THE DESIGN OF A DIGITAL COMMUNICATION PROTOCOL FOR USE IN A MODULAR HYPERTHERMIA SYSTEM

BY

ROBERT ALAN CARGNONI

B.S., University of Illinois, 1983

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1985

Urbana, Illinois

DEDICATION

This thesis is dedicated to my parents, Bob and Pat, and to my two sisters, Karen (Kiki) and Sharon (Shar Bear).

ACKNOWLEDGEMENTS

First of all, I would like to thank Professor Charles A. Cain for originally introducing me to this thesis project, and Professor Leon A. Frizzell for assisting with the project coordination. I would also like to thank Professor Richard L. Magin for his constant project coordination efforts.

The majority of the research and laboratory work for this project was conducted at URI Therm-X, Inc. To Dr. Steven Foster and the other Therm-X employees, thank you for answering the numerous questions I raised during the development of this thesis. I would like to thank my graduate advisor, Professor Everette C. Burdette, for providing the proper facilities for this project and for assisting me during the writing of this document.

I would especially like to thank two more people who played vital roles in the completion of this thesis. The first person is Christopher W. Badger, who spent countless hours discussing the thesis with me and deserves much of the credit for the success of this project. The second person is Mrs. Wanda Elliott, who did an excellent job of preparing this manuscript and never once complained about the time limits I imposed on her.

Last but not least, I would like to thank my family for their constant support and encouragement throughout my academic career. Their faith in me was my greatest motivation for completing this endeavor and their love will always be my most valuable possession.

TABLE OF CONTENTS

CHAPTER 1

OBJECTIVES

The primary objective of this thesis is to describe in detail the digital communication protocol which was designed for the ultrasonic hyperthermia system currently being developed by URI Therm-X, Inc. A communication protocol by contemporary definition is basically a set of rules for operating a communication system, where the rules are designed to solve operating problems in specified areas [McNamara, 1978]. These areas are enumerated and examined individually later in this thesis, as are other aspects of the total communication system. From a specification viewpoint, each element of the protocol is mentioned, beginning with any alternatives that were originally considered and the reason or reasons for choosing one particular option over the others. From an implementation viewpoint, the end product software algorithms are described and the programs written to perform these algorithms are also included in this document.

It should be noted that the work done for this thesis project dealt more with design problems than with digital communication theory. The primary project objective was simply to provide error-free communication in the completed system. Consequently, the rationale behind choosing one concept over another was based almost solely on design trade-offs encountered in the specification process. Also, the usual benchmarks for communication systems, such as bandwidth and data transmission rates, were not really applicable in this case because of the nature of the transmitted information. Instead, the test of the protocol was its ability to recover from transmission errors. During the development phase, simulated transmission errors were applied to the system, and the results from these tests were used to refine the protocol and to debug the software. However, once the protocol was completed, all of the tests worked correctly, so the documentation of the actual test results served little purpose. Consequently, test data proving or disproving the successful operation of the protocol were not obtained for this thesis, except to note that the implemented protocol functioned properly in the integrated system.

The thesis is divided into 10 chapters and seven appendices. Chapter 2 is an introduction explaining the motivation for designing the protocol. A brief overview of hyperthermia and of the communication flow between the individual modules of the hyperthermia system is also included in this chapter. The interface standard utilized in the hyperthermia system is discussed in Chapter 3, together with a discussion concerning several communication system parameters affected by the standard. Chapter 4 defines several terms used throughout the remainder of the thesis and also presents a message classification scheme used exclusively in this system. Chapter 5 examines each protocol element and completely specifies the protocol in general terms. Chapter 6 presents the specific messages used in the hyperthermia system and illustrates the typical transactions between the different system modules. Chapter 7 provides an introduction to the programs that implement the communication protocol while Chapter 8 briefly mentions the tests employed to ensure the correctness of these programs. Chapter 9 discusses several major alternatives that could be incorporated into future systems and Chapter 10 is a summary. Appendix A gives a brief description of several features of the Intel 8031 microcontroller, the processor used exclusively in this thesis. Appendices B, C, and D contain flowcharts for the communication programs first introduced in Chapter 7 and the last three appendices contain full listings of the Intel 8031 assembly level code, complete with brief descriptions of the variable names used in the programs.

CHAPTER 2

INTRODUCTION

Many of the protocol aspects discussed in this thesis are general enough to be employed in any communication system. On the other hand, several of the protocol features are implemented to fulfill particular communication requirements of this hyperthermia system. For this reason, a short discussion of the system is first presented.

2.1. Hyperthermia System Characteristics

Hyperthermia as a cancer therapy involves the selective heating of tumors to a therapeutic temperature greater than the temperature of the surrounding tissue. The biologic rationale behind hyperthermia treatments include the fact that temperatures greater than 42^{o}C will kill cells exponentially as a function of time, and more importantly, the fact that the toxic effects of the heat will selectively kill some radioresistant tumor cells [Stewart, et al., 1984]. When used in conjunction with either chemotherapy or radiotherapy, this combined modality process has the potential of decreasing the required doses of either chemicals or radiation while increasing the therapeutic gain factor (ratio of therapeutic effect on tumor versus normal tissue). Although current research in the field has yielded very favorable results, clinical hyperthermia as a whole is still beset with several problems [Stewart, et al., 1984].

The most significant problem is the inability, in general, to deliver uniform and/or therapeutic thermal doses to malignant tumor sites. It was implied earlier that temperatures greater than 42°C adversely affect normal tissue as well as tumor, so it is imperative that heat-producing energy be delivered to tumor mass with minimal effect on normal tissue. A high degree of selectivity in the heating process resolves this first problem. This feature allows energy to be delivered to the tumor site to produce beneficial heating effects, without severely affecting the surrounding healthy tissue temperature.

A second problem confronting hyperthermia deals with the resolution, both temperature and spatial, required for a clinical thermometry system used in monitoring induced temperature increases. One method of obtaining adequate temperature resolution involves well-known thermoelectric thermometry techniques which economically provide fast response, reasonable long-term stability, and accuracy over a wide temperature range. Multielemental temperature sensing equipment, highly adaptable to different anatomic sites, provides reasonable spatial resolution.

Finally, when combined therapy modalities are utilized, the appropriate timing and sequence of heat and ionizing radiation therapy or chemotherapy must be determined [Stewart, et al., 1984]. In order to solve this last problem, adequate treatment feedback data must be obtained and an appropriate algorithm must be available to process the data. This last requirement indicates a need for a highly automated, programmable system.

A hyperthermia system capable of solving the three problems just discussed is presented in basic block diagram form in Figure 1 (all figures appear following the closing summary). The dashed boxes indicate the two primary peripheral groups, the applicator subsystem and the thermometry subsystem. Communication in the hyperthermia system occurs on the paths between the central control computer and the two independently programmable, computerized subsystems. These paths are shown as the heavier lines in Figure 1.

The central control computer is a Digital Equipment Corporation MICRO PDP-11 (PDP-11) running a real-time operating system. It handles all user interactive functions such as displaying the treatment schedule on the operator terminal, accepting keyboard input, and displaying control variables and actual treatment feedback on the color monitor. Less apparent to the user are its functions of interacting with both subsystems over RS-232-C standard communication links and controlling energy deposition according to an adaptive thermal modeling algorithm. In addition, several background tasks such as the continual monitoring of the status of each subsystem are also handled.

The applicator subsystem is composed of several modules whose overall function is to apply a controlled ultrasound field to the tumor area. The Intel 8031 microcontroller interacts with the central control computer and updates its own control signals to alter the heating pattern of the multielement ultrasound applicator. The Intel 8031 also continually checks its own digital outputs against the desired outputs received from the PDP-11, monitors forward and reflected power levels of the RF amplifiers, and reports any discrepancies to the PDP-11.

The thermometry subsystem's Intel 8031 microcontroller also interacts with the central control computer and provides the temperature data needed as input to the adaptive thermal modeling algorithm. The physical interface to the patient is accomplished through the use of thermocouple probes that monitor several locations in the treatment area. Thus, the central control computer combines with the applicator and thermometry subsystems to yield a full hyperthermia system capable of producing a highly selective heating pattern, of acquiring accurate temperatures from a limited number of treatment points, and of coordinating this information to optimize heating under time, temperature, and tissue dependent physiological conditions both within and in the area surrounding a tumor mass [Goss, et al., 1984].

In order to insure maximum controllability of the system, a closed-loop feedback control scheme is implemented. An overview of this loop entails the PDP-11 sending a desired heating pattern to the applicator subsystem, which applies the appropriate ultrasound field to the treatment site. The resulting temperatures created in the tissue are sensed by the thermometry subsystem, which transmits these new values to the PDP-11. The PDP-11 in turn processes these data in the adaptive thermal modeling algorithm and outputs the revised heating pattern to the applicator subsystem, thus beginning the loop once again.

From an engineering standpoint, this closed-loop control scheme provides a means for the PDP-11 to monitor the performance of both subsystems. Abnormal temperature feedback could possibly indicate problems with one or both subsystems, allowing the PDP-11 to either attempt corrective action or to prompt for operator assistance. From a clinical standpoint, the closed-loop is necessary to maintain maximum effectiveness of the treatment, which is accomplished only when the tumor site and the surrounding tissue temperatures are held in their respective desirable ranges. Several physiological factors such as differing thermal conductivities of the tissues and blood flow rate through the treatment area combine to complicate this process. More specifically, the blood perfusion acts like a heat sink and is a variable, unpredictable parameter in and around tumors where the anatomy is highly abnormal [Stewart, et al., 1984]. The constant feedback allows the adaptive thermal modeling algorithm to alter the heating scheme and counter the physiological effects. In order for this closed-loop control scheme to function properly, it is imperative that reliable communication links be established on the paths between the central control computer and the two subsystems. However, before these links are examined in detail, a brief overview of the communication flow is presented as background information.

2.2. Communication Flow Overview

In the previous section it was mentioned that the thermometry subsystem transmits temperature values to the central control computer. It was also mentioned that the central control computer interacts with the applicator subsystem to alter the applied heating pattern. Both of these transactions occur in steady-state communication conditions and represent the bulk of the messages sent during a typical hyperthermia treatment. However, these steady-state conditions are attained only after several other transactions on each communication link are first completed. The thermometry subsystem link is considered first.

The first communication action of the thermometry subsystem is to send its identification and its status to the central control computer when requested. This same status indicator is also used later to notify the central control computer that the subsystem is fully initiated and calibrated. The corresponding calibration data are then transmitted to the central control computer to be stored on disk. The next transaction fully initializes the thermometry subsystem and steady-state communication flow commences. As previously indicated, this state exists when the central control computer periodically requests and receives temperature update transmissions.

Similar to the thermometry subsystem, the first action of the applicator subsystem is to send its identification and its status to the central control computer. The next three transactions are designed to set up an initial heating pattern by controlling the frequency, output voltage, and individual duty cycles of the RF amplifiers. Once the pattern is established, the central control computer fully initializes the applicator subsystem and the energy deposition begins. Steady-state communication flow on this link exists when the central control computer periodically transmits changes in the output voltage and the individual duty cycles of the RF amplifiers to the applicator subystem.

In addition to the transactions required to reach and maintain steady-state communications, several other messages are implemented to deal with transmission errors and other communication problems. These "special case" transactions are very important because they ensure that the applicator subsystem will always be disabled if severe transmission problems do occur.

This feature is necessary because improper operation of the applicator subsystem poses a potential danger to the patient in therapy due to loss of control of output power, and therefore, heating.

All of the transactions presented in this section are discussed in greater detail later in the thesis. However, it is obvious even from this short overview that proper operation of the hyperthermia system depends on the correct transmission of messages, which is dependent on both the communication protocol developed in this thesis and the condition of the physical interface used in the hyperthermia system. Therefore, the interface standard employed in this system is considered next.

CHAPTER 3

INTERFACE STANDARDS

In a complete communication system, the interface standard and the communication protocol encompass the majority of the hardware and software communication functions, respectively. It is oftentimes difficult to categorize certain communication aspects, such as framing or the baud rate, under one heading or another. In this thesis, framing and the baud rate are considered to be more hardware related; consequently, they are presented in this chapter.

3.1. Choice of Standards

The two interface standards originally considered for the hyperthermia system were the Institute of Electrical and Electronics Engineers (IEEE) Standard 488-1978 Bus Interface and the Electronic Industries Association (EIA) RS-232-C Standard. In order to simplify matters, the two standards are referred to as IEEE 488 and RS-232-C hereafter. The latter interface standard is used in this hyperthermia system for several reasons. First, RS-232-C readily supports asynchronous transmissions. Second, RS-232-C is a widely accepted industry standard for computer-to-peripheral interfaces which is attractive from a compatability standpoint. The third reason is the ease of implementation associated with the standard, and the final reason is the low hardware overhead involved. Although the IEEE 488 standard was not employed in this particular system, future utilization of the IEEE 488 interface is possible, and would overcome some line control limitations which were encountered using RS-232-C. A more detailed examination of this alternative is discussed in Chapter 9, Future Considerations.

3.1.1. Configuration

Choosing the RS-232-C standard requires that each of the communicating modules be configured as either data terminal equipment (DTE) or data communication equipment (DCE). All three modules in the hyperthermia system are configured as DCE, and therefore, communication links involved must be cross connected, which is explained below. Using the RS-232-C standard implies that a 25-pin interface consisting of ground, data, control, and timing lines is utilized. When a cross connection between devices is employed, a maximum of nine lines of the 25 possible need to be connected to handle asynchronous data [Seyer, 1984]. The specific functions of these nine lines are illustrated in Figure 2a.

The protective ground (pin 1) should be electrically bonded to the equipment frame, and if used, should be passed straight through the interface to the protective ground located at the other end of the cable. The signal ground (pin 7) establishes the common reference for all other signals except pin 1, and is connected directly to the signal ground located at the other end of the cable. The data leads include the transmitted data and received data leads, pins 2 and 3, respectively. Their connection depends on the media type (simplex, half-duplex or full-duplex) used on each communication link. The communication protocol implemented in the hyperthermia system utilizes a full-duplex media type, which corresponds to subset E in the RS-232-C standard [EIA Standard RS-232-C, 1969]. Using this media type requires that the transmitted data lead on one DCE be connected to the received data lead on the other DCE, and vice versa. Thus, the data leads are cross connected, creating a four-pin interface like that shown in Figure 2b.

The remaining five leads perform control functions and are used to provide hardware handshaking capabilities, if desired. For this hyperthermia system, none of these control interfaces are implemented, the reasons being ease of implementation and hardware limitations imposed at both DCE interfaces. However, if utilized correctly, these five leads, or a subset of them, could possibly solve some control problems encountered in this system or could enhance the communication capability of the link with relatively small software and hardware overhead increase. This alternative is examined in more detail in Chapter 9, Future Considerations.

3.1.2. Physical Interface

As mentioned previously, the interface between the central control computer and each subsystem consists of only four lines. However, a 25-pin connector is still used at each module in order to adhere to the RS-232-C standard. Shielded transmission cable is used to lower the possibility of external interference creating transmission errors.

3.2. Framing

The framing of each byte of data is another protocol aspect established by the RS-232-C standard. According to the standard, the data format for asynchronous serial transmission consists of a start bit, five to eight data bits (one of which can be a parity bit), and one or two stop bits [PSIO Manual]. In this communication protocol, a variation of this scheme is used, specifically: a start bit (logical zero), eight data bits, an even parity bit, and a stop bit (logical one). Eight data bits are used so that the extended American National Standard Code for Information Interchange (ASCII) can be supported. Use of ASCII is advantageous because a standard computer terminal can then be used to emulate any of the system modules for communication test purposes. Examination of the code reveals that only seven of the bits are actually used to represent characters. The most significant bit is normally a logical zero, but in this protocol it is sometimes used as a sequence number bit, the use of which is explained later in this thesis. The even parity bit is included for error detection purposes and is also discussed in greater detail later.

3.3. Baud Rate

The Intel 8031 microcontroller is capable of supporting most of the commonly used baud rates. A rate of 1200 bits per second is currently used in this hyperthermia system, but a possible increase to 2400 bits per second has been considered. Neither this increase, nor an increase to a much higher baud rate, say 9600 bits per second, presents any foreseeable problems. However, the optimum transmission rate should not be determined until the full system including graphics updates, subsystem control, and algorithm calculations is fully operational. At that time, trade-offs involving interrupt frequency versus transmission time can be discussed and different baud rates can be examined in the system to produce an optimum solution.

CHAPTER 4

STRUCTURE CLASSIFICATION

The main purpose of this chapter is to introduce several generic message classifications that encompass all types of transmissions utilized in this hyperthermia system. However, before proceeding with this, some background information is needed. The following section defines several terms that are used throughout the remainder of this thesis.

4.1. Definition of Terms

Although error detection is not discussed in detail until Section 5.3.1, one of the error detection mechanisms is briefly presented here. Specifically, all single character messages transmitted in this system are copied and transmitted twice more to create a three-byte character set.

Triplet - In this communication protocol, a triplet is any set of three identical characters transmitted as a single message or as part of a single message.

Full Data Block - A full data block constitutes a single message and is comprised of three basic parts: a header, the data bytes, and two checksum bytes. It is important to note that the data bytes and the checksum bytes are transmitted one time only. The same cannot be said of the header.

Header - Residing at the beginning of every full data block, a header is always an alpha character triplet.

Command - A command is any message that initiates a transaction. As illustrated in the following section, a command assumes various forms.

Acknowledgement - An acknowledgement is any message that completes a transaction. Similar to the command, it also assumes various forms.

Structure - Excluding special cases for the time being, a structure is defined as being any command and its corresponding acknowledgement. Several classes of structures are presented below.

4.2. Classifications

All of the structures used in the current hyperthermia system can be grouped into one of six classifications. Also, a seventh class is defined even though it is not currently used. It is mentioned here because it provides a transmission format for much larger blocks of data than those used currently. New messages created for future systems, or for enhanced capabilities in this system, will probably fit into one of these seven classifications:

Class 1 - An alpha character triplet is sent as the command and no acknowledgement is expected back. This classification is a pseudo-structure because of the missing acknowledgement and is considered to be a special case. A diagram illustrating the transmission of this pseudo-structure is presented in Figure 3a, together with a figure legend and a special note, both of which pertain to all of Figure 3.

Class 2 - An alpha character triplet is sent as the command and a predetermined alpha character triplet is the expected acknowledgement. A diagram for this structure transmission is illustrated in Figure 3b.

Class 3 - An alpha character triplet is sent as the command and a variable character triplet, not necessarily alpha type, is the expected acknowledgement. For this particular structure, there is a small set of valid replies which the acknowledgement must be an element of. A diagram for this structure transmission is illustrated in Figure 3c.

Class 4 - In this group, the command consists of two triplets instead of just one. The first triplet is an alpha character and the second triplet is a numeric character representing data necessary to complete the command. The expected acknowledgement is a predetermined alpha character triplet. A diagram for this structure transmission is illustrated in Figure 3d.

Class 5 - A full data block is sent as the command and the expected acknowledgement is a predetermined alpha character triplet. A diagram for this structure transmission is illustrated in Figure 3e.

Class 6 - This group is the complement of Class 5. In this case, an alpha character triplet is the command and a full data block is the expected acknowledgement. A diagram for this structure transmission is illustrated in Figure 3f.

Class 7 - This group establishes a format for larger full data blocks which can be used to replace the full data blocks used in either Class 5 or Class 6 communication. An alpha character triplet is still used as a header. However, the data bytes are broken up into sub-blocks of predetermined size and two checksum bytes are generated for each sub-block. Thus, the header is sent first and is immediately followed by the first sub-block and checksum. A predetermined alpha character triplet is the expected acknowledgement. After the correct acknowledgement is received, the next sub-block and checksum are sent and the process continues until all the data are transmitted. The reason for sending the data in sections is not obvious now because retransmission requests have yet to be discussed. To explain briefly, if the receiving module detects a transmission error, it is able to request a retransmission after complete reception of that sub-block. At the other end, the transmitting module expects a certain acknowledgement, but may receive a retransmission request instead. If it does, it retransmits the full data block. In the eventuality that an error does occur, this class allows retransmissions to commence before the full data block has been received, thus saving on retransmission time. A diagram for this structure transmission, where the full data block is the command, is illustrated in Figure 3g.

CHAPTER 5

COMMUNICATION PROTOCOL

To insure reliable communication links, a well-defined communication protocol needs to be specified and implemented. A classically defined protocol solves operating problems in the following areas: framing, error control, sequence control, transparency, line control, special cases, timeout control, and start-up control [McNamara, 1978]. Each of these areas is addressed either directly or indirectly in the following discussion. However, prior to this discussion, a special note concerning the Intel 8031 microcontroller is first presented.

5.1. Comment on Intel 8031 Microcontroller

As stated previously, the central control computer in Figure 1 is a PDP-11. However, an Intel 8031 microcontroller was used to emulate the central control computer during the development of the communication protocol. The resulting dual-Intel 8031 configuration on each communication link facilitated testing of the different protocol aspects and expedited the project work. Since the PDP-11 also uses assembly level language to implement serial communication functions, much of the communication protocol has already been implemented on the PDP-11.

A second concern regarding the Intel 8031 is that some of its features were the limiting factors in trade-off decisions. Consequently, familiarity with the Intel 8031 and its capabilities may facilitate understanding of why a certain option was chosen. However, rather than discussing Intel 8031 features at several points throughout the chapter, a complete microcontroller description is contained in Appendix A. With this approach, the protocol discussion is left uninterrupted and the Intel 8031 information can be easily referenced.

5.2. Encoding Techniques

According to the message framing technique used, protocols are divided into three categories: character oriented, byte count oriented, and bit oriented. A character oriented protocol uses special characters to control communication flow. Byte count oriented protocols use a header which includes a beginning special character and assorted control information. A count following this header indicates how much data are actually being transmitted. Bit oriented protocols employ special bit strings to delineate message bits.

The protocol presented in this thesis exhibits characteristics found in both character oriented protocols and byte count oriented protocols, so it is difficult to classify it as one or the other. The decision to use this combined-protocol approach was heavily influenced by the prior exposure to a similar communication scheme [Silverman, 1984]. Commands and acknowledgements passed between modules are oftentimes just character triplets, which indicates a deference to a character oriented format. However, full data blocks of predetermined size are preceded by a header character, which is similar to a byte count oriented format. The following sections discuss several of the discriminating characteristics of this communication protocol in greater detail.

5.2.1. Character Encoding

In this communication protocol, the character triplets sent without supplementary data constitute a complete command or acknowledgement. Additionally, the character triplets used as headers signal the beginning of subsequent data byte transmissions. Each of these character triplets initiates a different action in the receiving module. Therefore, an encoding scheme is required to differentiate between the unique messages each character triplet represents. Several encoding schemes, including the one implemented in the hyperthermia system, are discussed below.

One possible encoding scheme requires that all ASCII characters used in triplets be sequentially ordered. Using this approach, a jump table can then be used to decode incoming triplets. An obvious advantage of this method is the ease of decoding associated with it, but this advantage is basically negated due to the relatively small number of valid character triplets that can be received. A major drawback of this scheme is the fact that the Intel 8031 provides no direct command used to perform relative jumps, although code can be written to perform the same function. Another disadvantage of sequential ordering is the fact that a transmission error could quite possibly change one valid message into another valid message because of the similarity of the ASCII characters used. This permutation can cause the wrong code section in the program to be entered. An extreme solution to this permutation problem is presented next.

A second encoding scheme utilizes the concept of distance, which is the number of bit transformations necessary to change one character into another. For example, the binary representations of the ASCII characters "A" and "B" are 1000001 and 1000010, respectively. These two characters exhibit a distance of two. The message representations for this second encoding scheme would be the set of characters exhibiting the greatest average distance between any two of the characters.

This approach is commonly referred to as m-out-of-n encoding, where n is the number of data bits per byte, and m is the fixed number of logical ones in each byte. With the ASCII characters used in this protocol, n is equal to seven, and m depends on the number of messages used. To illustrate, if ASCII characters are used to represent two messages, four-out-of-seven encoding yields 1111000 and 0001111 for the representations. A distance of six is exhibited in this example.

The goal of this second encoding scheme is to choose unique representations for each character that exhibit as little overlap of the m logical ones as is possible. Consequently, the greatest average distance between any two characters is established and single or double bit transformations do not necessarily yield another valid character. An obvious advantage of this method is the inherent error detection possible, assuming multiple bit transformations are not prevalent. However, creation of the unique characters is not a trivial task, and once the representations are established, message additions can create overlap problems.

In the third encoding scheme, which is the one used in the hyperthermia system, all characters are mnemonics indicating their respective function. Specific examples of this scheme are presented in the next chapter. The most signficant advantage of this approach is the pure simplicity of it. Although an incoming message must be compared with all valid characters until a match is found, the overhead involved is negligible due to the small number of messages. Also, new message addition presents no problem at all and the average distance between characters is greater than that found in the sequential ordering approach.

5.2.2. Data Encoding

In this hyperthermia system, several commands and acknowledgements are not complete without supplementary data. This section deals with the encoding schemes used to prepare these data for transmission. The majority of the information that needs to be transmitted ranges from zero to 15, so four-bit binary representations are used for each value. Although it is possible to concatenate two values into one byte and halve the necessary transmissions, this format is not utilized for two reasons.

The first reason concerns limit checks. If concatenation is used, resulting data bytes can range from 00 hexidecimal to FF hexidecimal. Because of this fact, transmission errors affecting data bytes would never be detected by simple limit checks conducted on received messages.

The second reason for not concatenating values concerns the amount of numeric information actually being transmitted in this initial system. The time limit imposed on the transmission of full data blocks is not so stringent that it requires halving the number of data byes. Future systems may require much larger amounts of information to be passed between modules, at which time it would be appropriate to condense the data and effectively halve the necessary transmission time.

Although concatenation is not used in this hyperthermia system, a majority of the information transmitted is modified before being sent. Values in the calibration data transmitted from the thermometry subsystem to the central control computer can range from 00 hexidecimal to FF hexidecimal, so further modification of individual bytes is not possible. However, the remaining values passed in this system range from zero to 15. Encoding of these values prior to transmission entails adding the value 30 hexidecimal to every byte. One advantage of using this scheme is that all received data bytes now lie in the range from 30 hexidecimal to 3F hexidecimal so limit checks will detect some transmission errors.

Another reason to add 30 hexidecimal to the bytes is that the resulting values are all ASCII characters found on any standard terminal keyboard. Using this format, it is possible to use a terminal to test a majority of the commands and acknowledgements containing supplementary data. The actual ASCII characters created by this encoding are presented in Table 1 (all tables appear following the Figures section).

5.3 Error Control

In all electrical information communication systems, transmission errors caused by external interference must be dealt with in some manner. In general, this error control is administered in two phases, error detection and error correction. The ultimate goal is to provide error control completely transparent to the user, except in special cases where the operator may have to initiate some form of corrective action.

Typically, transmission errors in communication systems tend to occur in bursts where a number of bits are affected. Single bit errors are less common, especially when higher baud rates are in use. To illustrate, a 0.01 second noise burst is not uncommon in telecommunication systems. At a baud rate of only 1200 bits per second, this burst has the potential of altering up to 12 bits of data. It then follows that a noise burst of the same duration can affect a proportionately larger number of bits when higher baud rates are employed. Additionally, these periods of high error rate are generally separated by relatively long intervals of little or no noise. Consequently, the error rate averaged over an hour is typically one error bit in every 100,000 bits received [McNamara, 1978]. The significant point illustrated here is that a system with adequate error control should be able to detect these burst errors and still have sufficient time to correct them before another burst occurs.

5.3.1. Error Detection

As a rule, the error detection scheme used in any system is highly dependent on both the type of data being transmitted and the communication capabilities of the individual modules. For example, one of the more common methods used to detect transmission errors is the Hamming code. It requires that four extra bits be appended to every seven bits of data in order to provide single bit error detection and correction. Since the Intel 8031 can send a maximum of nine data bits per byte in every serial transmission, a Hamming code application was not considered to be feasible. A second system considered to be very effective at detecting communication errors uses Cyclic Redundancy Checks (CRC). Both special hardware circuits and table-driven software algorithms designed to perform CRC calculations have been developed and are currently available on the market. However, CRC-equipped systems are designed to handle large volumes of synchronous data so this approach was not really appropriate for this system either. Instead, the error detection schemes actually used in this hyperthermia system were parity checks, redundant character (triplets) checks, expected acknowledgement checks, and checksums.

One simple form of error detection is the addition of a single parity bit to every transmitted byte. Prior to transmission, the eight data bits are loaded into the accumulator of the Intel 8031, which generates and stores the appropriate even parity bit into a Program Status Word (PSW) bit location. This parity bit is then copied to the most significant bit place in the transmit buffer and sent as the ninth data bit. When the Intel 8031 receives a character, it moves the eight least significant bits into the accumulator, thereby creating a new even parity bit. This generated parity bit is compared with the received ninth data bit and any discrepancies initiate a corrective action. The associated software overhead is very low and this action ensures that all single byte transmission errors comprised of an odd number of bit transformations are detected. However, errors involving even numbers of bit changes are left undetected by this parity check, so additional detection methods are also utilized.

Another method, character triplets, was first mentioned in Chapter 4. The redundancy is used to decrease the chance of an error going undetected. Upon reception, the three characters of the triplet must be identical or corrective action is initiated. Therefore, any invalid pattern not detected by the parity check must be repeated twice more to pass the redundancy check.

In a full data block transmission, the overhead required to repeat all information three times is too high, so a different detection method is used for the data bytes themselves. Prior to transmission, each data byte is added to a checksum originally initiated to zero. Eventual overflow of this checksum requires that it be two bytes long. Both of the bytes are then transmitted, low-order byte first, immediately following the data bytes. At the receiving end, a new additive checksum is also computed as each byte is received. This computed checksum is compared with the received checksum, and once again corrective action is initiated if the sums differ. A parity check done on the checksum bytes themselves completes the data block error detection efforts.

The final error detection mechanism considered in this section is used only by the central control computer. For reasons presented later, a master-slave relationship is established on both communication links. The subsystems are not allowed to initiate transactions, they can only acknowledge commands sent from the central control computer. Consequently, the central control computer always knows which acknowledgement is expected. Excluding special case commands, which are also discussed later, the received acknowledgement must then match the expected acknowledgement. If not, a transmission error has occurred. A successful detection by any of the processes presented above constitutes a call for some type of corrective action, which is discussed next.

5.3.2. Error Correction

There are two basic ways to provide error correction in a communication system. One method, called forward error correction, requires that the transmitter send enough supplementary information along with each character or data block to allow the receiver to correct the erroneous bit or bits by itself. The second method requires that the receiver request a retransmission whenever it receives erroneous information.

An example of the first method is the Hamming code discussed earlier which requires four extra bits to provide single bit correction capabilities. Another example of the first method commonly used with blocks of ASCII data is referred to as a vertical redundancy check/longitudinal redundancy check (VRC/LRC). The vertical redundancy check is simply the equivalent of the even parity bit appended to each character. The longitudinal redundancy check is similar to a checksum in the respect that all of the data bytes are exclusive-ored (XOR) together to yield a single byte. The resulting LRC is then transmitted as the last byte in the data block. Multiple bit errors can be located using a VRC/LRC map so the receiving module is able to correct the bits immediately. Figure 4 illustrates this process. Now considering disadvantages, just as a single parity bit cannot detect even-number-of-bit errors, certain error patterns also escape detection by a VRC/LRC map. One such case is illustrated in Figure 5. A second drawback of using this method is the software overhead associated with it, especially when programming at the assembly language level. Mainly because of this reason, the VRC/LRC map is not employed in this hyperthermia system.

The second method of error correction, which is the one used in this hyperthermia system, does not require supplementary data to be transmitted, but instead relies on a simple retransmission policy. Whenever an error is detected, the receiving module sends a retransmission request to the transmitting module. It must be noted that there are only certain times when a retransmission request can be sent. During the transmission of any character triplet not followed by supplementary data bytes, if either the first or second bytes are erroneous, no retransmission request may be made until the third byte has been received. Additionally, communication in the hyperthermia system is designed so that a receiving module knows when to expect a full data block as an acknowledgement. In this case, the request is sent only after the last checksum byte has been received. It does not matter whether the error occurred in the header, the data bytes, or the checksum bytes; the transmitting module is always allowed to finish sending first.

To illustrate, first assume that an offending noise burst affects only the first character triplet transmitted. Upon reception of the erroneous triplet, the receiving module requests a retransmission. This request is then processed by the transmitting module and the first character triplet is sent again. If it is received correctly, the receiving module can then process the command or acknowledgement accordingly. However, the possibility exists that the second attempt is also transmitted incorrectly. In this case, the retransmission request is sent to the transmitting module once again and a retransmission counter in the receiving module is incremented. This exchange continues until either the original triplet is transmitted cleanly, or the limit of the retransmission counter is reached. If the former occurs, the triplet is processed, the retransmission counter is reset, and normal communication continues. If the latter occurs, the receiving module assumes that the transmission line is bad and a shutdown process, described later in this chapter, is initiated. At the present time, the retransmission limit is set at three, so on the fourth consecutive reception of bad information, the shutdown begins.

The above cases depict a scenario where only transmission in one direction is interfered with. However, it is just as likely that for a short period of time both modules receive erroneous transmissions. If this occurs, both units begin to send retransmission requests. Under the worst conditions, a dual shutdown is eventually initiated. Unfortunately, even if both units recover before the retransmission limits are reached, it is quite possible that the first module to recover will send the wrong message. This possibility is illustrated in Figure 6. The significant point to note is the fact that in both cases, Module Two has received the same sequence of messages from Module One, but different retransmissions are actually required. Thus, not enough information is present to choose the correct reply. The addition of sequence numbers to every byte in a character triplet alleviates this problem.

A single bit sequence number is ideal for this system because of the eighth data bit left unused by ASCII characters. In order to maintain strict sequence control, a restriction is imposed on the hyperthermia system communications. Specifically, a transaction where the sequence state equals zero must be completed before a transaction where the sequence state equals one is initiated, and vice versa. To fulfill this requirement, every command in this system is normally followed by an acknowledgement. This constraint is the reason for establishing the master-slave relationship on both communication links. Allowing both modules to send commands creates problems because it disrupts the sequence control. Therefore, the central control computer initiates all transactions and the respective subsystems complete them. Exceptions to this master-slave scheme exist and are discussed in the next chapter.

A set of rules used to accomplish the sequence control is presented in Table 2. Applying these rules to the original retransmission problem from Figure 6 yields the correct solution, illustrated in Figure 7. Several other retransmission problems, and the correct handling of them, are presented in Figure 8. The ability to recover from these transmission errors represents a major step in providing complete error correction. However, the possibility exists that some errors will never trigger the detection mechanisms already discussed. This scenario is examined next.

5.4. Timing Functions

The aforementioned detection processes are all based on the premise that commands and acknowledgements always arrive at the receiving module, whether they are faulty or correct. Given certain conditions, this constraint renders the detection processes useless. For example, two identical, valid command bytes with the proper parity will not trigger any detection mechanisms because the triplet is incomplete. Likewise, a parity-correct full data block minus a single checksum byte will never reach the checksum compare stage. In both cases, the receiving module effectively waits for the character triplet or full data block to be completed before it proceeds with complete error detection. Thus, a lost message or even a single lost byte hangs the error detection process.

To counter these problems, several software timers are utilized in the individual modules as a last means of detecting transmission errors. Because of the master-slave relationship established on each communication line, different types of timing functions are needed. Those functions are discussed in the next four sections.

5.4.1. Expected Acknowledgement Timer

Since the central control computer initiates all transactions, only an incomplete acknowledgement creates a problem. Whether the command is a character triplet or requires supplementary data, an Expected Acknowledgement Timer is activated immediately after transmission of the last byte. Since only one timer is activated for all different structures, the timer period is long enough so that a full data block can arrive and be processed as the acknowledgement.

Several possibilities exist once the timer is activated. If a complete acknowledgement, correct or incorrect, is received by the central control computer, the timer is deactivated and reinitialized for the next transaction. An incorrect acknowledgement warrants the start of the retransmission process already discussed, while a correct acknowledgement completes the structure. On the other hand, an incomplete acknowledgement allows the Expected Acknowledgement Timer to overflow. If an overflow occurs, the central control computer assumes that the initial message never reached the subsystem, so it sends the original command again. It is possible that this assumption is incorrect; in other words, the initial command was received by the subsystem, but the returning acknowledgement was lost instead. However, retransmitting the original command does not create a problem because the sequence number algorithm allows both instances to occur and still restores proper communication. The two possibilities are presented in Figure 9.

Whether a retransmission request is initiated by the Expected Acknowledgement Timer overflow, or by the previously discussed detection procedures, the retransmission counter is still incremented. Thus, any combination of error detections or timer overflows totaling four consecutive mishaps initiates the shutdown process. Therefore, the Expected Acknowledgement Timer provides the last transmission error detection mechanism in the central control computer. The possible lost-message problems of the subsystems are more complex and require multiple timers.

5.4.2. Triplet Timer

Since the subsystem waits to receive a command from the central control computer, it is not able to utilize any type of expected command timer under normal operating conditions. The one exception occurs during a retransmission request to the central control computer, which is covered presently. To continue, the subsystem cannot activate any timers until the first command byte is received from the central control computer. Upon reception of that first byte, the Triplet Timer, designed to ensure that three bytes are received, is activated. The reception of the third byte deactivates and reinitializes the timer, and processing of the message is started. A timer overflow increments the subsystem's retransmission counter and prompts a retransmission request. The transmission of this request in turn activates a second timer, which is discussed next.

5.4.3. Expected Retransmission Timer

For each subsystem, requesting a retransmission represents the only instance where an immediate acknowledgement is expected. Therefore, this timing function is similar to the Expected Acknowledgement Timer previously discussed. A correct retransmission deactivates the Expected Retransmission Timer and reinitializes both the timer and the retransmission counter. A timer overflow prompts the subsystem to send another retransmission request and increment its retransmission counter. Like before, consecutive timer overflows eventually cause a shutdown. Therefore, the Triplet Timer and the Expected Retransmission Timer in the subsystem effectively simulate the function performed by the Expected Acknowledgement Timer in the central control computer.

5.4.4. Line Viability Timer

To determine the condition of the communication line in use, the central control computer needs only to initiate a transaction and then examine the quality of the received acknowledgement. If either of the RS-232-C data transmission lines is out, the Expected Acknowledgement Timer overflows four consecutive times and the shutdown state is entered. However, because the subsystems do not initiate transactions, a new problem arises. If the transmit line from the subsystem to the central control computer is out, only subsystem-initiated messages are lost, and the subsystem can still be shut down by the central control computer. On the other hand, if the transmit line from the central control computer to the subsystem is out, the subsystem would theoretically wait forever for the next command. All commands are lost and the control loop so vital to the success of this hyperthermia system is broken. Additionally, a potentially dangerous scenario emerges in the case where the PDP-11 loses control of the ultrasound applicator. Therefore, another timing function is installed in both subsystems to prevent this problem from occurring.

Specifically, a Line Viability Timer is created to detect relatively long periods when no commands are received. Once the system is initialized, this timer is activated and runs continually until the treatment is concluded. The reception of every first byte of a command reinitializes the timer but does not deactivate it. In the event of a timer overflow, the subsystem attempts to inform the central control computer it is shutting down, and then does so. The timer period is longer than any no-transmission period normally allowed, so the one-time-only approach is not too drastic in this case. No other alternatives are present at the time, so the subsystem initiates a shutdown process, which is considered next.

5.5. Shutdowns

The shutdown process differs depending on which module initiates the procedure. Only the subsystems can be shut down, the central control computer is always left running. If a subsystem initiates the shutdown process, it first informs the central control computer of the impending shutdown and then turns itself off. This is obviously one case where the central control computer is not expecting a transmission, but due to the nature of the problem, any communication repercussions it causes are inconsequential.

If the situation arises where the central control computer encounters severe communication problems, it commands the subsystem to shut down. However, in this case, it does not expect any acknowledgement back. In the event that the subsystem never receives the command, the Line Viability Timer eventually overflows and the shutdown is self-initiated.

All shutdown states discussed so far have been complete and final, and require a total system reinitialization to recover. Although they are not currently utilized in the hyperthermia system, it is possible to first try a partial shutdown, followed by some type of an attempt to salvage the current treatment. For instance, since the ultrasound applicator is multielemental, a detected malfunction of only a few elements may not demand a full shutdown. It may be possible to just shut down the faulty elements and still continue treatment by readjusting the intensities of the remaining elements. In this case, a partial shutdown may coincide with a prompt for operator assistance and a time limit on any corrective action may be imposed on the operator. This concept is still in the formative stage and additional data on system performance and tissue response are needed before further decisions can be made. At that time, feedback from system tests can be used to determine the feasibility of partial shutdowns.

CHAPTER 6

COMMAND-ACKNOWLEDGEMENT STRUCTURES

This chapter specifies the actual commands and acknowledgements used in this hyperthermia system and discusses the function each message performs. In all cases, full structures are examined, so each command is paired with its corresponding acknowledgement, if it has one. The mnemonic encoding of each character is presented, in addition to the specific class each structure represents.

6.1. Common Structures

Although the functions performed by the two major subsystems are very different, several of the structures used are identical. Before focusing on the individual subsystems, these common structures are first discussed.

-- Name/Status - Identification/Status

The Name/Status command is used by the central control computer for two purposes. The first transmission is designed to verify the correct hardware configuration of the system and prevent instances where the serial transmission ports of the central control computer have been incorrectly assigned. If an incorrect reply is received, the operator is informed so that corrective action can be taken. Subsequent transmissions of the Name/Status command by the central control computer are used to inquire about the status of the respective subsystem. As the status changes, the Identification/Status acknowledgement also changes. Thus, this structure represents a Class 3 transaction.

The mnemonic used for the Name/Status command is the alpha character "N." The Identification/Status byte is broken up into two parts; the low nibble is the module identification (ID), and the high nibble indicates one of eight possible status states. Only three bits are reserved for indicating status because the eighth data bit, the most significant bit of the high nibble, is reserved for sequence numbers. The thermometry subsystem's ID is the hexidecimal value 0B and the ultrasound applicator subsystem's ID is the hexidecimal value 0C. The possible statuses of each subsystem are discussed in later sections.

-- Initialize and Go - Done

Once the central control computer receives a "system ready" status from each subsystem, it sends the Initialize and Go command to each subsystem. When the thermometry subsystem receives this command, it becomes able to transmit temperature data. Initialize and Go also allows the applicator subsystem to begin applying ultrasonic energy. The command is represented by the alpha character "I" and the expected acknowledgement from both subsystems is a simple Done, represented by the character "D." This structure represents a Class 2 transaction.

-- Retransmit

The retransmission request mentioned in previous chapters is formally represented by the Retransmit message. Either module on a communication line can send it in response to an erroneous message received from the other module. The Retransmit structure is difficult to classify because the Retransmit message itself can function like a command or an acknowledgement. When the message is sent by the central control computer, it functions like a command because it initiates a transaction and an acknowledgement is expected back. When the message is sent by a subsystem, it completes the previous transaction so it is considered to be an acknowledgement. Additionally, the message completing the Retransmit structure may be either a character triplet without supplementary data, or a character triplet with supplementary data. Therefore, Retransmit structures can represent Class 2, 3, 4, 5, or 6 communications. The mnemonic encoding for the Retransmit command is the alpha character "R."

-- Shutdown

The shutdown process mentioned previously is initiated by the Shutdown command. Since an acknowledgement is never expected for this command, it is a Class 1 structure. The mnemonic encoding for the Shutdown command is the alpha character "S."

The four transactions just discussed represent the common structures used on both communication links. Specialized structures created for each subsystem are considered next.

6.2. Thermometry Subsystem Communication Link

The Intel 8031 microcontroller-based multielement thermocouple thermometry unit used in this hyperthermia system combines well-known thermoelectric thermometry with state-of-the-art microprocessor-based control, calibration, and display capabilities [TX-100 Operating Instruction Manual, 1984]. Assembly-level code for the communication functions is stored in program memory provided on the Central Processing Unit (CPU) board, which integrates all the support circuitry for the Intel 8031 microcontroller. The External Communication Board (ECB) contains standard interface chips to connect the TTL-compatible serial port of the microcontroller to the RS-232-C communication lines. Additionally, the interface port is optically isolated from the rest of the subsystem.

6.2.1. Operation Overview

Using small copper-constantan thermocouples, 16 individual points in the treatment area are monitored. However, before accurate temperatures can be obtained, softkeys located on the front panel must be used to step through an internal single-point calibration scheme. The calibration coefficients and other useful calibration data are then stored in random access memory (RAM) on the CPU board. Each thermocouple channel selected by the user is then monitored by the microcontroller and thermoelectric voltages developed in the individual probes are obtained. With the help of an Amplifier and Analog-to-Digital (AAD) board, the Intel 8031 performs the necessary numeric calculations which result in the desired calibrated temperature data. These temperature data are also stored in RAM on the CPU board and are periodically sent to the red digital displays (LED) located on the subsystem front panel. In addition, a front panel liquid crystal alphanumeric display (LCD) indicates current subsystem communication status.

6.2.2. Specialized Structures

Besides using the four common command-acknowledgement structures already discussed, the thermometry subsystem utilizes the two transactions presented below.

-- Load - Unload

The Load command is used by the central control computer to obtain calibration data from the thermometry unit. The single-point calibration scheme run by the user creates five bytes of calibration coefficients for every channel actually calibrated. Thus, a maximum of 80 coefficient bytes can be produced. In addition, 48 more bytes of miscellaneous calibration data are stored in RAM on the CPU board. Therefore, the central control computer expects a full data block back starting with the Unload header and containing 128 data bytes. This transaction is included so the calibration information received from the subsystem can be safely stored on disk in the PDP-11 in case it is needed later. As the treatment continues, the temperatures received by the thermometry unit are monitored by the PDP-11, and if major discrepancies appear, the calibration information can be reloaded back to the thermometry subsystem in hopes of salvaging the treatment. The alpha character "L" is the mnemonic encoding for the Load command and the Unload header is represented by the alpha character "U." This structure represents a Class 6 transaction.

-- Temperatures - Receive Temperatures

The Temperatures command is sent by the central control computer whenever new temperature data are needed. Each of the 16 channels has a four digit readout accurate to the hundreds place. The resulting 64 data bytes are not stored as ASCII characters so some reformatting is necessary before transmission occurs. The central control computer expects a full data block back starting with the Receive Temperatures header and containing 64 data bytes. The Temperatures command is represented by the character "T" and because the Retransmit message already uses "R," the Receive Temperatures header is encoded with an "E." This structure represents a Class 6 transaction.

6.2.3. Status Codes

At the present time, only three of the eight possible status codes are defined for the thermometry subsystem. Status 0 indicates that the thermometry subsystem is still initializing. Status 1 means that the unit is ready to begin the calibration routine. Status 2 is reached when the calibration is completed and the thermometry subsystem is ready to unload its calibration information. For the remainder of the treatment, the thermometry unit remains in Status 2 until a system reset occurs.

6.2.4. Transaction Sequence

The communication flow for the thermometry subsystem-central control computer link was first presented in general terms in Section 2.2. This flow can now be illustrated using the actual structures presented in this chapter. Since transmission errors and sequencing control were considered in previous sections, this section deals only with ideal transmission conditions. In order to illustrate the typical transaction sequence for the thermometry unit, a series of error-free exchanges are presented in Figure 10.

6.3 Applicator Subsystem Communication Link

The Intel 8031 microcontroller-based applicator subsystem provides control for 16 independent RF power amplifiers. Similar to the thermometry unit, the interface port is optically isolated from the rest of the subsystem. Also, the External Communication Board and the CPU board mentioned previously are duplicated in this subsystem, so the same programming and communication capabilities are present. The addition of a 16-channel controller card provides the ability to successfully control the applied ultrasound field. The two internal frequencies available in the module, 1 MHz and 3 MHz, are software controllable. Additionally, the controller card allows the power output from each amplifier to be varied independently by changing the duty cycle of each amplifier input over 10% increments. The power level of all 16 amplifiers in common is also variable over the full output power range in 16 steps. This is accomplished by varying the voltage output of the main DC power supply.

6.3.1. Operation Overview

The adaptive thermal modeling algorithm run in the central control computer provides new control parameters to the applicator subsystem in terms of changes in the main power supply output and/or changes in individual duty cycle values. Once a change has been received, the applicator processes the new values and applies it to produce the desired outputs. Elaborate processes similar to the calibration scheme and the temperature acquisition algorithm are not used in the applicator subsystem, and in general, the internal processing required for this subsystem is not too intensive. However, the communication capabilities of this subsystem are greater because processing for several specialized structures is necessary to maintain control of the subsystem.

6.3.2. Specialized Structures

In addition to the four common structures discussed earlier, a total of five additional structures are employed by the applicator subsystem.

-- Frequency - Done

The Frequency command is sent by the central control computer to establish the internal subsystem frequency at either 1 MHz, the current default value, or 3 MHz. To differentiate between the two values, the ASCII representation for the desired frequency is transmitted immediately after the actual Frequency command. Therefore, this transaction is a Class 4 example. The expected acknowledgement is a Done message. Mnemonic encodings for the Frequency command and the Done acknowledgement are the alpha characters "F" and "D," respectively.

-- Voltage - Done

A second Class 4 structure is the Voltage command used to set the main DC power supply output level. Since 16 steps are possible, the hexidecimal values for zero through F are used to represent the voltage levels. As discussed earlier, the value 30 hexidecimal is added to the desired digit prior to transmission. The applicator subsystem receives the Voltage command and the voltage level, processes the value, and sends back the Done acknowledgement. The character "V" is the encoding for the Voltage command.

-- Receive Duty Cycles - Done

This command is used by the central control computer to alter the duty cycles of the 16 individual amplifiers. Even if all the channels are not used, 16 duty cycle bytes are still sent. The possible duty cycles are represented by 0 through A hexidecimal so the value 30 hexidecimal is again added before transmission. The Receive Duty Cycles command is a full data block starting with a header and containing 16 data bytes. The expected acknowledgement is Done, so the structure is a Class 5 transaction. Because "R" is already used, the alpha character "E" is used to represent the Receive Duty Cycles header.

-- Wait - Done

The Wait command is included so that the central control computer can temporarily disable the applicator subsystem output, without loss of any of the control information previously sent. In addition, new control information can be received and processed while in this wait state and the Initialize and Go command can then be used to enable the outputs again. The mnemonic "W" is used for this command and the structure is a Class 2 transaction since Done is the expected acknowledgement.

-- Help

The Help command is the only command sent from a subsystem that truly initiates a transaction. It is not actually used in the current system, but will probably be used in partial shutdown situations. Internal fault checking conducted by the applicator subsystem includes a comparison of the controller board digital outputs with the desired duty cycle values obtained from the central control computer, and comparison of the reflected and forward power levels of the amplifiers with their respective predetermined limits. Any detected discrepancy immediately triggers a self-shutdown in the current hyperthermia system. However, by utilizing a Help command, a treatment salvage may be possible. Once the central control computer receives the Help command, it immediately issues the Name/Status command and the ensuing status it receives back indicates the problem. The central control computer can then take some form of corrective action, or issue a Shutdown command if it is indeed necessary. Help is a pseudo-Class 2 structure since the expected acknowledgement is a Name/Status command, although it might also be considered a Class 1 special case transaction. The alpha character "H" is the encoding for this command.

6.3.3. Status Codes

Because shutdowns are used to handle all internal faults in the applicator subsystem, the only status code defined at this time is the zero code, which indicates that the module is ready.

6.3.4. Transaction Sequence

In general, the transaction flow across the applicator subsystem communication link is more varied than that encountered with the thermometry unit. Similar to the approach used for the thermometry subsystem, a typical transaction sequence during error-free operation of the applicator subsystem is presented in Figure 11.

6.4. Comment

In order to clarify the material presented in this chapter, a list of all the structures discussed for both subsystem links is presented in Table 3.

CHAPTER 7

SOFTWARE

The communication protocol presented in this thesis is fully implemented in three Intel 8031 assembly level programs. Flowcharts, which provide the best documentation of the programs, are presented in Appendices B, C, and D. Listings of these programs together with brief descriptions of the variable names used in the programs are located in Appendices E, F, and G. This chapter contains short verbal descriptions of the programs for introductory purposes.

7.1. Central Control Computer

As stated in Chapter 2, a PDP-11 is the central control computer used in the actual hyperthermia system. Since the central control computer program used in this thesis is written for an Intel 8031, the PDP-11 functions not directly related to communication features, such as the adaptive thermal modeling algorithm, are omitted. This simulation program was written solely to test the communication features presented in this thesis. In addition, this particular simulation program interacts only with the thermometry subsystem software. The implementation of protocol aspects, such as sequence number control and timing functions, only needed to be written and debugged for one subsystem. Once proven, they can be easily duplicated in the applicator subsystem.

The emulated central control computer source code can effectively be broken down into three major sections. The first section, which contains code for the main routine, is entitled INIT. The second section, CENTRAL, and the third section, SERIAL, are both interrupt handlers for the Timer 0 interrupt and the serial port interrupt, respectively.

INIT contains the code to initiate all transactions encountered in error-free operation (see Figure 10). A loop designed to transmit Name/Status commands is implemented along with a delay necessary to keep the thermometry unit's calibration scheme from being interrupted. Code to transmit the Load command and the Initialize and Go command is also present in INIT, as is the code for the Temperatures command transmission. The latter command is initiated from inside an eight-second delay loop designed to simulate the repeated transmission of the Temperatures command during steady-state communication flow.

CENTRAL contains code to implement the sole timing function, the Expected Acknowledgement Timer, required in the central control computer. Although the Expected Acknowledgement Timer is activated and deactivated from different points in the program, the actual timer and the code to handle timer overflows and possible shutdown initiations are included in CENTRAL.

SERIAL, the last major section, can actually be examined in two subsections. The first subsection deals strictly with transmit interrupts, which occur when the Intel 8031 finishes transmitting a byte. Code contained here ensures that three repetitions are sent for every character triplet and that all full data blocks are properly transmitted. Additionally, once a Shutdown command has been completely transmitted, the program jumps to a routine that disables all interrupts and simulates module shutdown. Features such as sequence number generation, Expected Acknowledgement Timer activation, and parity bit generation are all implemented at appropriate places in the code.

The second subsection deals only with receive interrupts. All received character triplets and full data blocks are first checked for proper format. Error detection features, such as parity checks and expected reply checks, are relatively centralized in this reception subsection while other features such as timer deactivation, sequence number checks, and limit checks are more interspersed throughout SERIAL. The code to process all of the expected acknowledgements is contained here along with the code for special case messages such as Retransmit or Shutdown. The flowchart pertaining to the emulated central control computer program is presented in Appendix B and Appendix E contains descriptions of the variable names used and the actual assembly-level code.

7.2. Thermometry Subsystem

The program written for the thermometry subsystem is very similar to the program just discussed and is also divided into the same three sections. However, the main routine in this program, also entitled INIT, handles all of the user interaction through the softkeys and the majority of the message displays on the front panel LCD. Calls to several subroutines, such as the calibration routine, are contained in INIT in addition to several subroutines pertaining to internal processing in the thermometry subsystem. However, these subroutines are not considered in this discussion or in the flowcharts because they are not communication functions.

The subroutines and calls are included in the program listing only to preserve the program continuity.

The remaining two sections, the Timer 0 interrupt handler, TIMERS, and the serial port interrupt handler, SERIAL, are nearly identical in purpose to their counterparts in the emulated central control computer program. The basic difference between the Timer 0 routines is that three timers instead of just one are now implemented. The obvious difference between the serial interrupt handlers is the role reversal involved. SERIAL in the thermometry subsystem program processes commands instead of acknowledgements in its receive interrupt subsection and handles the acknowledgements in its transmit interrupt subsection. Except for these basic differences, the other protocol functions are implemented as before. The flowchart for the thermometry subsystem program is presented in Appendix C, and the code and the descriptions of the variable names are located in Appendix F.

7.3. Applicator Subsystem

The program for the applicator subsystem differs from the first two programs discussed in the respect that it lacks protocol features such as sequence number control and timing functions. The current PDP-11 program designed to interface with this applicator subsystem program also lacks the same protocol features. Since these features were tested on the thermometry subsystem communication link, they can be incorporated into the applicator subsystem program at the same time they are added to the PDP-11 program. The applicator subsystem program also contains three major sections because of the added internal fault checks. The main routine, INIT, initializes the subsystem and then enters a loop that continually checks the digital outputs of the controller card. The Timer 0 interrupt handler, DUTY, implements the forward power-reverse power amplifier checks discussed earlier. In addition, it also handles the duty cycle gating for the 16 amplifier channels. As expected, the serial port interrupt handler, SERIAL, is similar to that for the thermometry subsystem except that different structures are processed. The flowchart for the applicator subsystem program is presented in Appendix D and Appendix G contains the actual code and descriptions of the variable names used.

CHAPTER 8

RELIABILITY AND TESTING

In order to ensure that the specified communication protocol was implemented correctly, the different aspects of it were first tested independently. Once all the faults were corrected in one test program, another protocol feature was added to the code, and tests designed to check the new feature were then conducted. Many of the piecemeal tests were fairly trivial, but they served to ensure the correctness of the protocol implementation as each feature was added. Consequently, problems encountered in a new test version had to be caused by the new feature addition, which greatly simplified the debugging process.

One of the first tests conducted involved parity error detection. A terminal with configuration switches on the back panel, as opposed to one exhibiting keyboard setup, was utilized for this test. With the switches configured for odd instead of even parity, valid ASCII characters were transmitted to the Intel 8031. The receiving program processed the bytes and entered a delay loop which allowed the terminal to be reset for even parity. The resulting error indicators returned to the terminal then validated the parity error detection code.

Another protocol aspect tested was the transmission and reception of character triplets. Since a terminal could also be used for this test, many possible error conditions were examined. Besides testing only character triplets without supplementary data, header triplets for full data blocks were also examined. As noted earlier, retransmission requests stemming from triplet errors can only be sent at predetermined points in the communications, so this concept was also checked at this time.

Retransmission requests, both transmitted and received, were thoroughly checked out once all the desired codewords were implemented. In the central control computer test, the expected reply notion was tested and successfully triggered a retransmission request. Likewise, several permutations of the data block format were transmitted, and retransmission requests were always received back. Thus, the limit checks and checksum generators were considered to be working properly for all of the data blocks.

Once the hardware was complete, and it was known that the communication software was detecting all transmission errors except for lost characters, a terminal was connected to each of the subsystems. All of the central control computer commands were then tried and it was verified that both the correct reply was received back and that the hardware responded correctly to each command. Additionally, a series of four consecutive bad transmissions to the subsystems first prompted retransmission requests and finally a self-shutdown.

All four timing functions were then implemented in their respective programs. Because terminals were again used to conduct preliminary tests, several of the timer periods had to be lengthened to allow for the keyboard entry of characters. Error situations were created, and once all four functions were observed to be working correctly, the two programs were allowed to communicate directly. A test of temperature acquisitions was then left running for an extended period of time to ensure that none of the timers was triggering an unneeded retransmission request.

Up to this point, all aspects of the protocol were able to be tested first with a terminal-to-Intel 8031 connection, and then with a dual-Intel 8031 connection. However, terminals could not be used to test the sequence number generators because the addition of a sequence number "1" as the eighth bit creates a non-ASCII character. Consequently, all sequence control tests were conducted by allowing the two programs to communicate with each other. Eliminating the terminal was inconvenient in the respect that any new test usually required a short modification to be made to both programs instead of just one. However, many of the anticipated problems were tried and the results were used to refine the sequence number control software.

CHAPTER 9

FUTURE CONSIDERATIONS

At the present time, the PDP-11 communication program utilizes a subset of the full communication protocol. Although integration of the remainder of the protocol presents no foreseeable problems, one area could possibly use some revision. Help, which is a special case command, can not utilize sequence number control because it does not follow the normal communication flow. If the Help command is received cleanly by the PDP-11, the sequence number can be ignored and the proper action can be initiated. However, if the transmission is garbled, the PDP-11 will request a retransmission. Because of the sequence number algorithm, the applicator subsystem will receive this request and then send back a retransmission request of its own. This transaction will prompt the central control computer to send the previous command, and the Help message will effectively be ignored.

One possible way to avoid this incident would be to remove the Help command completely from the software and perform the same function with a combination of hardware and software. In this protocol, the Help command is used only as a signal to the central control computer that a problem has been encountered in the applicator subsystem. The ensuing Name/Status - Identification/Status exchange actually informs the central control computer of the applicator problem. Therefore, a fifth lead added to the RS-232-C connection could be used to transmit this signal and accomplish the same function. This lead could be connected to an external pin already provided by the Intel 8031. The lead at the PDP-11 side could be connected to one of the pins supplied for hardware handshaking functions. A low state would indicate no problem and a high state would initiate the Name/Status transmission from the central control computer. Relatively few hardware and software modifications are necessary to make this change and the resulting procedure probably would be more reliable than that currently used.

A second possible modification involving a RS-232-C hardware change involves connecting some of the control leads, presented earlier in Figure 2a, to provide handshaking capabilities. The function of each lead would probably differ slightly from that defined by the RS-232-C standard, but the net effect would be to ease the master-slave relationship currently established on each communication link. Thus, each module on the link would be able to initiate transactions without disrupting the communication flow. This option would increase the overall capabilities of the protocol, but would require several hardware modifications and would increase the software overhead associated with each exchange.

Another way to obtain handshaking capabilities involves an even more radical hardware change. Specifically, the current interface standard for the system, EIA RS-232-C, could be switched over to the IEEE 488 standard. Although major hardware interface modifications would be required to complete this change, the resulting system gains some important advantages. A knowledge of the standard is assumed for the following discussion, so users may want to reference the IEEE Standard 488-1978 "Digital Interface for Programmable Instrumentation." Additionally, the interface at the subsystems is of primary interest, so the consequences of the change are examined basically from the viewpoint of an Intel 8031 user.

From a cost standpoint, the change is feasible because of a reasonably priced four-chip set currently available from Intel. The Intel 8291A GPIB Talker/Listener (Intel 8291A) implements most of the IEEE 488 standard's required functions. Without any Intel 8031 involvement, this chip can handle data transfer, handshake protocols, listener/talker address procedures, device clearing and triggering, service requests, and parallel and serial polling schemes [Intel's Microsystem Components Handbook, 1984].

In terms of this hyperthermia system, one specific advantage of using an Intel 8291A is that the handshake protocols utilizing the Source Handshake (SH) and Acceptor Handshake (AH) guarantee the success of asynchronous transfers. Lost messages would trigger protocol errors which immediately interrupt the microprocessor. A second advantage concerns the transmission of data blocks. A built-in End-of-Sequence (EOS) register can be used to store delimiters for multibyte transmissions. A match with a received byte then asserts the End or Indicator (EOI) line which verifies the receipt of the block. A third advantage involves the Service Request (SR) function each subsystem would control. The serial status poll generated by this function would be ideal for implementing the special case commands, Help and Shutdown. Additionally, the Parallel Poll (PP) function could be used at start-up and even at set intervals during the treatment to obtain subsystem status readings. Another advantage concerns the Remote/Local (RL) function, which could be used to prevent the thermometry subsystem from being interrupted during the calibration process. Finally, it would also be possible to reset the entire system using the Device Clear (DC) and Device Trigger (DT) functions.

A second chip, the Intel 8292 GPIB Controller (Intel 8292), can be added to the Intel 8291A to form a complete IEEE 488 interface, capable of handling the transfer control protocol. Although multiple controllers would not be necessary in this system, the addition of an Intel 8292 would allow the subsystem to synchronously gain control of the bus in critical situations by using the Interface Clear (IFC) function. This prevents the destruction of any bytes on the data lines and would ensure a prompt response to the subsystem call. Completing the four-chip set are two Intel 8293 GPIB Transceivers which allow direct interface to the General Purpose Interface Bus.

A disadvantage associated with the IEEE 488 standard involves cabling restrictions, which state that the maximum length of cable that shall be used to connect together a group of devices within one bus system is: (1) two meters times the number of devices or, (2) 20 meters, whichever is less. Therefore, individual cable lengths cannot exceed four meters without the addition of in-line drivers which are also commercially available.

A second disadvantage stems from the fact that the standard was designed to apply generally to laboratory and production test environments which are relatively electrically quiet. Consequently, the hyperthermia system is probably considered to be an extended application, and would require different electrical and mechanical specifications to provide increased noise immunity and greater separation of devices.

Additional hardware changes include the system reconfiguration needed to interface the Intel 8031 to both the Intel 8291A and the Intel 8292. Also, start-up procedures create some extra software overhead, but this occurs only during the total system initialization. In retrospect, an interface switch to the IEEE 488 standard would not be a trivial undertaking, but the end result may provide justification for doing so. Error control aspects of the protocol would still be necessary, but communication in general would be more reliable, and the talker/listener capabilities added to the system would greatly increase the power of the overall communication system.

CHAPTER 10

SUMMARY

In this thesis, a digital communication protocol for use in a modular hyperthermia system was presented. The specification of the protocol itself was described in very general terms in the hope that future enhancements and/or additions can be integrated easily into the system. Specific structures used in the protocol were also examined, along with actual software implementations. The programs written to implement the communication protocol were tested and observed to be working correctly. Several ideas and recommendations pertaining to future protocol work and to complete communication systems were discussed. The presentation was organized in such a way as to allow one to focus on individual aspects of the protocol, if desired.

FIGURES

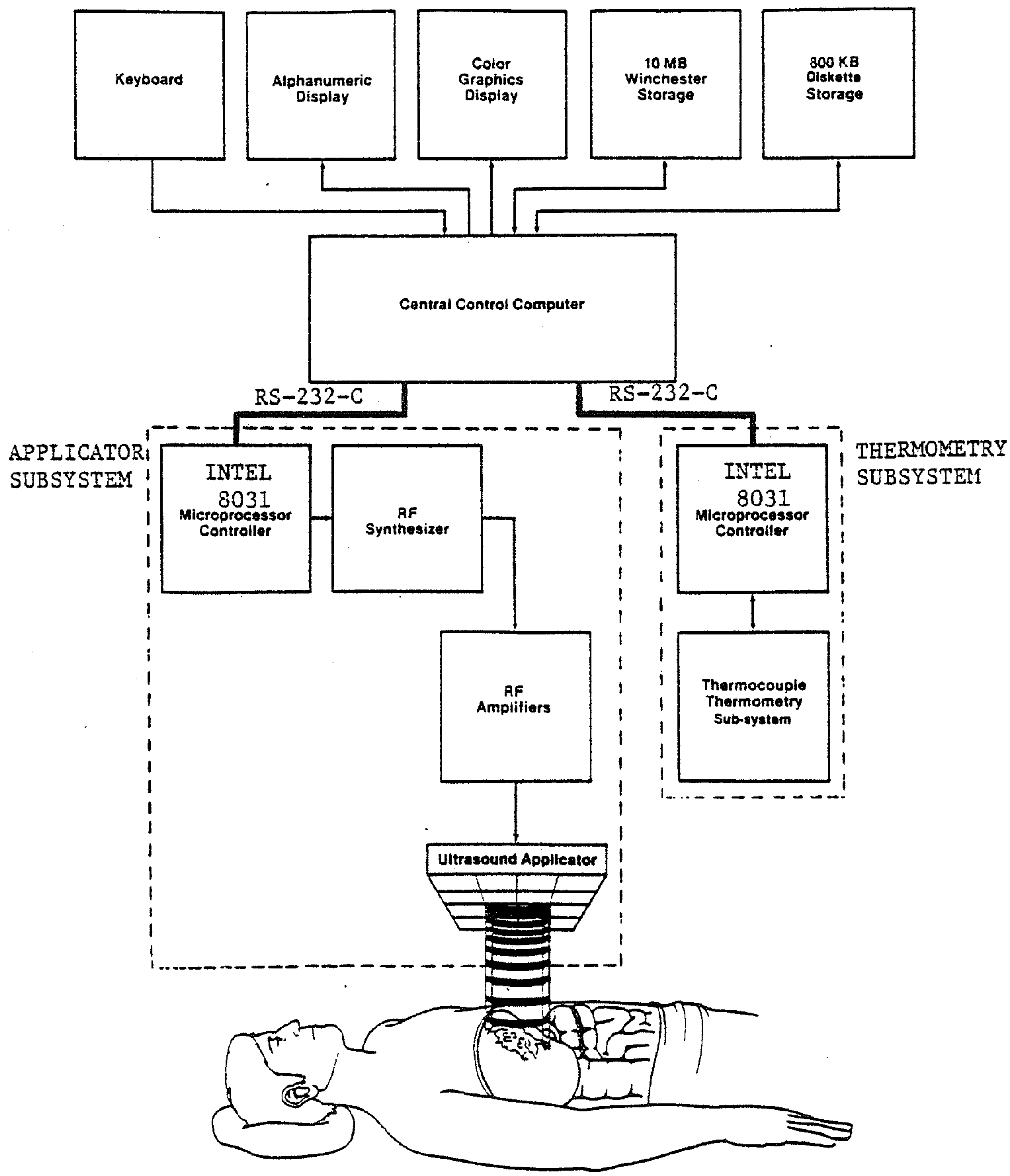
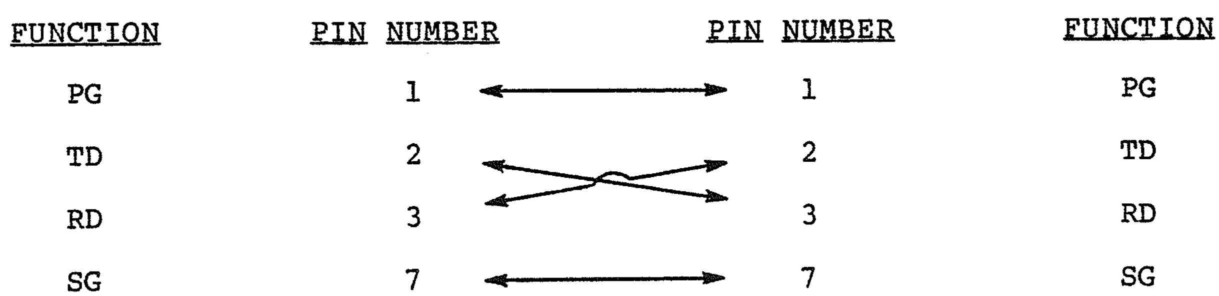


Figure 1. Hyperthermia system block diagram.

PIN NUMBER	CATEGORY	FUNCTION
1	Ground	Protective ground (PG)
2	Data	Transmitted data (TD)
3	Data	Received data (RD)
4	Control	Request to send (RTS)
5	Control	Clear to send (CTS)
6	Control	Data set ready (DSR)
7	Ground	Signal ground (SG)
8	Control	Data carrier detect (DCD)
20	Control	Data terminal ready (DTR)

(a)



(b)

Figure 2. RS-232-C configuration. (a) Nine asynchronous leads for RS-232-C. (b) Four-pin cross connected interface.

```
CLASS 1                  CLASS 2                  CLASS 3
                         A---->                   A---->

                         A---->                   A---->

                         A---->                   A---->

A---->                    <----A                   <----V

A---->                    <----A                   <----V

A---->                    <----A                   <----V

 (a)                       (b)                      (c)


CLASS 4
A---->

A---->

A---->

D---->          LEGEND:  A - alpha character byte

D---->                   V - variable character byte

D---->                   H - header alpha character byte

 <----A                  D - data byte

 <----A                 CL - checksum low byte

 <----A                 CH - checksum high byte

                    ----> - direction of command flow

 (d)                <---- - direction of acknowledgement flow

                NOTE:  In all diagrams, time progresses down

                       the table
```

Figure 3. Structure classifications. (a) Class 1 structure. (b) Class 2 structure. (c) Class 3 structure. (d) Class 4 structure.

CLASS 5

```
 H---->
 H---->
 H---->
 D---->
 D---->
    .
    .
    .
 D---->
CL---->
CH---->
   <----A
   <----A
   <----A
```

(e)

CLASS 6

```
A---->
A---->
A---->
  <----H
  <----H
  <----H
  <----D
  <----D
      .
      .
      .
  <----D
  <----CL
  <----CH
```

(f)

CLASS 7

```
 H---->
 H---->
 H---->
 D---->
 D---->
    .
    .
    .
 D---->
CL---->
CH---->
   <----A
   <----A
   <----A
 D---->
 D---->
    .
    .
    .
 D---->
CL---->
CH---->
   <----A
   <----A
   <----A
 D---->
 D---->
    .
    .
    .
 D---->
CL---->
CH---->
   <----A
   <----A
   <----A
```

(g)

Figure 3. Structure classifications (cont.). (e) Class 5 structure. (f) Class 6 structure. (g) Class 7 structure.

	VRC									VRC							
BYTE 1	0	1	0	0	0	0	0	1		0	1	0	0	0	0	0	1
BYTE 2	1	1	0	1	0	1	0	0		1	1	X	X	X	X	X	0
BYTE 3	0	0	0	1	0	1	0	0		0	0	0	1	0	1	0	0
BYTE 4	1	0	0	1	1	0	0	1		1	0	0	1	1	0	0	1
BYTE 5	1	0	0	1	1	1	0	0		1	0	0	1	1	1	0	0
BYTE 6	1	1	0	1	0	1	1	1		1	1	0	1	0	1	1	1
BYTE 7	1	1	0	0	1	0	1	0		1	1	0	0	1	0	1	0
BYTE 8	0	0	1	0	0	0	1	0		0	0	1	0	0	0	1	0
LRC	1	0	1	1	1	0	1	1		1	0	1	1	1	0	1	1

(a) (b)

	VRC																
BYTE 1	0	1	0	0	0	0	0	1		1	0	0	0	0	0	1
BYTE 2	0	1	1	0	1	0	1	0	◄	1	0	1	0	1	0	0
BYTE 3	0	0	0	1	0	1	0	0		0	0	1	0	1	0	0
BYTE 4	1	0	0	1	1	0	0	1		0	0	1	1	0	0	1
BYTE 5	1	0	0	1	1	1	0	0		0	0	1	1	1	0	0
BYTE 6	1	1	0	1	0	1	1	1		1	0	1	0	1	1	1
BYTE 7	1	1	0	0	1	0	1	0		1	0	0	1	0	1	0
BYTE 8	0	0	1	0	0	0	1	0		0	1	0	0	0	1	0
LRC	1	0	0	0	0	1	0	1								
			▲	▲	▲	▲	▲									

(c) (d)

Figure 4. VRC/LRC example. (a) Transmitted data block with transmitted VRC/LRC map. (b) Received data block with transmitted VRC/LRC map. Erroneous bits are marked by an X. (c) Received data block with generated VRC/LRC map. Arrows indicate where checkbits differ from Figure 4b. Erroneous bits occur at arrow intersections. (d) Corrected data block.

VRC								
1	1	0	1	0	1	0	0	
0	0	0	1	0	1	1	1	
0	0	0	1	1	0	0	0	
1	0	1	1	0	0	0	1	
1	1	0	1	0	0	0	1	
0	0	1	0	1	0	0	0	
1	1	0	0	0	1	1	0	
0	1	0	1	0	1	0	1	LRC

(a)

VRC								
1	1	0	1	0	1	0	0	
0	0	0	1	0	1	1	1	
0	0	0	X	1	X	0	0	
1	0	1	1	0	0	0	1	
1	1	0	X	0	X	0	1	
0	0	1	0	1	0	0	0	
1	1	0	0	0	1	1	0	
0	1	0	1	0	1	0	1	LRC

(b)

Figure 5. VRC/LRC failure. (a) Correct data block and corresponding VRC/LRC map. (b) Incorrect data block with the same VRC/LRC map. Erroneous bits are marked by an X.

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	CMD	----->	CMD	One (Module One) sends a command. Two (Module Two) receives it.
2	XXX	<--/--	ACK	Two sends an acknowledgement. Transmission is garbled.
3	RR	--/-->	XXX	One sends a retransmission request. Transmission is garbled.
4	XXX	<--/--	RR	Two sends a retransmission request. Transmission is garbled.
5	RR	----->	RR	One sends a retransmission request. Two receives it.
6			?	Two should send the acknowledgement again.

(a)

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	CMD	----->	CMD	One sends a command. Two receives it.
2	ACK	<-----	ACK	Two sends an acknowledgement. One receives it.
3	CMD	--/-->	XXX	One sends a new command. Transmission is garbled.
4	XXX	<--/--	RR	Two sends a retransmission request. Transmission is garbled.
5	RR	----->	RR	One sends a retransmission request. Two receives it.
6			?	Two should send a retransmission request again.

(b)

Figure 6. Dual retransmission request problem. (a) Lost acknowledgement. (b) Lost command.

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	0CMD	----->	0CMD	One (Module One) sends a command with 0 sequence number. Two (Module Two) receives it and changes its own sequence number.
2	XXX	<--/--	0ACK	Two sends an acknowledgement with 0 sequence number. Transmission is garbled.
3	0RR	--/-->	XXX	One sends a retransmission request with 0 sequence number. Transmission is garbled.
4	XXX	<--/--	0RR	Two sends a retransmission request with 0 sequence number. Transmission is garbled.
5	0RR	----->	0RR	One sends a retransmission request with 0 sequence number. Two receives it.
6	0ACK	<-----	0ACK	Two sends previous acknowledgement with 0 sequence number. One receives it and changes its own sequence number.
7	1CMD	----->	1CMD	One sends a new command with 1 sequence number. Two receives it and changes its own sequence number. Communication continues.

(a)

Figure 7. Dual retransmission request solution. (a) Lost acknowledgement.

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	0CMD	----->	0CMD	One (Module One) sends a command with 0 sequence number. Two (Module Two) receives it and changes its own sequence number.
2	0ACK	<-----	0ACK	Two sends an acknowledgement with 0 sequence number. One receives it and changes its own sequence number.
3	1CMD	--/-->	XXX	One sends a new command with 1 sequence number. Transmission is garbled.
4	XXX	<--/--	0RR	Two sends a retransmission request with 0 sequence number. Transmission is garbled.
5	1RR	----->	1RR	One sends a retransmission request with 1 sequence number. Two receives it.
6	0RR	<-----	0RR	Two sends a retransmission request with 0 sequence number. One receives it.
7	1CMD	----->	1CMD	One sends previous command with 1 sequence number. Two receives it and changes its own sequence number. Communication continues.

(b)

Figure 7. Dual retransmission request solution (cont.). (b) Lost command.

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	0CMD	--/-->	XXX	One (Module One) sends a command with 0 sequence number. Transmission is garbled.
2	0RR	<--/--	1RR	Two (Module Two) sends a retransmission request with 1 sequence number. Transmission changes the sequence number
3	0RR	----->	0RR	One sends a retransmission request with 0 sequence number. Two receives it.
4	1RR	<-----	1RR	Two sends a retransmission request with 1 sequence number. One receives it.
5	0CMD	----->	0CMD	One sends previous command with 0 sequence number. Two receives it and changes its own sequence number. Communication continues.

(a)

Figure 8. Various retransmission problems with sequence number control solutions. (a) Retransmission request with wrong sequence number.

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	0CMD	----->	0CMS	One (Module One) sends a command with 0 sequence number. Two (Module Two) receives it and changes its own sequence number.
2	1ACK	<--/--	0ACK	Two sends an acknowledgement with 0 sequence number. Transmission changes the sequence number.
3	0RR	----->	0RR	One sends a retransmission request with 0 sequence number. Two receives it.
4	0ACK	<-----	0ACK	Two sends previous acknowledgement with 0 sequence number. One receives it and changes its own sequence number. Communication continues.

(b)

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	0CMD	--/-->	1CMD	One sends a command with 0 sequence number. Transmission changes the sequence number.
2	1RR	<-----	1RR	Two sends a retransmission request with 1 sequence number. One receives it.
3	0CMD	----->	0CMD	One sends previous command with 0 sequence number. Two receives it and changes its own sequence number. Communication continues.

(c)

Figure 8. Various retransmission problems with sequence number control solutions (cont.). (b) Acknowledgement with wrong sequence number. (c) Command with wrong sequence number.

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	0CMD	--/-->	0CMD	One (Module One) sends a command with 0 sequence number. Transmission is lost.
2	0CMD	----->	0CMD	Expected Acknowledgement Timer overflows. One sends previous command with 0 sequence number. Two (Module Two) receives it and changes its own sequence number.
3	0ACK	<-----	0ACK	Two sends an acknowledgement with 0 sequence number. One receives it and changes its own sequence number. Communication continues.

(a)

STEP	MODULE ONE		MODULE TWO	COMMENTS
1	0CMD	----->	0CMD	One sends a command with 0 sequence number. Two receives it and changes its own sequence number.
2	0ACK	<--/--	0ACK	Two sends an acknowledgement with 0 sequence number. Transmission is lost.
3	0CMD	----->	0CMD	Expected Acknowledgement Timer overflows. One sends previous command with 0 sequence number. Two receives it and changes its own sequence number.
4	0RR	<-----	0RR	Two sends a retransmission request with 0 sequence number. One receives it.
5	0RR	----->	0RR	One sends a retransmission request with 0 sequence number. Two receives it.
6	0ACK	<-----	0ACK	Two sends previous acknowledgement with 0 sequence number. One receives it and changes its own sequence number. Communication continues.

(b)

Figure 9. Sequence number control for the Expected Acknowledgement Timer. (a) Lost command. (b) Lost acknowledgement.

CENTRAL CONTROL COMPUTER (CCC)		THERMOMETRY SUBSYSTEM (TS)	COMMENTS
"N"	---->	"N"	CCC sends Name/Status command to TS
0B	<----	0B	TS sends "System Not Ready" acknowledgement to CCC
"N"	---->	"N"	CCC sends Name/Status command TS
1B	<----	1B	TS sends "System Ready" acknowledgement to CCC
"N"	---->	"N"	CCC sends Name/Status command to TS
	. . .		TS steps through calibration routine
2B	<----	2B	TS sends "System Calibrated" acknowledgement to CCC
"L"	---->	"L"	CCC sends Load command to TS
"U" block	<----	"U" block	TS sends Unload block to CCC
"I"	---->	"I"	CCC sends Initialize and Go command to TS
"D"	<----	"D"	TS sends Done acknowledgement to CCC
"T"	---->	"T"	CCC sends Temperatures command to TS
"E" block	<----	"E" block	TS sends Receive Temperatures block to CCC
	. . .		Temperature exchanges repeated until end of treatment

Figure 10. Typical error-free transaction sequence for the thermometry subsystem.

CENTRAL CONTROL COMPUTER (CCC)		APPLICATOR SUBSYSTEM (AS)	COMMENTS
"N"	---->	"N"	CCC sends Name/Status command to AS
0C	<----	0C	AS sends "System Ready" acknowledgement to CCC
"F" and data	---->	"F" and data	CCC sends Frequency command to AS
"D"	<----	"D"	AS sends Done reply to CCC
"V" and data	---->	"V" and data	CCC sends Voltage command to AS
"D"	<----	"D"	AS sends Done reply to CCC
"E" block	---->	"E" block	CCC sends Receive Duty Cycles block to AS
"D"	<----	"D"	AS sends Done reply to CCC
"I"	---->	"I"	CCC sends Initialize and Go command to AS
"D"	<----	"D"	AS sends Done reply to CCC
	. . .		
"E" block or "V" and data	---->	"E" block or "V" and data	Duty cycles and voltage exchanges repeated until end of treatment
"D"	<----	"D"	
	. . .		

Figure 11. Typical error-free transaction sequence for the applicator subsystem.

TABLES

Table 1. Data Byte Encoding

BINARY VALUE OF DATA	HEXIDECIMAL VALUE OF CHARACTER SENT	ASCII CHARACTER
0000	30	0
0001	31	1
0010	32	2
0011	33	3
0100	34	4
0101	35	5
0110	36	6
0111	37	7
1000	38	8
1001	39	9
1010	3A	:
1011	3B	;
1100	3C	<
1101	3D	=
1110	3E	>
1111	3F	?

Table 2. Sequence Number Control Rules

Initialization

- Central control computer sequence number = 0
- Intel 8031 sequence number = 1
- Central control computer sends first command with sequence number = 0

MESSAGE RECEIVED	CENTRAL CONTROL COMPUTER ACTION	INTEL 8031 ACTION
Invalid character	-Send Retransmit with current seq. no.	-Send Retransmit with current seq. no.
Valid character with same seq. no.	-Complement sequence number -Send next command with new seq. no.	-Send Retransmit with seq. no.
Valid character with different seq. no.	-Send Retransmit with current seq. no.	-Complement sequence number -Send next acknowledgement with new seq. no.
Retransmit with same seq. no.	-Send Retransmit with current seq. no.	-Send previous acknowledgement with new seq. no.
Retransmit with different seq. no.	-Send previous command with current seq. no.	-Send Retransmit with current seq. no.

Table 3. Command-Acknowledgement Structures

SUBSYSTEM	STRUCTURE	MNEMONICS	CLASS
Both	Name/Status-Identification/ Status	"N"	3
Both	Initialize and Go-Done	"I","D"	2
Both	Retransmit	"R"	2,3,4, 5,6
Both	Shutdown	"S"	1
Thermometry	Load-Unload	"L","U"	6
Thermometry	Temperatures-Receive Temperatures	"T","E"	6
Applicator	Frequency-Done	"F","D"	4
Applicator	Voltage-Done	"V","D"	4
Applicator	Receive Duty Cycles-Done	"E","D"	5
Applicator	Wait-Done	"W","D"	2
Applicator	Help	"H"	1 or 2

APPENDIX A

INTEL 8031 MICROCONTROLLER

Intel's 8031 microcontroller is a control-oriented single chip computer intended for use in real-time applications such as intelligent computer peripherals. It can be used as a stand-alone job-specific processor or circuitry can be added to yield a central processing unit (CPU) capable of supporting numerous tasks. The latter is the case in this hyperthermia system. Both the CPU board and the Intel 8031 microcontroller possess many interesting features, but only the more important points pertaining to serial communication functions are discussed here. If additional information is desired, please consult Intel's Microcontroller User's Manual listed in the references.

A.1. Timing

The Intel 8031 provides two 16-bit registers, Timer 0 and Timer 1, that can be used as either timers or event counters. For all applications in this system, both registers are used as timers, where time is kept by counting machine cycles which are equal to one-twelfth the oscillator frequency. Two special registers, Timer Control Register (TCOM) and Timer Mode Control Register (TMOD), are used to define the operating modes and control the functions of the timers. The TCOM allows the user to turn either timer on or off and contains several bits that are set and cleared by the hardware. The TMOD allows the user to configure each of the timers in one of four possible operating modes. Modes 0 and 3 are never used in this application and need not be discussed.

Mode 1 configures the timer as a 16-bit counter where overflow conditions return the counter to the zero state. Mode 2 configures the timer as a 8-bit counter with automatic reload. This second mode is important because it allows the user to generate a number of commonly used baud rates derived from the 11.059 MHz crystal oscillator located on the CPU board. By changing the value stored in the reload register or by altering the contents of another special register, PCON, baud rates ranging from 300 bits per second up to 19,200 bits per second can be obtained.

A.2. Serial Interface

The serial port is full-duplex and is also receive-buffered, so it can commence reception of a second byte before a previously received byte has been read from the receive register. Both transmitted and received data are passed through special register SBUF, where a write to SBUF loads the transmit register and a read from SBUF accesses a physically separate receive register. Like the timers, the serial port can operate in one of four modes, three of which are mentioned here. If mode 1 is used, 10 bits are transmitted, consisting of a start bit (0), eight data bits (LSB first), and a stop bit (1). During reception, the stop bit goes into RB8, a single bit in SCON, the Serial Port Control Register. The baud rate in mode 1 is variable. In modes 2 and 3, 11 bits are transmitted consisting of a start bit (0), eight data bits (LSB first), a programmable ninth data bit, and a stop bit (1). The ninth data bit must be stored in TB8, also in SCON, prior to transmission. In these modes, RB8 is filled with the ninth data bit and the stop bit is ignored. The only difference between modes 2 and 3 is the baud rate: mode 3 allows a variable baud rate, whereas mode 2's baud rate is programmable to either 1/32 or 1/64 the oscillator frequency. The aforementioned Serial Port Control Register, besides holding TB8 and RB8, also defines the operating mode and controls certain functions of the serial port. One bit, REN, enables reception and is both set and cleared by software. Two other bits, TI, the transmit interrupt flag which signals the end of a transmission, and RI, the receive interrupt flag which signals the end of a reception, are set by hardware, but must be cleared by software.

A.3. Interrupts

The Intel 8031 provides five separate interrupt sources: two external requests, the two timer overflows, and the serial port interrupt. Although both timers are used in this application, only Timer 0 is allowed to interrupt. Timer 1 is used exclusively as a baud rate generator and the overflows are of no programming interest. The interrupt handler for Timer 0 contains several software timers which were mentioned in the main discussion. The serial port interrupt service routine is of greatest interest because it is entered for both reception and transmission interrupts and contains most of the communication code.

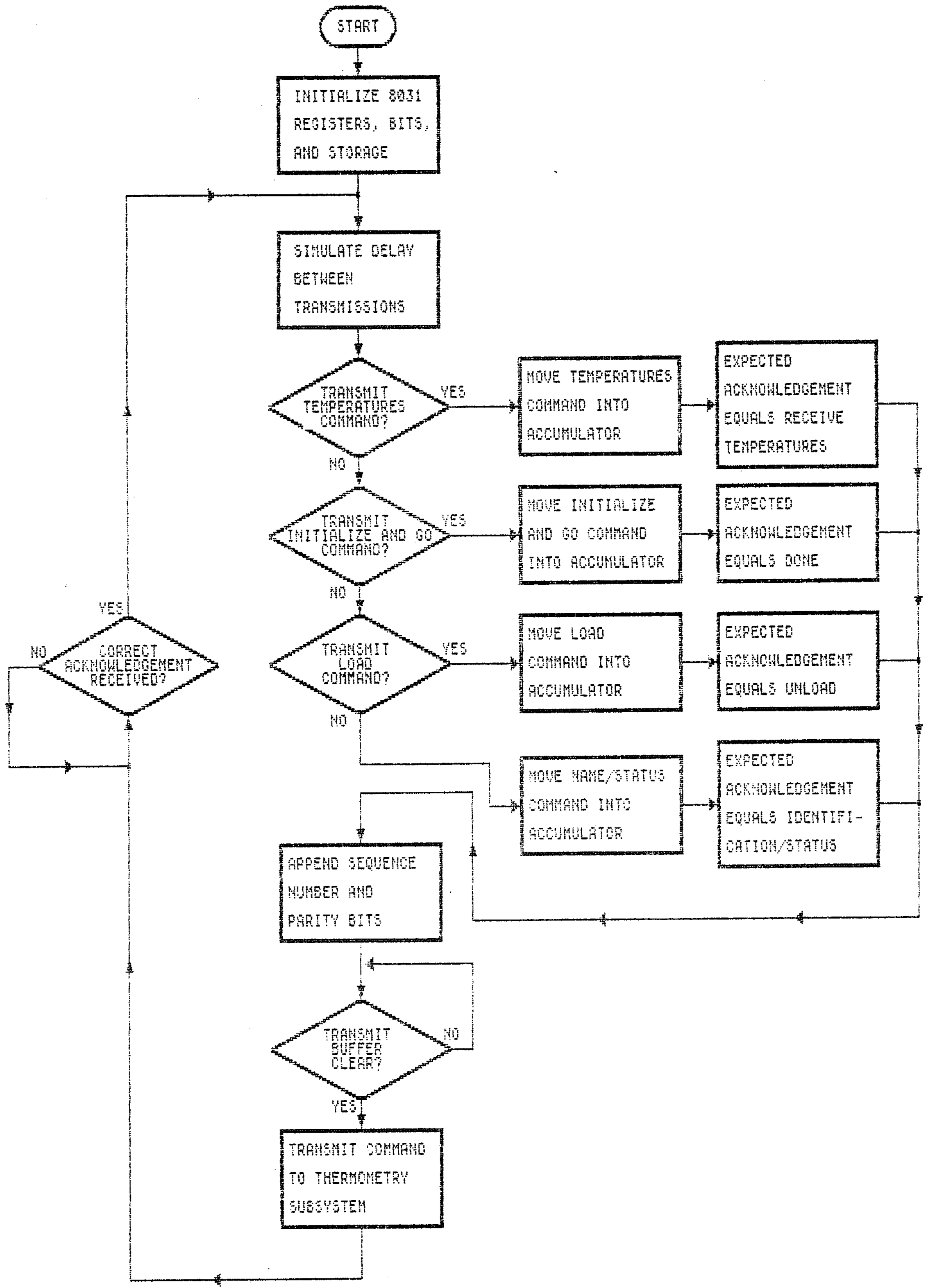
The Interrupt Enable Register (IE) contains six software-controllable bits, one for each of the five sources plus a universal enable bit. The Interrupt Priority Register (IP) contains five software-controllable bits that define the priority of each interrupt to one of two priority levels. An interrupt of one priority level cannot be serviced if an interrupt of equal or higher priority is already in progress. In the event that two sources of the same priority level interrupt simultaneously, the Intel 8031 uses an internal polling sequence to determine which interrupt gets serviced first. One note of interest is the fact that raising the priority level of the interrupt currently being serviced allows that routine to interrupt itself. This feature serves to ease some of the time constraints of real time, interrupt-driven applications.

APPENDIX B

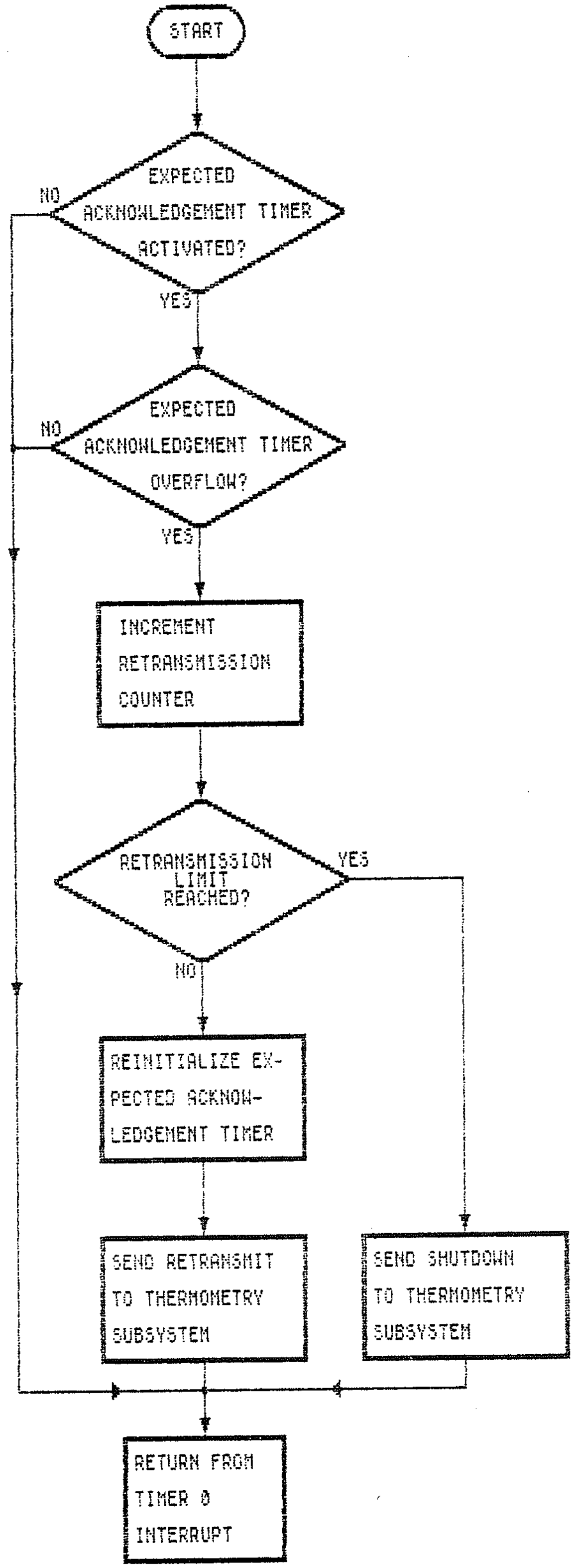
CENTRAL CONTROL COMPUTER PROGRAM FLOWCHART

This appendix contains the flowchart for the central control computer program entitled DECD.ASM. The chart is broken into three sections corresponding to the main routine, INIT, the Timer 0 interrupt handler, TIMERS, and the serial port interrupt handler, SERIAL. Appendix E contains the actual code.

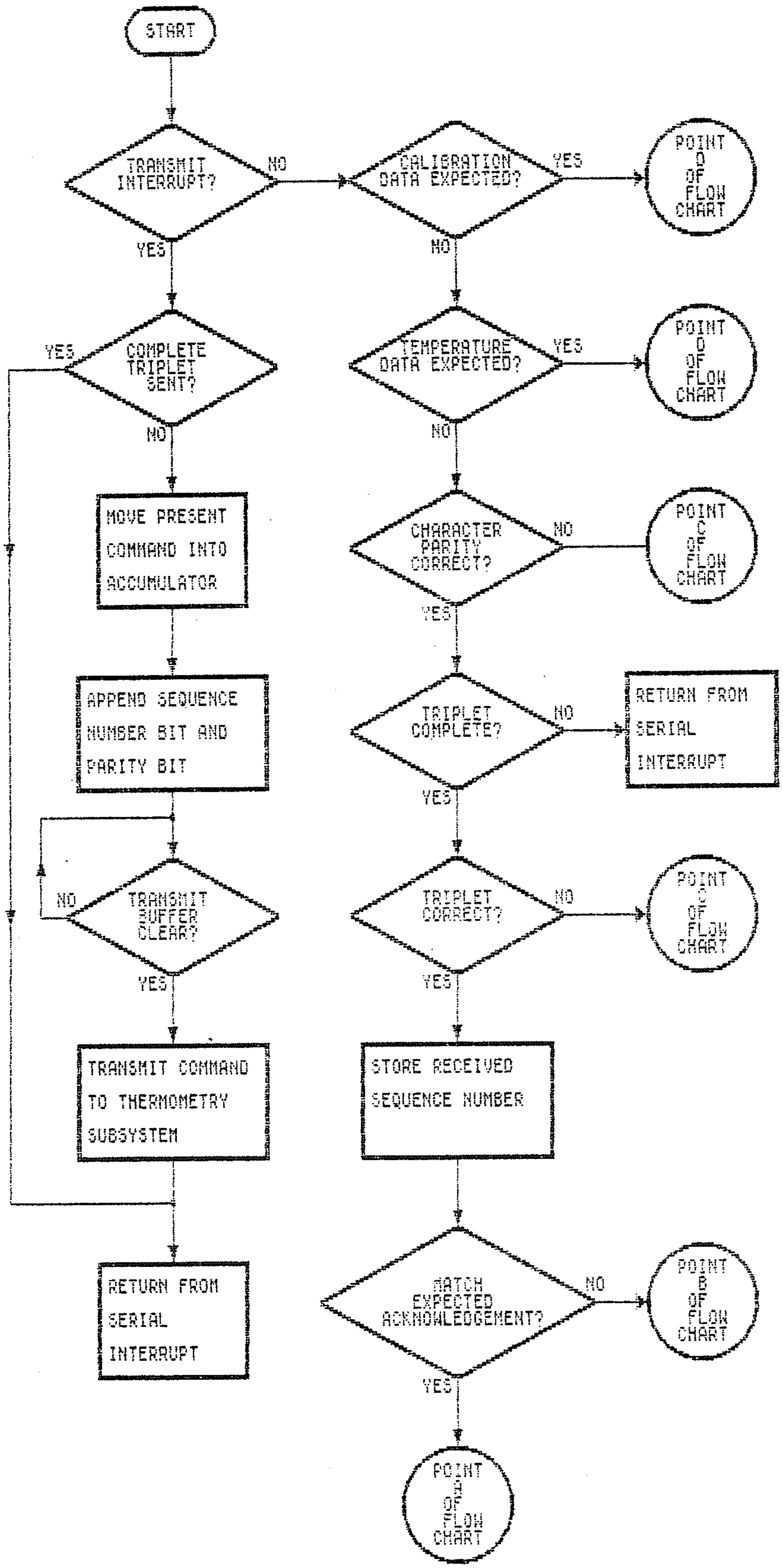
FLOWCHART FOR DECO - INIT



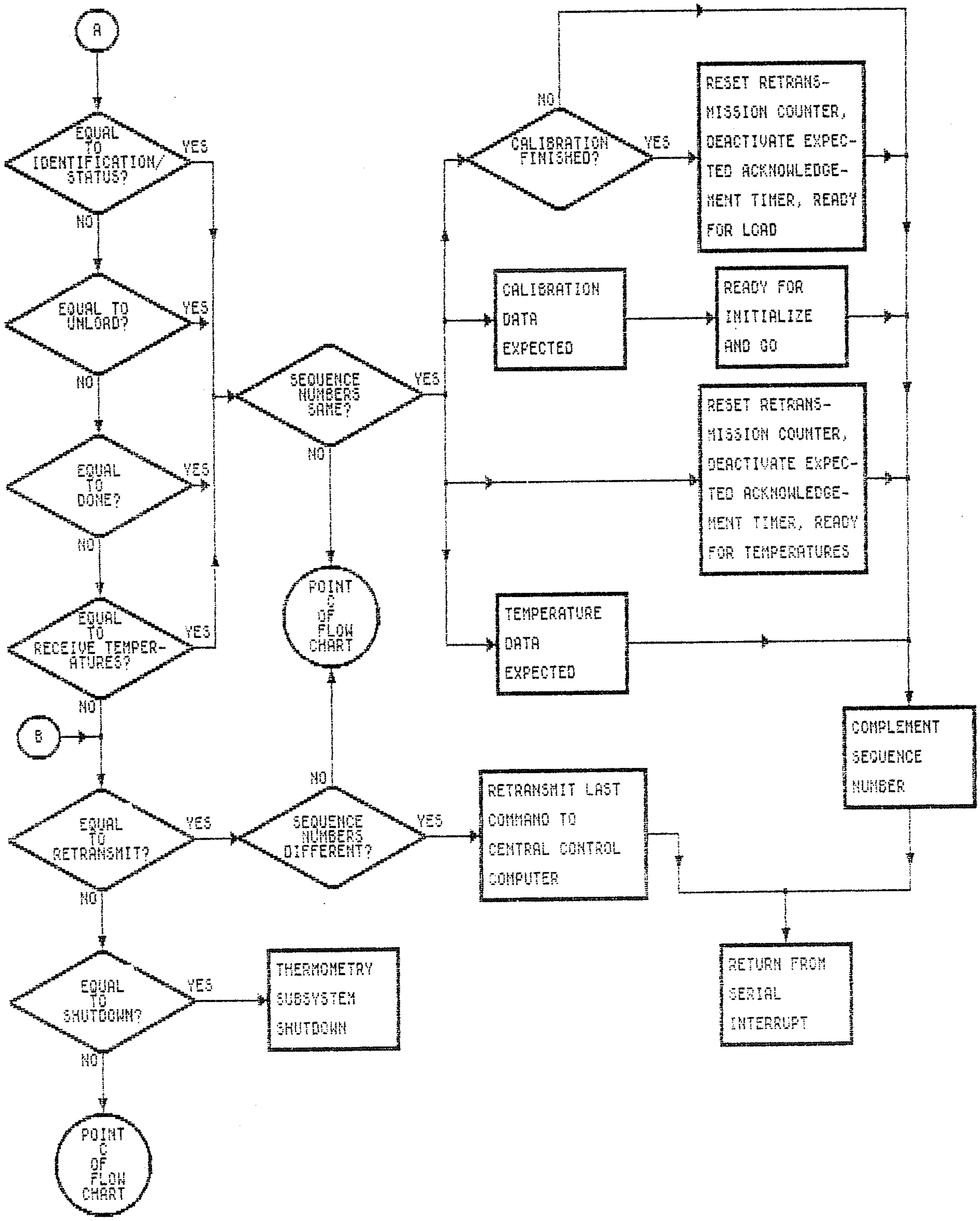
FLOWCHART FOR DECD - TIMERS



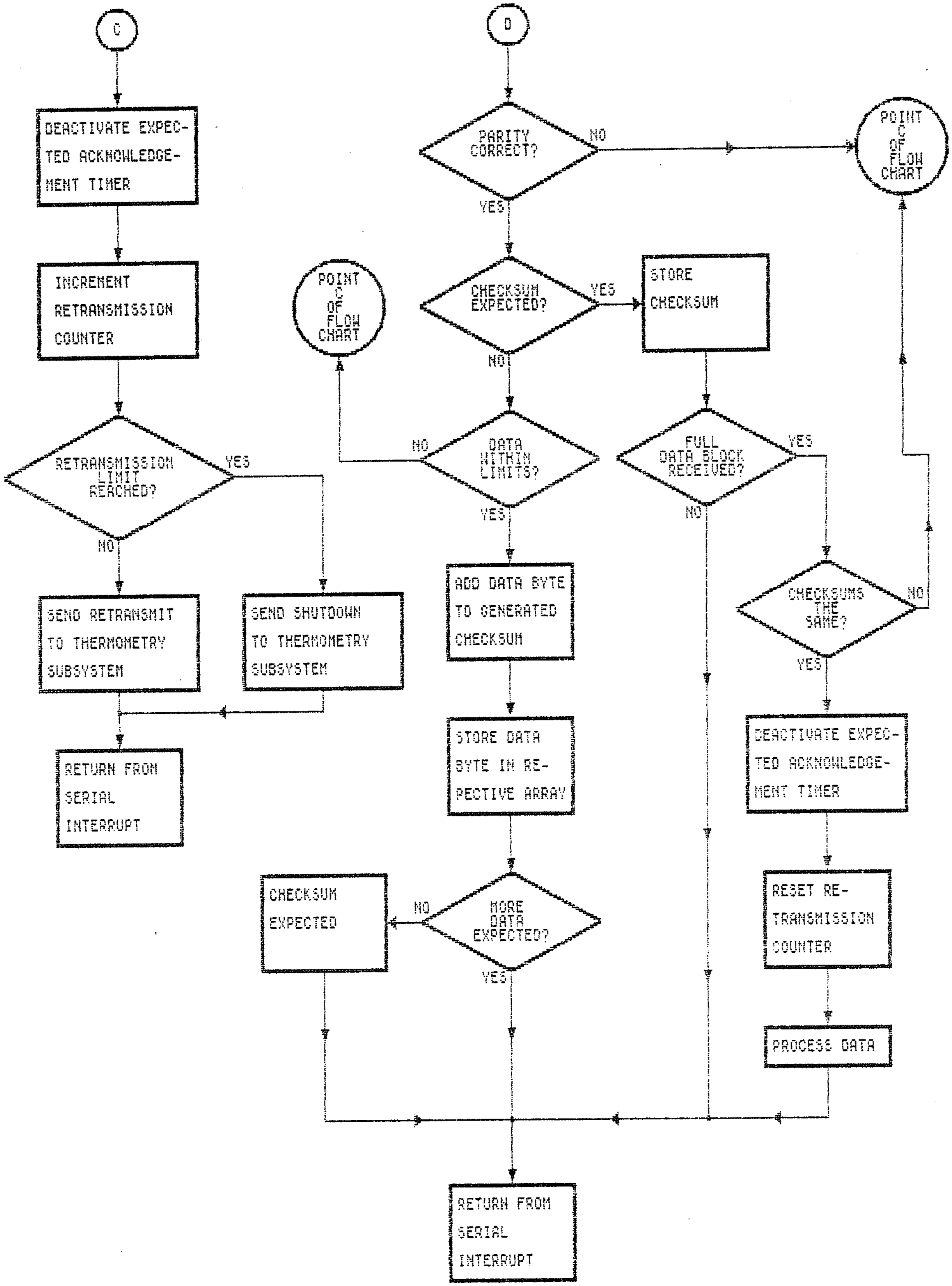
FLOWCHART FOR DECD - SERIAL



FLOWCHART FOR DECO - SERIAL (CONT.)



FLOWCHART FOR DECD - SERIAL (CONT.)

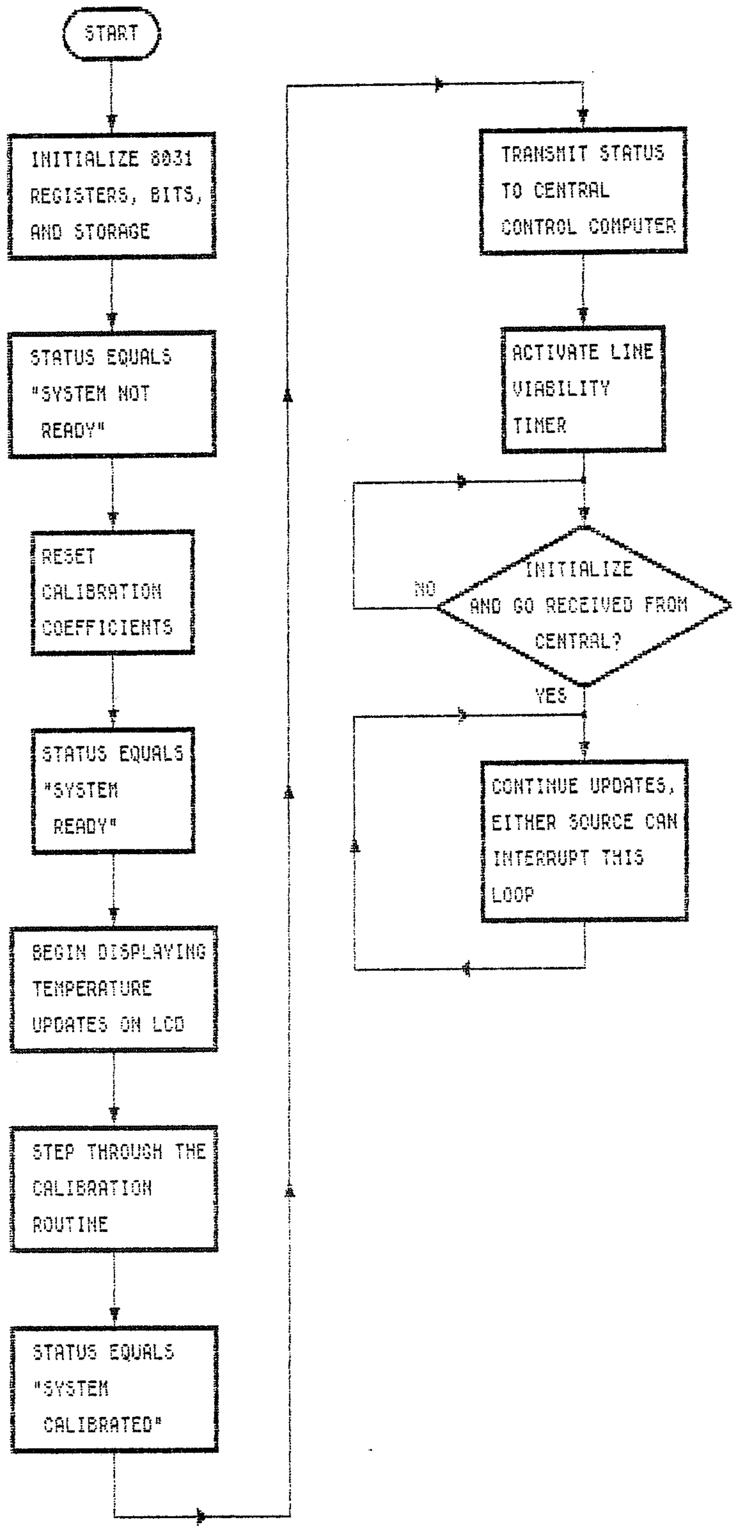


APPENDIX C

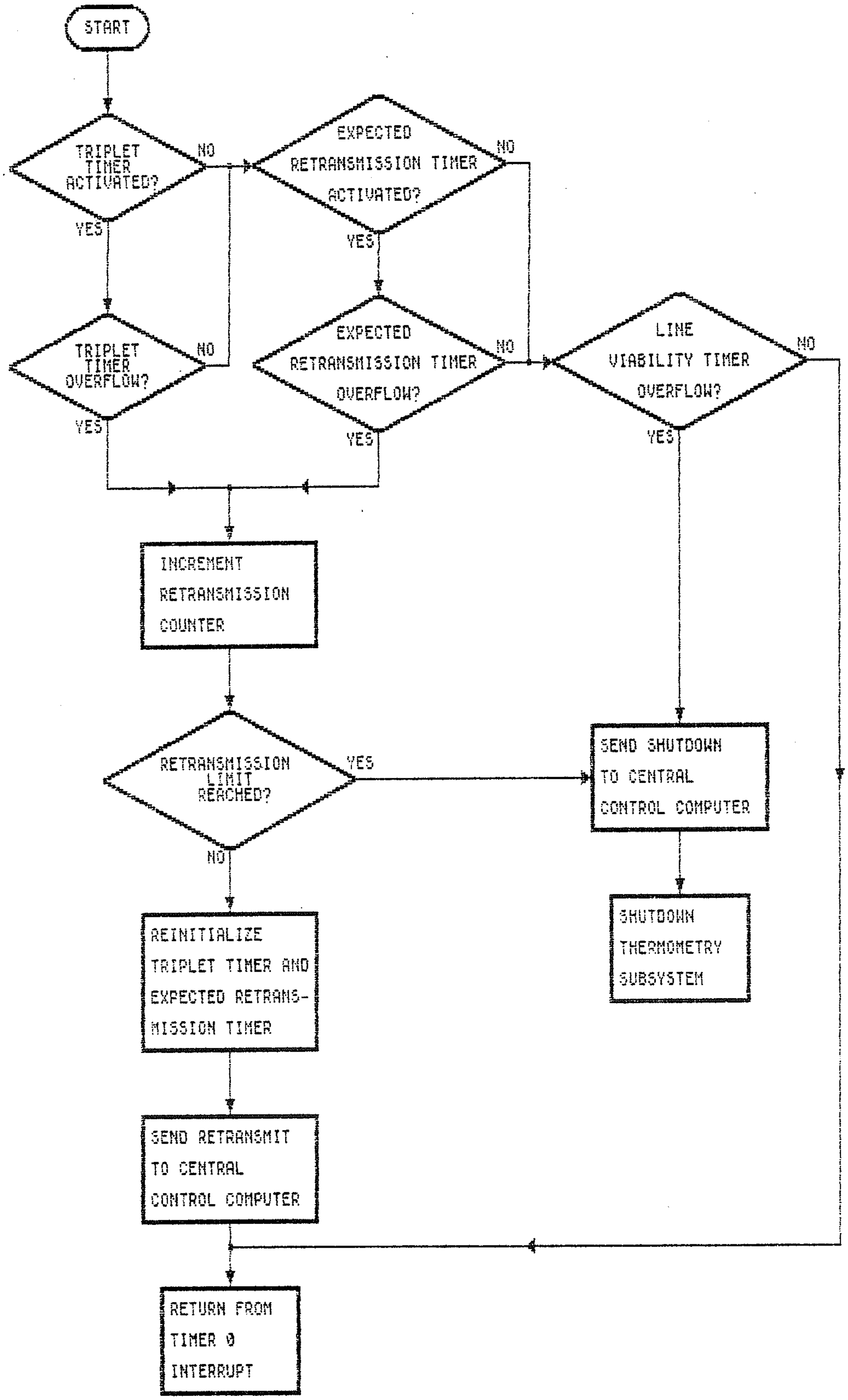
THERMOMETRY SUBSYSTEM PROGRAM FLOWCHART

This appendix contains the flowchart for the thermometry subsystem program entitled TX100D.ASM. The chart is broken into three sections corresponding to the main routine, INIT, the Timer 0 interrupt handler, TIMERS, and the serial port interrupt handler, SERIAL. Appendix F Contains the actual code.

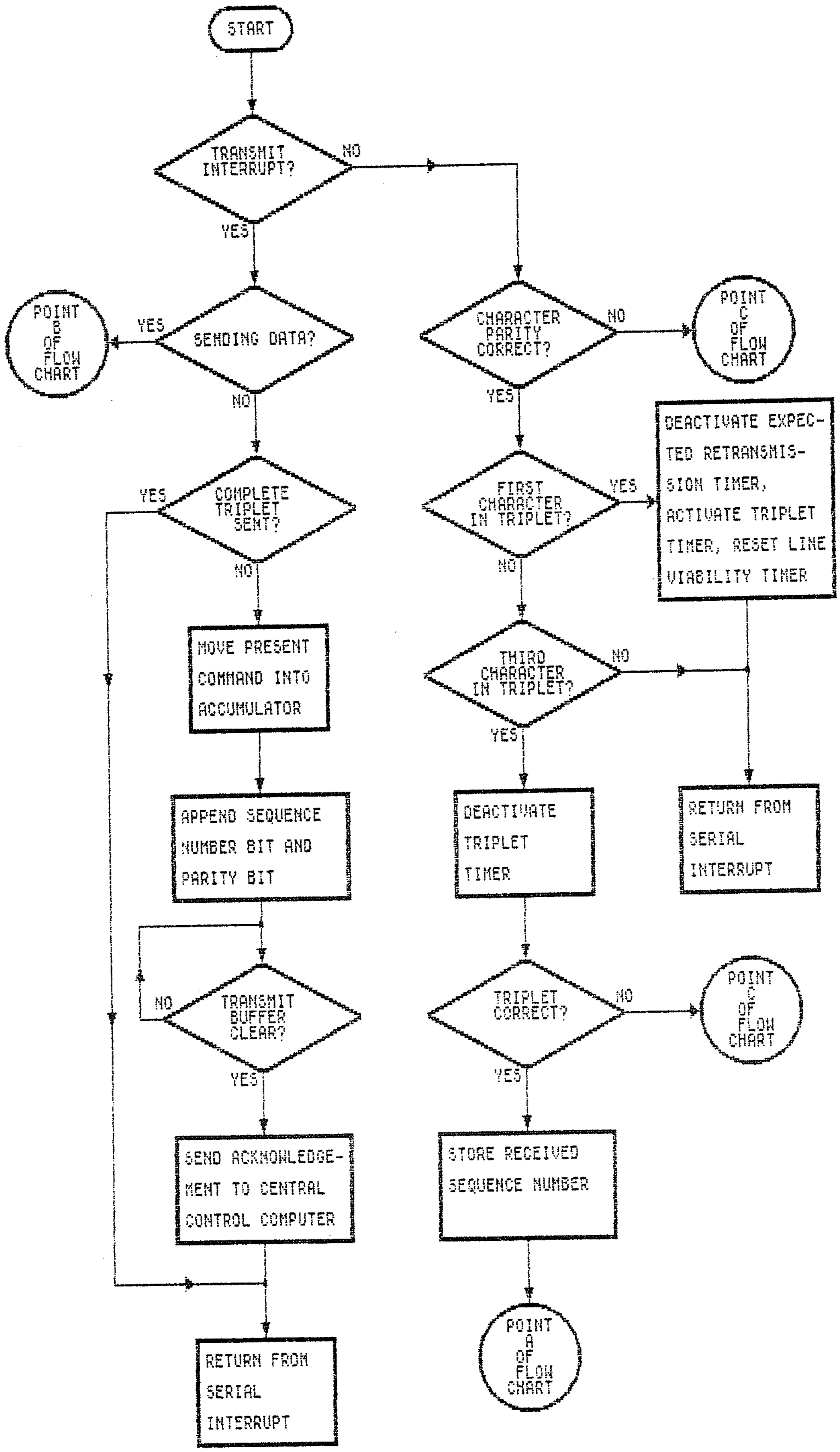
FLOWCHART FOR TX1000 - INIT



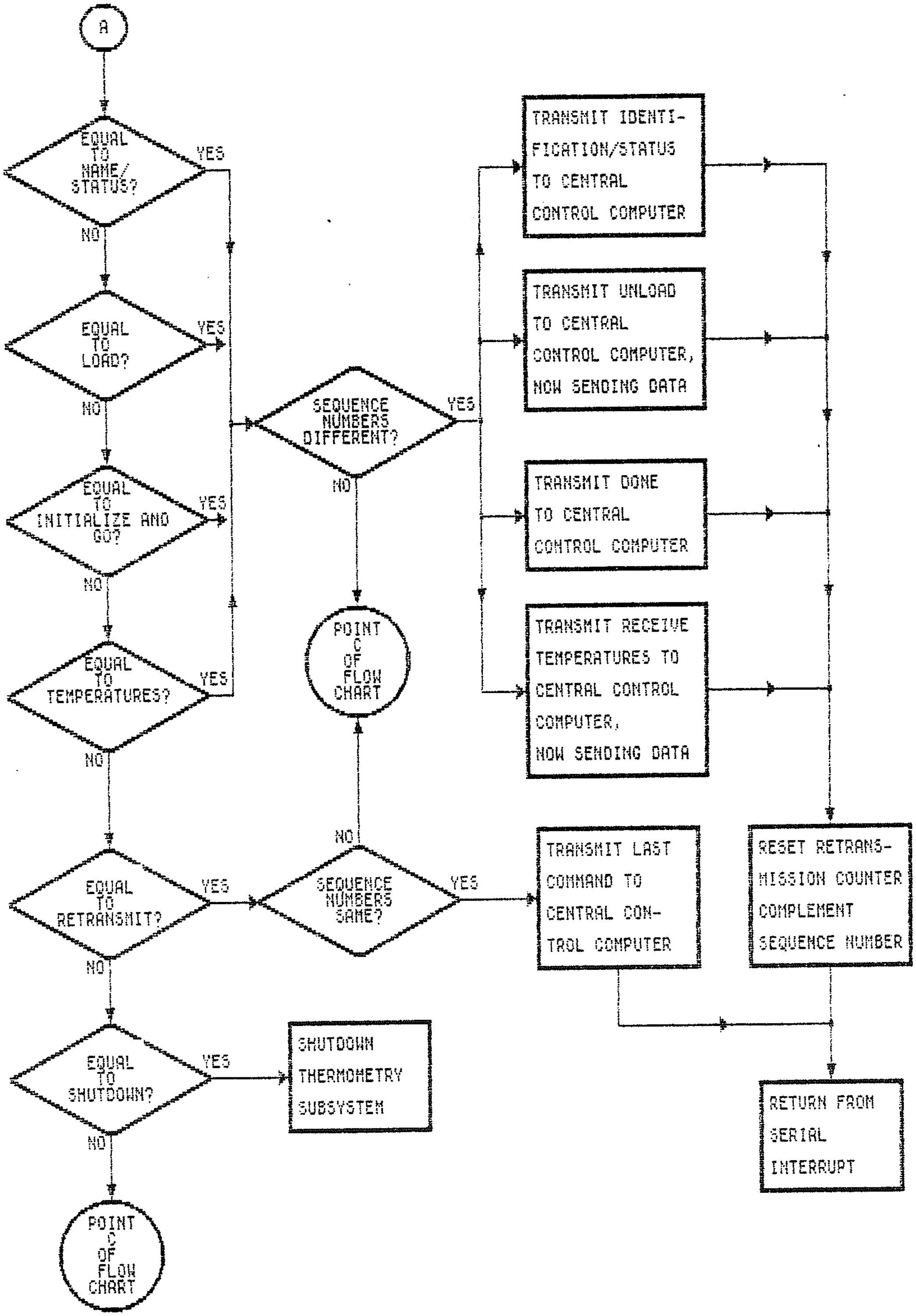
FLOWCHART FOR TX100D - TIMERS



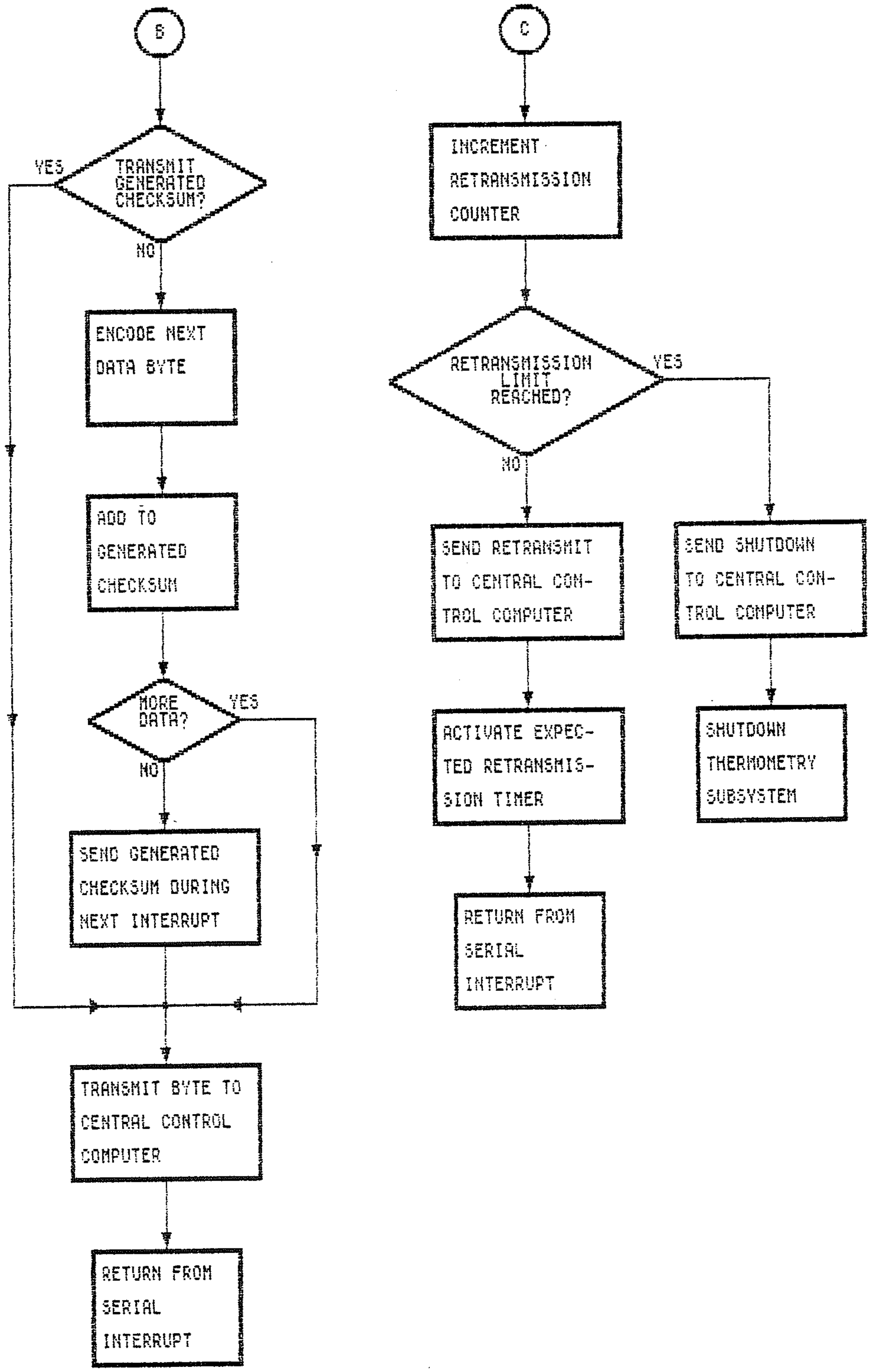
FLOWCHART FOR TX100D - SERIAL



FLOWCHART FOR TX1000 - SERIAL (CONT.)



FLOWCHART FOR TX100D - SERIAL (CONT.)

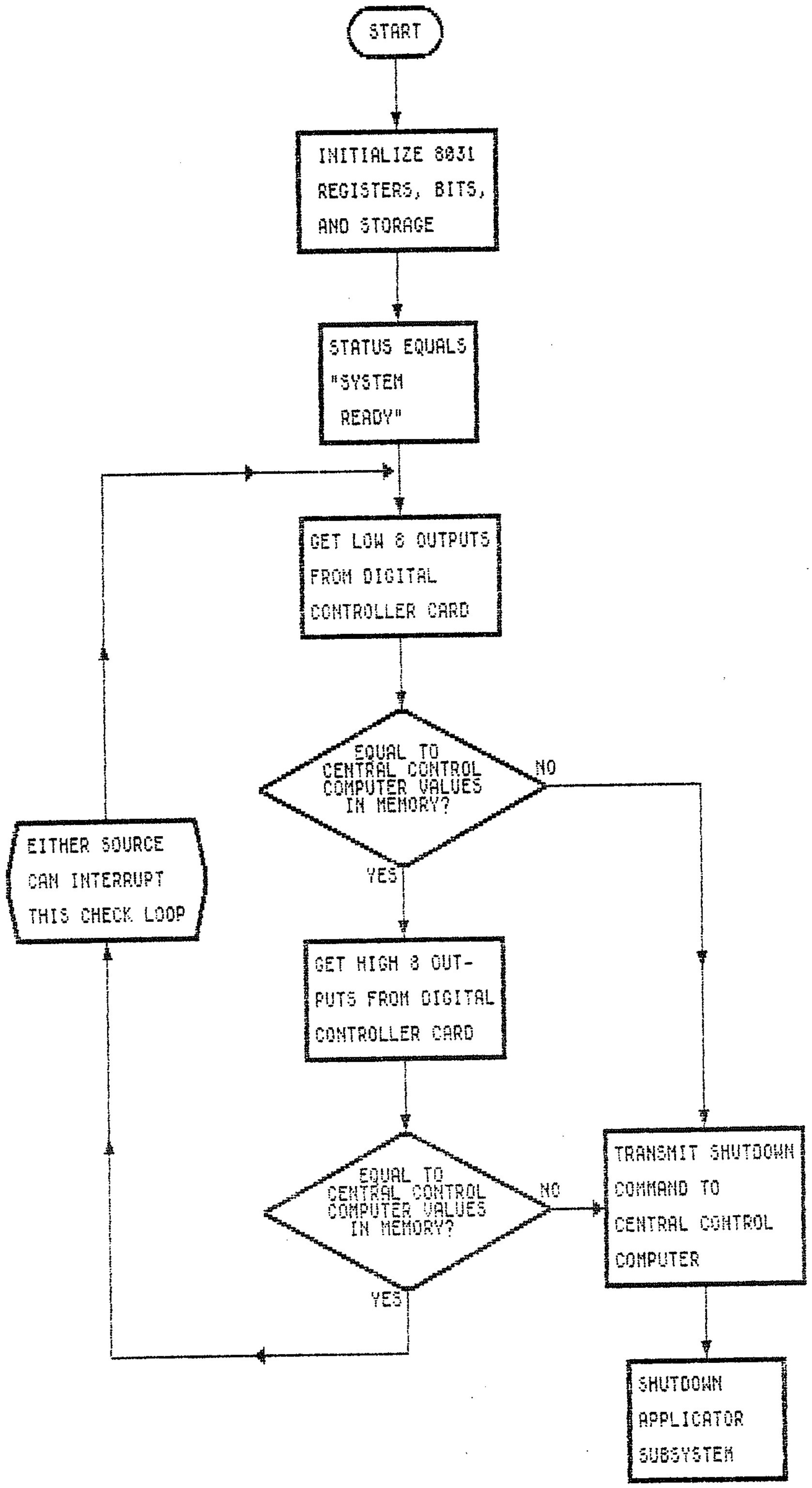


APPENDIX D

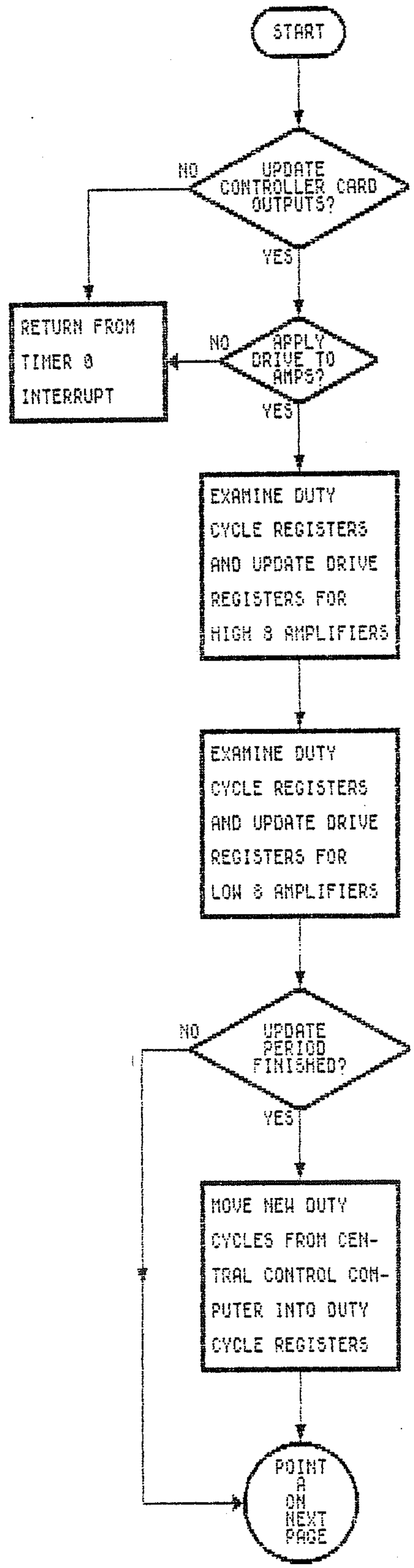
APPLICATOR SUBSYSTEM PROGRAM FLOWCHART

This appendix contains the flowchart for the applicator subsystem program entitled US100.ASM. The chart is broken into three sections corresponding to the main routine, INIT, the Timer 0 interrupt handler, DUTY, and the serial port interrupt handler, SERIAL. Appendix G contains the actual code.

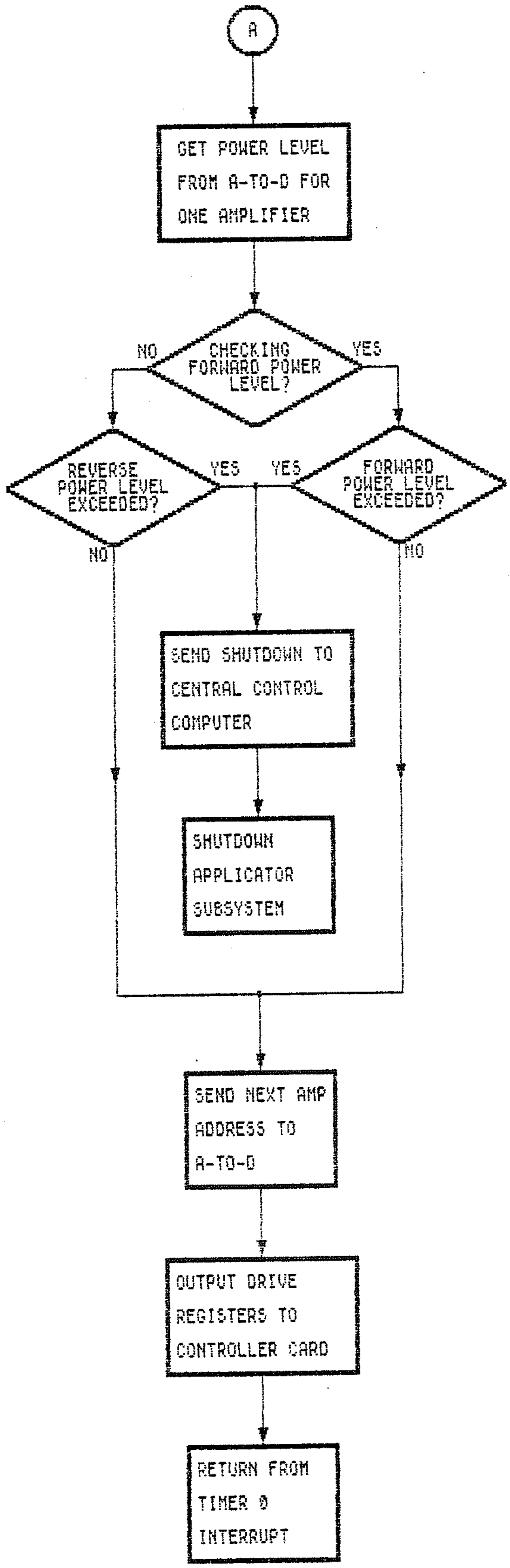
FLOWCHART FOR US100 - INIT



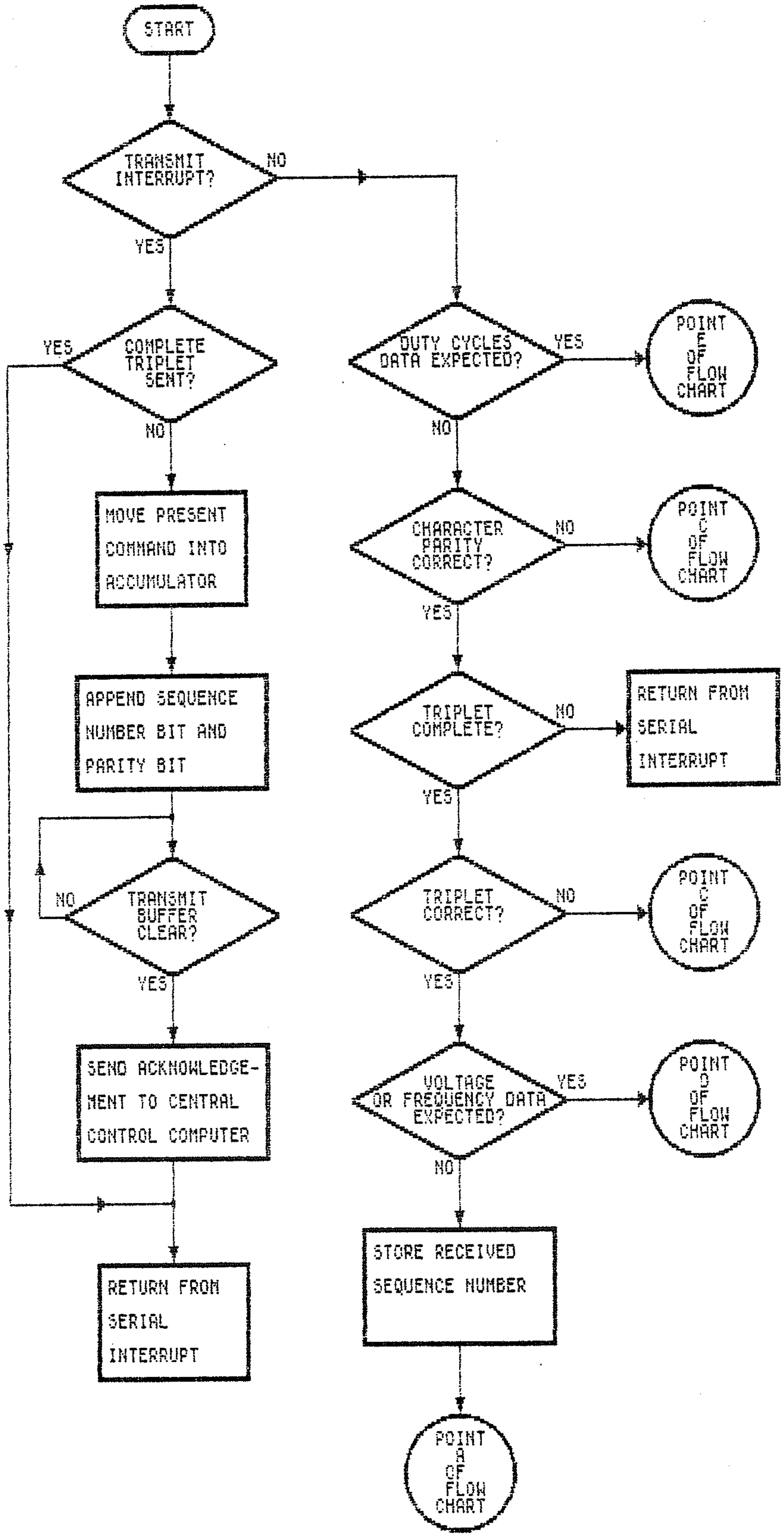
FLOWCHART FOR US100 - DUTY



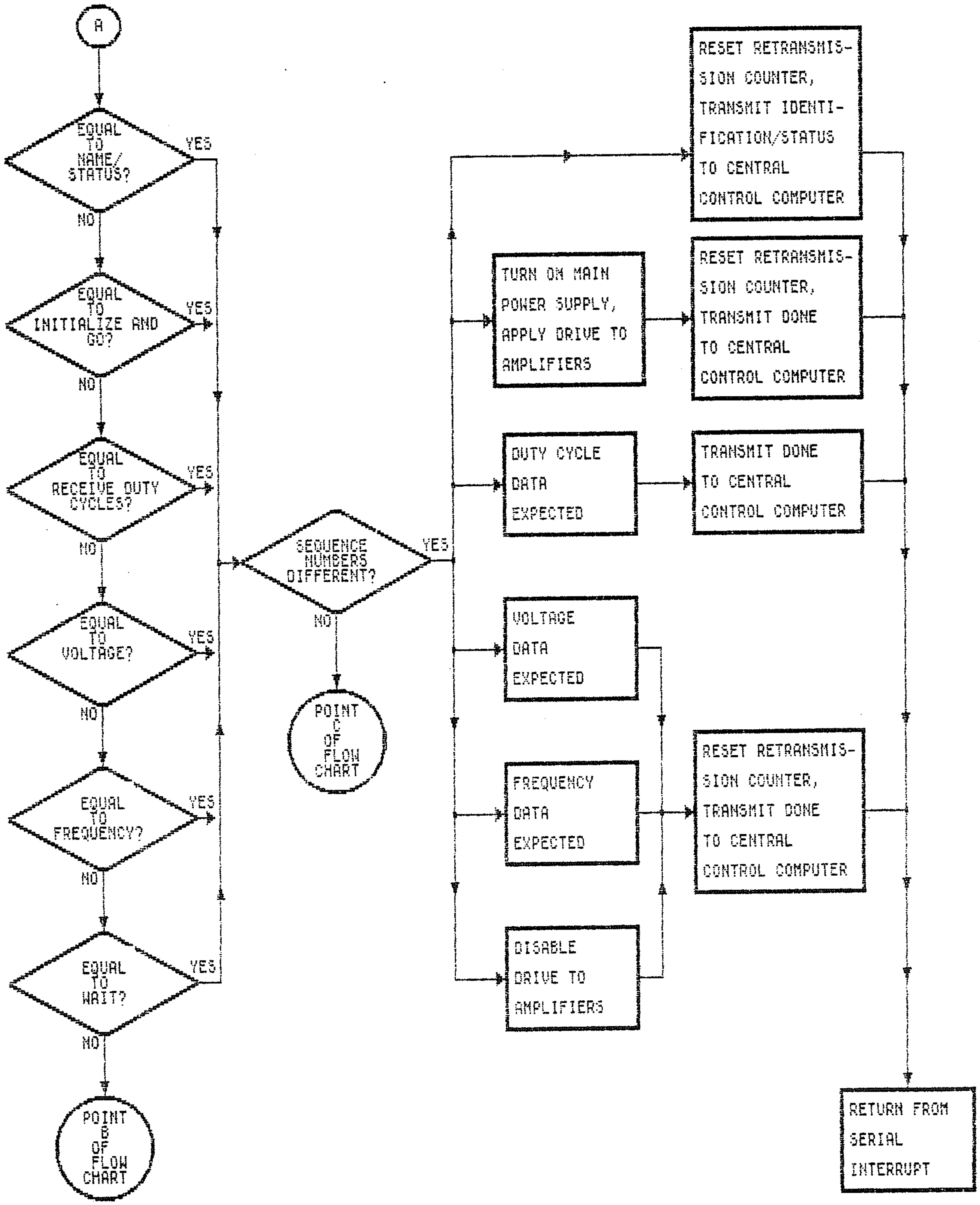
FLOWCHART FOR US100 - DUTY (CONT.)



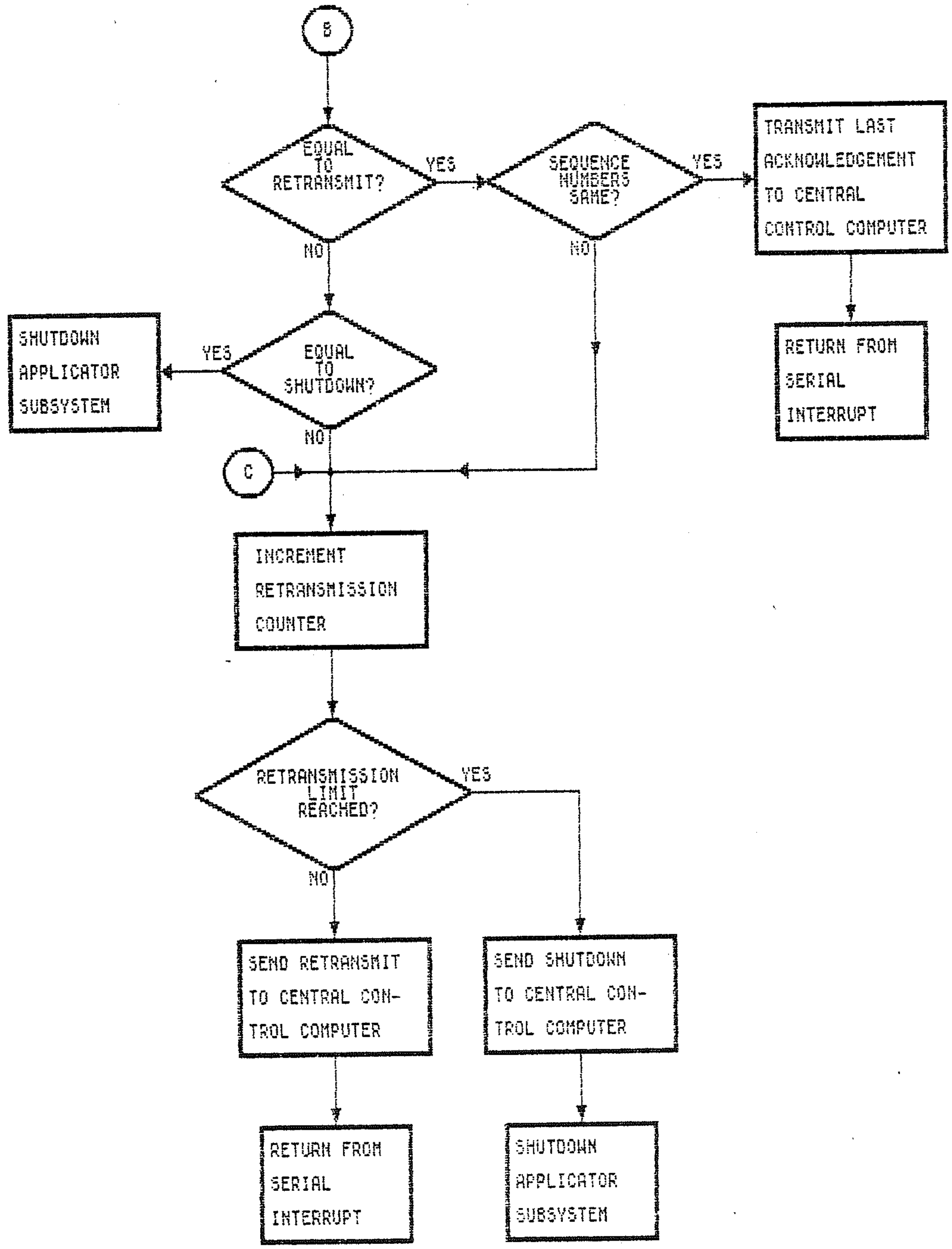
FLOWCHART FOR US100 - SERIAL



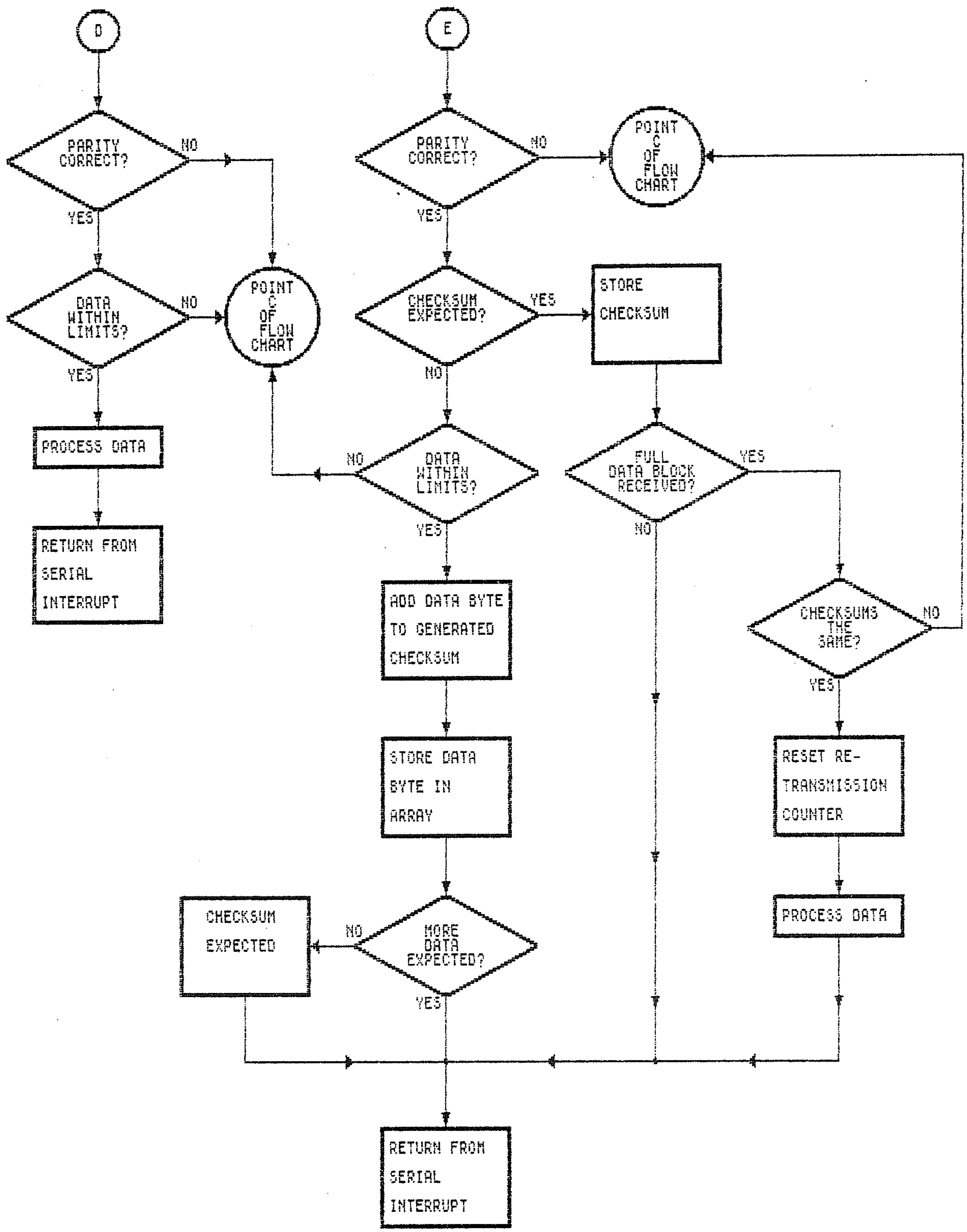
FLOWCHART FOR US100 - SERIAL (CONT.)



FLOWCHART FOR US100 - SERIAL (CONT.)



FLOWCHART FOR US100 - SERIAL (CONT.)



APPENDIX E

CENTRAL CONTROL COMPUTER PROGRAM LISTING

The programs contained in this appendix and the two following appendices were all written for the Intel 8031 microcontroller. A Microsoft Softcard System installed in an Apple IIe computer runs the CP/M-80 operating system and supports the XASM51 8051 CROSS-ASSEMBLER (Version 1.09) develped by Avocet Systems, Inc. This cross-assembler accepts a CP/M-80 source code file and creates an Intel Hex format output file. Commands in the operating system then convert this output file into pure binary object code.

The cross-assembler requires the object code origin to be placed at or above the address 100 hexidecimal. However, all of the Intel 8031 interrupt vectors jump to addresses below 100 hexidecimal. Specifically, a hardware reset, a Timer 0 interrupt, and a serial port interrupt cause jumps to the hexidecimal addresses 0000, 000B, and 0023, respectively. To remedy this situation, several bytes of binary code representing Intel 8031 jump commands are placed at these locations by utilizing another CP/M-80 function. The resulting instructions cause program jumps to the respective service routines, all of which are located safely above the address 100 hexidecimal.

This particular appendix contains the source code listing for the central control computer program entitled DECD.ASM. Brief descriptions of the variable names encountered in the code are presented in the program preface.

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
;
;=============================================================
;DEC VERSION D
;
;--THIS PROGRAM IS DESIGNED TO INTERACT WITH
;  THE THERMOMETRY SYSTEM PROGRAM ENTITLED
;  TX100D.ASM.
;
;--DESCRIPTIONS OF THE VARIABLES USED IN THIS
;  PROGRAM ARE PRESENTED BELOW.
;
;=============================================================
;
;=============================================================
;DESCRIPTION OF BYTE VARIABLES
;
;  OUTREG - STORES THE CONSTANT, CMDCNT, WHICH
;  IS DECREMENTED WHENEVER ANOTHER CHARACTER
;  IN A REDUNDANT COMMAND SET IS TRANSMITTED.
;
;  INREG - STORES THE CONSTANT, RPYCNT, WHICH
;  IS DECREMENTED WHENEVER ANOTHER CHARACTER
;  IN A REDUNDANT REPLY SET IS RECEIVED.
;
;  PRESCMD - STORES THE COMMAND PRESENTLY
;  BEING TRANSMITTED TO THE TX-100. SAVED TO
;  ALLOW FOR COMMAND DUPLICATION IN THE REDUN-
;  DANT SET.
;
;  LASTCMD - STORES THE COMMAND LAST TRANSMIT-
;  TED TO THE TX-100.  SAVED IN CASE THE TX-
;  100 REQUESTS A RETRANSMISSION.
;
;  XPECTED - STORES THE REPLY EXPECTED BACK
;  FROM THE TX-100 THAT WILL COMPLETE THE
;  STRUCTURE.
;
;  PRESRPY - STORES THE REPLY PRESENTLY BEING
;  RECEIVED FROM THE TX-100.  SAVED TO ENSURE
;  THAT ALL THREE REPLIES ARE IDENTICAL.
;
;  SOFTIME - STORES A CONSTANT THAT IS THEN
;  DECREMENTED TO SIMULATE DELAYS BETWEEN COM-
;  MAND TRANSMISSIONS TO THE TX-100.  THIS
;  CONSTANT CHANGES WHEN THE COMMAND SECTION
;  IS MOVED FROM THE TIMER 0 ROUTINE TO THE
;  MAIN ROUTINE.
;
;  RXCNT - STORES THE CONSTANT, RXMLIM, WHICH
;  IS DECREMENTED WHENEVER ANOTHER RETRANSMIS-
;  SION REQUEST IS SENT TO THE TX-100.
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
;
;  DIGIT - STORES A NUMBER, 1-4, CORRESPOND-
;  TO THE TEMPERATURE DIGIT CURRENTLY BEING
;  RECEIVED FROM THE TX-100 (4 DIGITS PER
;  TEMPERATURE PROBE).
;
;  PROBE - STORES A NUMBER, 1-16, CORRESPOND-
;  ING TO THE PROBE THAT TEMPERATURES ARE
;  CURRENTLY BEING RECEIVED FOR (16 PROBES).
;
;  OFFSET - STORES THE OFFSET FROM THE BASE
;  ADDRESS OF THE 64-BYTE TEMPERATURE ARRAY.
;  USED TO CORRECTLY STORE THE RECEIVED TX-
;  100 TEMPERATURE DATA.  OBTAINED BY MULTI-
;  PLYING THE PROBE NUMBER BY 4.
;
;  LPTR,HPTR - TEMPORARILY USED TO STORE THE
;  DPH REGISTER DURING RAM-TO-RAM TRANSFERS.
;
;  CHKSUML - STORES THE LOW BYTE OF THE GEN-
;  ERATED CHECKSUM.
;
;  CHKSUMH - STORES THE HIGH BYTE OF THE GEN-
;  ERATED CHECKSUM.
;
;  TXSUML - STORES THE LOW BYTE OF THE CHECK-
;  SUM RECEIVED FROM THE TX-100.
;
;  TXSUMH - STORES THE HIGH BYTE OF THE CHECK-
;  SUM RECEIVED FROM THE TX-100.
;
;  CNTR - SCRATCHPAD REGISTER.
;
;  COUNT1,COUNT2,COUNT3 - USED TO IMPLEMENT A
;  WAIT LOOP IN THE MAIN ROUTINE.  SIMULATES
;  THE DELAY BETWEEN COMMAND TRANSMISSIONS TO
;  THE TX-100.  USED IN CONJUNCTION WITH THE
;  SOFTIME REGISTER.
;
;  EAT - STORES A CONSTANT THAT IS THEN
;  DECREMENTED TO CREATE THE PROPER EXPECTED
;  REPLY TIMER PERIOD.
;
;==============================================
;
;==============================================
;DESCRIPTION OF BIT VARIABLES
;
;  STATUS1,STATUS2 - RESERVED BIT PLACES.
;
;  XMITBIT - USED IN THE XMITOK SUBROUTINE.
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
                ;   SET WHEN A TRANSMISSION IS CURRENTLY IN
                ;   PROGRESS.  CLEARED WHEN THE TRANSMISSION
                ;   IS FINISHED.
                ;
                ;   BADDATA - USED DURING DATA BLOCK TRANS-
                ;   MISSIONS.  SET WHEN BAD TEMPERATURE DATA
                ;   HAVE BEEN RECEIVED FROM THE TX-100.  ALSO
                ;   USED IN VERSION C FOR THE CALIBRATION DATA
                ;   BLOCK.
                ;
                ;   BADCHAR - SET WHEN A BAD CHARACTER HAS BEEN
                ;   RECEIVED FROM THE TX-100.  CLEAR FOR GOOD
                ;   TRANSMISSIONS.
                ;
                ;   SEQSTRT - SET WHEN A REDUNDANT REPLY SET
                ;   HAS ALREADY BEEN STARTED.  INDICATES THAT
                ;   REPLY DUPLICATION MUST BE CHECKED.  CLEAR
                ;   UNTIL THE FIRST BYTE IN THE SET IS RE-
                ;   CEIVED.
                ;
                ;   TEMPBIT - SET WHEN THE RECEIVE TEMPERATURES
                ;   REPLY IS RECEIVED FROM THE TX-100.  INDI-
                ;   CATES THAT A TEMPERATURE DATA BLOCK IS
                ;   BEING PROCESSED.
                ;
                ;   STATUS - SET ONLY WHEN THE NAME/STATUS COM-
                ;   MAND IS BEING SENT TO THE TX-100.  CLEARED
                ;   WHEN THE COMMAND CHANGES TO EITHER LOAD OR
                ;   INITIALIZE AND GO.
                ;
                ;   GO - SET ONLY WHEN THE INITIALIZE AND GO
                ;   COMMAND IS BEING SENT TO THE TX-100.
                ;   CLEARED WHEN THE COMMAND CHANGES TO TEMPER-
                ;   ATURES.
                ;
                ;   TEMPS - SET ONLY WHEN THE TEMPERATURES COM-
                ;   MAND IS BEING SENT TO THE TX-100.
                ;
                ;   CSLRCVD - SET WHEN THE LOW BYTE OF THE TX-
                ;   100 CHECKSUM IS EXPECTED.  CLEARED AFTER
                ;   THE FULL DATA BLOCK HAS BEEN RECEIVED.
                ;
                ;   CSHRCVD - SET WHEN THE HIGH BYTE OF THE TX-
                ;   100 CHECKSUM IS EXPECTED.  CLEARED AFTER
                ;   THE FULL DATA BLOCK HAS BEEN RECEIVED.
                ;
                ;   TXREADY - SET WHEN THE "SYSTEM READY" REPLY
                ;   (#1BH) IS RECEIVED FROM THE TX-100. SIGNALS
                ;   THE DEC TO SEND ONE MORE NAME/STATUS COM-
                ;   MAND AND THEN WAIT FOR A REPLY.
                ;
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
;   NORXMIT - SET WHEN THE TEMPERATURE BLOCK IS
;   EXPECTED.  DELAYS RETRANSMISSION REQUESTS
;   UNTIL AFTER THE FULL DATA BLOCK HAS BEEN
;   RECEIVED.
;
;   STATUS3 - MORE RESERVED BIT PLACES.
;
;   NORXMT - SET WHEN THE CALIBRATION BLOCK IS
;   EXPECTED.  DELAYS RETRANSMISSION REQUESTS
;   UNTIL AFTER THE FULL DATA BLOCK HAS BEEN
;   RECEIVED.
;
;   LU - SET ONLY WHEN THE LOAD COMMAND IS
;   BEING SENT TO THE TX-100.  CLEARED WHEN THE
;   COMMAND CHANGES TO TEMPERATURES.
;
;   CALBIT - SET WHEN THE UNLOAD COMMAND IS
;   RECEIVED FROM THE TX-100.  INDICATES THAT A
;   CALIBRATION DATA BLOCK IS BEING PROCESSED.
;
;   CALBIT2 - SET WHEN THE LOAD COMMAND IS RE-
;   CEIVED FROM THE TX-100.  INDICATES THAT A
;   FULL CALIBRATION DATA BLOCK MUST BE TRANS-
;   MITTED TO THE TX-100.  CLEARED WHEN THE
;   CALIBRATION DATA BLOCK IS FINISHED.
;
;   CDATA - SET TO INDICATE THAT THE CALIBRA-
;   TION DATA BLOCK IS NOT FINISHED TRANSMIT-
;   TING.  CLEARED AFTER THE LAST CHECKSUM BYTE
;   HAS BEEN SENT.
;
;   SENDCSL - SET WHEN THE LOW BYTE OF THE GEN-
;   ERATED CHECKSUM NEEDS TO BE SENT.  CLEARED
;   AFTER IT IS SENT.
;
;   SENDCSH - SET WHEN THE HIGH BYTE OF THE
;   GENERATED CHECKSUM NEEDS TO BE SENT.
;   CLEARED AFTER IT IS SENT.
;
;   LINE - SET WHEN THE LAST COMMAND TRANSMIT-
;   TED WAS A SINGLE LINE.  CLEARED  WHEN THE
;   LAST COMMAND WAS A DATA BLOCK.
;
;   SEQNUM - STORES THE DEC SEQUENCE NUMBER IN
;   THE MOST SIGNIFICANT BIT PLACE.  LOGICALLY
;   ORED WITH THE COMMAND TO BE TRANSMITTED IN
;   ORDER TO APPEND THE SEQUENCE BIT.
;
;   TXSEQNO - SET WHEN THE TX-100 SEQUENCE NUM-
;   BER IS EQUAL TO ONE.  CLEARED WHEN THE TX-
;   100 SEQUENCE NUMBER IS EQUAL TO ZERO.
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
                ;
                ;  SNSAME - SET WHEN THE TX-100 SEQUENCE NUM-
                ;  BER EQUALS THE DEC SEQUENCE NUMBER.
                ;  CLEARED WHEN NOT EQUAL.
                ;
                ;  NOSAVE - SET WHEN A RETRANSMISSION REQUEST
                ;  IS SENT TO THE TX-100.  INDICATES THAT THE
                ;  REQUEST SHOULD NOT BE STORED IN THE LASTCMD
                ;  REGISTER.  THIS IS NECESSARY TO MAINTAIN
                ;  SEQUENCE NUMBER CONTROL.
                ;
                ;  DECSNO - THE ACTUAL DEC SEQUENCE NUMBER.
                ;  RESIDES IN THE MSB OF SEQNUM.
                ;
                ;  HOLDBIT - SET WHEN THE DEC IS WAITING FOR
                ;  A REPLY FROM THE TX-100.  INDICATES THAT
                ;  THE NEXT COMMAND SHOULD NOT BE SENT YET.
                ;  CLEARED WHEN THE EXPECTED REPLY IS RECEIVED
                ;  AND THE DEC IS ALLOWED TO SEND ANOTHER
                ;  COMMAND.
                ;
                ;  TSTART - SET TO ACTIVATE THE EXPECTED REPLY
                ;  TIMER.  CLEARED WHEN DEACTIVATED.
                ;
                ;  DECOFF - SET WHEN THE DEC IS IN A SIMULATED
                ;  SHUTDOWN MODE.  AFTER THE SHUTDOWN COMMAND
                ;  HAS BEEN SENT TO THE TX-100,  THE DEC
                ;  CEASES ALL ACTIVITY.
                ;
                ;==============================================
                ;
                ;==============================================
                ;MEMORY ALLOCATION
                ;==============================================
0008            OUTREG  EQU     08H
0009            INREG   EQU     09H
000A            PRESCMD EQU     0AH
000B            LASTCMD EQU     0BH
000C            XPECTED EQU     0CH
000D            PRESRPY EQU     0DH
000E            SOFTIME EQU     0EH
000F            RXCNT   EQU     0FH
0010            DIGIT   EQU     10H
0011            PROBE   EQU     11H
0012            OFFSET  EQU     12H
0013            LPTR    EQU     13H
0014            HPTR    EQU     14H
0015            CHKSUML EQU     15H
0016            CHKSUMH EQU     16H
0017            TXSUML  EQU     17H
0018            TXSUMH  EQU     18H
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
0019            CNTR     EQU     19H
0025            COUNT1   EQU     25H
0026            COUNT2   EQU     26H
0027            COUNT3   EQU     27H
0028            EAT      EQU     28H
                ;
                ;==============================================
                ;BIT ALLOCATION
                ;==============================================
0020            STATUS1 EQU      20H
0021            STATUS2 EQU      21H
0022            STATUS3 EQU      22H
0023            STATUS4 EQU      23H
0024            SEQNUM   EQU     24H
                ;
0000            XMITBIT EQU      STATUS1.0
0001            BADDATA EQU      STATUS1.1
0002            BADCHAR EQU      STATUS1.2
0003            SEQSTRT EQU      STATUS1.3
0004            TEMPBIT EQU      STATUS1.4
0005            STATUS   EQU     STATUS1.5
0006            GO       EQU     STATUS1.6
0007            TEMPS    EQU     STATUS1.7
                ;
0008            CSLRCVD EQU      STATUS2.0
0009            CSHRCVD EQU      STATUS2.1
000A            TXREADY EQU      STATUS2.2
000B            NORXMIT EQU      STATUS2.3
000C            NORXMT   EQU     STATUS2.4
000D            LU       EQU     STATUS2.5
000E            CALBIT   EQU     STATUS2.6
000F            CALBIT2 EQU      STATUS2.7
                ;
0010            CDATA    EQU     STATUS3.0
0011            SENDCSL EQU      STATUS3.1
0012            SENDCSH EQU      STATUS3.2
0013            LINE     EQU     STATUS3.3
0014            TXSEQNO EQU      STATUS3.4
0015            SNSAME   EQU     STATUS3.5
0016            NOSAVE   EQU     STATUS3.6
0017            HOLDBIT EQU      STATUS3.7
                ;
0018            TSTART   EQU     STATUS4.0
0019            DECOFF   EQU     STATUS4.1
                ;
0027            DECSNO   EQU     SEQNUM.7
                ;
                ;==============================================
                ;PORT EQUIVALENTS
                ;==============================================
0093            MODE     EQU     P1.3
```

```
AVOCET SYSTEMS 8051 CROSS-ASSEMBLER -  VERSION 1.09

SOURCE FILE NAME: DECD.ASM

 0094            LCDSTS  EQU     P1.4
 0096            RS      EQU     P1.6
                 ;
                 ;=============================================
                 ;COMMAND-REPLY EQUIVALENTS
                 ;=============================================
 004E            NAMSTAT EQU     4EH
 004C            LOAD    EQU     4CH
 0055            UNLOAD  EQU     55H
 0049            INITGO  EQU     49H
 0044            DONE    EQU     44H
 0054            XMITTMP EQU     54H
 0045            RECTMP  EQU     45H
 0052            REXMIT  EQU     52H
 0053            SHUTDWN EQU     53H
                 ;
                 ;=============================================
                 ;CONSTANTS
                 ;=============================================
 0003            CMDCNT  EQU     3
 0003            RPYCNT  EQU     3
 0004            RXMLIM  EQU     4
 0001            UPDATE1 EQU     1
 0003            UPDATE2 EQU     3
 002A            UPDATE3 EQU     42
 000B            STAT1   EQU     0BH
 001B            STAT2   EQU     1BH
 002B            STAT3   EQU     2BH
 0030            LLIMI   EQU     30H
 0039            ULIMI   EQU     39H
                 ;
 0000            =============================================
                 ;RAM ADDRESSES
                 ;=============================================
 0100            LEDTA   EQU     $0100,$013F
 0200            STORAGE EQU     $0200,$023F
 0300            CALPRB  EQU     $0300,$030F
 0310            PRBSTS  EQU     $0310,$031F
 0320            ALLPRB  EQU     $0320,$032F
 0330            ALPHA1  EQU     $0330,$037F
 0380            BETA1   EQU     $0380,$03CF
                 ;
                 ;=============================================
                 ;MAIN ROUTINE
                 ;=============================================
 0100                    ORG     $0100
                 ;
                 ;=============================================
                 ;VECTOR JUMPS
                 ;       0000-020100
                 ;       000B-020102
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
                    ;       0023-020105
                    ;
                    ;=============================================================
0100 8006           INITJMP:SJMP    INIT
                    ;
0102 0201FA         TIMEJMP:LJMP    TIMERS
                    ;
0105 020258         SERJMP: LJMP    SERIAL
                    ;
                    ;=============================================================
                    ;8031 INITIALIZATIONS
                    ;=============================================================
0108 758140         INIT    MOV     SP,#$40
                    ;
                    ;TIMER SET-UP
010B 758921                 MOV     TMOD,   #21H
010E 758DE8                 MOV     TH1,    #0E8H
0111 D28C                   SETB    TRO
0113 D28E                   SETB    TR1
                    ;
                    ;SERIAL SET-UP
0115 7598D0                 MOV     SCON,   #0D0H
                    ;
                    ;INTERRUPT SET-UP
0118 D2A9                   SETB    ETO
011A D2AC                   SETB    ES
                    ;
                    ;=============================================================
                    ;BIT INITIALIZATIONS
                    ;=============================================================
011C C204                   CLR     TEMPBIT
011E C200                   CLR     XMITBIT
0120 C202                   CLR     BADCHAR
0122 C201                   CLR     BADDATA
0124 C203                   CLR     SEQSTRT
0126 D205                   SETB    STATUS
0128 C206                   CLR     GO
012A C207                   CLR     TEMPS
012C C20A                   CLR     TXREADY
012E C20B                   CLR     NORXMIT
0130 C20C                   CLR     NORXMT
0132 C20D                   CLR     LU
0134 C20E                   CLR     CALBIT
0136 C20F                   CLR     CALBIT2
0138 C210                   CLR     CDATA
013A D213                   SETB    LINE
013C C211                   CLR     SENDCSL
013E C212                   CLR     SENDCSH
0140 C216                   CLR     NOSAVE
0142 C219                   CLR     DECOFF
                    ;
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
                   ;=============================================
                   ;STORAGE INITIALIZATION
                   ;=============================================
0144 752400                  MOV     SEQNUM, #00H
0147 120685                  CALL    NEWSEQ
014A 120670                  CALL    TIMESET
014D 750803                  MOV     OUTREG, #CMDCNT
0150 750F00                  MOV     RXCNT,  #00H
0153 750E01                  MOV     SOFTIME,#UPDATE1
                   ;
0156 751940                  MOV     CNTR,    #40H
0159 900200                  MOV     DPTR,    #STORAGE
015C 740F                    MOV     A,       #0FH
015E F0            CLEAR:    MOVX    @DPTR,   A
015F 0582                    INC     DPL
0161 D519FA                  DJNZ    CNTR,    CLEAR
                   ;
0164 120696                  CALL    LEDOFF
0167 1206FF                  CALL    CLRLCD
016A 900748                  MOV     DPTR,    #MES0
016D 12071B                  CALL    PMLCD
                   ;
0170 D2AF                    SETB    EA
                   ;
                   ;=============================================
                   ;CENTRAL COMMANDS SECTION
                   ;=============================================
0172 752500        CENTRAL:MOV       COUNT1, #00H
0175 752600        WAIT1:    MOV     COUNT2, #00H
0178 75270A        WAIT2:    MOV     COUNT3, #0AH
017B D527FD                  DJNZ    COUNT3, $
017E D526F7                  DJNZ    COUNT2, WAIT2
0181 D525F1                  DJNZ    COUNT1, WAIT1
0184 D50EEB                  DJNZ    SOFTIME,CENTRAL
                   ;
0187 200752                  JB      TEMPS,  GROUP3
018A 200639                  JB      GO,     GROUP2
018D 200D1E                  JB      LU,     GROUP1
                   ;
0190 744E                    MOV     A,      #NAMSTAT
0192 4524                    ORL     A,      SEQNUM
0194 F50A                    MOV     PRESCMD,A
0196 A2D0                    MOV     C,      P
0198 12068B                  CALL    XMITOK
019B 929B                    MOV     TB8,    C
019D F599                    MOV     SBUF,   A
019F 750C0B                  MOV     XPECTED,#0BH
01A2 750E01                  MOV     SOFTIME,#UPDATE1
01A5 300A4A                  JNB     TXREADY,FINISH1
01A8 C20A                    CLR     TXREADY
01AA D217                    SETB    HOLDBIT
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
 01AC 8044              SJMP    FINISH1
                ;
 01AE D20C      GROUP1: SETB    NORXMT
 01B0 744C              MOV     A,      #LOAD
 01B2 4524              ORL     A,      SEQNUM
 01B4 F50A              MOV     PRESCMD,A
 01B6 A2D0              MOV     C,      P
 01B8 12068B            CALL    XMITOK
 01BB 929B              MOV     TB8,    C
 01BD F599              MOV     SBUF,   A
 01BF 750C55            MOV     XPECTED,#UNLOAD
 01C2 D217              SETB    HOLDBIT
 01C4 802C              SJMP    FINISH1
                ;
 01C6 7449      GROUP2: MOV     A,      #INITGO
 01C8 4524              ORL     A,      SEQNUM
 01CA F50A              MOV     PRESCMD,A
 01CC A2D0              MOV     C,      P
 01CE 12068B            CALL    XMITOK
 01D1 929B              MOV     TB8,    C
 01D3 F599              MOV     SBUF,   A
 01D5 750C44            MOV     XPECTED,#DONE
 01D8 D217              SETB    HOLDBIT
 01DA 8016              SJMP    FINISH1
                ;
 01DC D20B      GROUP3: SETB    NORXMIT
 01DE 7454              MOV     A,      #XMITTMP
 01E0 4524              ORL     A,      SEQNUM
 01E2 F50A              MOV     PRESCMD,A
 01E4 A2D0              MOV     C,      P
 01E6 12068B            CALL    XMITOK
 01E9 929B              MOV     TB8,    C
 01EB F599              MOV     SBUF,   A
 01ED 750C45            MOV     XPECTED,#RECTMP
 01F0 D217              SETB    HOLDBIT
                ;
 01F2 D213      FINISH1:SETB    LINE
 01F4 2017FD            JB      HOLDBIT,$
 01F7 020172            LJMP    CENTRAL
                ;
                ;==========================================
                ;TIMERO INTERRUPT HANDLER - 3 SECOND DELAY
                ;==========================================
 01FA 30185A    TIMERS: JNB     TSTART, TIME6
 01FD D52857            DJNZ    EAT,    TIME6
 0200 C0E0              PUSH    ACC
 0202 C0D0              PUSH    PSW
 0204 C082              PUSH    DPL
 0206 C083              PUSH    DPH
 0208 C0F0              PUSH    B
 020A 050F              INC     RXCNT
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
 020C E50F                MOV     A,       RXCNT
 020E B40409              CJNE    A,       #RXMLIM,        TIME2
                ;
 0211 120670              CALL    TIMESET
 0214 D219                SETB    DECOFF
 0216 7453                MOV     A,       #SHUTDWN
 0218 801E                SJMP    TIME4
                ;
 021A 120685    TIME2:    CALL    NEWSEQ
 021D 20160D              JB      NOSAVE,  TIME3
 0220 1206FF              CALL    CLRLCD
 0223 900874              MOV     DPTR,    #MES12
 0226 12071B              CALL    PMLCD
 0229 E50B                MOV     A,       LASTCMD
 022B 800B                SJMP    TIME4
                ;
 022D 1206FF    TIME3:    CALL    CLRLCD
 0230 90088D              MOV     DPTR,    #MES13
 0233 12071B              CALL    PMLCD
 0236 7452                MOV     A,       #REXMIT
                ;
 0238 4524      TIME4:    ORL     A,       SEQNUM
 023A A2D0                MOV     C,       P
 023C 201602              JB      NOSAVE,  TIME5
 023F F50A                MOV     PRESCMD,A
 0241 12068B    TIME5:    CALL    XMITOK
 0244 929B                MOV     TB8,     C
 0246 F599                MOV     SBUF,    A
                ;
 0248 75282A              MOV     EAT,     #UPDATE3
 024B C298                CLR     RI
 024D D0F0                POP     B
 024F D083                POP     DPH
 0251 D082                POP     DPL
 0253 D0D0                POP     PSW
 0255 D0E0                POP     ACC
 0257 32        TIME6:    RETI
                ;
                ;==================================================
                ;SERIAL INTERRUPT HANDLER
                ;==================================================
 0258 C0E0      SERIAL:   PUSH    ACC
 025A C0D0                PUSH    PSW
 025C C082                PUSH    DPL
 025E C083                PUSH    DPH
 0260 C0F0                PUSH    B
 0262 209841              JB      RI,      RECEIVE
 0265 C299                CLR     TI
 0267 C200                CLR     XMITBIT
 0269 301005              JNB     CDATA,   NODATA
 026C 120614              CALL    CALOUT
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
 026F 801D                     SJMP     CMDDONE
 0271 D50825      NODATA:  DJNZ     OUTREG, NOTFIN
 0274 200502                   JB       STATUS, NODATA1
 0277 D218                     SETB     TSTART
 0279 850A0B      NODATA1:MOV      LASTCMD,PRESCMD
 027C 750803                   MOV      OUTREG, #CMDCNT
 027F 301903                   JNB      DECOFF, CALCHK
 0282 020466                   LJMP     KILL
 0285 300F06      CALCHK:  JNB      CALBIT2,CMDDONE
 0288 120608                   CALL     CINIT
 028B 120614                   CALL     CALOUT
 028E D0F0        CMDDONE:POP      B
 0290 D083                     POP      DPH
 0292 D082                     POP      DPL
 0294 D0D0                     POP      PSW
 0296 D0E0                     POP      ACC
 0298 32                       RETI
 0299 201605      NOTFIN:  JB       NOSAVE, NOTFIN1
 029C E50A                     MOV      A,      PRESCMD
 029E 020481                   LJMP     SERRET
 02A1 7452        NOTFIN1:MOV      A,      #REXMIT
 02A3 020481                   LJMP     SERRET
                  ;
 02A6 300203      RECEIVE:JNB      BADCHAR,PARITY1
 02A9 020433                   LJMP     BADSEQ
 02AC E599        PARITY1:MOV      A,      SBUF
 02AE 8599F0                   MOV      B,      SBUF
 02B1 200E12                   JB       CALBIT, TMPJMP1
 02B4 200412                   JB       TEMPBIT,TMPJMP2
 02B7 20D006                   JB       P,      ODDPAR1
 02BA 309A0F                   JNB      RB8,    SEQCHEK
 02BD 020433                   LJMP     BADSEQ
 02C0 209A09      ODDPAR1:JB       RB8,    SEQCHEK
 02C3 020433                   LJMP     BADSEQ
                  ;
 02C6 02049E      TMPJMP1:LJMP     CALBLK
 02C9 020521      TMPJMP2:LJMP     TEMPBLK
                  ;
 02CC 300308      SEQCHEK:JNB      SEQSTRT,GOODSEQ
 02CF B50D02                   CJNE     A,      PRESRPY,        TJMP
 02D2 8007                     SJMP     RPY0
 02D4 020433      TJMP:    LJMP     BADSEQ
                  ;
 02D7 D203        GOODSEQ:SETB     SEQSTRT
 02D9 F50D                     MOV      PRESRPY,A
                  ;
 02DB D50905      RPY0:    DJNZ     INREG,  TJMP0
 02DE 120685                   CALL     NEWSEQ
 02E1 8003                     SJMP     GETTXSN
 02E3 020491      TJMP0:   LJMP     NOCMD
                  ;
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
02E6 C214        GETTXSN:CLR     TXSEQNO
02E8 30E706              JNB     ACC.7,   RITERPY
02EB D214                SETB    TXSEQNO
02ED C2E7                CLR     ACC.7
02EF F5F0                MOV     B,       A
                 ;
02F1 300504      RITERPY:JNB     STATUS,  RITE1
02F4 740F                MOV     A,       #0FH
02F6 55F0                ANL     A,       B
02F8 B50C04      RITE1:  CJNE    A,       XPECTED,        WRONG
02FB E5F0                MOV     A,       B
02FD 8005                SJMP    RPY1A
02FF E5F0        WRONG:  MOV     A,       B
0301 0203E9              LJMP    RPY5
                 ;
                 ;VALID REPLY COMPARISON
                 ;
0304 B40B11      RPY1A:  CJNE    A,       #STAT1,         RPY1B
0307 120676              CALL    COMPARE
030A 301543              JNB     SNSAME,  TJMP2
030D 1206FF              CALL    CLRLCD
0310 900748              MOV     DPTR,    #MES0
0313 12071B              CALL    PMLCD
0316 802C                SJMP    TJMP1
0318 B41B13      RPY1B:  CJNE    A,       #STAT2,         RPY1C
031B 120676              CALL    COMPARE
031E 30152F              JNB     SNSAME,  TJMP2
0321 1206FF              CALL    CLRLCD
0324 900761              MOV     DPTR,    #MES1
0327 12071B              CALL    PMLCD
032A D20A                SETB    TXREADY
032C 8016                SJMP    TJMP1
032E B42B22      RPY1C:  CJNE    A,       #STAT3,         RPY2
0331 120676              CALL    COMPARE
0334 301519              JNB     SNSAME,  TJMP2
0337 1206FF              CALL    CLRLCD
033A 90077A              MOV     DPTR,    #MES2
033D 12071B              CALL    PMLCD
0340 C205                CLR     STATUS
0342 D20D                SETB    LU
0344 C216        TJMP1:  CLR     NOSAVE
0346 B227                CPL     DECSNO
0348 C217                CLR     HOLDBIT
034A 750F00              MOV     RXCNT,   #00H
034D 020491              LJMP    NOCMD
0350 02043F      TJMP2:  LJMP    RXMIT
                 ;
0353 B45534      RPY2:   CJNE    A,       #UNLOAD,        RPY3
0356 120676              CALL    COMPARE
0359 201503              JB      SNSAME,  RPY2OK
035C 02043F              LJMP    RXMIT
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
 035F C216       RPY2OK: CLR     NOSAVE
 0361 B227               CPL     DECSNO
 0363 1206FF             CALL    CLRLCD
 0366 900829             MOV     DPTR,    #MES9
 0369 12071B             CALL    PMLCD
 036C D20E       CINIT2: SETB    CALBIT
 036E 7512D0             MOV     OFFSET, #0D0H
 0371 751500             MOV     CHKSUML,#00H
 0374 751600             MOV     CHKSUMH,#00H
 0377 C208               CLR     CSLRCVD
 0379 C209               CLR     CSHRCVD
 037B 020491             LJMP    NOCMD
                 ;
 037E B44C09     RPY2A:  CJNE    A,       #LOAD,          RPY3
 0381 C213               CLR     LINE
 0383 D20F               SETB    CALBIT2
 0385 7455               MOV     A,       #UNLOAD
 0387 020481             LJMP    SERRET
                 ;
 038A B44428     RPY3:   CJNE    A,       #DONE,          RPY4
 038D 120670             CALL    TIMESET
 0390 120676             CALL    COMPARE
 0393 201503             JB      SNSAME, RPY3OK
 0396 02043F             LJMP    RXMIT
 0399 C216       RPY3OK: CLR     NOSAVE
 039B B227               CPL     DECSNO
 039D 1206FF             CALL    CLRLCD
 03A0 900793             MOV     DPTR,    #MES3
 03A3 12071B             CALL    PMLCD
 03A6 C206               CLR     GO
 03A8 D207               SETB    TEMPS
 03AA C217               CLR     HOLDBIT
 03AC 750E03             MOV     SOFTIME,#UPDATE2
 03AF 750F00             MOV     RXCNT,   #00H
 03B2 020491             LJMP    NOCMD
                 ;
 03B5 B44531     RPY4:   CJNE    A,       #RECTMP,        RPY5
 03B8 120676             CALL    COMPARE
 03BB 201503             JB      SNSAME, RPY4OK
 03BE 02043F             LJMP    RXMIT
 03C1 C216       RPY4OK: CLR     NOSAVE
 03C3 B227               CPL     DECSNO
 03C5 1206FF             CALL    CLRLCD
 03C8 9007AC             MOV     DPTR,    #MES4
 03CB 12071B             CALL    PMLCD
 03CE D204       TINIT:  SETB    TEMPBIT
 03D0 751004             MOV     DIGIT,   #04H
 03D3 751100             MOV     PROBE,   #00H
 03D6 E511               MOV     A,       PROBE
 03D8 23                 RL      A
 03D9 23                 RL      A
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
 03DA F512                MOV     OFFSET, A
 03DC 751500              MOV     CHKSUML,#00H
 03DF 751600              MOV     CHKSUMH,#00H
 03E2 C208                CLR     CSLRCVD
 03E4 C208                CLR     CSLRCVD
 03E6 020491              LJMP    NOCMD
                 ;
 03E9 B4522B     RPY5:    CJNE    A,      #REXMIT,        RPY6
 03EC 120670              CALL    TIMESET
 03EF 120676              CALL    COMPARE
 03F2 301503              JNB     SNSAME, SENDCMD
 03F5 02043F              LJMP    RXMIT
 03F8 C216       SENDCMD: CLR     NOSAVE
 03FA 1206FF              CALL    CLRLCD
 03FD 9007C5              MOV     DPTR,   #MES5
 0400 12071B              CALL    PMLCD
 0403 C217                CLR     HOLDBIT
 0405 750F00              MOV     RXCNT,  #00H
 0408 301305              JNB     LINE,   BLOCK
 040B E50B                MOV     A,      LASTCMD
 040D 020481              LJMP    SERRET
 0410 D20F       BLOCK:   SETB    CALBIT2
 0412 7455                MOV     A,      #UNLOAD
 0414 020481              LJMP    SERRET
                 ;
 0417 B45315     RPY6:    CJNE    A,      #SHUTDWN,       TJMP7
 041A 120670              CALL    TIMESET
 041D C2AF                CLR     EA
 041F D217                SETB    HOLDBIT
 0421 120696              CALL    LEDOFF
 0424 1206FF              CALL    CLRLCD
 0427 9007DE              MOV     DPTR,   #MES6
 042A 12071B              CALL    PMLCD
 042D 80FE                SJMP    $
 042F C202       TJMP7:   CLR     BADCHAR
 0431 800C                SJMP    RXMIT
                 ;
                 ;BAD SEQUENCE RECEIVED
 0433 D202       BADSEQ:  SETB    BADCHAR
 0435 D50904              DJNZ    INREG,  BADSEQ1
 0438 C202                CLR     BADCHAR
 043A 8003                SJMP    RXMIT
 043C 020491     BADSEQ1: LJMP    NOCMD
                 ;
                 ;RETRANSMISSION DESIRED
 043F 300C05     RXMIT:   JNB     NORXMT,RXMIT1
 0442 D201                SETB    BADDATA
 0444 02036C              LJMP    CINIT2
 0447 300B04     RXMIT1:  JNB     NORXMIT,RXMIT2
 044A D201                SETB    BADDATA
 044C 8080                SJMP    TINIT
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
044E 120670    RXMIT2: CALL    TIMESET
0451 D216              SETB    NOSAVE
0453 D217              SETB    HOLDBIT
0455 120685            CALL    NEWSEQ
0458 050F              INC     RXCNT
045A E50F              MOV     A,      RXCNT
045C B40417            CJNE    A,      #RXMLIM,        AGAIN
045F D219              SETB    DECOFF
0461 7453              MOV     A,      #SHUTDWN
0463 020481            LJMP    SERRET
0466 C2AF      KILL:   CLR     EA
0468 120696            CALL    LEDOFF
046B 1206FF            CALL    CLRLCD
046E 900810            MOV     DPTR,   #MES8
0471 12071B            CALL    PMLCD
0474 80FE              SJMP    $
0476 1206FF    AGAIN:  CALL    CLRLCD
0479 9007F7            MOV     DPTR,   #MES7
047C 12071B            CALL    PMLCD
047F 7452              MOV     A,      #REXMIT
               ;
0481 4524      SERRET: ORL     A,      SEQNUM
0483 A2D0              MOV     C,      P
0485 201602            JB      NOSAVE, SERRET1
0488 F50A              MOV     PRESCMD,A
048A 12068B    SERRET1:CALL    XMITOK
048D 929B              MOV     TB8,    C
048F F599              MOV     SBUF,   A
0491 C298      NOCMD:  CLR     RI
0493 D0F0              POP     B
0495 D083              POP     DPH
0497 D082              POP     DPL
0499 D0D0              POP     PSW
049B D0E0              POP     ACC
049D 32                RETI
               ;
               ;=====================================================
               ;RECEIVES CAL DATA FROM THE TX-100
               ;=====================================================
049E 300103    CALBLK: JNB     BADDATA,PARITY3
04A1 0204F2            LJMP    MOVPNT
04A4 20D008    PARITY3:JB      P,      ODDPAR3
04A7 309A0D            JNB     RB8,    TXCS1
04AA D201              SETB    BADDATA
04AC 0204F2            LJMP    MOVPNT
04AF 209A05    ODDPAR3:JB      RB8,    TXCS1
04B2 D201              SETB    BADDATA
04B4 0204F2            LJMP    MOVPNT
               ;
04B7 30081F    TXCS1:  JNB     CSLRCVD,DATA
04BA 200907            JB      CSHRCVD,TXCSH1
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
04BD D209                  SETB    CSHRCVD
04BF F517                  MOV     TXSUML,A
04C1 020517                LJMP    CFINISH
04C4 F518       TXCSH1:    MOV     TXSUMH,A
04C6 120670                CALL    TIMESET
04C9 E517                  MOV     A,      TXSUML
04CB B5154C                CJNE    A,      CHKSUML,        CAGAIN
04CE E518                  MOV     A,      TXSUMH
04D0 B51647                CJNE    A,      CHKSUMH,        CAGAIN
04D3 300128                JNB     BADDATA,REINIT
04D6 02051A                LJMP    CAGAIN
                ;
04D9 C3         DATA:      CLR     C
04DA E5F0                  MOV     A,      B
04DC 2515                  ADD     A,      CHKSUML
04DE F515                  MOV     CHKSUML,A
04E0 5002                  JNC     NOCARY2
04E2 0516                  INC     CHKSUMH
                ;
04E4 900300     NOCARY2:MOV        DPTR,   #CALPRB
04E7 1582                  DEC     DPL
04E9 E582                  MOV     A,      DPL
04EB 2512                  ADD     A,      OFFSET
04ED F582                  MOV     DPL,    A
04EF E5F0                  MOV     A,      B
04F1 F0                    MOVX    @DPTR,  A
                ;
04F2 2008C2     MOVPNT:    JB      CSLRCVD,TXCS1
04F5 D5121F                DJNZ    OFFSET, CFINISH
04F8 D208                  SETB    CSLRCVD
04FA C209                  CLR     CSHRCVD
04FC 8019                  SJMP    CFINISH
                ;
04FE 1206FF     REINIT:    CALL    CLRLCD
0501 900842                MOV     DPTR,   #MES10
0504 12071B                CALL    PMLCD
0507 C20E                  CLR     CALBIT
0509 C20C                  CLR     NORXMT
050B 750F00                MOV     RXCNT,  #00H
050E C20D                  CLR     LU
0510 D206                  SETB    GO
0512 750E01                MOV     SOFTIME,#UPDATE1
0515 C217                  CLR     HOLDBIT
                ;
0517 020491     CFINISH:LJMP       NOCMD
051A C201       CAGAIN:    CLR     BADDATA
051C C20E                  CLR     CALBIT
051E 02044E                LJMP    RXMIT2
                ;
                ;==============================================================
                ;RECEIVES TEMP DATA FROM TX-100
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
                      ;===============================================================
0521 300103           TEMPBLK:JNB      BADDATA,PARITY2
0524 0205A5                   LJMP     MOVPNTR
0527 20D008           PARITY2:JB       P,       ODDPAR2
052A 309A0D                   JNB      RB8,     TXCS
052D D201                     SETB     BADDATA
052F 0205A5                   LJMP     MOVPNTR
0532 209A05           ODDPAR2:JB       RB8,     TXCS
0535 D201                     SETB     BADDATA
0537 0205A5                   LJMP     MOVPNTR
                      ;
053A 300815           TXCS:   JNB      CSLRCVD,LLCHEK1
053D 200907                   JB       CSHRCVD,TXCSH
0540 D209                     SETB     CSHRCVD
0542 F517                     MOV      TXSUML,A
0544 0205FE                   LJMP     TFINISH
0547 F518             TXCSH:  MOV      TXSUMH,A
0549 120670                   CALL     TIMESET
054C 300174                   JNB      BADDATA,LOAD1
054F 020601                   LJMP     TAGAIN
                      ;
0552 C3               LLCHEK1:CLR      C
0553 9430                     SUBB     A,       #LLIMI
0555 5005                     JNC      ULCHEK1
0557 D201                     SETB     BADDATA
0559 0205A5                   LJMP     MOVPNTR
055C C3               ULCHEK1:CLR      C
055D 7439                     MOV      A,       #ULIMI
055F 95F0                     SUBB     A,       B
0561 5005                     JNC      TDATA
0563 D201                     SETB     BADDATA
0565 0205A5                   LJMP     MOVPNTR
                      ;
0568 C3               TDATA:  CLR      C
0569 E5F0                     MOV      A,       B
056B 2515                     ADD      A,       CHKSUML
056D F515                     MOV      CHKSUML,A
056F 5002                     JNC      NOCARY1
0571 0516                     INC      CHKSUMH
                      ;
0573 900200           NOCARY1:MOV      DPTR,    #STORAGE
0576 1582                     DEC      DPL
0578 E582                     MOV      A,       DPL
057A 2512                     ADD      A,       OFFSET
057C 2510                     ADD      A,       DIGIT
057E F582                     MOV      DPL,     A
0580 740F                     MOV      A,       #0FH
0582 55F0                     ANL      A,       B
0584 F5F0                     MOV      B,       A
0586 E510                     MOV      A,       DIGIT
0588 B4040D                   CJNE     A,       #04H,              THIRD
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
058B E5F0              MOV      A,        B
058D 6004              JZ       NOTENS
058F 4480              ORL      A,        #80H
0591 8002              SJMP     FOURTH
0593 748F     NOTENS:  MOV      A,        #8FH
0595 F0       FOURTH:  MOVX     @DPTR,    A
0596 800D              SJMP     MOVPNTR
0598 B40305   THIRD:   CJNE     A,        #03H,              SECOND
059B E5F0              MOV      A,        B
059D F0                MOVX     @DPTR,    A
059E 8005              SJMP     MOVPNTR
05A0 E5F0     SECOND:  MOV      A,        B
05A2 4480              ORL      A,        #80H
05A4 F0                MOVX     @DPTR,    A
              ;
05A5 200892   MOVPNTR:JB        CSLRCVD,TXCS
05A8 D51053            DJNZ     DIGIT,    TFINISH
05AB E511              MOV      A,        PROBE
05AD B40F06            CJNE     A,        #0FH,              MOREDAT
05B0 D208              SETB     CSLRCVD
05B2 C209              CLR      CSHRCVD
05B4 8048              SJMP     TFINISH
05B6 0511     MOREDAT:INC       PROBE
05B8 05E0              INC      ACC
05BA 23                RL       A
05BB 23                RL       A
05BC F512              MOV      OFFSET,   A
05BE 751004            MOV      DIGIT,    #04H
05C1 803B              SJMP     TFINISH
              ;
05C3 E517     LOAD1:   MOV      A,        TXSUML
05C5 B51539            CJNE     A,        CHKSUML,           TAGAIN
05C8 E518              MOV      A,        TXSUMH
05CA B51634            CJNE     A,        CHKSUMH,           TAGAIN
05CD 751301            MOV      LPTR,     #01H
05D0 751402            MOV      HPTR,     #02H
05D3 758200            MOV      DPL,      #00H
05D6 751940            MOV      CNTR,     #40H
05D9 851483   LOOP:    MOV      DPH,      HPTR
05DC E0                MOVX     A,        @DPTR
05DD 851383            MOV      DPH,      LPTR
05E0 F0                MOVX     @DPTR,    A
05E1 0582              INC      DPL
05E3 D519F3            DJNZ     CNTR,     LOOP
05E6 1206A8            CALL     LED
05E9 1206FF            CALL     CLRLCD
05EC 900793            MOV      DPTR,     #MES3
05EF 12071B            CALL     PMLCD
05F2 C204              CLR      TEMPBIT
05F4 C20B              CLR      NORXMIT
05F6 750F00            MOV      RXCNT,    #00H
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
05F9 750E03             MOV     SOFTIME,#UPDATE2
05FC C217               CLR     HOLDBIT
                ;
05FE 020491     TFINISH:LJMP    NOCMD
0601 C201       TAGAIN: CLR     BADDATA
0603 C204               CLR     TEMPBIT
0605 02044E             LJMP    RXMIT2
                ;
                ;==============================================
                ;INITIALIZES POINTER TO ALLPRB ARRAY
                ;==============================================
0608 D210       CINIT:  SETB    CDATA
060A 7512D0             MOV     OFFSET, #0D0H
060D 751500             MOV     CHKSUML,#00H
0610 751600             MOV     CHKSUMH,#00H
0613 22                 RET
                ;
                ;==============================================
                ;OUTPUTS CALIBRATION DATA TO TX-100
                ;==============================================
0614 30112F     CALOUT: JNB     SENDCSL,SENDCAL
0617 201209             JB      SENDCSH,SENDCS1
061A D212               SETB    SENDCSH
061C E515               MOV     A,      CHKSUML
061E F5F0               MOV     B,      A
0620 020664             LJMP    COUT
0623 750801     SENDCS1:MOV     OUTREG, #01H
0626 1206FF             CALL    CLRLCD
0629 90085B             MOV     DPTR,   #MES11
062C 12071B             CALL    PMLCD
062F C20D               CLR     LU
0631 D206               SETB    GO
0633 750E03             MOV     SOFTIME,#UPDATE2
0636 C217               CLR     HOLDBIT
0638 E516               MOV     A,      CHKSUMH
063A C20F               CLR     CALBIT2
063C C210               CLR     CDATA
063E C212               CLR     SENDCSH
0640 C211               CLR     SENDCSL
0642 F5F0               MOV     B,      A
0644 801E               SJMP    COUT
                ;
0646 900300     SENDCAL:MOV     DPTR,   #CALPRB
0649 1582               DEC     DPL
064B E582               MOV     A,      DPL
064D 2512               ADD     A,      OFFSET
064F F582               MOV     DPL,    A
0651 E0                 MOVX    A,      @DPTR
0652 F5F0               MOV     B,      A
0654 C3                 CLR     C
0655 2515               ADD     A,      CHKSUML
```

```
AVOCET SYSTEMS 8051 CROSS-ASSEMBLER -  VERSION 1.09

SOURCE FILE NAME: DECD.ASM


 0657 F515               MOV      CHKSUML,A
 0659 5002               JNC      NOCARY3
 065B 0516               INC      CHKSUMH
                ;
 065D D51204    NOCARY3:DJNZ      OFFSET, COUT
 0660 D211               SETB     SENDCSL
 0662 C212               CLR      SENDCSH
                ;
 0664 E5F0      COUT:    MOV      A,       B
 0666 A2D0               MOV      C,       P
 0668 12068B             CALL     XMITOK
 066B 929B               MOV      TB8,     C
 066D F599               MOV      SBUF,    A
 066F 22                 RET
                ;
                ;==================================================
                ;RESETS EXPECTED ACKNOWLEDGE TIMER
                ;==================================================
 0670 75282A    TIMESET:MOV       EAT,     #UPDATE3
 0673 C218               CLR      TSTART
 0675 22                 RET
                ;
                ;==================================================
                ;COMPARES SEQUENCE NUMBERS
                ;==================================================
 0676 202706    COMPARE:JB        DECSNO, COMP2
 0679 301406             JNB      TXSEQNO,COMP3
 067C C215      COMP1:   CLR      SNSAME
 067E 22                 RET
 067F 3014FA    COMP2:   JNB      TXSEQNO,COMP1
 0682 D215      COMP3:   SETB     SNSAME
 0684 22                 RET
                ;
                ;==================================================
                ;REINITIALIZES FOR NEW CMD SEQUENCE
                ;==================================================
 0685 750903    NEWSEQ:  MOV      INREG,  #RPYCNT
 0688 C203               CLR      SEQSTRT
 068A 22                 RET
                ;
                ;==================================================
                ;ENSURES THAT XMIT BUF IS CLEAR
                ;==================================================
 068B 300005    XMITOK:  JNB      XMITBIT,XOK
 068E 3099FA             JNB      TI,      XMITOK
 0691 C299               CLR      TI
 0693 D200      XOK:     SETB     XMITBIT
 0695 22                 RET
                ;
                ;==================================================
                ;LED FUNCTIONS
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
                   ;=================================================
0696 751940        LEDOFF: MOV      CNTR,    #40H
0699 740F                  MOV      A,       #0FH
069B 900100                MOV      DPTR,    #LEDTA
069E F0            LEDOFF1:MOVX     @DPTR,   A
069F 0582                  INC      DPL
06A1 D519FA                DJNZ     CNTR,    LEDOFF1
06A4 1206A8                CALL     LED
06A7 22                    RET
                   ;
06A8 751908        LED     MOV      CNTR,#$08
06AB 7800                  MOV      R0,#$00
06AD C293          LED3    CLR      MODE
06AF 7583B0                MOV      DPH,#$B0
06B2 E8                    MOV      A,R0
06B3 C3                    CLR      C
06B4 13                    RRC      A
06B5 F582                  MOV      DPL,A
06B7 7490                  MOV      A,#%10010000
06B9 F0                    MOVX     @DPTR,A
06BA 00                    NOP
06BB 00                    NOP
06BC D293                  SETB     MODE
06BE C2D5                  CLR      F0
06C0 900100        LED4    MOV      DPTR,#LEDTA
06C3 E8                    MOV      A,R0
06C4 C3                    CLR      C
06C5 33                    RLC      A
06C6 33                    RLC      A
06C7 2582                  ADD      A,DPL
06C9 F582                  MOV      DPL,A
06CB E0                    MOVX     A,@DPTR
06CC FA                    MOV      R2,A
06CD 0582                  INC      DPL
06CF E0                    MOVX     A,@DPTR
06D0 FB                    MOV      R3,A
06D1 0582                  INC      DPL
06D3 E0                    MOVX     A,@DPTR
06D4 FC                    MOV      R4,A
06D5 0582                  INC      DPL
06D7 E0                    MOVX     A,@DPTR
06D8 FD                    MOV      R5,A
06D9 7583B0        LED2    MOV      DPH,#$B0
06DC E8                    MOV      A,R0
06DD C3                    CLR      C
06DE 13                    RRC      A
06DF F582                  MOV      DPL,A
06E1 EA                    MOV      A,R2
06E2 F0                    MOVX     @DPTR,A
06E3 00                    NOP
06E4 00                    NOP
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
06E5 00                       NOP
06E6 00                       NOP
06E7 EB                       MOV     A,R3
06E8 F0                       MOVX    @DPTR,A
06E9 00                       NOP
06EA 00                       NOP
06EB 00                       NOP
06EC 00                       NOP
06ED EC                       MOV     A,R4
06EE F0                       MOVX    @DPTR,A
06EF 00                       NOP
06F0 00                       NOP
06F1 00                       NOP
06F2 00                       NOP
06F3 ED                       MOV     A,R5
06F4 F0                       MOVX    @DPTR,A
06F5 08                       INC     R0
06F6 B2D5                     CPL     F0
06F8 20D5C5                   JB      F0,LED4
06FB D519AF                   DJNZ    CNTR,LED3
06FE 22                       RET
                    ;
                    ;==============================================
                    ;LCD FUNCTIONS
                    ;==============================================
06FF C296           CLRLCD    CLR     RS
0701 7438                     MOV     A,#$38
0703 120738                   CALL    LCD
0706 120738                   CALL    LCD
0709 7406                     MOV     A,#$06
070B 120738                   CALL    LCD
070E 740C                     MOV     A,#$0C
0710 120738                   CALL    LCD
0713 7401                     MOV     A,#$01
0715 120738                   CALL    LCD
0718 D296                     SETB    RS
071A 22                       RET
                    ;
071B D296           PMLCD     SETB    RS
071D 7400                     MOV     A,#$00
071F 93                       MOVC    A,@A+DPTR
0720 F8                       MOV     R0,A
0721 A3                       INC     DPTR
0722 7400           PMLCD1    MOV     A,#$00
0724 93                       MOVC    A,@A+DPTR
0725 120738                   CALL    LCD
0728 A3                       INC     DPTR
0729 D8F7                     DJNZ    R0,PMLCD1
072B 22                       RET
                    ;
072C C296           NXTLN     CLR     RS
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
072E 74C0                  MOV     A,#$C0
0730 120738                CALL    LCD
0733 D296                  SETB    RS
0735 22                    RET
                  ;
0736 2430                  ADD     A,#$30
0738 C082         LCD      PUSH    DPL
073A C083                  PUSH    DPH
073C 2094FD                JB      LCDSTS,$
073F 908000                MOV     DPTR,#$8000
0742 F0                    MOVX    @DPTR,A
0743 D083                  POP     DPH
0745 D082                  POP     DPL
0747 22                    RET
                  ;
                  ;==========================================================
                  ;MESSAGES
                  ;==========================================================
0748 18           MES0:    DB      24
0749 20202020              DW      '  ','  '
074D 2020504C              DW      '  ','PL'
0751 45415345              DW      'EA','SE'
0755 20574149              DW      ' W','AI'
0759 54202020              DW      'T ','  '
075D 20202020              DW      '  ','  '
                  ;
0761 18           MES1     DB      24
0762 20202020              DW      '  ','  '
0766 20205359              DW      '  ','SY'
076A 5354454D              DW      'ST','EM'
076E 20524541              DW      ' R','EA'
0772 44592020              DW      'DY','  '
0776 20202020              DW      '  ','  '
                  ;
077A 18           MES2     DB      24
077B 20202053              DW      '  ',' S'
077F 59535445              DW      'YS','TE'
0783 4D204341              DW      'M ','CA'
0787 4C494252              DW      'LI','BR'
078B 41544544              DW      'AT','ED'
078F 20202020              DW      '  ','  '
                  ;
0793 18           MES3:    DB      24
0794 20205359              DW      '  ','SY'
0798 5354454D              DW      'ST','EM'
079C 20434F4D              DW      ' C','OM'
07A0 4D554E49              DW      'MU','NI'
07A4 43415449              DW      'CA','TI'
07A8 4E472020              DW      'NG','  '
                  ;
07AC 18           MES4:    DB      24
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
 07AD 20524543              DW       ' R','EC'
 07B1 45495649              DW       'EI','VI'
 07B5 4E472054              DW       'NG',' T'
 07B9 454D5045              DW       'EM','PE'
 07BD 52415455              DW       'RA','TU'
 07C1 52455320              DW       'RE','S '
                    ;
 07C5 18            MES5:   DB       24
 07C6 52455452              DW       'RE','TR'
 07CA 414E534D              DW       'AN','SM'
 07CE 49545449              DW       'IT','TI'
 07D2 4E472054              DW       'NG',' T'
 07D6 4F205458              DW       'O ','TX'
 07DA 2D313030              DW       '-1','00'
                    ;
 07DE 18            MES6    DB       24
 07DF 20205458              DW       '  ','TX'
 07E3 2D313030              DW       '-1','00'
 07E7 2053454C              DW       ' S','EL'
 07EB 462D5348              DW       'F-','SH'
 07EF 5554444F              DW       'UT','DO'
 07F3 574E2020              DW       'WN','  '
                    ;
 07F7 18            MES7    DB       24
 07F8 20204E45              DW       '  ','NE'
 07FC 45442052              DW       'ED',' R'
 0800 45545241              DW       'ET','RA'
 0804 4E534D49              DW       'NS','MI'
 0808 5353494F              DW       'SS','IO'
 080C 4E202020              DW       'N ','  '
                    ;
 0810 18            MES8:   DB       24
 0811 20205348              DW       '  ','SH'
 0815 5554444F              DW       'UT','DO'
 0819 574E2053              DW       'WN',' S'
 081D 454E5420              DW       'EN','T '
 0821 42592044              DW       'BY',' D'
 0825 45432020              DW       'EC','  '
                    ;
 0829 18            MES9:   DB       24
 082A 20202052              DW       '  ',' R'
 082E 45434549              DW       'EC','EI'
 0832 56494E47              DW       'VI','NG'
 0836 2043414C              DW       ' C','AL'
 083A 20444154              DW       ' D','AT'
 083E 41202020              DW       'A ','  '
                    ;
 0842 18            MES10:  DB       24
 0843 20202020              DW       '  ','  '
 0847 54582D31              DW       'TX','-1'
 084B 30302055              DW       '00',' U'
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

```
084F 4E4C4F41              DW      'NL','OA'
0853 44454420              DW      'DE','D '
0857 20202020              DW      '  ','  '
                   ;
085B 18            MES11:  DB      24
085C 20202020              DW      '  ','  '
0860 2054582D              DW      ' T','X-'
0864 31303020              DW      '10','0 '
0868 4C4F4144              DW      'LO','AD'
086C 45442020              DW      'ED','  '
0870 20202020              DW      '  ','  '
                   ;
0874 18            MES12:  DB      24
0875 4E4F2041              DW      'NO',' A'
0879 434B4E4F              DW      'CK','NO'
087D 574C4544              DW      'WL','ED'
0881 47452052              DW      'GE',' R'
0885 45434549              DW      'EC','EI'
0889 56454420              DW      'VE','D '
                   ;
088D 18            MES13:  DB      24
088E 20205245              DW      '  ','RE'
0892 584D4954              DW      'XM','IT'
0896 2054494D              DW      ' T','IM'
089A 45522052              DW      'ER',' R'
089E 554E4F55              DW      'UN','OU'
08A2 54202020              DW      'T ','  '
                   ;
                   ;
0000                       END
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM

----- SYMBOL TABLE -----

	0000	ES	00AC	MOREDAT	05B6
	0000	ETO	00A9	MOVPNT	04F2
ACC	00E0	F0	00D5	MOVPNTR	05A5
AGAIN	0476	FINISH1	01F2	NAMSTAT	004E
ALLPRB	0320	FOURTH	0595	NEWSEQ	0685
ALPHA1	0330	GETTXSN	02E6	NOCARY1	0573
B	00F0	GO	0006	NOCARY2	04E4
BADCHAR	0002	GOODSEQ	02D7	NOCARY3	065D
BADDATA	0001	GROUP1	01AE	NOCMD	0491
BADSEQ	0433	GROUP2	01C6	NODATA	0271
BADSEQ1	043C	GROUP3	01DC	NODATA1	0279
BETA1	0380	HOLDBIT	0017	NORXMIT	000B
BLOCK	0410	HPTR	0014	NORXMT	000C
CAGAIN	051A	INIT	0108	NOSAVE	0016
CALBIT	000E	INITGO	0049	NOTENS	0593
CALBIT2	000F	INITJMP	0100	NOTFIN	0299
CALBLK	049E	INREG	0009	NOTFIN1	02A1
CALCHK	0285	KILL	0466	NXTLN	072C
CALOUT	0614	LASTCMD	000B	ODDPAR1	02C0
CALPRB	0300	LCD	0738	ODDPAR2	0532
CDATA	0010	LCDSTS	0094	ODDPAR3	04AF
CENTRAL	0172	LED	06A8	OFFSET	0012
CFINISH	0517	LED2	06D9	OUTREG	0008
CHKSUMH	0016	LED3	06AD	P	00D0
CHKSUML	0015	LED4	06C0	P1	0090
CINIT	0608	LEDOFF	0696	PARITY1	02AC
CINIT2	036C	LEDOFF1	069E	PARITY2	0527
CLEAR	015E	LEDTA	0100	PARITY3	04A4
CLRLCD	06FF	LINE	0013	PMLCD	071B
CMDCNT	0003	LLCHEK1	0552	PMLCD1	0722
CMDDONE	028E	LLIMI	0030	PRBSTS	0310
CNTR	0019	LOAD	004C	PRESCMD	000A
COMP1	067C	LOAD1	05C3	PRESRPY	000D
COMP2	067F	LOOP	05D9	PROBE	0011
COMP3	0682	LPTR	0013	PSW	00D0
COMPARE	0676	LU	000D	RB8	009A
COUNT1	0025	MES0	0748	RECEIVE	02A6
COUNT2	0026	MES1	0761	RECTMP	0045
COUNT3	0027	MES10	0842	REINIT	04FE
COUT	0664	MES11	085B	REXMIT	0052
CSHRCVD	0009	MES12	0874	RI	0098
CSLRCVD	0008	MES13	088D	RITE1	02F8
DATA	04D9	MES2	077A	RITERPY	02F1
DECOFF	0019	MES3	0793	RPY0	02DB
DECSNO	0027	MES4	07AC	RPY1A	0304
DIGIT	0010	MES5	07C5	RPY1B	0318
DONE	0044	MES6	07DE	RPY1C	032E
DPH	0083	MES7	07F7	RPY2	0353
DPL	0082	MES8	0810	RPY2A	037E
EA	00AF	MES9	0829	RPY2OK	035F
EAT	0028	MODE	0093	RPY3	038A

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: DECD.ASM
----- SYMBOL TABLE ----

RPY3OK	0399
RPY4	03B5
RPY4OK	03C1
RPY5	03E9
RPY6	0417
RPYCNT	0003
RS	0096
RXCNT	000F
RXMIT	043F
RXMIT1	0447
RXMIT2	044E
RXMLIM	0004
SBUF	0099
SCON	0098
SECOND	05A0
SENDCAL	0646
SENDCMD	03F8
SENDCS1	0623
SENDCSH	0012
SENDCSL	0011
SEQCHEK	02CC
SEQNUM	0024
SEQSTRT	0003
SERIAL	0258
SERJMP	0105
SERRET	0481
SERRET1	048A
SHUTDWN	0053
SNSAME	0015
SOFTIME	000E
SP	0081
STAT1	000B
STAT2	001B
STAT3	002B
STATUS	0005
STATUS1	0020
STATUS2	0021
STATUS3	0022
STATUS4	0023
STORAGE	0200
TAGAIN	0601
TB8	009B
TDATA	0568
TEMPBIT	0004
TEMPBLK	0521
TEMPS	0007
TFINISH	05FE
TH1	008D
THIRD	0598
TI	0099
TIME2	021A
TIME3	022D
TIME4	0238
TIME5	0241
TIME6	0257
TIMEJMP	0102
TIMERS	01FA
TIMESET	0670
TINIT	03CE
TJMP	02D4
TJMP0	02E3
TJMP1	0344
TJMP2	0350
TJMP7	042F
TMOD	0089
TMPJMP1	02C6
TMPJMP2	02C9
TR0	008C
TR1	008E
TSTART	0018
TXCS	053A
TXCS1	04B7
TXCSH	0547
TXCSH1	04C4
TXREADY	000A
TXSEQNO	0014
TXSUMH	0018
TXSUML	0017
ULCHEK1	055C
ULIMI	0039
UNLOAD	0055
UPDATE1	0001
UPDATE2	0003
UPDATE3	002A
WAIT1	0175
WAIT2	0178
WRONG	02FF
XMITBIT	0000
XMITOK	068B
XMITTMP	0054
XOK	0693
XPECTED	000C

APPENDIX F

THERMOMETRY SUBSYSTEM PROGRAM LISTING

This appendix contains the source code listing for the thermometry subsystem program entitled TX100D.ASM. Brief descriptions of the variable names encountered in the code are presented in the program preface. In addition, several comments concerning program development are located in the Appendix E introduction.

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
;
;=============================================================
;TX100 VERSION D
;
;--THIS PROGRAM IS DESIGNED TO INTERACT WITH
;  THE DEC SIMULATION PROGRAM ENTITLED
;  DECD.ASM.  THE CODE CONTAINED IN THE
;  "RESET CALIBRATION" AND THE "DISPLAY
;  UPDATE" SECTIONS WAS WRITTEN BY S. FOSTER
;  AND IS PART OF HIS ORIGINAL TX-100 PROGRAM.
;  THE MAJORITY OF THAT PROGRAM HAS BEEN MOVED
;  TO ANOTHER FILE ENTITLED ROM.ASM.
;
;--BECAUSE TX100D.ASM CONTAINS CODE FROM TWO
;  SEPARATE PROGRAMS, BYTE AND BIT ALLOCATIONS
;  ARE DIVIDED INTO TWO CATEGORIES: THOSE
;  PERTAINING TO S. FOSTER'S TX-100 PROGRAM,
;  AND THOSE PERTAINING TO THE COMMUNICATION
;  PROTOCOL PROGRAM.  DESCRIPTIONS OF THE VAR-
;  IABLES USED IN THE COMMUNICATION PROGRAM
;  ARE PRESENTED BELOW.
;
;=============================================================
;
;=============================================================
;DESCRIPTION OF BYTE VARIABLES
;
;  OUTREG - STORES THE CONSTANT, RPYCNT, WHICH
;  IS DECREMENTED WHENEVER ANOTHER CHARACTER
;  IN A REDUNDANT REPLY SET IS TRANSMITTED.
;
;  INREG - STORES THE CONSTANT, CMDCNT, WHICH
;  IS DECREMENTED WHENEVER ANOTHER CHARACTER
;  IN A REDUNDANT COMMAND SET IS RECEIVED.
;
;  LASTRPY - STORES THE REPLY LAST TRANSMIT-
;  TED TO THE DEC.  SAVED IN CASE THE TX-100
;  REQUESTS A RETRANSMISSION.
;
;  PRESRPY - STORES THE REPLY PRESENTLY BEING
;  TRANSMITTED TO THE DEC.  SAVED TO ALLOW FOR
;  REPLY DUPLICATION IN THE REDUNDANT SET.
;
;  PRESCMD - STORES THE COMMAND PRESENTLY
;  BEING RECEIVED FROM THE DEC.  SAVED TO EN-
;  SURE THAT ALL THREE COMMANDS ARE IDENTICAL.
;
;  RXCNT - STORES THE CONSTANT, RXMLIM, WHICH
;  IS DECREMENTED WHENEVER ANOTHER RETRANS-
;  MISSION REQUEST IS SENT TO THE DEC.
;
```

```
;  DIGIT - STORES A NUMBER, 1-4, CORRESPOND-
;  ING TO THE TEMPERATURE DIGIT CURRENTLY
;  BEING SENT TO THE DEC (4 DIGITS PER TEMP-
;  ERATURE PROBE).
;
;  PROBE - STORES A NUMBER, 1-16, CORRESPOND-
;  ING TO THE PROBE THAT TEMPERATURES ARE
;  CURRENTLY BEING SENT FOR (16 PROBES).
;
;  OFFSET - STORES THE OFFSET FROM THE BASE
;  ADDRESS OF THE 64-BYTE TEMPERATURE ARRAY.
;  USED TO CORRECTLY SEND THE TEMPERATURE
;  DATA.  OBTAINED BY MULTIPLYING THE PROBE
;  NUMBER BY 4.
;
;  CHKSUML - STORES THE LOW BYTE OF THE GEN-
;  ERATED CHECKSUM.
;
;  CHKSUMH - STORES THE HIGH BYTE OF THE GEN-
;  ERATED CHECKSUM.
;
;  DECSUML - STORES THE LOW BYTE OF THE CHECK-
;  SUM RECEIVED FROM THE DEC.
;
;  DECSUMH - STORES THE HIGH BYTE OF THE
;  CHECKSUM RECEIVED FROM THE DEC.
;
;  EST - STORES A CONSTANT THAT IS THEN
;  DECREMENTED TO CREATE THE PROPER REDUNDANT
;  SET TIMER PERIOD.  MAY BE REFERENCED AS THE
;  EXPECTED SET TIMER IN THIS PROGRAM.
;
;  RT - STORES A CONSTANT THAT IS THEN DECRE-
;  MENTED TO CREATE THE PROPER EXPECTED RE-
;  TRANSMISSION TIMER PERIOD.
;
;  VT - STORES A CONSTANT THAT IS THEN DECRE-
;  MENTED TO CREATE THE PROPER LINE VIABILITY
;  TIMER PERIOD.
;
;============================================
;
;============================================
;DESCRIPTION OF BIT VARIABLES
;
;  STATUS1,STATUS2 - RESERVED BIT PLACES.
;
;  IDSTAT - STORES THE IDENTIFICATION/STATUS
;  REPLY.  ONCE STATUSES ARE DEFINED, INDIVI-
;  DUAL BITS CAN BE TOGGLED TO PRODUCE THE
;  CORRECT STATUS WORD.
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
;
;  XMITBIT - USED IN THE XMITOK SUBROUTINE.
;  SET WHEN A TRANSMISSION IS CURRENTLY IN
;  PROGRESS.  CLEARED WHEN THE TRANSMISSION IS
;  FINISHED.
;
;  BADCHAR - SET WHEN A BAD CHARACTER HAS BEEN
;  RECEIVED FROM THE DEC.  CLEAR FOR GOOD
;  TRANSMISSIONS.
;
;  SEQSTRT - SET WHEN A REDUNDANT COMMAND SET
;  HAS ALREADY BEEN STARTED.  INDICATES THAT
;  COMMAND DUPLICATION MUST BE CHECKED.  CLEAR
;  UNTIL THE FIRST BYTE IN THE SET IS RE-
;  CEIVED.
;
;  LINE - SET WHEN THE LAST REPLY TRANSMITTED
;  WAS A SINGLE LINE.  CLEAR WHEN THE LAST
;  REPLY WAS A DATA BLOCK.
;
;  DONESEQ - SET WHEN THE DONE REPLY HAS BEEN
;  SENT TO THE DEC.  INDICATES THAT THE COM-
;  MUNICATION LINK IS ESTABLISHED.
;
;  TEMPBIT - SET WHEN THE TEMPERATURE COMMAND
;  IS RECEIVED FROM THE DEC.  INDICATES THAT
;  A FULL TEMPERATURE DATA BLOCK MUST BE
;  TRANSMITTED TO THE DEC.  CLEARED WHEN THE
;  TEMPERATURE DATA BLOCK IS FINISHED.
;
;  TDATA - SET TO INDICATE THAT THE TEMPERA-
;  TURE DATA BLOCK IS NOT FINISHED TRANSMIT-
;  TING.  CLEARED AFTER THE LAST CHECKSUM BYTE
;  HAS BEEN SENT.
;
;  SENDCSL - SET WHEN THE LOW BYTE OF THE GEN-
;  ERATED CHECKSUM NEEDS TO BE SENT.  CLEARED
;  AFTER IT IS SENT.
;
;  SENDCSH - SET WHEN THE HIGH BYTE OF THE
;  GENERATED CHECKSUM NEEDS TO BE SENT.
;  CLEARED AFTER IT IS SENT.
;
;  TXOFF - SET WHEN THE TX-100 NEEDS TO BE
;  SHUTDOWN.  AFTER THE SHUTDOWN COMMAND HAS
;  BEEN SENT TO THE DEC, THE TX-100 CEASES ALL
;  ACTIVITY.
;
;  READY - SET AFTER THE LAST NAME/STATUS
;  COMMAND HAS BEEN RECEIVED FROM THE DEC.
;  INDICATES THAT THE TX-100 IS READY TO BE
```

```
;   CALIBRATED.
;
;   PAUSE - SET AFTER THE "SYSTEM READY" REPLY
;   (#1BH) HAS BEEN SENT TO THE DEC.  INDI-
;   CATES THAT THE NEXT IDENTIFICATION/STATUS
;   REPLY SHOULD BE HELD UNTIL AFTER CALIBRA-
;   TION IS COMPLETE.
;
;   STATUS3 - MORE RESERVED BIT PLACES.
;
;   CALBIT - SET WHEN THE LOAD COMMAND IS RE-
;   CEIVED FROM THE DEC.  INDICATES THAT A FULL
;   CALIBRATION DATA BLOCK MUST BE TRANSMITTED
;   TO THE DEC.  CLEARED WHEN THE CALIBRATION
;   DATA BLOCK IS FINISHED.
;
;   CDATA - SET TO INDICATE THAT THE CALIBRA-
;   TION DATA BLOCK IS NOT FINISHED TRANSMIT-
;   TING.  CLEARED AFTER THE LAST CHECKSUM BYTE
;   HAS BEEN SENT.
;
;   CALSENT - SET AFTER THE FIRST SECTION OF
;   CALIBRATION DATA HAS BEEN SENT TO THE DEC.
;   INDICATES THAT THE SECOND SECTION NEEDS TO
;   BE TRANSMITTED.
;
;   CALIN - SET WHEN THE UNLOAD COMMAND IS
;   RECEIVED FROM DEC.  INDICATES THAT A CAL-
;   BRATION DATA BLOCK IS BEING PROCESSED.
;
;   CALRCVD - SET WHEN THE FIRST SECTION OF
;   CALIBRATION DATA HAS BEEN RECEIVED FROM THE
;   DEC.  INDICATES THAT THE SECOND SECTION
;   STILL NEEDS TO BE PROCESSED.
;
;   CSLRCVD - SET WHEN THE LOW BYTE OF THE DEC
;   CHECKSUM IS EXPECTED.  CLEARED AFTER THE
;   FULL DATA BLOCK HAS BEEN RECEIVED.
;
;   CSHRCVD - SET WHEN THE HIGH BYTE OF THE
;   DEC CHECKSUM IS EXPECTED.  CLEARED AFTER
;   THE FULL DATA BLOCK HAS BEEN RECEIVED.
;
;   BADDATA - USED DURING DATA BLOCK TRANS-
;   MISSIONS.  SET WHEN BAD CALIBRATION DATA
;   HAVE BEEN RECEIVED FROM THE DEC.
;
;   LOADED - SET AFTER THE FULL CALIBRATION
;   DATA BLOCK HAS BEEN RECEIVED.  INDICATES
;   THAT THE DEC HAS RELOADED THE TX-100 WITH
;   THE CALIBRATION DATA.
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
;
;  NORXMIT - SET WHEN THE CALIBRATION DATA
;  BLOCK IS EXPECTED.  DELAYS RETRANSMISSION
;  REQUESTS UNTIL AFTER THE FULL BLOCK HAS
;  BEEN RECEIVED.
;
;  CALRX - SET WHEN THE CALIBRATION DATA BLOCK
;  IS BEING TRANSMITTED TO THE DEC.  CLEARED
;  WHEN THE TEMPERATURE DATA BLOCK IS BEING
;  TRANSMITTED TO THE DEC.  IF A RETRANSMIS-
;  SION REQUEST IS RECEIVED, THE TX-100 KNOWS
;  WHICH BLOCK TO RETRANSMIT.
;
;  STATUS4 - MORE RESERVED BIT PLACES.
;
;  SEQNUM - STORES THE TX-100 SEQUENCE NUMBER
;  IN THE MOST SIGNIFICANT BIT PLACE.  LOG-
;  ICALLY ORED WITH THE REPLY TO BE TRANSMIT-
;  TED IN ORDER TO APPEND THE SEQUENCE BIT.
;
;  DECSNO - SET WHEN THE DEC SEQUENCE NUMBER
;  IS EQUAL TO ONE.  CLEARED WHEN THE THE DEC
;  SEQUENCE NUMBER IS EQUAL TO ZERO.
;
;  SNSAME - SET WHEN THE DEC SEQUENCE NUMBER
;  EQUALS THE TX-100 SEQUENCE NUMBER.
;  CLEARED WHEN NOT EQUAL.
;
;  NOSAVE - SET WHEN A RETRANSMISSION REQUEST
;  IS SENT TO THE DEC.  INDICATES THAT THE
;  REQUEST SHOULD NOT BE STORED IN THE LASTRPY
;  REGISTER.  THIS IS NECESSARY TO MAINTAIN
;  SEQUENCE NUMBER CONTROL.
;
;  TXSEQNO - THE ACTUAL TX-100 SEQUENCE NUM-
;  BER.  RESIDES IN THE MSB OF SEQNUM.
;
;  T1START - SET TO ACTIVATE THE REDUNDANT SET
;  TIMER.  CLEARED WHEN DEACTIVATED.
;
;  T2START - SET TO ACTIVATE THE EXPECTED RE-
;  TRANSMISSION TIMER.  CLEARED WHEN DEACTI-
;  VATED.
;
;  T3START - SET TO ACTIVATE THE LINE
;  VIABILITY TIMER.  CLEARED WHEN DEACTIVATED.
;===============================================
;
;===============================================
;MEMORY ALLOCATION-TX100
;===============================================
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
0008            XEXP    EQU     $08
0009            XMAN    EQU     $09,$0B
000C            YEXP    EQU     $0C
000D            YMAN    EQU     $0D,$0F
0010            EXP1    EQU     $10
0011            MAN1    EQU     $11,$13
0014            CNTR    EQU     $14
0015            EXP2    EQU     $15
0016            MAN2    EQU     $16,$18
0019            TMP     EQU     $19
001A            CHNUM   EQU     $1A
001B            VREXP   EQU     $1B
001C            VRMAN   EQU     $1C,$1E
001F            UPRAT   EQU     $1F
0027            N       EQU     $27
0028            NMAN    EQU     $28,2A
002B            COUNT1  EQU     $2B
002C            COUNT2  EQU     $2C
002D            COUNT3  EQU     $2D
                ;
                ;============================================
                ;MEMORY ALLOCATION-COMMUNICATIONS
                ;============================================
0030            OUTREG  EQU     30H
0031            INREG   EQU     31H
0032            LASTRPY EQU     32H
0033            PRESRPY EQU     33H
0034            PRESCMD EQU     34H
0035            RXCNT   EQU     35H
0036            DIGIT   EQU     36H
0037            PROBE   EQU     37H
0038            OFFSET  EQU     38H
0039            CHKSUML EQU     39H
003A            CHKSUMH EQU     3AH
003B            DECSUML EQU     3BH
003C            DECSUMH EQU     3CH
003D            EST     EQU     3DH
003E            RT      EQU     3EH
003F            VT      EQU     3FH
                ;
                ;============================================
                ;BIT ALLOCATION-TX100
                ;============================================
0020            BITS    EQU     $20
                ;
0000            XSGN    EQU     $00
0001            YSGN    EQU     $01
0002            SGN1    EQU     $02
0003            SGN2    EQU     $03
0004            SIGN    EQU     $04
0005            UPDATE  EQU     $05
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
0008            F1       EQU     $08
0009            F2       EQU     $09
000A            VRSGN    EQU     $0A
000B            CALFG    EQU     $0B
000C            RND      EQU     $0C
000D            ORNG     EQU     $0D
                ;
                ;==========================================
                ;BIT ALLOCATION-COMMUNICATIONS
                ;==========================================
0022            STATUS1  EQU     22H
0023            STATUS2  EQU     23H
0024            STATUS3  EQU     24H
0025            STATUS4  EQU     25H
0026            IDSTAT   EQU     26H
002E            SEQNUM   EQU     2EH
                ;
0010            XMITBIT  EQU     STATUS1.0
0011            BADCHAR  EQU     STATUS1.1
0012            SEQSTRT  EQU     STATUS1.2
0013            LINE     EQU     STATUS1.3
0014            DONESEQ  EQU     STATUS1.4
0015            TEMPBIT  EQU     STATUS1.5
0016            TDATA    EQU     STATUS1.6
0017            SENDCSL  EQU     STATUS1.7
                ;
0018            SENDCSH  EQU     STATUS2.0
0019            TXOFF    EQU     STATUS2.1
001A            READY    EQU     STATUS2.2
001B            PAUSE    EQU     STATUS2.3
001C            CALBIT   EQU     STATUS2.4
001D            CDATA    EQU     STATUS2.5
001E            CALSENT  EQU     STATUS2.6
001F            CALIN    EQU     STATUS2.7
                ;
0020            CALRCVD  EQU     STATUS3.0
0021            CSLRCVD  EQU     STATUS3.1
0022            CSHRCVD  EQU     STATUS3.2
0023            BADDATA  EQU     STATUS3.3
0024            LOADED   EQU     STATUS3.4
0025            NORXMIT  EQU     STATUS3.5
0026            CALRX    EQU     STATUS3.6
0027            DECSNO   EQU     STATUS3.7
                ;
0028            SNSAME   EQU     STATUS4.0
0029            NOSAVE   EQU     STATUS4.1
002A            T1START  EQU     STATUS4.2
002B            T2START  EQU     STATUS4.3
002C            T3START  EQU     STATUS4.4
                ;
0077            TXSEQNO  EQU     SEQNUM.7
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
                    ;
                    ;===============================================
                    ;PORT EQUIVALENTS
                    ;===============================================
0091                SW      EQU     P1.1
0093                MODE    EQU     P1.3
0094                LCDSTS  EQU     P1.4
0095                CLRSW   EQU     P1.5
0096                RS      EQU     P1.6
00D2                OV      EQU     PSW.2
                    ;
                    ;===============================================
                    ;COMMAND-REPLY EQUIVALENTS
                    ;===============================================
004E                NAMSTAT EQU     4EH
004C                LOAD    EQU     4CH
0055                UNLOAD  EQU     55H
0049                INITGO  EQU     49H
0044                DONE    EQU     44H
0054                XMITTMP EQU     54H
0045                RECTMP  EQU     45H
0052                REXMIT  EQU     52H
0053                SHUTDWN EQU     53H
                    ;
                    ;===============================================
                    ;CONSTANTS
                    ;===============================================
0003                CMDCNT  EQU     3
0003                RPYCNT  EQU     3
0004                RXMLIM  EQU     4
0007                UPDATE1 EQU     7
002A                UPDATE2 EQU     42
0000                UPDATE3 EQU     0
                    ;
0000                ===============================================
                    ;RAM ADDRESSES-TX100
                    ;===============================================
0100                CALPRB  EQU     $0100,$010F
0110                PRBSTS  EQU     $0110,$011F
0200                ALPHA1  EQU     $0200,$024F
0250                BETA1   EQU     $0250,$02FF
0300                LEDTA   EQU     $0300,$033F
0340                TEMP    EQU     $0340,$038F
                    ;
                    ;===============================================
                    ;RAM ADDRESSES-COMMUNICATIONS
                    ;===============================================
0120                ALLPRB  EQU     $0120,$012F
                    ;
                    ;===============================================
                    ;SUBROUTINE ADDRESSES
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
                     ;==============================================
 16D2                ADD      EQU      $16D2
 107F                ATD      EQU      $107F
 1067                ATDPR    EQU      $1067
 18D0                BCD      EQU      $18D0
 1215                CAL      EQU      $1215
 1000                CLRLCD   EQU      $1000
 17E6                DIV      EQU      $17E6
 18F8                FIX      EQU      $18F8
 1931                FLT      EQU      $1931
 1057                LCD      EQU      $1057
 1544                LED      EQU      $1544
 1454                MOVX     EQU      $1454
 15C7                MUL      EQU      $15C7
 102D                NXTLN    EQU      $102D
 101C                PMLCD    EQU      $101C
 151D                POP      EQU      $151D
 14F4                PUSH     EQU      $14F4
 19CD                TLSF     EQU      $19CD
 19D4                VLSF     EQU      $19D4
 10AF                WAIT     EQU      $10AF
 11A7                WRITE    EQU      $11A7
 10C2                WT3MS    EQU      $10C2
 1989                XRCL1    EQU      $1989
 19AB                XSTO1    EQU      $19AB
 1486                YMOV     EQU      $1486
 1475                YMOVVR   EQU      $1475
                     ;
                     ;
                     ;==============================================
                     ;MAIN ROUTINE
                     ;==============================================
 0100                         ORG      $0100
                     ;
                     ;==============================================
                     ;VECTOR JUMPS
                     ;        0000-020100
                     ;        000B-020102
                     ;        0023-020105
                     ;==============================================
 0100 8006           INITJMP:SJMP      INIT
                     ;
 0102 0205BB         TIMEJMP:LJMP      TIMERS
                     ;
 0105 020618         SERJMP: LJMP      SERIAL
                     ;
                     ;==============================================
                     ;8031 INITIALIZATIONS
                     ;==============================================
 0108 758140         INIT     MOV      SP,#$40
                     ;
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
                        ;TIMER SET-UP
 010B 758921                    MOV     TMOD,   #21H
 010E 758DE8                    MOV     TH1,    #0E8H
 0111 D28C                      SETB    TR0
 0113 D28E                      SETB    TR1
                        ;
                        ;SERIAL SET-UP
 0115 7598D0                    MOV     SCON,   #0D0H
                        ;
                        ;INTERRUPT SET-UP
 0118 D2A9                      SETB    ET0
 011A D2AC                      SETB    ES
                        ;
                        ;==============================================
                        ;BIT INITIALIZATIONS
                        ;==============================================
 011C C205                      CLR     UPDATE
                        ;
 011E C219                      CLR     TXOFF
 0120 C215                      CLR     TEMPBIT
 0122 C210                      CLR     XMITBIT
 0124 C216                      CLR     TDATA
 0126 C214                      CLR     DONESEQ
 0128 C211                      CLR     BADCHAR
 012A C212                      CLR     SEQSTRT
 012C C218                      CLR     SENDCSH
 012E C217                      CLR     SENDCSL
 0130 C223                      CLR     BADDATA
 0132 C21B                      CLR     PAUSE
 0134 C21A                      CLR     READY
 0136 C225                      CLR     NORXMIT
 0138 C224                      CLR     LOADED
 013A C21C                      CLR     CALBIT
 013C C21D                      CLR     CDATA
 013E C21E                      CLR     CALSENT
 0140 C21F                      CLR     CALIN
 0142 C220                      CLR     CALRCVD
 0144 C229                      CLR     NOSAVE
 0146 C22A                      CLR     T1START
 0148 C22B                      CLR     T2START
 014A C22C                      CLR     T3START
                        ;
                        ;==============================================
                        ;STORAGE INITIALIZATION
                        ;==============================================
 014C 752E00                    MOV     SEQNUM, #00H
 014F D277                      SETB    TXSEQNO
 0151 120993                    CALL    NEWSEQ
 0154 753D07                    MOV     EST,    #UPDATE1
 0157 753E2A                    MOV     RT,     #UPDATE2
 015A 753F00                    MOV     VT,     #UPDATE3
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
 015D 753003              MOV     OUTREG, #RPYCNT
 0160 75260B              MOV     IDSTAT, #0BH
 0163 753500              MOV     RXCNT,  #00H
                  ;
 0166 752B10              MOV     COUNT1, #10H
 0169 7400                MOV     A,      #00H
 016B 900120              MOV     DPTR,   #ALLPRB
 016E F0          INIT4:  MOVX    @DPTR,  A
 016F 0582                INC     DPL
 0171 D52BFA              DJNZ    COUNT1, INIT4
                  ;
 0174 7840                MOV     R0,#$40
 0176 740F                MOV     A,#$0F
 0178 900300              MOV     DPTR,#LEDTA
 017B F0          INIT3   MOVX    @DPTR,A
 017C 0582                INC     DPL
 017E D8FB                DJNZ    R0,INIT3
                  ;
 0180 D2AF                SETB    EA
                  ;
                  ;==============================================
                  ;RESET CALIBRATION
                  ;==============================================
 0182 C200                CLR     XSGN
 0184 7508FF              MOV     XEXP,#$FF
 0187 750B2A              MOV     XMAN+2,#$2A
 018A 750AD8              MOV     XMAN+1,#$D8
 018D 7509B0              MOV     XMAN,#$B0
 0190 7800                MOV     R0,#$00
 0192 900200      INT5    MOV     DPTR,#ALPHA1
 0195 E8                  MOV     A,R0
 0196 121454              CALL    MOVX
 0199 08                  INC     R0
 019A B810F5              CJNE    R0,#$10,INT5
 019D C200                CLR     XSGN
 019F 750800              MOV     XEXP,#$00
 01A2 750B30              MOV     XMAN+2,#$30
 01A5 750A65              MOV     XMAN+1,#$65
 01A8 75097F              MOV     XMAN,#$7F
 01AB 7800                MOV     R0,#$00
 01AD 900250      INT2    MOV     DPTR,#BETA1
 01B0 E8                  MOV     A,R0
 01B1 121454              CALL    MOVX
 01B4 08                  INC     R0
 01B5 B810F5              CJNE    R0,#$10,INT2
 01B8 C20A                CLR     VRSGN
 01BA 751B80              MOV     VREXP,#$80
 01BD 751C00              MOV     VRMAN,#$00
 01C0 751D00              MOV     VRMAN+1,#$00
 01C3 751E00              MOV     VRMAN+2,#$00
 01C6 751F78              MOV     UPRAT,#120
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
 01C9 752710               MOV      N,#16
 01CC 752A10               MOV      NMAN+2,#$10
 01CF 752900               MOV      NMAN+1,#$00
 01D2 752800               MOV      NMAN,#$00
                  ;
                  ;===============================================
                  ;START OF OPERATION
                  ;===============================================
 01D5 121544               CALL     LED
 01D8 121000               CALL     CLRLCD
 01DB 9004B1               MOV      DPTR,#MES1
 01DE 12101C               CALL     PMLCD
 01E1 12102D               CALL     NXTLN
 01E4 9004C6               MOV      DPTR,#MES2
 01E7 12101C               CALL     PMLCD
 01EA 751A00               MOV      CHNUM,#$00
 01ED 909000               MOV      DPTR,#$9000
 01F0 F0                   MOVX     @DPTR,A
 01F1 1210AF               CALL     WAIT
 01F4 75261B               MOV      IDSTAT, #1BH
                  ;
 01F7 121000      RDY      CALL     CLRLCD
 01FA 9004DF               MOV      DPTR,#MES3
 01FD 12101C               CALL     PMLCD
 0200 7443                 MOV      A,#'C'
 0202 121057      RDY1     CALL     LCD
                  ;
 0205 301AFD               JNB      READY,  $
 0208 12102D      RDY3     CALL     NXTLN
 020B 900510               MOV      DPTR,#MES4
 020E 12101C               CALL     PMLCD
                  ;
 0211 D205                 SETB     UPDATE
 0213 120300      RDY2     CALL     SWITCH
 0216 B40A28               CJNE     A,#$0A,RDY4
 0219 121215               CALL     CAL
 021C 752B10               MOV      COUNT1, #10H
 021F 752C00               MOV      COUNT2, #00H
 0222 758301               MOV      DPH,     #01H
 0225 7400        CALLOOP:MOV       A,       #00H
 0227 252C                 ADD      A,       COUNT2
 0229 F582                 MOV      DPL,     A
 022B E0                   MOVX     A,       @DPTR
 022C B4FF0B               CJNE     A,       #0FFH,          ENDLOOP
 022F F5F0                 MOV      B,       A
 0231 7420                 MOV      A,       #20H
 0233 252C                 ADD      A,       COUNT2
 0235 F582                 MOV      DPL,     A
 0237 E5F0                 MOV      A,       B
 0239 F0                   MOVX     @DPTR,   A
 023A 052C        ENDLOOP:INC       COUNT2
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
023C D52BE6              DJNZ    COUNT1, CALLOOP
023F 80B6       TJ1:     SJMP    RDY
0241 B40EFB     RDY4     CJNE    A,#$0E,TJ1
0244 121000              CALL    CLRLCD
0247 900529              MOV     DPTR,    #MES5
024A 12101C              CALL    PMLCD
024D 75262B              MOV     IDSTAT, #2BH
0250 C21B                CLR     PAUSE
0252 E526                MOV     A,       IDSTAT
0254 452E                ORL     A,       SEQNUM
0256 F533                MOV     PRESRPY,A
0258 A2D0                MOV     C,       P
025A 120999              CALL    XMITOK
025D 929B                MOV     TB8,     C
025F F599                MOV     SBUF,    A
0261 D22C                SETB    T3START
0263 1210AF              CALL    WAIT
0266 3014FD              JNB     DONESEQ,$
0269 C214                CLR     DONESEQ
026B 121000              CALL    CLRLCD
026E 90053E              MOV     DPTR,    #MES6
0271 12101C              CALL    PMLCD
0274 12102D              CALL    NXTLN
0277 900557              MOV     DPTR,    #MES7
027A 12101C              CALL    PMLCD
027D 120300     LOOP:    CALL    SWITCH
0280 80FB                SJMP    LOOP
                ;
                ;===============================================
                ;DISPLAY UPDATE SECTION
                ;===============================================
0300                     ORG     0300H
                ;
0300 1214F4     SWITCH   CALL    PUSH
0303 D295                SETB    CLRSW
0305 C295                CLR     CLRSW
0307 309107              JNB     SW,BKGND
030A D295       SW1      SETB    CLRSW
030C C295                CLR     CLRSW
030E 2091F9              JB      SW,SW1
0311 200506     BKGND    JB      UPDATE,BKGND10
0314 0204A3              LJMP    BKGND18
0317 0203C1     BKGND11  LJMP    BKGND1
031A E51A       BKGND10  MOV     A,CHNUM
031C B410F8              CJNE    A,#$10,BKGND11
031F 121544              CALL    LED
0322 7440                MOV     A,#%01000000
0324 121067              CALL    ATDPR
0327 C201                CLR     YSGN
0329 750C00              MOV     YEXP,#$00
032C 750F02              MOV     YMAN+2,#$02
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
 032F 750E57                 MOV     YMAN+1,#$57
 0332 750D40                 MOV     YMAN,#$40
 0335 1215C7                 CALL    MUL
 0338 C201                   CLR     YSGN
 033A 750C01                 MOV     YEXP,#$01
 033D 750F18                 MOV     YMAN+2,#$18
 0340 750E87                 MOV     YMAN+1,#$87
 0343 750D31                 MOV     YMAN,#$31
 0346 1216D2                 CALL    ADD
 0349 1219D4                 CALL    VLSF
 034C D201                   SETB    YSGN
 034E 750C01                 MOV     YEXP,#$01
 0351 750F01                 MOV     YMAN+2,#$01
 0354 750E87                 MOV     YMAN+1,#$87
 0357 750D2B                 MOV     YMAN,#$2B
 035A 1216D2                 CALL    ADD
 035D A200                   MOV     C,XSGN
 035F 920A                   MOV     VRSGN,C
 0361 85081B                 MOV     VREXP,XEXP
 0364 850B1E                 MOV     VRMAN+2,XMAN+2
 0367 850A1D                 MOV     VRMAN+1,XMAN+1
 036A 85091C                 MOV     VRMAN,XMAN
 036D 751410                 MOV     CNTR,#$10
 0370 7800                   MOV     RO,#$00
 0372 900110                 MOV     DPTR,#PRBSTS
 0375 E0            GND1     MOVX    A,@DPTR
 0376 6001                   JZ      GND2
 0378 08                     INC     RO
 0379 0582          GND2     INC     DPL
 037B D514F7                 DJNZ    CNTR,GND1
 037E E51F                   MOV     A,UPRAT
 0380 88F0                   MOV     B,RO
 0382 84                     DIV     AB
 0383 20D22F                 JB      OV,GND4
 0386 C5F0                   XCH     A,B
 0388 33                     RLC     A
 0389 98                     SUBB    A,RO
 038A 4002                   JC      GND3
 038C 05F0                   INC     B
 038E 85F027        GND3     MOV     N,B
 0391 750801                 MOV     XEXP,#$01
 0394 750B01                 MOV     XMAN+2,#$01
 0397 750A00                 MOV     XMAN+1,#$00
 039A 750900                 MOV     XMAN,#$00
 039D 750C01                 MOV     YEXP,#$01
 03A0 85F00F                 MOV     YMAN+2,B
 03A3 750E00                 MOV     YMAN+1,#$00
 03A6 750D00                 MOV     YMAN,#$00
 03A9 1217E6                 CALL    DIV
 03AC 850B2A                 MOV     NMAN+2,XMAN+2
 03AF 850A29                 MOV     NMAN+1,XMAN+1
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
03B2 850928            MOV     NMAN,XMAN
03B5 751A00   GND4     MOV     CHNUM,#$00
03B8 309103            JNB     SW,BKGND12
03BB 0204A9            LJMP    BKGND7
03BE 020311   BKGND12  LJMP    BKGND
03C1 D2E4     BKGND1   SETB    ACC.4
03C3 F5F0              MOV     B,A
03C5 1210C2            CALL    WT3MS
03C8 900110            MOV     DPTR,#PRBSTS
03CB E51A              MOV     A,CHNUM
03CD 2582              ADD     A,DPL
03CF F582              MOV     DPL,A
03D1 E4                CLR     A
03D2 300D1E            JNB     ORNG,BKGND4
03D5 F0                MOVX    @DPTR,  A
03D6 900300            MOV     DPTR,#LEDTA
03D9 E51A              MOV     A,CHNUM
03DB 23                RL      A
03DC 23                RL      A
03DD 2582              ADD     A,DPL
03DF F582              MOV     DPL,A
03E1 751404            MOV     CNTR,#$04
03E4 740F              MOV     A,#$0F
03E6 C2AC              CLR     ES
03E8 F0       BKGND3   MOVX    @DPTR,A
03E9 0582              INC     DPL
03EB D514FA            DJNZ    CNTR,BKGND3
03EE D2AC              SETB    ES
03F0 0204A1            LJMP    BKGND2
03F3 F4       BKGND4   CPL     A
03F4 F0                MOVX    @DPTR,A
03F5 7800              MOV     R0,#$00
03F7 7900              MOV     R1,#$00
03F9 7A00              MOV     R2,#$00
03FB 7B00              MOV     R3,#$00
03FD AC27              MOV     R4,N
03FF E5F0     BKGND6   MOV     A,B
0401 12107F            CALL    ATD
0404 309103            JNB     SW,BKGND13
0407 0204A9            LJMP    BKGND7
040A DCF3     BKGND13  DJNZ    R4,BKGND6
040C 850100            MOV     $00,$01
040F 850201            MOV     $01,$02
0412 850302            MOV     $02,$03
0415 C204              CLR     SIGN
0417 121931            CALL    FLT
041A C201              CLR     YSGN
041C 750C00            MOV     YEXP,#$00
041F 852A0F            MOV     YMAN+2,NMAN+2
0422 85290E            MOV     YMAN+1,NMAN+1
0425 85280D            MOV     YMAN,NMAN
```

```
0428 1215C7              CALL    MUL
042B 900200              MOV     DPTR,#ALPHA1
042E E51A                MOV     A,CHNUM
0430 121486              CALL    YMOV
0433 1215C7              CALL    MUL
0436 900250              MOV     DPTR,#BETA1
0439 E51A                MOV     A,CHNUM
043B 121486              CALL    YMOV
043E 1216D2              CALL    ADD
0441 121475              CALL    YMOVVR
0444 1216D2              CALL    ADD
0447 900340              MOV     DPTR,#TEMP
044A E51A                MOV     A,CHNUM
044C 121454              CALL    MOVX
044F 1219CD              CALL    TLSF
0452 1219AB              CALL    XSTO1
0455 C201                CLR     YSGN
0457 750C01              MOV     YEXP,#$01
045A 750D00              MOV     YMAN,#$00
045D 750E00              MOV     YMAN+1,#$00
0460 750F64              MOV     YMAN+2,#$64
0463 1215C7              CALL    MUL
0466 1218F8              CALL    FIX
0469 1218D0              CALL    BCD
046C C2D5                CLR     F0
046E 900303              MOV     DPTR,#LEDTA+3
0471 E51A                MOV     A,CHNUM
0473 23                  RL      A
0474 23                  RL      A
0475 2582                ADD     A,DPL
0477 F582                MOV     DPL,A
0479 EC          BK8     MOV     A,R4
047A 54F0                ANL     A,#$F0
047C 6005                JZ      BK12
047E C4                  SWAP    A
047F 4480                ORL     A,#$80
0481 8002                SJMP    BK11
0483 748F        BK12    MOV     A,#$8F
0485 C2AC        BK11    CLR     ES
0487 F0                  MOVX    @DPTR,A
0488 1582                DEC     DPL
048A EC          BK13    MOV     A,R4
048B 540F                ANL     A,#$0F
048D F0                  MOVX    @DPTR,A
048E 1582                DEC     DPL
0490 EB                  MOV     A,R3
0491 C4                  SWAP    A
0492 540F                ANL     A,#$0F
0494 4480                ORL     A,#$80
0496 F0                  MOVX    @DPTR,A
0497 1582                DEC     DPL
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
0499 EB                       MOV     A,R3
049A 540F                     ANL     A,#$0F
049C 4480                     ORL     A,#$80
049E F0                       MOVX    @DPTR,A
049F D2AC                     SETB    ES
04A1 051A         BKGND2      INC     CHNUM
04A3 209103       BKGND18     JB      SW,BKGND7
04A6 020311                   LJMP    BKGND
04A9 12151D       BKGND7      CALL    POP
04AC 90A000       J1          MOV     DPTR,#$A000
04AF E0                       MOVX    A,@DPTR
04B0 22                       RET
                  ;
                  ;==================================================
                  ;MESSAGES
                  ;==================================================
04B1 14           MES1        DB      20
04B2 20202020                 DW      '  ','  '
04B6 2D20504C                 DW      '- ','PL'
04BA 45415345                 DW      'EA','SE'
04BE 20574149                 DW      ' W','AI'
04C2 54202D20                 DW      'T ','- '
                  ;
04C6 18           MES2        DB      24
04C7 20535953                 DW      ' S','YS'
04CB 54454D20                 DW      'TE','M '
04CF 4F504552                 DW      'OP','ER'
04D3 4154494F                 DW      'AT','IO'
04D7 4E204348                 DW      'N ','CH'
04DB 45434B20                 DW      'EC','K '
                  ;
04DF 17           MES3        DB      23
04E0 20202020                 DW      '  ','  '
04E4 20205359                 DW      '  ','SY'
04E8 5354454D                 DW      'ST','EM'
04EC 20524541                 DW      ' R','EA'
04F0 44592020                 DW      'DY','  '
04F4 2020                     DW      '  '
04F6 DF                       DB      $DF
                  ;
04F7 18           MES4A       DB      24
04F8 4C4F4144                 DW      'LO','AD'
04FC 20202020                 DW      '  ','  '
0500 20202020                 DW      '  ','  '
0504 20202020                 DW      '  ','  '
0508 20202020                 DW      '  ','  '
050C 434F4E54                 DW      'CO','NT'
                  ;
0510 18           MES4        DB      24
0511 43414C20                 DW      'CA','L '
0515 20202020                 DW      '  ','  '
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
0519 20202020            DW      '  ','  '
051D 20202020            DW      '  ','  '
0521 20202020            DW      '  ','  '
0525 434F4E54            DW      'CO','NT'
                 ;
0529 14          MES5    DB      20
052A 20202053            DW      '  ',' S'
052E 59535445            DW      'YS','TE'
0532 4D204341            DW      'M ','CA'
0536 4C494252            DW      'LI','BR'
053A 41544544            DW      'AT','ED'
                 ;
053E 18          MES6    DB      24
053F 20202020            DW      '  ','  '
0543 2054582D            DW      ' T','X-'
0547 31303020            DW      '10','0 '
054B 53595354            DW      'SY','ST'
054F 454D2020            DW      'EM','  '
0553 20202020            DW      '  ','  '
                 ;
0557 18          MES7    DB      24
0558 20434F4D            DW      ' C','OM'
055C 4D554E49            DW      'MU','NI'
0560 43415449            DW      'CA','TI'
0564 4E472057            DW      'NG',' W'
0568 49544820            DW      'IT','H '
056C 44454320            DW      'DE','C '
                 ;
0570 18          MES8    DB      24
0571 20202053            DW      '  ',' S'
0575 48555444            DW      'HU','TD'
0579 4F574E20            DW      'OW','N '
057D 494E4954            DW      'IN','IT'
0581 49415445            DW      'IA','TE'
0585 44202020            DW      'D ','  '
                 ;
0589 18          MES9    DB      24
058A 42592044            DW      'BY',' D'
058E 45432043            DW      'EC',' C'
0592 454E5452            DW      'EN','TR'
0596 414C2043            DW      'AL',' C'
059A 4F4D5055            DW      'OM','PU'
059E 54455220            DW      'TE','R '
                 ;
05A2 18          MES10   DB      24
05A3 20202020            DW      '  ','  '
05A7 42592054            DW      'BY',' T'
05AB 582D3130            DW      'X-','10'
05AF 30205359            DW      '0 ','SY'
05B3 5354454D            DW      'ST','EM'
05B7 20202020            DW      '  ','  '
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
                    ;
                    ;=====================================================
                    ;TIMERO INTERRUPT HANDLER - .5,3,18 SEC DELAYS
                    ;=====================================================
05BB C0E0           TIMERS: PUSH    ACC
05BD C0D0                   PUSH    PSW
05BF C082                   PUSH    DPL
05C1 C083                   PUSH    DPH
05C3 C0F0                   PUSH    B
05C5 302A05                 JNB     T1START,TIME2
05C8 D53D02                 DJNZ    EST,    TIME2
05CB 8006                   SJMP    TIME3
                    ;
05CD 302B21         TIME2:  JNB     T2START,TIME5
05D0 D53E1E                 DJNZ    RT,     TIME5
                    ;
05D3 0535           TIME3:  INC     RXCNT
05D5 E535                   MOV     A,      RXCNT
05D7 B40408                 CJNE    A,      #RXMLIM,        TIME4
                    ;
05DA C2A9                   CLR     ETO
05DC D219                   SETB    TXOFF
05DE 7453                   MOV     A,      #SHUTDWN
05E0 801B                   SJMP    TIME6
                    ;
05E2 D229           TIME4:  SETB    NOSAVE
05E4 120993                 CALL    NEWSEQ
05E7 753D07                 MOV     EST,    #UPDATE1
05EA 753E2A                 MOV     RT,     #UPDATE2
05ED 7452                   MOV     A,      #REXMIT
05EF 800C                   SJMP    TIME6
                    ;
05F1 302C19         TIME5:  JNB     T3START,TIME8
05F4 D53F16                 DJNZ    VT,     TIME8
05F7 C2A9                   CLR     ETO
05F9 D219                   SETB    TXOFF
05FB 7453                   MOV     A,      #SHUTDWN
                    ;
05FD 452E           TIME6:  ORL     A,      SEQNUM
05FF A2D0                   MOV     C,      P
0601 202902                 JB      NOSAVE, TIME7
0604 F533                   MOV     PRESRPY,A
0606 120999         TIME7:  CALL    XMITOK
0609 929B                   MOV     TB8,    C
060B F599                   MOV     SBUF,   A
                    ;
060D D0F0           TIME8:  POP     B
060F D083                   POP     DPH
0611 D082                   POP     DPL
0613 D0D0                   POP     PSW
0615 D0E0                   POP     ACC
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
0617 32                     RETI
                  ;
                  ;=========================================================
                  ;SERIAL INTERRUPT HANDLER
                  ;=========================================================
0618 1214F4       SERIAL: CALL    PUSH
061B 20984A               JB      RI,       RECEIVE
061E C299                 CLR     TI
0620 C210                 CLR     XMITBIT
0622 301D05               JNB     CDATA,    NOCAL
0625 12088D               CALL    CALOUT
0628 802D                 SJMP    RPYDONE
062A 301605       NOCAL:  JNB     TDATA,NODATA
062D 1208EB               CALL    TEMPOUT
0630 8025                 SJMP    RPYDONE
0632 D53026       NODATA: DJNZ    OUTREG,   NOTFIN
0635 C229                 CLR     NOSAVE
0637 853332               MOV     LASTRPY,PRESRPY
063A 753003               MOV     OUTREG,   #RPYCNT
063D 301903               JNB     TXOFF,    CALCHK
0640 0207B3               LJMP    KILL
0643 301C08       CALCHK: JNB     CALBIT,   TEMPCHK
0646 12086C               CALL    CINIT
0649 12088D               CALL    CALOUT
064C 8009                 SJMP    RPYDONE
064E 301506       TEMPCHK:JNB     TEMPBIT,RPYDONE
0651 120878               CALL    TINIT
0654 1208EB               CALL    TEMPOUT
0657 12151D       RPYDONE:CALL    POP
065A 32                   RETI
065B 202905       NOTFIN: JB      NOSAVE,   NOTFIN1
065E E533                 MOV     A,        PRESRPY
0660 0207D0               LJMP    SERRET
0663 7452         NOTFIN1:MOV     A,        #REXMIT
0665 0207D0               LJMP    SERRET
                  ;
0668 301103       RECEIVE:JNB     BADCHAR,PARITY1
066B 020789               LJMP    BADSEQ
066E E599         PARITY1:MOV     A,        SBUF
0670 8599F0               MOV     B,        SBUF
0673 201F0F               JB      CALIN,    TMPJMP1
0676 20D006               JB      P,        ODDPAR1
0679 309A0C               JNB     RB8,      SEQCHEK
067C 020789               LJMP    BADSEQ
067F 209A06       ODDPAR1:JB      RB8,      SEQCHEK
0682 020789               LJMP    BADSEQ
                  ;
0685 0207E6       TMPJMP1:LJMP    CALBLK
                  ;
0688 301208       SEQCHEK:JNB     SEQSTRT,GOODSEQ
068B B53402               CJNE    A,        PRESCMD,        TJMP
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
068E 800A                    SJMP     CMD0
0690 020789      TJMP:       LJMP     BADSEQ
                 ;
0693 D212        GOODSEQ:SETB     SEQSTRT
0695 F534                    MOV      PRESCMD,A
0697 120961                  CALL     TSET
                 ;
069A D53108      CMD0:       DJNZ     INREG,   TJMP0
069D 12096C                  CALL     ESTSET
06A0 120993                  CALL     NEWSEQ
06A3 8003                    SJMP     GETDSN
06A5 0207E0      TJMP0:      LJMP     NOREPLY
                 ;
06A8 C227        GETDSN:     CLR      DECSNO
06AA 30E706                  JNB      ACC.7,   CMD1
06AD D227                    SETB     DECSNO
06AF C2E7                    CLR      ACC.7
06B1 F5F0                    MOV      B,       A
                 ;
                 ;VALID COMMAND COMPARISON
06B3 B44E22      CMD1:       CJNE     A,       #NAMSTAT,         CMD2
06B6 120972                  CALL     COMPARE
06B9 302803                  JNB      SNSAME,  CMD1OK
06BC 020798                  LJMP     RXMIT
06BF B277        CMD1OK:     CPL      TXSEQNO
06C1 D213                    SETB     LINE
06C3 753500                  MOV      RXCNT,   #00H
06C6 E526                    MOV      A,       IDSTAT
06C8 201B08                  JB       PAUSE,   TJMP1B
06CB B41B02                  CJNE     A,       #1BH,             TJMP1A
06CE D21B                    SETB     PAUSE
06D0 0207D0      TJMP1A:     LJMP     SERRET
06D3 D21A        TJMP1B:     SETB     READY
06D5 0207E0                  LJMP     NOREPLY
                 ;
06D8 B44C30      CMD2:       CJNE     A,       #LOAD,            CMD4
06DB 120972                  CALL     COMPARE
06DE 302803                  JNB      SNSAME,  CMD2OK
06E1 020798                  LJMP     RXMIT
06E4 B277        CMD2OK:     CPL      TXSEQNO
06E6 D21C                    SETB     CALBIT
06E8 D226                    SETB     CALRX
06EA C213                    CLR      LINE
06EC 753500                  MOV      RXCNT,   #00H
06EF 7455                    MOV      A,       #UNLOAD
06F1 0207D0                  LJMP     SERRET
                 ;
06F4 B45514      CMD3:       CJNE     A,       #UNLOAD,          CMD4
06F7 D21F        CINIT2:     SETB     CALIN
06F9 7538A0                  MOV      OFFSET,  #0A0H
06FC 753900                  MOV      CHKSUML,#00H
```

```
06FF 753A00                MOV     CHKSUMH,#00H
0702 C221                  CLR     CSLRCVD
0704 C222                  CLR     CSHRCVD
0706 C205                  CLR     UPDATE
0708 0207E0                LJMP    NOREPLY
                 ;
070B B44917      CMD4:     CJNE    A,      #INITGO,        CMD5
070E 120972                CALL    COMPARE
0711 302803                JNB     SNSAME, CMD4OK
0714 020798                LJMP    RXMIT
0717 B277        CMD4OK:   CPL     TXSEQNO
0719 D214                  SETB    DONESEQ
071B D213                  SETB    LINE
071D 753500                MOV     RXCNT,  #00H
0720 7444                  MOV     A,      #DONE
0722 0207D0                LJMP    SERRET
                 ;
0725 B45419      CMD5:     CJNE    A,      #XMITTMP,       CMD6
0728 120972                CALL    COMPARE
072B 302803                JNB     SNSAME, CMD5OK
072E 020798                LJMP    RXMIT
0731 B277        CMD5OK:   CPL     TXSEQNO
0733 D215                  SETB    TEMPBIT
0735 C226                  CLR     CALRX
0737 C213                  CLR     LINE
0739 753500                MOV     RXCNT,  #00H
073C 7445                  MOV     A,      #RECTMP
073E 0207D0                LJMP    SERRET
                 ;
0741 B45225      CMD6:     CJNE    A,      #REXMIT,        CMD7
0744 120972                CALL    COMPARE
0747 202803                JB      SNSAME, SENDRPY
074A 020798                LJMP    RXMIT
074D 753500      SENDRPY:MOV       RXCNT,  #00H
0750 301305                JNB     LINE,   BLOCK
0753 E532                  MOV     A,      LASTRPY
0755 0207D0                LJMP    SERRET
0758 202607      BLOCK:    JB      CALRX,  BLOCK2
075B D215                  SETB    TEMPBIT
075D 7445                  MOV     A,      #RECTMP
075F 0207D0                LJMP    SERRET
0762 D21C        BLOCK2:   SETB    CALBIT
0764 7455                  MOV     A,      #UNLOAD
0766 0207D0                LJMP    SERRET
                 ;
0769 B45319      CMD7:     CJNE    A,      #SHUTDWN,       TJMP8
076C C2AF                  CLR     EA
076E 120981                CALL    LEDOFF
0771 121000                CALL    CLRLCD
0774 900570                MOV     DPTR,   #MES8
0777 12101C                CALL    PMLCD
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
077A 12102D              CALL    NXTLN
077D 900589              MOV     DPTR,   #MES9
0780 12101C              CALL    PMLCD
0783 80FE                SJMP    $
0785 C211       TJMP8:   CLR     BADCHAR
0787 800F                SJMP    RXMIT
                ;
                ;BAD SEQUENCE RECEIVED
0789 D211       BADSEQ:  SETB    BADCHAR
078B D53107              DJNZ    INREG,  BADSEQ1
078E 12096C              CALL    ESTSET
0791 C211                CLR     BADCHAR
0793 8003                SJMP    RXMIT
0795 0207E0     BADSEQ1:LJMP     NOREPLY
                ;
                ;RETRANSMISSION DESIRED
0798 302505     RXMIT:   JNB     NORXMIT,RXMIT1
079B D223                SETB    BADDATA
079D 0206F7              LJMP    CINIT2
07A0 D229       RXMIT1:  SETB    NOSAVE
07A2 120993              CALL    NEWSEQ
07A5 0535                INC     RXCNT
07A7 E535                MOV     A,      RXCNT
07A9 B40420              CJNE    A,      #RXMLIM,        AGAIN
07AC D219                SETB    TXOFF
07AE 7453                MOV     A,      #SHUTDWN
07B0 0207D0              LJMP    SERRET
07B3 C2AF       KILL:    CLR     EA
07B5 120981              CALL    LEDOFF
07B8 121000              CALL    CLRLCD
07BB 900570              MOV     DPTR,   #MES8
07BE 12101C              CALL    PMLCD
07C1 12102D              CALL    NXTLN
07C4 9005A2              MOV     DPTR,   #MES10
07C7 12101C              CALL    PMLCD
07CA 80FE                SJMP    $
07CC D22B       AGAIN:   SETB    T2START
07CE 7452                MOV     A,      #REXMIT
                ;
07D0 452E       SERRET:  ORL     A,      SEQNUM
07D2 A2D0                MOV     C,      P
07D4 202902              JB      NOSAVE, SERRET1
07D7 F533                MOV     PRESRPY,A
07D9 120999     SERRET1:CALL     XMITOK
07DC 929B                MOV     TB8,    C
07DE F599                MOV     SBUF,   A
07E0 C298       NOREPLY:CLR      RI
07E2 12151D              CALL    POP
07E5 32                  RETI
                ;
                ;==========================================================
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
                    ;RECEIVES CAL DATA FROM THE DEC
                    ;=====================================================
07E6 302303         CALBLK: JNB     BADDATA,PARITY2
07E9 02083F                 LJMP    MOVPNTR
07EC 20D008         PARITY2:JB      P,      ODDPAR2
07EF 309A0D                 JNB     RB8,    DECCS
07F2 D223                   SETB    BADDATA
07F4 02083F                 LJMP    MOVPNTR
07F7 209A05         ODDPAR2:JB      RB8,    DECCS
07FA D223                   SETB    BADDATA
07FC 02083F                 LJMP    MOVPNTR
                    ;
07FF 30211C         DECCS:  JNB     CSLRCVD,DATA
0802 202207                 JB      CSHRCVD,DECCSH
0805 D222                   SETB    CSHRCVD
0807 F53B                   MOV     DECSUML,A
0809 020862                 LJMP    CFINISH
080C F53C           DECCSH: MOV     DECSUMH,A
080E E53B                   MOV     A,      DECSUML
0810 B53952                 CJNE    A,      CHKSUML,        CAGAIN
0813 E53C                   MOV     A,      DECSUMH
0815 B53A4D                 CJNE    A,      CHKSUMH,        CAGAIN
0818 30233C                 JNB     BADDATA,REINIT
081B 020865                 LJMP    CAGAIN
                    ;
081E C3             DATA:   CLR     C
081F E5F0                   MOV     A,      B
0821 2539                   ADD     A,      CHKSUML
0823 F539                   MOV     CHKSUML,A
0825 5002                   JNC     NOCARY3
0827 053A                   INC     CHKSUMH
                    ;
0829 202005         NOCARY3:JB      CALRCVD,GETPRB
082C 900200                 MOV     DPTR,   #ALPHA1
082F 8003                   SJMP    ADJUST1
0831 900100         GETPRB: MOV     DPTR,   #CALPRB
0834 1582           ADJUST1:DEC     DPL
0836 E582                   MOV     A,      DPL
0838 2538                   ADD     A,      OFFSET
083A F582                   MOV     DPL,    A
083C E5F0                   MOV     A,      B
083E F0                     MOVX    @DPTR,  A
                    ;
083F 2021BD         MOVPNTR:JB      CSLRCVD,DECCS
0842 D5381D                 DJNZ    OFFSET, CFINISH
0845 302008                 JNB     CALRCVD,PINIT2
0848 C220                   CLR     CALRCVD
084A D221                   SETB    CSLRCVD
084C C222                   CLR     CSHRCVD
084E 8012                   SJMP    CFINISH
0850 753830         PINIT2: MOV     OFFSET, #30H
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
0853 D220                SETB    CALRCVD
0855 800B                SJMP    CFINISH
                ;
0857 D224       REINIT:  SETB    LOADED
0859 C21F                CLR     CALIN
085B C225                CLR     NORXMIT
085D D205                SETB    UPDATE
085F 753500              MOV     RXCNT,  #00H
                ;
0862 0207E0     CFINISH:LJMP     NOREPLY
0865 C223       CAGAIN:  CLR     BADDATA
0867 C21F                CLR     CALIN
0869 0207A0              LJMP    RXMIT1
                ;
                ;==============================================
                ;INITIALIZES POINTER TO ALPHA ARRAY
                ;==============================================
086C D21D       CINIT:   SETB    CDATA
086E 7538A0              MOV     OFFSET, #0A0H
0871 753900              MOV     CHKSUML,#00H
0874 753A00              MOV     CHKSUMH,#00H
0877 22                  RET
                ;
                ;==============================================
                ;INITIALIZES POINTER TO LEDTA ARRAY
                ;==============================================
0878 D216       TINIT:   SETB    TDATA
087A 753604              MOV     DIGIT,  #04H
087D 753700              MOV     PROBE,  #00H
0880 E537                MOV     A,      PROBE
0882 23                  RL      A
0883 23                  RL      A
0884 F538                MOV     OFFSET, A
0886 753900              MOV     CHKSUML,#00H
0889 753A00              MOV     CHKSUMH,#00H
088C 22                  RET
                ;
                ;==============================================
                ;OUTPUTS CALIBRATION DATA TO DEC
                ;==============================================
088D 30171D     CALOUT:  JNB     SENDCSL,SENDCAL
0890 201809              JB      SENDCSH,SENDCS1
0893 D218                SETB    SENDCSH
0895 E539                MOV     A,      CHKSUML
0897 F5F0                MOV     B,      A
0899 0208DF              LJMP    COUT
089C 753001     SENDCS1:MOV      OUTREG, #01H
089F E53A                MOV     A,      CHKSUMH
08A1 C21C                CLR     CALBIT
08A3 C21D                CLR     CDATA
08A5 C218                CLR     SENDCSH
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
08A7 C217                 CLR      SENDCSL
08A9 F5F0                 MOV      B,       A
08AB 8032                 SJMP     COUT
                ;
08AD 201E05     SENDCAL:JB       CALSENT,SENDPRB
08B0 900200               MOV      DPTR,    #ALPHA1
08B3 8003                 SJMP     ADJUST
08B5 900100     SENDPRB:MOV      DPTR,    #CALPRB
08B8 1582       ADJUST:  DEC      DPL
08BA E582                 MOV      A,       DPL
08BC 2538                 ADD      A,       OFFSET
08BE F582                 MOV      DPL,     A
08C0 E0                   MOVX     A,       @DPTR
08C1 F5F0                 MOV      B,       A
08C3 C3                   CLR      C
08C4 2539                 ADD      A,       CHKSUML
08C6 F539                 MOV      CHKSUML,A
08C8 5002                 JNC      NOCARY2
08CA 053A                 INC      CHKSUMH
                ;
08CC D53810     NOCARY2:DJNZ     OFFSET, COUT
08CF 301E08               JNB      CALSENT,PINIT
08D2 D217                 SETB     SENDCSL
08D4 C218                 CLR      SENDCSH
08D6 C21E                 CLR      CALSENT
08D8 8005                 SJMP     COUT
08DA 753830     PINIT:    MOV      OFFSET, #30H
08DD D21E                 SETB     CALSENT
                ;
08DF E5F0       COUT:     MOV      A,       B
08E1 A2D0                 MOV      C,       P
08E3 120999               CALL     XMITOK
08E6 929B                 MOV      TB8,     C
08E8 F599                 MOV      SBUF,    A
08EA 22                   RET
                ;
                ;==============================================
                ;OUTPUTS TEMPERATURE DATA TO DEC
                ;==============================================
08EB 30171D     TEMPOUT:JNB      SENDCSL,CALCHEK
08EE 201809               JB       SENDCSH,SENDCS
08F1 D218                 SETB     SENDCSH
08F3 E539                 MOV      A,       CHKSUML
08F5 F5F0                 MOV      B,       A
08F7 020955               LJMP     TOUT2
08FA 753001     SENDCS:   MOV      OUTREG, #01H
08FD E53A                 MOV      A,       CHKSUMH
08FF C215                 CLR      TEMPBIT
0901 C216                 CLR      TDATA
0903 C218                 CLR      SENDCSH
0905 C217                 CLR      SENDCSL
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
0907 F5F0               MOV     B,        A
0909 804A               SJMP    TOUT2
                ;
090B 900120     CALCHEK:MOV     DPTR,     #ALLPRB
090E E582               MOV     A,        DPL
0910 2537               ADD     A,        PROBE
0912 F582               MOV     DPL,      A
0914 E0                 MOVX    A,        @DPTR
0915 B40002             CJNE    A,        #00H,          SENDTMP
0918 8015               SJMP    NOTZERO
                ;
091A 900300     SENDTMP:MOV     DPTR,     #LEDTA
091D 1582               DEC     DPL
091F E582               MOV     A,        DPL
0921 2536               ADD     A,        DIGIT
0923 2538               ADD     A,        OFFSET
0925 F582               MOV     DPL,      A
0927 E0                 MOVX    A,        @DPTR
0928 540F               ANL     A,        #0FH
092A B40F02             CJNE    A,        #0FH,          NOTZERO
092D 7400               MOV     A,        #00H
092F 2430       NOTZERO:ADD     A,        #30H
0931 F5F0               MOV     B,        A
0933 C3                 CLR     C
0934 2539               ADD     A,        CHKSUML
0936 F539               MOV     CHKSUML,A
0938 5002               JNC     NOCARY1
093A 053A               INC     CHKSUMH
                ;
093C D53616     NOCARY1:DJNZ    DIGIT,    TOUT2
093F E537               MOV     A,        PROBE
0941 B40F06             CJNE    A,        #0FH,          TOUT1
0944 D217               SETB    SENDCSL
0946 C218               CLR     SENDCSH
0948 800B               SJMP    TOUT2
094A 0537       TOUT1:  INC     PROBE
094C 05E0               INC     ACC
094E 23                 RL      A
094F 23                 RL      A
0950 F538               MOV     OFFSET, A
0952 753604             MOV     DIGIT,    #04H
                ;
0955 E5F0       TOUT2:  MOV     A,        B
0957 A2D0               MOV     C,        P
0959 120999             CALL    XMITOK
095C 929B               MOV     TB8,      C
095E F599               MOV     SBUF,     A
0960 22                 RET
                ;
                ;==========================================================
                ;START AND RESET OF TIMERS
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

```
                    ;==============================================
0961 753E2A         TSET:   MOV     RT,     #UPDATE2
0964 753F00                 MOV     VT,     #UPDATE3
0967 D22A                   SETB    T1START
0969 C22B                   CLR     T2START
096B 22                     RET
                    ;
096C 753D07         ESTSET: MOV     EST,    #UPDATE1
096F C22A                   CLR     T1START
0971 22                     RET
                    ;
                    ;==============================================
                    ;COMPARES SEQUENCE NUMBERS
                    ;==============================================
0972 207706         COMPARE:JB      TXSEQNO,COMP2
0975 302706                 JNB     DECSNO, COMP3
0978 C228           COMP1:  CLR     SNSAME
097A 22                     RET
097B 3027FA         COMP2:  JNB     DECSNO, COMP1
097E D228           COMP3:  SETB    SNSAME
0980 22                     RET
                    ;
                    ;==============================================
                    ;BLANKS LED DISPLAY
                    ;==============================================
0981 751440         LEDOFF: MOV     CNTR,   #40H
0984 740F                   MOV     A,      #0FH
0986 900300                 MOV     DPTR,   #LEDTA
0989 F0             LEDOFF1:MOVX    @DPTR,  A
098A 0582                   INC     DPL
098C D514FA                 DJNZ    CNTR,   LEDOFF1
098F 121544                 CALL    LED
0992 22                     RET
                    ;
                    ;==============================================
                    ;REINITIALIZES FOR NEW CMD SEQUENCE
                    ;==============================================
0993 753103         NEWSEQ: MOV     INREG,  #CMDCNT
0996 C212                   CLR     SEQSTRT
0998 22                     RET
                    ;
                    ;==============================================
                    ;ENSURES THAT TRANSMIT BUFFER IS CLEAR
                    ;==============================================
0999 301005         XMITOK: JNB     XMITBIT,XOK
099C 3099FA                 JNB     TI,     XMITOK
099F C299                   CLR     TI
09A1 D210           XOK:    SETB    XMITBIT
09A3 22                     RET
                    ;
                    ;
```

```
AVOCET SYSTEMS 8051 CROSS-ASSEMBLER -  VERSION 1.09

SOURCE FILE NAME: TX100D.ASM

 0000                        END
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM
----- SYMBOL TABLE ----

	0000
	0000
ACC	00E0
ADD	16D2
ADJUST	08B8
ADJUST1	0834
AGAIN	07CC
ALLPRB	0120
ALPHA1	0200
ATD	107F
ATDPR	1067
B	00F0
BADCHAR	0011
BADDATA	0023
BADSEQ	0789
BADSEQ1	0795
BCD	18D0
BETA1	0250
BITS	0020
BK11	0485
BK12	0483
BK13	048A
BK8	0479
BKGND	0311
BKGND1	03C1
BKGND10	031A
BKGND11	0317
BKGND12	03BE
BKGND13	040A
BKGND18	04A3
BKGND2	04A1
BKGND3	03E8
BKGND4	03F3
BKGND6	03FF
BKGND7	04A9
BLOCK	0758
BLOCK2	0762
CAGAIN	0865
CAL	1215
CALBIT	001C
CALBLK	07E6
CALCHEK	090B
CALCHK	0643
CALFG	000B
CALIN	001F
CALLOOP	0225
CALOUT	088D
CALPRB	0100
CALRCVD	0020
CALRX	0026
CALSENT	001E
CDATA	001D
CFINISH	0862
CHKSUMH	003A
CHKSUML	0039
CHNUM	001A
CINIT	086C
CINIT2	06F7
CLRLCD	1000
CLRSW	0095
CMD0	069A
CMD1	06B3
CMD1OK	06BF
CMD2	06D8
CMD2OK	06E4
CMD3	06F4
CMD4	070B
CMD4OK	0717
CMD5	0725
CMD5OK	0731
CMD6	0741
CMD7	0769
CMDCNT	0003
CNTR	0014
COMP1	0978
COMP2	097B
COMP3	097E
COMPARE	0972
COUNT1	002B
COUNT2	002C
COUNT3	002D
COUT	08DF
CSHRCVD	0022
CSLRCVD	0021
DATA	081E
DECCS	07FF
DECCSH	080C
DECSNO	0027
DECSUMH	003C
DECSUML	003B
DIGIT	0036
DIV	17E6
DONE	0044
DONESEQ	0014
DPH	0083
DPL	0082
EA	00AF
ENDLOOP	023A
ES	00AC
EST	003D
ESTSET	096C
ETO	00A9
EXP1	0010
EXP2	0015
F0	00D5
F1	0008
F2	0009
FIX	18F8
FLT	1931
GETDSN	06A8
GETPRB	0831
GND1	0375
GND2	0379
GND3	038E
GND4	03B5
GOODSEQ	0693
IDSTAT	0026
INIT	0108
INIT3	017B
INIT4	016E
INITGO	0049
INITJMP	0100
INREG	0031
INT2	01AD
INT5	0192
J1	04AC
KILL	07B3
LASTRPY	0032
LCD	1057
LCDSTS	0094
LED	1544
LEDOFF	0981
LEDOFF1	0989
LEDTA	0300
LINE	0013
LOAD	004C
LOADED	0024
LOOP	027D
MAN1	0011
MAN2	0016
MES1	04B1
MES10	05A2
MES2	04C6
MES3	04DF
MES4	0510
MES4A	04F7
MES5	0529
MES6	053E
MES7	0557
MES8	0570
MES9	0589
MODE	0093
MOVPNTR	083F

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: TX100D.ASM
----- SYMBOL TABLE -----

MOVX	1454	RND	000C	TIME3	05D3
MUL	15C7	RPYCNT	0003	TIME4	05E2
N	0027	RPYDONE	0657	TIME5	05F1
NAMSTAT	004E	RS	0096	TIME6	05FD
NEWSEQ	0993	RT	003E	TIME7	0606
NMAN	0028	RXCNT	0035	TIME8	060D
NOCAL	062A	RXMIT	0798	TIMEJMP	0102
NOCARY1	093C	RXMIT1	07A0	TIMERS	05BB
NOCARY2	08CC	RXMLIM	0004	TINIT	0878
NOCARY3	0829	SBUF	0099	TJ1	023F
NODATA	0632	SCON	0098	TJMP	0690
NOREPLY	07E0	SENDCAL	08AD	TJMP0	06A5
NORXMIT	0025	SENDCS	08FA	TJMP1A	06D0
NOSAVE	0029	SENDCS1	089C	TJMP1B	06D3
NOTFIN	065B	SENDCSH	0018	TJMP8	0785
NOTFIN1	0663	SENDCSL	0017	TLSF	19CD
NOTZERO	092F	SENDPRB	08B5	TMOD	0089
NXTLN	102D	SENDRPY	074D	TMP	0019
ODDPAR1	067F	SENDTMP	091A	TMPJMP1	0685
ODDPAR2	07F7	SEQCHEK	0688	TOUT1	094A
OFFSET	0038	SEQNUM	002E	TOUT2	0955
ORNG	000D	SEQSTRT	0012	TR0	008C
OUTREG	0030	SERIAL	0618	TR1	008E
OV	00D2	SERJMP	0105	TSET	0961
P	00D0	SERRET	07D0	TXOFF	0019
P1	0090	SERRET1	07D9	TXSEQNO	0077
PARITY1	066E	SGN1	0002	UNLOAD	0055
PARITY2	07EC	SGN2	0003	UPDATE	0005
PAUSE	001B	SHUTDWN	0053	UPDATE1	0007
PINIT	08DA	SIGN	0004	UPDATE2	002A
PINIT2	0850	SNSAME	0028	UPDATE3	0000
PMLCD	101C	SP	0081	UPRAT	001F
POP	151D	STATUS1	0022	VLSF	19D4
PRBSTS	0110	STATUS2	0023	VREXP	001B
PRESCMD	0034	STATUS3	0024	VRMAN	001C
PRESRPY	0033	STATUS4	0025	VRSGN	000A
PROBE	0037	SW	0091	VT	003F
PSW	00D0	SW1	030A	WAIT	10AF
PUSH	14F4	SWITCH	0300	WRITE	11A7
RB8	009A	T1START	002A	WT3MS	10C2
RDY	01F7	T2START	002B	XEXP	0008
RDY1	0202	T3START	002C	XMAN	0009
RDY2	0213	TB8	009B	XMITBIT	0010
RDY3	0208	TDATA	0016	XMITOK	0999
RDY4	0241	TEMP	0340	XMITTMP	0054
READY	001A	TEMPBIT	0015	XOK	09A1
RECEIVE	0668	TEMPCHK	064E	XRCL1	1989
RECTMP	0045	TEMPOUT	08EB	XSGN	0000
REINIT	0857	TH1	008D	XSTO1	19AB
REXMIT	0052	TI	0099	YEXP	000C
RI	0098	TIME2	05CD	YMAN	000D

```
AVOCET SYSTEMS 8051 CROSS-ASSEMBLER -  VERSION 1.09

SOURCE FILE NAME: TX100D.ASM
----- SYMBOL TABLE ----

YMOV      1486          YMOVVR   1475          YSGN      0001
```

APPENDIX G

APPLICATOR SUBSYSTEM PROGRAM LISTING

This appendix contains the source code listing for the applicator subsystem program entitled US100.ASM. Brief descriptions of the variable names encountered in the code are presented in the program preface. In addition, several comments concerning program development are located in the Appendix E introduction.

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
;
;===============================================
;US100 VERSION 0
;
;--THIS PROGRAM IS DESIGNED TO INTERACT WITH
;  C.W. BADGER'S CURRENT PDP-11 PROGRAM.
;  IT CONTAINS NO CODE TO IMPLEMENT TIMING
;  FUNCTIONS OR SEQUENCE NUMBER CONTROL.
;
;--THE TOTAL DUTY CYCLE PERIOD IS 0.1 SECOND
;  LONG AND IT IS BROKEN UP INTO 10 SLOTS TO
;  YIELD A DUTY CYCLE TIME SLOT OF 0.01
;  SECOND.  DUTY CYCLES CAN RANGE FROM 0% TO
;  100% IN 10% INCREMENTS.
;
;--DESCRIPTIONS OF THE VARIABLE NAMES USED
;  IN THIS PROGRAM ARE PRESENTED BELOW.
;
;===============================================
;
;===============================================
;DESCRIPTION OF BYTE VARIABLES
;
;  OUTREG - STORES THE CONSTANT, RPYCNT, WHICH
;  IS DECREMENTED WHENEVER ANOTHER CHARACTER
;  IN A REDUNDANT REPLY SET IS TRANSMITTED.
;
;  INREG - STORES THE CONSTANT, CMDCNT, WHICH
;  IS DECREMENTED WHENEVER ANOTHER CHARACTER
;  IN A REDUNDANT COMMAND SET IS RECEIVED.
;
;  LASTRPY - STORES THE REPLY LAST TRANSMIT-
;  TED TO THE DEC.  SAVED IN CASE THE US-100
;  REQUESTS A RETRANSMISSION.
;
;  PRESRPY - STORES THE REPLY PRESENTLY BEING
;  TRANSMITTED TO THE DEC.  SAVED TO ALLOW FOR
;  REPLY DUPLICATION IN THE REDUNDANT SET.
;
;  PRESCMD - STORES THE COMMAND PRESENTLY
;  BEING RECEIVED FROM THE DEC.  SAVED TO EN-
;  SURE THAT ALL THREE COMMANDS ARE IDENTICAL.
;
;  RXCNT - STORES THE CONSTANT, RXMLIM, WHICH
;  IS DECREMENTED WHENEVER ANOTHER RETRANS-
;  MISSION REQUEST IS SENT TO THE DEC.
;
;  OFFSET - STORES THE OFFSET FROM THE BASE
;  ADDRESS OF THE 16-BYTE DUTY CYCLE ARRAY.
;  USED TO CORRECTLY RECEIVE THE DUTY CYCLE
;  DATA.
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
;
;   TMPDUTY - STORES THE 16 BYTES OF DUTY CYCLE
;   DATA.
;
;   CHKSUML - STORES THE LOW BYTE OF THE GEN-
;   ERATED CHECKSUM.
;
;   CHKSUMH - STORES THE HIGH BYTE OF THE GEN-
;   ERATED CHECKSUM.
;
;   DECSUML - STORES THE LOW BYTE OF THE CHECK-
;   SUM RECEIVED FROM THE DEC.
;
;   DECSUMH - STORES THE HIGH BYTE OF THE
;   CHECKSUM RECEIVED FROM THE DEC.
;
;   SOFTIME - STORES A CONSTANT THAT IS THEN
;   DECREMENTED TO PROVIDE THE CORRECT DUTY
;   CYCLE TIME SLOT.  LOADED WITH CONSTANT "UP-
;   DATE" TO CREATE A 0.01 SECOND TIME SLOT.
;
;   LOBITS - STORES THE 8 BITS THAT INDICATE
;   WHETHER THE LOW 8 AMPLIFIERS WILL BE ON OR
;   OFF DURING THE NEXT DUTY CYCLE TIME SLOT.
;
;   HIBITS - SAME AS LOBITS, EXCEPT FOR THE
;   HIGH 8 AMPLIFIERS.
;
;   LOWAIT - STORES THE LOBITS REGISTER DURING
;   A WAIT STATE.
;
;   HIWAIT - STORES THE HIBITS REGISTER DURING
;   A WAIT STATE.
;
;   CURSLOT - STORES THE CURRENT DUTY CYCLE
;   TIME SLOT (1 THROUGH 10).
;
;   TEMP1, TEMP2, COUNT1, COUNT2 - SCRATCHPAD
;   REGISTERS USED FOR RAM-TO-RAM TRANSFERS
;   AND VARIOUS DATA COUNTS.
;
;   AMP - STORES THE AMPLIFIER NUMBER CURRENTLY
;   BEING CHECKED FOR FORWARD POWER OR REVERSE
;   POWER.
;
;   PSTS - STORES THE CURRENT SETTING OF THE
;   MAIN POWER SUPPLY.
;
;=================================================
;
;=================================================
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
;DESCRIPTIONS OF BIT VARIABLES
;
;  STATUS1,STATUS2,STATUS3 - RESERVED BIT
;  PLACES.
;
;  IDSTAT - STORES THE IDENTIFICATION/STATUS
;  REPLY.  ONCE STATUSES ARE DEFINED, INDIVI-
;  DUAL BITS CAN BE TOGGLED TO PRODUCE THE
;  CORRECT STATUS WORD.
;
;  XMITBIT - USED IN THE XMITOK SUBROUTINE.
;  SET WHEN A TRANSMISSION IS CURRENTLY IN
;  PROGRESS.  CLEARED WHEN THE TRANSMISSION IS
;  FINISHED.
;
;  BADCHAR - SET WHEN A BAD CHARACTER HAS BEEN
;  RECEIVED FROM THE DEC.  CLEAR FOR GOOD
;  TRANSMISSIONS.
;
;  SEQSTRT - SET WHEN A REDUNDANT COMMAND SET
;  HAS ALREADY BEEN STARTED.  INDICATES THAT
;  COMMAND DUPLICATION MUST BE CHECKED.  CLEAR
;  UNTIL THE FIRST BYTE IN THE SET IS RE-
;  CEIVED.
;
;  DUTYBIT - SET WHEN THE RECEIVE DUTY CYCLES
;  COMMAND IS RECEIVED FROM THE DEC.  INDI-
;  CATES THAT A DUTY CYCLES BLOCK IS BEING
;  PROCESSED.
;
;  VOLTBIT - SET WHEN THE VOLTAGE COMMAND IS
;  RECEIVED FROM THE DEC.  INDICATES THAT A
;  VOLTAGE DATA TRIPLET IS BEING PROCESSED.
;
;  FREQBIT - SET WHEN THE FREQUENCY COMMAND IS
;  RECEIVED FROM THE DEC.  INDICATES THAT A
;  FREQUENCY DATA TRIPLET IS BEING PROCESSED.
;
;  BADDATA - USED DURING DATA CLOCK RECEP-
;  TIONS.  SET WHEN BAD DUTY CYCLE DATA HAVE
;  BEEN RECEIVED FROM THE DEC.
;
;  USOFF - SET WHEN THE US-100 NEEDS TO BE
;  SHUTDOWN.  AFTER THE SHUTDOWN COMMAND HAS
;  BEEN SENT TO THE DEC, THE US-100 CEASES ALL
;  ACTIVITY.
;
;  CSLRCVD - SET WHEN THE LOW BYTE OF THE DEC
;  CHECKSUM IS EXPECTED.  CLEARED AFTER THE
;  FULL DATA BLOCK HAS BEEN RECEIVED.
;
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
                    ;  CSHRCVD - SET WHEN THE HIGH BYTE OF THE
                    ;  DEC CHECKSUM IS EXPECTED.  CLEARED AFTER
                    ;  THE FULL DATA BLOCK HAS BEEN RECEIVED.
                    ;
                    ;  RSET - SET DURING THE SYSTEM RESET.
                    ;  CLEARED WHEN THE FIRST DUTY CYCLE BLOCK HAS
                    ;  BEEN RECEIVED.  INDICATES THAT ONLY THE
                    ;  FIRST BLOCK CAN BE LOADED INTO THE OLDDUTY
                    ;  ARRAY.
                    ;
                    ;  HEAT - SET WHEN THE DUTY CYCLE ROUTINE CAN
                    ;  BE ENTERED.  CLEARED TO SHUT OFF THE
                    ;  AMPLIFIERS.
                    ;
                    ;  LOWFREQ - SET WHEN THE SYSTEM FREQUENCY IS
                    ;  1 MHZ.  CLEARED WHEN THE SYSTEM FREQUENCY
                    ;  IS 3 MHZ.
                    ;
                    ;  RSET1 - SET DURING THE SYSTEM RESET.
                    ;  CLEARED AFTER THE FIRST INITIALIZE AND GO
                    ;  COMMAND IS RECEIVED.  INDICATES THAT THE
                    ;  DEFAULT VOLTAGE IS USED ONLY AFTER INITIAL
                    ;  POWER UP.
                    ;
                    ;  FORWARD - SET WHEN A FORWARD POWER CHECK
                    ;  IS BEING CONDUCTED ON AN AMPLIFIER.
                    ;  CLEARED FOR A REVERSE CHECK.
                    ;
                    ;  DIGITAL - INDICATES WHEN THE SECTION TO
                    ;  CHECK THE DIGITAL OUTPUTS OF THE CON-
                    ;  TROLLER CARD CAN BE ENTERED.  SET DURING
                    ;  SYSTEM RESET.  CLEARED WHEN A FORWARD-
                    ;  REVERSE POWER PROBLEM HAS BEEN DETECTED.
                    ;
                    ;==============================================
                    ;
                    ;==============================================
                    ;MEMORY ALLOCATION
                    ;==============================================
0008                OUTREG  EQU     08H
0009                INREG   EQU     09H
000A                LASTRPY EQU     0AH
000B                PRESRPY EQU     0BH
000C                PRESCMD EQU     0CH
000D                RXCNT   EQU     0DH
000E                OFFSET  EQU     0EH
000F                TMPDUTY EQU     0FH,1EH
001F                CHKSUML EQU     1FH
0020                CHKSUMH EQU     20H
0021                DECSUML EQU     21H
0022                DECSUMH EQU     22H
```

```
AVOCET SYSTEMS 8051 CROSS-ASSEMBLER -  VERSION 1.09

SOURCE FILE NAME: US100.ASM


 0023              SOFTIME EQU     23H
 0024              LOBITS  EQU     24H
 0025              HIBITS  EQU     25H
 0026              LOWAIT  EQU     26H
 0027              HIWAIT  EQU     27H
 0028              CURSLOT EQU     28H
 0029              TEMP1   EQU     29H
 002A              TEMP2   EQU     2AH
 0032              COUNT1  EQU     32H
 0033              COUNT2  EQU     33H
                   ;
 0030              AMP     EQU     30H
 0031              PSTS    EQU     31H
                   ;=============================================
                   ;BIT ALLOCATION
                   ;=============================================
 002C              STATUS1 EQU     2CH
 002D              STATUS2 EQU     2DH
 002E              STATUS3 EQU     2EH
 002F              IDSTAT  EQU     2FH
                   ;
 0060              XMITBIT EQU     STATUS1.0
 0061              BADCHAR EQU     STATUS1.1
 0062              SEQSTRT EQU     STATUS1.2
 0063              DUTYBIT EQU     STATUS1.3
 0064              VOLTBIT EQU     STATUS1.4
 0065              FREQBIT EQU     STATUS1.5
 0066              BADDATA EQU     STATUS1.6
 0067              USOFF   EQU     STATUS1.7
                   ;
 0068              CSLRCVD EQU     STATUS2.0
 0069              CSHRCVD EQU     STATUS2.1
 006A              RSET    EQU     STATUS2.2
 006B              HEAT    EQU     STATUS2.3
 006C              LOWFREQ EQU     STATUS2.4
 006D              RSET1   EQU     STATUS2.5
 006E              FORWARD EQU     STATUS2.6
 006F              DIGITAL EQU     STATUS2.7
                   ;
 0070              PWR     EQU     STATUS3.0
                   ;
                   ;=============================================
                   ;8031 PORT EQUIVALENTS
 0000              ,=============================================
 0090              PULSE   EQU     P1.0
 0091              DIS1M   EQU     P1.1
 0092              HVON    EQU     P1.2
 0093              DIS3M   EQU     P1.3
 0094              HVOFF   EQU     P1.4
                   ;
                   ;=============================================
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
                ;COMMAND-REPLY  EQUIVALENTS
                ;===============================================
0004E           NAMSTAT EQU     4EH
0049            INITGO  EQU     49H
0044            DONE    EQU     44H
0045            RECINT  EQU     45H
0056            VOLT    EQU     56H
0046            FREQ    EQU     46H
0057            WAIT    EQU     57H
0052            REXMIT  EQU     52H
0048            HELP    EQU     48H
0053            SHUTDWN EQU     53H
0054            TOGGLE  EQU     54H
                ;
                ;===============================================
                ;CONSTANTS
                ;===============================================
0003            CMDCNT  EQU     3
0003            RPYCNT  EQU     3
0004            RXMLIM  EQU     4
000A            SLOTLIM EQU     10
0024            UPDATE  EQU     36
0030            LLIMI   EQU     30H
003A            ULIMI   EQU     3AH
003F            ULIMH   EQU     3FH
                ;
                ;===============================================
                ;RAM ADDRESSES
                ;===============================================
0000            NEWDUTY EQU     0000H,000FH
0100            OLDDUTY EQU     0100H,010FH
0200            WAITDUT EQU     0200H,020FH
                ;
0300            PWLVL   EQU     0300H
                ;
                ;===============================================
                ;MEMORY MAP ADDRESSES
                ;===============================================
9000            ADADDRS EQU     9000H
B000            ARRAYLO EQU     0B000H,0B001H
E000            VADDRS  EQU     0E000H
                ;
                ;===============================================
                ;MAIN ROUTINE
                ;===============================================
0100                    ORG     0100H
                ;
                ;===============================================
                ;VECTOR JUMPS
                ;       0000-020100
                ;       000B-020102
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
                  ;       0023-020104
                  ;=============================================
0100 8006         INITJMP:SJMP     INIT
                  ;
0102 020195       TIMEJMP:LJMP     DUTY
                  ;
0105 020286       SERJMP: LJMP     SERIAL
                  ;
                  ;=============================================
                  ;8031 INITIALIZATION
                  ;=============================================
0108 758140       INIT    MOV      SP,      #40H
010B 12051A               CALL     OFF
                  ;
                  ;TIMER SET-UP
010E 758922               MOV      TMOD,    #22H
0111 758C00               MOV      THO,     #00H
0114 758DE8               MOV      TH1,     #0E8H
0117 D28C                 SETB     TRO
0119 D28E                 SETB     TR1
                  ;
                  ;SERIAL SET-UP
011B 7598D0               MOV      SCON,    #0D0H
                  ;
                  ;INTERRUPT SET-UP
011E D2A9                 SETB     ETO
0120 D2AC                 SETB     ES
0122 C2B9                 CLR      PTO
                  ;
                  ;=============================================
                  ;BIT INITIALIZATIONS
                  ;=============================================
0124 C260                 CLR      XMITBIT
0126 C261                 CLR      BADCHAR
0128 C266                 CLR      BADDATA
012A D26A                 SETB     RSET
012C D26D                 SETB     RSET1
012E C263                 CLR      DUTYBIT
0130 C264                 CLR      VOLTBIT
0132 C265                 CLR      FREQBIT
0134 D291                 SETB     DIS1M
0136 D293                 SETB     DIS3M
0138 D26C                 SETB     LOWFREQ
013A C267                 CLR      USOFF
013C D26F                 SETB     DIGITAL
                  ;
013E C270                 CLR      PWR
0140 D26E                 SETB     FORWARD
                  ;=============================================
                  ;STORAGE INITIALIZATION
                  ;=============================================
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
0142 12057C             CALL    NEWSEQ
0145 750803             MOV     OUTREG, #RPYCNT
0148 752F0C             MOV     IDSTAT, #0CH
014B 750D00             MOV     RXCNT,  #00H
014E 752324             MOV     SOFTIME,#UPDATE
                ;
0151 753000             MOV     AMP,    #00H
0154 90058D             MOV     DPTR,   #TABLE
0157 E530               MOV     A,      AMP
0159 93                 MOVC    A,      @A+DPTR
015A 909000             MOV     DPTR,   #ADADDRS
015D F0                 MOVX    @DPTR,  A
                ;
015E D2AF               SETB    EA
                ;
                ;================================================
                ;HARDWARE CHECK - CURRENTLY SENDS A SHUTDOWN
                ;================================================
0160 306FFD     CHECK:  JNB     DIGITAL,$
0163 C2AF               CLR     EA
0165 90B000             MOV     DPTR,   #ARRAYLO
0168 E0                 MOVX    A,      @DPTR
0169 B52409             CJNE    A,      LOBITS,         ERROR
016C A3                 INC     DPTR
016D E0                 MOVX    A,      @DPTR
016E B52504             CJNE    A,      HIBITS,         ERROR
0171 D2AF               SETB    EA
0173 80EB               SJMP    CHECK
                ;
0175 D267       ERROR:  SETB    USOFF
0177 C2A9               CLR     ET0
0179 D2AC               SETB    ES
017B D2AF               SETB    EA
017D 7453               MOV     A,      #SHUTDWN
017F F50B               MOV     PRESRPY,A
0181 A2D0               MOV     C,      P
0183 120582             CALL    XMITOK
0186 929B               MOV     TB8,    C
0188 F599               MOV     SBUF,   A
018A 80FE               SJMP    $
                ;
                ;================================================
                ;TIMER0 INTERRUPT HANDLER, DUTY CYCLE ROUTINE
                ;================================================
018C 02027B     TJ1:    LJMP    FINISH1
018F 020285     TJ2:    LJMP    FINISH2
0192 020271     TJ3:    LJMP    OUTPUT
                ;
0195 D523F7     DUTY:   DJNZ    SOFTIME,TJ2
0198 D2B9               SETB    PT0
019A C0E0               PUSH    ACC
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
 019C C0D0                 PUSH    PSW
 019E C082                 PUSH    DPL
 01A0 C083                 PUSH    DPH
 01A2 B290                 CPL     PULSE
 01A4 306BE5               JNB     HEAT,    TJ1
 01A7 752324               MOV     SOFTIME,#UPDATE
                  ;
 01AA 753308               MOV     COUNT2, #08H
 01AD 90010F               MOV     DPTR,    #OLDDUTY+000FH
                  ;
 01B0 E0          LOOPHI:  MOVX    A,       @DPTR
 01B1 F529                 MOV     TEMP1,   A
 01B3 E528                 MOV     A,       CURSLOT
 01B5 C3                   CLR     C
 01B6 9529                 SUBB    A,       TEMP1
 01B8 B3                   CPL     C
 01B9 E525                 MOV     A,       HIBITS
 01BB 33                   RLC     A
 01BC F525                 MOV     HIBITS, A
 01BE 1582                 DEC     DPL
 01C0 D533ED               DJNZ    COUNT2, LOOPHI
 01C3 753308               MOV     COUNT2, #08H
                  ;
 01C6 E0          LOOPLO:  MOVX    A,       @DPTR
 01C7 F529                 MOV     TEMP1,   A
 01C9 E528                 MOV     A,       CURSLOT
 01CB C3                   CLR     C
 01CC 9529                 SUBB    A,       TEMP1
 01CE B3                   CPL     C
 01CF E524                 MOV     A,       LOBITS
 01D1 33                   RLC     A
 01D2 F524                 MOV     LOBITS, A
 01D4 1582                 DEC     DPL
 01D6 D533ED               DJNZ    COUNT2, LOOPLO
 01D9 0528                 INC     CURSLOT
 01DB E528                 MOV     A,       CURSLOT
 01DD B40AB2               CJNE    A,       #SLOTLIM,        TJ3
 01E0 752800               MOV     CURSLOT,#00H
 01E3 900000               MOV     DPTR,    #NEWDUTY
 01E6 752A00               MOV     TEMP2,   #00H
 01E9 752901               MOV     TEMP1,   #01H
 01EC 753310               MOV     COUNT2, #10H
 01EF E0          XTOY:    MOVX    A,       @DPTR
 01F0 852983               MOV     DPH,     TEMP1
 01F3 F0                   MOVX    @DPTR,   A
 01F4 0582                 INC     DPL
 01F6 852A83               MOV     DPH,     TEMP2
 01F9 D533F3               DJNZ    COUNT2, XTOY
                  ;
                  ;==============================================
                  ;A-TO-D TEST, FORWARD POWER, REVERSE POWER
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
                    ;===============================================
01FC 909000         AMP0:    MOV     DPTR,    #ADADDRS
01FF F0                      MOVX    @DPTR,   A
0200 F8                      MOV     R0,      A
0201 0582                    INC     DPL
0203 E0                      MOVX    A,       @DPTR
0204 540F                    ANL     A,       #0FH
0206 F9                      MOV     R1,      A
0207 900300                  MOV     DPTR,    #PWLVL
020A E530                    MOV     A,       AMP
020C 23                      RL      A
020D 2582                    ADD     A,       DPL
020F F582                    MOV     DPL,     A
0211 E8                      MOV     A,       R0
0212 F0                      MOVX    @DPTR,   A
0213 0582                    INC     DPL
0215 E9                      MOV     A,       R1
0216 F0                      MOVX    @DPTR,   A
0217 306E26                  JNB     FORWARD,AMP2
021A C3                      CLR     C
021B 7408                    MOV     A,       #08H
021D 99                      SUBB    A,       R1
021E 5015                    JNC     AMP1
0220 6013                    JZ      AMP1
                    ;
                    ;FORWARD PROBLEM
                    ;
0222 D267                    SETB    USOFF
0224 C26F                    CLR     DIGITAL
0226 7453                    MOV     A,       #SHUTDWN
0228 F50B                    MOV     PRESRPY,          A
022A A2D0                    MOV     C,       P
022C 120582                  CALL    XMITOK
022F 929B                    MOV     TB8,     C
0231 F599                    MOV     SBUF,    A
0233 8046                    SJMP    FINISH1
                    ;
0235 0530           AMP1:    INC     AMP
0237 E530                    MOV     A,       AMP
0239 B4102D                  CJNE    A,       #10H,             AMP4
023C C26E                    CLR     FORWARD
023E 8029                    SJMP    AMP4
                    ;
0240 C3             AMP2:    CLR     C
0241 7402                    MOV     A,       #02H
0243 99                      SUBB    A,       R1
0244 5015                    JNC     AMP3
0246 6013                    JZ      AMP3
                    ;
                    ;REVERSE PROBLEM
                    ;
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
0248 D267                SETB    USOFF
024A C26F                CLR     DIGITAL
024C 7453                MOV     A,      #SHUTDWN
024E F50B                MOV     PRESRPY,A
0250 A2D0                MOV     C,      P
0252 120582              CALL    XMITOK
0255 929B                MOV     TB8,    C
0257 F599                MOV     SBUF,   A
0259 8020                SJMP    FINISH1
                 ;
025B 0530        AMP3:   INC     AMP
025D E530                MOV     A,      AMP
025F B42007              CJNE    A,      #20H,           AMP4
0262 753000              MOV     AMP,    #00H
0265 E530                MOV     A,      AMP
0267 D26E                SETB    FORWARD
                 ;
0269 90058D      AMP4:   MOV     DPTR,   #TABLE
026C 93                  MOVC    A,      @A+DPTR
026D 909000              MOV     DPTR,   #ADADDRS
0270 F0                  MOVX    @DPTR,  A
                 ;
                 ;
0271 90B000      OUTPUT: MOV     DPTR,   #ARRAYLO
0274 E524                MOV     A,      LOBITS
0276 F0                  MOVX    @DPTR,  A
0277 A3                  INC     DPTR
0278 E525                MOV     A,      HIBITS
027A F0                  MOVX    @DPTR,  A
                 ;
027B D083        FINISH1:POP     DPH
027D D082                POP     DPL
027F D0D0                POP     PSW
0281 D0E0                POP     ACC
0283 C2B9                CLR     PT0
0285 32          FINISH2:RETI
                 ;
                 ;===============================================
                 ;SERIAL INTERRUPT HANDLER
                 ;===============================================
0286 C0E0        SERIAL: PUSH    ACC
0288 C0D0                PUSH    PSW
028A C0F0                PUSH    B
028C C082                PUSH    DPL
028E C083                PUSH    DPH
0290 209823              JB      RI,     RECEIVE
0293 C299                CLR     TI
0295 C260                CLR     XMITBIT
0297 D50817              DJNZ    OUTREG, NOTFIN
029A 750803              MOV     OUTREG, #RPYCNT
029D 850B0A              MOV     LASTRPY,PRESRPY
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
02A0 306703              JNB     USOFF,  RPYDONE
02A3 0204FB              LJMP    KILL
02A6 D083      RPYDONE:POP     DPH
02A8 D082              POP     DPL
02AA D0F0              POP     B
02AC D0D0              POP     PSW
02AE D0E0              POP     ACC
02B0 32                RETI
02B1 E50B      NOTFIN: MOV     A,      PRESRPY
02B3 020502            LJMP    SERRET
               ;
02B6 306103    RECEIVE:JNB     BADCHAR,PARITY1
02B9 0204D5            LJMP    BADSEQ
02BC E599      PARITY1:MOV     A,      SBUF
02BE 8599F0            MOV     B,      SBUF
02C1 206315            JB      DUTYBIT,TMPJMP1
02C4 20641B            JB      VOLTBIT,SEQCHEK
02C7 206518            JB      FREQBIT,SEQCHEK
02CA 20D006            JB      P,      ODDPAR1
02CD 309A12            JNB     RB8,    SEQCHEK
02D0 0204D5            LJMP    BADSEQ
02D3 209A0C    ODDPAR1:JB      RB8,    SEQCHEK
02D6 0204D5            LJMP    BADSEQ
               ;
02D9 0203C4    TMPJMP1:LJMP    DUTYBLK
02DC 02045E    TMPJMP2:LJMP    VOLTIN
02DF 0204A0    TMPJMP3:LJMP    FREQIN
               ;
02E2 306208    SEQCHEK:JNB     SEQSTRT,GOODSEQ
02E5 B50C02            CJNE    A,      PRESCMD,        TMPJMP4
02E8 8007              SJMP    NUMBER
02EA 0204D5    TMPJMP4:LJMP    BADSEQ
               ;
02ED D262      GOODSEQ:SETB    SEQSTRT
02EF F50C              MOV     PRESCMD,A
               ;
02F1 2064E8    NUMBER: JB      VOLTBIT,TMPJMP2
02F4 2065E8            JB      FREQBIT,TMPJMP3
               ;
02F7 D50905            DJNZ    INREG,  TJMP1
02FA 12057C            CALL    NEWSEQ
02FD 8003              SJMP    CMD1
02FF 02050D    TJMP1:  LJMP    NOREPLY
               ;
0302 B44E08    CMD1:   CJNE    A,      #NAMSTAT,       CMD2
0305 750D00            MOV     RXCNT,  #00H
0308 E52F              MOV     A,      IDSTAT
030A 020502            LJMP    SERRET
               ;
030D B44930    CMD2:   CJNE    A,      #INITGO,        CMD3
0310 D270              SETB    PWR
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
0312 B292                 CPL     HVON
0314 B292                 CPL     HVON
0316 206C04               JB      LOWFREQ,CMD2A
0319 C293                 CLR     DIS3M
031B 8002                 SJMP    CMD2B
031D C291       CMD2A:    CLR     DIS1M
031F 852624     CMD2B:    MOV     LOBITS, LOWAIT
0322 852725               MOV     HIBITS, HIWAIT
0325 D26B                 SETB    HEAT
0327 306D08               JNB     RSET1,  DEFAULT
032A 740B                 MOV     A,      #0BH
032C F531                 MOV     PSTS,   A
032E C26D                 CLR     RSET1
0330 8002                 SJMP    LOAD
0332 E531       DEFAULT:MOV       A,      PSTS
0334 90E000     LOAD:     MOV     DPTR,   #VADDRS
0337 F0                   MOVX    @DPTR,  A
0338 750D00               MOV     RXCNT,  #00H
033B 7444                 MOV     A,      #DONE
033D 020502               LJMP    SERRET
                ;
0340 B44515     CMD3:     CJNE    A,      #RECINT,          CMD4
0343 D263       DINIT:    SETB    DUTYBIT
0345 750E0F               MOV     OFFSET, #TMPDUTY
0348 753210               MOV     COUNT1,#10H
034B 751F00               MOV     CHKSUML,#00H
034E 752000               MOV     CHKSUMH,#00H
0351 C268                 CLR     CSLRCVD
0353 C269                 CLR     CSHRCVD
0355 02050D               LJMP    NOREPLY
                ;
0358 B45605     CMD4:     CJNE    A,      #VOLT,            CMD5
035B D264                 SETB    VOLTBIT
035D 02050D               LJMP    NOREPLY
                ;
0360 B44605     CMD5:     CJNE    A,      #FREQ,            CMD6
0363 D265                 SETB    FREQBIT
0365 02050D               LJMP    NOREPLY
                ;
0368 B45720     CMD6:     CJNE    A,      #WAIT,            CMD7
036B C26B                 CLR     HEAT
036D D291                 SETB    DIS1M
036F D293                 SETB    DIS3M
0371 852426               MOV     LOWAIT, LOBITS
0374 7524FF               MOV     LOBITS, #0FFH
0377 852527               MOV     HIWAIT, HIBITS
037A 7525FF               MOV     HIBITS, #0FFH
037D 90E000               MOV     DPTR,   #VADDRS
0380 7400                 MOV     A,      #00H
0382 F0                   MOVX    @DPTR,  A
0383 750D00               MOV     RXCNT,  #00H
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
0386 7444                 MOV      A,       #DONE
0388 020502               LJMP     SERRET
                  ;
038B B45208      CMD7:    CJNE     A,       #REXMIT,        CMD8
038E 750D00               MOV      RXCNT,   #00H
0391 E50A                 MOV      A,       LASTRPY
0393 020502               LJMP     SERRET
                  ;
0396 B45307      CMD8:    CJNE     A,       #SHUTDWN,       TJMP9
0399 C2AF                 CLR      EA
039B 12051A               CALL     OFF
039E 80FE                 SJMP     $
03A0 8003        TJMP9:   SJMP     CMD9
03A2 0204D5               LJMP     BADSEQ
                  ;==================================================
                  ;TEST COMMAND FOR THE HARDWARE
                  ;==================================================
03A5 B45417      CMD9:    CJNE     A,       #TOGGLE,        TJMP10
03A8 B270                 CPL      PWR
03AA 307006               JNB      PWR,     HV2
03AD B292                 CPL      HVON
03AF B292                 CPL      HVON
03B1 8004                 SJMP     HV3
03B3 B294        HV2:     CPL      HVOFF
03B5 B294                 CPL      HVOFF
03B7 750D00      HV3:     MOV      RXCNT,   #00H
03BA 7444                 MOV      A,       #DONE
03BC 020502               LJMP     SERRET
03BF C261        TJMP10:  CLR      BADCHAR
03C1 0204E1               LJMP     RXMIT
                  ;
                  ;==================================================
                  ;PROCESS DUTY CYCLE BLOCK
                  ;==================================================
03C4 306602      DUTYBLK:JNB       BADDATA,PARITY2
03C7 8050                 SJMP     MOVPNTR
03C9 20D008      PARITY2:JB        P,       ODDPAR2
03CC 309A0D               JNB      RB8,     DECCS
03CF D266                 SETB     BADDATA
03D1 020419               LJMP     MOVPNTR
03D4 209A05      ODDPAR2:JB        RB8,     DECCS
03D7 D266                 SETB     BADDATA
03D9 020419               LJMP     MOVPNTR
                  ;
03DC 306812      DECCS:   JNB      CSLRCVD,LLCHEK1
03DF 206907               JB       CSHRCVD,DECCS1
03E2 D269                 SETB     CSHRCVD
03E4 F521                 MOV      DECSUML,A
03E6 020454               LJMP     DFINISH
03E9 F522        DECCS1:  MOV      DECSUMH,A
03EB 306639               JNB      BADDATA,FILL
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
03EE 020457             LJMP    DAGAIN
                   ;
03F1 C3        LLCHEK1:CLR     C
03F2 9430               SUBB    A,      #LLIMI
03F4 5005               JNC     ULCHEK1
03F6 D266               SETB    BADDATA
03F8 020419             LJMP    MOVPNTR
03FB C3        ULCHEK1:CLR     C
03FC 743A               MOV     A,      #ULIMI
03FE 95F0               SUBB    A,      B
0400 5005               JNC     DUTYDAT
0402 D266               SETB    BADDATA
0404 020419             LJMP    MOVPNTR
                   ;
0407 C3        DUTYDAT:CLR     C
0408 E5F0               MOV     A,      B
040A 251F               ADD     A,      CHKSUML
040C F51F               MOV     CHKSUML,A
040E 5002               JNC     NOCARRY
0410 0520               INC     CHKSUMH
                   ;
0412 740F      NOCARRY:MOV     A,      #0FH
0414 55F0               ANL     A,      B
0416 A80E               MOV     R0,     OFFSET
0418 F6                 MOV     @R0,    A
                   ;
0419 2068C0    MOVPNTR:JB      CSLRCVD,DECCS
041C 050E               INC     OFFSET
041E D53233             DJNZ    COUNT1, DFINISH
0421 D268               SETB    CSLRCVD
0423 C269               CLR     CSHRCVD
0425 802D               SJMP    DFINISH
                   ;
0427 E521      FILL:    MOV     A,      DECSUML
0429 B51F2B             CJNE    A,      CHKSUML,        DAGAIN
042C E522               MOV     A,      DECSUMH
042E B52026             CJNE    A,      CHKSUMH,        DAGAIN
0431 750D00             MOV     RXCNT,  #00H
0434 C263               CLR     DUTYBIT
0436 900000             MOV     DPTR,   #NEWDUTY
0439 780F               MOV     R0,     #TMPDUTY
043B 753210             MOV     COUNT1, #10H
043E 120574             CALL    TEMPTOX
0441 306A10             JNB     RSET,   DFINISH
0444 900100             MOV     DPTR,   #OLDDUTY
0447 780F               MOV     R0,     #TMPDUTY
0449 753210             MOV     COUNT1, #10H
044C 120574             CALL    TEMPTOX
044F C26A               CLR     RSET
0451 752800             MOV     CURSLOT,#00H
                   ;
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
0454 02050D      DFINISH:LJMP    NOREPLY
0457 C266        DAGAIN: CLR     BADDATA
0459 C263                CLR     DUTYBIT
045B 0204E1              LJMP    RXMIT
                 ;
                 ;==========================================
                 ;PROCESS VOLTAGE VALUE
                 ;==========================================
045E 20D007      VOLTIN: JB      P,        ODDPAR3
0461 309A0B              JNB     RB8,      LLCHEK2
0464 C264                CLR     VOLTBIT
0466 8079                SJMP    RXMIT
0468 209A04      ODDPAR3:JB      RB8,      LLCHEK2
046B C264                CLR     VOLTBIT
046D 8072                SJMP    RXMIT
046F C3          LLCHEK2:CLR     C
0470 9430                SUBB    A,        #LLIMI
0472 5004                JNC     ULCHEK2
0474 C264                CLR     VOLTBIT
0476 8069                SJMP    RXMIT
0478 C3          ULCHEK2:CLR     C
0479 743F                MOV     A,        #ULIMH
047B 95F0                SUBB    A,        B
047D 5004                JNC     VDATA
047F C264                CLR     VOLTBIT
0481 805E                SJMP    RXMIT
0483 D50917      VDATA:  DJNZ    INREG,    VDATA1
0486 750D00              MOV     RXCNT,    #00H
0489 12057C              CALL    NEWSEQ
048C 740F                MOV     A,        #0FH
048E 55F0                ANL     A,        B
0490 F531                MOV     PSTS,     A
0492 90E000              MOV     DPTR,     #VADDRS
0495 F0                  MOVX    @DPTR,    A
0496 C264                CLR     VOLTBIT
0498 7444                MOV     A,        #DONE
049A 020502              LJMP    SERRET
049D 02050D      VDATA1: LJMP    NOREPLY
                 ;
                 ;==========================================
                 ;PROCESS FREQUENCY VALUE
                 ;==========================================
04A0 20D007      FREQIN: JB      P,        ODDPAR4
04A3 309A0B              JNB     RB8,      LLCHEK3
04A6 C265                CLR     FREQBIT
04A8 8037                SJMP    RXMIT
04AA 209A04      ODDPAR4:JB      RB8,      LLCHEK3
04AD C265                CLR     FREQBIT
04AF 8030                SJMP    RXMIT
04B1 B43108      LLCHEK3:CJNE    A,        #31H,          ULCHEK3
04B4 C291                CLR     DIS1M
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
 04B6 D293                SETB     DIS3M
 04B8 D26C                SETB     LOWFREQ
 04BA 800C                SJMP     FSET
 04BC B43322     ULCHEK3:CJNE      A,        #33H,           RXMIT
 04BF D5094B              DJNZ     INREG,    NOREPLY
 04C2 C293                CLR      DIS3M
 04C4 D291                SETB     DIS1M
 04C6 C26C                CLR      LOWFREQ
 04C8 750D00     FSET:    MOV      RXCNT,    #00H
 04CB 12057C              CALL     NEWSEQ
 04CE C265                CLR      FREQBIT
 04D0 7444                MOV      A,        #DONE
 04D2 020502              LJMP     SERRET
                 ;
                 ;==============================================
                 ;BAD CHARACTER RECEIVED
                 ;==============================================
 04D5 D261       BADSEQ:  SETB     BADCHAR
 04D7 D50904              DJNZ     INREG,    BADSEQ1
 04DA C261                CLR      BADCHAR
 04DC 8003                SJMP     RXMIT
 04DE 02050D     BADSEQ1:LJMP      NOREPLY
                 ;
                 ;==============================================
                 ;RETRANSMISSION REQUEST
                 ;==============================================
 04E1 12057C     RXMIT:   CALL     NEWSEQ
 04E4 C264                CLR      VOLTBIT
 04E6 C265                CLR      FREQBIT
 04E8 050D                INC      RXCNT
 04EA E50D                MOV      A,        RXCNT
 04EC B40407              CJNE     A,        #RXMLIM,        AGAIN
 04EF 7453                MOV      A,        #SHUTDWN
 04F1 D267                SETB     USOFF
 04F3 020502              LJMP     SERRET
 04F6 7452       AGAIN:   MOV      A,        #REXMIT
 04F8 020502              LJMP     SERRET
                 ;
                 ;==============================================
                 ;SHUTDOWN ROUTINE
                 ;==============================================
 04FB C2AF       KILL:    CLR      EA
 04FD 12051A              CALL     OFF
 0500 80FE                SJMP     $
                 ;
                 ;==============================================
                 ;RETURN FROM SERIAL INTERRUPT
                 ;==============================================
 0502 A2D0       SERRET:  MOV      C,        P
 0504 F50B                MOV      PRESRPY,A
 0506 120582              CALL     XMITOK
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
 0509 929B                  MOV     TB8,      C
 050B F599                  MOV     SBUF,     A
 050D C298         NOREPLY:CLR      RI
 050F D083                  POP     DPH
 0511 D082                  POP     DPL
 0513 D0F0                  POP     B
 0515 D0D0                  POP     PSW
 0517 D0E0                  POP     ACC
 0519 32                    RETI
                   ;
                   ;==============================================
                   ;SUBROUTINE OFF
                   ;==============================================
 051A C2AF         OFF:     CLR     EA
                   ;
 051C D292                  SETB    HVON
 051E D294                  SETB    HVOFF
 0520 C294                  CLR     HVOFF
                   ;
 0522 C26B                  CLR     HEAT
 0524 D291                  SETB    DIS1M
 0526 D293                  SETB    DIS3M
 0528 90B000                MOV     DPTR,     #ARRAYLO
 052B 74FF                  MOV     A,        #0FFH
 052D F524                  MOV     LOBITS,   A
 052F F525                  MOV     HIBITS,   A
 0531 F526                  MOV     LOWAIT,   A
 0533 F527                  MOV     HIWAIT,   A
 0535 F0                    MOVX    @DPTR,    A
 0536 A3                    INC     DPTR
 0537 F0                    MOVX    @DPTR,    A
                   ;
 0538 753100                MOV     PSTS,     #00H
 053B 90E000                MOV     DPTR,     #VADDRS
 053E E531                  MOV     A,        PSTS
 0540 F0                    MOVX    @DPTR,    A
                   ;
 0541 753212                MOV     COUNT1,   #12H
 0544 780F                  MOV     R0,       #TMPDUTY
 0546 7400                  MOV     A,        #00H
 0548 F6           BLANK:   MOV     @R0,      A
 0549 08                    INC     R0
 054A D532FB                DJNZ    COUNT1,   BLANK
 054D 780F                  MOV     R0,       #TMPDUTY
 054F 900000                MOV     DPTR,     #NEWDUTY
 0552 753210                MOV     COUNT1,   #10H
 0555 120574                CALL    TEMPTOX
 0558 780F                  MOV     R0,       #TMPDUTY
 055A 900100                MOV     DPTR,     #OLDDUTY
 055D 753210                MOV     COUNT1,   #10H
 0560 120574                CALL    TEMPTOX
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM

```
 0563 780F                 MOV     R0,      #TMPDUTY
 0565 900200               MOV     DPTR,    #WAITDUT
 0568 753210               MOV     COUNT1, #10H
 056B 120574               CALL    TEMPTOX
 056E 752800               MOV     CURSLOT,#00H
 0571 D2AF                 SETB    EA
 0573 22                   RET
                   ;
                   ;==============================================
                   ;SUBROUTINE TEMPTOX
                   ;==============================================
 0574 E6           TEMPTOX:MOV     A,       @R0
 0575 F0                   MOVX    @DPTR,   A
 0576 08                   INC     R0
 0577 A3                   INC     DPTR
 0578 D532F9               DJNZ    COUNT1, TEMPTOX
 057B 22                   RET
                   ;
                   ;
 057C 750903       NEWSEQ: MOV     INREG,   #CMDCNT
 057F C262                 CLR     SEQSTRT
 0581 22                   RET
                   ;
 0582 306005       XMITOK: JNB     XMITBIT,XOK
 0585 3099FA               JNB     TI,      XMITOK
 0588 C299                 CLR     TI
 058A D260         XOK:    SETB    XMITBIT
 058C 22                   RET
                   ;
                   ;==============================================
                   ;CONSTANTS
                   ;==============================================
 058D 1D131618     TABLE   DW      $1D13,$1618,$121E,$1519
 0595 0C0B0D0E             DW      $0C0B,$0D0E,$0406,$0F00
 059D 11101C1B             DW      $1110,$1C1B,$1F17,$141A
 05A5 0201070A             DW      $0201,$070A,$0503,$0908
                   ;
 0000                      END
```

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM
----- SYMBOL TABLE -----

	0000	DPL	0082	ODDPAR1	02D3
	0000	DUTY	0195	ODDPAR2	03D4
ACC	00E0	DUTYBIT	0063	ODDPAR3	0468
ADADDRS	9000	DUTYBLK	03C4	ODDPAR4	04AA
AGAIN	04F6	DUTYDAT	0407	OFF	051A
AMP	0030	EA	00AF	OFFSET	000E
AMP0	01FC	ERROR	0175	OLDDUTY	0100
AMP1	0235	ES	00AC	OUTPUT	0271
AMP2	0240	ET0	00A9	OUTREG	0008
AMP3	025B	FILL	0427	P	00D0
AMP4	0269	FINISH1	027B	P1	0090
ARRAYLO	B000	FINISH2	0285	PARITY1	02BC
B	00F0	FORWARD	006E	PARITY2	03C9
BADCHAR	0061	FREQ	0046	PRESCMD	000C
BADDATA	0066	FREQBIT	0065	PRESRPY	000B
BADSEQ	04D5	FREQIN	04A0	PSTS	0031
BADSEQ1	04DE	FSET	04C8	PSW	00D0
BLANK	0548	GOODSEQ	02ED	PT0	00B9
CHECK	0160	HEAT	006B	PULSE	0090
CHKSUMH	0020	HELP	0048	PWLVL	0300
CHKSUML	001F	HIBITS	0025	PWR	0070
CMD1	0302	HIWAIT	0027	RB8	009A
CMD2	030D	HV2	03B3	RECEIVE	02B6
CMD2A	031D	HV3	03B7	RECINT	0045
CMD2B	031F	HVOFF	0094	REXMIT	0052
CMD3	0340	HVON	0092	RI	0098
CMD4	0358	IDSTAT	002F	RPYCNT	0003
CMD5	0360	INIT	0108	RPYDONE	02A6
CMD6	0368	INITGO	0049	RSET	006A
CMD7	038B	INITJMP	0100	RSET1	006D
CMD8	0396	INREG	0009	RXCNT	000D
CMD9	03A5	KILL	04FB	RXMIT	04E1
CMDCNT	0003	LASTRPY	000A	RXMLIM	0004
COUNT1	0032	LLCHEK1	03F1	SBUF	0099
COUNT2	0033	LLCHEK2	046F	SCON	0098
CSHRCVD	0069	LLCHEK3	04B1	SEQCHEK	02E2
CSLRCVD	0068	LLIMI	0030	SEQSTRT	0062
CURSLOT	0028	LOAD	0334	SERIAL	0286
DAGAIN	0457	LOBITS	0024	SERJMP	0105
DECCS	03DC	LOOPHI	01B0	SERRET	0502
DECCS1	03E9	LOOPLO	01C6	SHUTDWN	0053
DECSUMH	0022	LOWAIT	0026	SLOTLIM	000A
DECSUML	0021	LOWFREQ	006C	SOFTIME	0023
DEFAULT	0332	MOVPNTR	0419	SP	0081
DFINISH	0454	NAMSTAT	004E	STATUS1	002C
DIGITAL	006F	NEWDUTY	0000	STATUS2	002D
DINIT	0343	NEWSEQ	057C	STATUS3	002E
DIS1M	0091	NOCARRY	0412	TABLE	058D
DIS3M	0093	NOREPLY	050D	TB8	009B
DONE	0044	NOTFIN	02B1	TEMP1	0029
DPH	0083	NUMBER	02F1	TEMP2	002A

AVOCET SYSTEMS 8051 CROSS-ASSEMBLER - VERSION 1.09

SOURCE FILE NAME: US100.ASM
----- SYMBOL TABLE -----

TEMPTOX	0574	TMPJMP1	02D9	USOFF	0067
TH0	008C	TMPJMP2	02DC	VADDRS	E000
TH1	008D	TMPJMP3	02DF	VDATA	0483
TI	0099	TMPJMP4	02EA	VDATA1	049D
TIMEJMP	0102	TOGGLE	0054	VOLT	0056
TJ1	018C	TR0	008C	VOLTBIT	0064
TJ2	018F	TR1	008E	VOLTIN	045E
TJ3	0192	ULCHEK1	03FB	WAIT	0057
TJMP1	02FF	ULCHEK2	0478	WAITDUT	0200
TJMP10	03BF	ULCHEK3	04BC	XMITBIT	0060
TJMP9	03A0	ULIMH	003F	XMITOK	0582
TMOD	0089	ULIMI	003A	XOK	058A
TMPDUTY	000F	UPDATE	0024	XTOY	01EF

REFERENCES

Badger, C.W., private communication.

EIA Standard RS-232-C, Electronic Industries Association, 1969.

Goss, S.A., Burdette, E.C., Cain, C.A., Magin, R.L., Frizzell, L.A., Chen, M.M., Holmes, K.R., Badger, C.W., and McCarthy, J., "Systems Concept for Controlled Delivery of Clinical Ultrasound Hyperthermia," Proc. of the 29th Annual Meeting of the American Institute of Ultrasound in Medicine, J. Ultra. Med., vol. 3, no. 9, p. 27, 1984.

IEEE Standard Digital Interface for Programmable Instrumentation, The Institute of Electrical and Electronics Engineers, Inc., 1978.

Intel Corporation, Microcontroller User's Manual, May 1982.

Intel Corporation, Microsystem Components Handbook, Volume 2, 1984.

McNamara, J.E., Technical Aspects of Data Communication, Digital Press, Educational Services Department, 1978.

Seyer, M.D., RS-232 Made Easy: Connecting Computers, Printers, Terminals, and Modems, Prentice-Hall, Inc., 1984.

Silverman, S.G., "The Design of an Ultrasound Phased Array Controller for Use in Hyperthermia," M.S. Thesis, University of Illinois, Urbana, Illinois, 1984.

Stewart, J.R., Bagshaw, M.A., Corry, P.M., Gerner, E.W., Gibbs, Jr., F.A., Hahn, G.M., Lele, P. P., and Oleson, J.R., "Hyperthermia as a Treatment of Cancer," Cancer Treatment Symposia, vol. 1, pp. 135-144, 1984.

URI Therm-X, Inc., TX-100 Operating Instruction Manual, December, 1984.

Videx, Inc., PSIO Dual Function Interface Card Installation and Operation Manual, Feb., 1983.