrate circulatory systems include a heart for pumping and a closed system of vessels in which the blood is contained. Because of the continuous and closed nature of these vessels, they are called "closed" circulatory systems. On the other hand, circulatory systems of arthropods and most molluscs are called "open" circulatory systems. The circulatory system in these animals has secondarily expanded and largely pushed the true body cavity, the coelom, into a few small spaces. To a varying extent, the arterial blood vessels remain, but venous return is by way of large sinuses that have become the functional body cavity. Therefore, the organs lie in blood filled sinuses in addition to receiving an arterial blood supply.

The three phyla mentioned above are very important from biological and economic viewpoints and have been researched extensively. Even so, there is relatively little known about the dynamic characteristics of "open" circulatory systems.

Animals with an open circulatory system are mostly small and the study of their circulatory system is very difficult in-vivo. Because of its larger size and easy access to a pair of aortas that leave the anterior end of the heart, the horseshoe crab *Limulus polyphemus* was chosen for this investigation. The specimens being studied range in size from 20 to 25 centimeters across their midsection and weigh 1100 to 2200 grams. Two sections of aorta approximately one to one and a half cm long and three to five mm in diameter can be removed from an animal of this size. The objective of the current research is to determine the physical properties of these aortas under dynamic conditions that mimic normal circulation. This includes stress vs. strain characteristics obtained while fluid is flowing through the aorta. Data of this type can be obtained from pressure-volume curves obtained using the system developed and described in this thesis. The following is a description of the development of an instrumentation system used for the measurement of the viscoelastic properties of the aortas of an arthropod, the horseshoe crab *Limulus polyphemus*. This is part of an ongoing study of the hemodynamics of open circulatory systems.

PURPOSE

The purpose of this study was primarily to develop an instrumentation system which will mimic the normal pressure and flow characteristics of blood driven by the heart through the aortas of a *Limulus polyphemus*. Secondly, there was a need to measure the pressure of the fluid pumped through the aorta section and to determine the amount of fluid displaced as the aorta expanded during a "systolic" cycle. Finally, there was the need to determine the phase shift between the pressure and volume curves.

The expected change in volume of the aorta with each cardiac cycle was only a few microliters. This is a small volume and its direct measurement is difficult. The displaced volume and the hydrostatic pressure exerted upon the specimen will also vary with the back pressure applied to the system. This back pressure is set by the experimenter and corresponds to the circultory system's "peripheral resistance".

LITERATURE REVIEW

It has been found that the heart rate for a *Limulus polyphemus* ranges between 12 and 51 beats per minute (bpm) with an average of around 30 bpm (Redmond, Jorgensen, and Bourne, 1982). With this range of heart rates, typical of many biological systems, the data that must be acquired are well within the capabilities of most off-the-shelf data acquisition systems. As laboratory research is becoming more automated and the availability of personal computers increases, PC based data acquisition systems are also becoming very popular (Keithley). Appendix D contains a list of popular data acquisition hardware and software manufacturers for IBM PC type machines (Conner, 1986). By no means is this a complete list because new data acquisition hardware and software is announced and released at an ever increasing rate (Conner, 1986). It is important to note that there is a wide variety of different functions supported by each different manufacturer for both hardware and software.

The three main architectures for PC-based data acquisition systems are "board-level" architecture, "expansion chassis" architecture, and thirdly, a combination of one or two external boards in an expansion chassis and one "board-level" interface card in the PC (Keithley). There are advantages and disadvantages to each type of architecture. Although cost is not solely dependent upon the type of architecture, generally, the "board-level" hardware is the least expensive costing anywhere from $200 to over $2000 (Conner, 1986). The expansion chassis and what can be referred to as the "modified-expansion chassis" architecture are usually the most expensive types of hardware and range in price from $1000 to over $5000. The board-level product provides a compact, low cost, and versatile data acquisition system for laboratory settings where basic analog-to-digital, timer/counter, and digital I/O are all that is needed. These systems usually have 8 or 16 analog input channels with resolutions of either 8, 12, or 16 bits per channel (Conner, 1986). Board-level hardware is limited to the number of channels which are provided and to the types of I/O functions provided without any possible expansion. They also require an open slot on the PC expansion bus into which they are inserted. "If your task is small, well-defined, and static, a board-level product is the answer" (Keithley).

The other two, more open, types of architecture offer more flexibility in the number of channels and functions that are available (Keithley). With expansion chassis architecture, hardware updates can be simplified by replacing only the card which contains the function in question. Some of the special functions which are available for this type of architecture are: transducer signal conditioners, transducer excitation, and stepper-motor control (Keithley). Further, there are intelligent data acquisition systems in which there is a microprocessor performing many of the same functions as the host computer. An "intelligent" system will outperform a board-level system in many applications and can run in the background without any degradation in host computer processing speed. Without a need for high speed background operation or stand alone monitoring, the intelligent systems are generally an expensive overkill in most laboratory situations (Keithley).

When choosing a data acquisition system some important questions need to be answered. Can "canned" software be used for the intended application or will the user have to write his/her own software? Is the intended application fully defined? Is there an anticipated need for an expandable system and if so, how much expansion will be desired? Is there a need for data acquisition with "real-time" graphical output or will post-processing of the data be sufficient? And finally, how much money is to be spent? These are some questions which should be carefully considered before actually purchasing a system.

SYSTEM DEVELOPMENT AND DESIGN

The instrumentation and data acquisition system which is described here is a combination of a number of subsystems. These subsystems include 1) a stepper-motor driven syringe pump, 2) a Zenith Z-158 Personal Computer, 3) hardware and software for the Zenith PC, 4) a Micromint BCC-52 single-board computer, 5) interface hardware for the BCC-52, 6) a Statham P23Db pressure transducer and Beckman 411 amplifier, 7) a custom chamber to hold an aorta sample, and 8) a Type 7212 Narco Bio-systems Impedance Pneumograph Coupler and 7070 Amplifier.

Stepper Motor Fundamentals

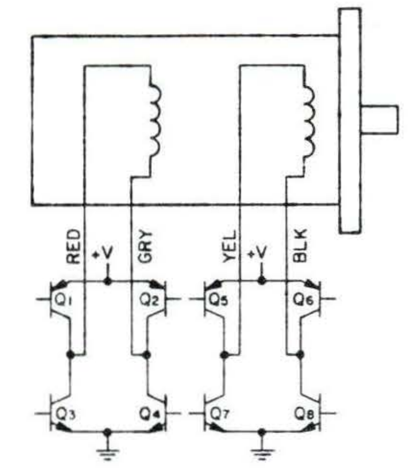
A stepper motor is a device that produces discrete mechanical rotational movement in response to electrical pulses. The amount of angular movement is controlled by the construction of the motor and the means by which it is pulsed. Stepper motors are designed with step-angles of anywhere from 0.72 to 90 degrees (Airpax, 1982). Through various gear box arrangements the step-angle is virtually limitless but always discrete. Using a small step-angle motor and multiple steps will produce less overshoot, stiffer movements, and greater accuracy than a larger step-angle motor (Airpax, 1982). For example, if a movement of 60 degrees is desired, a 7.5 degree motor stepping 8 times will work better than a 60 degree motor stepping once.

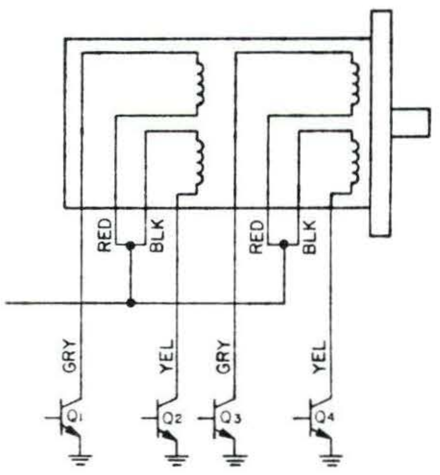
Although not a critical factor for this system, the accuracy of a stepper motor is +/- 6.5% noncumulative and at no load (Airpax, 1982). Since the step error is noncumulative, the over all error is zero in a complete revolution. This assumes that the motor is operating in a normal torque and voltage range.

The stepper-motor used in this system was an original component of the syringe pump selected for this project. Consequently, because most of the mechanical considerations had already been taken into account, it was easier to adapt this stepper-motor to computer control. One important mechanical parameter that this eliminated was "motor torque". The original designers of the syringe pump did a very good job of specifying a stepper motor with suitable torque. Secondly, the syringe pump's mechanical linkage to the syringe was perfect for the task at hand. The most important considerations were electrical.

Stepper motors are generally available with two types of windings 1) bipolar, and 2) unipolar. See Figure 1 for a sketch of each type of coil arrangement (Airpax, 1982). The stepper motor in the syringe pump has a unipolar type of winding. Unipolar wound motors simplify the switching and timing needed to step the motor. In the bipolar winding

Figure 1. Bipolar and Unipolar winding schematics and stepping sequences (Airpax, 1982)





BIPOLAR

UNIPOLAR

Normal
4 Step Sequence

CW ROTATION / CCW ROTATION

Step	Q_1-Q_4	Q_2-Q_3	Q_5-Q_8	Q_6-Q_7
1	ON	OFF	ON	OFF
2	ON	OFF	OFF	ON
3	OFF	ON	OFF	ON
4	OFF	ON	ON	OFF
1	ON	OFF	ON	OFF

CW ROTATION / CCW ROTATION

Step	Q_1	Q_2	Q_3	Q_4
1	ON	OFF	ON	OFF
2	ON	OFF	OFF	ON
3	OFF	ON	OFF	ON
4	OFF	ON	ON	OFF
1	ON	OFF	ON	OFF

½ Step
8 Step Sequence

CW ROTATION ↓ / CCW ROTATION ↑

1	ON	OFF	ON	OFF
2	ON	OFF	OFF	OFF
3	ON	OFF	OFF	ON
4	OFF	OFF	OFF	ON
5	OFF	ON	OFF	ON
6	OFF	ON	OFF	OFF
7	OFF	ON	ON	OFF
8	OFF	OFF	ON	OFF
1	ON	OFF	ON	OFF

CW ROTATION ↓ / CCW ROTATION ↑

1	ON	OFF	ON	OFF
2	ON	OFF	OFF	OFF
3	ON	OFF	OFF	ON
4	OFF	OFF	OFF	ON
5	OFF	ON	OFF	ON
6	OFF	ON	OFF	OFF
7	OFF	ON	ON	OFF
8	OFF	OFF	ON	OFF
1	ON	OFF	ON	OFF

Wave Drive
4 Step Sequence

CW ROTATION / CCW ROTATION

1	ON	OFF	OFF	OFF
2	OFF	OFF	OFF	ON
3	OFF	ON	OFF	OFF
4	OFF	OFF	ON	OFF
1	ON	OFF	OFF	OFF

CW ROTATION / CCW ROTATION

1	ON	OFF	OFF	OFF
2	OFF	OFF	OFF	ON
3	OFF	ON	OFF	OFF
4	OFF	OFF	ON	OFF
1	ON	OFF	OFF	OFF

arrangement, the switching requires eight transistors and precise timing that does not allow the power supply to be shorted to ground.

Two of the most common stepping sequences are shown in Figure 1. Using the unipolar-normal-4-step sequence, 15 degree steps were produced. Some initial testing was done at this step-angle but the smaller half-step-8-step sequence was chosen to gain the maximum amount of control over the motor and thus the syringe movement. Reversing the stepping sequence will change the direction of the stepper motor.

Ultimately, the stepper motor was interfaced to a single-board computer. The complete details of this interfacing will be discussed later but additional motor details should be discussed here. As with any motor, there are inductive problems to overcome. Whenever the motor is started, stopped, or switched to the next phase, there will be inductive voltage spikes produced in the system. These spikes can be potentially lethal to the control circuit. In this case, protective measures were taken to reduce any hazard to the control circuit. To prevent the voltage across each inductive winding from "spiking" as it is switched off, a 1N4006 clamping diode is placed across each motor winding (Moyer, 1982). Another consideration is the stepping process itself. If the motor and the stepping sequence become out of phase then control over the motor will be lost or steps will be missed. This problem occurs most often when the controller tries to sequence too fast or there is too much "torque loading" on the motor. The maximum sequencing rate is determined by the characteristics of the system in which the motor is operating. Some factors in determining maximum stepping rate include motor supply voltage and current, type of drive circuit and load on the motor. For a fast stepping rate, a large di/dt (derivative of current with respect to time) in each phase of the motor is required. High di/dt necessitates a large supply voltage for the motor (Moyer, 1982). The stepper-motor in this project was supplied with its rated maximum supply voltage.

Single-Board Computer Highlights

The single-board computer used as a controller for this project is a BCC52 BASIC Computer/Controller manufactured by The Micromint, INC. As shipped, the BCC52 requires a power supply (+5v,+12v,-12v) and a terminal device for operation. At the heart of the board is an Intel 8052AH microcontroller. At 4 1/2 by 6 1/2 inches, the board is very compact and contains only 17 integrated circuits. The whole system includes memory space for RAM/ROM/EPROM, EPROM programmer circuitry, three parallel ports, and two serial ports. At the time of this writing, the BCC52 retails for $199.00. As is true in the computer industry, the product could be less expensive tomorrow.

Microcontroller

The 8052AH is an 8-bit microcontroller based on Intel's MCS-51 architecture. Figure 2 shows a block diagram of this architecture (Intel, 1985b). The MCS-51 family includes six different microcontrollers. The major features of each microcontroller are (Intel, 1985b):

:8-bit CPU

:on-chip oscillator and clock circuitry

:32 I/O lines

:64K address space for external data memory

:64K address space for external program memory

:two 16-bit timer/counters (three on 8052AH)

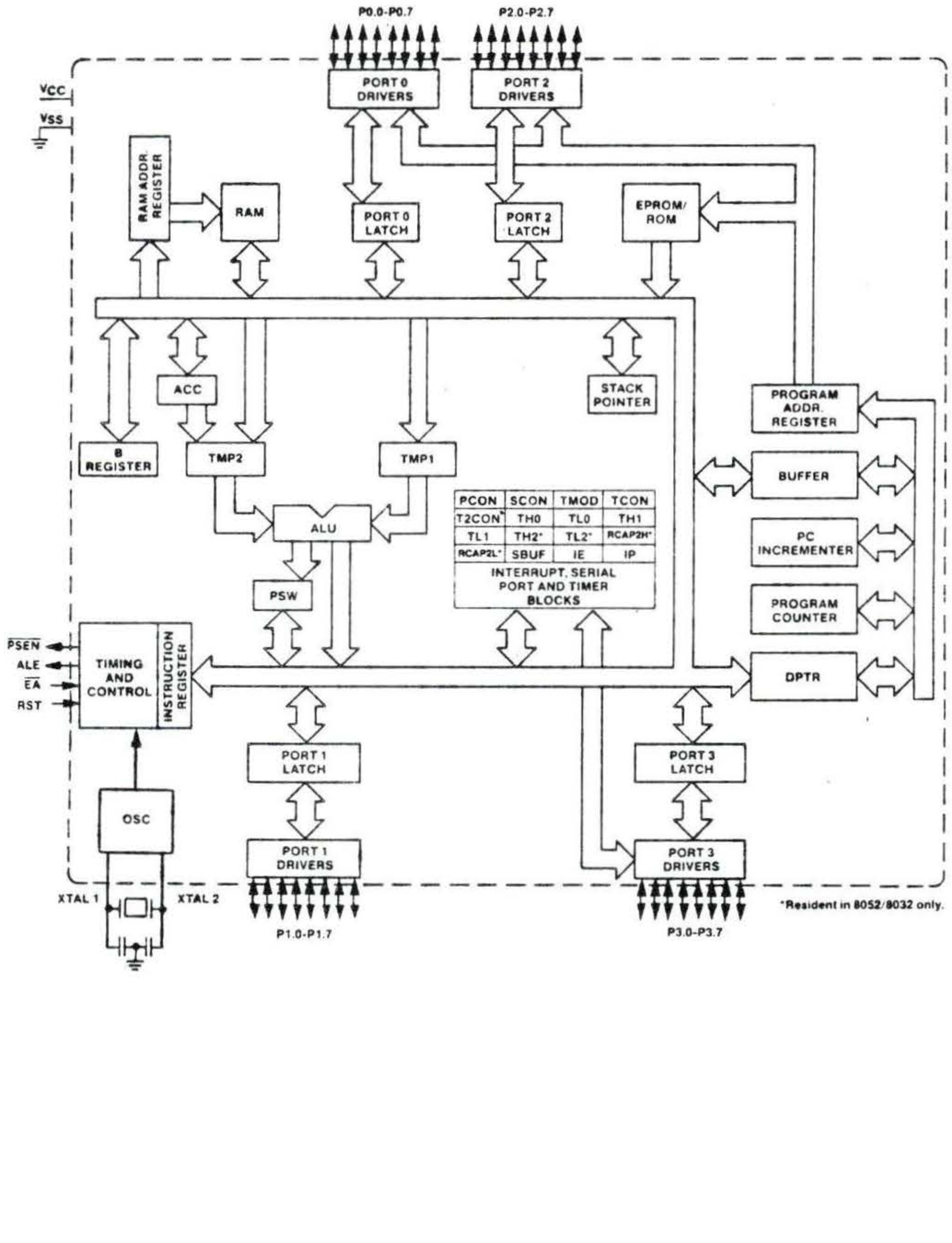
:a five-source interrupt structure (six on 8052AH)
with two priority levels.

:full duplex serial port

:boolean processor

The 8052AH is fabricated using HMOS (high performance n-channel MOS) technology (Intel, 1985b). HMOS technology greatly reduces power consumption for the microcontroller over bi-polar technology but uses a factor of 100 times more power than CMOS (complementary n-channel MOS) technology. The maximum power dissipation for the 8052AH is 1.5 watts.

Figure 2. Block diagram of MCS-51 architecture
(Intel, 1985b)


P0.0-P0.7
P2.0-P2.7
VCC
VSS
PORT 0 DRIVERS
PORT 2 DRIVERS
RAM ADDR REGISTER
RAM
PORT 0 LATCH
PORT 2 LATCH
EPROM/ ROM
ACC
STACK POINTER
PROGRAM ADDR. REGISTER
B REGISTER
TMP2
TMP1
BUFFER
PCON
SCON
TMOD
TCON
T2CON*
TH0
TL0
TH1
TL1
TH2*
TL2*
RCAP2H*
RCAP2L*
SBUF
IE
IP
INTERRUPT, SERIAL PORT AND TIMER BLOCKS
ALU
PC INCREMENTER
PSW
PROGRAM COUNTER
PSEN
ALE
EA
RST
TIMING AND CONTROL
INSTRUCTION REGISTER
DPTR
PORT 1 LATCH
PORT 3 LATCH
OSC
PORT 1 DRIVERS
PORT 3 DRIVERS
XTAL 1
XTAL 2
*Resident in 8052/8032 only.
P1.0-P1.7
P3.0-P3.7

Depending on the type of circuit, HMOS allows greater speed than CMOS. On-chip program memory for the 8052AH is 8K of ROM and also 256 bytes of on-chip data memory (Intel, 1985a). Of the 25 special function registers, only four were used in this design. All of the internal data storage registers are accessible through BASIC or assembly language. However, the special function registers are directly accessible only in assembly language.

The Accumulator at address 0E hex is used as in any microprocessor. It is a temporary register used for short-term storage, transfers of data between registers, or for I/O procedures.

Each bit of the Program Status Word (PSW) at address 0D hex has a special function. Bits 3 and 4 are used to select the working internal register bank. In the 8052AH, there are 4 internal register banks, RB0 through RB3. Figure 3 shows how to select each register bank with the PSW and the memory locations of each register bank (Intel, 1985a). These internal registers can be accessed through BASIC or in assembly language. In BASIC, access to internal memory is through the DBY command. In assembly language, access is either directly to the memory location or through the Working Register Bank.

PROGRAM STATUS WORD BIT 3	BIT 4	SELECTED REGISTER BANK	MEMORY LOCATION
0	0	RB0	00H - 07H
0	1	RB1	08H - 0FH
1	0	RB2	10H - 17H
1	1	RB3	18H - 1FH

Figure 3. Working register bank selection and register bank location

I/O on the BCC52 is memory-mapped which means that any I/O device is mapped into external memory space. In order to access memory or I/O devices in external memory, 16-bit addresses are required. The BASIC XBY command is used to read or write to external memory locations. For example in order to read an external memory location (0C800H) and assign its value to a variable "TEMP", the following line of BASIC code would be needed:

```
TEMP = XBY(0C800H)
```

To access external memory through assembly language it is necessary to use the Data Pointer (DPTR). Two 8-bit registers DPL at address 83 hex and DPH at address 82 hex contain the low-byte and high-byte of the DPTR, respectively. The DPTR can be manipulated as a single 16-bit register or as two 8-bit registers (Intel, 1985a). An assembly language example to perform a similar task as above and put the data in internal memory location 18H would be:

```
MOV DPTR,#0C800H
MOVX A,@DPTR
MOV 18H,A
```

One method to recognize an interrupt request, on external interrupt line 1 of the BCC52 while in an assembly language subroutine, is to examine bit 3 of the TCON (TCON.3) register at address 88 hex. If TCON.3 is set, then an interrupt request has occurred at INT1. The Timer/Counter Control Register (TCON) handles run and overflow control for timers 1 and 2, and it monitors hardware interrupts 1 and 2.

External interrupts

A powerful feature supported by the BCC52 board is that of an external user interrupt. As mentioned earlier, the 8052AH supports six interrupt sources, two of which are external interrupt lines INT0 and INT1 (Intel, 1985b). On the BCC52 INT0 is used for Direct Memory Access (DMA) Request and should generally be left to function that way. INT1, though, is for the user. Hardware interrupts can be wired to this line and the software to handle them can be written in either BASIC or assembly language. In BASIC, the interrupt is enabled with the ONEX1 command. Interrupts will be signaled when INT1 goes low. At this time, the interrupt service routine will be performed. There are many ways to handle external interrupts in assembly language, but this discussion will deal only with the manner which was used for this project. Here, INT1 is used and its status must be monitored during an assembly language subroutine.

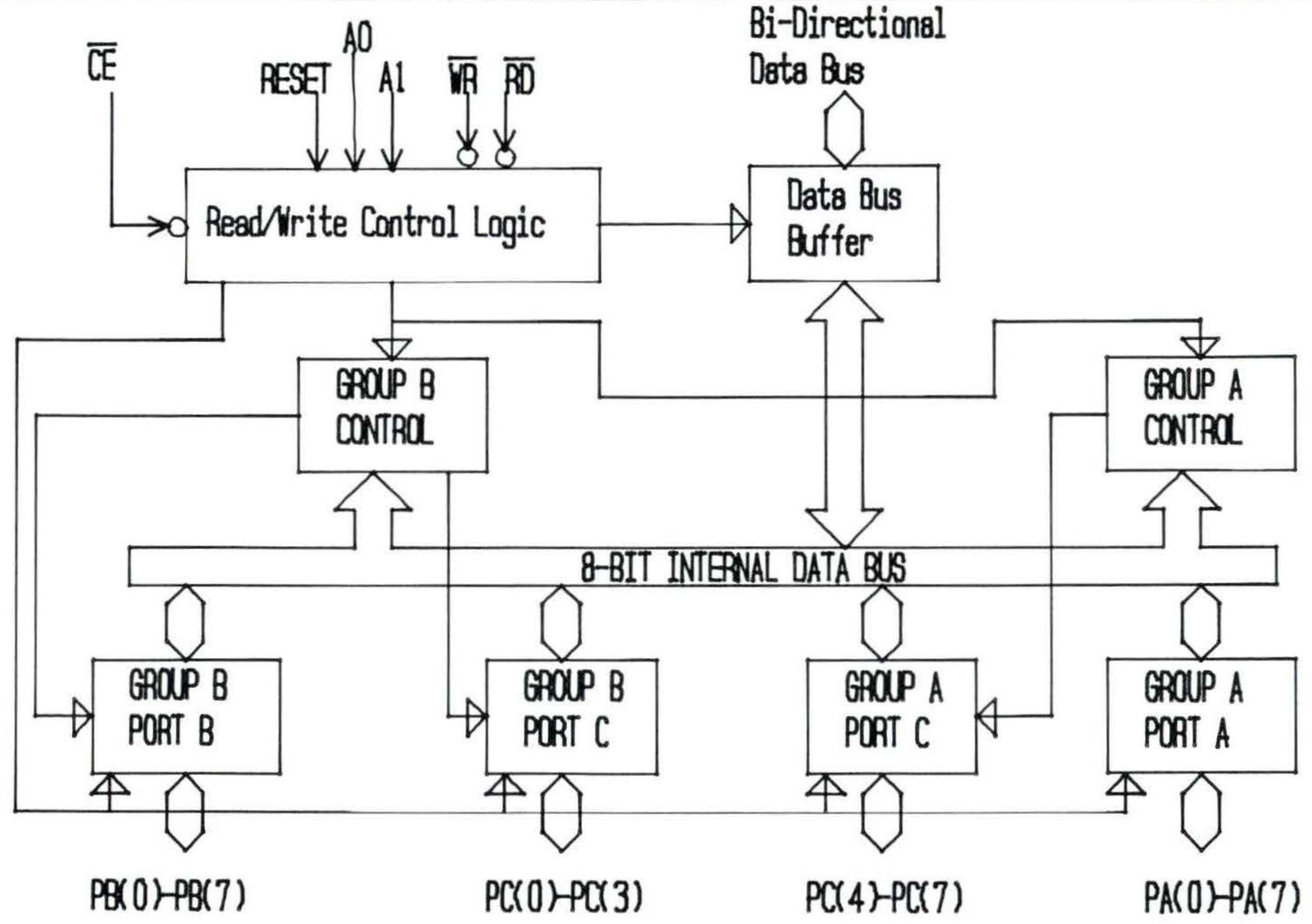
In this case, the interrupt signifies when to exit the assembly language subroutine and finish execution of the BASIC calling program. TCON.3 is set whenever a high-to-low transition is detected at INT1. It is important to note that in the interrupt service routine TCON.3 must be cleared by the user. Otherwise, the next time TCON is read, a false interrupt will be signaled. For more information on the types of interrupts available and some of the ways to handle them on the 8052AH, see Intel's microcontroller handbook (Intel, 1985b).

The BCC52 board has other features incorporated into it that make it a very flexible design tool or instrument controller. The first of these features to be discussed is an Intel 8255 Programmable Peripheral Interface (PPI) integrated circuit.

Programmable peripheral interface

The 8255 is a very powerful I/O chip which is fully compatible with all Intel microprocessor families. Its purpose is to interface peripheral equipment to a micro-controller or microprocessor system bus. A block diagram of the 8255 architecture is shown in Figure 4 (Intel, 1987).

Figure 4. Block diagram of 8255 architecture



8255 BLOCK DIAGRAM

Twenty-four programmable I/O lines are subdivided into three 8-bit ports, one of which is subdivided into an upper and lower nibble of 4-bits each. Ports A and B each contain an 8-bit data output latch/buffer and an 8-bit data input latch/buffer. Port C can be divided into two 4-bit ports under mode control. In addition to an 8-bit data output latch/buffer and an 8-bit data input buffer, there are two 4-bit latches. Port C is often used as "handshaking" lines for Ports A and B.

It is important to note that the data bus buffer is a 3-state bidirectional 8-bit buffer. Data are transmitted or received by the buffer under control of the CPU. Because the bus buffer is 3-stated, the load on the data bus is virtually zero when the 8255 is not being addressed.

The address lines (A0 and A1), chip select (CS), read (RD), write (WR), and reset are all hardwired to the 8052AH on the BCC52 board. The only control that the user has is in selecting the external memory address where the 8255 resides. Jumper 2 (JP2) and Jumper 3 (JP3) on the BCC52 are used to select the 8255's location in external memory (The Micromint, 1984). JP2 is used to select the base address of either E000H or C000H. JP3 is then used to determine the offset from the base address for the actual location of the 8255. Offset options include 800H, 900H, A00H, B00H, C00H, D00H, E00H, and F00H. On this system, the base address is set to

C000H and the offset is at 800H for an actual address of C800H. The four addresses necessary for complete control of the 8255 are:

PORT	ADDRESS
A	0C800H
B	0C801H
C	0C802H
Control	0C803H

There are three modes in which the chip can operate. Mode 0, the first mode, provides basic input and output from each of the three ports (A, B, and C). In this mode there are two 8-bit ports and two 4-bit ports. Any of the ports can be either input or output and the outputs are latched whereas the inputs are not latched. Finally, there are 16 different I/O configurations possible in this mode and they are (Intel, 1987):

Port A	Port B	Port C (upper)	Port C (lower)
output	output	output	output
output	output	output	input
output	output	input	output
output	output	input	input
output	input	output	output
output	input	output	input

output	input	input	output
output	input	input	input
input	output	output	output
input	output	output	input
input	output	input	output
input	output	input	input
input	input	output	output
input	input	output	input
input	input	input	output
input	input	input	input

Mode 1 is for strobed input and output (Intel, 1987). This facilitates I/O from specified ports with handshaking or strobes. In this mode, the I/O lines are divided into two groups (Group A and B). Each group contains a 4-bit control/data port and an 8-bit data port. Inputs and outputs of the 8-bit ports are latched and the port can be configured as either input or output. The 4-bit port is used for either control or status of the 8-bit port of its group.

The last mode (Mode 2) is used for strobed bidirectional bus I/O (Intel, 1987). This allows the use of a single 8-bit bus for both transmitting and receiving data. "Handshaking" signals are provided to maintain proper data flow on the bus similar to that of Mode 1. Only Group A is used for this mode. There is one 8-bit bidirectional bus port (Port A) and one 5-bit control/status port (Port C) in this configuration. Both inputs and outputs are latched.

Each group can be defined with a different mode for a wide variety of applications. Before any of the I/O ports can be used, a mode definition must be sent to the Control Port. Each time a new control word is sent to this port, all of the output registers and status flip-flops are reset. The control word for mode definition is as follows:

Control Word Position	Definition
	Group B
Bit 0	Port C (lower) 1 = input 2 = output
Bit 1	Port B 1 = input 0 = output
Bit 2	Mode Selection 1 = Mode 1 0 = Mode 0
	Group A
Bit 3	Port C (upper) 1 = input 0 = output
Bit 4	Port A 1 = input 0 = output

Bits 6,5	Mode Definition
	1X = Mode 2 01 = Mode 1 00 = Mode 0
Bit 7	Mode Set Flag
	1 = active

For this project Mode 0, with Port A as output and Port C (lower) as an input, is used. Port B and Port C (upper) are not used. Neither Mode 1 nor Mode 2 were used. The Control word definition for this setup is 10001001B which is 89H.

MCS BASIC-52

As has already been alluded to, a very powerful form of BASIC resides in the 8052AH. MCS BASIC-52 occupies the 8K of internal ROM in the 8052AH microcontroller (Intel, 1985b). It is much larger and more complete than many of the "tiny" BASICs that are frequently provided with microcontrollers. All of the arithmetic and I/O routines contained in MCS BASIC-52 are accessible in assembly language. Without the Utility ROM A+B (The Micromint, 1986), the only editing done in BASIC is before the return key has been pressed. The delete key is used to back space over unwanted text to erase it and then the correct text may be retyped. Once the return key is pressed, the line has to be completely retyped if changes are desired. With the ROM A+B utilities, there is a line editor which allows changes to be made to existing lines. The editing commands consist of a set of control codes, each having its own function. Also ROM A+B contains a RENUM command which allows the BASIC program lines to be renumbered with all line number references corrected.

When the BCC52 is powered up or reset, internal 8052AH memory is cleared, internal registers and pointers are initialized, and external memory is sized and cleared through MCS BASIC-52 control (Intel, 1985a). The special function registers TMOD, TCON, and T2CON are initialized. Their initial values are:

Register	Value
TCON	0F4H
TMOD	10H
T2CON	34H

BASIC also initializes MTOP to the top of external RAM. For a complete discussion of MCS BASIC-52, see the MCS BASIC-52 Users Manual (Intel, 1985a).

Text editor and assembler

In the ROM A+B utilities is a text editor and assembler for the 8052AH. TEDIT, the text editor, allows the user to write and edit 8052AH assembly language source code. It's a functional editor with its own set of editing commands. The commands are about the same as those of EDIT, the BASIC program editor and consist, primarily, of a set of control codes (The Micromint, 1986). The source code is usually saved in an EPROM so that it can be recalled for future revisions.

There are two assemblers available in ROM A+B. One is a line assembler (ASMS) which assembles each line of the program as it is entered and replaces it with the corresponding object code. It only supports backward referenced labels. A more complete assembler is ASM. It is designed to work with text files created with TEDIT and supports backward and forward labels (The Micromint, 1986). It's a two-pass assembler and after it's completed the assembly, the object code is put in external memory starting with the location specified with the origin statement.

Memory reorganization for assembly language subroutine usage

The MCS BASIC-52 CALL command is used to invoke an assembly language routine. In version 1.1 of the 8052AH the format of the CALL command is:

```
          CALL 0
or        CALL 1
```

An integer between 0 and 127 follows the command and is used to vector to a location in external memory (Intel, 1985a). The integer is multiplied by two and added to 4100H to determine the absolute vector location. So, the first sample line above will vector to location 4100H and the second will vector to 4102H. This allows the use of a small amount of memory to build a jump table of assembly language subroutines.

In order to use assembly language subroutines on the BCC52 some modifications need to be made in the memory distribution. The memory above 4000H must be configured as data memory and not as program memory. This is done in one of two ways. The value of MTOP can be moved down to adjust the memory partition to the correct value or, the easiest way to make the change, the utilities ROM (ROM A+B) can be put into the second RAM position. This leaves the required amount of program-space RAM for the microcontroller at locations 0000H-1FFFH. The next segment of memory at 2000H-3FFFH is for the ROM A+B utilities. The ROM creates the necessary break between program and data memory. The next two large memory locations at 4000H-5FFFH and 6000H-7FFFH are for either RAM or ROM memory which contain data or object code memory.

Two other changes are required on the BCC52 for completion of memory reorganization. Jumper JP5 must be changed to the "Data Storage Area Only" position and jumper JP6 must be changed to the position which reflects the position of JP5.

EPROM programmer

At the top of the memory space, at address 8000H-9FFFH, is room for an EPROM socket. Associated with the EPROM socket is the necessary hardware and firmware for programming 8K (2764) or 16K (27128) EPROMs. The user must provide a 21vdc power supply for use during programming. See Appendix C for a 21vdc power supply schematic. The timing for programming is generated in the 8052AH (The Micromint, 1985). This allows for storage of BASIC programs and assembly language source code. BASIC programs are stored sequentially in the EPROM until it is full. This is much faster than saving programs on cassette tape. An error checking algorithm is used during programming and if an error is detected, programming is halted and an error message is displayed. Only one assembly language source code file can be saved on an EPROM at a time.

Serial communication

In order to "talk" to the BCC52 with either a terminal or host computer an RS-232 serial port is provided on the board. Connection to the RS-232 port is through a DB-25 connector and cable. The cable consists of only three lines Transmit, Receive, and Ground. There are no hardware "handshaking" lines for communication to the consol device. The communication parameters are 7 data bits, 1 stop bit, no parity, and no "handshaking". With an 11.0592 MHz crystal, the maximum communication rate is 19200 bits per second, which is 2400 baud (The Micromint, 1984). The firmware to support serial communications is provided in the 8052AH.

A second serial port is provided for use with a serial printer. This port has the same communication parameters as the terminal port. Any command that uses the printer will be followed with a "#" character. For example, to list a BASIC program to the printer port instead of to the screen, the command would be: LIST#.

BCC52 Software

The primary purpose of the BCC52 computer was to control the syringe pump. With computer control, stepping patterns could be varied almost infinitely. In this project, the need was to simulate the pressure and flow patterns of the *Limulus polyphemus* heart. MCS BASIC-52 was too slow for optimum control over the stepper-motor in the syringe pump. Therefore, the actual timing and output routines were written in assembly language. In addition, for the experimenter to have full control over the simulated physiological parameters, the necessary inputs for the software were:

1) Number of stepper-motor pulses per cardiac cycle.

2) Systolic Time in milliseconds.

3) Heart Rate in beats per minute.

The program was written so that all inputs and consol outputs were performed in BASIC and then the timing was performed in assembly language.

BASIC program

A listing for the BASIC program is in Appendix A. This program, which controls the syringe pump, prompts the user for the number of stepper-motor pulses per heart beat, the systolic pumping time, and the effective heart rate that is to be simulated. Once the user has provided the required information, the program, based on the number of stepper-motor pulses per beat, corrects the systolic time to be an integral number of half-millisecond intervals. The diastolic time is calculated to be an integral number of ten-millisecond intervals based on the corrected systolic time and the heart rate that the user has requested. The adjusted values of heart rate, diastolic time, systolic time, and the time between each stepper-motor pulse is then printed out on the terminal device for the user. The final step in the BASIC program is to pass the adjusted parameters for the time between motor pulses (TBMP), diastolic time (OFTM), and the number of stepper-motor pulses per beat (MP) to an assembly language routine and call the routine. When control is returned to the BASIC program a message is printed telling the user whether the stepper motor was stopped because the syringe was empty or because the run/stop switch was set to "stop".

Assembly language subroutine

Appendix B contains the documented source code listing for the assembly language subroutine. This routine loads the values of MP, TBMP, and OFTM into three internal memory registers of the 8052AH and proceeds to process the data until one of two conditions have occurred. The routine is ended when either the switch connected to port C bit 0 has been toggled to the "stop" position, or an external interrupt has been signaled on INT1. The external interrupt is connected to the "empty syringe" cut off system to be described later. If neither of these conditions have occurred then the subroutine continues to execute.

During the systolic portion of the subroutine, a delay which represents the time between stepper-motor pulses, is executed and then the next sequence for the stepper motor is output to the stepper motor through the 8255 and the stepper-motor interface circuit described above. After the correct number of pulses have been output (systolic time), then another delay routine is executed. This routine corresponds to the diastolic time. The two break out conditions are checked at the end of this period and depending on their state, the whole routine runs again or control is returned to the BASIC program.

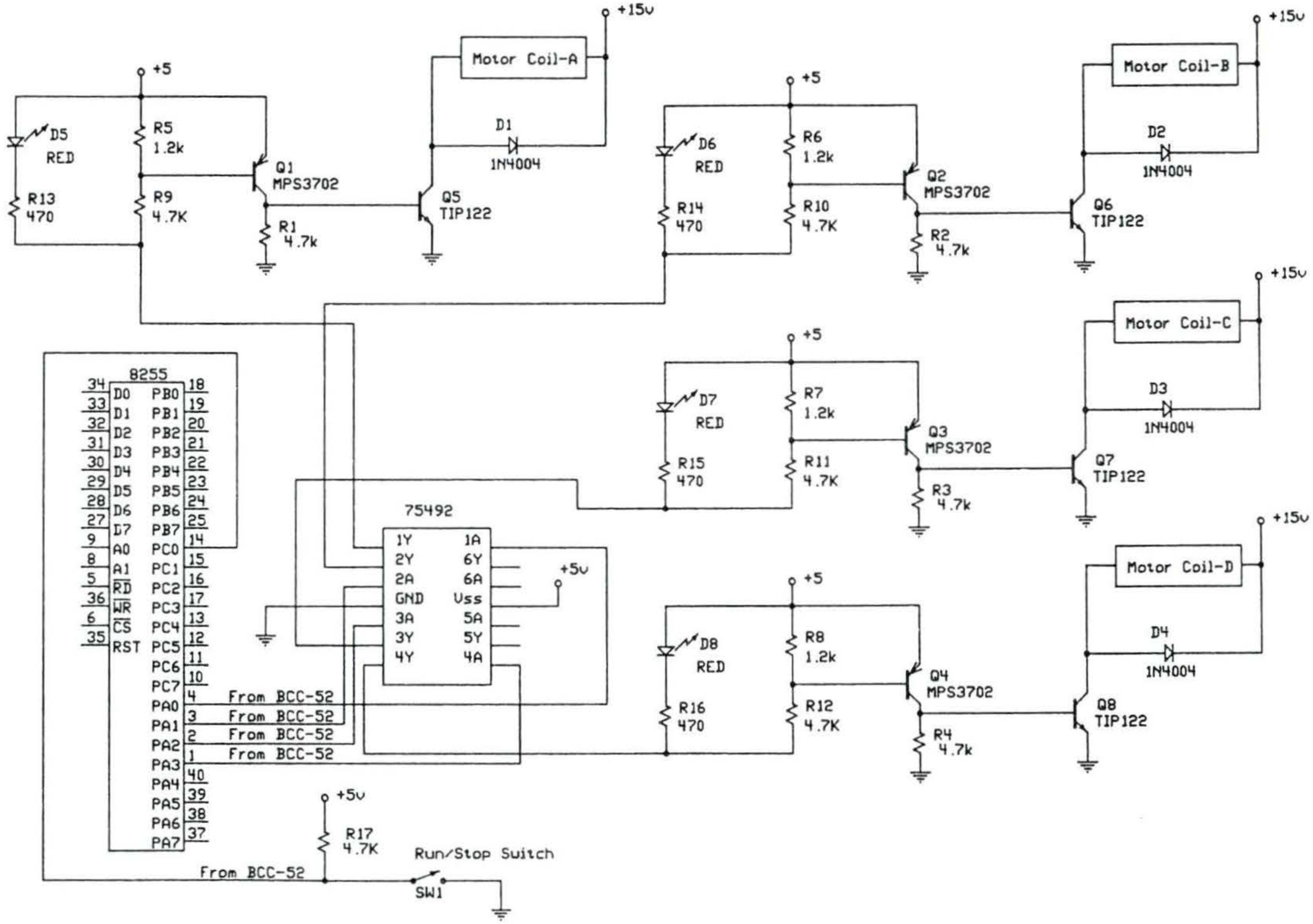
Syringe Pump

A Sage Instruments model 351 syringe pump was used in this project. Early in the project, it was determined that the driving motor in the syringe pump was a stepper motor. Without any syringe pump circuit documentation, it was also decided to completely disable all circuitry provided with the pump. Therefore, an interface between the BCC52 and the syringe pump stepper motor was needed.

Interface circuit to BCC52

Figure 5 shows the interface hardware which connects the BCC52 to the syringe pump stepper motor. The I/O port lines of the 8255 on the BCC52 are accessible through a 0.100-inch dual header connector located on the BCC52 board. In order to buffer the outputs from the 8255, an SN75492 MOS-TO-LED Driver was connected directly to the outputs of Port-A. The 75492 is rated for up to a 10v supply. Input currents are low for MOS compatibility and the outputs can sink up to 250 mA (Texas Instruments, 1983). Port A bits 0-3 were used to control the four phases of the stepper motor. Each phase of the stepper motor was con-trolled with TIP122 switching transistors, shown in Figure 5 as Q5 through Q8.

Figure 5. Interface circuit between BCC52 and syringe pump stepper-motor [Auxillary Run/Stop switch is connected to Port C, bit 0]


+5
+15v
Motor Coil-A
Motor Coil-B
Motor Coil-C
Motor Coil-D
D5 RED
D6 RED
D7 RED
D8 RED
R5 1.2k
R6 1.2k
R7 1.2k
R8 1.2k
R9 4.7K
R10 4.7K
R11 4.7K
R12 4.7K
R13 470
R14 470
R15 470
R16 470
R1 4.7k
R2 4.7k
R3 4.7k
R4 4.7k
Q1 MPS3702
Q2 MPS3702
Q3 MPS3702
Q4 MPS3702
Q5 TIP122
Q6 TIP122
Q7 TIP122
Q8 TIP122
D1 1N4004
D2 1N4004
D3 1N4004
D4 1N4004
8255
D0 D1 D2 D3 D4 D5 D6 D7 A0 A1 RD WR CS RST
PB0 PB1 PB2 PB3 PB4 PB5 PB6 PB7 PC0 PC1 PC2 PC3 PC4 PC5 PC6 PC7 PA0 PA1 PA2 PA3 PA4 PA5 PA6 PA7
75492
1Y 2Y 2A GND 3A 3Y 4Y
1A 6Y 6A Vss 5A 5Y 4A
+5v
From BCC-52
R17 4.7K
Run/Stop Switch
SW1

These are 8 ampere, Darlington, complementary silicon power transistors (National Semiconductor, 1980). They are more than adequate to handle the 200 mA steady-state current in each phase of the stepper motor. An advantage to them is that they did not require any "heat sinking" to stay within their operating temperature range. Steady-state power consumption per phase of the stepper-motor at 15vdc supply voltage is 7.5 watts. The switching voltage at this stage of the circuit is 15vdc, the same as the motor voltage.

In order to maintain some isolation between the motor voltage and the TTL level of the computer, a second set of switching transistors, Q1 - Q4, was inserted between the 75492 and Q5 - Q8. These MPS3702 P-N-P transistors were used to control the base drive for the TIP122 switching transistors and yet operate at TTL voltage levels (5.0 vdc).

A set of light-emitting-diodes (LEDs) was also inserted in parallel with transistors Q1-Q4 for system monitoring purposes. They can be used to verify the stepping sequence and to determine if the 8255 outputs are operating.

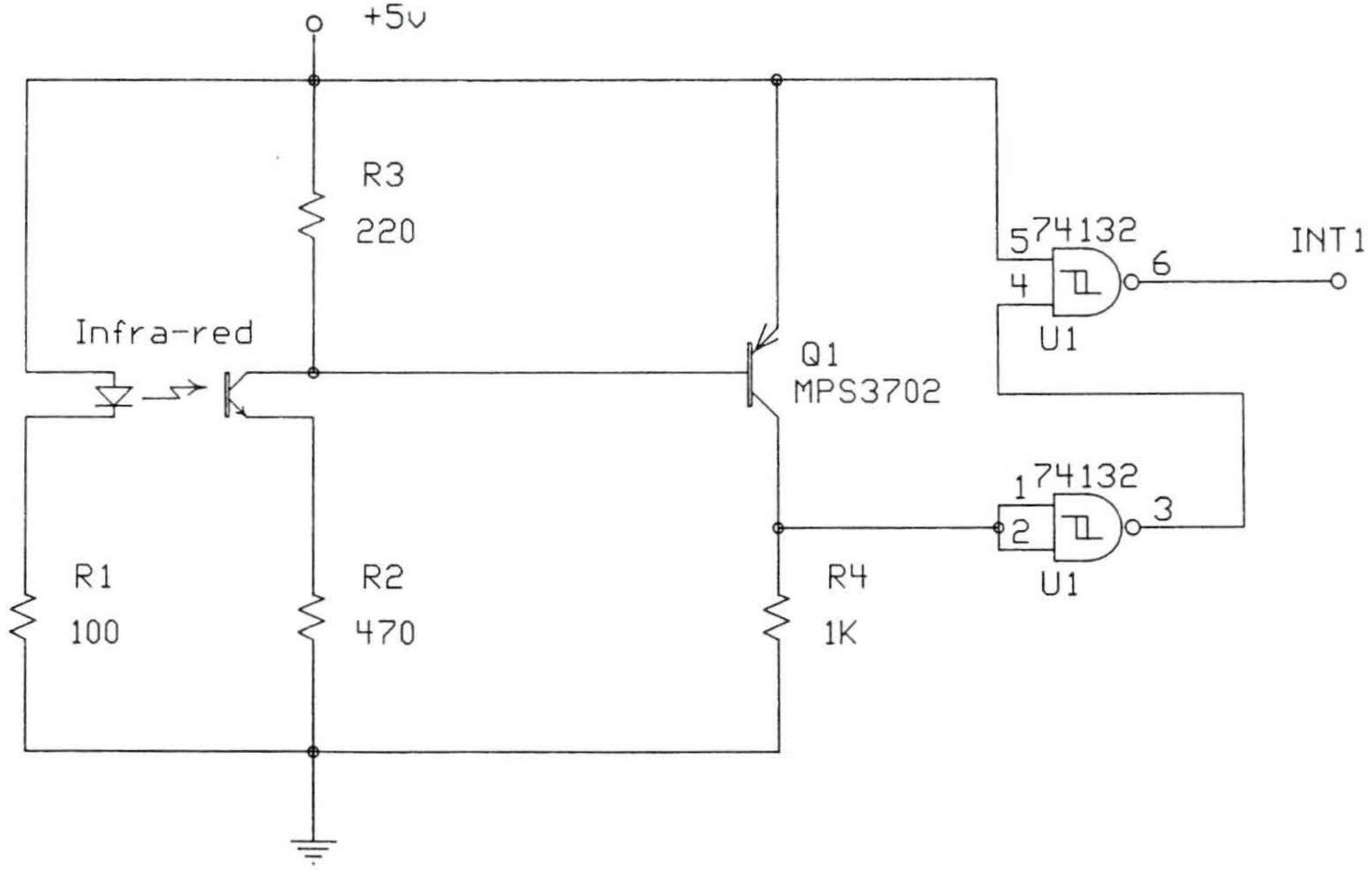
Initially the stepper motor was powered with an external 12 volt DC power supply. After receiving documentation on the syringe pump power supply and control circuit, it was decided to use the syringe pump's own power supply. The original syringe pump circuit powered the stepper motor at

15vdc (Sage Instruments, 1975). For this reason and because at 12vdc the stepper-motor tended to fall behind at higher stepping rates, the syringe pump's 15vdc supply was reconfigured to power the stepper motor.

Empty syringe detector

A second circuit, used to detect when the syringe has become empty, was also designed and incorporated into the syringe pump. As mentioned earlier, the BCC52 has an external interrupt line (INT1) for user interface. Figure 6 shows the "empty syringe" detection circuit. An infra-red emitter and detector pair is used to detect the passage of a specified point on the syringe pump carriage assembly. The gap between the emitter and the detector is 2mm. For positive triggers, the emitter diode is driven almost to its maximum of 40 mA. An MPS3702 P-N-P transistor, Q1, is controlled by the infra-red detector. When the detector is driven sufficiently to provide enough base drive for Q1, Q1 turns on. The first Schmitt triggered NAND is wired to operate as an inverter and is connected to the collector of Q1. INT1 is pulled low by the second Schmitt triggered NAND gate. Then the interrupt service routine stops the syringe drive.

Figure 6. "Empty syringe" detection circuit


+5v
R3
220
5
74132
6
INT1
4
U1
Infra-red
Q1
MPS3702
1
74132
2
3
U1
R1
100
R2
470
R4
1K

Mechanical modifications

Mechanical modifications to the syringe pump included a fixture for the "empty syringe" detection system and cabling for the interface circuit. Figure 7 shows the syringe pump with all modifications, including both the "empty syringe" detection system and the cable comming from the interface circuits described above.

Test Chamber

After an aorta is removed from the animal, it is subjected to test under predescribed conditions in a special chamber. Figure 8 shows the chamber with an aorta installed and ready for testing. The lower portion of the chamber is milled from one-inch thick plexiglas and the top is cut from quarter-inch thick plexiglas. The milled out internal working area is 7.5 cm long by 2.8 cm wide and 1.9 cm deep. The aorta is inserted between the inlet and outlet tubes which suspend it in a sea-water bath. When the top is sealed, the bath in which the aorta expands has only one place to go and that's out into the volume measuring system. It's effectively a controlled closed system that directly reflects the volume change of the aorta, during a systolic cycle, to sea-water moving in and out of a small tube. In order to have an accurate representation of the volume changes, all air bubbles must be removed from the chamber.

Figure 7. Syringe pump after all modifications [A - cable connected to interface circuits, B - carriage assembly, C - infra-red beam cutoff, D - infra-red emitter and detector, E - 50 ml syringe]

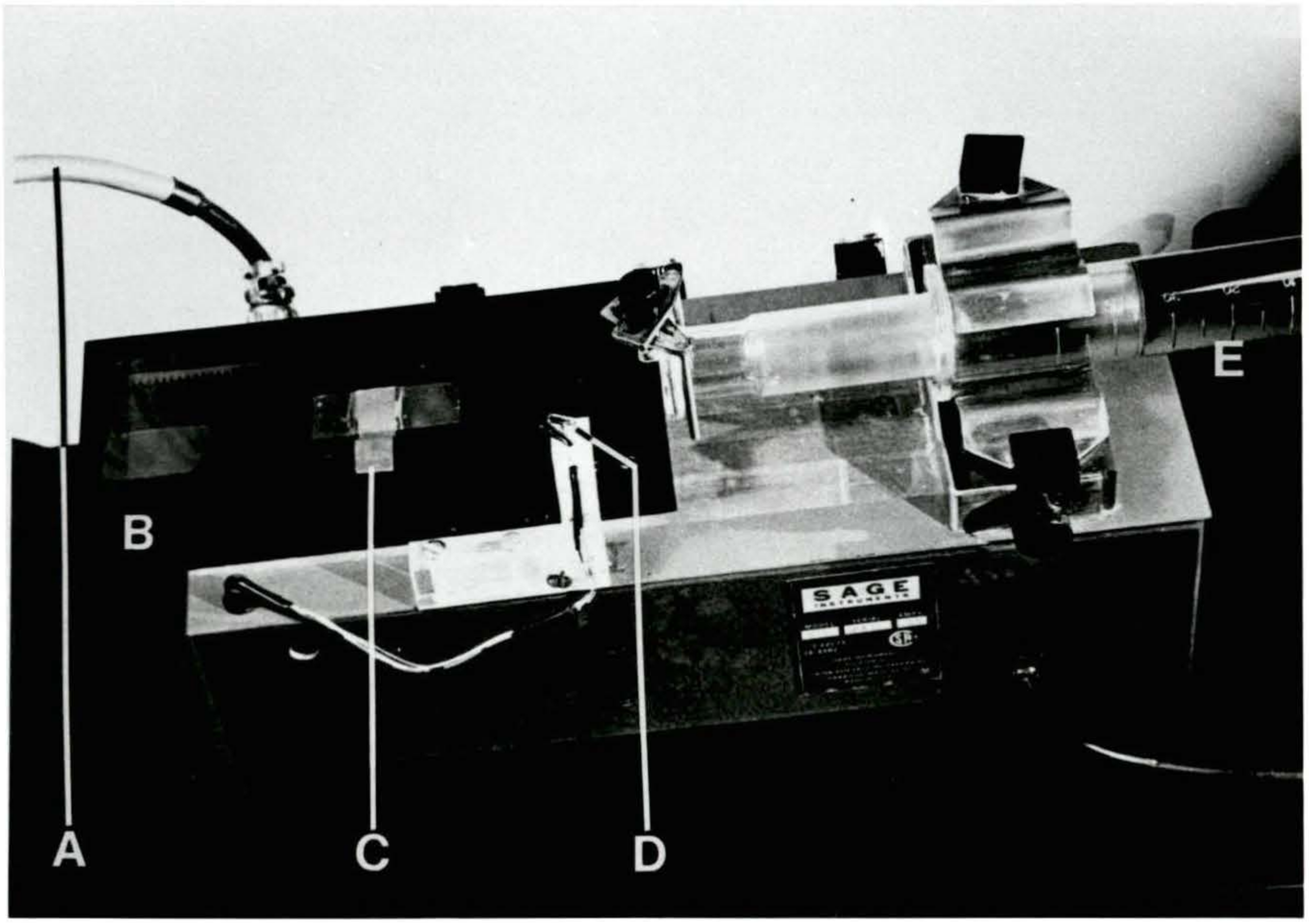
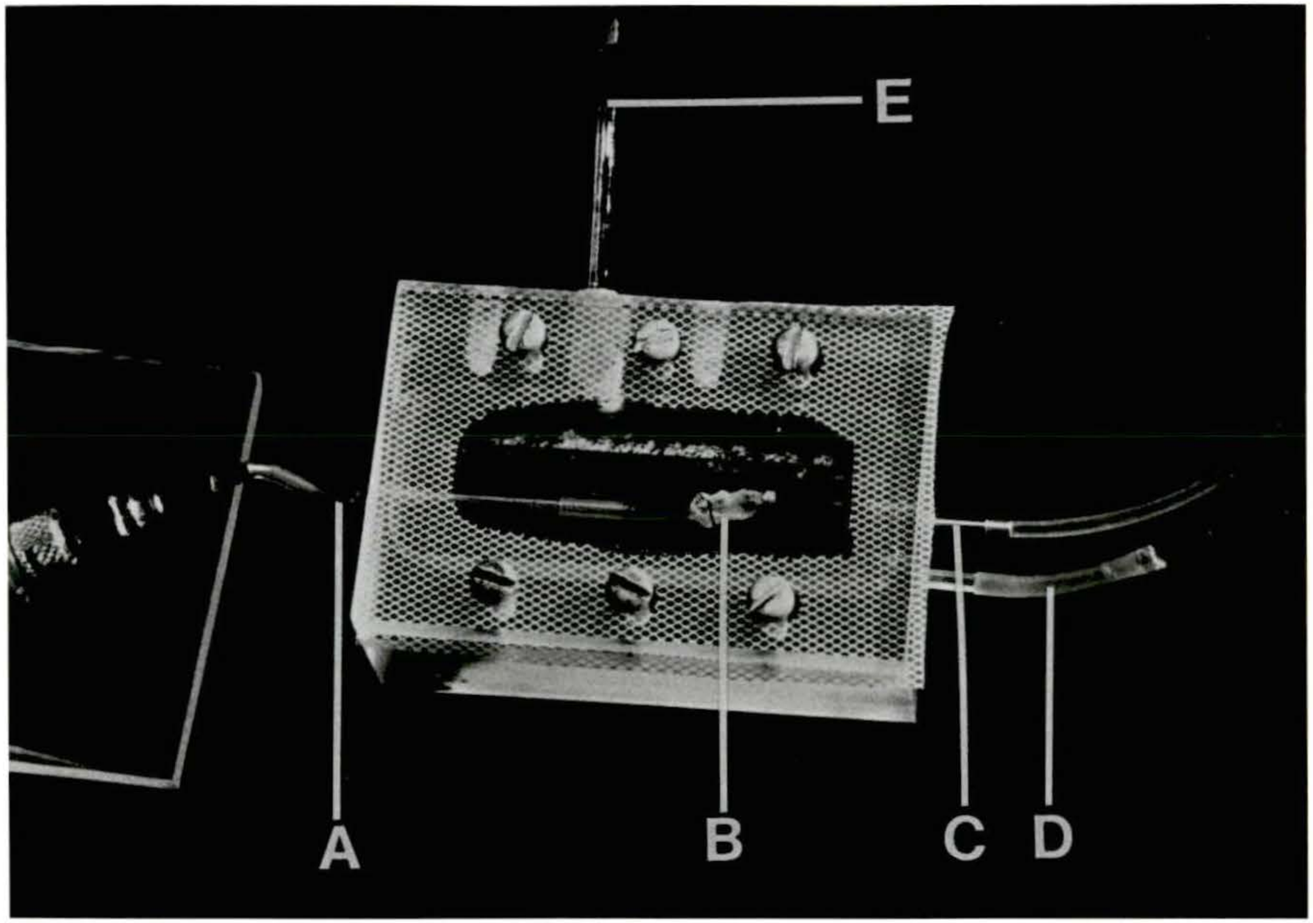

A
B
C
D
E
SAGE INSTRUMENTS

Figure 8. Aorta Test Chamber [A - inlet tube, B - aorta, C - outlet tube, D - connection to micrometer syringe,E - volume detection transducer]


E
A
B
C
D

Any air bubbles present will cause an added phase shift in the volume measuring system.

Volume Measurement

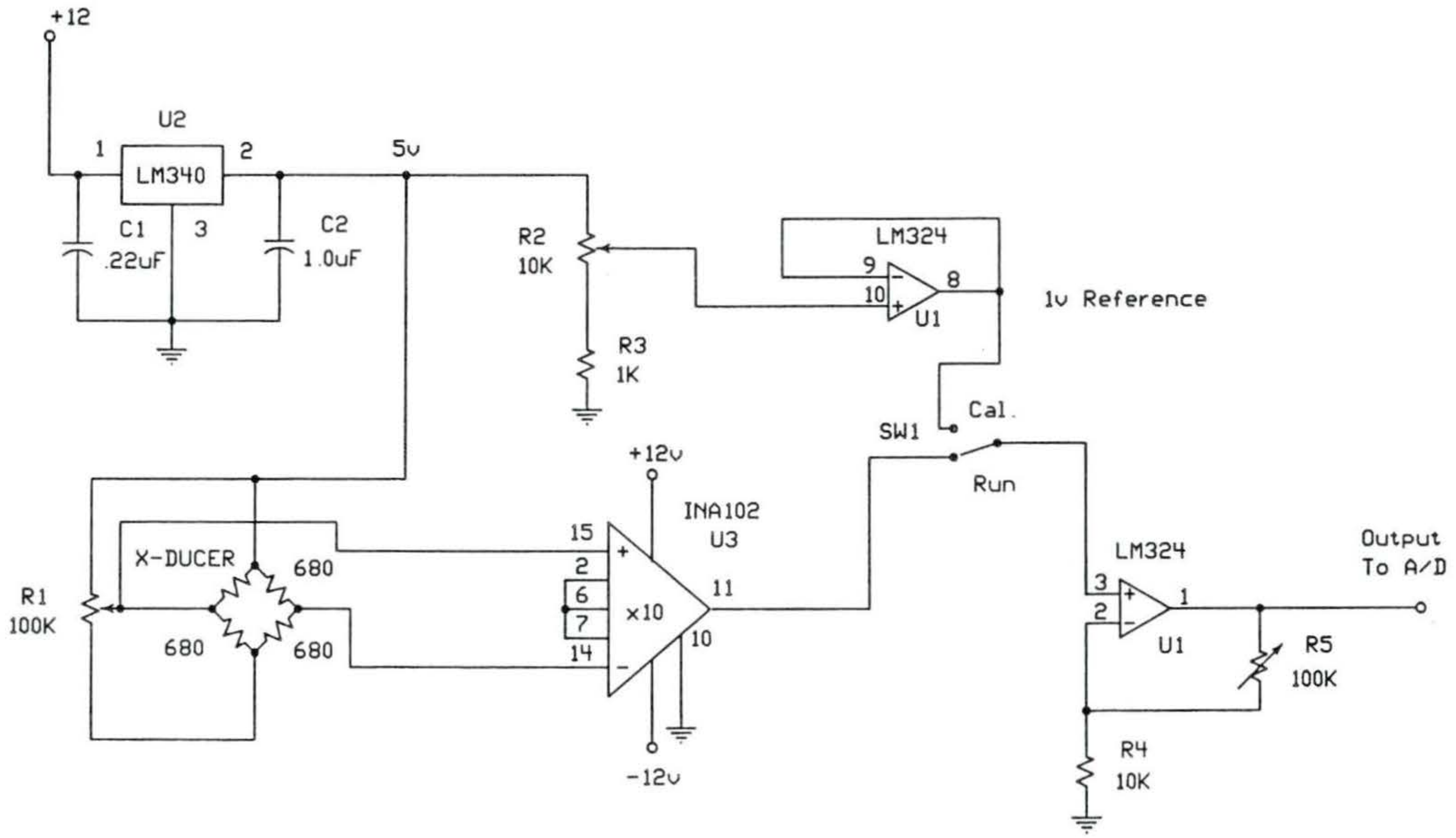
One of the most difficult tasks was developing a means to detect the change in volume of the aorta as fluid was pumped through it. From initial calculations, only a few microliters of change were expected. At this small level, it is difficult to detect volume change without affecting the pressure on the aorta. Ideally, a single transducer with an open end and no back pressure or resistance is desired. Multiple attempts at a transducer of this nature were made, but only two workable ideas resulted.

The first idea involved a small capillary tube, approximately 1mm in inside diameter and 10cm long, with two lengths of nichrome wire stretched through it and a small bead of mercury acting as a wiper as in a variable resistor. Fundamentally, this method worked. But, from a practical standpoint, it was unreliable. The nichrome wire was approximately 0.025 millimeters in diameter and had a resistance of 600 ohms per foot. The process of threading the wire through the capillary tube and securing it on each end was tedious, at best. If the capillary was successfully threaded, the next problem was attaching some sort of lead wires to the nichrome wire without breaking either the capillary tube or the nichrome wire.

The process was successfully completed on two transducers and volume detection was attempted. The detection circuit is shown in Figure 9. The transducer served as one leg of a Wheatstone bridge. Testing showed that linear output was produced but there was a significant amount of hysterisis due to the friction of the mercury contacting the walls of the capillary tube. A second major problem was due to contaminants in the system, especially the mercury. Any contaminants in the mercury or on the nichrome wire caused electrical discontinuities which caused the output of the Wheatstone bridge to force the instrumentation amplifier U3 into saturation. This, in turn, caused the final stage of amplification U1 to go into saturation and, therefore, meaningless data were produced. Even after several attempts at cleaning the system, contamination from the sea-water bath made the system unworkable.

The second system, which is currently in use, is still an impedance measuring method but from a different perspective. A 3.2 mm outside diameter capillary tube approximately 18 cm long with an internal diameter of 1.8 mm contains two lengths of thin gold wire stretched inside the tube. Gold wire was chosen because of its low electrical resistance and inert properties. Thus the sea-water bath

Figure 9. Volume detection circuit for nichrome wire transducer


+12
U2
1
LM340
2
5v
3
C1
.22uF
C2
1.0uF
R2
10K
R3
1K
LM324
9
10
8
U1
1v Reference
Cal.
SW1
Run
+12v
INA102
U3
15
2
6
7
14
x10
11
10
-12v
X-DUCER
680
680
680
R1
100K
LM324
3
2
1
U1
Output
To A/D
R5
100K
R4
10K

only will cause a change in resistance along the capillary tube, and will have minimal corrosive effects. The transducer is connected to an Impedance Pneumograph Coupler Type 7212 (Narco Bio-Systems, 1970) which is used to measure impedance between two electrodes. Typical applications of the Impedance Pneumograph Coupler Type 7212 include measurement of swallowing, change in muscle dimensions, pulse-wave velocities, cardiac outputs, eye movements, and gastro-intestinal motility (Narco Bio-Systems, 1970).

The Impedance Pneumograph Coupler Type 7212 passes a constant 50 KHz four microamp rms current through the electrodes. A voltage which is proportional to the impedance across the electrodes is amplified and processed for recording or digitizing (Narco Bio-systems, 1970). In this case, the output voltage is digitized simultaneously with the aortic pressure pulse. More on the digitization process will be covered later.

Operation of the volume detection system is as follows. After the test chamber is sealed, a micrometer syringe, filled with sea-water, is attached at point "D" in Figure 8. The micrometer syringe is used to calibrate and set the reference point on the volume detection transducer. This "set-point" is 2 cm out from the edge of the test chamber. Calibration is performed by moving a fixed amount of fluid into and out of the chamber via the micrometer syringe while recording the output from the Impedance Pneumograph Coupler Type 7212.

When the sea-water level is around the "set-point", the output of the Impedance Pneumograph Coupler Type 7212 is approximately linear. If the fluid level is moved out towards the end of the transducer, then the output from the Impedance Pneumograph Coupler Type 7212 becomes logarithmic. For this reason, the "set-point" is close to the chamber. If the operating point of the sea-water is any closer to the interior of the chamber, there is a risk of drawing air into the chamber as the aorta contracts.

Pressure Measurement

A Statham P23Db wire strain gage pressure transducer connected to a Beckman 411 amplifier via a Beckman 9853A Voltage/Pulse/Pressure coupler is used to measure and record the simulated aortic pressure. The 9853A coupler provides the necessary excitation voltage for the pressure transducer (Beckman Instruments, 1970a). It also has the necessary hardware to balance out any offset in the pressure transducer so that the output is zero when there is zero input pressure.

The 411 amplifier has a miniature phone jack on the bottom of it labeled "AUX. OUT". "AUX. OUT" provides a high-level output of the recorder input (Beckman Instruments, 1970b). This high-level output is calibrated for 2.83 volts peak-to-peak amplitude which would represent full-scale deflection on the chart recorder (Beckman Instruments, 1970b).

Some initial pressure recordings were made on the chart recorder but ultimately the data were digitized. These data, from the "AUX. OUT" of the 411 amplifier, were fed into the data acquisition system. The 9853A in conjunction with the 411 and a Statham pressure transducer is a very well established method of recording pressures. In this project, instead of using the chart recorder, the pressure data were digitized in conjunction with volume data.

Host Computer

A need for a data acquisition host and terminal device is served by a Zenith Z-158 Personal Computer. The Z-158 is fully IBM PC compatible, runs at 8MHz, has two 360 Kbyte floppy disk drives, 512 Kbytes of internal RAM, and has six open expansion slots for other devices. The data acquisition system is a Metrabyte "board-level" type of system coupled with a Dataq Instruments real-time data acquisition system. Also, the Microsoft Windows operating system was used so that multiple applications could be accessed from one PC. Therefore, to talk to the BCC52, the terminal emulator provided with Microsoft Windows is used.

Terminal emulator

The BCC52 is very simple to communicate with if the hardware and software of the host device can be configured to talk without any hardware or software "handshaking". The Z-158 is capable of communicating without the hardware "handshaking" and the terminal emulation software with Microsoft Windows, called Terminal, is able to perform without any software "handshaking" (Microsoft, 1985a). One disadvantage to using Terminal is that the user has to know the basics of the Microsoft Windows operating system. The advantage, though, is that the Microsoft Windows operating system allows multiple applications to be open at the same time. This must not be confused with multitasking, where multiple applications can run simultaneously. The specifics of the Microsoft Windows operating system will be left for the interested reader and can be found in the Microsoft Windows User's Manual (Microsoft, 1985b).

The "Terminal" program allows for many different emulation and communication parameter options along with the ability to upload, download, and capture files. For this project, the default VT52 terminal type was chosen along with the following communication parameters: 7 data-bits, 1 stop-bit, no parity, no handshaking, direct computer connection.

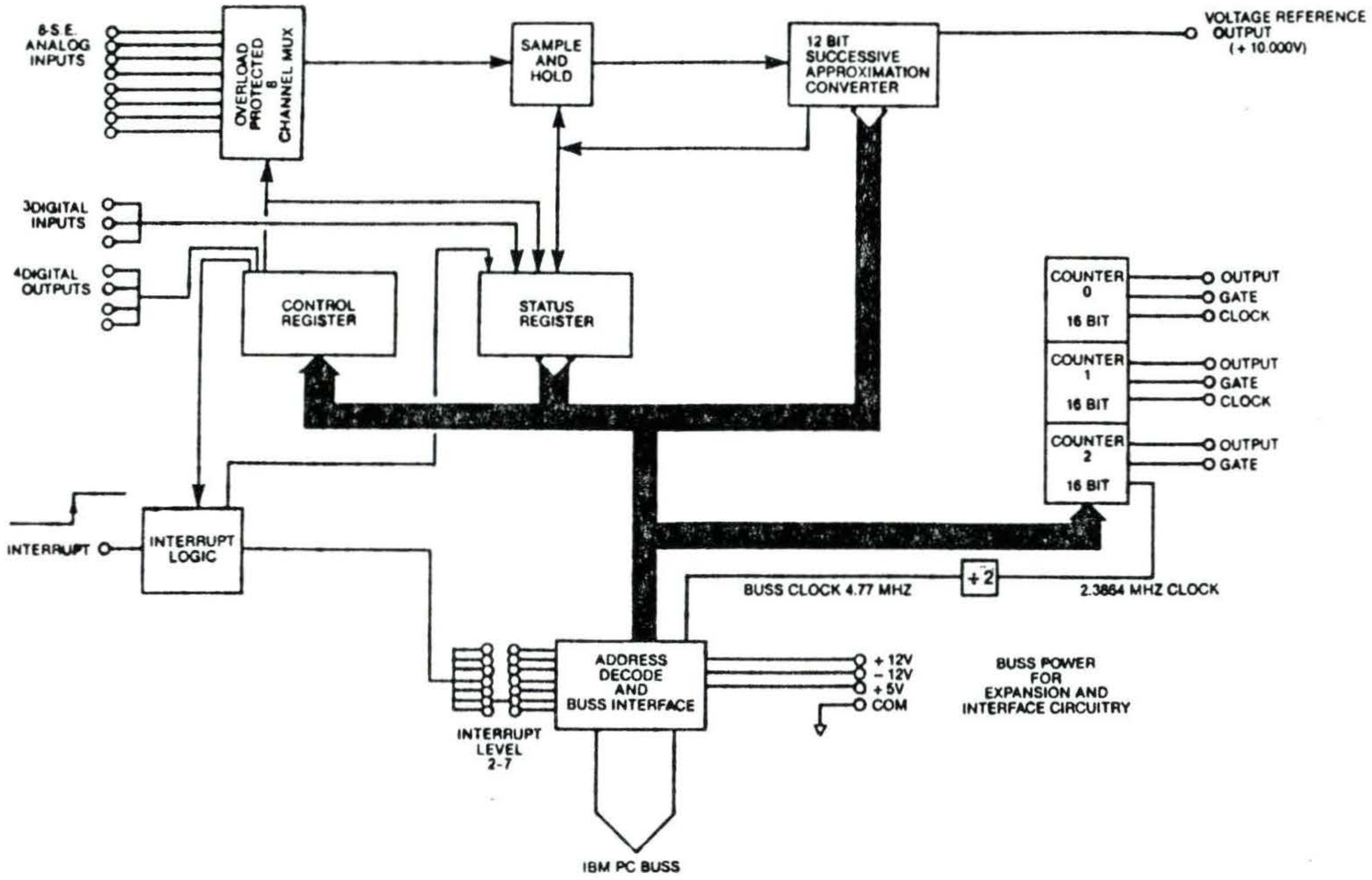
Data acquisition

The data acquisition system runs in a separate window on the Z-158. A combination of a Metrabyte Dash-8 data acquisition board and a Dataq Instruments WFS-200 real-time display board allow real-time acquisition and graphical display of up to eight channels at one time.

Dash-8 The Dash-8 is an 8-channel, 12-bit high-speed A/D converter and timer/counter board for the IBM PC (Metrabyte, 1984). A block diagram of the Dash-8 architecture is shown in Figure 10. The eight A/D input lines are multiplexed and fed to a single 12-bit successive approximation A/D converter with sample and hold. Average conversion time is 25 microseconds with a worst case time of 35 microseconds. This will provide a one-channel sampling rate of up to 30,000 samples/second as long as the overhead time for the controlling software is minimal. The maximum resolution for each channel is 2.44 millivolts at full scale input of +/- 5 volts. Each input is single-ended and can withstand a continuous +/- 30 volt overload (Metrabyte, 1984).

Other features on the Dash-8 include an Intel 8253 Programmable Counter/Timer, 4-bits of digital output lines, 3-bits of digital input lines, a precision 10vdc voltage reference, and an external interrupt input. These features are not employed in this project.

Figure 10. Dash-8 architecture block diagram (Metrabyte, 1984)


8-S.E. ANALOG INPUTS
OVERLOAD PROTECTED 8 CHANNEL MUX
SAMPLE AND HOLD
12 BIT SUCCESSIVE APPROXIMATION CONVERTER
VOLTAGE REFERENCE OUTPUT (+10.000V)
3DIGITAL INPUTS
4DIGITAL OUTPUTS
CONTROL REGISTER
STATUS REGISTER
COUNTER 0 16 BIT
OUTPUT
GATE
CLOCK
COUNTER 1 16 BIT
OUTPUT
GATE
CLOCK
COUNTER 2 16 BIT
OUTPUT
GATE
INTERRUPT
INTERRUPT LOGIC
BUSS CLOCK 4.77 MHZ
÷2
2.3864 MHZ CLOCK
ADDRESS DECODE AND BUSS INTERFACE
+12V
−12V
+5V
COM
BUSS POWER FOR EXPANSION AND INTERFACE CIRCUITRY
INTERRUPT LEVEL 2-7
IBM PC BUSS

Also provided with the hardware is a floppy disk of utility programs to run the Dash-8. These programs include routines to install the hardware, for calibration and test, for data linearization, for graphical display, and an I/O driver for operating all the functions from BASIC or machine language (Metrabyte, 1984). For this project, control over the Dash-8 was through the CODAS software system. A more detailed discussion of all the features of the Dash-8 are in the Dash-8 Manual (Metrabyte, 1984).

CODAS In order to obtain and display multiple channels of A/D data, a WFS-200 Waveform Scroller board (Dataq Instruments, 1987) and the CODAS software is used. The WFS-200 Waveform Scroller board works as an add-on to the system video card. The video data from the system video board are sent to the WFS-200 Waveform Scroller board which is then connected to the monitor. Through the CODAS software, there are ten different display formats available. A listing of these and the number of channels that each can display is as follows.

DISPLAY FORMAT NUMBER	WINDOW DESCRIPTION	MAXIMUM DISPLAYABLE CHANNELS
1	One full screen window.	1
2	One full screen widow; dual overlapping.	2

3	Two vertical windows; non-overlapping.	2
4	Two vertical windows; dual overlapping.	4
5	Two horizontal windows; non-overlapping.	2
6	Two horizontal windows; dual overlapping.	4
7	Four quarter screen windows; non-overlapping.	4
8	Four quarter screen windows; dual overlapping.	8
9	Four horizontal windows; non-overlapping.	4
10	Eight screen windows; non-overlapping.	8

Each of the display formats is designed to optimize both the horizontal and vertical resolution of the data being displayed (Dataq Instruments, 1987).

As mentioned earlier, the CODAS system is a real-time system. This means that the display and data acquisition process are all taking place in real time. Using the Z-158 and the Dash-8, the maximum sampling rate for two channels is 3 KHz. Data can also be saved to disk during real-time data acquisition. Control of the system is through a set of keyboard commands; most of which are function keys. A list of the keystrokes and their corresponding function is shown below (Dataq Instruments, 1987).

KEYSTROKE	FUNCTION PERFORMED
F1	Select scroll or oscilloscope mode.
F2	Enable ADC channels.
F3	Oscilloscope mode triggering.
F4	Freeze display.
F5	Waveform compression/Time base enable.
F6	Save default conditions.
F7	Screen annotation control.
F8	Options menu (Alarm, Remote event, Remote store).
F9	Disable storage to disk.
F10	Enable storage to disk.
A	Enable/disable alarm function.
H	Enable HELP menu.
Q	Exit CODAS and return to DOS.
SHIFT & Num.	Select display format.
Number	Enable waveform window to ADC channel.
Num. = Num.	Assign waveform window to ADC channel.
Space bar	Event marker.
Right or Left Arrow	Sample rate selection/waveform compression.
Up or Down Arrow	Scale selected channel.
PgUp & PgDn	Offset selected channel.

HOME & END   Program ADC gain.

Once the data are collected using the CODAS program, a second software package, POSTACQ, is used to post process the data, review it, or port it to other software packages. POSTACQ is a complimentary program to CODAS and is part of the CODAS system.

POSTACQ   The post acquisition package uses a set of keystrokes, simular to those of CODAS, to control the post processing functions. POSTACQ allows the user to measure the exact time between any two points in the file and the amplitude of each point. Calibration constants can also be inserted in the file so that data are read out in specified units instead of volts. This is very useful when digitizing pressures because once the calibration constants are set, all the data are read out in cm of water or mm of mercury, etc. Another useful function is the Copy and Paste function.

Copy and Paste allows user selected portions of the file to be copied and pasted into separate data files in five different storage formats. These storage formats include (Dataq Instruments, 1987):

1) ASCII (Sequential)
2) Binary (with CODAS header)
3) Binary (without CODAS header)

4) ASYST/ASYSTANT

5) LOTUS

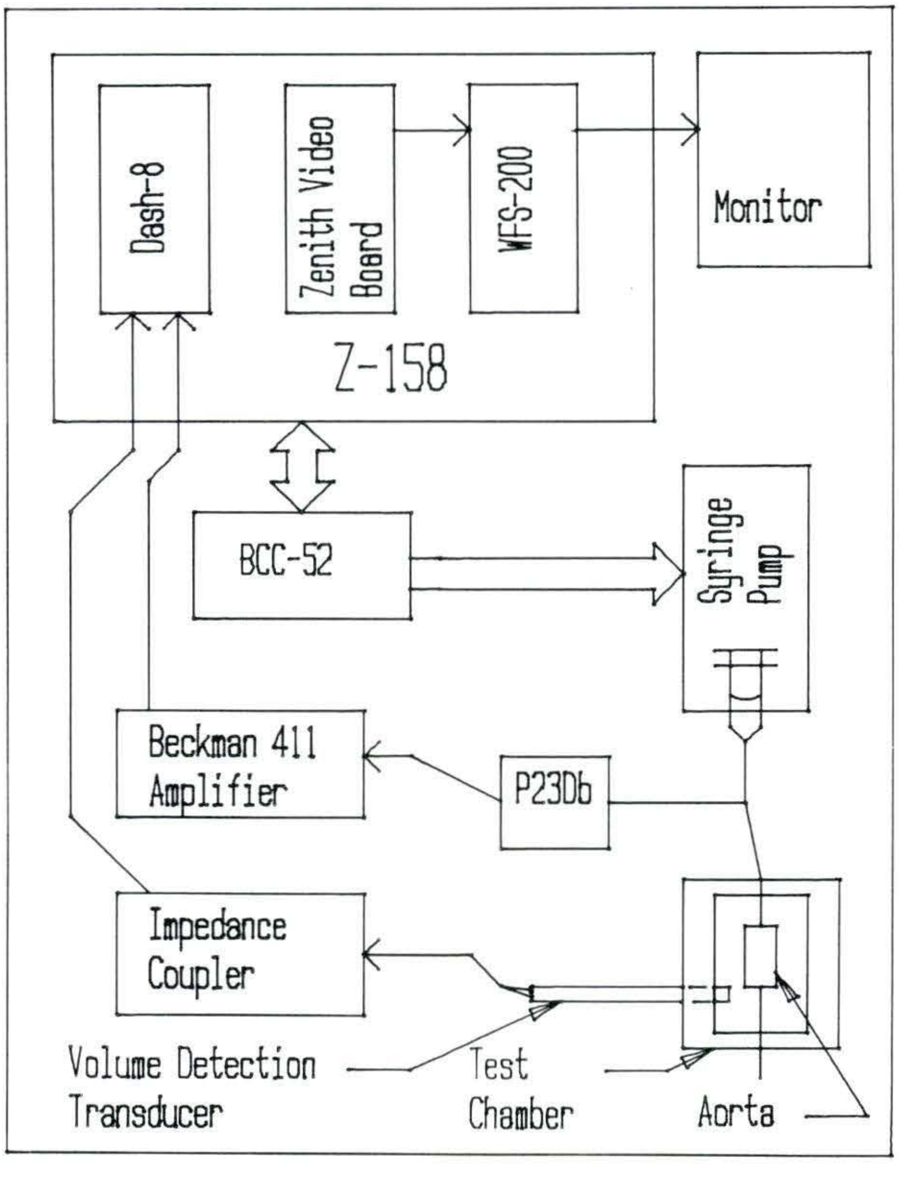
The LOTUS format was used in this project for producing a file which could then be loaded into a LOTUS 1-2-3 worksheet. Once in a worksheet, hardcopy output of selected waveforms or data analysis could be performed. This also provides a form for porting the data to other analytical software requiring the LOTUS file format.

Complete System

Using all the above subsystems, a complete system to mimic and record pressure and flow characteristics of blood driven through the aorta of a *Limulus polyphemus* has been constructed. Figure 11 shows a block diagram of the system. There are two computers running simultaneously. The BCC52 which is controlling the syringe pump and the Z-158 which is performing data acquisition and/or terminal emulation. There is also the Beckman pressure transducer system and the Narco Bio-Systems impedance measuring system.

Initially the Z-158 is booted with MS-DOS and the Microsoft Windows Operating system is loaded. In the first window, the TERMINAL program is loaded and communication is initiated with the BCC52. At this time, the BCC52 is powered up but the syringe pump is still unpowered.

Figure 11. Complete Analysis System Block Diagram


Dash-8
Zenith Video Board
WFS-200
Monitor
Z-158
BCC-52
Syringe Pump
Beckman 411 Amplifier
P23Db
Impedance Coupler
Volume Detection Transducer
Test Chamber
Aorta

Appendix E shows the BCC52 computer system and interface hardware. The BASIC program for the BCC52 has been saved in such a manner that, at power up, it will be run automatically. Also the assembly language object code has been saved in an EPROM so that when it is plugged into the third memory socket (4000H-5FFFH) of the BCC52 the BASIC program can call it directly. This feature prevents the user from having to move the object code into RAM, from an EPROM, before running the BASIC program. All the user has to do is answer the prompts from the BASIC program for: the number of stepper-motor pulses per heart beat, the systolic time (in milliseconds), and the heart rate (in beats/min.). The adjusted parameters will be printed back out on the consol device for the user and stepper-motor control sequences will begin.

The syringe pump is started and stopped in either of three ways. The power switch on the syringe pump is used to control the power supply for the stepper motor. So it must be "on" in order for the motor to function. The other two methods are used primarily to stop the pump. If the syringe is almost empty then the "empty syringe" detection circuit will trigger an external interrupt and the software will halt the syringe drive. Also, a "Run/Stop" switch is connected to port C of the 8255 for control over the program. The simple connection of this switch is also shown in Figure 5.

With the syringe pump control circuit initialized, the next step is to open a second window for the data acquisition system. This window contains the CODAS software and will display the data which are being acquired through the Dash-8. When opening the window, the size of the data file to be collected must be specified (in Kbytes) along with a file name. The real-time display will begin as soon as the file name and size is specified. This is very useful for adjusting the pressure and volume measuring systems because you can see the digitized input on the screen. Data collection is controlled by the user through CODAS commands.

Now, the back pressure ("peripheral resistance") against the aorta is adjusted by raising or lowering the end of the outlet tube along a vertical meter stick. This corresponds to the static hydrostatic peripheral resistance in centimeters of water. The volume detection system is adjusted with the micrometer syringe so that the liquid column in the volume detection cappillary tube is at a set-point approximately two cm out from the chamber.

The Beckman pressure transducer coupler and amplifier are powered up and adjusted for optimal output. The coupler is balanced so that at zero input there is zero output. The Impedance Pneumograph Coupler Type 7212 is also powered up. It is set to the "Direct Mode" and its balance control is set so that the output is at a minimum on the data acquisition display.

Turning the power for the syringe pump "on" starts fluid flow through the aorta. Minor adjustments can be made to either the Beckman pressure transducer amplifier or the Impedance Pneumograph Coupler Type 7212 as the effects are monitored on the Z-158 monitor through the CODAS real-time data acquisition system software. Once the desired data are being displayed, data collection is started with the F10 function key and stopped with the F9 key or when the file is full.

If any changes are to be made to the simulation parameters, the BASIC program has to be halted and the TERMINAL window must be made the active window. Now communications with the BCC52 are reinstated and the simulation program can be rerun by typing "RUN". The CODAS data acquisition program is restarted by the same procedure as discribed above. Once the pressure and volume measuring systems are initialized they generally don't require any significant readjustment during an analysis session.

RESULTS

A set of sample aortic pressure waveforms from a live *Limulus polyphemus* is shown in Figure 12. To obtain these waveforms, a hole the size of a number 28 hypodermic needle was drilled through the shell of a *Limulus polyphemus* in the approximate location of the aortas. Then, a number 28 needle connected to a Statham P23Db pressure transducer with small PE tubing was carefully inserted through the hole and carefully probed until the pressure pulse was detected on the CODAS real-time data acquisition display. The animal, in a sea-water tank, was given a few minutes to settle down after the needle was inserted, then a five-minute record of pulses was collected. Three of the best waveforms are shown in the figure. Each represents a different heart rate; 8.5, 15, and 18 beats per minute. Presumably the differences in heart rate are due to changes in physical activity and/or the physiological state of the animal. Short-term recordings were made in order to avoid the problem of blood coagulation plugging the catheter needle. This procedure was performed with the assistance of Dr. James Redmond and Mary Johnson, Iowa State University.

Figures 13, 14, and 15 are recordings of simulated aortic pressures produced and recorded using the computerized system described previously. Simulated heart rates shown in

Figure 12. Aortic Pressure. Sample waveforms taken from live *Limulus polyphemus* [A - heart rate of 8.5 beats/minute, B - heart rate of 15 beats/minute, C - heart rate of 18 beats/minute]

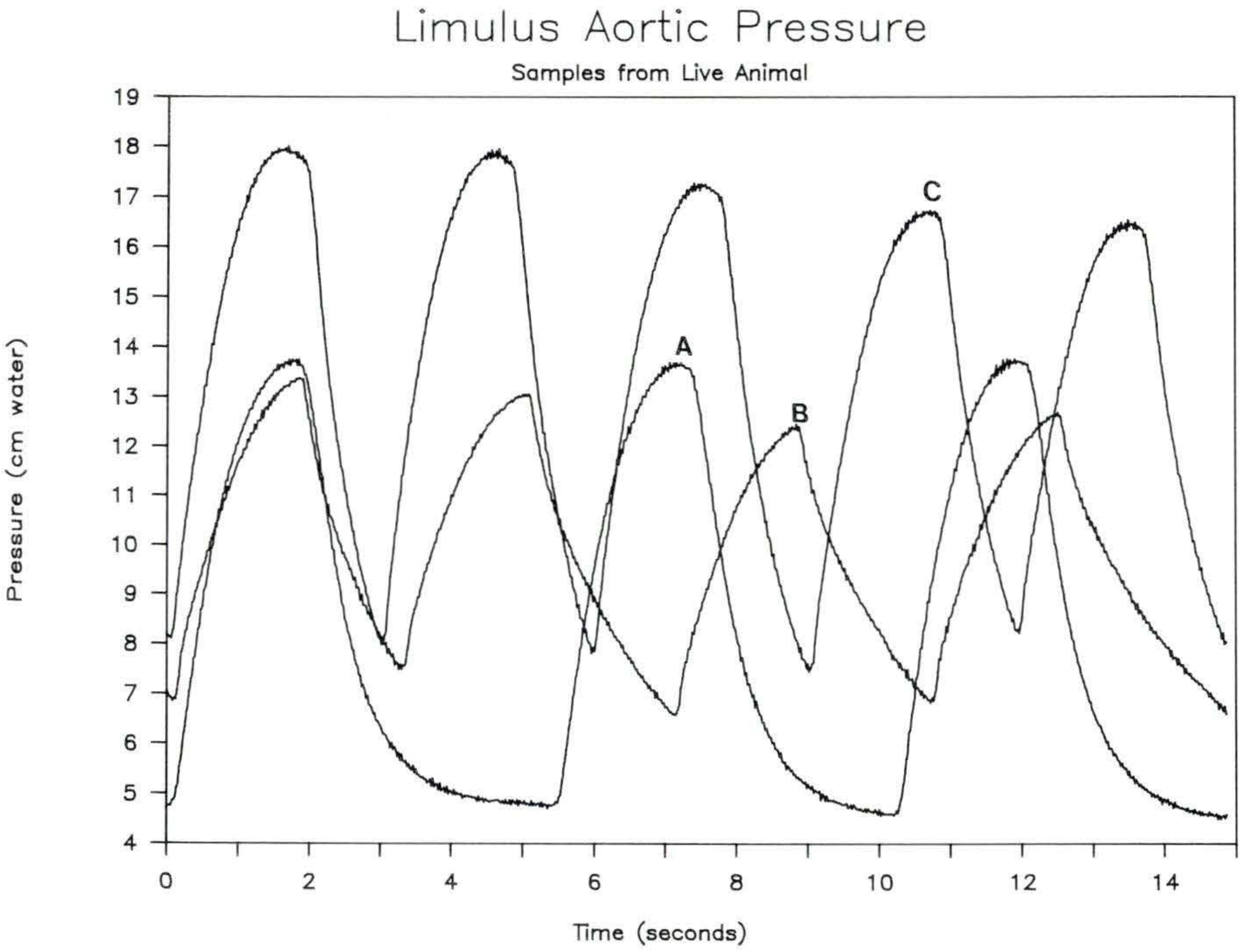

Limulus Aortic Pressure
Samples from Live Animal
Pressure (cm water)
Time (seconds)
A
B
C
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
0
2
4
6
8
10
12
14

Figure 13. Aortic Pressure. Sample waveforms taken from simulation system. Simulated heart rate is 20 beats/minute [A - 25 stepper-motor pulses/heart beat, B - 50 stepper-motor pulses/heart beat, C - 75 stepper-motor pulses/heart beat]

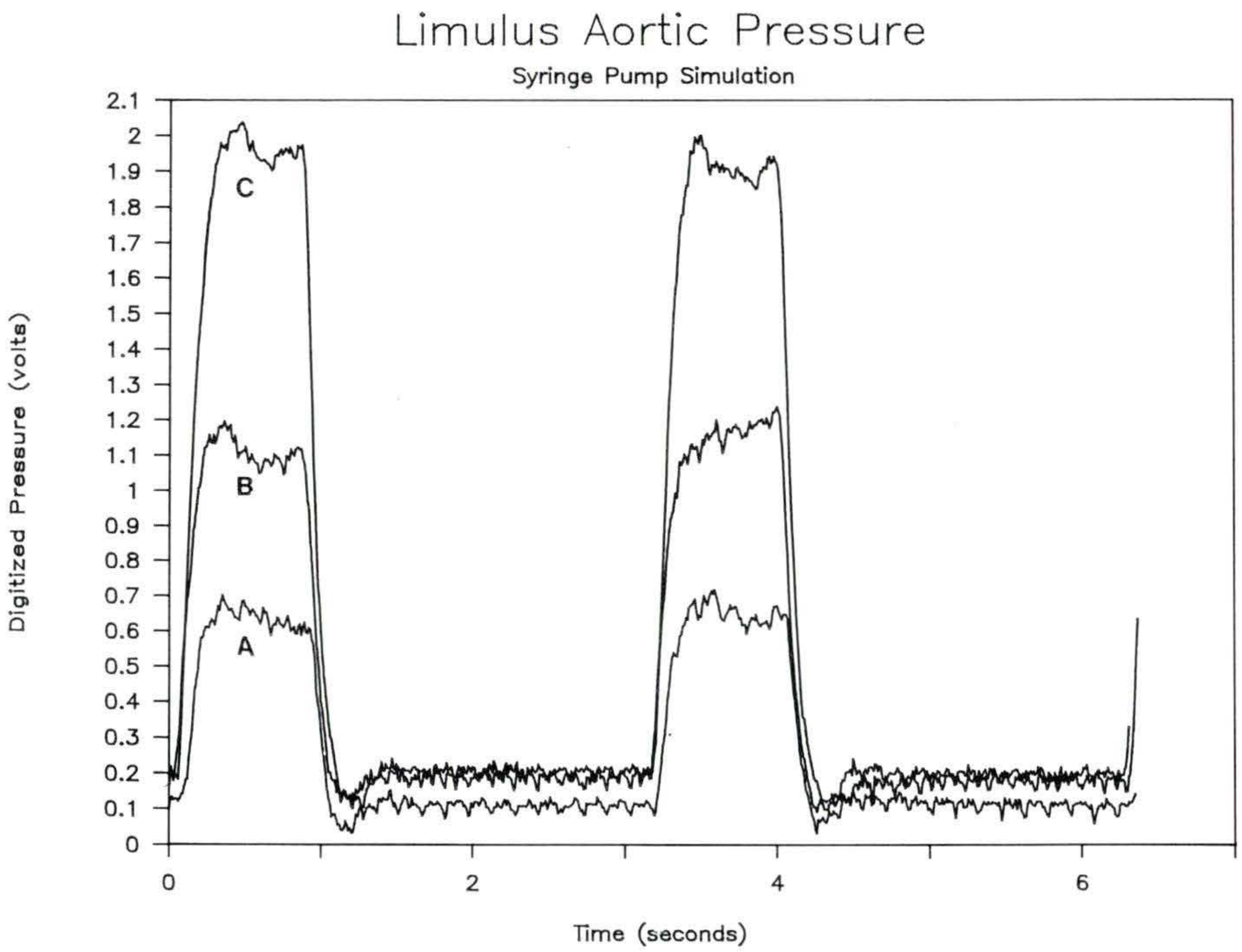

Limulus Aortic Pressure
Syringe Pump Simulation
Digitized Pressure (volts)
2.1
2
1.9
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
C
B
A
0
2
4
6
Time (seconds)

Figure 14. Aortic Pressure. Sample waveforms taken from simulation system. Simulated heart rate is 40 beats/minute [A - 25 stepper-motor pulses/heart beat, B - 50 stepper-motor pulses/heart beat, C - 75 stepper-motor pulses/heart beat]

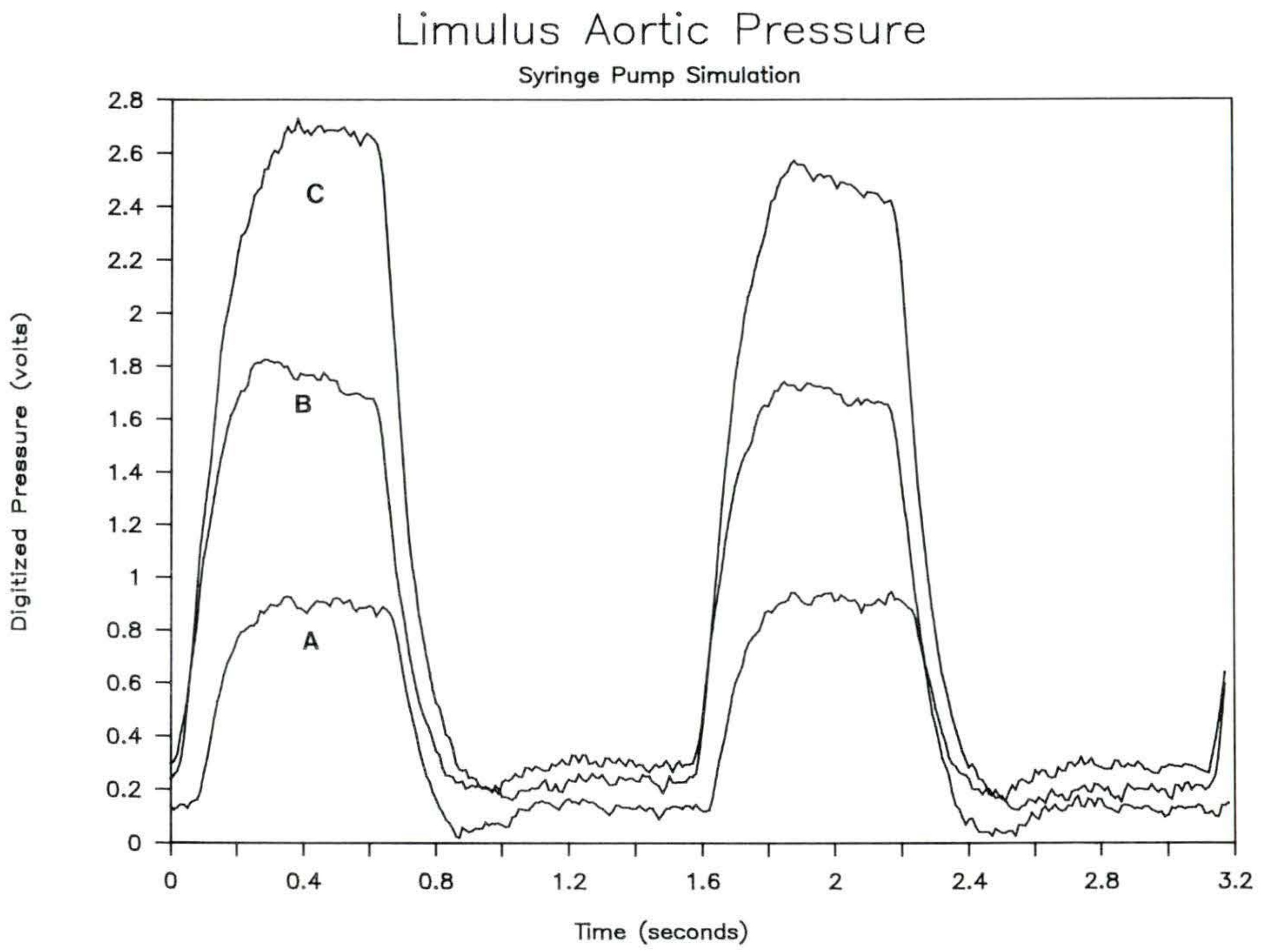
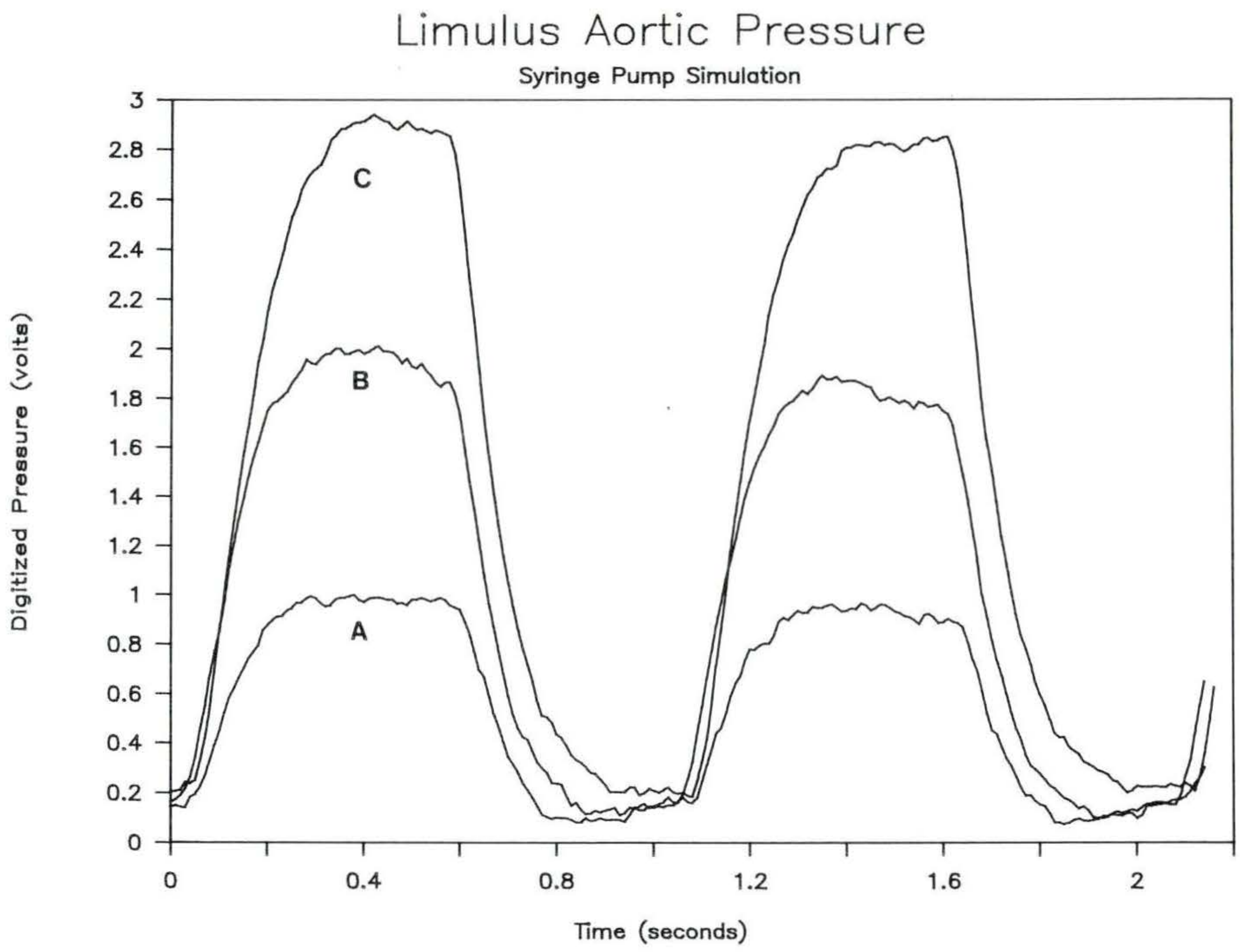

Limulus Aortic Pressure
Syringe Pump Simulation
Digitized Pressure (volts)
Time (seconds)
A
B
C
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
0.4
0.8
1.2
1.6
2
2.4
2.8
3.2

Figure 15. Aortic Pressure. Sample waveforms taken from simulation system. Simulated heart rate is 60 beats/minute [A - 25 stepper-motor pulses/heart beat, B - 50 stepper-motor pulses/heart beat, C - 75 stepper-motor pulses/heart beat]



Limulus Aortic Pressure
Syringe Pump Simulation
Digitized Pressure (volts)
Time (seconds)
C
B
A
3
2.8
2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
0.4
0.8
1.2
1.6
2

these figures are 20, 40, and 60 beats per minute. At each heart rate, three different waveforms are presented. Each different waveform was produced by varying the number of stepper-motor pulses per systolic cycle. Increased cardiac output as a function of the number of stepper-motor pulses per systolic cycle is distinctly shown in each of these figures.

The shape of the downward side of the pressure curve in the simulated waveform is generally steeper than the characteristics shown in the live animal. It was found that by varying the size, length, and material of the tubing which connects the syringe to the test chamber the shape of the pressure curve can be greatly altered. The rubber tube being used now provides the best all around pressure waveforms. This is one area in which more investigation may prove helpful.

A second point to note about these curves deals with the units on the ordinate axis of the figures. The units are in volts not cm of water or mm of mercury. These are digitized waveforms collected at a sampling rate of 200 samples per second for two channels, pressure and volume, which is effectively 100 samples per second per channel. The reduction in sampling rate for multiple channels is due to the characteristics of the Dash-8 data acquisition board which has a single multiplexed D/A for all eight analog input channels. Calibration constants could have been added to the data set in order to represent the data in units of pressure instead of volts but were not because the main goal of this data set was to verify timing and function of the BCC52 software.

One of the main goals of this system was to provide the data from which stress vs. strain curves for the *Limulus polyphemus* aorta, under varying dynamic conditions, could be produced. The data for these curves are obtained from the pressure and volume curves acquired during specific simulations of natural conditions on an aorta. Three sets of pressure and volume curves are shown in Figures 16, 17, and 18. The heart rate for each of these curves is 60 beats per minute. Increased cardiac output is clearly demonstrated as the number of stepper-motor pulses per systolic cycle is increased.

Again, the data are represented in its raw digitized form and not in a calibrated, true units form. If phase shift was all that was desired from the data, it would not matter what form the data were in as long as the time base for the pressure and volume data were the same. After careful examination of the data acquisition system, it was determined that each channel of a data is not sampled simultaneously. This is a result of the multiplexed A/D design of the data acquisition board itself. At the sampling

Figure 16. Aortic Pressure/Volume Curves. Sample waveforms taken from the simulation system. Simulated heart rate is 60 beats/minute. Stepper-motor rate is 25 pulses/beat [A - pressure waveform, B - volume waveform]

Limulus Aortic Pressure/Volume Curves

Syringe Pump Simulation

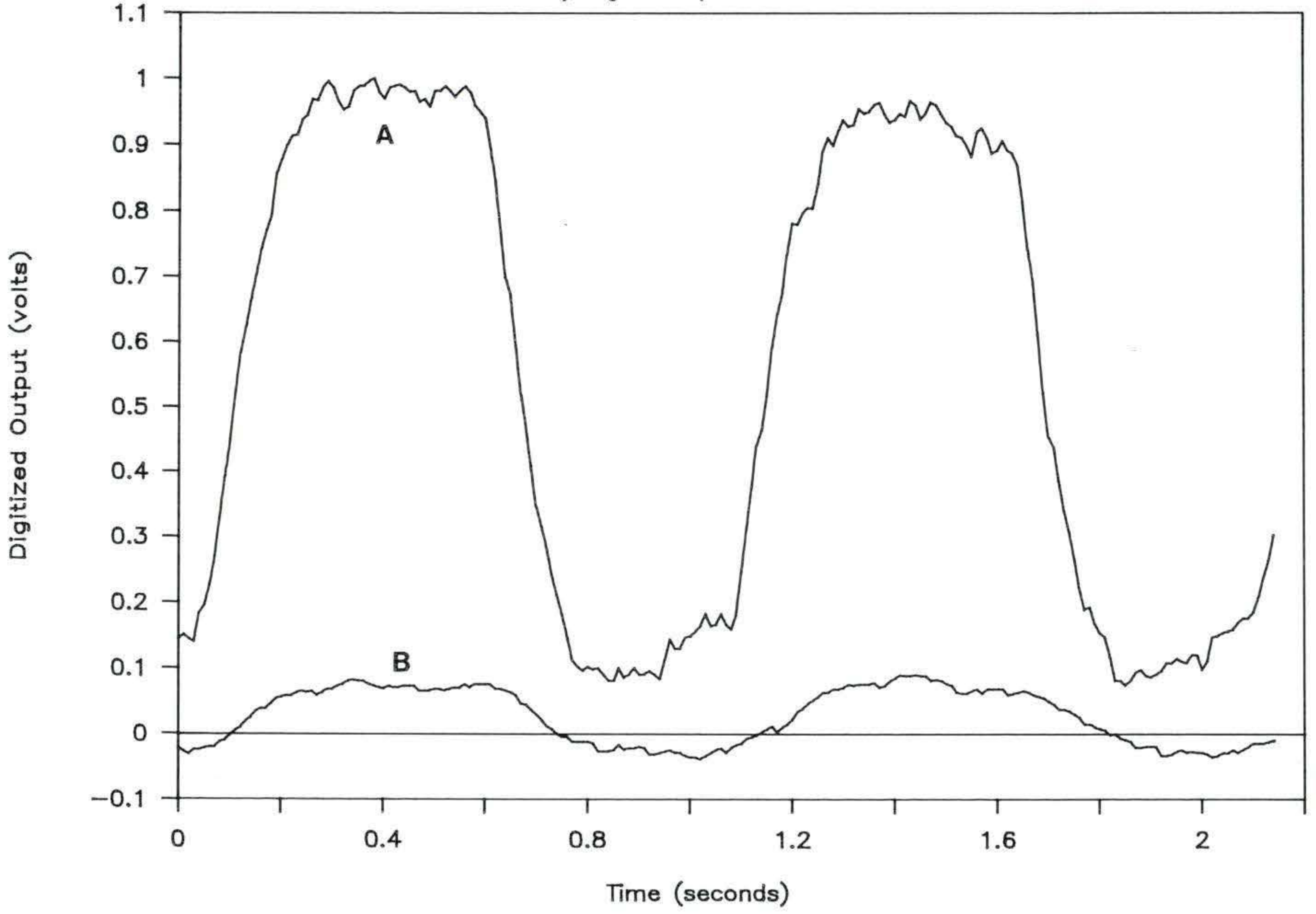


Figure 17. Aortic Pressure/Volume Curves. Sample waveforms taken from the simulation system. Simulated heart rate is 60 beats/minute. Stepper-motor rate is 50 pulses/beat [A - pressure waveform, B - volume waveform]

Limulus Aortic Pressure/Volume Curves

Syringe Pump Simulation

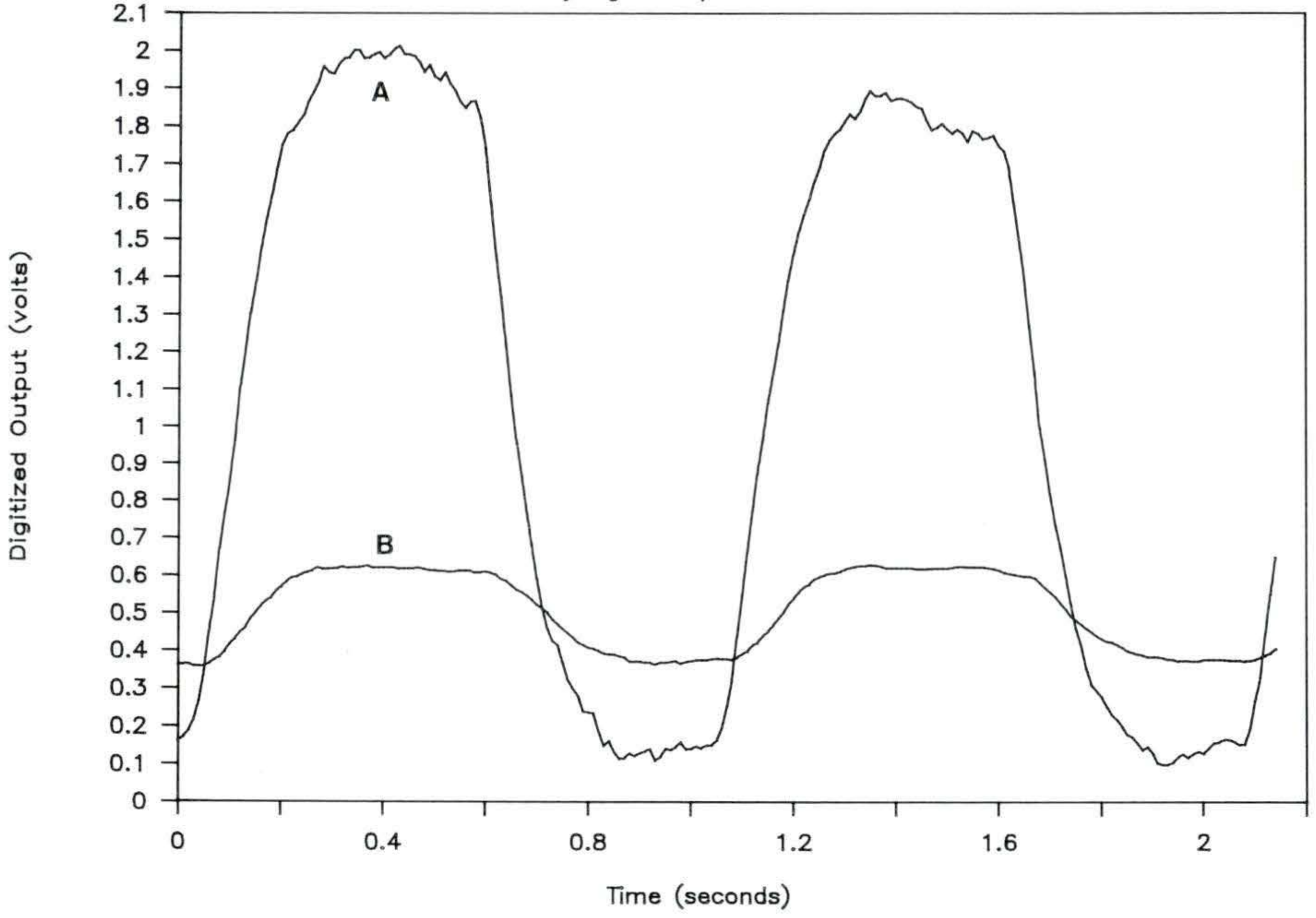
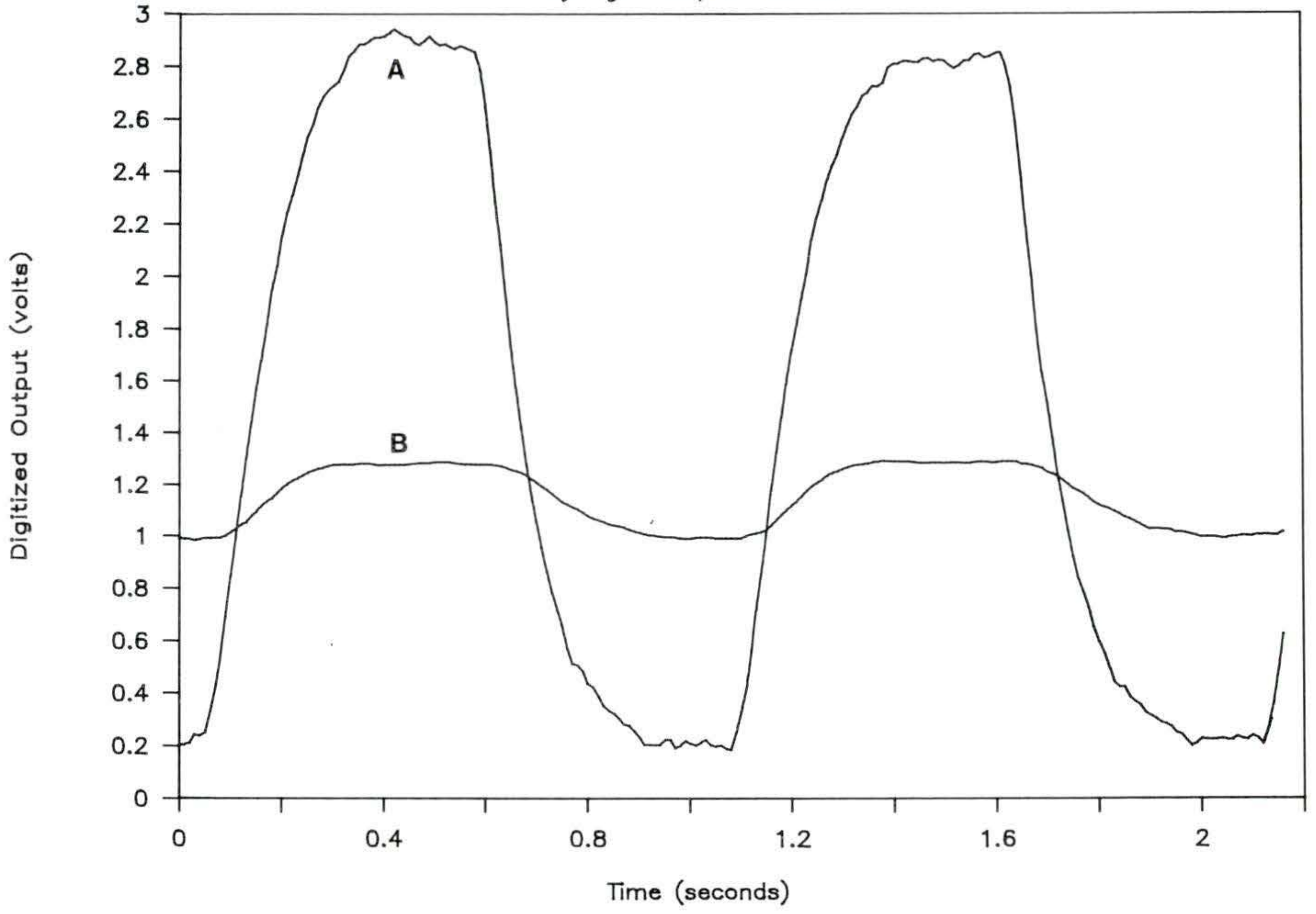


Figure 18. Aortic Pressure/Volume Curves. Sample waveforms taken from the simulation system. Simulated heart rate is 60 beats/minute. Stepper-motor rate is 75 pulses/beat [A - pressure waveform, B - volume waveform]

Limulus Aortic Pressure/Volume Curves

Syringe Pump Simulation



rate used, 200 samples per second, the timing error between channels will be 5 milliseconds. The effective sampling rate for each channel is 100 samples per second. It will take more experimentation to determine whether or not this phenomenon has a significant effect on the data from the pressure/volume curves collected with this system.

RECOMMENDATIONS FOR FURTHER STUDY

As with any project, there is always a look back to evaluate what has been done. This project was interesting and proved to be a good adaptation of computers in the ongoing study of the properties of the *Limulus polyphemus* circulatory system, especially the *Limulus polyphemus* aorta. From an engineering standpoint, there are some areas of interest that time has not allowed for further study.

Since receiving the documentation from the syringe pump manufacturer (Sage Instruments, 1975) an easier method of modifing the pump has come to mind. Appendix F shows an alternative proposal for interfacing to the stepper motor inside the Sage Instruments syringe pump.

One area that would be a project in itself is the study of the spectral characteristics of the aortic pressure waveforms produced by the *Limulus polyphemus* and by the simulation system. This would show how closely the simulation is to the real world. It would also provide a firm basis for advance stepper-motor control software. If advanced software was written, a comparison between the pressure/volume curves produced with the present software and the new software could determine which frequencies in the pressure pulse are the most significant in the stress/strain curves of the Limulus polyphemus aorta.

Another stepper-motor control software modification might include some form of incremental pumping towards the end of the systolic cycle. So after the peak of the cycle, the time between stepper-motor pulses would be increased logarithmically to vary the downward shape of the systolic pressure pulse.

Another volume detection system might be based upon an infra-red beam passed through the chamber and concentrated on the area of the aorta. On the other side of the chamber would be a detector for the infra-red energy which is transmitted through the aorta.

Finally, to improve the shape of the simulated pressure curve further revisions of the fluid delivery system between the syringe pump and the test chamber could be tried. This could include variations on the material, diameter, and length of the tubing. Also a "T" could be inserted between the syringe pump and the test chamber with a length of tubing which is clamped off at the end. This would serve as a compliance capacitor and by varying the distance between the "T" and the clamp different amounts of capacitance could be added.

BIBLIOGRAPHY

Airpax. 1982. Stepper Motor Handbook. Airpax Corp., Cheshire, Connecticut.

Beckman Instruments. 1970a. Operator Manual for Type 9853A Voltage/Pulse/Pressure Coupler. Beckman Instruments Inc., Schiller Park, Illinois.

Beckman Instruments. 1970b. Operator Manual for Type R-411 Dynagraph Recorder. Beckman Instruments Inc., Schiller Park, Illinois.

Conner, Margery S. "Analog-I/O boards and software for IBM PCs." Engineering Design News, 12 June 1986, pp. 116-137.

Dataq Instruments. 1987. Codas User's Manual. Dataq Instruments, Inc., Akron, Ohio.

Intel. 1985a. MCS BASIC-52 Users Manual. Intel Corp., Santa Clara, California.

Intel. 1985b. Microcontroller Handbook. Intel Corp., Santa Clara, California.

Intel. 1987. Microprocessor and Peripheral Handbook. Volume II. Intel Corp., Santa Clara, California.

Judd, Robert, and Berry Phillips. "Personal computers take on data acquisition and control." Electronic Products, 3 March 1986, pp. 40-44.

Keithley Instruments. Choosing PC-Based Data Acquisition Hardware. Keithley Instruments Inc., Data Acquisition and Control Division, 28775 Aurora Road, Cleveland, Ohio.

Metrabyte. 1984. Dash-8 Manual. Metrabyte Corp., Taunton, Massachusetts.

Microsoft. 1985a. Microsoft Windows operating environment. Desktop Applications User's Guide. Microsoft Corp., Bellevue, Washington.

Microsoft. 1985b. Microsoft Windows operating environment. User's Guide. Microsoft Corp., Bellevue, Washington.

Moyer, Curtis D. 1982. AN-876 Using Power MOSFETS in Stepping Motor Control. Motorola Semiconductor Products Inc., Phoenix, Arizona.

Narco Bio-Systems. 1970. Impedance Pneumograph Coupler Type 7212 Manual. Narco Bio-systems Inc., Houston, Texas.

National Semiconductor. 1980. Linear databook. National Semiconductor Corp., Santa Clara, California.

Redmond, J. R., D. D. Jorgensen, and G. B. Bourne. 1982. Circulatory physiology of Limulus. Pages 133-146 in J. Bonaventura, C. Bonaventura, and S. Tesh, eds. Physiology and biology of horseshoe crabs. Studies on normal and environmentally stressed animals. Alan R. Liss, New York.

Sage Instruments. 1975. Discrete rate syringe pump models 351 and 352 service manual. Sage Instruments, Division of Orion Research Inc., Cambridge, Massachusetts.

Texas Instruments. 1983. Display Driver Handbook. Texas Instruments, Dallas, Texas.

The Micromint. 1984. BCC52 BASIC Computer/Controller Users Manual. The Micromint, Inc., Vernon, Connecticut.

The Micromint. 1986. BCC52 Extensions Manual, ROM A, ROM A+B. The Micromint, Inc., Vernon, Connecticut.

APPENDIX A: SYRINGE PUMP STEPPER MOTOR MAIN CALLING PROGRAM

```
4       REM BASIC PROGRAM WHICH CALLS ASSEMBLY LANG. MOTOR
           DRIVER 6-24-88

6       FOR I=1 TO 24 :  PRINT "  " :  NEXT I

10      INPUT "INPUT NUMBER OF MOTOR PULSES PER BEAT <IE.50>
           ",MP

20      REM CALCULATE LIMITS AND INCREMENT FOR SYSTOLIC INPUT
           PROMPT

30     LLIMIT=MP*0.5

35     UPLIM=0.5*MP*255

40      PRINT "  "

50      PRINT "SYSTOLIC TIME MUST BE
           BETWEEN",LLIMIT,"AND",UPLIM,"MILLISECONDS"

60      PRINT " (",LLIMIT,"MILLISECOND INCREMENTS)"

70      PRINT "   "

80      INPUT "INPUT SYSTOLIC TIME IN MILLISECONDS  ",MS

90      PRINT "  "

95      REM CALCULATE MAXIMUM HEART RATE FROM ABOVE PARAMETERS

100    MXHTRT=1000*60/MS

105     REM CALCULATE MINIMUM HEART RATE FROM ABOVE PARAMETERS

110    MINHTRT=60/((MS+2550)/1000)

120    JUNK=MINHTRT-INT(MINHTRT)

130     IF JUNK<>0 THEN MINHTRT=INT(MINHTRT)+1

140     PRINT "WITH ABOVE PARAMETERS, HEART RATE RANGE IS
           BETWEEN ",MINHTRT,"AND",

150     PRINT MXHTRT,"BEATS / MIN"
```

```
160    INPUT "INPUT HEART RATE (BEATS / MINUTE) ",HTRT

170    IF (HTRT>MXHTRT).OR.(HTRT<MINHTRT) THEN  PRINT "    " :
          GOTO 140

175    REM CALCULATE TIME BETWEEN MOTOR PULSES (TBMP)

180   TBMP=MS/MP

185    REM 190-240 ENSURE THAT TBMP IS DIVISABLE BY .5
          MILLISECOND

190   JUNK=TBMP-INT(TBMP)

200    IF (JUNK>=0).AND.(JUNK<0.5) THEN CARRY=0 :  GOTO 220

210    IF (JUNK>=0.5).AND.(JUNK<1) THEN CARRY=0.5 :  GOTO 220

220   JUNK=JUNK/0.5

230    IF INT(TBMP)<1 THEN CARRY=CARRY+0.5

240    IF (JUNK<>0).OR.(JUNK<>1) THEN TBMP=INT(TBMP)+CARRY

245    REM UPDATE SYSTOLIC TIME (MS) AFTER TBMP IS ADJUSTED

250   MS=TBMP*MP

255    REM CALCULATE OFF TIME (OFTM)

260   OFTM=(1000*60/HTRT)-(MS)

265    REM 270-310 ENSURE THAT OFTM IS EVENLY DIVISABLE BY 10
          MILLISECONDS

270   OFTM2=OFTM/10

280   JUNK=OFTM2-INT(OFTM2)

290    IF (JUNK>=0).AND.(JUNK<0.5) THEN CARRY=0 :  GOTO 310

300    IF (JUNK>=0.5).AND.(JUNK<1) THEN CARRY=10 :  GOTO 310

310   OFTM=10*(INT(OFTM2))+CARRY

320    FOR I=1 TO 5 :  PRINT "   " :  NEXT I

325    REM 330-390 PRINT OUT ADJUSTED SYSTEM PARAMETERS

328    PRINT "              ADJUSTED PARAMETERS"
```

```
330     PRINT "SYSTOLIC TIME = ",MS,"MILLISECONDS"

340     PRINT "DIASTOLIC TIME = ",OFTM,"MILLISECONDS"

350    EHTRT=60/((MS+OFTM)/1000)

360     PRINT "EFFECTIVE HEART RATE = ",EHTRT,"BEATS / MINUTE"

370     PRINT "  "

380     PRINT "MOTOR PULSES / BEAT = ",MP

390     PRINT "TIME BETWEEN EACH MOTOR PULSE =
           ",TBMP,"MILLISECONDS"

400     REM CONFIGURE 8255 PORTS "A" AND "B" AS OUTPUT, PORT
           "C" AS INPUT

420    XBY(0C803H)=89H

430     REM PORT "A" ADDRESS IS 0C800H

435     REM PORT "C" ADDRESS IS 0C802H

500     REM PREPARE VARIABLES TO PASS TO ASSEMBLY ROUTINE

510    OFTM3=OFTM/10

520    TBMP2=TBMP*2

530     REM VARIABLE "MP" CONTAINS NUMBER OF MOTOR PULSES PER
           BEAT

535     REM PUSH VARIABLES ONTO STACK

540     PUSH OFTM3

550     PUSH TBMP2

560     PUSH MP

570     PRINT "   "

580     PRINT "RUNNING MOTOR SUBROUTINE"

590    DBY(62)=90

600    DBY(80)=5

610    DBY(81)=1
```

```
620   DBY(82)=9

630   DBY(83)=8

640   DBY(84)=10

650   DBY(85)=2

660   DBY(86)=6

670   DBY(87)=4

680    CALL 0

690   TST=DBY(79)

700    IF TST=1 THEN  PRINT "END OF SYRINGE" :  GOTO 800

710    IF TST=2 THEN  PRINT "PROGRAM HALTED BY RUN/STOP
          SWITCH" :  GOTO 800

800    END
```

APPENDIX B: ASSEMBLY LANGUAGE SUBROUTINE FOR SYRINGE PUMP STEPPER-MOTOR CONTROL

Notes:

Register bank 3 (RB3) is at internal memory locations 18H through 1FH.

Register bank 0 (RB0) is at internal memory locations 00H through 07H.

Internal memory locations 20H and 21H are also reserved for assembly language programs.

Register banks 1 and 2 are not for general programing use and should be left alone.

```
4100                      ORG 4100H LOC 4100H

4100  53 D0 E7            ANL PSW,#0E7H
                              Select Register Bank 0.

4103  74 01               MOV A,#01H

4105  12 00 30            LCALL 30H
                              8052AH Subroutine to Pop a
                              value off the user stack.
                              Pops MP (number of
                              stepper-motor pulses).

4108  89 21               MOV 21H,R1
                              Saves MP in internal
                              memory location 21H.

410A  74 01               MOV A,#01H

410C  12 00 30            LCALL 30H
                              8052AH Subroutine to Pop a
                              value off the user stack.
                              Pops TBMP2 (time between
                              motor pulses).

410F  89 20               MOV 20H,R1
                              Saves TBMP2 in internal
                              memory location 20H.
```

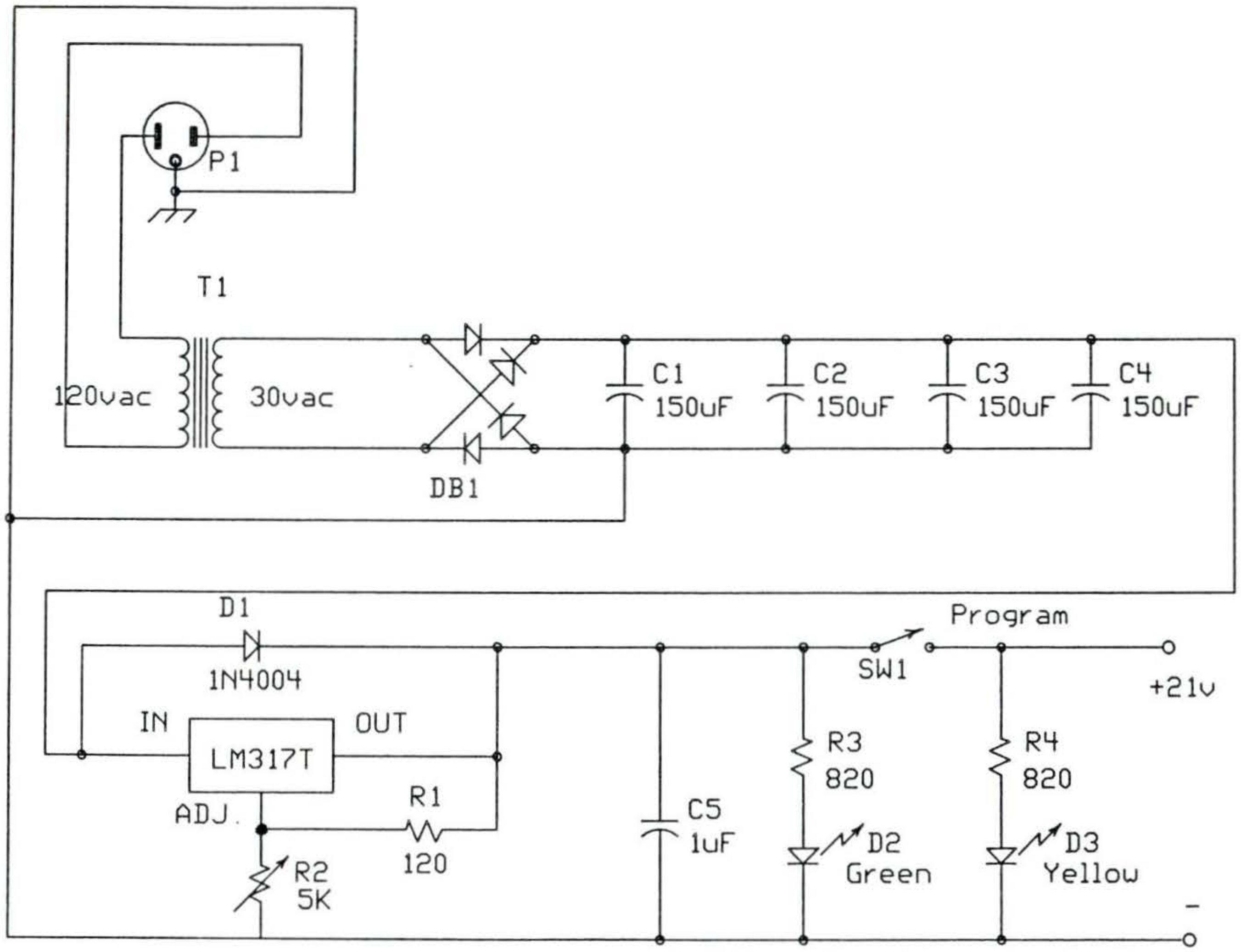
```
4111  74 01                   MOV A,#01H

4113  12 00 30                LCALL 30H
                                  8052AH Subroutine to Pop a
                                  value off the user stack.
                                  Pops OFTM3 (diastolic off
                                  time).

4116  89 1F                   MOV 1FH,R1
                                  Saves OFTM3 in internal
                                  memory location 1FH.

4118  53 D0 FF                ANL PSW,#0FFH
                                  Select RB3.

411B  90 C8 00                MOV DPTR,#0C800H
                                  Initializes the data
                                  pointer to the address of
                                  port A of the 8255
                                  (0C800H).

411E  78 50                   MOV R0,#50H
                                  Moves the first data value
                                  in the stepper-motor
                                  sequence into RB3:R0.

4120  85 21 1A       SYST:    MOV 1AH,21H
                                  Moves MP into internal
                                  memory location 1AH.

4123  75 1B E6       HLFMS:   MOV 1BH,#0E6H
                                  Moves the half millisecond
                                  timing constant into
                                  internal memory location
                                  1BH.

4126  85 20 19                MOV 19H,20H
                                  Moves TBMP2 into internal
                                  memory location 19H.

4129  D5 1B FD       LOOP1:   DJNZ 1BH,LOOP1

412C  75 1B E6                MOV 1BH,#0E6H
                                  Above two lines constitute
                                  the half millisecond time
                                  delay.

412F  D5 19 F7                DJNZ 19H,LOOP1
```

```
4132  E6                         MOV A,@R0
                                     Move the stepper-motor
                                     data for the next step to
                                     the Accumulator.

4133  F0                         MOVX @DPTR,A
                                     Output stepper-motor data.

4134  08                         INC R0
                                     Increment to next stepper-
                                     motor data point.

4135  B8 58 02                   CJNE R0,#58H,CONT
                                     This causes the motor
                                     sequence data pointer to
                                     move back to the top of
                                     the data set and start
                                     over once the last point
                                     has been reached.

4138  78 50                      MOV R0,#50H

413A  D5 1A E6         CONT:     DJNZ 1AH,HLFMS
                                     End of systolic timing
                                     routine.

413D  E5 1F            DIAST:    MOV A,1FH

413F  F5 19                      MOV 19H,A
                                     Move OFTM3 to internal
                                     memory location 19H.

4141  75 1B E0         TNMS:     MOV 1BH,#0E0H
                                     Inside timing loop
                                     constant for 10
                                     millisecond timing loop.

4144  75 1C 13                 . MOV 1CH,#13H
                                     Outside timing loop
                                     constant for 10
                                     millisecond timing loop.

4147  D5 1B FD         LOOP2:    DJNZ 1BH,LOOP2

414A  75 1B FF                   MOV 1BH,#0FFH

414D  D5 1C F7                   DJNZ 1CH,LOOP2
```

```
4150  D5 19 EE                   DJNZ 19H,TNMS
                                     Above 4 lines constitute
                                     10 millisecond time delay.

4153  E4                         CLR A
                                     Clear Accumulator.

4154  E5 88                      MOV A,TCON
                                     Move the value of the
                                     Timer/Counter register to
                                     the Accumulator.

4156  20 E3 0E                   JB ACC.3,INTDN
                                     Test TCON.3 if it is set
                                     then jump to INTDN
                                     (interrupt done label).

4159  90 C8 02                   MOV DPTR,#0C802H
                                     Set data pointer to
                                     address of Port C.

415C  E0                         MOVX A,@DPTR

415D  90 C8 00                   MOV DPTR,#0C800H
                                     Read Port C (location of
                                     Run/Stop switch).

4160  20 E0 BD                   JB ACC.0,SYST
                                     Test Port C bit 0 if it is
                                     set then continue running
                                     the subroutine otherwise
                                     exit subroutine.

4163  75 4F 02                   MOV 4FH,#02H
                                     Put a value of 2 into
                                     location 4FH.  The BASIC
                                     routine reads this memory
                                     location to determine why
                                     the subroutine was exited.

4166  22                         RET

4167  75 4F 01           INTDN:  MOV 4FH,#01H
                                     Put a value of 1 into
                                     location 4FH.  The BASIC
                                     routine reads this memory
                                     location to determine why
                                     the subroutine was exited.
```

```
416A   C2 8B                          CLR TCON.3
                                          Clear TCON.3 so that the
                                          next interrupt can be
                                          detected.

416C   22                             RET
```

APPENDIX C: 21V EPROM PROGRAMMER POWER SUPPLY SCHEMATIC


P1
T1
120vac
30vac
DB1
C1
150uF
C2
150uF
C3
150uF
C4
150uF
D1
1N4004
IN
OUT
LM317T
ADJ.
R1
120
R2
5K
C5
1uF
R3
820
R4
820
D2
Green
D3
Yellow
Program
SW1
+21v
-

APPENDIX D:

DATA ACQUISITION HARDWARE AND SOFTWARE MANUFACTURERS FOR IBM PC'S (BASED ON CONNER, 1986)

Action Instruments Inc. 8601 Aero Dr., San Diego, California 92123. (619) 279-5726

Advanced Peripherals Inc. 12650 W. Geauga Plaza, Chesterland, Ohio 44026. (216) 729-3927

Analog Devices. Two Technology Way, Norwood, Massachusetts 02062. (617) 329-4700

Anasco. Audubon Rd., Wakefield, Massachusetts 01880. (617) 246-0300

Argis Inc. Box 373, Hudson, Massachusetts 01749. (617) 562-9673

BBN Software Products Corp. 10 Fawcett St., Cambridge, Massachusetts 02238. (617) 864-1780

Burr-Brown. Box 11400, Tucson, Arizona 85734. (602) 746-1111

Cyborg Corp. 55 Chapel St., Newton, Massachusetts 02158. (617) 964-9020

Data Motion. Box 889, Orland Park, Illinois 60642. (312) 495-4007

Data Translation. 100 Locke Dr., Marlboro, Massachusetts 01752. (617) 481-3700

Dataq Instruments Inc. 100 Linclon St., Akron, Ohio 44308. (216) 434-4284

General Research Corp. 7655 Old Springhouse Rd., McLean, Virginia 22102. (703) 893-5900

Gould Inc. Recording Systems Div. 3631 Perkins Ave., Cleveland, Ohio 44114. (216) 361-3315

Hamilton/HGL Software. 6 Pearl Ct., Allendale, New Jersey 07401. (201) 327-1444

Hart Sceintific. 177 W. 300 South, Provo, Utah 84601. (801) 375-7221

HEM Engineering Co. 17025 Cresent Dr., Southfield, Michigan 48076. (313) 559-5607

ICS. 8601 Aero Dr., San Diego, California 92123. (619) 279-0084

Integrated Systems Products Inc. 6028 Fremont Circle, Camarillo, California 93010. (805) 987-5125

Interactive Microware Inc. Box 139, State College, Pennsylvania 16804. (814) 238-8294

Interactive Structures Inc. 218 Great Valley Parkway, Malvern, Pennsylvania 19355. (215) 644-8877

Keithley Instruments Inc. 28775 Aurora Rd, Cleveland, Ohio 44139. (216) 248-0400

Laboratory Technologies Corp. 255 Ballardvale St., Wilmington, Massachusetts 01887. (617) 657-5400

Lawson Labs Inc. 5700 Raibe Rd., Columbia Falls, Montana 59912. (406) 387-5355

Macmillan Software Co. 866 Third Ave., New York, New York 10022. (212) 702-3241

Metrabyte Corp. 440 Myles Standish Rd., Tauton, Massachusetts 02780. (617) 880-3000

Microcomputer Systems Inc. 1814 Ryder Dr., Baton Rouge, Louisiana 70808. (504) 769-2154

Microstar Laboratories Inc. 2863 152 Ave. NE, Redmond, Washington 98052. (206) 881-4286

Microway Inc. Box 79, Kingston, Massachusetts 02364. (617) 746-7341

Northwest Analytical Inc. 520 NW Davis, Portland, Oregon 97209. (503) 224-7727

Qua Tech Inc. 478 E. Exchange St., Akron, Ohio 44304. (216) 434-3154

RC Electronics Inc. 5386-D Hollister Ave., Santa Barbara, California 93111. (805) 964-6708

Scientific Solutions Inc. 6225 Cochran Rd., Solon, Ohio 44139. (216) 349-4030

Signal Technology Inc. 5951 Encina Rd., Goleta, California 93117. (805) 683-3771

Strawberry Tree Computers. 1010 W. Fremont Ave., Sunnyvale, California 94087. (408) 736-3083

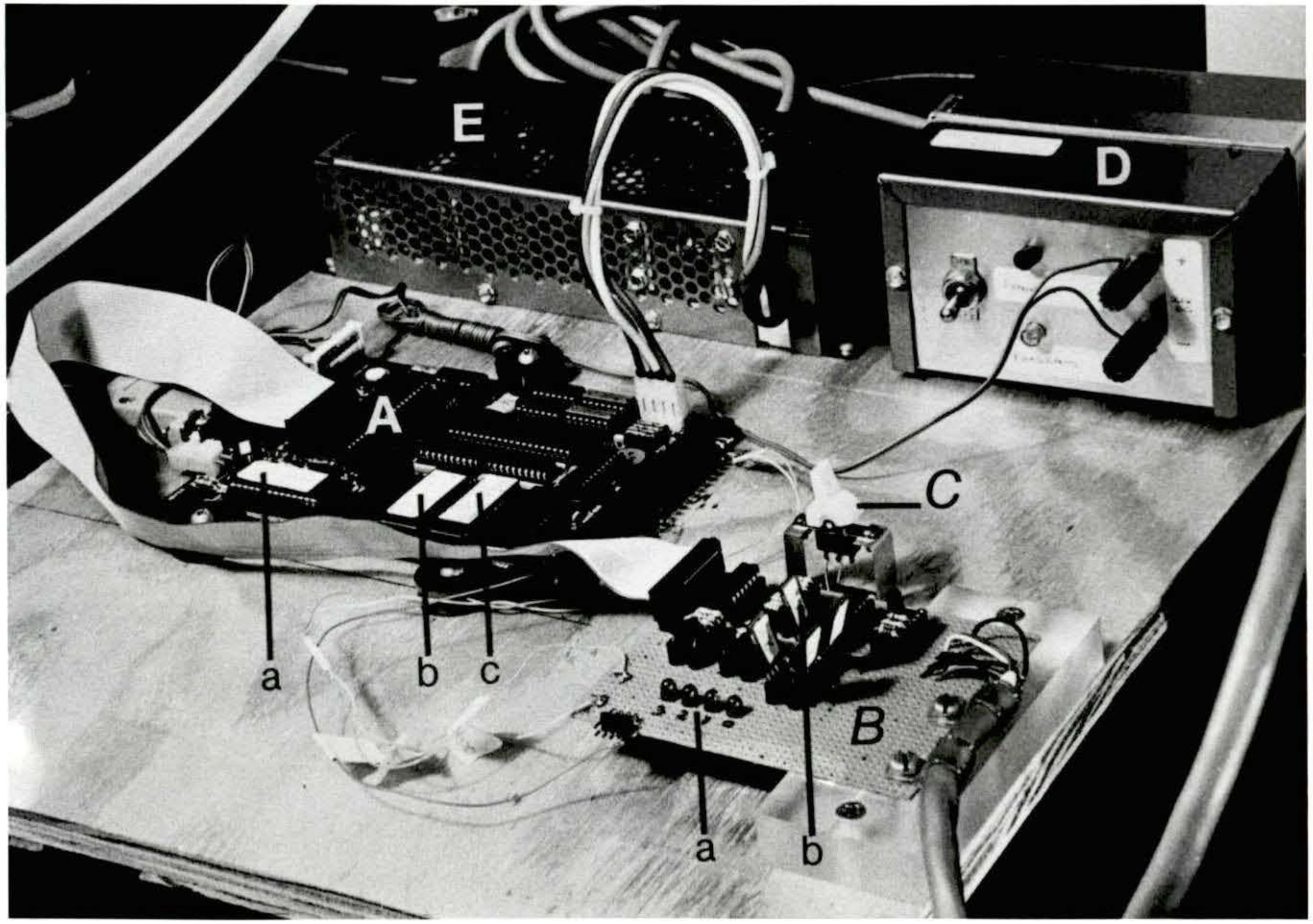
Taurus Computer Products Inc. 1755 Woodward Dr., Ottawa, Ontario K2C 0P9, Canada. (613) 226-5361

Unkel Software Inc. 62 Bridge St., Lexington, Massachusetts 02173. (617) 861-0181

Witt Engineering. 1802 Jefferson Ave., Glenview, Illinois 60025. (312) 724-0920

APPENDIX E: BCC52 COMPUTER SYSTEM AND INTERFACE HARDWARE

The following picture shows: A - the BCC52 computer, B - the syringe pump interface hardware, C - the "run/stop" switch, D - the 21v dc power supply for the EPROM programmer, and E - the +5v, +12v, and -12v power supply for the BCC52. On the BCC52 (A): a - the EPROM socket containing the BASIC program for the stepper-motor controller, b - the EPROM containing the object code for the assembly language subroutine, and c - the ROM A+B utilities. On the syringe pump interface hardware board (B): a - the LEDs used to monitor the 8255 outputs, and b - the TIP122 switching transistors which control the stepper-motor phases.


E
D
A
C
a
b
c
B
a
b

APPENDIX F: ALTERNATE SYRINGE PUMP CIRCUIT MODIFICATION

After receiving documentation detailing the Model 351 syringe pump, an alternate method of circuit modification was evident. Figure F.1 shows a sketch of the stepper-motor driver circuit on the Model 351 syringe pump. This sketch was derived from the schematic of the syringe pump control circuit (Sage Instruments, 1975) as provided in the Model 351 service manual. It is the final drive circuit which controls the stepper motor.

Figure F.2 shows the proposed modifications. By adding a single pair of connector blocks on the syringe pump, it would be very easy to either control the stepper-motor via an outside control circuit or via its own control circuit. All that would be necessary is a modification to the base drive for the switching transistors Q4 - Q7. The following list shows which connections need to be made for specified stepper-motor control.

For original syringe pump functions:

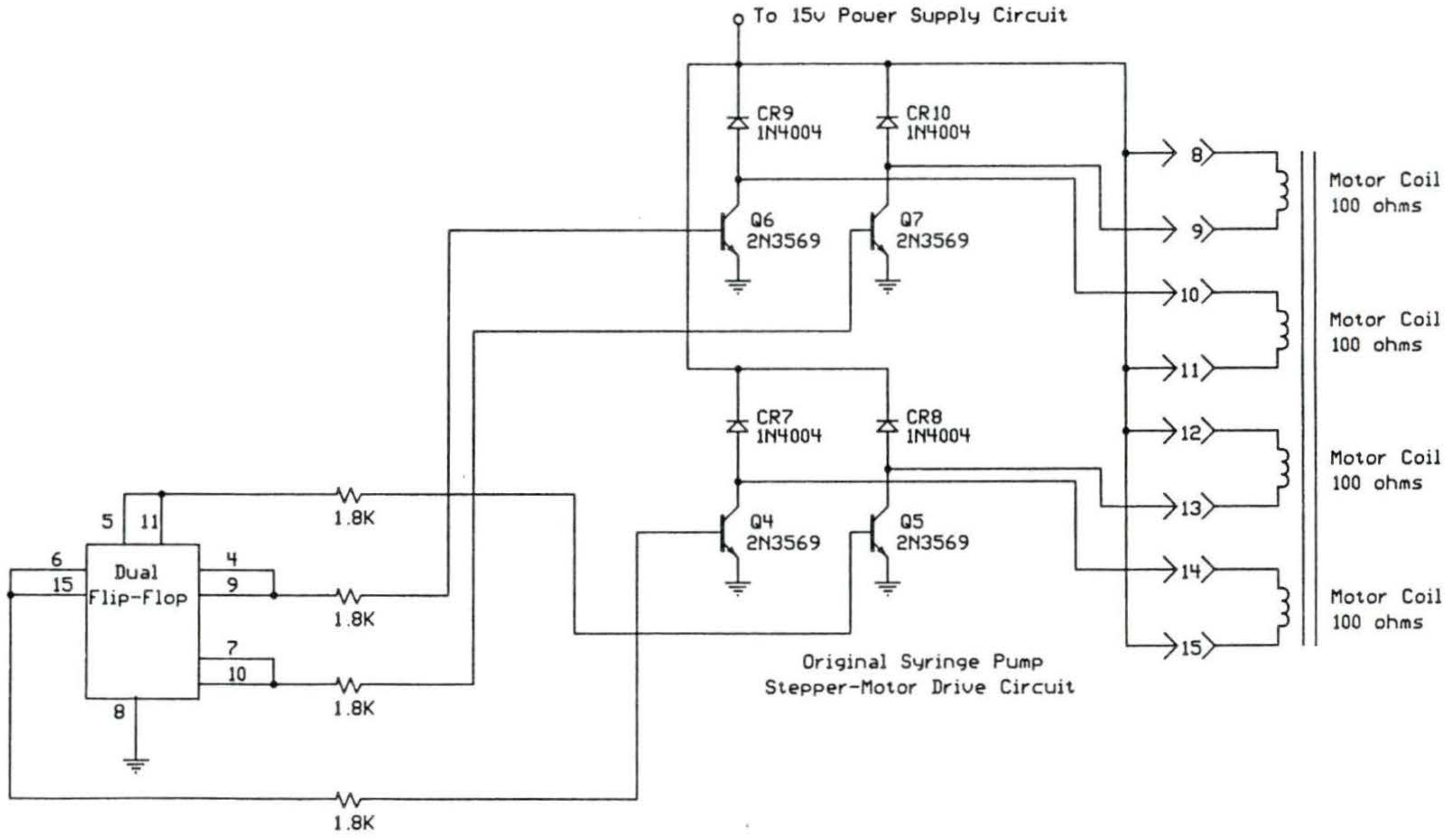
TO	FROM
A	a
B	b
C	c
D	d
E	n/c

For outside control over the stepper-motor:

TO	FROM
A	aa
B	bb
C	cc
D	dd
E	common ground.

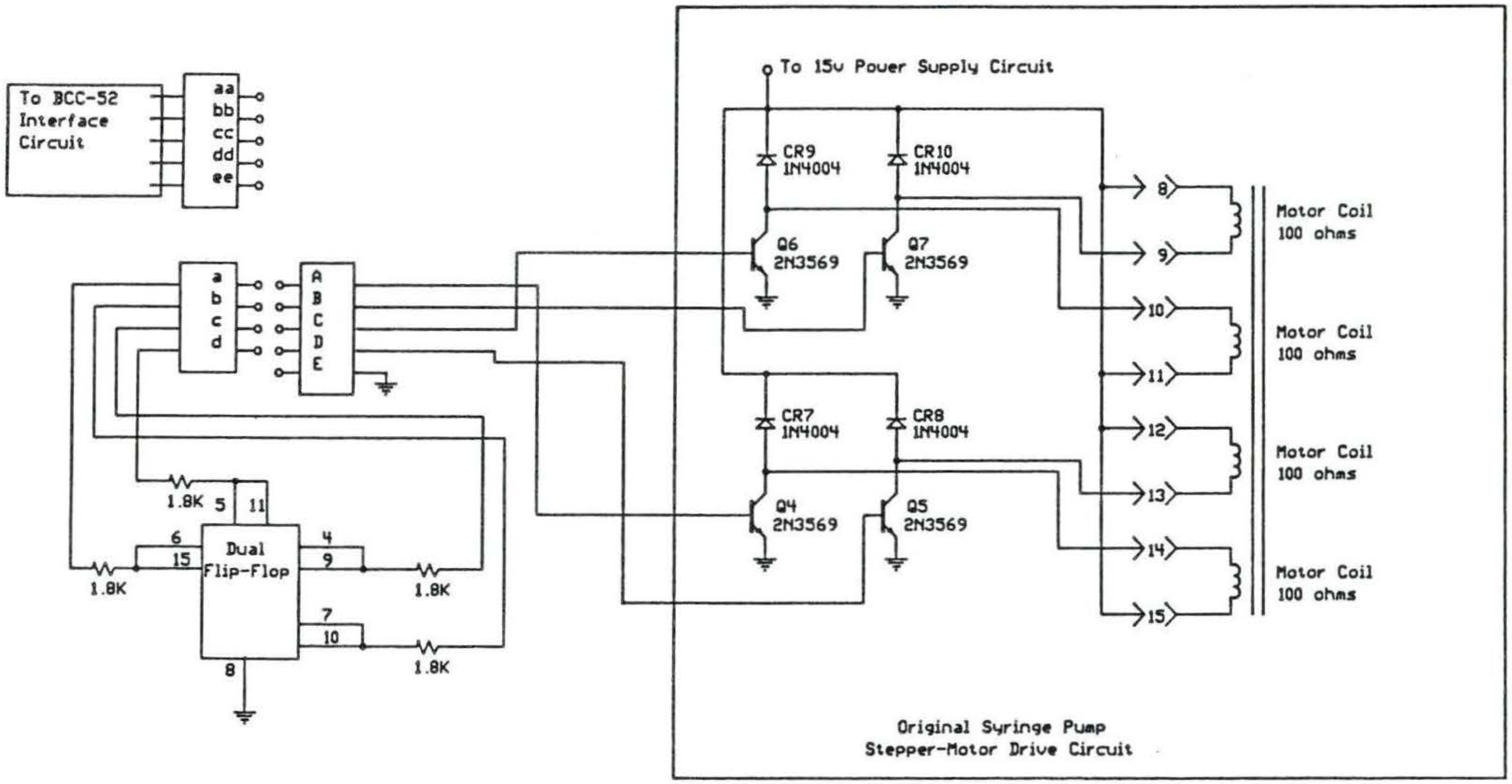
This method of modification would allow the syringe pump to be operated in either the mode it was originally intended for, or as controlled from an outside source.

Figure F.1. Sketch of original syringe pump stepper-motor control circuit



Original Syringe Pump
Stepper-Motor Drive Circuit

Figure F.2. Sketch of proposed syringe pump control circuit modifications


To BCC-52 Interface Circuit
aa
bb
cc
dd
ee
a
b
c
d
A
B
C
D
E
1.8K
5
11
6
15
Dual Flip-Flop
4
9
7
10
8
1.8K
1.8K
1.8K
To 15v Power Supply Circuit
CR9 1N4004
CR10 1N4004
Q6 2N3569
Q7 2N3569
CR7 1N4004
CR8 1N4004
Q4 2N3569
Q5 2N3569
8
9
10
11
12
13
14
15
Motor Coil 100 ohms
Motor Coil 100 ohms
Motor Coil 100 ohms
Motor Coil 100 ohms
Original Syringe Pump Stepper-Motor Drive Circuit

ACKNOWLEDGEMENTS

I would like to thank my loving wife, Terri, for the many hours of patience and support during my graduate studies. She stood by me faithfully through all the long days and late nights. The support of our families through these two years has also been very special. They were there anytime I needed them.

I would also like to express appreciation to my major professors, Dr. Swift and Dr. Redmond, for their help and support during all of this. I can't thank them enough for the technical and moral support they have given. Their combined editing efforts have helped immensely during the writing of this thesis.

Thanks also to Mary Johnson for her patience during the development and testing of this system. It was a long design cycle.

Finally, I wish to express special thanks to Dr. Mary Helen Greer and Dexter K. Ishii. In their own special ways they provided both technical guidance and moral support for which I can never thank them enough.

I wish all of you well in your future endeavors.