Project No. A-2891 DATE: 3/20/81

Project Director: John T. Scoville ~~School~~/Lab ECSL/CTAD

Sponsor: Naval Construction Battalion Center, Port Hueneme, CA 93043

Type Agreement: Delivery Order #N68305-81-F-0026 (under BOA #F33657-80-G-0077)

Award Period: From 2/17/81 To ~~6/16/81~~ 8/31/81 (Performance) 8/16/81 (Reports)

Sponsor Amount: $49,733 Contracted through:

Cost Sharing: GTRI~~/GIT~~

Title: Non-Personal Services to Provide Metering Effort at NAB, Little Creek, VA.

ADMINISTRATIVE DATA OCA CONTACT Faith G. Costello

1) Sponsor Technical Contact: Mr. Karlin Canfield; (805) 982-3328; Civil Engineering Laboratory, Naval Construction Battalion Center, Port Hueneme, CA 93043

2) Sponsor Admin./Contractual Contact: Officer in Charge of Contracts, Civil Engineering Laboratory, Naval Construction Battalion Center, Port Hueneme, CA 93043

Reports: See Deliverable Schedule Security Classification: N/A

Defense Priority Rating: DO-C-2 under DMS Reg. 1

RESTRICTIONS

See Attached Government Supplemental Information Sheet for Additional Requirements

Travel: Foreign travel must have prior approval - Contact OCA in each case. Domestic travel requires sponsor approval where total will exceed greater of $500 or 125% of approved proposal budget category.

Equipment: Title vests with Government; except that items costing less than $1,000 vest with GIT upon acquisition; if prior approval to purchase is obtained from Contracting Officer.

COMMENTS:

COPIES TO:

Administrative Coordinator	Research Security Services	EES Information Office (2)
Research Property Management	Reports Coordinator (OCA)	Project File (OCA)
Accounting Office	Legal Services (OCA)	Other:
Procurement Office	Library, Technical Reports	

GEORGIA INSTITUTE OF TECHNOLOGY OFFICE OF CONTRACT ADMINISTRATION

SPONSORED PROJECT TERMINATION SHEET

Date 11/16/81

Project Title: Non-Personal Services to Provide Metering Effort at NAB, Little Creek, VA

Project No: A-2891

Project Director: John T. Scoville

Sponsor: Naval Construction Battalion Center

Effective Termination Date: 8/31/81

Clearance of Accounting Charges: 8/31/81

Grant/Contract Closeout Actions Remaining:

- [x] Final Invoice and Closing Documents
- [ ] Final Fiscal Report
- [ ] Final Report of Inventions
- [ ] Govt. Property Inventory & Related Certificate
- [ ] Classified Material Certificate
- [ ] Other ____________

Assigned to: ECSL/CTAD (~~School~~/Laboratory)

COPIES TO:

Administrative Coordinator
Research Property Management
Accounting
Procurement/EES Supply Services
Research Security Services
~~Reports Coordinator (OCA)~~
Legal Services (OCA)
Library
EES Public Relations (2)
Computer Input
Project File
Other ____________

FORM OCA 10:781

A-2891

Georgia Institute of Technology
ENGINEERING EXPERIMENT STATION
ATLANTA, GEORGIA 30332

April 14, 1981

Officer in Charge of Contracts
Civil Engineering Laboratory
Naval Construction Battalion Center
Port Hueneme, California 93043

Dear Sir:

Subject: March, 1981 Progress Report for Project A-2891
Delivery Order #N68305-81-F-0026

The following tasks were accomplished during the period from February 17, 1981 through March 31, 1981:

1. A report reviewing electrical power and steam consumption metering techniques was generated. Delivery to CEL is expected to be April 15, 1981.

2. A survey of available Multibus components was conducted so that various hardware configurations for the data gathering device could be evaluated. The following components were chosen after it was determined that this selection would be more than adequate to meet the device requirements:

   a. Intel SBC-80/24 single board computer.
   b. Intel SBC-094 lo-power CMOS RAM.
   c. Analog Devices RTI-1200 analog input system.
   d. Digital Pathways TCU-410 timing control unit.

3. It was found that an implementation device utilizing Intel software would not be feasible. After an investigation of other alternatives, it became evident that an equally powerful system could be constructed around Digital Research Corporation's CP/M operating system. The implementation device will be supported by extra RAM memory, floppy disk storage, and EPROM programming capabilities.

4. Procurement of the necessary hardware was begun. This included the single board computer components, power supplies, disk drives, software and minicomputer interface.

AN EQUAL EMPLOYMENT/EDUCATION OPPORTUNITY INSTITUTION

5. The initial definition and design of the software to be incorporated in the data gathering was begun.

The following areas of work will be pursued during April:

1. Completion of the software design

2. Configuration and checkout of any hardware that may be delivered.

The March budget sheet shows that of the original budgeted $49,733, $9,527 has been spent.

Sincerely,

Mr. James W. Eakes
Research Engineer I
Computer Technology and
Applications Division
Electronics and Computer
Systems Laboratory

JWE/jm

cc: Mr. Karlin Canfield
Civil Engineering Laboratory

Approved by ______________________
John T. Scoville
Project Director and
Division Chief

A-2891

Georgia Institute of Technology
ENGINEERING EXPERIMENT STATION
ATLANTA, GEORGIA 30332

May 15, 1981

Officer in Charge of Contracts
Civil Engineering Laboratory
Naval Construction Battalion Center
Port Hueneme, California 93043

Dear Sir:

Subject: April, 1981 Progress Report for Project A-2891-000
Delivery Order #N68305-81-F-0026

The following tasks were accomplished during the period from April 1, 1981, through April 30, 1981:

1. The CP/M software was configured for the implementation device hardware. The software needed to boot the CP/M system was also written.

2. Design of the software modules for the data gathering device was continued.

3. The preliminary copy of a technical report entitled "Electrical Power and Steam Consumption Metering Techniques" was completed and delivered.

The following areas of work will be pursued in May:

1. The data gathering device software will be implemented and tested.

2. Packaging of the device components will be done as the necessary hardware is delivered.

3. Development of the Interdata software will begin.

AN EQUAL EMPLOYMENT/EDUCATION OPPORTUNITY INSTITUTION

The April budget sheet shows that of the original budgeted $49,733.00, $29,466.93 has been spent. The amount spent during the month of April was $19,933.00.

Sincerely,

Mr. James W. Eakes
Research Engineer I
Computer Technology and
Applications Division
Electronics and Computer
Systems Laboratory

JWE/jm

cc: Karlin Canfield
Civil Engineering Laboratory

Approved by: ____________________
John T. Scoville
Division Chief and
Project Director

A-2891

Georgia Institute of Technology
ENGINEERING EXPERIMENT STATION
ATLANTA, GEORGIA 30332

June 15, 1981

Officer in Charge of Contracts
Civil Engineering Laboratory
Naval Construction Battalion Center
Port Hueneme, California 93043

Reference: Delivery #N68305-81-F-0026 under BOA #F33657-80-G-0077 Non-Personal Services to Provide Metering Effort at NAB, Little Creek, Virginia, Georgia Tech Project A-2891-000

Subject: Letter Progress Report Number 3 for Reporting Period of 5/1/81 to 5/31/81

Dear Sir:

The following tasks were accomplished during the period from May 1, 1981 through May 31, 1981:

1. The data gathering device hardware was mounted in a NEMA 12 enclosure.

2. Development of the Interdata software was begun. This software is a FORTRAN task that will retrieve the collected data.

3. Implementation and testing of the software modules for the data gathering device was continued.

The following areas of work will be pursued in June:

1. Development of the data gathering device software will be continued.

2. The development system hardware will be mounted in an Optima enclosure.

3. The Interdata software effort will also be continued.

AN EQUAL EMPLOYMENT/EDUCATION OPPORTUNITY INSTITUTION

The May, 1981 budget sheet shows that of the original budgeted $49,733.00, $32,741.59 has been spent. During the month of May, 1981, $16,140.66 was spent.

Mr. Karlin Canfield of the Civil Engineering Laboratory visited Georgia Tech on June 5, 1981. After discussing several matters, it was decided that Little Creek, Virginia would not be the ideal location to install the data gathering device. To eliminate any potential problems, we suggest that the device be installed in Building 501, Damneck, Virginia sometime in August, 1981. This alteration in plans would require an additional $2,500 and a final report/documentation deadline of August 31, 1981. Your consideration of this matter will be greatly appreciated.

Sincerely,

Mr. James W. Eakes
Research Engineer I
Computer Technology and
Applications Division
Electronics and Computer
Systems Laboratory

JWE/jm

cc: Karlin Canfield
Civil Engineering Laboratory
Milton Bennett and Faith Costello, OCA

Approved by:

John T. Scoville, Chief
Computer Technology and
Applications Division

A-2891

Georgia Institute of Technology
ENGINEERING EXPERIMENT STATION
ATLANTA, GEORGIA 30332

July 20, 1981

Officer in Charge of Contracts
Civil Engineering Laboratory
Naval Construction Battalion Center
Port Hueneme, California 93043

Reference: Delivery #N68305-81-F-0026 Under BOA #F33657-80-G0077 - Non-Personal Services to Provide Metering Effort at NAB, Damneck, Virginia, Georgia Tech Project A-2891-000

Subject: Letter Progress Report #4 for Reporting Period of 6/1/81 to 6/30/81

Dear Sir:

The following tasks were accomplished during the period from June 1, 1981 through June 30, 1981.

1. Implementation and testing of the software modules for the data gathering device was continued.

2. Modem and line printer utilities were added to the development system.

The following areas of work will be pursued in July:

1. The data gathering device software will be completed and tested.

2. Software to initiate automatic dialing will be added to the Interdata task. This will be tested when the modem equipment is received.

The June, 1981 budget sheet shows that of the original budgeted $49,733.00, $43,286.00 has been spent.

Approved by:

John T. Scoville, Chief
Computer Technology and
Applications Division

Sincerely,

Mr. James W. Eakes
Research Engineer I
Computer Technology and
Applications Division
Electronics and Computer
Systems Laboratory

JWE/jm

cc: Karlin Canfield
Civil Engineering Laboratory

To:

Civil Engineering Laboratory
Naval Construction and Battalion Center
Port Hueneme, California 93043

Under:

BOA #F33657-80-G-0077

FINAL REPORT AND DOCUMENTATION
FOR
PURCHASE ORDER #N68305-81-F-0026
FOR
NON-PERSONAL SERVICES TO PROVIDE
METERING EFFORT AT NAB, DAMNECK, VIRGINIA

By:

James W. Eakes
Research Engineer I
Computer Technology and Applications Division
Electronics and Computer Systems Laboratory
ENGINEERING EXPERIMENT STATION
Georgia Institute of Technology
Atlanta, Georgia 30332

Under:

GIT Project A-2891-000

Contracting Through:

Georgia Tech Research Institute
Georgia Institute of Technology
Atlanta, Georgia 30332

August 31, 1981

Table of Contents

1.0 Introduction

The purpose of this contract is to provide the government with tools that may be used to determine the effectiveness of an Energy Monitoring and Control System (EMCS). The actual effectiveness of an EMCS can only be determined by monitoring the energy consumption before and after the installation of an EMCS. Energy consumption is most accurately recorded by digital computers with analog and digital interfaces to the physical data environment. Sensors monitoring various physical phenomena are attached directly to the computer. The computer may then correlate data in real time at very high speeds and store the results in engineering units. Under this contract, such a computer was developed using an industry standard microprocessor-based computer. In addition to this, three other tasks were accomplished to support the EMCS evaluation:

1. A microcomputer development system was constructed so that software for the data gathering device could be further enhanced at the Civil Engineering Laboratory (CEL);
2. A minicomputer software package for data retrieval was written to allow remote interrogation of the data gathering device; and
3. A report reviewing electrical power and steam consumption metering techniques was generated to provide information about the physical problems related to taking energy consumption data.

The report on metering techniques is an enclosure with this document. The data gathering device, microcomputer development system, and minicomputer interface are described in detail in Appendices A, B, and C of this document.

2.0 Data Gathering Device

The data gathering device is a portable data acquisition computer designed for remote data collection. The device utilizes the Intel Multibus microcomputer interface bus. The core of the system is an Intel iSBC-80/24 single-board computer. It is supported with four kilobytes of battery-protected RAM memory, a sixteen-channel analog input system, and a battery-protected calendar clock. The system power supply provides the necessary d.c. voltages in addition to power fail logic for an orderly power down sequence. Analog and digital sensor inputs are provided with Intel termination panels. The system components are housed within a wall mounting NEMA-12 enclosure on a removable panel. The enclosure is equipped with one air intake fan and one exhaust fan. An automatic-answer telephone modem is also housed in the enclosure.

The data gathering device software (installed in the first three Intel iSBC-80/24 EPROM sockets) is a data scanning program that records data in time-based intervals. The software assumes that the sensor parameters are stored in the fourth EPROM. Individual sensor data is recorded and stored in the battery-protected RAM memory. The digital data recorded is the percentage of the time interval that the sensor is read 'ON'. Analog data is recorded as high value, low value, and average value for the time interval. The analog data values are calculated using a third order sensor characteristic equation ($Data = a + bx + cx^2 + dx^3$). The coefficients of the equation are supplied by the user in the sensor EPROM. The data may be retrieved by a local terminal or through a telephone modem. When the system is prompted to send data, it is transmitted serially to the requesting device. The data is converted from raw form and transmitted in

ASCII characters over the serial link. Shown below is an example data dump in standard format:

```
MM/DD/YY HH:MM:SS                      (STARTING TIME)
1 NAME1 HI=VALUE LO=VALUE AVE=VALUE                        (ANALOG VALUE 1)
1 NAMEN HI=VALUE LO=VALUE AVE=VALUE                        (ANALOG VALUE N)
0 NAME1 XXX% ON                                            (DIGITAL VALUE 1)
0 NAMEN XXX% ON                                            (DIGITAL VALUE N)
MM/DD/YY HH:MM:SS                      (ENDING TIME)

DATA RECORD #2

DATA RECORD #3

DATA RECORD #N

POWER UP                               POWER DOWN
MM/DD/YY HH:MM:SS                      MM/DD/YY HH:MM:SS
```

The data gathering device records data around the clock. The records must be retrieved periodically since the device itself has a limited RAM storage area. The storage capability of the device is a function of the data record time interval and the number of sensors attached. The device can store data for a period of less than one hour to greater than twenty hours depending on these parameters. When 32 or less sensors are attached, the data scan rate is high enough to insure an overall system accuracy that is approximately equal to the average accuracy of the sensors. If the device had eight analog sensors and eight digital sensors attached, each sensor would be read at least 125 times per minute. For a detailed description of the data gathering device, see Appendix A.

3.0 Microcomputer Development System

The development system was constructed to provide CEL with software development capabilities needed to enhance or alter the data gathering device function. The system implements the CP/M disk-based operating system. CP/M supports 8080/Z80 assembly language development as well as PL-1, FORTRAN, PASCAL, and numerous versions of BASIC.

The development system hardware consists of the following components:

- Two double-density floppy-disk drives.
- A system chassis with power supply, I/O connectors, and an eight-slot Multibus cardcage.
- An Intel SBC-80/24 processor, 64K bytes of RAM memory and a data gathering device card set.
- Automatic-answer telephone modem.
- 2716 EPROM Programmer.

The software required to load and run the CP/M operating system and an 8080 monitor are activated upon power-up of the system. This software is loaded into high RAM and 'emulates' the operation of an Intel MDS system. (When CP/M is purchase it is assumed that it will be used on an Intel MDS system. To avoid alteration of the distribution version of CP/M, the CEL development system was designed to 'emulate' an Intel MDS system). When the operating system is loaded, there are 62K bytes of memory available for the user. At this time any CP/M system program or any software written to run under CP/M may be executed. For a more detailed description of the system, see Appendix B.

4.0 Minicomputer Interface

The minicomputer interface is included to allow remote interrogation of the data gathering device. This interface consists of an auto-dial modem and a FORTRAN software package for a Perkin-Elmer minicomputer. The software is composed of three interactive tasks that are designed to automatically call the data gathering device, retrieve the data, and store the data on a disk file. When the main task is running, this sequence of events will occur automatically on some fixed time interval. If communication problems are detected, an error message will be stored in the disk file. The user must supply the proper phone number and the time interval between data calls. See Appendix C for a detailed account of this interface.

APPENDIX A

DATA GATHERING DEVICE

Appendix A. Data Gathering Device

General Description

The CEL data gathering device is an eight-bit microcomputer with analog and digital inputs and two serial ports. The system software scans the inputs and stores the data readings in blocks based upon time intervals. The data organization depends upon the information stored in the sensor table EPROM. The data is stored as byte or floating point quantities in battery-protected RAM memory. When the device is commanded to transmit the data it is converted to ASCII characters and transmitted through one of its serial ports. One port may be used for a local terminal device and the other for a telephone modem.

The EPROM table that contains sensor information must be properly constructed for the software to operate. The table contains the number of digital and analog sensors and a description of each sensor. The device software continuously scans this table and the inputs to construct the time based data records. When the RAM area for data is full, the device returns to the base of this area and continues to record data. A portion of the RAM is reserved for storing power up and power down dates and times. Up to eight date and time records for power up and power down may be stored. This gives the user the ability to determine when power failed and what data record(s) are invalid due to the power failures.

Device Components

The data gathering device was constructed using Intel Multibus compatible microcomputer components. The components that were chosen for the data gathering device are listed below.

- Intel SBC-80/24 single board computer.
- Intel SBC-094 4K CMOS RAM memory board.
- Digital Pathways TCU-410 Julian calendar clock module.
- Analog Devices RTI-1200 analog input system.
- Intel SBC-604 4-slot Multibus cardcage.
- Two Intel SBC-900 series digital/analog I/O termination panels.
- National Semiconductor BLC-635 power supply.
- Racal-Vadic 3455P auto-answer telephone modem.
- Intel SBX-351 serial multimodule board.
- General Electric NEMA-12 single-door enclosure with cooling fans.

These components combine to yield the following features:

- High speed 8085A-2 processor.
- Twelve vectored interrupt levels.
- Two RS-232 serial ports.
- Up to 32K bytes of EPROM memory.
- Eight kilobytes of RAM memory.
- Forty-eight digital input/output channels.
- Sixteen analog input channels.
- Continuous hardware clock providing year, month, day, hour, minute, and second.
- Power fail detection.
- Data communications at 1200 BPS over the switched telephone network.

All of the components are mounted on a removable panel in the enclosure. The 80/24 processor card must be installed in the top slot of the cardcage with the CMOS RAM card directly below it. The clock and the analog input cards may be put in either of the two remaining slots. The digital and analog inputs are connected to the termination panels with fifty connector flat ribbon cable that is provided with the panels. For trouble-shooting, the following device manuals should be consulted:

1. Intel SBC-80/24 Hardware Reference Manual.
2. Intel SBX-351 Hardware Reference Manual.
3. Intel SBC-094 Hardware Reference Manual.
4. Intel SBC-604 Hardware Reference Manual.
5. Digital Pathways TCU-410 User's Manual
6. Analog Devices RTI-1200 User's Manual.
7. National Semiconductor BLC-635 User's Manual.

Three fully nested interrupts are used in the system. The highest priority interrupt is power fail, which is generated by the power supply. The state of the system, date, and time are stored when this interrupt occurs. The second interrupt services communications through the modem port. This interrupt is generated by the carrier detect output of the modem. The third interrupt services a local terminal. An interrupt occurs every time a character is sent from the terminal device. This interrupt has the lowest priority.

Data Acquisition Software

The data acquisition software is written in 8080 assembly language. The corresponding machine code resides in the first three EPROMs on the 80/24 processor card. The software may be separated into seven distinct parts:

1. Hardware initialization - This module initializes all of the programmable devices on power up.

2. Data scan initialization - This routine calculates and stores the pointers and constants used to scan and record data. Some of these parameters are variable and the exact value is dependent upon the number of sensors in the sensor table.

3. Data accumulation - This program scans the sensor table and inputs to create an intermediate data record for a time interval specified by the user. This time interval is usually within the range of five to thirty minutes. For each digital input there is one 32-bit accumulator that stores one hundred times the number of 'ON' readings and one 32-bit accumulator that stores the total number of readings. For each analog input the high reading and low reading are stored as 16-bit integers. The sum of all readings is stored in a 32-bit accumulator and the total number of readings is also stored in a 32-bit accumulator. At the end of the prescribed time interval, this data is converted and moved to the permanent record area in RAM. For each digital sensor, the 'ON' readings accumulator is divided by the total number of readings. This operation produces a one-byte result that is the percentage of the time that the sensor was read 'on'. The analog high and low values are converted to floating point and inserted into the third order sensor equation. The results are stored in floating point. The average value is found by dividing the sum of all readings by the total number of readings. This integer value is then converted to floating point, inserted into the equation, and stored in floating point.

   The sensor data is stored in the order that sensors appear in the sensor table EPROM. The beginning time and ending time for the interval are stored with the data.

4. Interrupt service routines - Program control is automatically turned over to a service routine when an interrupt occurs. The first routine stores the state of the system and the time in battery-protected memory as power goes down.

   The second routine services communications through the modem port. After the modem is on-line and has detected a carrier, the routine sends a prompt character. The device then waits for a command character that requests the data. If this character is not received within a minute, the device will continue the data scan. If the character is received, the device will send the data stored in the record area. The data is converted to ASCII at this time.

The third routine services communications with a local terminal. If an arbitrary key is hit, the device will transmit the following sign-on message:

NCBC-CEL DATA GATHERING SYSTEM

D - DISPLAY DATA
M - ENTER 80/24 MONITOR
S - SET DATA AND TIME
T - DISPLAY DATE AND TIME

The upper case characters D, M, S, and T are commands to the device. The 'D' command causes all available data to be sent to the terminal. The 'M' comand transfers control to an 8080 monitor that was included as a trouble-shooting aid. The monitor is described in Section 6. The 'S' command is used to set the real time clock. The format of this command is shown below:

SMM DD YY HH MM[CR]
(S[MONTH][SP][DAY][SP][YEAR][SP][HOUR][SP][MINUTE][CR])

The seconds are automatically set to zero when the time is set. The 'T' command displays the current date and time and may be used to verify proper setting of the clock.

After an interrupt is serviced, the device will return to the data scan unless it was commanded to enter the 8080 monitor. In this case, the monitor must be given the exit command before data scanning is resumed.

5. Floating point routines - The floating point implemented in the software uses eight bits for the exponent and sixteen bits for the mantissa. The exponent is represented using sign magnitude excess-64 notation. In excess-64, the exponent corresponding to 2**0 is 40H and the most significant bit of the exponent indicates the sign of the number. Positive exponents are represented by numbers greater than 40H and negative exponents are represented by numbers less than or equal to 40H. The mantissa is always normalized following any floating point operation. The floating point package contains the routines necessary to provide addition, subtraction, multiplication, and division with floating point numbers. The software stores floating point numbers in memory in the following order: exponent, low byte mantissa, and high byte mantissa. The floating point sensor coefficients must appear in the EPROM table in this same order.

6. SBC 80/24 monitor - This monitor was added to provide the data gathering device with a hardware and software debugging tool. The monitor accepts and acts upon commands entered by the user from the local terminal. The commands allow manipulation of memory, I/O devices, and execution of user programs. When the monitor program is entered, it prompts the user for a command by displaying a period. The following are valid commands.

DISPLAY MEMORY - D

.D[low address] [high address][CR]

The contents of memory between the address parameters are displayed on the terminal.

EXIT MONITOR - E

.E[CR]

This command causes program control to be turned over to the data scanning software.

FILL MEMORY - F

.F[low address] [high address] [constant][CR]

The memory locations between the address parameters are filled with the desired eight-bit constant.

GO TO MEMORY ADDRESS - G

.G[address][CR]

Program execution is begun at the given memory address.

HEXADECIMAL SUM AND DIFFERENCE - H

.H[value 1] [value 2][CR]

The sum of the two values is displayed first and the difference is displayed second.

INPUT PORT - I

.I[port number][CR]

The specified port is read and the value is displayed on the terminal device.

MOVE MEMORY - M

.M[low address] [high address] [starting address][CR]

The block of memory defined by the low and high parameters is moved to RAM beginning at the input starting address.

OUTPUT TO PORT - O

.O[port number] [constant][CR]

The given eight-bit constant is output to the desired port.

SUBSTITUTE MEMORY - S

.S[address] (present value)-[new value]

The current value of the given address is displayed and then changed to the given new value.

DISPLAY TIME - T

.T

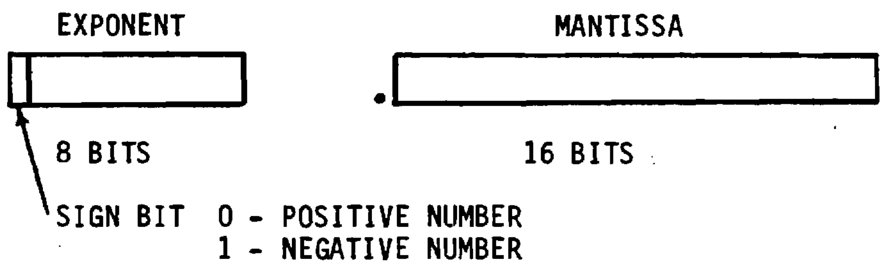
The clock module is read and the date and time is displayed on the terminal.

All numbers input to the monitor or output by the monitor are in hexadecimal.

7. Utility routines - This last section of software contains general purpose routines that are used by the other software modules. Included are the input/output and conversion routines.

The following pages contain memory maps, flowcharts, and listings of the device software. Software reference information should include the Intel Assembly Language Programming Manual.

FLOATING POINT REPRESENTATION



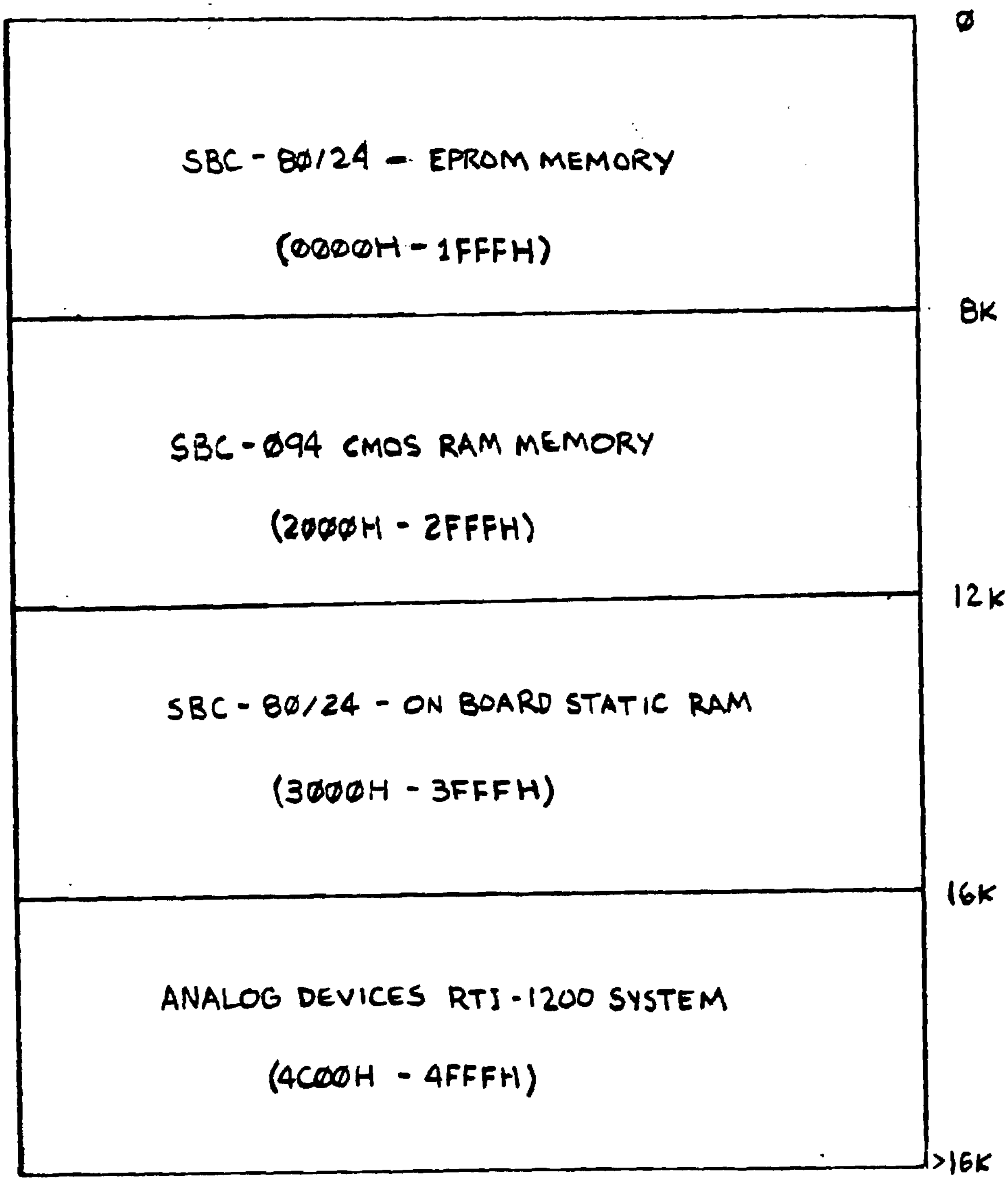
DATA STORAGE CAPACITY

1 DATA RECORD = 34 BYTES FOR TIME AND DATE STORAGE
+ 9 BYTES PER ANALOG SENSOR
+ 1 BYTE PER DIGITAL SENSOR

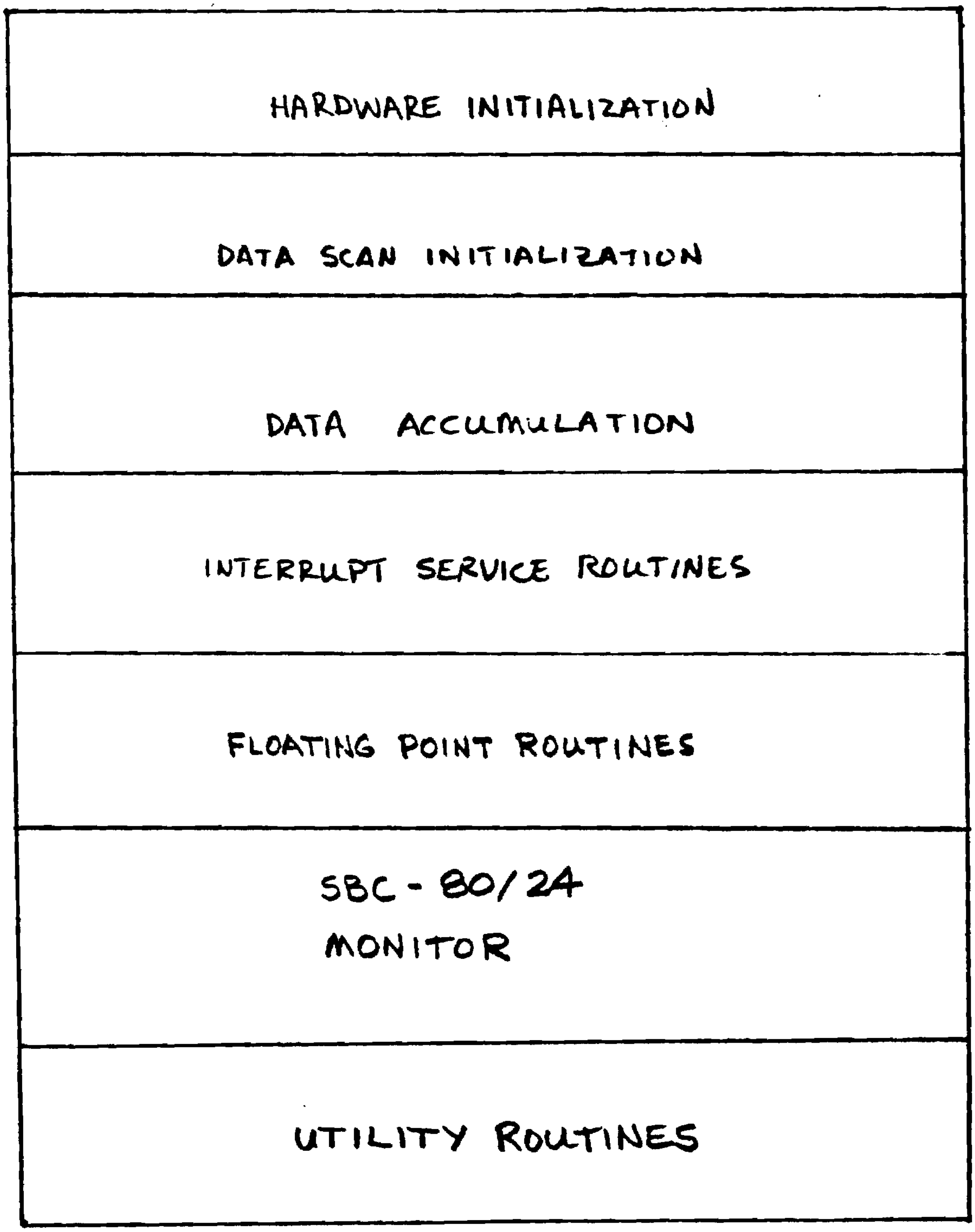
$$\text{NUMBER OF DATA RECORDS THAT CAN BE STORED} \cong \frac{3108}{(34 + 9(\text{NO. OF ANALOG SENSORS}) + (\text{NO. OF DIGITAL SENSORS}))}$$

$$\text{MAXIMUM TIME BETWEEN DATA RETRIEVALS} = \left[\begin{array}{l}\text{NUMBER OF DATA RECORDS}\\\text{THAT CAN BE STORED}\end{array}\right] \cdot \left[\begin{array}{l}\text{TIME INTERVAL}\\\text{OF DATA RECORD}\end{array}\right]$$

SYSTEM MEMORY MAP

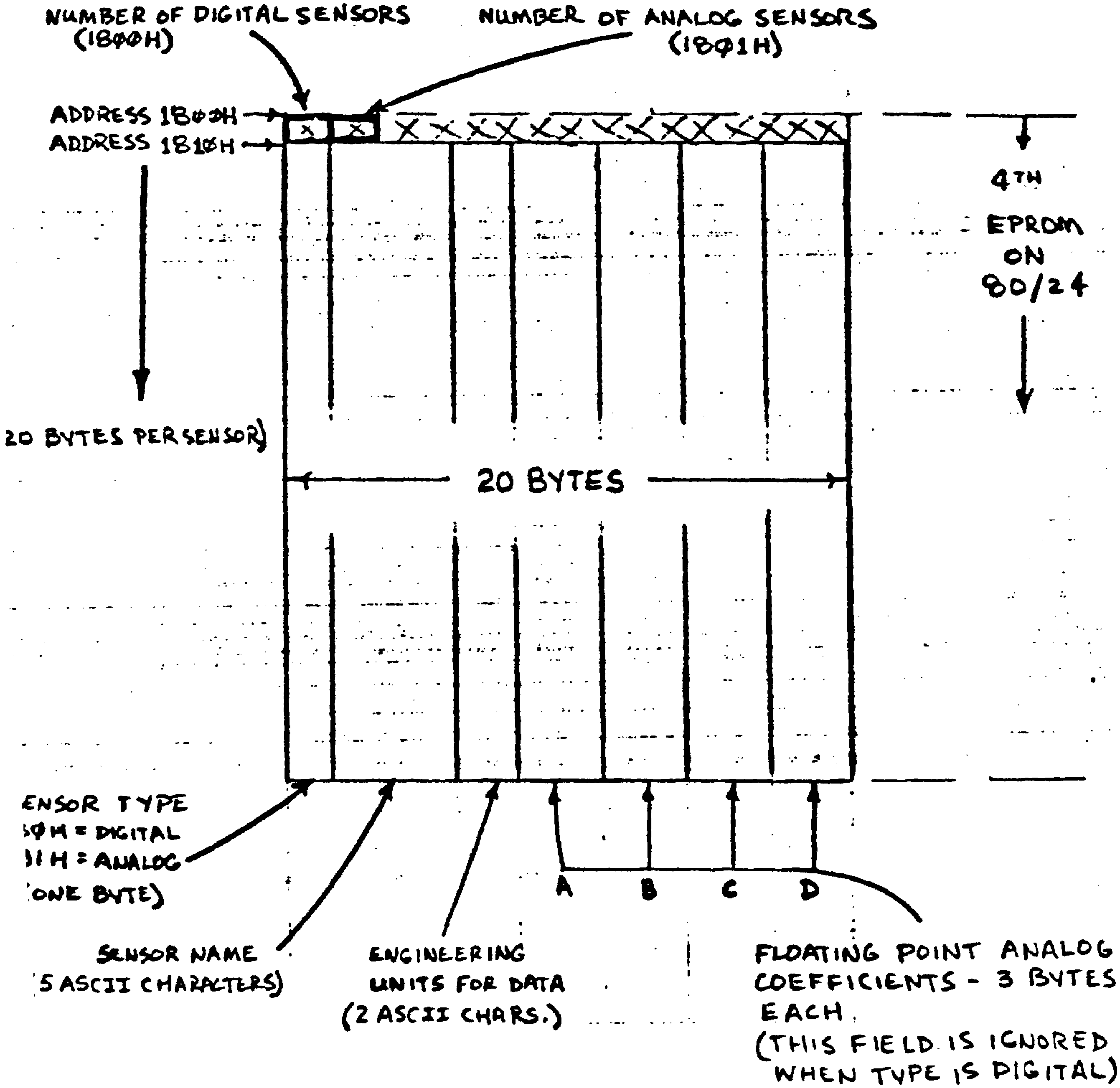


SOFTWARE MEMORY MAP

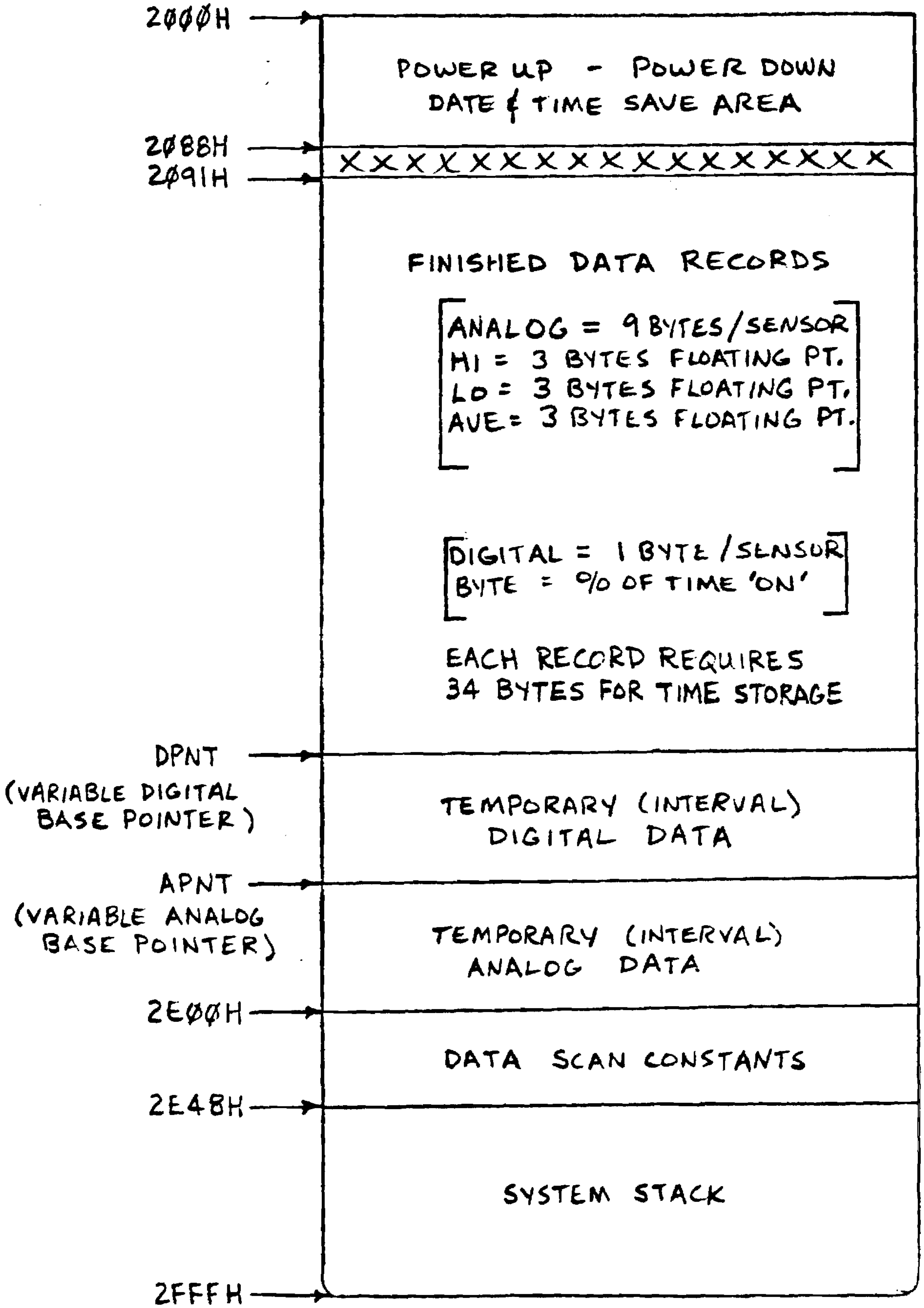


PROM TABLE

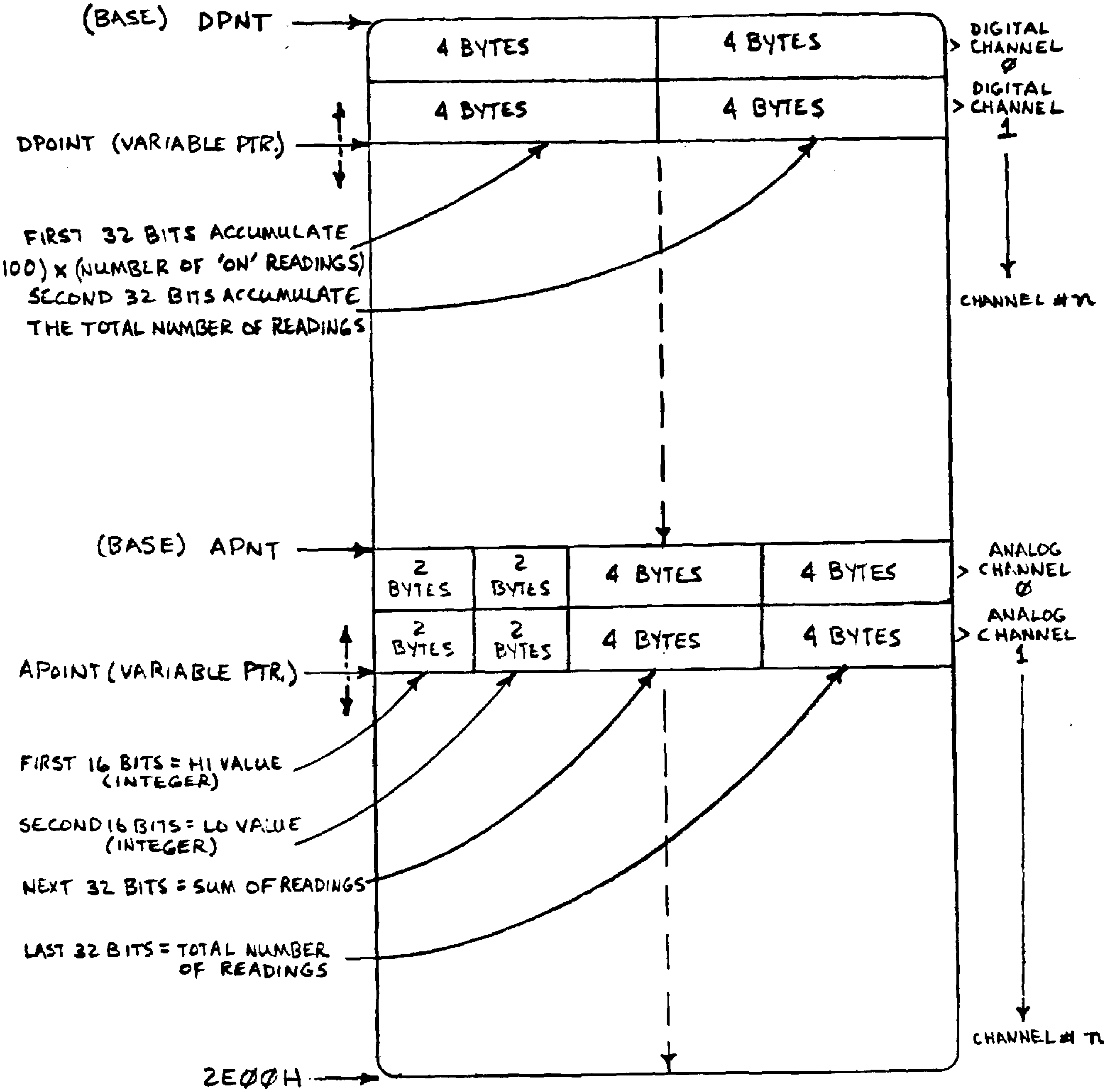
(CONTAINS SENSOR DATA)



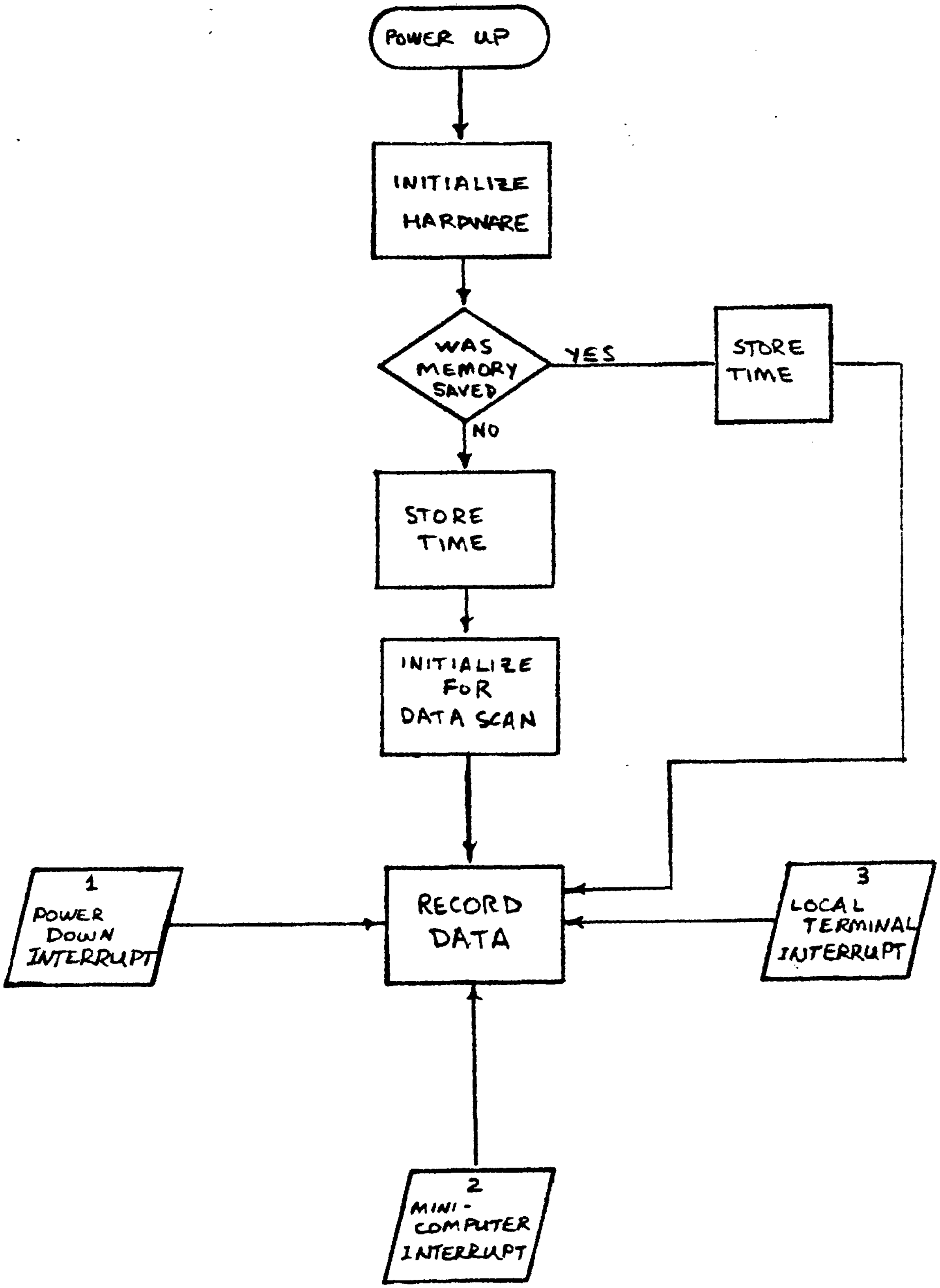
CMOS RAM MEMORY IMAGE



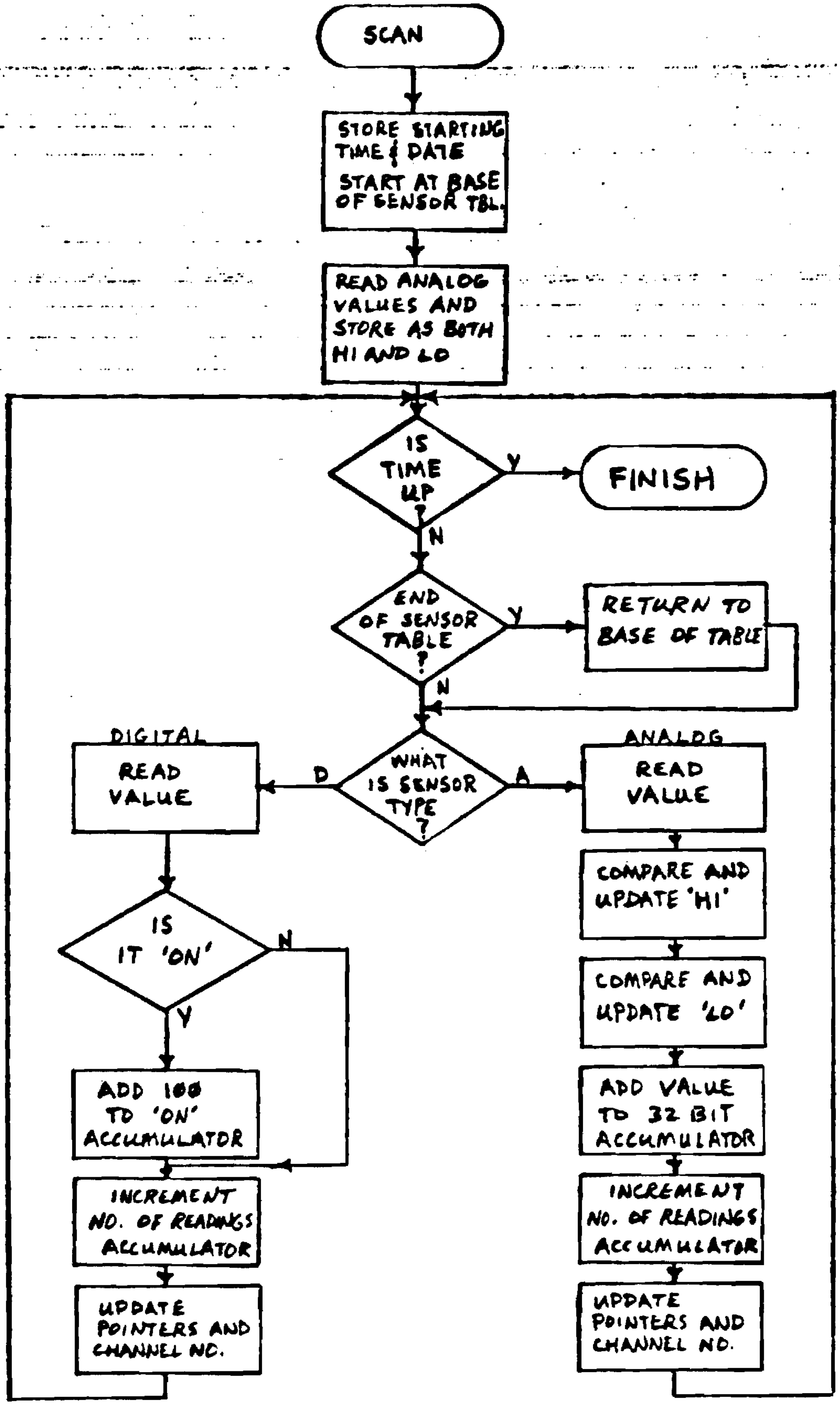
TEMPORARY (INTERVAL) DATA - MEMORY IMAGE



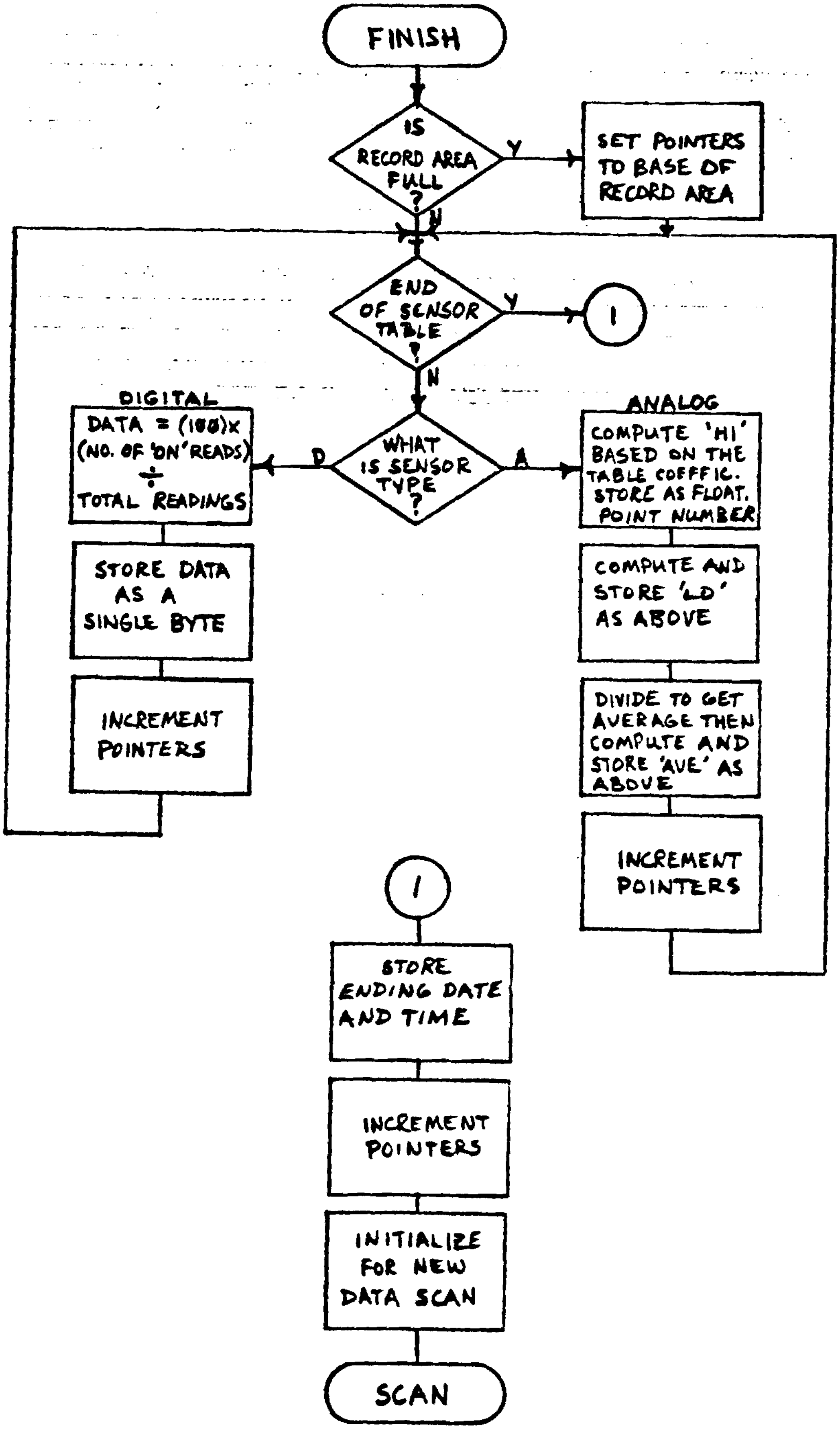
DATA GATHERING SYSTEM



DATA SCAN FOR INDIVIDUAL RECORDS (TEMPORARY INTERVAL DATA)

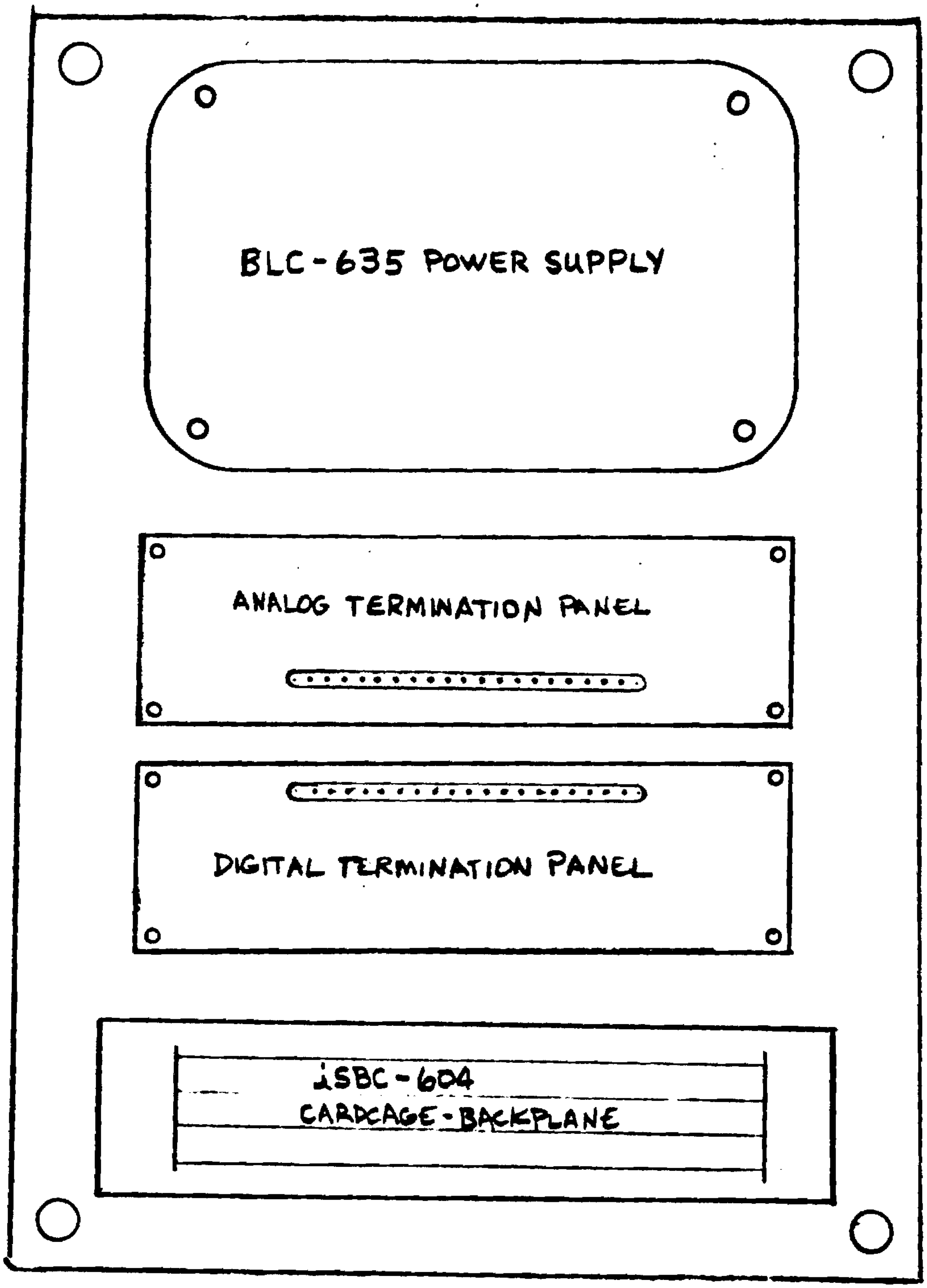


FINISH A DATA RECORD
(INTERPRET AND STORE TEMPORARY INTERVAL DATA)



DATA GATHERING SYSTEM

HARDWARE LAYOUT



```
                ;**************************************************
                ;
                ;       NCBC-CEL DATA GATHERING SYSTEM
                ;
                ;       AUGUST 1981
                ;
                ;**************************************************
                ;
                ;       SYSTEM EQUATES
                ;
                ;INTEL 80/24 DEVICE CONSTANTS:
                ;
                ;
                ;ON-BOARD 8253 PIT
                ;
00DF =          TIM1CP  EQU     0DFH    ;TIMER CONTROL PORT
00DE =          TIM1C2  EQU     0DEH    ;COUNTER 2(USART TIMER)
0038 =          B12     EQU     038H    ;RATE FACTOR FOR 1200 BPS
00B6 =          T1C2M3  EQU     0B6H    ;BYTE TO SET COUNTER 2 TO
                                        ;MODE 3 (RATE GENERATOR)
                ;
                ;MULTIMODULE 8253 PIT
                ;
00FB =          TIM2CP  EQU     0FBH    ;TIMER CONTROL PORT
00FA =          TIM2C2  EQU     0FAH    ;COUNTER 2(USART TIMER)
0040 =          BAUD12  EQU     040H    ;RATE FACTOR FOR 1200 BPS
00B6 =          T2C2M3  EQU     0B6H    ;BYTE TO SET COUNTER 2 TO
                                        ;MODE 3 (RATE GENERATOR)
                ;
                ;8251A USART EQUATES
                ;
                ;MODE INSTRUCTION
0002 =          BAUD16  EQU     002H    ;16X BAUD RATE FACTOR
0008 =          DBIT7   EQU     008H    ;7 DATA BITS
000C =          DBIT8   EQU     00CH    ;8 DATA BITS
0030 =          EVEN    EQU     030H    ;EVEN PARITY
0010 =          ODD     EQU     010H    ;ODD PARITY
0040 =          STOP1   EQU     040H    ;1 STOP BIT
                ;
                ;COMMAND INSTRUCTION
0001 =          TXEN    EQU     001H    ;TRANSMIT ENABLE
0002 =          DTR     EQU     002H    ;DATA TERMINAL READY
0004 =          RXEN    EQU     004H    ;RECEIVE ENABLE
0010 =          ERST    EQU     010H    ;ERROR RESET
0020 =          RTS     EQU     020H    ;REQUEST TO SEND
0040 =          URST    EQU     040H    ;INTERNAL RESET
                ;
                ;STATUS READ DEFINITION
0001 =          TXRDY   EQU     001H    ;TRANSMITTER READY
0002 =          RXRDY   EQU     002H    ;RECEIVER READY
0004 =          TXMT    EQU     004H    ;TRANSMITTER EMPTY
0038 =          RXERR   EQU     038H    ;RECEIVER ERROR
0080 =          DSR     EQU     080H    ;DATA SET READY
                ;
                ;ON-BOARD 8251A PCI (LOCAL TERMINAL)
00ED =          CONST   EQU     0EDH    ;CONSOLE STATUS/CONTROL
```

```
00EC =          CONSD   EQU     0ECH    ;CONSOLE DATA PORT
                ;
                ;MULTIMODULE 8251A PCI (MODEM)
00F1 =          MODMST  EQU     0F1H    ;MODEM STATUS/CONTROL
00F0 =          MODEM   EQU     0F0H    ;MODEM DATA PORT
                ;
                ;8255A PPI #1
                ;
00E4 =          PORTA1  EQU     0E4H    ;PORT A
00E5 =          PORTB1  EQU     0E5H    ;PORT B
00E6 =          PORTC1  EQU     0E6H    ;PORT C
00E7 =          CWR1    EQU     0E7H    ;CONTROL PORT #1
009B =          MODW1   EQU     09BH    ;MODE WORD TO INITIALIZE
                                        ;PORTS A, B, &C TO
                                        ;MODE 0 INPUTS
                ;
                ;8255A PPI #2
                ;
00E8 =          PORTA2  EQU     0E8H    ;PORT A
00E9 =          PORTB2  EQU     0E9H    ;PORT B
00EA =          PORTC2  EQU     0EAH    ;PORT C
00EB =          CWR2    EQU     0EBH    ;CONTROL PORT #2
0080 =          MODW2   EQU     080H    ;MOOE WORD TO INITIALIZE
                                        ;PORTS A, B, &C TO
                                        ;MODE 0 OUTPUTS
                ;
                ;8259A PCI
                ;
00D8 =          PIC     EQU     0D8H    ;ADDRESS OF 8259A
00D8 =          ICW1A   EQU     0D8H    ;ADDRESS TO WRITE FIRST
                                        ;INITIALIZATION BYTE (ICW1)
00D9 =          ICW2A   EQU     0D9H    ;ADDRESS TO WRITE SECOND
                                        ;INITIALIZATION BYTE (ICW2)
0056 =          ICW1    EQU     056H    ;SETS UP EDGE TRIGGERED
                                        ;INTERRUPTS, VECTOR TABLE
                                        ;INTERVAL OF 4, A7-A5=010
0000 =          ICW2    EQU     000H    ;A15-A8=00000000
0020 =          EOI     EQU     020H    ;END OF INTERRUPT INSTR.
                ;
                ;OTHER DEVICE CONSTANTS:
                ;
                ;DIGITAL PATHWAYS TCU-410 SYSTEM CLOCK
                ;
00A0 =          SECOND  EQU     0A0H    ;SECONDS COUNTER
00A1 =          MINUTE  EQU     0A1H    ;MINUTE COUNTER
00A2 =          HOUR    EQU     0A2H    ;HOUR COUNTER
00A3 =          DAY     EQU     0A3H    ;DAY COUNTER
00A4 =          MONTH   EQU     0A4H    ;MONTH COUNTER
00A5 =          YEAR    EQU     0A5H    ;YEAR COUNTER
00A1 =          SETTIM  EQU     0A1H    ;OUTPUT TO SET TIME
00A3 =          SETDAT  EQU     0A3H    ;OUTPUT TO SET DATE
00A6 =          CLSTAT  EQU     0A6H    ;INPUT CLOCK STATUS
00A6 =          RFC     EQU     0A6H    ;RESET FAST CLOCK
                ;
                ;ANALOG DEVICES RTI-1200 ANALOG INPUT SYSTEM
                ;
```

```
4FFE =          ADCHI    EQU    4FFEH    ;HIGH BYTE DATA
4FFD =          ADCLO    EQU    4FFDH    ;LO BYTE DATA
4FFC =          ASTAT    EQU    4FFCH    ;STATUS BYTE
4FFB =          CNVCMD   EQU    4FFBH    ;BEGIN CONVERSION COMMAND
4FFA =          MUXADR   EQU    4FFAH    ;ANALOG CHANNEL SELECT
4FF9 =          GNSEL    EQU    4FF9H    ;GAIN SELECT
4FF0 =          SETUP    EQU    4FF0H    ;SETUP BYTE
                ;
                ;
                ;**************************************************
                ;
                ;RESTART AND INTERRUPT JUMP TABLE
                ;
                ;**************************************************
                ;
0000                     ORG    0
0000 F3         RST0:    DI              ;THIS TABLE PROVIDES
0001 C35F00              JMP    INIT     ;THE JUMPS NECESSARY
0008                     ORG    8        ;TO ENTER RESTART OR
0008 C30000     RST1:    JMP    0        ;INTERRUPT SERVICE
0010                     ORG    16       ;ROUTINES.
0010 C30000     RST2:    JMP    0
0018                     ORG    24       ;IF AN UNEXPECTED
0018 C30000     RST3:    JMP    0        ;RESTART OR INTERRUPT
0020                     ORG    32       ;OCCURS, THE SYSTEM
0020 C30000     RST4:    JMP    0        ;WILL BE RE-INITIALIZED.
0024                     ORG    36
0024 C30000     TRAP:    JMP    0
0028                     ORG    40
0028 C30000     RST5:    JMP    0
002C                     ORG    44
002C C30000     RST55:   JMP    0
0030                     ORG    48
0030 C30000     RST6:    JMP    0
0034                     ORG    52
0034 C30000     RST65:   JMP    0
0038                     ORG    56
0038 C30000     RST7:    JMP    0
003C                     ORG    60
003C C30000     RST75:   JMP    0
                ;
0040                     ORG    64       ;POWER FAIL INTERRUPT
0040 C31206     IR0:     JMP    INT0     ;(HIGHEST PRIORITY)
0044                     ORG    68       ;ROUTINE TO SERVICE
0044 C33906     IR1:     JMP    INT1     ;MINICOMPUTER COMMUNICATION
0048                     ORG    72       ;ROUTINE TO SERVICE
0048 C38806     IR2:     JMP    INT2     ;LOCAL TERMINAL
004C                     ORG    76
004C C30000     IR3:     JMP    0        ;NOT USED
0050                     ORG    80
0050 C30000     IR4:     JMP    0        ;NOT USED
0054                     ORG    84
0054 C30000     IR5:     JMP    0        ;NOT USED
0058                     ORG    88
0058 C30000     IR6:     JMP    0        ;NOT USED
005C                     ORG    92
```

```
005C C30000     IR7:    JMP     0           ;NOT USED
                ;
                ;------------------------------------
                ;INITIALIZATION OF DATA GATHERING SYSTEM
                ;------------------------------------
                ;
                ;FUNCTION: INIT
                ;DESCRIPTION: THIS MODULE INITIALIZES ALL OF THE
                ;             PROGRAMMABLE DEVICES ON POWER UP.
                ;
005F D3D4       INIT:   OUT     0D4H        ;RESET POWER FAIL LATCH
                ;
                ;INITIALIZE 8253 INTERVAL TIMERS
                ;
0061 F3                 DI                          ;DISABLE INTERRUPTS
0062 3EB6               MVI     A,T1C2M3            ;USE ON BOARD 8253
0064 D3DF               OUT     TIM1CP
0066 3E38               MVI     A,B12               ;FOR 1200 BPS CLOCK
0068 D3DE               OUT     TIM1C2              ;(COUNTER 2 IS SET
006A 3E00               MVI     A,0                 ;TO MODE 3)
006C D3DE               OUT     TIM1C2
                ;
006E 3EB6               MVI     A,T2C2M3            ;USE MULTIMODULE
0070 D3FB               OUT     TIM2CP              ;8253 FOR 1200 BPS
0072 3E40               MVI     A,BAUD12            ;CLOCK FOR MODEM
0074 D3FA               OUT     TIM2C2              ;USART (COUNTER 2
0076 3E00               MVI     A,0                 ;IS SET TO MODE 3)
0078 D3FA               OUT     TIM2C2
                ;
                ;INITIALIZE 8251A USARTS
                ;
007A D3ED               OUT     CONST               ;GET TERMINAL USART
007C D3ED               OUT     CONST               ;INTO A KNOWN STATE
007E D3ED               OUT     CONST
0080 3E40               MVI     A,URST
0082 D3ED               OUT     CONST               ;RESET USART
0084 3E4E               MVI     A,BAUD16+DBIT8+STOP1    ;DATA FORMAT
0086 D3ED               OUT     CONST
0088 3E37               MVI     A,TXEN+RXEN+RTS+DTR+ERST;COMMAND
008A D3ED               OUT     CONST               ;INSTRUCTION
008C DBEC               IN      CONSD               ;CLEAR DATA PORT
                ;
008E D3F1               OUT     MODMST              ;GET MODEM USART
0090 D3F1               OUT     MODMST              ;INTO A KNOWN STATE
0092 D3F1               OUT     MODMST
0094 3E40               MVI     A,URST
0096 D3F1               OUT     MODMST              ;RESET USART
0098 3E4E               MVI     A,BAUD16+DBIT8+STOP1    ;DATA FORMAT
009A D3F1               OUT     MODMST
009C 3E37               MVI     A,TXEN+RXEN+ERST+RTS+DTR ;COMMAND
009E D3F1               OUT     MODMST
00A0 DBF0               IN      MODEM               ;CLEAR DATA PORT
                ;
                ;INITIALIZE 8255A PARALLEL PORTS
                ;
00A2 3E9B               MVI     A,MODW1             ;SET UP 8255A #1 FOR
```

```
00A4 D3E7           OUT    CWR1             ;MODE 0 INPUTS
               ;
00A6 3E80           MVI    A,MODW2          ;SET UP 8255A #2 FOR
00A8 D3EB           OUT    CWR2             ;MODE 0 OUTPUTS
               ;
               ;INITIALIZE 8259A INTERRUPT CONTROLLER
               ;
00AA 3E56           MVI    A,ICW1           ;USE 8259 IN THE
00AC D3D8           OUT    ICW1A            ;FULLY NESTED MODE
00AE 3E00           MVI    A,ICW2           ;WITH CALL ADDRESS
00B0 D3D9           OUT    ICW2A            ;STARTING AT 40H
               ;
               ;INITIALIZE RTI-120D ANALOG INPUT SYSTEM
               ;
00B2 3E00           MVI    A,0              ;SET UP FOR A GAIN
00B4 32F94F         STA    GNSEL            ;OF 1 AND DISABLE
00B7 32F04F         STA    SETUP            ;INTERRUPTS & PACER
               ;
               ;NOW WE MUST DETERMINE THE STATE OF THE SYSTEM
               ;AND START UP PROPERLY.
               ;
00BA 3A2A2E         LDA    CHKBYT           ;CHECK TO SEE IF RAM
00BD FE55           CPI    55H              ;WAS SAVED
00BF CAE400         JZ     MGOOD            ;IF SO, RESTORE
00C2 31F02F    MBAD: LXI   SP,2FF0H         ;IF NOT LOAD SP
00C5 0E20           MVI    C,' '
00C7 210020         LXI    H,2000H          ;FILL POWER UP TIME
00CA 118820         LXI    D,2088H          ;SAVE AREA WITH
00CD 71        MB1: MOV    M,C              ;ASCII SPACES
00CE CDA910         CALL   HILO
00D1 D2CD00         JNC    MB1
00D4 210020         LXI    H,2000H          ;SAVE TIME
00D7 CD1E11         CALL   SAVTIM
00DA 211120         LXI    H,2011H          ;STORE POINTER
00DD 222B2E         SHLD   PUPSAV
00E0 FB             EI                      ;ENABLE INTERRUPTS
00E1 C31F01         JMP    SCINIT
                                            ;THEN BEGIN DATA
                                            ;SCAN
00E4 2A2D2E    MGOOD: LHLD SPSAV            ;MEMORY WAS SAVED SO
00E7 F9             SPHL                    ;RECOVER THE STACK
00E8 DBA1           IN     MINUTE           ;SCAN THIS RECORD FOR
00EA 3C             INR    A                ;ONLY ONE MORE MINUTE
00EB 320F2E         STA    TIMUP
00EE 2A2B2E         LHLD   PUPSAV           ;NOW SAVE THE
                                            ;TIME IN DATA TABLE
00F1 7D             MOV    A,L              ;FIRST SEE IF TABLE
00F2 FE88           CPI    88H              ;FOR TIME SAVE IS FULL
00F4 D2FD00         JNC    MG1
00F7 CAFD00         JZ     MG1
00FA C30C01         JMP    MG2
00FD 210020    MG1: LXI    H,2000H          ;YES FULL
0100 CD1E11         CALL   SAVTIM
0103 211120         LXI    H,2011H
0106 222B2E         SHLD   PUPSAV
0109 C31901         JMP    MG3
```

```
010C CD1E11     MG2:    CALL    SAVTIM          ;NO - MORE SPACE
010F 2A2B2E             LHLD    PUPSAV
0112 111100             LXI     D,17
0115 19                 DAD     D               ;INCREMENT POINTER
0116 222B2E             SHLD    PUPSAV
0119 E1         MG3:    POP     H               ;RESTORE THE STATE
011A D1                 POP     D               ;OF THE SYSTEM AND
011B C1                 POP     B               ;CONTINUE
011C F1                 POP     PSW
011D FB                 EI                      ;ENABLE INTERRUPTS
011E C9                 RET
                ;
                ;
                ;       DATA SCAN EQUATES
                ;
                ;
2E00 =          APNT    EQU     2E00H   ;BASE OF ANALOG SCRATCH AREA
2E02 =          APOINT  EQU     2E02H   ;VARIABLE ANALOG POINTER
2E04 =          DPNT    EQU     2E04H   ;BASE OF DIGITAL SCRATCH AREA
2E06 =          DPOINT  EQU     2E06H   ;VARIABLE DIGITAL POINTER
2E08 =          ACHNM   EQU     2E08H   ;ANALOG CHANNEL NUMBER
2E09 =          DCHNM   EQU     2E09H   ;DIGITAL CHANNEL NUMBER
2E0A =          NUMAN   EQU     2E0AH   ;#ANALOG CHANNELS - 1
2E0B =          NUMDIG  EQU     2E0BH   ;#DIGITAL CHANNELS - 1
2E0C =          CASUS   EQU     2E0CH   ;CURRENT AN. CH. UNDER SCAN
2E0D =          CDSUS   EQU     2E0DH   ;CURRENT DI. CH. UNDER SCAN
2E0E =          TIMLIM  EQU     2E0EH   ;DURATION OF SCAN IN MINUTES
2E0F =          TIMUP   EQU     2E0FH   ;MINUTE VALUE WHEN TIME IS UP
2E10 =          IPFL    EQU     2E10H   ;IN PROGRESS FLAG
2E11 =          STSEC   EQU     2E11H   ;VALUE OF SECONDS AT STARTUP
2E12 =          RECOV   EQU     2E12H   ;RECORD OVERFLOW ADDRESS
2E16 =          CRECBS  EQU     2E16H   ;BASE OF CURRENT RECORD
2E18 =          TNOS    EQU     2E18H   ;TOTAL NUMBER OF SENSORS
2E19 =          TBLOV   EQU     2E19H   ;PROM TABLE OVERFLOW VALUE
2E1B =          SNBYT   EQU     2E1BH   ;CONVERSION SIGN FLAG
2E1C =          INDEX   EQU     2E1CH   ;INDEX FOR DECIMAL CONVERSION
2E1E =          DIVS    EQU     2E1EH   ;ADDRESS OF 32 BIT DIVISOR
2E20 =          DIVD    EQU     2E20H   ;ADDRESS OF 32 BIT DIVIDEND
2E22 =          REM     EQU     2E22H   ;DIVISION SCRATCHPAD
2E26 =          REMSV   EQU     2E26H   ;    "          "
2E2A =          CHKBYT  EQU     2E2AH   ;RAM CHECK LOCATION
2E2B =          PUPSAV  EQU     2E2BH   ;ADDRESS FOR POWER-UP TIME
2E2D =          SPSAV   EQU     2E2DH   ;PLACE FOR POWERDOWN STACK P.
2E2F =          TMPSM   EQU     2E2FH   ;4 BYTE SCRATCH AREA
2E33 =          FBYT1   EQU     2E33H   ;4 BYTE 1 = 00000001
2E37 =          FB100   EQU     2E37H   ;4 BYTE 100 = 00000064
2E3B =          TADDR   EQU     2E3BH   ;CURRENT PROM TABLE ADDRESS
2E3D =          TMPVAL  EQU     2E3DH   ;FLOATING POINT SCRATCH AREA
2E40 =          LRADD   EQU     2E40H   ;UPPER ADDRESS OF RECORDS
2E42 =          FBFLAG  EQU     2E42H   ;FULL DATA RECORD INDICATOR
2E43 =          CRBSAV  EQU     2E43H   ;SAVE AREA FOR CRECBS
2E45 =          TADSAV  EQU     2E45H   ;SAVE AREA FOR TADDR
2E47 =          IOBYTE  EQU     2E47H   ;OUTPUT TYPE INDICATOR
                ;
                ;INITIALIZATION OF DATA SCAN
                ;
```

```
011F 210018    SCINIT: LXI   H,1800H ;COMPUTE TOTAL NUMBER OF
0122 7E                MOV   A,M     ;SENSORS AND STORE IN RAM
0123 23                INX   H
0124 86                ADD   M
0125 32182E            STA   TNOS
0128 111400            LXI   D,20    ;NOW CALCULATE AND STORE
012B 2600              MVI   H,0     ;TABLE OVERFLOW VALUE
012D 6F                MOV   L,A     ;TBLOV=(TNOS*20-1)+1810H
012E CD860B            CALL  IMULT   ;HL=TNOS*20
0131 2B                DCX   H       ;HL=TNOS*20-1
0132 111018            LXI   D,1810H
0135 19                DAD   D       ;HL=TBLOV
0136 22192E            SHLD  TBLOV
0139 210018            LXI   H,1800H ;CALCULATE AND STORE
013C 7E                MOV   A,M     ;NUMDIG & NUMAN
013D 3D                DCR   A
013E 320B2E            STA   NUMDIG
0141 23                INX   H
0142 7E                MOV   A,M
0143 3D                DCR   A
0144 320A2E            STA   NUMAN
0147 3E0F              MVI   A,15    ;STORE TIME LIMIT AS
0149 320E2E            STA   TIMLIM  ;15 MINUTES
014C 3E00              MVI   A,0     ;INITIALIZE IN PROGRESS
014E 32102E            STA   IPFL    ;FLAG, CASUS, CDSUS,
0151 320C2E            STA   CASUS   ;& FBFLAG TO ZERO
0154 320D2E            STA   CDSUS
0157 32422E            STA   FBFLAG
015A 219120            LXI   H,2091H ;INITIALIZE BASE ADDRESS
015D 22162E            SHLD  CRECBS  ;OF RECORD STORAGE
0160 210118            LXI   H,1801H ;CALCULATE APNT & DPNT
0163 7E                MOV   A,M     ;BASE POINTERS
0164 2600              MVI   H,0     ;APNT=2E00H-(#ANAL.*12)
0166 6F                MOV   L,A
0167 110C00            LXI   D,12
016A CD860B            CALL  IMULT   ;HL=#ANALOG*12
016D CDE10B            CALL  NEGD    ;HL= -(#ANALOG*12)
0170 11002E            LXI   D,2E00H
0173 19                DAD   D       ;HL=APNT
0174 22002E            SHLD  APNT    ;STORE VALUE
0177 210018            LXI   H,1800H
017A 7E                MOV   A,M     ;DPNT=APNT-(#DIG*8)
017B 2600              MVI   H,0
017D 6F                MOV   L,A
017E 110800            LXI   D,8
0181 CD860B            CALL  IMULT   ;HL=#DIGITAL*8
0184 CDE10B            CALL  NEGD    ;HL= -(#DIGITAL*8)
0187 EB                XCHG          ;MOVE TO DE
0188 2A002E            LHLD  APNT
018B 19                DAD   D       ;HL=DPNT
018C 22042E            SHLD  DPNT    ;STORE VALUE
018F 5D                MOV   E,L     ;CALCULATE RECORD OVERFLOW
0190 54                MOV   D,H     ;ADDRESS AND STORE
0191 21C500            LXI   H,197   ;RECOV=DPNT-197
0194 CDE10B            CALL  NEGD
0197 19                DAD   D
```

```
0198 22122E            SHLD    RECOV
019B 3E01              MVI     A,1      ;INITIALIZE FBYT1
019D 21332E            LXI     H,FBYT1
01A0 77                MOV     M,A
01A1 3E00              MVI     A,0
01A3 23                INX     H
01A4 77                MOV     M,A
01A5 23                INX     H
01A6 77                MOV     M,A
01A7 23                INX     H
01A8 77                MOV     M,A
01A9 3E64              MVI     A,100    ;INITIALIZE FB100
01AB 21372E            LXI     H,FB100
01AE 77                MOV     M,A
01AF 3E00              MVI     A,0
01B1 23                INX     H
01B2 77                MOV     M,A
01B3 23                INX     H
01B4 77                MOV     M,A
01B5 23                INX     H
01B6 77                MOV     M,A
01B7 3E00              MVI     A,0      ;SET IOBYTE FOR CONSOLE
01B9 32472E            STA     IOBYTE
01BC 2A042E            LHLD    DPNT     ;ZERO THE DATA
01BF 11FF2D            LXI     D,2DFFH  ;SCRATCHPAD AREA
01C2 0E00              MVI     C,0
01C4 71        SC1:    MOV     M,C
01C5 CDA910            CALL    HILO
01C8 D2C401            JNC     SC1
               ;
               ;
               ;**************************************************
               ;
               ;       MAIN DATA SCAN LOOP
               ;
               ;**************************************************
               ;
01CB 3A102E    SCAN:   LDA     IPFL     ;LOAD IN PROGRESS FLAG
01CE 87                ADD     A        ;CONTINUE IF IN PROGRESS
01CF C27302            JNZ     CONT     ;IF NOT START UP NEW SCAN
01D2 2A002E            LHLD    APNT     ;MOVE ANALOG AND DIGITAL
01D5 22022E            SHLD    APOINT   ;SCRATCH POINTERS TO WORK
01D8 2A042E            LHLD    DPNT     ;AREA
01DB 22062E            SHLD    DPOINT
01DE 3E00              MVI     A,0      ;NOW ZERO THE CHANNEL
01E0 32092E            STA     DCHNM    ;NUMBERS
01E3 32082E            STA     ACHNM
01E6 211018            LXI     H,1810H  ;LOAD TABLE POINTER
01E9 223B2E            SHLD    TADDR
01EC 2A192E    SCAN1:  LHLD    TBLOV    ;INITIALIZE ANALOG HI-LO
01EF EB                XCHG             ;VALUES BY STORING THE
01F0 2A3B2E            LHLD    TADDR    ;CURRENT ANALOG VALUES
01F3 CDE10B            CALL    NEGD
01F6 19                DAD     D
01F7 7C                MOV     A,H
01F8 17                RAL
```

```
01F9 D2FF01             JNC     SCAN2
01FC C33E02             JMP     SCAN4
D1FF 2A3B2E    SCAN2:   LHLD    TADDR     ;LOOK IN TABLE FDR
0202 7E                 MOV     A,M       ;ANALDG SENSORS
D203 FE3D               CPI     30H
D2D5 CA34D2             JZ      SCAN3     ;NOT ANALDG
0208 CDC5D3             CALL    GETAV     ;GET VALUE
D20B EB                 XCHG
D2DC 2A022E             LHLD    APDINT    ;STORE IN SCRATCH AREA
020F 73                 MDV     M,E
0210 23                 INX     H
D211 72                 MOV     M,D
0212 23                 INX     H
D213 73                 MOV     M,E
0214 23                 INX     H
D215 72                 MOV     M,D
0216 2A022E             LHLD    APDINT
0219 110C00             LXI     D,12
021C 19                 DAD     D
D21D 22D22E             SHLD    APOINT    ;UPDATE APOINT
0220 2A3B2E             LHLD    TADDR
0223 111400             LXI     D,20
D226 19                 DAD     D
0227 223B2E             SHLD    TADDR     ;UPDATE TADDR
D22A 3A082E             LDA     ACHNM
D22D 3C                 INR     A         ;INCREMENT THE CHANNEL
022E 32082E             STA     ACHNM
0231 C3EC01             JMP     SCAN1
D234 111400    SCAN3:   LXI     D,20      ;INCREMENT TADDR
0237 19                 DAD     D
0238 223B2E             SHLD    TADDR
023B C3EC01             JMP     SCAN1
023E 211018    SCAN4:   LXI     H,1810H   ;REINITIALIZE
0241 223B2E             SHLD    TADDR
0244 3E00               MVI     A,0
0246 32082E             STA     ACHNM
0249 2A002E             LHLD    APNT
024C 22022E             SHLD    APOINT
024F 2A162E             LHLD    CRECBS    ;NOW STORE START UP TIME
0252 CD1E11             CALL    SAVTIM    ;IN MEMORRY
0255 3E01               MVI     A,1       ;SET IN PROGRESS FLAG
0257 32102E             STA     IPFL
025A DBA1               IN      MINUTE    ;NOW CALCULATE TIMUP
025C 47                 MOV     B,A
025D 3A0E2E             LDA     TIMLIM
0260 80                 ADD     B
0261 FE3B               CPI     59        ;IF RESULT IS > 59 THEN
0263 D26902             JNC     SUB60     ;TIMUP=TIMUP-60
0266 C36B02             JMP     STORT     ;STORE VALUE
0269 DE3C      SUB60:   SBI     60
D26B 320F2E    STORT:   STA     TIMUP
026E DBA0               IN      SECOND    ;STORE STARTING SECONDS
0270 32112E             STA     STSEC
               ;
0273 DBA1      CONT:    IN      MINUTE    ;SEE IF DATA SCAN TIME IS UP
0275 47                 MOV     B,A
```

```
0276 3A0F2E              LDA    TIMUP
0279 B8                  CMP    B        ;COMPARE WITH ENDING MINUTES
027A C29002              JNZ    CONT1    ;MORE TIME LEFT
027D DBAD                IN     SECOND
027F 47                  MOV    B,A
0280 3A112E              LDA    STSEC    ;COMPARE WITH ENDING SECONDS
0283 B8                  CMP    B
0284 CA8D02              JZ     NOTIME
0287 DA8D02              JC     NOTIME   ;TIME IS UP
028A C39002              JMP    CONT1    ;MORE TIME LEFT
028D C328D4    NOTIME:   JMP    FINISH
0290 2A192E    CONT1:    LHLD   TBLOV    ;SEE IF WE HAVE REACHED THE
0293 5D                  MOV    E,L      ;END OF THE SENSOR TABLE
0294 54                  MOV    D,H
0295 2A3B2E              LHLD   TADDR
0298 CDE10B              CALL   NEGD
029B 19                  DAD    D
029C 7C                  MOV    A,H
029D 17                  RAL
029E D2A702              JNC    CONT2    ;NO
02A1 211018              LXI    H,1810H  ;YES, REINITIALIZE TABLE
02A4 223B2E              SHLD   TADDR    ;TO STARTING ADDRESS
02A7 2A3B2E    CONT2:    LHLD   TADDR    ;GET THE SENSOR TYPE
02AA 7E                  MOV    A,M
02AB FE30                CPI    30H
02AD CAF302              JZ     DIGSR    ;SENSOR IS DIGITAL
02B0 3A0A2E    ANSR:     LDA    NUMAN    ;MUST BE ANALOG
02B3 47                  MOV    B,A      ;DETERMINE PROPER CHANNEL
02B4 3A0C2E              LDA    CASUS    ;ADDRESS
02B7 B8                  CMP    B
02B8 CACF02              JZ     CONT3
02BB DACF02              JC     CONT3    ;CHANNEL IS VALID
02BE 3E00                MVI    A,0      ;NOT VALID SET TO ZERO
02C0 320C2E              STA    CASUS
02C3 32082E              STA    ACHNM
02C6 2A002E              LHLD   APNT     ;REINITIALIZE SCRATCH
02C9 22022E              SHLD   APOINT   ;POINTER
02CC C3D202              JMP    CONT4
02CF 32082E    CONT3:    STA    ACHNM    ;ACHNM = CASUS
02D2 CD3603    CONT4:    CALL   AUPDT    ;UPDATE ANALOG VALUE
02D5 3A0C2E              LDA    CASUS    ;INCREMENT CURRENT CHANNEL
02D8 3C                  INR    A        ;NUMBER, UPDATE SCRATCH
02D9 320C2E              STA    CASUS    ;POINTER & TABLE ADDRESS
02DC 2A022E              LHLD   APOINT
02DF 110C00              LXI    D,12
02E2 19                  DAD    D
02E3 22022E              SHLD   APOINT
02E6 2A3B2E              LHLD   TADDR
02E9 111400              LXI    D,20
02EC 19                  DAD    D
02ED 223B2E              SHLD   TADDR
02F0 C3CB01              JMP    SCAN     ;KEEP SCANNING
02F3 3A0B2E    DIGSR:    LDA    NUMDIG   ;DIGITAL SCAN UPDATE
02F6 47                  MOV    B,A      ;DETERMINE PROPER CHANNEL
02F7 3A0D2E              LDA    CDSUS    ;ADDRESS
02FA B8                  CMP    B
```

```
02FB CA1203            JZ      CONT5
02FE DA1203            JC      CONT5    ;VALID ADDRESS
0301 3E00              MVI     A,0      ;NOT VALID, SET TO ZERO
0303 320D2E            STA     CDSUS
0306 32092E            STA     DCHNM
0309 2A042E            LHLD    DPNT     ;INITIALIZE DIGITAL SCRATCH
030C 22062E            SHLD    DPOINT   ;POINTER
030F C31503            JMP     CONT6
0312 32092E   CONT5:   STA     DCHNM    ;DCHNM = CDSUS
0315 CDD903   CONT6:   CALL    DUPDT    ;UPDATE DIGITAL VALUE
0318 3A0D2E            LDA     CDSUS    ;INCREMENT CURRENT CHANNEL
031B 3C                INR     A        ;NUMBER, UPDATE DIGITAL
031C 320D2E            STA     CDSUS    ;SCRATCH POINTER AND TABLE
031F 2A062E            LHLD    DPOINT   ;ADDRESS
0322 110800            LXI     D,8
0325 19                DAD     D
0326 22062E            SHLD    DPOINT
0329 2A3B2E            LHLD    TADDR
032C 111400            LXI     D,20
032F 19                DAD     D
0330 223B2E            SHLD    TADDR
0333 C3CB01            JMP     SCAN     ;KEEP SCANNING
                      ;
                      ;ANALOG DATA ACCUMULATION ROUTINE - READS RTI-1200
                      ;DATA AND UPDATES INTERMEDIATE SCRATCHPAD AREA
                      ;
0336 CDC503   AUPDT:   CALL    GETAV    ;GET ANALOG VALUE IN HL
0339 E5                PUSH    H        ;SAVE VALUE
033A CD710B            CALL    FLOAT    ;CONVERT TO FP AND SAVE
033D E5                PUSH    H
033E C5                PUSH    B
033F 2A022E            LHLD    APOINT   ;POINT TO PREVIOUS HI VALUE
0342 7E                MOV     A,M      ;GET VALUE
0343 47                MOV     B,A
0344 23                INX     H
0345 7E                MOV     A,M
0346 68                MOV     L,B
0347 67                MOV     H,A
0348 CD710B            CALL    FLOAT    ;CONVERT TO FP
034B EB                XCHG
034C 79                MOV     A,C      ;SAVE EXPONENT
034D C1                POP     B        ;RECOVER OTHER VALUE
034E E1                POP     H
034F 47                MOV     B,A      ;REPLACE EXPONENT
0350 CD170C            CALL    FPCMP    ;COMPARE VALUES - IF CY=1
0353 DA5903            JC      AUP1     ;THEN NEW VALUE IS LARGER
0356 C36403            JMP     AUP2
0359 C1       AUP1:    POP     B        ;GET VALUE AND SAVE
035A C5                PUSH    B        ;IN MEMORY
035B 2A022E            LHLD    APOINT   ;STORE AS NEW HI VALUE
035E EB                XCHG
035F 79                MOV     A,C
0360 12                STAX    D
0361 13                INX     D
0362 78                MOV     A,B
0363 12                STAX    D
```

```
0364 E1        AUP2:  POP    H        ;NOW COMPARE WITH LO VALUE
0365 E5               PUSH   H        ;GET VALUE
0366 CD710B           CALL   FLOAT    ;CONVERT TO FP AND SAVE
0369 E5               PUSH   H
036A C5               PUSH   B
036B 2A022E           LHLD   APOINT   ;GET CURRENT LO VALUE
036E 23               INX    H
036F 23               INX    H
0370 7E               MOV    A,M
0371 47               MOV    B,A
0372 23               INX    H
0373 7E               MOV    A,M
0374 68               MOV    L,B
0375 67               MOV    H,A
0376 CD710B           CALL   FLOAT    ;CONVERT
0379 EB               XCHG
037A 79               MOV    A,C      ;SAVE EXPONENT
037B C1               POP    B
037C E1               POP    H
037D 47               MOV    B,A      ;REPLACE EXPONENT
037E CD170C           CALL   FPCMP    ;COMPARE VALUES - IF CY=0
0381 D28703           JNC    AUP3     ;NEW VALUE<=PREVIOUS LO
0384 C39703           JMP    AUP4
0387 CA9703    AUP3:  JZ     AUP4     ;VALUES WERE EQUAL IF Z SET
038A C1               POP    B
038B C5               PUSH   B        ;GET VALUE & SAVE
038C 2A022E           LHLD   APOINT   ;STORE NEW LO VALUE
038F EB               XCHG
0390 13               INX    D
0391 13               INX    D
0392 79               MOV    A,C
0393 12               STAX   D
0394 13               INX    D
0395 78               MOV    A,B
0396 12               STAX   D
                      ;NOW ADD THE VALUE TO THE DATA ACCUMULATOR
0397 112F2E    AUP4:  LXI    D,TMPSM  ;MOVE VALUE TO TEMP BUFFER
039A E1               POP    H        ;GET VALUE
039B 7D               MOV    A,L
039C 12               STAX   D
039D 13               INX    D
039E 7C               MOV    A,H
039F 12               STAX   D
03A0 13               INX    D
03A1 3E00             MVI    A,0
03A3 12               STAX   D
03A4 13               INX    D
03A5 12               STAX   D
03A6 2A022E           LHLD   APOINT   ;SET UP FOR 32 BIT ADD
03A9 110400           LXI    D,4
03AC 19               DAD    D        ;RESULT WILL REMAIN IN
03AD 4D               MOV    C,L      ;TEMPORARY DATA AREA
03AE 44               MOV    B,H
03AF 212F2E           LXI    H,TMPSM
03B2 CD2C0D           CALL   B4ADD
                      ;NOW INCREMENT THE NUMBER OF SUMMATIONS
```

```
03B5 2A022E            LHLD    APOINT
03B8 110800            LXI     D,8
03BB 19                DAD     D
03BC 4D                MOV     C,L
03BD 44                MOV     B,H
03BE 21332E            LXI     H,FBYT1 ;HL POINTS TO 32 BIT ONE
03C1 CD2C0D            CALL    B4ADD
03C4 C9                RET             ;RETURN
                ;
                ;ROUTINE TO READ THE ANALOG DEVICES RTI-1200 BOARD
                ;
03C5 3A082E     GETAV: LDA     ACHNM   ;GET CHANNEL # IN A REG
03C8 32FA4F            STA     MUXADR  ;SELECT CHANNEL
03CB 32FB4F            STA     CNVCMD  ;ISSUE CONVERT COMMAND
03CE 3AFC4F     GA1:   LDA     ASTAT   ;WAIT FOR END OF CONVERSION
03D1 07                RLC
03D2 D2CE03            JNC     GA1
03D5 2AFD4F            LHLD    ADCLO   ;GET DATA IN HL
03D8 C9                RET             ;H=HI DATA, L=LO DATA
                ;
                ;DIGITAL DATA ACCUMULATION ROUTINE - UPDATES THE
                ;NUMBER OF 'ON' READINGS AND TOTAL NUMBER OF READS
                ;
03D9 3A092E     DUPDT: LDA     DCHNM   ;DETERMINE WHICH PORT TO
03DC FE07              CPI     7       ;INPUT FROM
03DE D2F503            JNC     DUP2    ;NO CARRY -> PORT B
03E1 3C                INR     A       ;MUST BE PORT A
03E2 47                MOV     B,A     ;ROTATE POPER MASK VALUE
03E3 3E00              MVI     A,0     ;INTO THE A REG
03E5 37                STC
03E6 17         DUP1:  RAL
03E7 05                DCR     B
03E8 C2E603            JNZ     DUP1
03EB 47                MOV     B,A     ;SAVE MASK IN B
03EC DBE4              IN      PORTA1  ;READ PORT
03EE A0                ANA     B       ;LOGICALLY AND THE MASK
03EF CA0A04            JZ      DUP4    ;VALUE WAS 0
03F2 C31604            JMP     DUP5    ;VALUE WAS 1
03F5 D607       DUP2:  SUI     7       ;GET VALUE FROM PORT B
03F7 47                MOV     B,A     ;ROTATE TO GET PROPER
03F8 3E00              MVI     A,0     ;MASK VALUE
03FA 37                STC
03FB 17         DUP3:  RAL
03FC 05                DCR     B
03FD C2FB03            JNZ     DUP3
0400 47                MOV     B,A     ;SAVE MASK IN B
0401 DBE5              IN      PORTB1  ;GET VALUE
0403 A0                ANA     B       ;LOGICALLY AND THE MASK
0404 CA0A04            JZ      DUP4    ;VALUE WAS 0
0407 C31604            JMP     DUP5    ;VALUE WAS 1
040A 2A062E     DUP4:  LHLD    DPOINT  ;INCREMENT TOTAL COUNTS
040D 4D                MOV     C,L
040E 44                MOV     B,H
040F 21332E            LXI     H,FBYT1 ;POINT TO A ONE
0412 CD2C0D            CALL    B4ADD   ;INCREMENT
0415 C9                RET             ;FINISHED
```

```
0416 2A062E   DUP5:    LHLD    DPOINT   ;INCREMENT 'ON' COUNTS
0419 110400            LXI     D,4      ;SET LOCATION POINTER
041C 19                DAD     D
041D 4D                MOV     C,L
041E 44                MOV     B,H
041F 21372E            LXI     H,FB100  ;POINT TO A 100
0422 CD2C0D            CALL    B4ADD
0425 C30A04            JMP     DUP4     ;NOW INCREMENT TOTAL COUNTS
              ;
              ;
              ;ROUTINE FINISH TAKES THE INTERVAL DATA FROM THE
              ;SCRATCHPAD AREA, CONVERTS IT AND STORES IT IN THE
              ;RECORD AREA
              ;
              ;
0428 211018   FINISH:  LXI     H,1810H  ;INITIALIZE TABLE ADDRESS
042B 223B2E            SHLD    TADDR    ;AND APOINT,DPOINT
042E 2A002E            LHLD    APNT
0431 22022E            SHLD    APOINT
0434 2A042E            LHLD    DPNT
0437 22062E            SHLD    DPOINT
043A 2A162E            LHLD    CRECBS   ;INITIALIZE RECORD ADDRESS
043D 111100            LXI     D,17     ;SINCE TIME HAS ALREADY
0440 19                DAD     D        ;BEEN STORED
0441 22162E            SHLD    CRECBS
0444 2A192E   FIN1:    LHLD    TBLOV    ;CHECK FOR PROM TABLE
0447 5D                MOV     E,L      ;OVERFLOW
0448 54                MDV     D,H
0449 2A3B2E            LHLD    TADDR
044C CDE10B            CALL    NEGD
044F 19                DAD     D
0450 7C                MOV     A,H
0451 17                RAL
0452 DA0805            JC      FIN2     ;YES-ALL DATA HAS BEEN STORED
0455 2A3B2E            LHLD    TADDR    ;NO-STORE MORE DATA
0458 7E                MOV     A,M      ;CHECK SENSOR TYPE
0459 FE30              CPI     30H
045B CAD104            JZ      STDIG    ;SENSOR IS DIGITAL
045E 2A022E            LHLD    APOINT   ;MUST BE ANALOG
D461 5E                MOV     E,M      ;COMPUTE AND STORE HI VALUE
0462 23                INX     H
0463 56                MOV     D,M
0464 EB                XCHG             ;HL=HI VALUE
0465 CD5C05            CALL    CALCV    ;COMPUTE FLOATING POINT
0468 EB                XCHG             ;VALUE AND STORE IN RECORD
0469 2A162E            LHLD    CRECBS
046C 71                MOV     M,C
046D 23                INX     H
046E 73                MOV     M,E
046F 23                INX     H
0470 72                MOV     M,D
0471 23                INX     H
0472 22162E            SHLD    CRECBS   ;SAVE CURRENT POINTER
0475 2A022E            LHLD    APOINT   ;NOW COMPUTE AND STORE
0478 23                INX     H        ;THE LO VALUE
0479 23                INX     H
```

```
047A 5E              MOV   E,M      ;GET LO VALUE
047B 23              INX   H
047C 56              MOV   D,M
047D EB              XCHG           ;MOVE TO HL
047E CD5C05          CALL  CALCV    ;COMPUTE VALUE AND STORE
0481 EB              XCHG
0482 2A162E          LHLD  CRECBS
0485 71              MOV   M,C
0486 23              INX   H
0487 73              MOV   M,E
0488 23              INX   H
0489 72              MOV   M,D
048A 23              INX   H
048B 22162E          SHLD  CRECBS   ;SAVE CURRENT POINTER
048E 2A022E          LHLD  APOINT   ;NOW COMPUTE AND STORE
0491 110400          LXI   D,4      ;AVERAGE VALUE
0494 19              DAD   D        ;FIRST DIVIDE SUM BY
0495 22202E          SHLD  DIVD     ;NUMBER OF SCANS
0498 19              DAD   D
0499 221E2E          SHLD  DIVS
049C CD710C          CALL  DIV32
049F 2A022E          LHLD  APOINT   ;GET VALUE
04A2 110400          LXI   D,4
04A5 19              DAD   D
04A6 5E              MOV   E,M
04A7 23              INX   H
04A8 56              MOV   D,M
D4A9 EB              XCHG           ;HL=AVERAGE VALUE
04AA CD5C05          CALL  CALCV    ;COMPUTE FLOATING POINT
04AD EB              XCHG           ;VALUE AND STORE IN
04AE 2A162E          LHLD  CRECBS   ;RECORD AREA
04B1 71              MOV   M,C
04B2 23              INX   H
04B3 73              MOV   M,E
04B4 23              INX   H
04B5 72              MOV   M,D
04B6 23              INX   H
04B7 22162E          SHLD  CRECBS   ;STORE CURRENT POINTER
04BA 2A022E          LHLD  APOINT   ;UPDATE TABLE ADDRESS
04BD 110C00          LXI   D,12     ;AND APOINT
04C0 19              DAD   D
04C1 22022E          SHLD  APOINT
04C4 2A3B2E          LHLD  TADDR
04C7 111400          LXI   D,20
04CA 19              DAD   D
04CB 223B2E          SHLD  TADDR
04CE C34404          JMP   FIN1     ;DONE
04D1 2A062E   STDIG: LHLD  DPOINT   ;DIGITAL - COMPUTE %ON
04D4 221E2E          SHLD  DIVS     ;SET UP FOR 32 BIT DIVIDE
04D7 110400          LXI   D,4
04DA 19              DAD   D
04DB 22202E          SHLD  DIVD
04DE CD710C          CALL  DIV32
04E1 2A062E          LHLD  DPOINT   ;NOW GET RESULT AND STORE
04E4 110400          LXI   D,4      ;IN RECORD AREA
04E7 19              DAD   D
```

```
04E8 5E                 MOV     E,M
04E9 2A162E             LHLD    CRECBS
04EC 73                 MOV     M,E
04ED 23                 INX     H
04EE 22162E             SHLD    CRECBS  ;SAVE CURRENT POINTER
04F1 2A062E             LHLD    DPOINT  ;UPDATE DPOINT AND
04F4 110800             LXI     D,B     ;TABLE ADDRESS
04F7 19                 DAD     D
04F8 22062E             SHLD    DPOINT
04FB 2A3B2E             LHLD    TADDR
04FE 111400             LXI     D,20
0501 19                 DAD     D
0502 223B2E             SHLD    TADDR
0505 C34404             JMP     FIN1    ;DONE
0508 2A162E     FIN2:   LHLD    CRECBS  ;SAVE ENDING TIME IN RECORD
050B CD1E11             CALL    SAVTIM
050E 2A162E             LHLD    CRECBS  ;UPDATE POINTER
0511 111100             LXI     D,17
0514 19                 DAD     D
0515 22162E             SHLD    CRECBS
0518 2A122E             LHLD    RECOV   ;SEE IF RECORD AREA IS FULL
051B 5D                 MOV     E,L
051C 54                 MOV     D,H
051D 2A162E             LHLD    CRECBS
0520 CDE10B             CALL    NEGD
0523 19                 DAD     D
0524 7C                 MOV     A,H
0525 17                 RAL
0526 DA2C05             JC      FIN3    ;YES - FULL
0529 C33D05             JMP     FIN4    ;NO - MORE SPACE
052C 2A162E     FIN3:   LHLD    CRECBS  ;SAVE TOP OF RECORD ADDRESS
052F 22402E             SHLD    LRADD
0532 3E01               MVI     A,1     ;SET FULL RECORD FLAG
0534 32422E             STA     FBFLAG
0537 219120             LXI     H,2091H ;REINITIALIZE RECORO POINTER
053A 22162E             SHLD    CRECBS
053D 3E00       FIN4:   MVI     A,0     ;NOW ZERO IN PROGRESS FLAG
053F 32102E             STA     IPFL    ;AND SCRATCHPAD AREA
0542 2A042E             LHLD    DPNT
0545 11FF2D             LXI     D,2DFFH
0548 0E00               MVI     C,0
054A 71         FIN5:   MOV     M,C
054B CDA910             CALL    HILO
054E D24A05             JNC     FIN5
0551 3E00               MVI     A,0     ;ZERO CASUS & CDSUS
0553 320C2E             STA     CASUS
0556 32DD2E             STA     CDSUS
0559 C3CB01             JMP     SCAN    ;START A NEW RECORD
                ;
                ;ROUTINE TO CALCULATE THE ANALOG VALUES ACCORDING
                ;TO THE COEFFICIENTS IN THE PROM TABLE - THE VALUE
                ;CALCULATED IS = A + B*X + C*X**2 + D*X**3 WHERE
                ;X IS THE ANALOG READING IN THE RANGE 0-4095 DECIMAL
                ;THE VALUE IS RETURNED WITH EXPONENT IN C AND THE
                ;MANTISSA IN HL
                ;
```

```
0558 E5         CALCV:  PUSH    H          ;SAVE INTEGER VALUE
055D 2A3B2E             LHLD    TADDR
0560 110800             LXI     D,8        ;GET A COEFF AND SAVE IN
0563 19                 DAD     D          ;PARTIAL SUM AREA
0564 113D2E             LXI     D,TMPVAL
0567 7E                 MOV     A,M
0568 12                 STAX    D
0569 23                 INX     H
056A 13                 INX     D
056B 7E                 MOV     A,M
056C 12                 STAX    D
056D 23                 INX     H
056E 13                 INX     D
056F 7E                 MOV     A,M
0570 12                 STAX    D
0571 E1                 POP     H          ;RECOVER X AND CONVERT
0572 CD710B             CALL    FLOAT
0575 E5                 PUSH    H
0576 C5                 PUSH    B          ;SAVE X IN FP
0577 2A3B2E             LHLD    TADDR      ;GET B COEFF
057A 110B00             LXI     D,11
057D 19                 DAD     D
057E 7E                 MOV     A,M
057F 23                 INX     H
0580 5E                 MOV     E,M
0581 23                 INX     H
0582 56                 MOV     D,M
0583 C1                 POP     B
0584 E1                 POP     H          ;RECOVER X AND SAVE X
0585 E5                 PUSH    H
0586 C5                 PUSH    B
0587 47                 MOV     B,A
0588 CDC80A             CALL    FMULT      ;MULTIPLY B BY X
058B E5                 PUSH    H          ;SAVE BX
058C C5                 PUSH    B
058D 213D2E             LXI     H,TMPVAL   ;GET A COEFF
0590 7E                 MOV     A,M
0591 23                 INX     H
0592 5E                 MOV     E,M
0593 23                 INX     H
0594 56                 MOV     D,M
0595 C1                 POP     B
0596 E1                 POP     H          ;RECOVER BX
0597 47                 MOV     B,A
0598 CD750A             CALL    FADD       ;ADD TO GET A+BX
059B EB                 XCHG
059C 213D2E             LXI     H,TMPVAL   ;MOVE RESULT BACK
059F 71                 MOV     M,C
05A0 23                 INX     H
05A1 73                 MOV     M,E
05A2 23                 INX     H
05A3 72                 MOV     M,D
05A4 2A3B2E             LHLD    TADDR      ;GET C COEFF
05A7 110E00             LXI     D,14
05AA 19                 DAD     D
05AB 7E                 MOV     A,M
```

```
05AC 23          INX    H
05AD 5E          MOV    E,M
05AE 23          INX    H
05AF 56          MOV    D,M
05B0 C1          POP    B
05B1 E1          POP    H        ;RECOVER AND SAVE X
05B2 E5          PUSH   H
05B3 C5          PUSH   B
05B4 47          MOV    B,A
05B5 CDC80A      CALL   FMULT    ;MULTIPLY C BY X
05B8 EB          XCHG
05B9 79          MOV    A,C      ;SAVE CX
05BA C1          POP    B
05BB E1          POP    H        ;RECOVER AND SAVE X AGAIN
05BC E5          PUSH   H
05BD C5          PUSH   B
05BE 47          MOV    B,A
05BF CDC80A      CALL   FMULT    ;MULTIPLY CX BY X
05C2 E5          PUSH   H
05C3 C5          PUSH   B        ;SAVE CX**2
05C4 213D2E      LXI    H,TMPVAL ;GET A+BX
05C7 7E          MOV    A,M
05C8 23          INX    H
05C9 5E          MOV    E,M
05CA 23          INX    H
05CB 56          MOV    D,M
05CC C1          POP    B        ;RECOVER CX**2
05CD E1          POP    H
05CE 47          MOV    B,A
05CF CD750A      CALL   FADD     ;NOW WE HAVE A+BX+CX**2
05D2 EB          XCHG
05D3 213D2E      LXI    H,TMPVAL ;MOVE BACK TO TEMP
05D6 71          MOV    M,C
05D7 23          INX    H
05D8 73          MOV    M,E
05D9 23          INX    H
05DA 72          MOV    M,D
05DB 2A3B2E      LHLD   TADDR    ;GET D COEFF
05DE 111100      LXI    D,17
05E1 19          DAD    D
05E2 7E          MOV    A,M
05E3 23          INX    H
05E4 5E          MOV    E,M
05E5 23          INX    H
05E6 56          MOV    D,M
05E7 C1          POP    B
05E8 E1          POP    H
05E9 E5          PUSH   H        ;RECOVER AND SAVE X
05EA C5          PUSH   B
05EB 47          MOV    B,A      ;MULTIPLY D BY X
05EC CDC80A      CALL   FMULT
05EF EB          XCHG
05F0 79          MOV    A,C      ;SAVE DX
05F1 C1          POP    B
05F2 E1          POP    H        ;RECOVER AND SAVE X
05F3 E5          PUSH   H
```

```
05F4 C5             PUSH    B
05F5 47             MOV     B,A
05F6 CDC80A         CALL    FMULT     ;MULTIPLY DX BY X
05F9 EB             XCHG
05FA 79             MOV     A,C       ;SAVE DX**2
05FB C1             POP     B
05FC E1             POP     H         ;RECOVER X
05FD 47             MOV     B,A
05FE CDC80A         CALL    FMULT     ;MULTIPLY DX**2 BY X
0601 E5             PUSH    H
0602 C5             PUSH    B         ;SAVE DX**3
0603 213D2E         LXI     H,TMPVAL  ;GET A+BX+CX**2
0606 7E             MOV     A,M
0607 23             INX     H
0608 5E             MOV     E,M
0609 23             INX     H
060A 56             MOV     D,M
060B C1             POP     B
060C E1             POP     H         ;RECOVER DX**3
060D 47             MOV     B,A
060E CD750A         CALL    FADD      ;A+BX+CX**2+DX**3 IN C, HL
0611 C9             RET
                ;
                ;**************************************************
                ;
                ;       INTERRUPT SERVICE ROUTINES
                ;
                ;**************************************************
                ;
                ;----
                ;INTO
                ;----
                ;FUNCTION: INTO
                ;DESCRIPTION: IRO SERVICES INTERRUPT REQUEST 0
                ;              WHICH IS GENERATED BY THE POWER FAIL
                ;              CIRCUITRY IN THE POWER SUPPLY.  THIS
                ;              IS THE HIGHEST PRIORITY INTERRUPT.
                ;
0612 3E55       INTO:   MVI     A,55H     ;STORE A CHECK BYTE IN RAM
0614 322A2E             STA     CHKBYT
0617 2A2B2E             LHLD    PUPSAV    ;STORE TIME
061A CD1E11             CALL    SAVTIM
061D 2A2B2E             LHLD    PUPSAV
0620 111100             LXI     D,17
0623 19                 DAD     D
0624 222B2E             SHLD    PUPSAV
0627 F5                 PUSH    PSW       ;SAVE THE STATE OF
0628 C5                 PUSH    B         ;THE SYSTEM
0629 D5                 PUSH    D
062A E5                 PUSH    H
062B 210000             LXI     H,0       ;SAVE STACK POINTER
062E 39                 DAD     SP
062F 222D2E             SHLD    SPSAV
0632 F3                 DI
0633 3E20               MVI     A,EOI     ;ISSUE END OF INTERRUPT
0635 D3D8               OUT     PIC       ;COMMAND
```

```
0637 FB                  EI
0638 76                  HLT
                ;
                ;----
                ;INT1
                ;----
                ;FUNCTION: INT1
                ;DESCRIPTION: INT1 SERVICES COMMUNICATIONS THROUGH
                ;             THE MODEM PORT - ALL AVAILABLE DATA
                ;             IS SENT TO THE CALLING DEVICE
                ;
0639 E5         INT1:    PUSH    H         ;SAVE MACHINE STATUS
063A D5                  PUSH    D
063B C5                  PUSH    B
063C F5                  PUSH    PSW
063D FB                  EI
063E 3E01                MVI     A,1       ;SET IOBYTE FOR MODEM
0640 32472E              STA     IOBYTE
0643 DBF0                IN      MODEM     ;CLEAR MODEM PORT
0645 DBA1                IN      MINUTE    ;SET UP TO WAIT FOR COMMAND
0647 3C                  INR     A         ;FOR AT LEAST 1 MINUTE
0648 3C                  INR     A
0649 57                  MOV     D,A
064A 0E2E                MVI     C,'.'     ;PROMPT CALLING DEVICE FOR
064C CDBF10              CALL    MO        ;A COMMAND
064F DBA1       INT1A:   IN      MINUTE
0651 BA                  CMP     D         ;SEE IF MINUTE IS UP
0652 CA7006              JZ      INT1C     ;YES - TIMEOUT CONDITION
0655 DBF1                IN      MODMST    ;LOOK FOR CHARACTER
0657 E602                ANI     RXRDY
0659 CA4F06              JZ      INT1A
065C DBF0                IN      MODEM     ;IS CHARACTER A D?
065E E67F                ANI     7FH
0660 FE44                CPI     'D'
0662 CA6806              JZ      INT1B     ;YES
0665 C37006              JMP     INT1C     ;NO
0668 CD4207     INT1B:   CALL    DDATA     ;SEND DATA
066B 0E2A                MVI     C,'*'     ;SEND END OF DATA CHAR.
066D CDBF10              CALL    MO
0670 3E00       INT1C:   MVI     A,0       ;EXIT AND RESTORE SYSTEM
0672 32472E              STA     IOBYTE    ;IOBYTE=CONSOLE
0675 3A0F2E              LDA     TIMUP     ;ADJUST TIMUP IF NECESSARY
0678 47                  MOV     B,A
0679 DBA1                IN      MINUTE
067B B8                  CMP     B
067C D28206              JNC     INT1D
067F C36A0A              JMP     RESTR     ;RESTORE SYSTEM
0682 320F2E     INT1D:   STA     TIMUP     ;TIMUP=CURRENT MINUTES
0685 C36A0A              JMP     RESTR     ;RESTORE SYSTEM
                ;
                ;
                ;----
                ;INT2
                ;----
                ;FUNCTION: INT2
                ;DESCRIPTION: INT2 SERVICES INTERRUPT REQUEST 2
```

```
                        ;               WHICH IS GENERATED BY THE LOCAL
                        ;               TERMINAL
                        ;
0688 E5         INT2:   PUSH    H       ;SAVE MACHINE STATUS
0689 D5                 PUSH    D
068A C5                 PUSH    B
068B F5                 PUSH    PSW
068C FB                 EI              ;ENABLE INTERRUPTS
068D DBEC               IN      CONSD   ;GET CHARACTER AND CHECK
068F E67F               ANI     7FH     ;FOR VALID COMMAND LETTER
0691 FE44               CPI     'D'
0693 CC4207             CZ      DDATA   ;DISPLAY DATA
0696 C29C06             JNZ     SKIP    ;NO CALL
0699 C36A0A             JMP     RESTR   ;YES - WAS CALLED
069C FE4D       SKIP:   CPI     'M'
069E CA9909             JZ      MONINT  ;GO TO MONITOR
06A1 FE53               CPI     'S'
06A3 CAB709             JZ      SETINT  ;SET DATE AND TIME
06A6 FE54               CPI     'T'
06A8 CAA309             JZ      TIMINT  ;DISPLAY DATE AND TIME
06AB 21BE06             LXI     H,MSG   ;PRINT MESSAGE
06AE 7E         INT2A:  MOV     A,M
06AF B7                 ORA     A
06B0 CABB06             JZ      INT2B
06B3 4F                 MOV     C,A
06B4 CD1010             CALL    CO
06B7 23                 INX     H
06B8 C3AE06             JMP     INT2A
06BB C36A0A     INT2B:  JMP     RESTR   ;RESTORE
06BE 0D0A       MSG:    DB      CR,LF
06C0 4E4342432D         DB      'NCBC-CEL DATA GATHERING SYSTEM'
06DE 0D0A0D0A           DB      CR,LF,CR,LF
06E2 44202D2044         DB      'D - DISPLAY DATA',CR,LF
06F4 4D202D2045         DB      'M - ENTER 80/24 MONITOR',CR,LF
070D 53202D2053         DB      'S - SET DATE AND TIME',CR,LF
0724 54202D2044         DB      'T - DISPLAY DATE AND TIME',CR,LF
073F 0D0A00             DB      CR,LF,D
0742 2A162E     DDATA:  LHLD    CRECBS  ;SAVE CRECBS AND TADDR
0745 22432E             SHLD    CRBSAV  ;POINTERS
0748 2A3B2E             LHLD    TADDR
074B 22452E             SHLD    TADSAV
074E 219120             LXI     H,2091H ;INITIALIZE RECORD BASE
0751 22162E             SHLD    CRECBS
0754 3A422E     DD1:    LDA     FBFLAG  ;CHECK AND SEE IF WE ARE
0757 B7                 ADD     A       ;THRU SENDING DATA TO THE
0758 CA6F07             JZ      INT2C   ;CONSOLE PORT
075B 2A402E             LHLD    LRADD
075E EB                 XCHG
075F 2A162E             LHLD    CRECBS
0762 CDE10B             CALL    NEGD
0765 1B                 DCX     D
0766 19                 DAD     D
0767 7C                 MOV     A,H
0768 17                 RAL
0769 DAF40B             JC      INT2N   ;YES, FINISHED
076C C3B007             JMP     INT2D   ;NO, DO ANOTHER RECORD
```

```
076F 2A432E     INT2C:  LHLD    CRBSAV   ;NOT A FULL RECORD -
0772 2B                 DCX     H        ;CHECK AGAIN
0773 EB                 XCHG
0774 2A162E             LHLD    CRECBS
0777 CDE10B             CALL    NEGD
077A 19                 DAD     D
077B 7C                 MOV     A,H
077C 17                 RAL
077D DAF408             JC      INT2N    ;YES, FINISHED
0780 CD2510     INT2D:  CALL    CRLF     ;DISPLAY ANOTHER RECORD
0783 CD2510             CALL    CRLF
0786 CD2510             CALL    CRLF
0789 3A472E             LDA     IOBYTE
078C 87                 ADD     A
078D C29C07             JNZ     SKIP1
0790 DBED               IN      CONST    ;PAUSE IF CHARACTER ENTERED
0792 E602               ANI     RXRDY
0794 CA9C07             JZ      SKIP1    ;NO PAUSE
0797 DBEC               IN      CONSD    ;DO NOT RESUME UNTIL ANOTHER
0799 CDFC0F             CALL    CI       ;CHARACTER IS HIT
079C 0611       SKIP1:  MVI     B,17     ;SEND START TIME
079E 2A162E             LHLD    CRECBS
07A1 4E         INT2E:  MOV     C,M
07A2 CDCA10             CALL    OUTPUT
07A5 23                 INX     H
07A6 05                 DCR     8
07A7 C2A107             JNZ     INT2E
07AA 22162E             SHLD    CRECBS   ;SAVE RECORD POINTER
07AD CD2510             CALL    CRLF
07B0 211018             LXI     H,1810H  ;INITIALIZE TABLE POINTER
0783 223B2E             SHLD    TADDR
07B6 2A192E     INT2F:  LHLD    T8LOV    ;CHECK FOR TABLE OVERFLOW
07B9 EB                 XCHG
07BA 2A382E             LHLD    TADDR
07BD CDE10B             CALL    NEGD
07C0 19                 DAD     D
07C1 7C                 MOV     A,H
07C2 17                 RAL
07C3 D2C907             JNC     INT2G    ;NO - GET MORE DATA
07C6 C3E008             JMP     INT2L    ;YES - END THE RECORD
07C9 2A3B2E     INT2G:  LHLD    TADDR    ;DETERMINE SENSOR TYPE
07CC 7E                 MOV     A,M      ;AND DISPLAY DATA
07CD FE30               CPI     30H
07CF CA7808             JZ      INTDI    ;MUST BE DIGITAL
07D2 4F                 MOV     C,A      ;ANALOG - OUTPUT TYPE
07D3 CDCA10             CALL    OUTPUT
07D6 0E20               MVI     C,' '
07D8 CDCA10             CALL    OUTPUT
07DB 23                 INX     H        ;OUTPUT SENSOR NAME
07DC 0605               MVI     B,5
07DE 4E         INT2H:  MOV     C,M
07DF CDCA10             CALL    OUTPUT
07E2 23                 INX     H
07E3 05                 DCR     B
07E4 C2DE07             JNZ     INT2H
07E7 0E20               MVI     C,' '
```

```
07E9 CDCA10          CALL    OUTPUT
07EC 0E48            MVI     C,'H'     ;NOW PRINT HI VALUE
07EE CDCA10          CALL    OUTPUT
07F1 0E49            MVI     C,'I'
07F3 CDCA10          CALL    OUTPUT
07F6 0E3D            MVI     C,'='
07F8 CDCA10          CALL    OUTPUT
07FB 2A162E          LHLD    CRECBS
07FE 4E              MOV     C,M
07FF 23              INX     H
0800 5E              MOV     E,M
0801 23              INX     H
0802 56              MOV     D,M
0803 EB              XCHG
0804 CD6A09          CALL    INT2X
0807 0E20            MVI     C,' '
0809 CDCA10          CALL    OUTPUT
080C 0E4C            MVI     C,'L'     ;NOW PRINT LO VALUE
08DE CDCA10          CALL    OUTPUT
0811 0E4F            MVI     C,'O'
0813 CDCA10          CALL    OUTPUT
0816 DE3D            MVI     C,'='
0818 CDCA1D          CALL    OUTPUT
081B 2A162E          LHLO    CRECBS
081E 11D3D0          LXI     D,3
0821 19              OAO     D
0822 4E              MOV     C,M
0823 23              INX     H
D824 5E              MDV     E,M
0825 23              INX     H
0826 56              MDV     D,M
0827 EB              XCHG
0828 C06A09          CALL    INT2X
082B DE20            MVI     C,' '
082D CDCA10          CALL    OUTPUT
083D 0E41            MVI     C,'A'     ;NOW PRINT AVERAGE VALUE
0832 CDCA10          CALL    OUTPUT
0835 0E56            MVI     C,'V'
0837 CDCA10          CALL    OUTPUT
083A DE3D            MVI     C,'='
083C CDCA10          CALL    OUTPUT
083F 2A162E          LHLD    CRECBS
0842 11D60D          LXI     0,6
0845 19              DAO     D
0846 4E              MDV     C,M
0847 23              INX     H
0848 5E              MOV     E,M
0849 23              INX     H
084A 56              MOV     D,M
084B 23              INX     H
084C 22162E          SHLD    CRECBS    ;SAVE CURRENT POINTER
D84F EB              XCHG
085D C06A09          CALL    INT2X
0853 DE2D            MVI     C,' '
0855 CDCA10          CALL    OUTPUT
0858 2A3B2E          LHLD    TADOR     ;PRINT UNITS
```

```
085B 110600           LXI    D,6
085E 19               DAD    D
085F 4E               MOV    C,M
0860 CDCA10           CALL   OUTPUT
0863 23               INX    H
0864 4E               MOV    C,M
0865 CDCA10           CALL   OUTPUT
0868 CD2510           CALL   CRLF
086B 2A3B2E           LHLD   TADDR     ;INCREMENT TADDR
086E 111400           LXI    D,20
0871 19               DAD    D
0872 223B2E           SHLD   TADDR
0875 C3B607           JMP    INT2F     ;CONTINUE
0878 4F       INTDI:  MOV    C,A       ;OUTPUT DIGITAL TYPE
0879 CDCA10           CALL   OUTPUT
087C 0E20             MVI    C,' '
087E CDCA10           CALL   OUTPUT
0881 23               INX    H
0882 0605             MVI    B,5       ;OUTPUT SENSOR NAME
0884 4E       INT2I:  MOV    C,M
0885 CDCA10           CALL   OUTPUT
0888 23               INX    H
0889 05               DCR    B
088A C28408           JNZ    INT2I
088D 0E20             MVI    C,' '
088F CDCA10           CALL   OUTPUT
0892 2A162E           LHLD   CRECBS    ;PRINT ON VALUE
0895 7E               MOV    A,M
0896 FE64             CPI    100
0898 CAA808           JZ     INT2J
089B CD4710           CALL   DCONV
089E CDCA10           CALL   OUTPUT
08A1 48               MOV    C,B
08A2 CDCA10           CALL   OUTPUT
08A5 C3B508           JMP    INT2K
08A8 0E31     INT2J:  MVI    C,'1'
08AA CDCA10           CALL   OUTPUT
08AD 0E30             MVI    C,'0'
08AF CDCA10           CALL   OUTPUT
08B2 CDCA10           CALL   OUTPUT
08B5 0E25     INT2K:  MVI    C,'%'
08B7 CDCA10           CALL   OUTPUT
08BA 0E20             MVI    C,' '
08BC CDCA10           CALL   OUTPUT
08BF 0E4F             MVI    C,'O'
08C1 CDCA10           CALL   OUTPUT
08C4 0E4E             MVI    C,'N'
08C6 CDCA10           CALL   OUTPUT
08C9 CD2510           CALL   CRLF
08CC 2A162E           LHLD   CRECBS    ;UPDATE CRECBS  & TADDR
08CF 23               INX    H
08D0 22162E           SHLD   CRECBS
08D3 2A3B2E           LHLD   TADDR
08D6 111400           LXI    D,20
08D9 19               DAD    D
08DA 223B2E           SHLD   TADDR
```

```
08DD C3B607              JMP     INT2F     ;CONTINUE
08E0 0611       INT2L:   MVI     B,17      ;FINISHED SENDING DATA -
08E2 2A162E              LHLD    CRECBS    ;SEND ENDING TIME
08E5 4E         INT2M:   MOV     C,M
08E6 CDCA10              CALL    OUTPUT
08E9 23                  INX     H
08EA 05                  DCR     B
08EB C2E508              JNZ     INT2M
08EE 22162E              SHLD    CRECBS    ;SAVE CURRENT POINTER
08F1 C35407              JMP     DD1       ;START WITH NEW RECORD
08F4 2A432E     INT2N:   LHLD    CRBSAV    ;FINISHED, RESTORE CRECBS
08F7 22162E              SHLD    CRECBS    ;AND TADDR POINTERS
08FA 2A452E              LHLD    TADSAV
08FD 223B2E              SHLD    TADDR
0900 CD2510              CALL    CRLF
0903 CD2510              CALL    CRLF
0906 211609              LXI     H,MSG1    ;SEND POWER UP & POWER
0909 7E         INT20:   MOV     A,M       ;DOWN INFO
090A B7                  ORA     A
090B CA3A09              JZ      MSG11
D90E 4F                  MOV     C,A
090F CDCA10              CALL    OUTPUT
0912 23                  INX     H
0913 C30909              JMP     INT20
0916 504F574552MSG1:     DB      'POWER UP                POWER DOWN'
0937 0A0D00              DB      LF,CR,0
093A 21D020     MSG11:   LXI     H,2000H
093D 1604                MVI     D,4
093F 1E11                MVI     E,17
0941 4E         INT2P:   MOV     C,M
0942 CDCA10              CALL    OUTPUT
0945 23                  INX     H
0946 1D                  DCR     E
0947 C24109              JNZ     INT2P
094A 0E20                MVI     C,' '
094C CDCA10              CALL    OUTPUT
094F CDCA10              CALL    OUTPUT
0952 1E11                MVI     E,17
0954 4E         INT2Q:   MOV     C,M
0955 CDCA10              CALL    OUTPUT
0958 23                  INX     H
0959 1D                  DCR     E
095A C25409              JNZ     INT2Q
095D CD2510              CALL    CRLF
0960 1E11                MVI     E,17
0962 15                  DCR     D
0963 C24109              JNZ     INT2P
0966 3E00                MVI     A,0
0968 87                  ADD     A
0969 C9                  RET
                ;
096A CD3C0D     INT2X:   CALL    FPINT     ;CONVERT AND SAVE VALUE
096D E5                  PUSH    H
096E D5                  PUSH    D
096F 210A20              LXI     H,200AH   ;SET 5 DIGIT INDEX
0972 221C2E              SHLD    INDEX
```

```
0975 3A1B2E              LDA     SNBYT    ;CHECK FOR NEGATIVE FRACTION
0978 E601                ANI     1
097A C29109              JNZ     INT2Z
097D E1         INT2Y:   POP     H
097E CDECOD              CALL    DECOT    ;OUTPUT WHOLE PORTION
0981 0E2E                MVI     C,'.'    ;PRINT DECIMAL POINT
0983 CDCA10              CALL    OUTPUT
0986 210820              LXI     H,2008H  ;FOUR DIGIT INDEX
0989 221C2E              SHLD    INDEX
098C E1                  POP     H
098D CDECOD              CALL    DECOT
0990 C9                  RET
0991 0E2D       INT2Z:   MVI     C,'-'
0993 CDCA10              CALL    OUTPUT
0996 C37D09              JMP     INT2Y
                ;
0999 210000     MONINT:  LXI     H,0      ;SAVE THE STACK POINTER
099C 39                  DAD     SP       ;AND GO TO THE MONITOR
099D 222D2E              SHLD    SPSAV
09A0 C33E0E              JMP     DEBUG
                ;
09A3 CD2510     TIMINT:  CALL    CRLF     ;DISPLAY DATE & TIME
09A6 0E54                MVI     C,'T'
09A8 CD1010              CALL    CO
09AB CD2510              CALL    CRLF
09AE CDA50F              CALL    MTIME
09B1 CD2510              CALL    CRLF
0984 C36A0A              JMP     RESTR    ;RESTORE SYSTEM
                ;
09B7 210000     SETINT:  LXI     H,0      ;SAVE STACK POINTER AND
09BA 39                  DAD     SP       ;THEN ATTEMPT TO SET CLOCK
09BB 222D2E              SHLD    SPSAV
098E CD2510              CALL    CRLF
09C1 0E53                MVI     C,'S'
09C3 CD1D10              CALL    CO
09C6 CD100A              CALL    GETNM    ;GET MONTH
09C9 5F                  MOV     E,A      ;SAVE IN A
09CA CD0F11              CALL    PCHK     ;CHECK FOR VALID DELIMITER
09CD DA61DA              JC      INTERR
09D0 C2610A              JNZ     INTERR   ;ERROR
09D3 CD1D0A              CALL    GETNM    ;GET DAY
09D6 67                  MDV     H,A      ;SAVE IN H
09D7 CD0F11              CALL    PCHK     ;CHECK FOR VALID DELIMITER
09DA DA610A              JC      INTERR
09DD C2610A              JNZ     INTERR   ;ERROR
09ED CD100A              CALL    GETNM    ;GET YEAR
09E3 57                  MOV     D,A
09E4 CD0F11              CALL    PCHK
09E7 DA610A              JC      INTERR
09EA C2610A              JNZ     INTERR   ;ERROR
09ED CD350A              CALL    INTDAT   ;SET DATE
09F0 CD100A              CALL    GETNM    ;GET HOUR
09F3 67                  MOV     H,A
09F4 CD0F11              CALL    PCHK
09F7 DA610A              JC      INTERR   ;ERROR
09FA C2610A              JNZ     INTERR
```

```
09FD CD100A            CALL    GETNM    ;GET MINUTES
0A00 6F                MOV     L,A
0A01 CD0F11            CALL    PCHK     ;CHECK FOR CARRIAGE RETURN
0A04 C2610A            JNZ     INTERR
0A07 D2610A            JNC     INTERR   ;ERROR
0A0A CD4E0A            CALL    INTTIM   ;SET TIME
0A0D C36A0A            JMP     RESTR    ;RESTORE SYSTEM
0A10 CD6D11   GETNM:   CALL    TI       ;GET ASCII CHARACTER
0A13 CD250A            CALL    CNVBN    ;CONVERT TO BINARY
0A16 07                RLC              ;SCALE FIRST DIGIT FOR
0A17 47                MOV     B,A      ;MOST SIGNIFICANT OF TWO
0A18 80                ADD     B        ;(MULTIPLY BY 10)
0A19 80                ADD     B
0A1A 80                ADD     B
0A1B 80                ADD     B
0A1C 47                MOV     B,A      ;SAVE IN B
0A1D CD6D11            CALL    TI       ;GET SECOND DIGIT IN ASCII
0A20 CD250A            CALL    CNVBN    ;CONVERT
0A23 80                ADD     B        ;GET BINARY VALUE OF BOTH
0A24 C9                RET              ;DIGITS IN A REGISTER
0A25 D630     CNVBN:   SUI     '0'      ;CONVERT ASCII TO BINARY -
0A27 FE0A              CPI     10       ;VALID ONLY IF 0 - 9
0A29 FA2F0A            JM      CNV1
0A2C C3610A            JMP     INTERR
0A2F FE00     CNV1:    CPI     0
0A31 FA610A            JM      INTERR
0A34 C9                RET

              ;
              ;INTDAT SETS THE DATE - THE YEAR MUST BE IN D, THE
              ;MONTH IN E, AND THE DAY IN H
              ;
0A35 D3A3     INTDAT:  OUT     SETDAT   ;ENTER SET DATE MODE
0A37 DBA5     YEARIN:  IN      YEAR     ;LET YEAR INCREMENT TO
0A39 BA                CMP     D        ;PROPER VALUE
0A3A C2370A            JNZ     YEARIN
0A3D DBA4     MNTHIN:  IN      MONTH    ;LET MONTH INCREMENT
0A3F BB                CMP     E
0A40 C23D0A            JNZ     MNTHIN
0A43 DBA3     DAYIN:   IN      DAY      ;LET DAY INCREMENT
0A45 BC                CMP     H
0A46 C2430A            JNZ     DAYIN
0A49 D3A6              OUT     RFC
0A4B D3A6              OUT     RFC      ;RESET FAST CLOCK
0A4D C9                RET
              ;
              ;INTTIM SETS THE TIME - THE HOUR MUST BE IN H, AND
              ;THE MINUTES IN L
              ;
0A4E D3A1     INTTIM:  OUT     SETTIM   ;ENTER SET TIME MODE
0A50 DBA2     HOURIN:  IN      HOUR     ;LET HOUR INCREMENT TO
0A52 BC                CMP     H        ;PROPER VALUE
0A53 C2500A            JNZ     HOURIN
0A56 DBA1     MININ:   IN      MINUTE   ;LET MINUTE INCREMENT
0A58 BD                CMP     L
0A59 C2560A            JNZ     MININ
0A5C D3A6              OUT     RFC
```

```
0A5E D3A6                OUT     RFC      ;RESET FAST CLOCK
0A60 C9                  RET
                ;
0A61 0E23       INTERR:  MVI     C,'#'    ;ISSUE KEYBOARD ERROR PROMPT
0A63 CD1010              CALL    CO
0A66 2A2D2E              LHLD    SPSAV    ;RECOVER STACK AND RESTORE
0A69 F9                  SPHL
0A6A F3         RESTR:   DI
0A6B 3E20                MVI     A,EOI    ;ISSUE END OF INTERRUPT
0A6D D3D8                OUT     PIC      ;COMMAND
0A6F F1                  POP     PSW      ;RESTORE REGISTERS
0A70 C1                  POP     B
0A71 D1                  POP     D
0A72 E1                  POP     H
0A73 FB                  EI
0A74 C9                  RET
                ;
                ;
                ;***********************************************
                ;
                ;        FLOATING POINT ROUTINES
                ;
                ;***********************************************
                ;
                ;
                ;EQUATES:
                ;
0080 =          SIGN     EQU     80H      ;MASK FOR THE SIGN OF VALUES
007F =          EXP      EQU     7FH      ;MASK FOR EXPONENT PORTION
0040 =          EXP0     EQU     40H      ;EXPONENT FOR 2**0
0050 =          EXP16    EQU     50H      ;EXPONENT FOR 2**16
7FFF =          MAX      EQU     7FFFH    ;MAX VALUE FOR INTEGER
                ;
                ;
                ;----
                ;FADD
                ;----
                ;FUNCTION: FADD
                ;DESCRIPTION: FLOATING POINT ADD ROUTINE
                ;ENTRY CONDITIONS:
                ;        VALUE 1          EXP>B, MANTISSA>DE
                ;        VALUE 2          EXP>C, MANTISSA>HL
                ;EXIT CONDITIONS:
                ;        RESULT           EXP>C, MANTISSA>HL
                ;
0A75 F5         FADD:    PUSH    PSW      ;SAVE A AND FLAGS
0A76 C5         FADDX:   PUSH    B        ;SAVE ORIGINAL SIGNS
0A77 79                  MOV     A,C
0A78 E67F                ANI     EXP      ;OF THE TWO EXPONENTS
0A7A 4F                  MOV     C,A
0A7B 78                  MOV     A,B
0A7C E67F                ANI     EXP
0A7E 47                  MOV     B,A
0A7F CD170C              CALL    FPCMP    ;IS MAG.V1>MAG.V2 ?
0A82 D28E0A              JNC     FADD1    ;JUMP IF SO
0A85 78                  MOV     A,B      ;IF NOT EXCHANGE V1 & V2
```

```
0A86 41                 MOV     B,C
0A87 4F                 MOV     C,A
0A88 EB                 XCHG
0A89 E3                 XTHL            ;EXCHANGE ORIGINAL SIGNS
0A8A 7D                 MOV     A,L
0A8B 6C                 MOV     L,H
0A8C 67                 MOV     H,A
0A8D E3                 XTHL
0A8E 7B         FADD1:  MOV     A,B     ;EXP1>'A'
0A8F 91                 SUB     C       ;EXP1-EXP2
0A90 C4400C     FADD2:  CNZ     RBUMP1  ;SCALE THE SMALLER VALUE TO
0A93 3D                 DCR     A       ;THE LARGER VALUE
0A94 F2900A             JP      FADD2
0A97 E3         FADD3:  XTHL            ;GET ORIGINAL SIGNS
0A98 7D                 MOV     A,L     ;XOR THEM TO DETERMINE IF
0A99 AC                 XRA     H       ;SIGN'S ARE EQUAL
0A9A E3                 XTHL            ;PUT THINGS BACK
0A9B FCE10B             CM      NEGD    ;NEGATE SMALLER MANTISSA
0A9E 19         FADD4:  DAD     D       ;ADD THE TWO VALUES
0A9F D2AB0A             JNC     FADD5   ;IF NO CARRY, FIX THE EXP
                                        ;AND RETURN
0AA2 FAAB0A             JM      FADD5   ;IF THE SIGNS WERE EQUAL
                                        ;CARRY IS BIT OF RESULT
0AA5 CD410C             CALL    RBUMP   ;RIGHT SHIFT THE RESULT
0AA8 E3                 XTHL            ;INCREMENT RESULTANT EXP
0AA9 24                 INR     H
0AAA E3                 XTHL
0AAB C1         FADD5:  POP     B       ;EXP1 IS FINAL EXP
0AAC 48                 MOV     C,B     ;FINAL EXPONENT
0AAD CDF10B             CALL    ZEROD   ;CHECK FOR MANTISSA=0
0AB0 D2B50A             JNC     FADD6
0AB3 0E40               MVI     C,EXPO  ;ANSWER = 0.0000;EXP>40H
0AB5 D4F90B     FADD6:  CNC     NORM    ;NORMALIZE IF MANTISSAL<>0
0AB8 F1                 POP     PSW     ;RESTORE ORIGINAL ACCUM
0AB9 C9                 RET             ;AND FLAGS
                ;
                ;----
                ;FSUB
                ;----
                ;FUNCTION: FSUB
                ;DESCRIPTION: FLOATING POINT SUBTRACT ROUTINE
                ;RESULT = VALUE1 - VALUE2
                ;ENTRY CONDITIONS:
                ;       VALUE1          EXP>B, MANTISSA>DE
                ;       VALUE2          EXP>C, MANTISSA>HL
                ;EXIT CONDITONS:
                ;       RESULT          EXP>C, MANTISSA>HL
                ;
0ABA F5         FSUB:   PUSH    PSW     ;SAVE A AND THE FLAGS
0ABB CD0B0C             CALL    FPZR2   ;CHECK FOR VALUE2=D
0ABE DA760A             JC      FADDX   ;DONT COMPLEMENT SIGN IF SO
0AC1 79                 MOV     A,C     ;GET SIGN OF VALUE2
0AC2 EE80               XRI     SIGN    ;CDMPLIMENT IT
0AC4 4F                 MOV     C,A
0AC5 C3760A             JMP     FADDX   ;NOW IT CAN BE ADDED
                ;
```

```
                      ;-----
                      ;FMULT
                      ;-----
                      ;FUNCTION: FLOATING POINT MULTIPLY ROUTINE
                      ;DESCRIPTION:
                      ;ENTRY CONDITIONS:
                      ;        VALUE1            EXP>B, MANTISSA>DE
                      ;        VALUE2            EXP>C, MANTISSA>HL
                      ;EXIT CONDITIONS:
                      ;        RESULT            EXP>C, MANTISSA>HL
                      ;RESULTANT IS NORMALIZED
                      ;
0ACB F5               FMULT:   PUSH    PSW       ;SAVE A AND FLAGS
0AC9 79                        MOV     A,C       ;XOR THE SIGNS TO GET THE
0ACA A8                        XRA     B         ;RESULTANT SIGN
0ACB F5                        PUSH    PSW       ;SAVE FOR LATER
0ACC CD040C                    CALL    FPZR1     ;IS VALUE 1 = 0.0000
0ACF D2D70A                    JNC     FMULT3    ;JUMP IF NOT
0AD2 48                        MOV     C,B       ;ELSE EXCHANGE THE VALUES
0AD3 EB                        XCHG              ;SINCE 0.0000 IS ANSWER
0AD4 C3DD0A                    JMP     FMULT4    ;MULTIPLY DONE
0AD7 CD0B0C           FMULT3:  CALL    FPZR2     ;IS VALUE 2 = 0.0000
0ADA D2E30A                    JNC     FMULT5    ;DONE IF SO
0ADD F1               FMULT4:  POP     PSW       ;MAKE SURE SIGN IS +
0ADE AF                        XRA     A
0ADF F5                        PUSH    PSW
0AE0 C3FE0A                    JMP     MDRETN    ;END OF MULTIPLY
0AE3 79               FMULT5:  MOV     A,C       ;GET PURE EXPONENTS
0AE4 E67F                      ANI     EXP
0AE6 4F                        MOV     C,A
0AE7 78                        MOV     A,B
0AE8 E67F                      ANI     EXP
0AEA 81                        ADD     C         ;EXP1 + EXP2
0AEB D640                      SUI     EXP0      ;FINAL EXPONENT VALUE
0AED 4F                        MOV     C,A
0AEE CD860B                    CALL    IMULT     ;MULTIPLY THE MANTISSA'S
0AF1 7A               FMULT1:  MOV     A,D       ;CONTINUE SHIFTING UNTIL
0AF2 B7                        ORA     A         ;D&E<0
0AF3 FAFD0A                    JM      FMULT2
0AF6 0D                        DCR     C         ;DCR THE EXPONENT
0AF7 CD4A0C                    CALL    BUMP2     ;SHIFT LEFT L>H>E>D
0AFA C3F10A                    JMP     FMULT1    ;TILL NORMALIZED
0AFD EB               FMULT2:  XCHG
0AFE F1               MDRETN:  POP     PSW       ;RECOVER SIGN
0AFF E680                      ANI     SIGN      ;GET RESULTANT SIGN
0B01 B1                        ORA     C         ;FORM RESULTANT EXPONENT
0B02 4F                        MOV     C,A
0B03 F1                        POP     PSW       ;RESTORE A AND FLAGS
0B04 C9                        RET
                      ;
                      ;----
                      ;FDIV
                      ;----
                      ;FUNCTION: FDIV
                      ;DESCRIPTION: FLOATING POINT DIVIDE ROUTINE
                      ;VALUE1 / VALUE2
```

```
                   ;ENTRY CONDITIONS:
                   ;       VALUE1             EXP>B, MANTISSA>DE
                   ;       VALUE2             EXP>C, MANTISSA>HL
                   ;EXIT CONDITIONS:
                   ;       RESULT             EXP>C, MANTISSA>HL
                   ;RESULT IS NORMALIZED
                   ;
0BD5 F5            FDIV:   PUSH    PSW        ;SAVE A AND THE FLAGS
0B06 79                    MOV     A,C        ;GET SIGNS AND XOR THEM TO
0B07 A8                    XRA     B          ;GET RESULTANT SIGN
0B08 F5                    PUSH    PSW        ;SAVE RESULTANT SIGN
0B09 CD040C                CALL    FPZR1      ;IS VALUE1=0.00D0
0B0C D2170B                JNC     FDIV2      ;JUMP IF NOT
0B0F 48                    MOV     C,B        ;ANSWER IS 0.000D, EXP>40H
0B10 EB                    XCHG               ;MANTISSA>0.0000
0B11 F1                    POP     PSW        ;MAKE SURE SIGN IS +
0B12 AF                    XRA     A
0B13 F5                    PUSH    PSW
0B14 C3FE0A                JMP     MDRETN     ;END OF DIVIDE
0B17 CD0B0C        FDIV2:  CALL    FPZR2      ;IS VALUE2 = 0.0000
0B1A D2250B                JNC     FDIV3      ;JUMP IF NOT
0B1D 0E7F                  MVI     C,EXP      ;SET MAX EXP VALUE
0B1F 21FFFF                LXI     H,-1       ;SET MAX MANTISSA
0B22 C3FE0A                JMP     MDRETN     ;DONE WITH DIVIDE
0B25 79            FDIV3:  MOV     A,C        ;GET PURE EXPONENTS
0B26 E67F                  ANI     EXP
0B28 4F                    MOV     C,A
0B29 78                    MOV     A,B
0B2A E67F                  ANI     EXP
0B2C 91                    SUB     C
0B2D C640                  ADI     EXPO       ;ADJUST FOR RESULTANT
0B2F 4F                    MOV     C,A        ;EXP=EXP1-EXP2
0B30 CDA80B                CALL    IDIV       ;DIVIDE MANTISSA'S
0B33 7A            FDIV1:  MOV     A,D        ;NORMALIZE RESULTANT
0B34 B3                    ORA     E          ;BY SHIFTING 'DEHL' RIGHT
0B35 CAFE0A                JZ      MDRETN     ;TILL DE=0
0B38 37                    STC                ;SET CARRY TO SHIFT
0B39 CD410C                CALL    RBUMP      ;SHIFT RIGHT
0B3C 0C                    INR     C          ;ADJUST EXPONENT VALUE
0B3D C3FE0A                JMP     MDRETN
                   ;
                   ;---
                   ;FIX
                   ;---
                   ;FUNCTION: FIX
                   ;DESCRIPTION: ROUTINE TO CONVERT A FLOATING POINT
                   ;             VALUE INTO A 16-BIT INTEGER
                   ;ENTRY CONDITIONS:
                   ;       F.P. VALUE - EXP>C, MANTISSA>HL
                   ;EXIT CONDITIONS:
                   ;       INTEGER>HL, IN TWO'S COMPLEMENT FORM
                   ;
0B40 F5            FIX:    PUSH    PSW        ;SAVE A AND THE FLAGS
0B41 CD0B0C                CALL    FPZR2      ;IS VALUE=0.0000
0B44 DA690B                JC      RETN1
0B47 79                    MOV     A,C        ;CHECK FOR EXP<0
```

```
0B48 B7                 ORA     A           ;SET FLAGS FOR VALUE IN A
0B49 F5                 PUSH    PSW         ;SAVE EXPONENT,SIGN
0B4A E67F               ANI     EXP         ;STRIP SIGN BIT
0B4C 4F                 MOV     C,A
0B4D D640               SUI     EXPO        ;AND RETURN ZERO IF SO
0B4F F2580B             JP      FIXO        ;CLEAR HL IF VALUE TOO SMALL
0B52 210000             LXI     H,0
0B55 C3650B             JMP     RETN
0B58 3E50       FIXO:   MVI     A,EXP16     ;CHECK FOR EXP OVER 16
0B5A 91                 SUB     C           ;RETURN MAX VALUE IF SO
0B5B FA6B0B             JM      OVFL
0B5E C4400C     FIX1:   CNZ     RBUMP1      ;NORMALIZE RESULT TO 2**16
0B61 3D                 DCR     A
0B62 C25E0B             JNZ     FIX1
0B65 F1         RETN:   POP     PSW         ;RECOVER THE SIGN OF VALUE
0B66 FCE10B             CM      NEGD        ;NEGATE INTEGER IF MINUS
0B69 F1         RETN1:  POP     PSW         ;RESTORE A REG AND FLAGS
0B6A C9                 RET
0B6B 21FF7F     OVFL:   LXI     H,MAX       ;SET THE MAX VALUE
0B6E C3650B             JMP     RETN
                ;
                ;-----
                ;FLOAT
                ;-----
                ;FUNCTION: FLOAT
                ;DESCRIPTION: ROUTINE TO CONVERT 16-BIT INTEGER TO
                ;             TO FLOATING POINT REPRESENTATION
                ;ENTRY CONDITIONS:
                ;       INTEGER>HL, TWO'S COMPLEMENT
                ;EXIT CONDITIONS:
                ;       EXPONENT>C, MANTISSA>HL
                ;
0B71 0E40       FLOAT:  MVI     C,EXPO      ;2**0
0B73 CDF10B             CALL    ZEROD       ;SEE IF THE INTEGER IS 0
0B76 D8                 RC                  ;IF SO, EXP=0, MANTISSA=0
0B77 CDEB0B             CALL    ABSD        ;GET INTEGER TO ABSOLUTE FORM
0B7A 3E50               MVI     A,50H       ;SET THE EXPONENT SCALE
                                            ;AT 2**16 IN EXCESS 40
0B7C F2810B             JP      F1          ;SET SIGN IN MANTSSA BIT 7
0B7F F680               ORI     80H         ;OF THE EXPONENT BYTE
0B81 4F         F1:     MOV     C,A
0B82 CDF90B             CALL    NORM        ;NORMALIZE IT FROM HERE
                                            ;CY=SIGN OF ORIGINAL
0B85 C9                 RET
                ;
                ;-----
                ;IMULT
                ;-----
                ;FUNCTION: IMULT
                ;DESCRIPTION: ROUTINE TO MULTIPLY TWO DOUBLE BYTE
                ;             INTEGERS WITH FOUR BYTE RESULT
                ;ENTRY CONDITIONS:
                ;       MULTIPLIER>HL
                ;       MULTIPLICAND>DE
                ;EXIT CONDITIONS:
                ;       RESULT>DE&HL
```

```
                        ;
0BB6 F5         IMULT:  PUSH    PSW     ;SAVE REGISTERS
0B87 C5                 PUSH    B
0B8B E5                 PUSH    H
0BB9 D5                 PUSH    D       ;MOVE MULTIPLICAND TO BC
0B8A C1                 POP     B
0B8B 210000             LXI     H,0     ;INITIALIZE RESULTANT BYTES
0B8E E5                 PUSH    H
0B8F D1                 POP     D       ;CLEAR DE
0B90 3E10               MVI     A,16    ;ITERATION COUNTER
0B92 E3         M1:     XTHL            ;HL>MULTIPLIER
0B93 29                 DAD     H       ;SHIFT LEFT INTO CARRY
0B94 E3                 XTHL            ;BACK TO THE STACK
0B95 029DDB             JNC     M2      ;IF CARRY SET ADD THE
                                        ;MULTIPLICAND TO THE
0B98 09                 DAD     B       ;PARTIAL RESULT
0B99 D29D0B             JNC     M2
0B9C 13                 INX     D       ;ADJUST CARRY FROM PREVIOUS
                                        ;DAD INSTRUCTION
0B9D 3D         M2:     DCR     A       ;CHECK ITERATION COUNTER
0B9E C44A0C             CNZ     BUMP2   ;FOR COMPLETION
                                        ;SHIFT PARTIAL SUM LEFT
0BA1 C2920B             JNZ     M1      ;ITERATE UNTIL DONE
0BA4 F1                 POP     PSW     ;DUMMY POP TO ADJUST STACK
0BA5 C1                 POP     B       ;RESTORE BC
0BA6 F1                 POP     PSW     ;RESTORE A AND FLAGS
0BA7 C9                 RET
                        ;
                        ;----
                        ;IDIV
                        ;----
                        ;FUNCTION: IDIV
                        ;DESCRIPTION: ROUTINE TO DIVIDE TWO 16-BIT
                        ;             INTEGERS WITH A 4 BYTE RESULT
                        ;             RESULT HAS DECIMAL AT THE MIDPOINT
                        ;             OF THE RESULTANT REGISTERS
                        ;ENTRY CONDITIONS:
                        ;       DIVISOR>HL
                        ;       DIVIDEND>DE
                        ;EXIT CONDITIONS:
                        ;       RESULT>DE.HL
0BA8 7C         IDIV:   MOV     A,H     ;CHECK FOR DIVISOR=0
0BA9 B5                 ORA     L
0BAA CADC0B             JZ      DIV3
0BAD C5                 PUSH    B       ;SAVE BC
0BAE CDE10B             CALL    NEGD
0BB1 E5                 PUSH    H       ;SAVE THE -DIVISOR
0BB2 210000             LXI     H,0     ;INITIALIZE QUOTIENT
0BB5 E5                 PUSH    H
0BB6 C1                 PDP     B
0BB7 AF                 XRA     A       ;SET CARRY=0
0BB8 3E20               MVI     A,32    ;ITERATION COUNTER
0BBA CD550C     DIV1:   CALL    BUMP3   ;SHIFT 3 WORDS LEFT
0BBD E3                 XTHL            ;GET -DIVSOR
0BBE E5                 PUSH    H       ;AND RESTORE IT
0BBF 17                 RAL             ;SAVE CARRY FLAG
```

```
OBC0 09                 DAD     B       ;SUBTRACT IT FROM DIVIDEND
OBC1 1F                 RAR             ;GET RESULT FROM BUMP3
                                        ;SAVE RESULT OF SUBTRACTION
                                        ;IF CARRY SET
OBC2 DAC90B             JC      DIVX    ;IF NOT SET CHECK RESULT
OBC5 B7                 ORA     A
OBC6 F2CC0B             JP      DIV2    ;NOT SUCCESSFUL REITERATE
OBC9 E5         DIVX:   PUSH    H       ;MOVE RESULT
OBCA C1                 POP     B
OBCB 37                 STC             ;MAKE SURE CARRY IS SET
OBCC 17         DIV2:   RAL             ;MAKE SURE ITERATION
OBCD 37                 STC             ;COUNTER IS > 0
OBCE 3F                 CMC
OBCF 1F                 RAR
OBD0 E1                 POP     H       ;GET ORIGINAL -DIVISOR
OBD1 E3                 XTHL            ;STACK IT
OBD2 3D                 DCR     A       ;DECREMENT COUNTER
OBD3 C2BA0B             JNZ     DIV1
OBD6 CD4A0C             CALL    BUMP2   ;DOUBLE SHIFT
OBD9 F1                 POP     PSW     ;ADJUST STACK POINTER
OBDA C1                 POP     B       ;RESTORE B&C
OBDB C9                 RET             ;FINISHED
OBDC 2B         DIV3:   DCX     H       ;SET RESULT TO FULL SCALE
OBDD 11FFFF             LXI     D,-1
OBE0 C9                 RET
                ;
                ;----
                ;NEGD
                ;----
                ;FUNCTION: NEGD
                ;DESCRIPTION: ROUTINE TO NEGATE A DOUBLE BYTE
                ;             VALUE IN HL WITH RESULT IN HL
                ;
OBE1 F5         NEGD:   PUSH    PSW     ;SAVE STATUS
OBE2 7C                 MOV     A,H
OBE3 2F                 CMA
OBE4 67                 MOV     H,A
OBE5 7D                 MOV     A,L
OBE6 2F                 CMA
OBE7 6F                 MOV     L,A
OBE8 23                 INX     H
OBE9 F1                 POP     PSW
OBEA C9                 RET
                ;
                ;----
                ;ABSD
                ;----
                ;FUNCTION: ABSD
                ;DESCRIPTION: ROUTINE TO GET THE ABSOLUTE VALUE
                ;             OF A DOUBLE BYTE INTEGER IN HL
                ;EXIT CONDITIONS:
                ;       ABS[INTEGER]>HL, SIGN>SIGN FLAG
                ;
OBEB AF         ABSD:   XRA     A
OBEC 84                 ADD     H
OBED FCE10B             CM      NEGD
```

```
0BF0 C9                 RET
                ;
                ;-----
                ;ZEROD
                ;-----
                ;FUNCTION: ZEROD
                ;DESCRIPTION: ROUTINE TO DETERMINE IF DOUBLE BYTE
                ;             INTEGER IN HL IS 0, CARRY FLAG IS
                ;             SET IF SO
0BF1 7C         ZEROD:  MOV     A,H      ;ADD HL AND LOOK FOR 0
0BF2 B5                 ORA     L
0BF3 37                 STC
0BF4 CAF80B             JZ      RETZ
0BF7 3F                 CMC
0BF8 C9         RETZ:   RET
                ;
                ;----
                ;NORM
                ;----
                ;FUNCTION: NORM
                ;DESCRIPTION: ROUTINE TO NORMALIZE A FLOATING
                ;             POINT NUMBER
                ;ENTRY CONDITIONS:
                ;       EXP>C, MANTISSA>HL
                ;EXIT CONDITIONS: SAME
                ;
0BF9 AF         NORM:   XRA     A        ;SEE IF NUMBER NEGATIVE
0BFA 84                 ADD     H        ;IF SO IT IS NORMALIZED
0BFB FA030C             JM      RETY     ;IF NOT DCR EXP AND SHIFT
0BFE 0D                 DCR     C        ;LEFT THE MANTISSA UNTIL
0BFF 29                 DAD     H        ;NORMALIZED
0C00 C3F90B             JMP     NORM
0C03 C9         RETY:   RET
                ;
                ;-----
                ;FPZR1
                ;-----
                ;FUNCTION: FPZR1
                ;DESCRIPTION: ROUTINE TD CHECK FOR THE SPECIAL
                ;             CASE OF FLOATING POINT VALUE 0.0000
                ;             THE CARRY FLAG IS SET TO 1 IF SO
                ;             FPZR1 IS VALUE1, FPZR2 FOR VALUE2
                ;
0C04 78         FPZR1:  MOV     A,B      ;INTERCHANGE VALUES
0C05 EB                 XCHG
0C06 CD100C             CALL    FPZRX    ;COMPARE TO 0.0000
0C09 EB                 XCHG             ;RESTORE VALUES
0C0A C9                 RET
0C0B 79         FPZR2:  MOV     A,C      ;EXPONENT 2 IN A
0C0C CD100C             CALL    FPZRX    ;COMPARE
0C0F C9                 RET
0C10 EE40       FPZRX:  XRI     EXPO     ;IS EXP=40H
0C12 C0                 RNZ
0C13 CDF10B             CALL    ZEROD    ;IS MANTISSA=0
0C16 C9                 RET
                ;
```

```
                ;-----
                ;FPCMP
                ;-----
                ;FUNCTION: FPCMP
                ;DESCRIPTION: FPCMP DETERMINES THE RELATION OF
                ;             TWO FLOATING POINT VALUES AS FOLLOWS:
                ;       CY=0    VALUE1>=VALUE2
                ;       CY=1    VALUE1<VALUE2
                ;       Z=1     VALUE1=VALUE2
                ;
0C17 C5         FPCMP:  PUSH    B       ;SAVE REGISTERS
0C18 D5                 PUSH    D
0C19 E5                 PUSH    H
0C1A 3E7F               MVI     A,EXP   ;GET EXPONENT
0C1C A1                 ANA     C       ;EXP2
0C1D 4F                 MOV     C,A
0C1E 3E7F               MVI     A,EXP
0C20 A0                 ANA     B       ;EXP1
0C21 91                 SUB     C       ;EXP1-EXP2
0C22 F5                 PUSH    PSW     ;SAVE RESULT
0C23 C22F0C             JNZ     CMPG1   ;GOTO COMPARE IF NOT ENOUGH
0C26 F1                 POP     PSW     ;ADJUST STACK
0C27 CDE10B             CALL    NEGD    ;COMPARE MANTISSA'S BY
0C2A 19                 DAD     D       ;SUBTRACTION
0C2B 3F                 CMC             ;COMPLEMENT CARRY
0C2C F5                 PUSH    PSW     ;SAVE CARRY
0C2D 7D                 MOV     A,L     ;GET ZERO FLAG SET IF RESULT
0C2E B4                 ORA     H       ;WAS ZERO
0C2F E1         CMPG1:  POP     H       ;RECOVER CARRY
0C30 CA3C0C             JZ      CMPG3   ;IF EQUAL DONT COMPLEMENT CY
0C33 F5                 PUSH    PSW     ;EQUAL, SAVE FLAG
0C34 7D                 MOV     A,L     ;GET IT BACK INTO CY FLAG
0C35 1F                 RAR
0C36 1F                 RAR             ;COMPLEMENT CARRY IF SIGN'S
0C37 A8                 XRA     B       ;NEGATIVE
0C38 6F                 MOV     L,A     ;SAVE IN L
0C39 F1                 POP     PSW     ;GET EQUAL FLAG
0C3A 7D                 MOV     A,L     ;SET FINAL LOGICAL OPERATOR
0C3B 17                 RAL
0C3C E1         CMPG3:  POP     H       ;RESTORE REGISTERS
0C3D D1                 POP     D
0C3E C1                 POP     B
0C3F C9                 RET
                ;
                ;------
                ;RBUMP1
                ;------
                ;FUNCTION: RBUMP1
                ;DESCRIPTION: RBUMP1 SHIFTS HL RIGHT ONE PLACE
                ;
0C40 BF         RBUMP1: CMP     A       ;CLEAR CARRY FLAG
0C41 F5         RBUMP:  PUSH    PSW     ;SAVE A REG
0C42 7C                 MOV     A,H
0C43 1F                 RAR
0C44 67                 MOV     H,A
0C45 7D                 MOV     A,L
```

```
0C46 1F                   RAR
0C47 6F                   MOV     L,A
0C48 F1                   POP     PSW      ;RESTORE A
0C49 C9                   RET
                  ;
                  ;-----
                  ;BUMP2
                  ;-----
                  ;FUNCTION: BUMP2
                  ;DESCRIPTION: ROUTINE TO SHIFT DE&HL LEFT
                  ;
0C4A F5           BUMP2:  PUSH    PSW      ;SAVE A
0C4B EB                   XCHG
0C4C 29                   DAD     H
0C4D EB                   XCHG
0C4E 29                   DAD     H
0C4F D2530C               JNC     B1
0C52 13                   INX     D
0C53 FI           B1:     POP     PSW
0C54 C9                   RET
                  ;
                  ;-----
                  ;BUMP3
                  ;-----
                  ;FUNCTION: BUMP3
                  ;DESCRIPTION: BUMP3 SHIFTS BC&DE&HL LEFT
                  ;
0C55 F5           BUMP3:  PUSH    PSW      ;SAVE A
0C56 7D                   MOV     A,L
0C57 17                   RAL
0C58 6F                   MOV     L,A
0C59 7C                   MOV     A,H
0C5A I7                   RAL
0C5B 67                   MOV     H,A
0C5C 7B                   MOV     A,E
0C5D 17                   RAL
0C5E 5F                   MOV     E,A
0C5F 7A                   MOV     A,D
0C60 17                   RAL
0C61 57                   MOV     D,A
0C62 79                   MOV     A,C
0C63 17                   RAL
0C64 4F                   MOV     C,A
0C65 78                   MOV     A,B
0C66 17                   RAL
0C67 47                   MOV     B,A
0C68 D26E0C               JNC     B3
0C6B F1                   POP     PSW
0C6C 37                   STC
0C6D C9                   RET
0C6E F1           B3:     POP     PSW
0C6F BF                   CMP     A
0C70 C9                   RET
                  ;
                  ;-----
                  ;DIV32
```

```
                    ;-----
                    ;FUNCTION: DIV32
                    ;DESCRIPTION: DIV32 DIVIDES TWO 32 BIT UNSIGNED
                    ;             INTEGERS LOCATED IN MEMORY - THE
                    ;             DIVISOR IS LOCATED AT DIVS AND THE
                    ;             DIVIDEND AT DIVD, WITH THE RESULT
                    ;             RETURNED AS QUOTIENT=DIVD AND THE
                    ;             REMAINDER=DIVS
                    ;
0C71 0E04           DIV32:  MVI     C,4       ;FIRST NEGATE THE DIVISOR
0C73 2A1E2E                 LHLD    DIVS
0C76 7E             D32:    MOV     A,M
0C77 2F                     CMA
0C78 77                     MOV     M,A
0C79 23                     INX     H
0C7A 0D                     DCR     C
0C7B C2760C                 JNZ     D32
0C7E 2A1E2E                 LHLD    DIVS
0C81 7E                     MOV     A,M
0C82 3C                     INR     A
0C83 77                     MOV     M,A
0C84 CA8A0C                 JZ      AD1
0C87 C3A20C                 JMP     D32AA
0C8A 23             AD1:    INX     H
0C8B 7E                     MOV     A,M
0C8C 3C                     INR     A
0C8D 77                     MOV     M,A
0C8E CA940C                 JZ      AD2
0C91 C3A20C                 JMP     D32AA
0C94 23             AD2:    INX     H
0C95 7E                     MOV     A,M
0C96 3C                     INR     A
0C97 77                     MOV     M,A
0C98 CA9E0C                 JZ      AD3
0C9B C3A20C                 JMP     D32AA
0C9E 23             AD3:    INX     H
0C9F 7E                     MOV     A,M
0CA0 3C                     INR     A
0CA1 77                     MOV     M,A
0CA2 21222E         D32AA:  LXI     H,REM     ;INITIALIZE REMAINDER TO 0
0CA5 3E00                   MVI     A,0
0CA7 0E04                   MVI     C,4
0CA9 77             D32A:   MOV     M,A
0CAA 23                     INX     H
0CAB 0D                     DCR     C
0CAC C2A90C                 JNZ     D32A
0CAF 3E21                   MVI     A,33      ;LOAD LOOP COUNTER
0CB1 F5             D32B:   PUSH    PSW       ;AND SAVE LOOP COUNTER
0CB2 0E04                   MVI     C,4
0CB4 21222E                 LXI     H,REM     ;NOW MOVE REMAINDER TO THE
0CB7 11262E                 LXI     D,REMSV   ;REMAINDER SAVE AREA
0CBA 7E             D32C:   MOV     A,M
0CBB 12                     STAX    D
0CBC 23                     INX     H
0CBD 13                     INX     D
0CBE 0D                     DCR     C
```

```
0CBF C2BA0C          JNZ    D32C
0CC2 01222E          LXI    B,REM
0CC5 2A1E2E          LHLD   DIVS    ;ADD THE NEGATED DIVISOR TO
0CC8 CD2C0D          CALL   B4ADD   ;THE REMAINDER
0CCB D2FC0C          JNC    D32E    ;UNDERFLDW-RESTORE REMAINDER
0CCE 21222E          LXI    H,REM   ;NO CARRY - SWAP THE
0CD1 46              MOV    B,M     ;REMAINDER AND REMAINDER-
0CD2 23              INX    H       ;SAVE AREA
0CD3 4E              MDV    C,M
0CD4 23              INX    H
0CD5 56              MOV    D,M
0CD6 23              INX    H
0CD7 5E              MOV    E,M
0CD8 C5              PUSH   B       ;SAVE REMAINDER
0CD9 D5              PUSH   D
0CDA 21262E          LXI    H,REMSV ;MOVE REMSV TO REM
0CDD 11222E          LXI    D,REM
0CE0 0E04            MVI    C,4
0CE2 7E       D32D:  MOV    A,M
0CE3 12              STAX   D
0CE4 23              INX    H
0CE5 13              INX    D
0CE6 0D              DCR    C
0CE7 C2E20C          JNZ    D32D
0CEA D1              POP    D
0CEB C1              POP    B       ;RECOVER REM AND MOVE TO
0CEC 60              MOV    H,B     ;REMSV TO COMPLETE SWAP
0CED 69              MOV    L,C
0CEE 01262E          LXI    B,REMSV
0CF1 7C              MOV    A,H
0CF2 02              STAX   B
0CF3 03              INX    B
0CF4 7D              MOV    A,L
0CF5 02              STAX   B
0CF6 03              INX    B
0CF7 7A              MOV    A,D
0CF8 02              STAX   B
0CF9 03              INX    B
0CFA 7B              MOV    A,E
0CFB 02              STAX   B
0CFC 0E04     D32E:  MVI    C,4     ;MOVE REMSV TO REM
0CFE 21262E          LXI    H,REMSV
0D01 11222E          LXI    D,REM
0D04 7E       D32F:  MOV    A,M
0D05 12              STAX   D
0D06 23              INX    H
0D07 13              INX    D
0D08 0D              DCR    C
0D09 C2040D          JNZ    D32F
0D0C 0E04            MVI    C,4     ;NOW PERFORM AN B LOCATION
0D0E 2A202E          LHLD   DIVD    ;SHIFT WITH CARRY
0D11 7E       D32G:  MOV    A,M     ;DIVIDEND-->REMAINDER
0D12 17              RAL
0D13 77              MOV    M,A
0D14 23              INX    H
0D15 0D              DCR    C
```

```
0D16 C2110D              JNZ      D32G
0D19 0E04                MVI      C,4
0D1B 21222E              LXI      H,REM
0D1E 7E        D32H:     MOV      A,M
0D1F 17                  RAL
0D20 77                  MOV      M,A
0D21 23                  INX      H
0D22 0D                  DCR      C
0D23 C21E0D              JNZ      D32H
0D26 F1                  POP      PSW      ;RECOVER LOOP COUNTER
0D27 3D                  DCR      A
0D28 C2B10C              JNZ      D32B     ;KEEP LOOPING
0D2B C9                  RET
               ;
               ;-----
               ;B4ADD
               ;-----
               ;FUNCTION: B4ADD
               ;DESCRIPTION: B4ADD ADDS THE TWO 32 BIT UNSIGNED
               ;             INTEGERS ADDRESSED BY THE B AND H
               ;             REGISTER PAIRS - THE RESULT IS
               ;             RETURNED IN THE LOCATIONS ADDRESSED
               ;             BY B&C
               ;
0D2C 1E04      B4ADD:    MVI      E,4      ;NUMBER OF BYTES TO ADD
0D2E AF                  XRA      A        ;CLEAR CARRY FLAG
0D2F 0A        B4LP:     LDAX     B        ;LOAD BYTE OF FIRST
0D30 8E                  ADC      M        ;ADD BYTE OF SECOND
0D31 02                  STAX     B        ;STORE RESULT AT FIRST
0D32 1D                  DCR      E        ;DECREMENT COUNT
0D33 CA3B0D              JZ       RTB      ;DONE IF ZERO
0D36 03                  INX      B        ;INCREMENT BYTE POINTERS
0D37 23                  INX      H
0D38 C32F0D              JMP      B4LP
0D38 C9        RTB:      RET               ;RETURN
               ;
               ;-----
               ;FPINT
               ;-----
               ;FUNCTION: FPINT
               ;DESCRIPTION: FPINT CONVERTS A FLOATING POINT
               ;             NUMBER TO TWO 16 BIT HEX NUMBERS
               ;             THAT CAN EASILY BE CONVERTED TO
               ;             ASCII FOR DISPLAY
               ;ENTRY CONDITIONS:
               ;         FP NUMBER         EXP>C, MANTISSA>HL
               ;EXIT CONDITIONS:
               ;         RESULT            WHOLE>DE, FRACTION>HL (DE.HL)

               ;
0D3C 110000    FPINT:    LXI      D,0      ;INITIALIZE WHOLE PORTION
0D3F 79                  MOV      A,C      ;SAVE SIGN OF NUMBER ON
0D40 17                  RAL               ;THE STACK  IF CY=1 THEN
0D41 F5                  PUSH     PSW      ;THE NUMBER IS NEGATIVE
0D42 79                  MOV      A,C      ;RECOVER EXPONENT
0D43 E67F                ANI      7FH      ;STRIP SIGN BIT
```

```
0D45 FE40            CPI    40H      ;DETERMINE HOW TO SHIFT
0D47 CA670D          JZ     FRACT    ;NO SHIFT
0D4A D25B0D          JNC    SHLF     ;SHIFT LEFT
0D4D E60F    SHRT:   ANI    0FH      ;SHIFT RIGHT TO SET UP FOR
0D4F CD400C  SHRT1:  CALL   RBUMP1   ;FRACTION CALCULATION
0D52 3C              INR    A        ;EXP TELLS HOW MANY TIMES
0D53 E60F            ANI    0FH      ;TO SHIFT
0D55 CA670D          JZ     FRACT
0D58 C34F0D          JMP    SHRT1    ;SHIFT AGAIN
0D5B E60F    SHLF:   ANI    0FH      ;SHIFT LEFT TO PUT DECIMAL
0D5D CD4A0C  SHLF1:  CALL   BUMP2    ;POINT BETWEEN DE&HL
0D60 3D              DCR    A
0D61 CA670D          JZ     FRACT    ;THROUGH, COMPUTE FRACTION
0D64 C35D0D          JMP    SHLF1    ;SHIFT AGAIN
0D67 010000  FRACT:  LXI    B,0      ;INITIALIZE SCRATCH REG
0D6A D5              PUSH   D        ;SAVE WHOLE PORTION
0D6B EB              XCHG            ;DE = FRACTION
0D6C 21CC0D          LXI    H,FTBL   ;POINT TO FRACTION TABLE
0D6F 7A              MOV    A,D      ;MOVE MSB FOR SHIFTING
0D70 160F            MVI    D,0FH    ;COUNTER, SHIFT 16 TIMES
0D72 17      D1:     RAL
0D73 DA870D          JC     ADDN     ;ADD VALUE IF BIT = 1
0D76 23              INX    H        ;OTHERWISE INCREMENT THROUGH
0D77 23              INX    H        ;THE TABLE
0D78 15      D2:     DCR    D        ;DECREMENT COUNT
0D79 CA9D0D          JZ     ENDIT    ;JUMP IF FINISHED SHIFTING
0D7C F5              PUSH   PSW      ;TEST TO SEE IF IT IS TIME
0D7D 7A              MOV    A,D      ;TO SHIFT LSB
0D7E FE07            CPI    7H       ;HALFWAY THROUGH ?
0D80 CA970D          JZ     SWNUM    ;YES MOVE IN LSB
0D83 F1      D3:     POP    PSW      ;RECOVER VALUE
0D84 C3720D          JMP    D1       ;CONTINUE
0D87 D5      ADDN:   PUSH   D        ;SAVE D
0D88 56              MOV    D,M      ;LOAD VALUE
0D89 23              INX    H
0D8A 5E              MOV    E,M
0D8B 23              INX    H
0D8C E5              PUSH   H        ;SAVE H
0D8D 60              MOV    H,B
0D8E 69              MOV    L,C
0D8F 19              DAD    D        ;ADD TABLE VALUE
0D90 44              MOV    B,H      ;PUT RESULT BACK IN BC
0D91 4D              MOV    C,L
0D92 E1              POP    H        ;RESTORE HL&DE
0D93 D1              POP    D
0D94 C3780D          JMP    D2
0D97 F1      SWNUM:  POP    PSW      ;DISCARD OLD VALUE
0D98 7B              MOV    A,E      ;AND INSERT NEW VALUE
0D99 F5              PUSH   PSW      ;(LSB OF ORIGINAL FRACTION)
0D9A C3830D          JMP    D3
0D9D 60      ENDIT:  MOV    H,B      ;MOVE BC TO HL
0D9E 69              MOV    L,C
0D9F AF              XRA    A        ;CLEAR A
0DA0 D1              POP    D        ;RECOVER WHOLE PART
0DA1 83              ADD    E        ;CHECK FOR ZERO
0DA2 CAAE0D          JZ     E1
```

```
0DA5 AF         E0:     XRA     A        ;NONZERO, JUST FINISH
0DA6 321B2E             STA     SNBYT
0DA9 F1                 POP     PSW      ;RECOVER SIGN
0DAA DAC40D             JC      SGN      ;JUMP IF NEGATIVE
0DAD C9                 RET              ;POSITIVE & FINISHED
0DAE 82         E1:     ADD     D        ;CHECKING FOR ZERO
0DAF CAB50D             JZ      E2       ;JUMP IF SO
0DB2 C3A50D             JMP     E0       ;IF NOT JUST FINISH
0DB5 F1         E2:     POP     PSW      ;RECOVER SIGN,WANT TO KNOW
                                         ;IF WE HAVE +0 OR -0
0DB6 DABE0D             JC      E3       ;JUMP IF NEGATIVE
0DB9 AF                 XRA     A        ;POSITIVE, FINISH UP
0DBA 321B2E             STA     SNBYT
0DBD C9                 RET
0DBE 3E01       E3:     MVI     A,1      ;STORE NEGATIVE ZERO FLAG
0DC0 321B2E             STA     SNBYT
0DC3 C9                 RET              ;FINISHED
0DC4 7A         SGN:    MOV     A,D
0DC5 2F                 CMA              ;NEGATE VALUE
0DC6 57                 MOV     D,A
0DC7 7B                 MOV     A,E
0DC8 2F                 CMA
0DC9 5F                 MOV     E,A
0DCA 13                 INX     D
0DCB C9                 RET              ;RETURN
0DCC 1388       FTBL:   DW      8813H
0DCE 09C4               DW      0C409H
0DD0 04E2               DW      0E204H
0DD2 0271               DW      7102H
0DD4 0139               DW      3901H
0DD6 009C               DW      9C00H
0DD8 004E               DW      4E00H
0DDA 0027               DW      2700H
0DDC 0014               DW      1400H
0DDE 000A               DW      0A00H
0DE0 0005               DW      0500H
0DE2 0002               DW      0200H
0DE4 0001               DW      0100H
0DE6 0000               DW      0000H
0DE8 0000               DW      0000H
0DEA 0000               DW      0000H
                ;
                ;-----
                ;DECOT
                ;-----
                ;FUNCTION: DECOT
                ;DESCRIPTION: DECOT TAKES THE 16 BIT INTEGER
                ;             IN HL AND CONVERTS IT FOR BASE
                ;             10 DISPLAY ON THE CONSOLE
                ;
0DEC F5         DECOT:  PUSH    PSW      ;SAVE REGISTERS
0DED C5                 PUSH    B
0DEE E5                 PUSH    H
0DEF C1                 POP     B        ;MOVE VALUE TO BC
0DF0 D5                 PUSH    D
0DF1 2A1C2E             LHLD    INDEX    ;INDEX TO -10**N TABLE
```

```
0DF4 3EFF              MVI     A,-1      ;CHECK FOR A NEGATIVE #
0DF6 A8                XRA     B         ;WRITE '-' IF SO
0DF7 FA060E            JM      DEC1
0DFA 47                MOV     B,A       ;COMPLIMENT VALUE AND
0DFB C5                PUSH    B         ;SAVE BC
0DFC 0E2D              MVI     C,'-'     ;PRINT MINUS SIGN
0DFE CDCA10            CALL    OUTPUT
0E01 C1                POP     B         ;RESTORE BC
0E02 79                MOV     A,C
0E03 2F                CMA
0E04 4F                MOV     C,A
0E05 03                INX     B
0E06 11320E    DEC1:   LXI     D,TENX-2 ;ADDRESS OF -10**N
0E09 E5                PUSH    H         ;SAVE INDEX
0E0A 2600              MVI     H,0       ;SET MSD OF INDEX TO 0
0E0C 19                DAD     D         ;GET ADDRESS OF -10**N
0E0D 5E                MOV     E,M       ;GET -10**N
0E0E 23                INX     H
0E0F 56                MOV     D,M
0E10 EB                XCHG              ;HL>-10**N
0E11 09                DAD     B
0E12 D21C0E            JNC     DEC2      ;IF VALUE<10**N GO TO NEXT
0E15 E5                PUSH    H         ;SAVE REMAINING VALUE
0E16 C1                POP     B         ;IN B&C
0E17 E1                POP     H         ;GET DIGIT VALUE
0E18 24                INR     H         ;INCREMENT DIGIT VALUE
0E19 C3060E            JMP     DEC1      ;SUBT 10**N AGAIN
0E1C E1        DEC2:   POP     H         ;GET DIGIT VALUE
0E1D AF                XRA     A         ;CLEAR A AND ADD DIGIT VALUE
0E1E 84                ADD     H
0E1F C5                PUSH    B         ;SAVE BC
0E20 CD1B10            CALL    CONV      ;GET ASCII REPRESENTATION
0E23 CDCA10            CALL    OUTPUT
0E26 3E30              MVI     A,'0'     ;CLEAR LOWER NIBBLE
0E28 A1                ANA     C         ;KEEP BLANK IF NO DIGIT
0E29 67                MOV     H,A
0E2A C1                POP     B         ;RESTORE BC
0E2B 2D                DCR     L         ;N=N-2
0E2C 2D                DCR     L         ;CHECK FOR 5 DIGITS
0E2D C2060E            JNZ     DEC1
0E30 D1                POP     D         ;RESTORE REGISTERS
0E31 C1                POP     B
0E32 F1                POP     PSW
0E33 C9                RET
0E34 FFFF      TENX:   DW      -1
0E36 F6FF              DW      -10
0E38 9CFF              DW      -100
0E3A 18FC              DW      -1000
0E3C F0D8              DW      -10000
               ;
               ;
               ;
               ;***************************************************
               ;
               ;       SBC 80/24 MONITOR
               ;
```

```
                  ;***************************************************
                  ;
                  ;
000D =            CR        EQU      0DH        ;CARRIAGE RETURN
000A =            LF        EQU      0AH        ;LINE FEED
                  ;
                  ;
                  ;THIS MONITOR MAY BE ENTERED FROM THE DATA
                  ;GATHERING SYSTEM CONSOLE
                  ;
0E3E 214E0E       DEBUG:    LXI      H,SIGNON           ;SEND THE SIGNON
0E41 7E           DEB1:     MOV      A,M                ;MESSAGE TO THE
0E42 B7                     ORA      A                  ;CONSOLE
0E43 CA6B0E                 JZ       START
0E46 4F                     MOV      C,A
0E47 CD1010                 CALL     CO
0E4A 23                     INX      H
0E4B C3410E                 JMP      DEB1
                  ;
0E4E 0D0A         SIGNON:   DB       CR,LF
0E50 5342432038             DB       'SBC 80/24 MONITOR'
0E61 0D0A0D0A00             DB       CR,LF,CR,LF,0
0E66 0E23         ERROR:    MVI      C,'#'              ;ERROR PROMPT
0E6B CD1010                 CALL     CO
                  ;
                  ;         MAIN COMMAND LOOP
                  ;
0E6B FB           START:    EI                  ;ENABLE INTERRUPTS
0E6C CD2510                 CALL     CRLF       ;OUTPUT CR,LF
0E6F 0E2E                   MVI      C,'.'      ;PROMT CHARACTER
0E71 CD1010                 CALL     CO
0E74 CD6D11                 CALL     TI         ;GET CHARACTER AND ECHO IT
0E77 FE0D                   CPI      CR         ;CARRIAGE RETURN?
0E79 CA6B0E                 JZ       START
0E7C D641                   SUI      'A'        ;OTHERWISE TEST FOR (A-Z)
0E7E FA660E                 JM       ERROR
0E81 FE1A                   CPI      LCT        ;TOO LARGE?
0E83 F2660E                 JP       ERROR
0E86 0E02                   MVI      C,2        ;ASSUME COMMAND NEEDS
                                                ;TWO ARGUMENTS
0EB8 116B0E                 LXI      D,START    ;SIMULATE A CALL
0E8B D5                     PUSH     D
0EBC 21990E                 LXI      H,CTBL     ;JUMP TABLE
0E8F 5F                     MOV      E,A        ;COMPUTE INDEX
0E90 1600                   MVI      D,0
0E92 19                     DAD      D
0E93 19                     DAD      D
0E94 7E                     MOV      A,M        ;GET LSB OF ADDRESS
0E95 23                     INX      H
0E96 66                     MOV      H,M
0E97 6F                     MOV      L,A
0E98 E9                     PCHL                ;GO TO ROUTINE
                  ;
                  ;___________________________________________________
                  ;
                  ;         COMMAND BRANCH TABLE
```

```
                    ;
0E99 660E           CTBL:   DW      ERROR   ;A
0E9B 660E                   DW      ERROR   ;B
0E9D 660E                   DW      ERROR   ;C
0E9F CD0E                   DW      DISP    ;D - DISPLAY MEMORY
0EA1 EE0E                   DW      MEXIT   ;E - EXIT MONITOR
0EA3 F80E                   DW      FILL    ;F - FILL MEMORY
0EA5 070F                   DW      GOTO    ;G - GOTO MEMORY ADDRESS
0EA7 0D0F                   DW      HEXN    ;H - HEX SUM & DIFFERENCE
0EA9 280F                   DW      PORTI   ;I - INPUT PORT
0EAB 660E                   DW      ERROR   ;J
0EAD 660E                   DW      ERROR   ;K
0EAF 660E                   DW      ERROR   ;L
0EB1 4F0F                   DW      MOVE    ;M - MOVE MEMORY
0EB3 660E                   DW      ERROR   ;N
0EB5 600F                   DW      PORTO   ;O - OUTPUT TO PORT
0EB7 660E                   DW      ERROR   ;P
0EB9 660E                   DW      ERROR   ;Q
0EBB 660E                   DW      ERROR   ;R
0EBD 860F                   DW      SUBS    ;S - SUBSTITUTE MEMORY
0EBF A50F                   DW      MTIME   ;T - DISPLAY TIME
0EC1 660E                   DW      ERROR   ;U
0EC3 660E                   DW      ERROR   ;V
0EC5 660E                   DW      ERROR   ;W
0EC7 660E                   DW      ERROR   ;X
0EC9 660E                   DW      ERROR   ;Y
0ECB 660E                   DW      ERROR   ;Z
001A =              LCT     EQU     ($-CTBL)/2      ;TABLE SIZE
                    ;
                    ;____________________________________________________
                    ;
                    ;       D COMMAND - DISPLAY MEMORY
                    ;
0ECD CD9610         DISP:   CALL    EXPR    ;GET TWO ADDRESSES
0ED0 D1                     POP     D       ;GET HIGH ADDRESS
0ED1 E1                     POP     H       ;GET LOW ADDRESS
0ED2 CD2510         DI0:    CALL    CRLF    ;PRINT CR,LF
0ED5 CD3010                 CALL    DADR    ;AND ADDRESS
0ED8 CDF60F         DI1:    CALL    BLK     ;PRINT A SPACE
0EDB 7E                     MOV     A,M     ;GET DATA BYTE
0EDC CD3510                 CALL    DBYTE   ;PRINT IT
0EDF CDA910                 CALL    HILO    ;TEST FOR COMPLETION
0EE2 DA2510                 JC      CRLF    ;FINISHED
0EE5 7D                     MOV     A,L     ;PRINT CR,LF ON MULTIPLE
0EE6 E60F                   ANI     0FH     ;OF 16
0EE8 C2D80E                 JNZ     DI1
0EEB C3D20E                 JMP     DI0
                    ;
                    ;____________________________________________________
                    ;
                    ;       E COMMAND - EXIT MONITOR
                    ;
0EEE 2A2D2E         MEXIT:  LHLD    SPSAV   ;RESTORE STACK PTR. &
0EF1 F9                     SPHL            ;RESTORE SYTEM
0EF2 21BE06                 LXI     H,MSG
0EF5 C3AE06                 JMP     INT2A
```

```
                        ;
                        ;________________________________________________
                        ;
                        ;       F COMMAND - FILL MEMORY WITH CONSTANT
                        ;
0EF8 0C                 FILL:   INR     C       ;GET 3 ARGUMENTS
0EF9 CD9610                     CALL    EXPR
0EFC C1                         POP     B       ;C = 8 BIT CONSTANT
0EFD D1                         POP     D       ;HIGH ADDRESS
0EFE E1                         POP     H       ;LOW ADDRESS
0EFF 71                 FI1:    MOV     M,C     ;STORE CONSTANT IN MEMORY
0F00 CDA910                     CALL    HILO    ;FINISHED?
0F03 D2FF0E                     JNC     FI1
0F06 C9                         RET
                        ;
                        ;________________________________________________
                        ;
                        ;       G COMMAND - GO TO ADDRESS
                        ;
0F07 0D                 GOTO:   DCR     C       ;GET ADDRESS
0F08 CD9610                     CALL    EXPR
0F0B E1                         POP     H
0F0C E9                         PCHL            ;BEGIN EXECUTION
                        ;
                        ;________________________________________________
                        ;
                        ;       H COMMAND - COMPUTE SUM AND DIFFERENCE
                        ;
0F0D CD9610             HEXN:   CALL    EXPR    ;GET 2 NUMBERS
0F10 CD2510                     CALL    CRLF    ;SEND CR,LF
0F13 D1                         POP     D       ;DE=P2
0F14 E1                         POP     H       ;HL=P1
0F15 E5                         PUSH    H
0F16 19                         DAD     D       ;HL=P1+P2
0F17 CD3010                     CALL    DADR    ;DISPLAY SUM
0F1A CDF60F                     CALL    BLK     ;PRINT A SPACE
0F1D E1                         POP     H       ;RECOVER P1
0F1E 7D                         MOV     A,L     ;COMPUTE HL-DE
0F1F 93                         SUB     E
0F20 6F                         MOV     L,A
0F21 7C                         MOV     A,H
0F22 9A                         SBB     D
0F23 67                         MOV     H,A
0F24 CD3010                     CALL    DADR    ;DISPLAY DIFFERENCE
0F27 C9                         RET
                        ;
                        ;________________________________________________
                        ;
                        ;       I COMMAND - INPUT PORT AND DISPLAY
                        ;
0F28 0D                 PORTI:  DCR     C       ;GET ONE PARAMETER
0F29 CD9610                     CALL    EXPR
0F2C CD2510                     CALL    CRLF    ;SPACE TO NEXT LINE
0F2F C1                         POP     B
0F30 78                         MOV     A,B     ;IS IT 8 BITS
0F31 B7                         ORA     A
```

```
0F32 C2660E              JNZ    ERROR
0F35 3EDB                MVI    A,0DBH    ;OPCODE FOR INPUT
0F37 320030              STA    3000H     ;STORE IN RAM
0F3A 79                  MOV    A,C       ;RECOVER PORT ADDRESS
0F3B 320130              STA    3001H     ;STORE IN RAM
0F3E 3EC3                MVI    A,0C3H    ;OPCODE FOR JUMP
0F40 320230              STA    3002H     ;STORE IN RAM
0F43 214C0F              LXI    H,RETADR;RETURN ADDRESS
0F46 220330              SHLD   3003H     ;STORE IN RAM
0F49 C30030              JMP    3000H     ;EXECUTE INPUT CODE
0F4C =           RETADR  EQU    $         ;THIS IS WHERE WE WANT TO
                                          ;JUMP BACK TO
0F4C C33510              JMP    DBYTE     ;DISPLAY RESULT AND RETURN
                 ;
                 ;____________________________________________________
                 ;
                 ;       M COMMAND - MOVE A BLOCK OF MEMORY
                 ;
0F4F 0C          MOVE:   INR    C         ;GET 3 PARAMETERS
0F50 CD9610              CALL   EXPR
0F53 C1                  POP    B         ;DESTINATION ADDRESS
0F54 D1                  POP    D         ;SOURCE END ADDRESS
0F55 E1                  POP    H         ;SOURCE BEGIN ADDRESS
0F56 7E          MVO:    MOV    A,M       ;GET A BYTE
0F57 02                  STAX   B
0F58 03                  INX    B
0F59 CDA910              CALL   HILO
0F5C D2560F              JNC    MVO
0F5F C9                  RET
                 ;
                 ;____________________________________________________
                 ;
                 ;       O COMMAND - OUTPUT VALUE TO PORT
                 ;
0F60 CD9610      PORTO:  CALL   EXPR      ;GET 2 VALUES
0F63 CD2510              CALL   CRLF
0F66 C1                  POP    B         ;OUTPUT VALUE
0F67 D1                  POP    D         ;OUTPUT ADDRESS
0F68 78                  MOV    A,B       ;8 BITS?
0F69 B2                  ORA    D
0F6A C2660E              JNZ    ERROR
0F6D 3ED3                MVI    A,0D3H    ;OPCODE FOR OUTPUT
0F6F 320030              STA    3000H     ;STORE IN MEMORY
0F72 7B                  MOV    A,E       ;RECOVER PORT ADDRESS
0F73 320130              STA    3001H     ;STORE IN RAM
0F76 3EC3                MVI    A,0C3H    ;OPCODE FOR JUMP
0F78 320230              STA    3002H     ;STORE IN RAM
0F7B 21850F              LXI    H,RETAD   ;RETURN ADDRESS
0F7E 220330              SHLD   3003H     ;STORE IN RAM
0F81 79                  MOV    A,C
0F82 C30030              JMP    3000H     ;NOW OUTPUT DATA
0F85 =           RETAD   EQU    $         ;THIS IS WHERE WE WANT
0F85 C9                  RET              ;TO RETURN
                 ;
                 ;____________________________________________________
                 ;
```

```
                        ;       S COMMAND - SUBSTITUTE MEMORY
                        ;
0F86 CDD910     SUBS:   CALL    PARAM   ;GET MEMORY ADDRESS
0F89 D8                 RC              ;ONLY CR SO RETURN
0F8A 7E         SU0:    MOV     A,M     ;GET CURRENT VALUE
0F8B CD3510             CALL    DBYTE
0F8E 0E2D               MVI     C,'-'   ;PROMPT CHARACTER
0F90 CD1010             CALL    CO
0F93 CD0F11             CALL    PCHK
0F96 D8                 RC
0F97 CAA10F             JZ      SU1     ;SPACE - CONTINUE
0F9A EB                 XCHG            ;SAVE MEMORY ADDRESS
0F9B CDDF10             CALL    PA0     ;GET NEW VALUE
0F9E EB                 XCHG            ;E=VALUE
0F9F 73                 MOV     M,E     ;STORE NEW VALUE
0FA0 D8                 RC              ;RETURN IF CR
0FA1 23         SU1:    INX     H       ;NEXT ADDRESS
0FA2 C38A0F             JMP     SU0
                        ;
                        ;__________________________________________________
                        ;
                        ;       T COMMAND - DISPLAY DATE AND TIME
                        ;
0FA5 CD2510     MTIME:  CALL    CRLF
0FA8 DBA4               IN      MONTH   ;GET MONTH
0FAA CDDE0F             CALL    CNVDS   ;CONVERT TO BCD & DISPLAY
0FAD 0E2F               MVI     C,'/'
0FAF CD1010             CALL    CO      ;PRINT A /
0FB2 DBA3               IN      DAY     ;GET DAY
0FB4 CDDE0F             CALL    CNVDS   ;CONVERT AND DISPLAY
0FB7 0E2F               MVI     C,'/'   ;PRINT A /
0FB9 CD1010             CALL    CO
0FBC DBA5               IN      YEAR    ;GET YEAR
0FBE CDDE0F             CALL    CNVDS   ;CONVERT AND DISPLAY
0FC1 CDF60F             CALL    BLK     ;PRINT A SPACE
0FC4 DBA2               IN      HOUR    ;GET HOURS
0FC6 CDDE0F             CALL    CNVDS   ;CONVERT AND DISPLAY
0FC9 0E3A               MVI     C,':'   ;PRINT A :
0FCB CD1010             CALL    CO
0FCE DBA1               IN      MINUTE  ;GET MINUTES
0FD0 CDDE0F             CALL    CNVDS   ;CONVERT AND DISPLAY
0FD3 0E3A               MVI     C,':'   ;PRINT  A :
0FD5 CD1010             CALL    CO
0FD8 DBA0               IN      SECOND  ;GET SECONDS
0FDA CDDE0F             CALL    CNVDS   ;CONVERT AND DISPLAY
0FDD C9                 RET
0FDE 4F         CNVDS:  MOV     C,A     ;MOVE VALUE TO C
0FDF CDE60F             CALL    BINBCD  ;CONVERT TO BCD
0FE2 CD3510             CALL    DBYTE   ;DISPLAY ON CONSOLE
0FE5 C9                 RET
                        ;
                        ;
                        ;**************************************************
                        ;
                        ;       UTILITY ROUTINES
                        ;
```

```
                        ;*************************************************
                        ;
                        ;
                        ;------
                        ;BINBCD
                        ;------
                        ;FUNCTION: BINBCD
                        ;DESCRIPTION: BINBCD CONVERTS THE BINARY VALUE
                        ;             IN REGISTER C INTO A BCD EQUIVALENT
                        ;             AND RETURNS THE RESULT IN A
                        ;
0FE6 59                 BINBCD: MOV     E,C       ;MOVE VALUE TO E
0FE7 1600                       MVI     D,0       ;CLEAR D
0FE9 010A00                     LXI     B,0AH     ;SET B&C=10BASE10
0FEC CD6710                     CALL    DIVIDE    ;DE=DE/BC HL=DE MOD BC
0FEF 7B                         MOV     A,E
0FF0 07                         RLC
0FF1 07                         RLC
0FF2 07                         RLC
0FF3 07                         RLC
0FF4 B5                         ORA     L
0FF5 C9                         RET
                        ;
                        ;---
                        ;BLK
                        ;---
                        ;FUNCTION: BLK
                        ;DESCRIPTION: ROUTINE BLK PRINTS A SPACE
                        ;
0FF6 0E20               BLK:    MVI     C,' '
0FF8 CD1010                     CALL    CO
0FFB C9                         RET
                        ;
                        ;--
                        ;CI
                        ;--
                        ;FUNCTION: CI
                        ;DESCRIPTION: CI (CONSOLE INPUT) ACCEPTS A
                        ;             CHARACTER FROM THE CONSOLE AND
                        ;             RETURNS IT IN THE A REGISTER
                        ;
0FFC DBED               CI:     IN      CONST     ;INPUT CONSOLE STATUS
0FFE E602                       ANI     RXRDY     ;CHARACTER?
1000 CAFC0F                     JZ      CI
1003 DBEC                       IN      CONSD     ;INPUT CHARACTER
1005 E67F                       ANI     7FH       ;STRIP PARITY
1007 FE61                       CPI     61H       ;CONVERT TO UPPER CASE
1009 D8                         RC
100A FE7B                       CPI     7BH
100C F0                         RP
100D E65F                       ANI     5FH
100F C9                         RET
                        ;
                        ;--
                        ;CO
                        ;--
```

```
                        ;FUNCTION: CO
                        ;DESCRIPTION: CO (CONSOLE OUTPUT) SENDS THE
                        ;             CHARACTER IN THE C REGISTER TO
                        ;             THE CONSOLE
                        ;
1010 DBED               CO:     IN      CONST    ;IS THE TRANSMITTER
1012 E601                       ANI     TXRDY    ;READY ?
1014 CA1010                     JZ      CO
1017 79                         MOV     A,C
1018 D3EC                       OUT     CONSD
101A C9                         RET
                        ;
                        ;----
                        ;CONV
                        ;----
                        ;FUNCTION: CONV
                        ;DESCRIPTION: CONV CONVERTS A 4 BIT HEX VALUE
                        ;             TO AN ASCII CHARACTER
                        ;
101B E60F               CONV:   ANI     0FH      ;STRIP TO 4 BITS
101D C690                       ADI     90H
101F 27                         DAA
1020 CE40                       ACI     40H
1022 27                         DAA
1023 4F                         MOV     C,A      ;RETURN IN C
1024 C9                         RET
                        ;
                        ;----
                        ;CRLF
                        ;----
                        ;FUNCTION: CRLF
                        ;DESCRIPTION: CRLF SENDS A CARRIAGE RETURN AND
                        ;             LINE FEED TO THE CONSOLE
                        ;
1025 0E0A               CRLF:   MVI     C,LF     ;SEND LF
1027 CDCA10                     CALL    OUTPUT
102A 0E0D                       MVI     C,CR     ;SEND CR
102C CDCA10                     CALL    OUTPUT
102F C9                         RET
                        ;
                        ;----
                        ;DADR
                        ;----
                        ;FUNCTION: DADR
                        ;DESCRIPTION: DADR DISPLAYS THE HL REGISTERS
                        ;             IN HEX FORMAT
                        ;
1030 7C                 DADR:   MOV     A,H      ;PRINT MSB OF ADDRESS
1031 CD3510                     CALL    DBYTE
1034 7D                         MOV     A,L      ;PRINT LSB OF ADDRESS
                        ;
                        ;-----
                        ;DBYTE
                        ;-----
                        ;FUNCTION: DBYTE
                        ;DESCRIPTION: DBYTE DISPLAYS THE BYTE IN THE
```

```
                    ;               A REGISTER IN HEX FORMAT
                    ;
1035 F5             DBYTE:  PUSH    PSW     ;SAVE BYTE
1036 0F                     RRC
1037 0F                     RRC
1038 0F                     RRC
1039 0F                     RRC
103A CD1B10                 CALL    CONV    ;CONVERT
103D CD1010                 CALL    CO      ;DISPLAY
1040 F1                     POP     PSW
1041 CD1B10                 CALL    CONV    ;CONVERT
1044 C31010                 JMP     CO      ;DISPLAY
                    ;
                    ;-----
                    ;DCONV
                    ;-----
                    ;FUNCTION: DCONV
                    ;DESCRIPTION: DCONV TAKES THE HEX NUMBER IN THE A
                    ;               REG(RANGE 0-99 DEC) AND CONVERTS IT
                    ;               TO TWO ASCII CHARACTERS FOR BASE 10
                    ;               DISPLAY - THEY ARE RETURNED IN C&B
                    ;
1047 4F             DCONV:  MOV     C,A     ;MOVE NUMBER TO C AND
1048 CDE60F                 CALL    BINBCD  ;CONVERT TO BCD
104B F5                     PUSH    PSW     ;SAVE VALUE
104C 0F                     RRC             ;SHIFT TO CONVERT MSB 1ST
104D 0F                     RRC
104E 0F                     RRC
104F 0F                     RRC
1050 CD1B10                 CALL    CONV    ;CONVERT TO ASCII
1053 F1                     POP     PSW     ;RECOVER LSB
1054 C5                     PUSH    B       ;SAVE MSB
1055 CD1B10                 CALL    CONV    ;CONVERT LSB TO ASCII
1058 51                     MOV     D,C     ;MOVE LSB
1059 C1                     POP     B       ;RECOVER LSB
105A 42                     MOV     B,D     ;PUT LSB IN B
105B C9                     RET
                    ;
                    ;-----
                    ;DELAY
                    ;-----
                    ;FUNCTION: DELAY
                    ;DESCRIPTION: THIS ROUTINE PROVIDES A DELAY FOR
                    ;               THE NUMBER OF MILLISECONDS GIVEN
                    ;               BY THE A REGISTER.
                    ;
105C 0660           DELAY:  MVI     B,60H   ;VALUE FOR 1 MILLISECOND
105E 05             DLY1:   DCR     B
105F C25E10                 JNZ     DLY1
1062 3D                     DCR     A
1063 C25C10                 JNZ     DELAY
1066 C9                     RET
                    ;
                    ;------
                    ;DIVIDE
                    ;------
```

```
                        ;FUNCTION: DIVIDE
                        ;DESCRIPTION: DIVIDE - DIVIDES THE CONTENTS OF DE
                        ;              BY THE CONTENTS OF BC AND LEAVES THE
                        ;              RESULT IN DE AND REMAINDER IN HL
                        ;
1067 210000             DIVIDE: LXI     H,0         ;CLEAR HL
106A 3E10                       MVI     A,16        ;NUMBER OF BITS TO USE
106C F5                 DIV05:  PUSH    PSW         ;SAVE A
106D 29                         DAD     H           ;SHR HL
106E EB                         XCHG
106F 97                         SUB     A           ;CLEAR A
1070 29                         DAD     H           ;SHR DE
1071 EB                         XCHG
1072 8D                         ADC     L           ;ADD L TO A
1073 91                         SUB     C           ;SUBTRACT C
1074 6F                         MOV     L,A         ;MAKE L=A
1075 7C                         MOV     A,H         ;MAKE A=H
1076 98                         SBB     B           ;SUBTRACT B W/ BORROW
1077 67                         MOV     H,A         ;MAKE H=A
1078 13                         INX     D
1079 D27E10                     JNC     DIV10       ;BRANCH IF NO CARRY
107C 09                         DAD     B           ;ADD BC
107D 1B                         DCX     D           ;RESTORE D
107E F1                 DIV10:  POP     PSW         ;RESTORE A
107F 3D                         DCR     A           ;DCR COUNT
1080 C26C10                     JNZ     DIV05
1083 C9                         RET
                        ;
                        ;----
                        ;DREG
                        ;----
                        ;FUNCTION: DREG
                        ;DESCRIPTION: ROUTINE DREG DISPLAYS THE CONTENTS
                        ;              OF A USER REGISTER
                        ;
1084 23                 DREG:   INX     H
1085 5E                         MOV     E,M         ;POINT AT DISPLACEMENT
1086 16FF                       MVI     D,0FFH
1088 23                         INX     H
1089 46                         MOV     B,M         ;PRECISION
108A 23                         INX     H
108B 1A                         LDAX    D
108C CD3510                     CALL    DBYTE
108F 05                         DCR     B
1090 F8                         RM                  ;RETURN IF 8 BIT REG
1091 1B                         DCX     D
1092 1A                         LDAX    D
1093 C33510                     JMP     DBYTE
                        ;
                        ;----
                        ;EXPR
                        ;----
                        ;FUNCTION: EXPR
                        ;DESCRIPTION: EXPR SCANS A SERIES OF ADDRESSES
                        ;
1096 CDD910             EXPR:   CALL PARAM          ;GET HEX NUMBER IN HL
```

```
1099 E3                 XTHL                    ;AND PUSH ONTO STACK
109A E5                 PUSH    H
109B 0D                 DCR     C
109C D2A310             JNC     EX0
109F C2660E             JNZ     ERROR
10A2 C9                 RET
10A3 C29610     EX0:    JNZ     EXPR    ;GET ANOTHER
10A6 C3660E             JMP     ERROR
                ;
                ;----
                ;HILO
                ;----
                ;FUNCTION: HILO
                ;DESCRIPTION: HILO COMPARES HL TO DE
                ;
10A9 23         HILO:   INX     H       ;CHECK FOR HL=FFFFH
10AA 7C                 MOV     A,H
10AB B5                 ORA     L
10AC 37                 STC
10AD C8                 RZ
10AE 7B                 MOV     A,E
10AF 95                 SUB     L
10B0 7A                 MOV     A,D
10B1 9C                 SBB     H
10B2 C9                 RET
                ;
                ;--
                ;MI
                ;--
                ;FUNCTION: MI
                ;DESCRIPTION: MI IS THE MODEM PORT INPUT ROUTINE
                ;
10B3 DBF1       MI:     IN      MODMST  ;GET PORT STATUS
10B5 E602               ANI     RXRDY
10B7 CAB310             JZ      MI
10BA DBF0               IN      MODEM   ;GET CHARACTER
10BC E67F               ANI     7FH
10BE C9                 RET
                ;
                ;--
                ;MO
                ;--
                ;FUNCTION: MO
                ;DESCRIPTION: MO IS THE MODEM PORT OUTPUT ROUTINE
                ;
10BF DBF1       MO:     IN      MODMST  ;GET STATUS
10C1 E601               ANI     TXRDY
10C3 CABF10             JZ      MO
10C6 79                 MOV     A,C     ;OUTPUT C REGISTER
10C7 D3F0               OUT     MODEM
10C9 C9                 RET
                ;
                ;------
                ;OUTPUT
                ;------
                ;FUNCTION: OUTPUT
```

```
                   ;DESCRIPTION: OUTPUT LOOKS AT THE IOBYTE TO
                   ;              DETERMINE WHETHER DATA GOES TO
                   ;              THE MOOEM OR THE CONSOLE
                   ;
10CA 3A472E        OUTPUT: LDA     IOBYTE
10CD 87                    ADO     A
10CE CAD510                JZ      OUT1
10D1 CDBF1D                CALL    MO          ;MOOEM
1004 C9                    RET
1005 CD1010        OUT1:   CALL    CO          ;CONSOLE
1008 C9                    RET
                   ;
                   ;-----
                   ;PARAM
                   ;-----
                   ;FUNCTION: PARAM
                   ;DESCRIPTION: ROUTINE PARAM SCANS A HEX NUMBER
                   ;              THAT IS RETURNED IN HL
                   ;
10D9 CDOF11        PARAM:  CALL    PCHK        ;GET FIRST CHARACTER
10DC CA660E                JZ      ERROR       ;ERROR IF CR
100F 210000        PA0:    LXI     H,0         ;INITIALIZE NUMBER
10E2 47            PA1:    MOV     B,A         ;SAVE CHARACTER IN B
10E3 COFD10                CALL    NIBBLE      ;CONVERT TO HEX
10E6 DAF510                JC      PA2
10E9 29                    DAD     H
10EA 29                    0A0     H
10EB 29                    DA0     H
10EC 29                    0A0     H           ;X16
10ED B5                    ORA     L
10EE 6F                    MOV     L,A
10EF CD6D11                CALL    TI          ;GET SUBSEQUENT CHARACTERS
10F2 C3E210                JMP     PA1
10F5 78            PA2:    MOV     A,B
10F6 CD1211                CALL    P2C
10F9 C2660E                JNZ     ERROR
10FC C9                    RET
                   ;
                   ;------
                   ;NIBBLE
                   ;------
                   ;FUNCTION: NIBBLE
                   ;OESCRIPTION: NIBBLE CONVERTS ASCII TO HEX
                   ;
10FD D630          NIBBLE: SUI     '0'
10FF D8                    RC                  ;RETURN IF INVALIO
1100 C6E9                  AOI     '0' - 'G'
1102 08                    RC
1103 C606                  AOI     6
1105 F20B11                JP      NI0
1108 C607                  AOI     7
110A D8                    RC
110B C60A          NI0:    AOI     10
110D B7                    ORA     A
110E C9                    RET
                   ;
```

```
                   ;----
                   ;PCHK
                   ;----
                   ;FUNCTION: PCHK
                   ;DESCRIPTION: PCHK CHECKS FOR A VALID DELIMITER
                   ;
110F CD6D11        PCHK:   CALL    TI          ;GET AND ECHO CHARACTER
1112 FE20          P2C:    CPI     ' '         ;SPACE?
1114 C8                    RZ
1115 FE2C                  CPI     ','         ;COMMA?
1117 C8                    RZ
1118 FE0D                  CPI     CR
111A 37                    STC
111B C8                    RZ
111C 3F                    CMC
111D C9                    RET
                   ;
                   ;------
                   ;SAVTIM
                   ;------
                   ;FUNCTION: SAVTIM
                   ;DESCRIPTION: SAVTIM SAVES THE CURRENT DATE AND
                   ;             TIME IN THE MEMORY LOCATIONS POINTED
                   ;             TO BY THE HL REGISTERS - IT IS STORED
                   ;             IN ASCII AND USES 18 BYTES OF THE
                   ;             FORM: MM/DD/YY HH:MM:SS
                   ;
111E E5            SAVTIM: PUSH    H           ;SAVE ADDRESS
111F DBA4                  IN      MONTH       ;GET MONTH
1121 CD4710                CALL    DCONV       ;CONVERT
1124 E1                    POP     H           ;RECOVER ADDRESS AND STORE
1125 162F                  MVI     D,'/'       ;INFO IN MEMORY
1127 CD6611                CALL    SAVT1
112A E5                    PUSH    H           ;DO AGAIN FOR DAY
112B DBA3                  IN      DAY
112D CD4710                CALL    DCONV
1130 E1                    POP     H
1131 162F                  MVI     D,'/'
1133 CD6611                CALL    SAVT1
1136 E5                    PUSH    H           ;DO AGAIN FOR YEAR
1137 DBA5                  IN      YEAR
1139 CD4710                CALL    DCONV
113C E1                    POP     H
113D 1620                  MVI     D,' '
113F CD6611                CALL    SAVT1
1142 E5                    PUSH    H           ;DO AGAIN FOR HOUR
1143 DBA2                  IN      HOUR
1145 CD4710                CALL    DCONV
1148 E1                    POP     H
1149 163A                  MVI     D,':'
114B CD6611                CALL    SAVT1
114E E5                    PUSH    H           ;DO AGAIN FOR MINUTE
114F DBA1                  IN      MINUTE
1151 CD4710                CALL    DCONV
1154 E1                    POP     H
1155 163A                  MVI     D,':'
```

```
1157 CD6611              CALL    SAVT1
115A E5                  PUSH    H          ;DO ONCE MORE FOR SECONDS
115B DBA0                IN      SECOND
115D CD4710              CALL    DCONV
1160 E1                  POP     H
1161 71                  MOV     M,C
1162 23                  INX     H
1163 70                  MOV     M,B
1164 23                  INX     H
1165 C9                  RET
1166 71         SAVT1:   MOV     M,C
1167 23                  INX     H          ;MOVE C, B, &D REGISTERS
1168 70                  MOV     M,B        ;TO MEMORY
1169 23                  INX     H
116A 72                  MOV     M,D
116B 23                  INX     H
116C C9                  RET
                ;
                ;--
                ;TI
                ;--
                ;FUNCTION: TI
                ;DESCRIPTION: TI INPUTS A CHARACTER, ECHOS IT
                ;             AND RETURNS IT IN A
                ;
116D C5         TI:      PUSH    B          ;SAVE BC FOR CALLER
116E CDFC0F              CALL    CI
1171 FE03                CPI     03H        ;ABORT?
1173 CA660E              JZ      ERROR
1176 4F                  MOV     C,A        ;SET FOR DISPLAY
1177 CD1010              CALL    CO
117A 79                  MOV     A,C
117B C1                  POP     B
117C C9                  RET
```

APPENDIX B

SOFTWARE DEVELOPMENT SYSTEM

Appendix B. Software Development System

The CEL software development system operates very much like an Intel MDS system. It will run CP/M and ISIS operating systems. A simple bootstrap loader is provided to load and run the operating system.

The development system consists of the following hardware components:

- Intel SBC-80/24 CPU
- Chrislin 64K RAM memory
- Comark floppy disk subsystem
- Intel SBC-604/614 eight slot cardcage
- National Semiconductor BLC-665 power supply
- Data gathering device card set consisting of Digital Pathways TCU-410 clock module, Intel SBC-094 CMOS RAM memory, and Analog Devices RTI-1200 analog input system
- Racal-Vadic 3455 auto-answer modem

The system components are housed within an Optima Instrument Cabinet. Mounted on the front panel is the cardcage, power switch, and a momentary reset switch. The CPU, 64K RAM and floppy disk system must be installed to run the operating system. The other cards are optional and may be incorporated as the user desires.

When power is applied, the bootstrap and a simple 8080 monitor program is moved from PROM on the CPU to RAM starting at F800H. Program execution then begins at F800H. The software then determines if there is a CP/M system diskette in disk drive zero. If so, the operating system is loaded and executed. If there was no system diskette then the 80/24 monitor is entered. This is the same monitor that is described in Appendix A. This sequence of events occurs every time the reset switch is depressed.

The user should become familiar with the CP/M operating system manuals supplied with the device. Trouble-shooting procedures and schematics are included in the hardware reference manuals. A listing of the CP/M bootstrap software is included on the following pages.

```
                        ;**************************************************
                        ;
                        ;       CP/M BOOT PROGRAM AND MONITOR
                        ;
                        ;       AUGUST 1981
                        ;
                        ;**************************************************
                        ;
                        ;       SYSTEM EQUATES
                        ;
                        ;INTEL 80/24 DEVICE CONSTANTS:
                        ;
                        ;
                        ;
                        ;ON-BOARD 8253 PIT
                        ;
 00DF =                 TIM1CP  EQU     0DFH    ;TIMER CONTROL PORT
 00DE =                 TIM1C2  EQU     0DEH    ;COUNTER 2(USART TIMER)
 0007 =                 BAUD96  EQU     007H    ;RATE FACTOR FOR 9600 BPS
 00B6 =                 T1C2M3  EQU     0B6H    ;BYTE TO SET COUNTER 2 TO
                                                ;MODE 3 (RATE GENERATOR)
                        ;
                        ;MULTIMODULE 8253 PIT
                        ;
 00FB =                 TIM2CP  EQU     0FBH    ;TIMER CONTROL PORT
 00FA =                 TIM2C2  EQU     0FAH    ;COUNTER 2(USART TIMER)
 0040 =                 BAUD12  EQU     040H    ;RATE FACTOR FOR 1200 BPS
 00B6 =                 T2C2M3  EQU     0B6H    ;BYTE TO SET COUNTER 2 TO
                                                ;MODE 3 (RATE GENERATOR)
                        ;
                        ;8251A USART EQUATES
                        ;
                        ;MODE INSTRUCTION
 0002 =                 BAUD16  EQU     002H    ;16X BAUD RATE FACTOR
 0008 =                 DBIT7   EQU     008H    ;7 DATA BITS
 000C =                 DBIT8   EQU     00CH    ;8 DATA BITS
 0030 =                 EVEN    EQU     030H    ;EVEN PARITY
 0010 =                 ODD     EQU     010H    ;ODD PARITY
 0040 =                 STOP1   EQU     040H    ;1 STOP BIT
                        ;
                        ;COMMAND INSTRUCTION
 0001 =                 TXEN    EQU     001H    ;TRANSMIT ENABLE
 0002 =                 DTR     EQU     002H    ;DATA TERMINAL READY
 0004 =                 RXEN    EQU     004H    ;RECEIVE ENABLE
 0010 =                 ERST    EQU     010H    ;ERROR RESET
 0020 =                 RTS     EQU     020H    ;REQUEST TO SEND
 0040 =                 URST    EQU     040H    ;INTERNAL RESET
                        ;
                        ;STATUS READ DEFINITION
 0001 =                 TXRDY   EQU     001H    ;TRANSMITTER READY
 0002 =                 RXRDY   EQU     002H    ;RECEIVER READY
 0004 =                 TXMT    EQU     004H    ;TRANSMITTER EMPTY
 0038 =                 RXERR   EQU     038H    ;RECEIVER ERROR
 0080 =                 DSR     EQU     080H    ;DATA SET READY
                        ;
                        ;ON-BOARD 8251A PCI (LOCAL TERMINAL)
```

```
00ED =          CONST   EQU     0EDH    ;CONSOLE STATUS/CONTROL
00EC =          CONSD   EQU     0ECH    ;CONSOLE DATA PORT
                ;
                ;MULTIMODULE 8251A PCI (MODEM)
00F1 =          MODMST  EQU     0F1H    ;MODEM STATUS/CONTROL
00F0 =          MODEM   EQU     0F0H    ;MOOEM DATA PORT
                ;
                ;
                ;OTHER DEVICE CONSTANTS:
                ;
                ;DIGITAL PATHWAYS TCU-410 SYSTEM CLOCK
                ;
00A0 =          SECOND  EQU     0A0H    ;SECONDS COUNTER
00A1 =          MINUTE  EQU     0A1H    ;MINUTE COUNTER
00A2 =          HOUR    EQU     0A2H    ;HOUR COUNTER
00A3 =          DAY     EQU     0A3H    ;DAY COUNTER
00A4 =          MONTH   EQU     0A4H    ;MONTH COUNTER
00A5 =          YEAR    EQU     0A5H    ;YEAR COUNTER
00A1 =          SETTIM  EQU     0A1H    ;OUTPUT TO SET TIME
00A3 =          SETDAT  EQU     0A3H    ;OUTPUT TO SET DATE
00A6 =          CLSTAT  EQU     0A6H    ;INPUT CLOCK STATUS
00A6 =          RFC     EQU     0A6H    ;RESET FAST CLOCK
                ;
0000                    ORG     0
                ;
                ;ENABLE 8259
                ;
0000 F3                 DI
0001 3E12               MVI     A,12H   ;DISABLE INTERRUPTS AND
0003 D3D8               OUT     0D8H    ;MASK ALL INTERRUPTS
0005 3E00               MVI     A,0
0007 D3D9               OUT     0D9H
0009 3EFF               MVI     A,0FFH
000B D3D9               OUT     0D9H
                ;
                ;
                ;INITIALIZE 8253 INTERVAL TIMERS
                ;
000D 3EB6       INIT:   MVI     A,T1C2M3        ;USE ON-BOARD 8253
000F D3DF               OUT     TIM1CP          ;FOR 9600 BPS CLOCK
0011 3E07               MVI     A,BAUD96        ;FOR TERMINAL USART
0013 D3DE               OUT     TIM1C2          ;(COUNTER 2 IS SET
0015 3E00               MVI     A,0             ;TO MODE 3)
0017 D3DE               OUT     TIM1C2
                ;
0019 3EB6               MVI     A,T2C2M3        ;USE MULTIMODULE
001B D3FB               OUT     TIM2CP          ;8253 FOR 1200 BPS
001D 3E40               MVI     A,BAUD12        ;CLOCK FOR MODEM
001F D3FA               OUT     TIM2C2          ;USART (COUNTER 2
0021 3E00               MVI     A,0             ;IS SET TO MODE 3)
0023 D3FA               OUT     TIM2C2
                ;
                ;INITIALIZE 8251A USARTS
                ;
0025 D3ED               OUT     CONST           ;GET TERMINAL USART
0027 D3ED               OUT     CONST           ;INTO A KNOWN STATE
```

```
0029 D3ED            OUT     CONST
002B 3E40            MVI     A,URST
002D D3ED            OUT     CONST           ;RESET USART
002F 3E4E            MVI     A,BAUD16+DBIT8+STOP1    ;DATA FORMAT
0031 D3ED            OUT     CONST
0033 3E37            MVI     A,TXEN+RXEN+RTS+DTR+ERST;COMMAND
0035 D3ED            OUT     CONST           ;INSTRUCTION
0037 DBEC            IN      CONSD           ;CLEAR DATA PORT
             ;
0039 D3F1            OUT     MODMST          ;GET MODEM USART
003B D3F1            OUT     MODMST          ;INTO A KNOWN STATE
003D D3F1            OUT     MODMST
003F 3E40            MVI     A,URST
0041 D3F1            OUT     MODMST          ;RESET USART
0043 3E4A            MVI     A,BAUD16+DBIT7+STOP1    ;DATA FORMAT
0045 D3F1            OUT     MODMST
0047 3E17            MVI     A,TXEN+RXEN+ERST+DTR    ;COMMAND
0049 D3F1            OUT     MODMST
004B DBF0            IN      MODEM           ;CLEAR DATA PORT
             ;
             ;
             ;NOW WE WILL MOVE THE BOOT AND MONITOR INTO
             ;HIGH RAM SO THAT CP/M CAN USE 62K OF RAM
             ;
             ;
004D 216800          LXI     H,068H  ;HL = PROM ADDRESS
0050 1100F8          LXI     D,0F800H;DE = ADDRESS IN RAM WHERE
                                     ;WE WILL START LOADING
0053 0696            MVI     B,150   ;COUNT FACTOR
0055 0E06            MVI     C,6     ;LOOP COUNT
0057 7E      MOVE1:  MOV     A,M     ;GET A BYTE
0058 12              STAX    D       ;MOVE TO HIGH RAM
0059 23              INX     H       ;INCREMENT ADDRESS POINTERS
005A 13              INX     D
005B 05              DCR     B       ;DECREMENT COUNT
005C C25700          JNZ     MOVE1
005F 0696            MVI     B,150   ;NEW COUNT FACTOR
0061 0D              DCR     C       ;DECREMENT LOOP COUNTER
0062 C25700          JNZ     MOVE1
0065 C300F8          JMP     0F800H  ;FINISHED MOVE SO GO TO
                                     ;RAM AND TRY TO BOOT CP/M
                                     ;OTHERWISE GO TO MONITOR
             ;
             ;
             ;----------------------------------------------
             ;
             ;       BOOT PROCEDURE
             ;
             ;----------------------------------------------
             ;
F800                 ORG     0F800H
             ;
             ;DISK I/O EQUATES:
             ;
0078 =       BASE    EQU     78H     ;BASE ADDRESS OF CONTROLLER
0078 =       DSTAT   EQU     BASE    ;DISK STATUS INPUT PORT
```

```
0079 =        LOWAD    EQU      BASE+1   ;IOPB LOW ADDRESS
007A =        HIGHAD   EOU      BASE+2   ;IOPB HIGH ADDRESS
0079 =        RTYPE    EQU      BASE+1   ;DISK ERROR TYPE
007B =        RBYTE    EQU      BASE+3   ;DISK ERROR BYTE
007F =        RESET    EQU      BASE+7   ;DISK RESET
000D =        CR       EQU      ODH      ;CARRIAGE RETURN
000A =        LF       EQU      OAH      ;LINE FEED
0003 =        IOBYTE   EQU      3        ;ADDRESS USED BY PRINT DRIVER
1000 =        IOPBAD   EQU      1000H    ;ADDRESS WHERE IOPB IS
                                         ;MOVED FOR PROCESSING
                                         ;BY THE DISK CONTROLLER
                       ;
                       ;        JUMP TABLE
                       ;
                       ;THIS TABLE PROVIDES CP/M AN ENTRY POINT INTO THE
                       ;THE MONITOR I/O ROUTINES
                       ;
F800 C315F8                     JMP      INIT1    ;INITIALIZE FOR BOOT
F803 C311FA                     JMP      CI       ;CONSOLE INPUT
F806 C311FA                     JMP      CI
F809 C325FA                     JMP      CO       ;CONSOLE OUTPUT
F80C C3BBFA                     JMP      LO       ;LIST OUTPUT
F80F C3BBFA                     JMP      LO
F812 C344FA                     JMP      CSTS     ;CHECK SYSTEM INPUT STATUS
                       ;
                       ;        INITIALIZATION OF BOOT MONITOR
                       ;
F815 F3                INIT1:   DI                ;DISABLE INTERRUPTS
F816 31F0FF                     LXI      SP,OFFFOH ;LOAD STACK POINTER
F819 3E80                       MVI      A,80H    ;TURN OFF SHADOW PROM
F81B D3E7                       OUT      OE7H     ;ON 80/24 BY OUTPUTTING
F81D 3E00                       MVI      A,0      ;TO AN 8255
F81F D3E6                       OUT      OE6H
F821 3EC8                       MVI      A,200    ;LOAD IOBYTE FOR 200
F823 320300                     STA      IOBYTE   ;CHARACTERS TO PRINTER
F826 AF                         XRA      A        ;CLEAR A
F827 D37F                       OUT      RESET    ;RESET DISK CONTROLLER
                       ;
                       ;CHECK TO SEE IF DISK DRIVE 0 IS READY
                       ;IF SO THEN DO CP/M INITIALIZATION,
                       ;IF NOT THEN GO TO MONITOR
                       ;
F829 DB78                       IN       DSTAT
F82B E609                       ANI      9
F82D CA7FF8                     JZ       DEBUG
F830 E601                       ANI      1        ;SEE IF DRIVE 0 READY
F832 CA7FF8                     JZ       DEBUG
                       ;
                       ;DISK IS PRESENT SO RESET CONTROLLER
                       ;
F835 D37F              WRDY0:   OUT      RESET
F837 DB78                       IN       DSTAT
F839 E604                       ANI      4
F83B DB79                       IN       RTYPE
F83D DB7B                       IN       RBYTE
F83F C235F8                     JNZ      WRDY0
```

```
F842 210010            LXI     H,IOPBAD;MOVE IOPB DOWN TO
F845 1178F8            LXI     D,IOPB  ;CONTROLLER ADDRESSABLE
F848 0607              MVI     B,7     ;RAM SPACE
F84A 1A        MIOPB:  LDAX    D
F84B 77                MOV     M,A
F84C 23                INX     H
F84D 13                INX     D
F84E 05                DCR     B
F84F C24AF8            JNZ     MIOPB
F852 210010            LXI     H,IOPBAD;NOW OUTPUT ADDRESS TO
F855 7D                MOV     A,L     ;THE CONTROLLER
F856 D379              OUT     LOWAD   ;LOW ADDRESS
F858 7C                MOV     A,H
F859 D37A              OUT     HIGHAD  ;HIGH ADDRESS
                ;
                ;WAIT UNTIL LOAD IS COMPLETE THEN GO TO LOADED
                ;PROGRAM
                ;
F85B DB78      WDRDYO: IN      DSTAT
F85D E604              ANI     4
F85F CA5BF8            JZ      WDRDYO
F862 3E00              MVI     A,0     ;COVER UP INTEL MDS
F864 320B30            STA     300BH   ;BOOT-SWITCH READ
F867 320C30            STA     300CH
F86A 320D30            STA     300DH
F86D 3A0030            LDA     3000H   ;MAKE SURE WE GOT A GOOD
F870 FE31              CPI     31H     ;BOOT
F872 CA0030            JZ      3000H   ;CP/M BOOT ?
F875 C37FF8            JMP     DEBUG   ;BOOT NO GOOD, GO TO MONITOR
                ;
                ;DISK I/O PARAMETER BLOCK (IOPB) FOR BOOT
                ;
F878 80        IOPB:   DB      80H     ;IOCW, NO UPDATE BIT SET
F879 04                DB      4       ;READ DISK 0
F87A 1A                DB      26      ;READ 26 SECTORS
F87B 00                DB      0       ;FROM TRACK 0
F87C 01                DB      1       ;SECTOR 1
F87D 0030              DW      3000H   ;STARTING ADDRESS FOR LOAD
                ;
                ;
                ;**************************************************
                ;
                ;       SBC 80/24 MONITOR
                ;
                ;**************************************************
                ;
                ;
F87F 218FF8    DEBUG:  LXI     H,SIGNON        ;SEND THE SIGNON
F882 7E        DEB1:   MOV     A,M             ;MESSAGE TO THE
F883 B7                ORA     A               ;CONSOLE
F884 CAACF8            JZ      START
F887 4F                MOV     C,A
F888 CD25FA            CALL    CO
F88B 23                INX     H
F88C C382F8            JMP     DEB1
                ;
```

```
F88F 0D0A        SIGNON: DB      CR,LF
F891 5342432038          DB      'SBC 80/24 MONITOR'
F8A2 0D0A0D0A00          DB      CR,LF,CR,LF,0
F8A7 0E23        ERROR:  MVI     C,'#'     ;ERROR PROMPT
F8A9 CD25FA              CALL    CO
                 ;
                 ;       MAIN COMMAND LOOP
                 ;
F8AC FB          START:  EI                ;ENABLE INTERRUPTS
F8AD CD3AFA              CALL    CRLF      ;OUTPUT CR,LF
F880 0E2E                MVI     C,'.'     ;PROMT CHARACTER
F8B2 CD25FA              CALL    CO
F8B5 CD6BFB              CALL    TI        ;GET CHARACTER AND ECHO IT
F8B8 FE0D                CPI     CR        ;CARRIAGE RETURN?
F8BA CAACF8              JZ      START
F8BD D641                SUI     'A'       ;OTHERWISE TEST FOR (A-Z)
F8BF FAA7F8              JM      ERROR
F8C2 FE1A                CPI     LCT       ;TOO LARGE?
F8C4 F2A7F8              JP      ERROR
F8C7 0E02                MVI     C,2       ;ASSUME COMMAND NEEDS
                                           ;TWO ARGUMENTS
F8C9 11ACF8              LXI     D,START   ;SIMULATE A CALL
F8CC D5                  PUSH    D
F8CD 21DAF8              LXI     H,CTBL    ;JUMP TABLE
F8D0 5F                  MOV     E,A       ;COMPUTE INDEX
F8D1 1600                MVI     D,0
F8D3 19                  DAD     D
F8D4 19                  DAD     D
F8D5 7E                  MOV     A,M       ;GET LSB OF ADDRESS
F8D6 23                  INX     H
F8D7 66                  MOV     H,M
F8D8 6F                  MOV     L,A
F8D9 E9                  PCHL              ;GO TO ROUTINE
                 ;
                 ;________________________________________________
                 ;
                 ;       COMMAND BRANCH TABLE
                 ;
F8DA A7F8        CTBL:   DW      ERROR     ;A
F8DC A7F8                DW      ERROR     ;B
F8DE A7F8                DW      ERROR     ;C
F8E0 0EF9                DW      DISP      ;D - DISPLAY MEMORY
FBE2 A7F8                DW      ERROR     ;E
F8E4 2FF9                DW      FILL      ;F - FILL MEMORY
F8E6 3EF9                DW      GOTO      ;G - GOTO MEMORY ADDRESS
F8E8 44F9                DW      HEXN      ;H - HEX SUM & DIFFERENCE
F8EA 5FF9                DW      PORTI     ;I - INPUT PORT
F8EC A7F8                DW      ERROR     ;J
F8EE A7F8                DW      ERROR     ;K
F8F0 A7F8                DW      ERROR     ;L
F8F2 75F9                DW      MOVE      ;M - MOVE MEMORY
F8F4 A7F8                DW      ERROR     ;N
F8F6 86F9                DW      PORTO     ;O - OUTPUT TO PORT
F8F8 A7F8                DW      ERROR     ;P
F8FA A7F8                DW      ERROR     ;Q
F8FC A7F8                DW      ERROR     ;R
```

```
F8FE 9BF9               DW      SUBS    ;S - SUBSTITUTE MEMORY
F900 BAF9               DW      MTIME   ;T - DISPLAY TIME
F902 A7F8               DW      ERROR   ;U
F904 A7F8               DW      ERROR   ;V
F906 A7F8               DW      ERROR   ;W
F908 A7F8               DW      ERROR   ;X
F90A A7F8               DW      ERROR   ;Y
F90C A7F8               DW      ERROR   ;Z
001A =          LCT     EQU     ($-CTBL)/2      ;TABLE SIZE
                ;
                ;------------------------------------------------
                ;
                ;       D COMMAND - DISPLAY MEMORY
                ;
F90E CD9EFA     DISP:   CALL    EXPR    ;GET TWO ADDRESSES
F911 D1                 POP     D       ;GET HIGH ADDRESS
F912 E1                 POP     H       ;GET LOW ADDRESS
F913 CD3AFA     DIO:    CALL    CRLF    ;PRINT CR,LF
F916 CD4DFA             CALL    DADR    ;AND ADDRESS
F919 CD0BFA     DI1:    CALL    BLK     ;PRINT A SPACE
F91C 7E                 MOV     A,M     ;GET DATA BYTE
F91D CD52FA             CALL    DBYTE   ;PRINT IT
F920 CDB1FA             CALL    HILO    ;TEST FOR COMPLETION
F923 DA3AFA             JC      CRLF    ;FINISHED
F926 7D                 MOV     A,L     ;PRINT CR,LF ON MULTIPLE
F927 E60F               ANI     0FH     ;OF 16
F929 C219F9             JNZ     DI1
F92C C313F9             JMP     DIO
                ;
                ;------------------------------------------------
                ;
                ;       F COMMAND - FILL MEMORY WITH CONSTANT
                ;
F92F 0C         FILL:   INR     C       ;GET 3 ARGUMENTS
F930 CD9EFA             CALL    EXPR
F933 C1                 POP     B       ;C = 8 BIT CONSTANT
F934 D1                 POP     D       ;HIGH ADDRESS
F935 E1                 POP     H       ;LOW ADDRESS
F936 71         FI1:    MOV     M,C     ;STORE CONSTANT IN MEMORY
F937 CDB1FA             CALL    HILO    ;FINISHED?
F93A D236F9             JNC     FI1
F93D C9                 RET
                ;
                ;------------------------------------------------
                ;
                ;       G COMMAND - GO TO ADDRESS
                ;
F93E 0D         GOTO:   DCR     C       ;GET STARTING ADDRESS
F93F CD9EFA             CALL    EXPR
F942 E1                 POP     H
F943 E9                 PCHL            ;EXECUTE
                ;
                ;------------------------------------------------
                ;
                ;       H COMMAND - COMPUTE SUM AND DIFFERENCE
                ;
```

```
F944 CD9EFA    HEXN:   CALL    EXPR     ;GET 2 NUMBERS
F947 CD3AFA            CALL    CRLF     ;SEND CR,LF
F94A D1                POP     D        ;DE=P2
F94B E1                POP     H        ;HL=P1
F94C E5                PUSH    H
F94D 19                DAD     D        ;HL=P1+P2
F94E CD4DFA            CALL    DADR     ;DISPLAY SUM
F951 CD0BFA            CALL    BLK      ;PRINT A SPACE
F954 E1                POP     H        ;RECOVER P1
F955 7D                MOV     A,L      ;COMPUTE HL-DE
F956 93                SUB     E
F957 6F                MOV     L,A
F958 7C                MOV     A,H
F959 9A                SBB     D
F95A 67                MOV     H,A
F95B CD4DFA            CALL    DAOR     ;DISPLAY DIFFERENCE
F95E C9                RET
                       ;
                       ;______________________________________________
                       ;
                       ;       I COMMAND - INPUT PORT AND DISPLAY
                       ;
F95F 0D        PORTI:  DCR     C        ;GET ONE PARAMETER
F960 CD9EFA            CALL    EXPR
F963 CD3AFA            CALL    CRLF     ;SPACE TO NEXT LINE
F966 C1                POP     B
F967 78                MOV     A,B      ;IS IT 8 BITS
F968 B7                ORA     A
F969 C2A7F8            JNZ     ERROR    ;JUMP IF NOT
F96C 79                MOV     A,C      ;RECOVER PORT ADDRESS
F96D 3271F9            STA     IPORT    ;PUT IN RAM
F970 DB00              IN      00H      ;INPUT VALUE
F971 =         IPORT   EQU     $-1
F972 C352FA            JMP     DBYTE    ;DISPLAY RESULT AND RETURN
                       ;
                       ;______________________________________________
                       ;
                       ;       M COMMAND - MOVE A BLOCK OF MEMORY
                       ;
F975 0C        MOVE:   INR     C        ;GET 3 PARAMETERS
F976 CD9EFA            CALL    EXPR
F979 C1                POP     B        ;DESTINATION ADDRESS
F97A D1                POP     D        ;SOURCE END ADDRESS
F97B E1                POP     H        ;SOURCE BEGIN ADDRESS
F97C 7E        MVO:    MOV     A,M      ;GET A BYTE
F97D 02                STAX    B
F97E 03                INX     B
F97F CDB1FA            CALL    HILO
F982 D27CF9            JNC     MVO
F985 C9                RET
                       ;
                       ;______________________________________________
                       ;
                       ;       O COMMAND - OUTPUT VALUE TO PORT
                       ;
F986 CD9EFA    PORTO:  CALL    EXPR     ;GET 2 VALUES
```

```
F989 CD3AFA          CALL    CRLF
F98C C1              POP     B        ;OUTPUT VALUE
F98D D1              POP     D        ;OUTPUT ADDRESS
F98E 78              MOV     A,B      ;8 BITS?
F98F B2              ORA     D
F990 C2A7F8          JNZ     ERROR
F993 7B              MOV     A,E      ;RECOVER PORT ADDRESS
F994 3299F9          STA     OPORT    ;PUT IN RAM
F997 79              MOV     A,C      ;GET VALUE TO OUTPUT
F998 D300            OUT     00H
F999 =       OPORT   EQU     $-1
F99A C9              RET              ;TO RETURN
             ;
             ;------------------------------------------------
             ;
             ;       S COMMAND - SUBSTITUTE MEMORY
             ;
F99B CD26FB  SUBS:   CALL    PARAM    ;GET MEMORY ADDRESS
F99E D8              RC               ;ONLY CR SO RETURN
F99F 7E      SU0:    MOV     A,M      ;GET CURRENT VALUE
F9A0 CD52FA          CALL    DBYTE
F9A3 0E2D            MVI     C,'-'    ;PROMPT CHARACTER
F9A5 CD25FA          CALL    CO
F9A8 CD5CFB          CALL    PCHK
F9AB D8              RC
F9AC CAB6F9          JZ      SU1      ;SPACE - CONTINUE
F9AF EB              XCHG             ;SAVE MEMORY ADDRESS
F9B0 CD2CFB          CALL    PA0      ;GET NEW VALUE
F9B3 EB              XCHG             ;E=VALUE
F9B4 73              MOV     M,E      ;STORE NEW VALUE
F9B5 D8              RC               ;RETURN IF CR
F9B6 23      SU1:    INX     H        ;NEXT ADDRESS
F9B7 C39FF9          JMP     SU0
             ;
             ;------------------------------------------------
             ;
             ;       T COMMAND - DISPLAY DATE AND TIME
             ;
F9BA CD3AFA  MTIME:  CALL    CRLF
F9BD DBA4            IN      MONTH    ;GET MONTH
F9BF CDF3F9          CALL    CNVDS    ;CONVERT TO BCD & DISPLAY
F9C2 0E2F            MVI     C,'/'
F9C4 CD25FA          CALL    CO       ;PRINT A /
F9C7 DBA3            IN      DAY      ;GET DAY
F9C9 CDF3F9          CALL    CNVDS    ;CONVERT AND DISPLAY
F9CC 0E2F            MVI     C,'/'    ;PRINT A /
F9CE CD25FA          CALL    CO
F9D1 DBA5            IN      YEAR     ;GET YEAR
F9D3 CDF3F9          CALL    CNVDS    ;CONVERT AND DISPLAY
F9D6 CD0BFA          CALL    BLK      ;PRINT A SPACE
F9D9 DBA2            IN      HOUR     ;GET HOURS
F9DB CDF3F9          CALL    CNVDS    ;CONVERT AND DISPLAY
F9DE 0E3A            MVI     C,':'    ;PRINT A :
F9E0 CD25FA          CALL    CO
F9E3 DBA1            IN      MINUTE   ;GET MINUTES
F9E5 CDF3F9          CALL    CNVDS    ;CONVERT AND DISPLAY
```

```
F9E8 0E3A              MVI     C,':'    ;PRINT  A :
F9EA CD25FA            CALL    CO
F9ED DBA0              IN      SECOND   ;GET SECONDS
F9EF CDF3F9            CALL    CNVDS    ;CONVERT AND DISPLAY
F9F2 C9                RET
F9F3 4F        CNVDS:  MOV     C,A      ;MOVE VALUE TO C
F9F4 CDFBF9            CALL    BINBCD   ;CONVERT TO BCD
F9F7 CD52FA            CALL    DBYTE    ;DISPLAY ON CONSOLE
F9FA C9                RET
               ;
               ;************************************************
               ;
               ;        UTILITY ROUTINES
               ;
               ;************************************************
               ;
               ;
               ;------
               ;BINBCD
               ;------
               ;FUNCTION: BINBCD
               ;DESCRIPTION: BINBCD CONVERTS THE BINARY VALUE
               ;             IN REGISTER C INTO A BCD EQUIVALENT
               ;             AND RETURNS THE RESULT IN A
               ;
F9FB 59        BINBCD: MOV     E,C      ;MOVE VALUE TO E
F9FC 1600              MVI     D,0      ;CLEAR D
F9FE 010A00            LXI     B,0AH    ;SET B&C=10BASE10
FA01 CD6FFA            CALL    DIVIDE   ;DE=DE/BC HL=DE MOD BC
FA04 7B                MOV     A,E
FA05 07                RLC
FA06 07                RLC
FA07 07                RLC
FA0B 07                RLC
FA09 B5                ORA     L
FA0A C9                RET
               ;
               ;---
               ;BLK
               ;---
               ;FUNCTION: BLK
               ;DESCRIPTION: ROUTINE BLK PRINTS A SPACE
               ;
FA0B 0E20      BLK:    MVI     C,' '
FA0D CD25FA            CALL    CO
FA10 C9                RET
               ;
               ;--
               ;CI
               ;--
               ;FUNCTION: CI
               ;DESCRIPTION: CI (CONSOLE INPUT) ACCEPTS A
               ;             CHARACTER FROM THE CONSOLE AND
               ;             RETURNS IT IN THE A REGISTER
               ;
FA11 DBED      CI:     IN      CONST    ;INPUT CONSOLE STATUS
```

```
FA13 E602              ANI     RXRDY   ;CHARACTER?
FA15 CA11FA            JZ      CI
FA18 DBEC              IN      CONSD   ;INPUT CHARACTER
FA1A E67F              ANI     7FH     ;STRIP PARITY
FA1C FE61              CPI     61H     ;CONVERT TO UPPER CASE
FA1E D8                RC
FA1F FE7B              CPI     7BH
FA21 F0                RP
FA22 E65F              ANI     5FH
FA24 C9                RET
                ;
                ;--
                ;CO
                ;--
                ;FUNCTION: CO
                ;DESCRIPTION: CO (CONSOLE OUTPUT) SENDS THE
                ;              CHARACTER IN THE C REGISTER TO
                ;              THE CONSOLE
                ;
FA25 DBED       CO:    IN      CONST   ;IS THE TRANSMITTER
FA27 E601              ANI     TXRDY   ;READY ?
FA29 CA25FA            JZ      CO
FA2C 79                MOV     A,C
FA2D D3EC              OUT     CONSD
FA2F C9                RET
                ;
                ;----
                ;CONV
                ;----
                ;FUNCTION: CONV
                ;DESCRIPTION: CONV CONVERTS A 4 BIT HEX VALUE
                ;              TO AN ASCII CHARACTER
                ;
FA30 E60F       CONV:  ANI     0FH     ;STRIP TO 4 BITS
FA32 C690              ADI     90H
FA34 27                DAA
FA35 CE40              ACI     40H
FA37 27                DAA
FA38 4F                MOV     C,A     ;RETURN IN C
FA39 C9                RET
                ;
                ;----
                ;CRLF
                ;----
                ;FUNCTION: CRLF
                ;DESCRIPTION: CRLF SENDS A CARRIAGE RETURN AND
                ;              LINE FEED TO THE CONSOLE
                ;
FA3A 0E0D       CRLF:  MVI     C,CR    ;SEND CR
FA3C CD25FA            CALL    CO
FA3F 0E0A              MVI     C,LF    ;SEND   LF
FA41 C325FA            JMP     CO
                ;
                ;----
                ;CSTS
                ;----
```

```
                        ;FUNCTION: CSTS
                        ;DESCRIPTION: CSTS EXAMINES THE CONSOLE STATUS
                        ;
FA44 DBED               CSTS:   IN      CONST
FA46 E602                       ANI     2       ;SEE IF CHARACTER READY
FA48 3E00                       MVI     A,0     ;ASSUME NOT
FA4A C8                         RZ
FA4B 2F                         CMA
FA4C C9                         RET             ;RETURN
                        ;
                        ;----
                        ;DADR
                        ;----
                        ;FUNCTION: DADR
                        ;DESCRIPTION: DADR DISPLAYS THE HL REGISTERS
                        ;             IN HEX FORMAT
                        ;
FA4D 7C                 DADR:   MOV     A,H     ;PRINT MSB OF ADDRESS
FA4E CD52FA                     CALL    DBYTE
FA51 7D                         MOV     A,L     ;PRINT LSB OF ADDRESS
                        ;
                        ;-----
                        ;DBYTE
                        ;-----
                        ;FUNCTION: DBYTE
                        ;DESCRIPTION: DBYTE DISPLAYS THE BYTE IN THE
                        ;             A REGISTER IN HEX FORMAT
                        ;
FA52 F5                 DBYTE:  PUSH    PSW     ;SAVE BYTE
FA53 0F                         RRC
FA54 0F                         RRC
FA55 0F                         RRC
FA56 0F                         RRC
FA57 CD30FA                     CALL    CONV    ;CONVERT
FA5A CD25FA                     CALL    CO      ;DISPLAY
FA5D F1                         POP     PSW
FA5E CD30FA                     CALL    CONV    ;CONVERT
FA61 C325FA                     JMP     CO      ;DISPLAY
                        ;
                        ;-----
                        ;DELAY
                        ;-----
                        ;FUNCTION: DELAY
                        ;DESCRIPTION: DELAY PROVIDES A 1 MILLISECOND
                        ;             DELAY GIVEN BY THE NUMBER OF
                        ;             MILLISECONDS IN THE A REGISTER.
                        ;
FA64 0660               DELAY:  MVI     B,60H   ;VALUE FOR 1 MILLISECOND
FA66 05                 DLY1:   DCR     B
FA67 C266FA                     JNZ     DLY1
FA6A 3D                         DCR     A
FA6B C264FA                     JNZ     DELAY
FA6E C9                         RET
                        ;
                        ;------
                        ;DIVIDE
```

```
                        ;------
                        ;FUNCTION: DIVIDE
                        ;DESCRIPTION: DIVIDE - DIVIDES THE CONTENTS OF DE
                        ;             BY THE CONTENTS OF BC AND LEAVES THE
                        ;             RESULT IN DE AND REMAINDER IN HL
                        ;
FA6F 210000     DIVIDE: LXI     H,0       ;CLEAR HL
FA72 3E10               MVI     A,16      ;NUMBER OF BITS TO USE
FA74 F5         DIV05:  PUSH    PSW       ;SAVE A
FA75 29                 DAD     H         ;SHR HL
FA76 EB                 XCHG
FA77 97                 SUB     A         ;CLEAR A
FA78 29                 DAD     H         ;SHR DE
FA79 EB                 XCHG
FA7A 8D                 ADC     L         ;ADD L TO A
FA7B 91                 SUB     C         ;SUBTRACT C
FA7C 6F                 MOV     L,A       ;MAKE L=A
FA7D 7C                 MOV     A,H       ;MAKE A=H
FA7E 98                 SBB     B         ;SUBTRACT B W/ BORROW
FA7F 67                 MOV     H,A       ;MAKE H=A
FA80 13                 INX     D
FA81 D286FA             JNC     DIV10     ;BRANCH IF NO CARRY
FA84 09                 DAD     B         ;ADD BC
FA85 1B                 DCX     D         ;RESTORE D
FA86 F1         DIV10:  POP     PSW       ;RESTORE A
FA87 3D                 DCR     A         ;DCR COUNT
FA88 C274FA             JNZ     DIV05
FA8B C9                 RET
                        ;
                        ;----
                        ;DREG
                        ;----
                        ;FUNCTION: DREG
                        ;DESCRIPTION: ROUTINE DREG DISPLAYS THE CONTENTS
                        ;             OF A USER REGISTER
                        ;
FA8C 23         DREG:   INX     H
FA8D 5E                 MOV     E,M       ;POINT AT DISPLACEMENT
FA8E 16FF               MVI     D,0FFH
FA90 23                 INX     H
FA91 46                 MOV     B,M       ;PRECISION
FA92 23                 INX     H
FA93 1A                 LDAX    D
FA94 CD52FA             CALL    D8YTE
FA97 05                 DCR     B
FA98 F8                 RM                ;RETURN IF 8 BIT REG
FA99 1B                 DCX     D
FA9A 1A                 LDAX    D
FA98 C352FA             JMP     DBYTE
                        ;
                        ;----
                        ;EXPR
                        ;----
                        ;FUNCTION: EXPR
                        ;DESCRIPTION: EXPR SCANS A SERIES OF ADDRESSES
                        ;
```

```
FA9E CD26FB     EXPR:    CALL PARAM       ;GET HEX NUMBER IN HL
FAA1 E3                  XTHL             ;AND PUSH ONTO STACK
FAA2 E5                  PUSH     H
FAA3 0D                  DCR      C
FAA4 D2ABFA              JNC      EXO
FAA7 C2A7F8              JNZ      ERROR
FAAA C9                  RET
FAAB C29EFA     EXO:     JNZ      EXPR     ;GET ANOTHER
FAAE C3A7F8              JMP      ERROR
                ;
                ;----
                ;HILO
                ;----
                ;FUNCTION: HILO
                ;DESCRIPTION: HILO COMPARES HL TO DE
                ;
FAB1 23         HILO:    INX      H        ;CHECK FOR HL=FFFFH
FAB2 7C                  MOV      A,H
FAB3 B5                  ORA      L
FAB4 37                  STC
FAB5 C8                  RZ
FAB6 7B                  MOV      A,E
FAB7 95                  SUB      L
FAB8 7A                  MOV      A,D
FAB9 9C                  SBB      H
FABA C9                  RET
                ;
                ;--
                ;LO
                ;--
                ;FUNCTION: LO
                ;DESCRIPTION: LO IS THE MODEM AND LIST OUTPUT
                ;             ROUTINE THAT USES THE ETX-ACK
                ;             PROTOCOL (SEND 200 BYTES AND AN ETX
                ;             TO THE PRINTER THEN WAIT UNTIL WE
                ;             RECEIVE AN ACK)
                ;
FABB 3EB6       LO:      MVI      A,T2C2M3 ;REINITIALIZE 8253 TIMER
FABD D3FB                OUT      TIM2CP
FABF 3E40                MVI      A,BAUD12
FAC1 D3FA                OUT      TIM2C2
FAC3 3E00                MVI      A,0
FAC5 D3FA                OUT      TIM2C2
FAC7 D3F1                OUT      MODMST   ;REINITIALIZE 8251 USART
FAC9 D3F1                OUT      MODMST
FACB D3F1                OUT      MODMST
FACD 3E40                MVI      A,URST
FACF D3F1                OUT      MODMST
FAD1 3E4A                MVI      A,BAUD16+DBIT7+STOP1
FAD3 D3F1                OUT      MODMST
FAD5 3E37                MVI      A,TXEN+RXEN+ERST+RTS+DTR
FAD7 D3F1                OUT      MODMST
FAD9 DBF0                IN       MODEM
FADB 3E12                MVI      A,18     ;DELAY FOR A SHORT WHILE
FADD CD64FA              CALL     DELAY
FAE0 3A0300              LDA      IOBYTE   ;HOW MANY CHARACTERS HAVE
```

```
FAE3 47                 MOV     B,A       ;WE SENT ?
FAE4 3E00               MVI     A,0
FAE6 80                 ADD     B
FAE7 CAFAFA             JZ      L02       ;WE'VE SENT 200
FAEA DBF1       L01:    IN      MODMST    ;>200, SEND ANOTHER
FAEC E601               ANI     TXRDY
FAEE CAEAFA             JZ      L01
FAF1 79                 MOV     A,C
FAF2 D3F0               OUT     MODEM
FAF4 05                 DCR     B         ;DECREMENT COUNT
FAF5 78                 MOV     A,B
FAF6 320300             STA     IOBYTE
FAF9 C9                 RET
FAFA DBF1       L02:    IN      MODMST    ;SEND LAST CHARACTER AND
FAFC E601               ANI     TXRDY     ;THE ETX
FAFE CAFAFA             JZ      L02
FB01 79                 MOV     A,C
FB02 D3F0               OUT     MODEM
FB04 DBF1       L03:    IN      MODMST
FB06 E601               ANI     TXRDY
FB08 CA04FB             JZ      L03
FB0B 3E03               MVI     A,03H
FB0D D3F0               OUT     MODEM
FB0F 3EC8               MVI     A,200
FB11 320300             STA     IOBYTE    ;RELOAD THE IOBYTE FOR 200
FB14 DBF1       L04:    IN      MODMST    ;WAIT FOR THE ACK
FB16 E602               ANI     RXRDY
FB18 CA14FB             JZ      L04
FB1B DBF0               IN      MODEM
FB1D FE06               CPI     06H
FB1F CA25FB             JZ      L05
FB22 C314FB             JMP     L04
FB25 C9         L05:    RET
                ;
                ;-----
                ;PARAM
                ;-----
                ;FUNCTION: PARAM
                ;DESCRIPTION: ROUTINE PARAM SCANS A HEX NUMBER
                ;             THAT IS RETURNED IN HL
                ;
FB26 CD5CFB     PARAM:  CALL    PCHK      ;GET FIRST CHARACTER
FB29 CAA7FB             JZ      ERROR     ;ERROR IF CR
FB2C 210000     PA0:    LXI     H,0       ;INITIALIZE NUMBER
FB2F 47         PA1:    MOV     B,A       ;SAVE CHARACTER IN B
FB30 CD4AFB             CALL    NIBBLE    ;CONVERT TO HEX
FB33 DA42FB             JC      PA2
FB36 29                 DAD     H
FB37 29                 DAD     H
FB38 29                 DAD     H
FB39 29                 DAD     H         ;X16
FB3A B5                 ORA     L
FB3B 6F                 MOV     L,A
FB3C CD6BFB             CALL    TI        ;GET SUBSEQUENT CHARACTERS
FB3F C32FFB             JMP     PA1
FB42 78         PA2:    MOV     A,B
```

```
FB43 CD5FFB             CALL    P2C
FB46 C2A7F8             JNZ     ERROR
FB49 C9                 RET
                ;
                ;------
                ;NIBBLE
                ;------
                ;FUNCTION: NIBBLE
                ;DESCRIPTION: NIBBLE CONVERTS ASCII TO HEX
                ;
FB4A D630       NIBBLE: SUI     '0'
FB4C D8                 RC              ;RETURN IF INVALID
FB4D C6E9               ADI     '0' - 'G'
FB4F D8                 RC
FB50 C606               ADI     6
FB52 F258FB             JP      NIO
FB55 C607               ADI     7
FB57 D8                 RC
FB58 C60A       NIO:    ADI     10
FB5A B7                 ORA     A
FB5B C9                 RET
                ;
                ;----
                ;PCHK
                ;----
                ;FUNCTION: PCHK
                ;DESCRIPTION: PCHK CHECKS FOR A VALID DELIMITER
                ;
FB5C CD6BFB     PCHK:   CALL    TI      ;GET AND ECHO CHARACTER
FB5F FE20       P2C:    CPI     ' '     ;SPACE?
FB61 C8                 RZ
FB62 FE2C               CPI     ','     ;COMMA?
FB64 C8                 RZ
FB65 FE0D               CPI     CR
FB67 37                 STC
FB68 C8                 RZ
FB69 3F                 CMC
FB6A C9                 RET
                ;
                ;--
                ;TI
                ;--
                ;FUNCTION: TI
                ;DESCRIPTION: TI INPUTS A CHARACTER, ECHOS IT
                ;             AND RETURNS IT IN A
                ;
FB6B C5         TI:     PUSH    B       ;SAVE BC FOR CALLER
FB6C CD11FA             CALL    CI
FB6F FE03               CPI     03H     ;ABORT?
FB71 CAA7F8             JZ      ERROR
FB74 4F                 MOV     C,A     ;SET FOR DISPLAY
FB75 CD25FA             CALL    CO
FB78 79                 MOV     A,C
FB79 C1                 POP     B
FB7A C9                 RET
                ;
```

APPENDIX C

INTERDATA COMMUNICATIONS PACKAGE

Appendix C. Interdata Communications Package

The software developed for the Interdata system is organized into four phases of operation:

- Automatic initiation of data solicitation.
- Transfer of data from the microprocessor to the Interdata.
- Maintenance of a data log file on the Interdata disk subsystem.
- Error detection and error logging.

Each of these phases of operation are detailed in subsequent sections of this document. As an aid to these discussions it is recommended that the reader refer to the following documents with particular attention paid to the topics indicated:

FORTRAN VII Reference Manual 29-658R01

Topics: Data Elements

Common Storage Areas

FORTRAN VII Run Time Library S29-581R04 Real Time Extensions

Topics: Intertask Communications

Task Manipulation Operations

Task Handled Traps

System Clock

File Manipulation Operations

System I/O Operations

OS/32 System Planning and Configuration Guide 48-024R00

Topics: Object-Level SYSGEN

Standard OS/32 Devices

System Operation

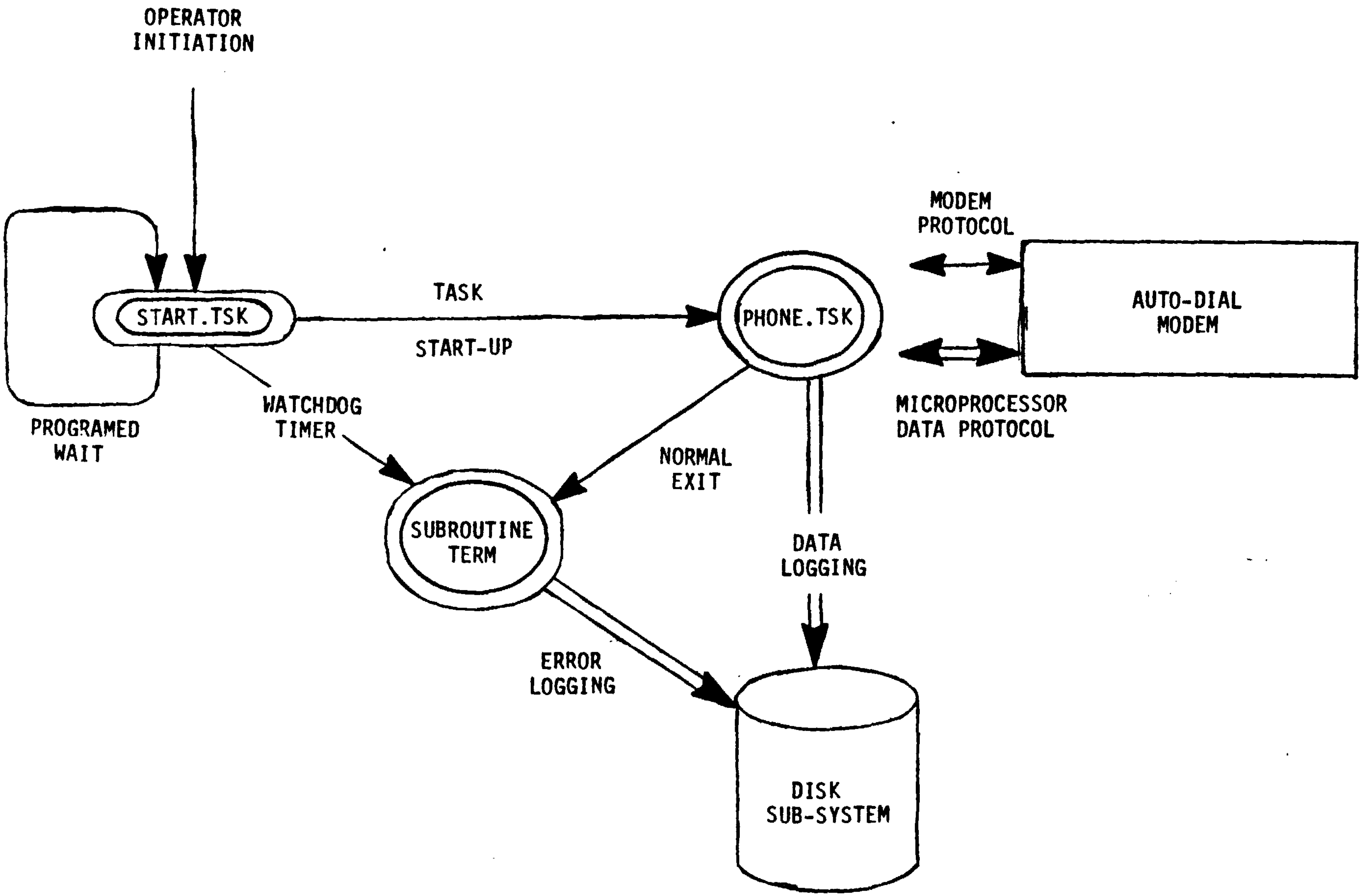
The software system is comprised of two tasks, START.TSK and PHONE.TSK. Included in PHONE.TSK is a subroutine named TERM which controls the orderly termination of system operation. These modules interact as described below and illustrated in the accompanying figure.

START.TSK - This task is loaded and started one time only. At a fixed interval (currently set to six hours) this task will load PHONE.TSK into a foreground program area and start its execution. At that point, START.TSK will initiate its own watchdog timer to ensure that PHONE.TSK reaches a normal termination within a reasonable length of time (currently set to two minutes). After the watchdog interval expires, a task trap is generated which will terminate the execution of PHONE.TSK. This abnormal task termination is noted by an appropriate error message generated by subroutine TERM.

PHONE.TSK - This task opens the data logging disk file or creates one if none exists. It then dials the phone to evoke the microprocessor data transmission routines. At this point, it expects to receive a prompt character from the microprocessor ('.'). Upon receiving the prompt, the Interdata sends a display command ('D') to the microprocessor at which time it expects to receive a data block. The end-of-data marker for this transfer has been defined as an asterisk ('*') in the first column of any line. After receiving the data block, this information is appended to the end of the data logging file on the disk subsystem.

Subroutine TERM - This subroutine is included as a part of PHONE.TSK but can be called by either task (PHONE.TSK or START.TSK). The purpose of this subroutine is to close all device and file channels then terminate program execution.

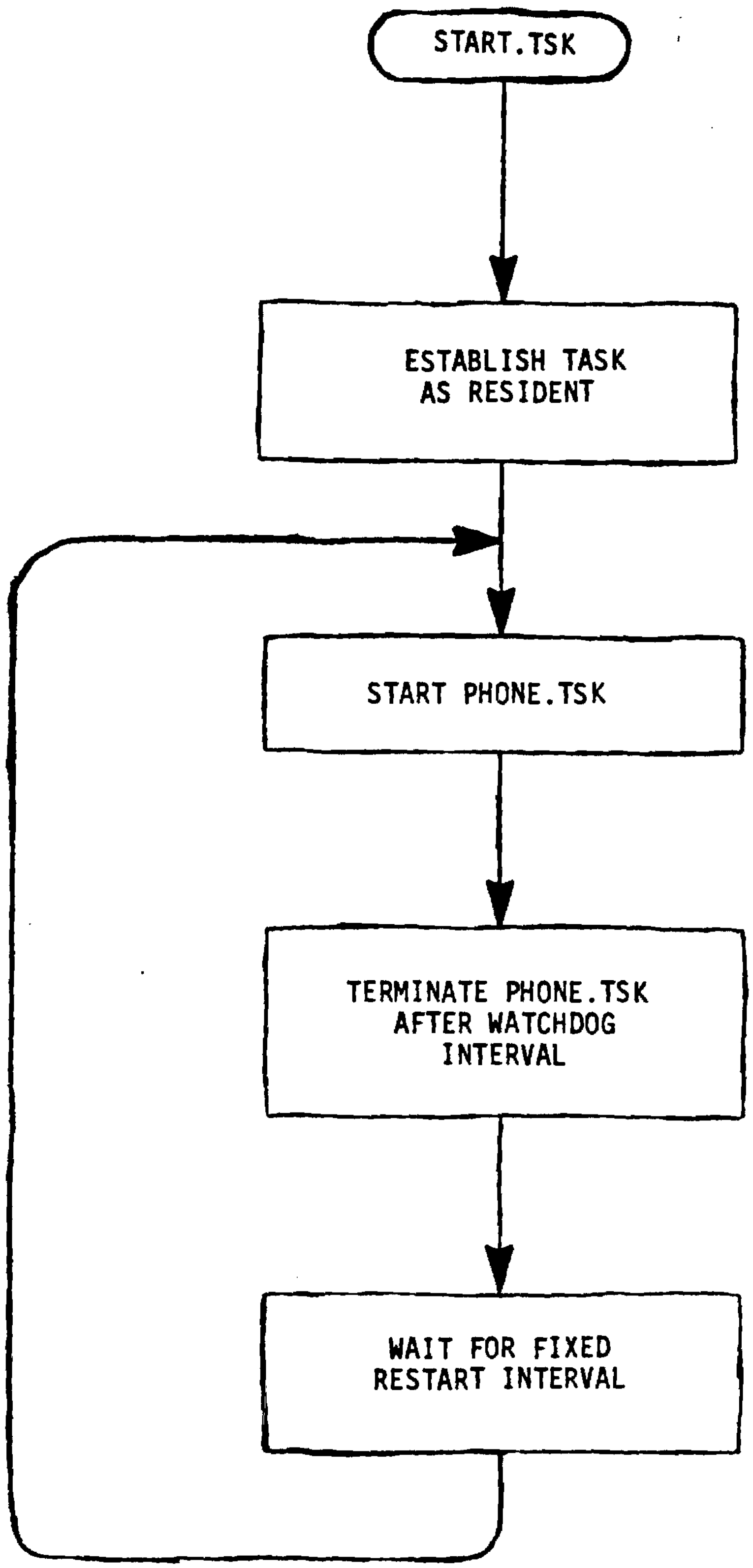
The specific action to be taken is determined by the value of the variable 'CODE'. If 'CODE' is not equal to five (5), then control did not reach the normal termination point and subroutine TERM was called by the watchdog timer in START.TSK. If this is the case, then the appropriate error message is to be recorded in the data logging file before termination.



INTERDATA SOFTWARE MODULE INTERACTION

TASK START

Purpose:	To control the periodic initiation of data gathering via task 'PHONE'.
Support Modules:	FORTRAN Run-Time Library
Support Files:	Task file named 'PHONE.TSK' (must be in user 0 directory)
Task Requirements:	Task control option Task communications option Resident task option Run from system console
Start Sequence:	LO START TA START ST
Termination Sequence:	CA START TA START O NON CA START


START.TSK
ESTABLISH TASK AS RESIDENT
START PHONE.TSK
TERMINATE PHONE.TSK AFTER WATCHDOG INTERVAL
WAIT FOR FIXED RESTART INTERVAL

FORTRAN-VIID R04-00

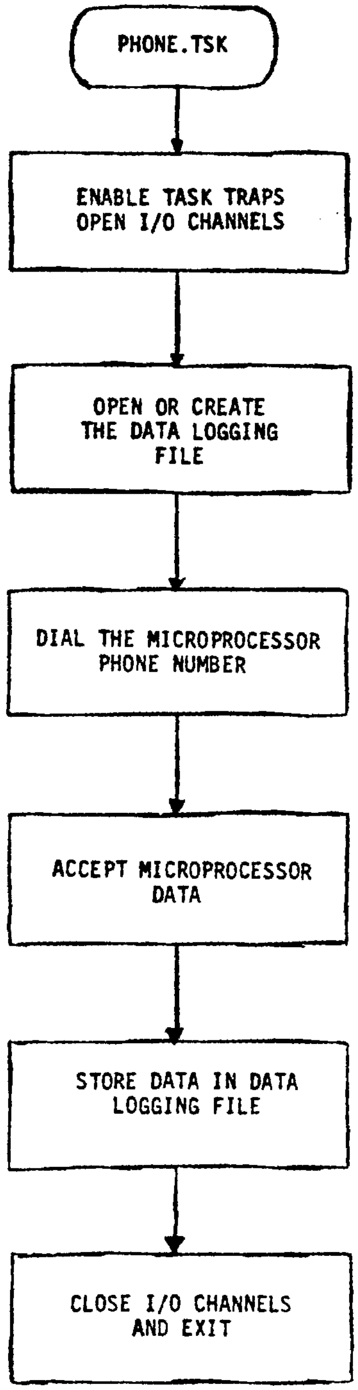
FORTRAN VIID: LICENSED RESTRICTED RIGHTS AS STATED IN LICENSE L-0174

```
                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                 C
                 C          ----   START.TSK V1.0    ----
                 C
                 C      THIS TASK IS THE KERNEL OF THE INTERDATA
                 C      TELEPHONE DATA GATHERING SYSTEM. AN
                 C      OPERATOR MUST START THIS TASK ONCE, THEN
                 C      AT PERIOIC INTERVALS THIS TASK WILL
                 C      INITIATE OPERATION OF TASK 'PHONE'. BOTH
                 C      TASKS MUST BE SAVED IN ACCOUNT ZERO (0)
                 C      IN ORDER TO UTILIZE THE PERIODIC RESART
                 C      FEATURE.
                 C
                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                 C
      000000I           IMPLICIT INTEGER (A-Z)
      000006I           DIMENSION TIMEA(3),TNAME(2),FNAME(3)
      000006I           DATA TNAME /'PHONE    '/
      000006I           DATA FNAME /'PHONE.TSK    '/
                 C
      000006I           CALL EST(0,STAT)
                 C
      000050I    1      CALL TIME(TIMEA)
      000068I           CALL OPENW (5,FNAME,0,0,0,STAT)
      0000B4I           CALL LOAD(TNAME,5,STAT)
      0000DCI           CALL CLOSE(5,STAT)
      000100I           CALL START (TNAME,0,0,0,STAT)
      000140I           CALL WAIT(2,3,STAT)
      000170I           CALL QUEUE(TNAME,0,STAT)
                 C
                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                 C
                 C      THIS NEXT SECTION OF CODE PAUSES THE
                 C      DESIRED WAIT TIME BEFORE RESTARTING TASK
                 C      'PHONE'. AN EXACT START TIME MAY BE SPEC-
                 C      IFIED BY USING A 'TRNON' SVC REQUEST RATHER
                 C      THEN A 'START' SVC REQUEST. NOTE THAT A
                 C      THIRD TASK IS REQUIRED FOR THIS FEATURE.
                 C
                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                 C
      000198I           CALL WAIT(358,3,STAT)
      0001C8I           GOTO 1
                 C
                 C
                 C
                 C
      0001CCI           END
```

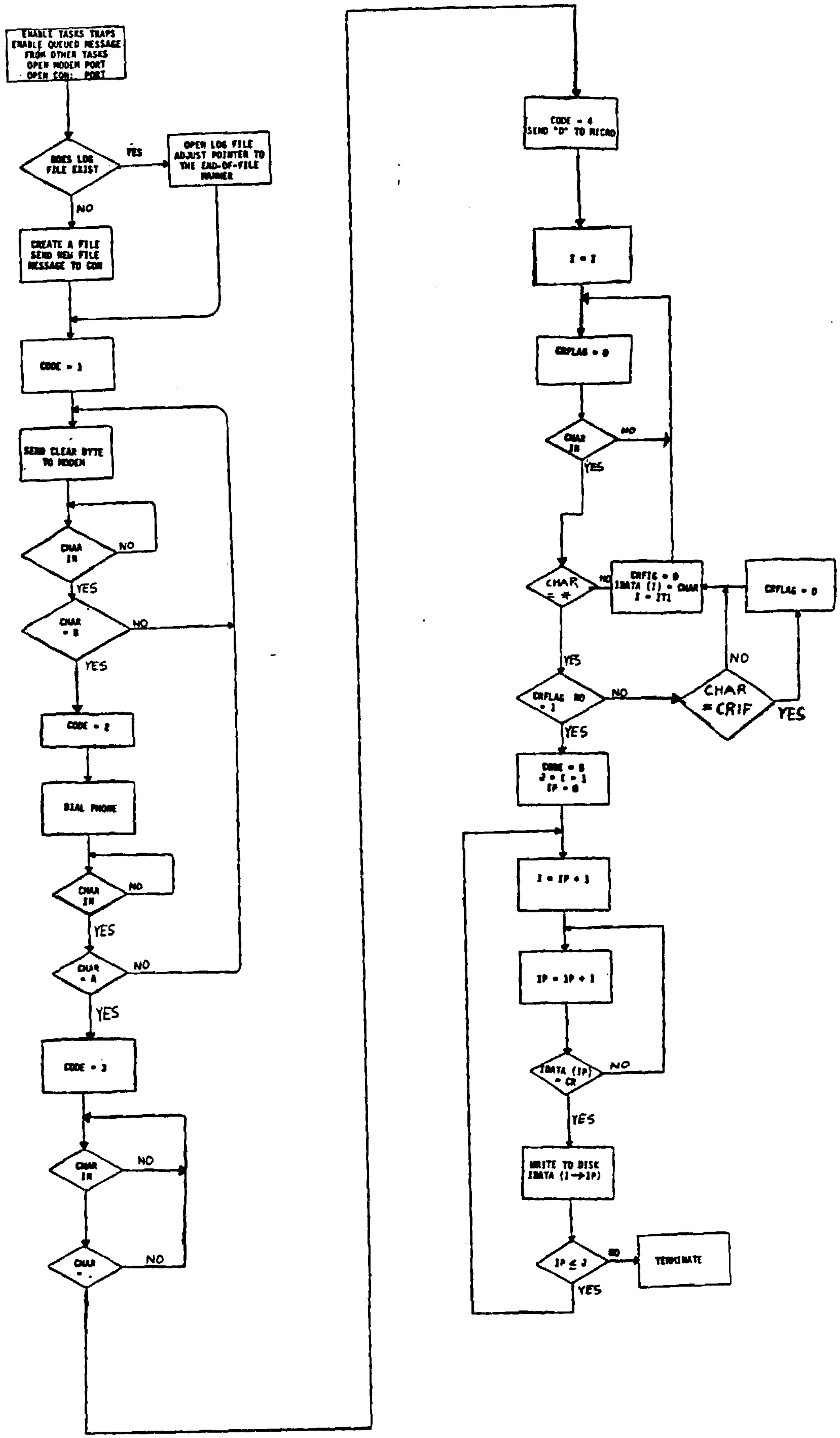
NO ERRORS:F7D R04-00 MAINPROG .MAIN 08/31/81 18:58:58 TABLE
STATEMENT BUFFER: 20 LINES/1321 BYTES STACK SPACE: 44 WORDS

TASK PHONE

Purpose:	To control the interaction process between a remote microprocessor and the Interdata system.
Support Modules:	FORTRAN Run Time Library Subroutine 'TERM'
Support Files:	Data File named 'A2891.DAT'
Task Requirements:	Task Communications Option
Start Sequence:	Execution automatically started by task 'START'
Termination Sequence:	Normal termination upon completion of data transfer, watch-dog termination upon expiration of timer in task 'START'. Premature termination possible by entering 'CA PHONE'.



GENERAL FLOW OF TASK 'PHONE'



DETAILED FLOW DIAGRAM OF TASK 'PHONE'

FORTRAN VITD: LICENSED RESTRICTED RIGHTS AS STATED IN LICENSE L-0174

```
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C    ----    PHONE.FTN V2.1    ----
                C
                C      THIS TASK IS STARTED PERIODICLY
                C      BY TASK 'START', AS PROCESSING
                C      ADVANCES THE PARAMETER 'CODE'
                C      IS INCREMENTED TO INDICATE THE
                C      CURRENT PHASE OF OPERATION.
                C
                C      NORMAL OPERATION IS TERMINATED
                C      WITH A CALL TO SUBROUTINE 'TERM'
                C      WHEN 'CODE' IS EQUAL TO FIVE (5)
                C
                C      ABNORMAL  OPERATION DUE TO THE
                C      WATCH-DOG TIME-OUT ( IN TASK
                C      START ) WILL CAUSE AN ERROR
                C      MESSAGE TO BE GENERATED.
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
      000000I          IMPLICIT INTEGER (A-Z)
      000006I          COMMON CODE
      000006I          EXTERNAL TERM
      000006I          CHARACTER IDATA(2300),CR,RESP
      000006I          DIMENSION IFILE(3),PBLK(10)
      000006I          DOUBLE PRECISION DIAL(2)
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C      DEFINE DATA VALUES FOR SYSIO
                C      FUNCION CODES, DATA FILE NAME,
                C      PHONE NUMBER, AND ASCII CR
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
      000006I          DATA IFILE /'A2891.DAT    '/
      000006I          DATA DIAL  /'<STX>3555<   '/
      000006I          DATA FCI    /X'4B'/
      000006I          DATA FCO    /X'2B'/
      000006I          ENCODE (CR,FMT='(A)') X'0D00'
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C      OPEN THE MODEM CHANNEL FOR
                C      COMMUNICATIONS WITH THE REMOTE
                C      MICROPROCESSOR, AND THE CONSOLE
                C      CHANNEL FOR ERROR AND STATUS
                C      LOGGING.
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
      000934I          CALL INIT
      00093CI          CALL ENABLE(2,TERM)
      0009B0I          CALL OPENW(2,'CRT3:',4,0,0,STAT)
      000A2CI          CALL OPENW(6,'CON:',4,0,0,STAT)
                C
```

FORTRAN VIID: LICENSED RESTRICTED RIGHTS AS STATED IN LICENSE L-0174

```
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C      TRY TO OPEN THE DATA LOG FILE.
                C
                C      IF A STATUS WORD OF ZERO IS
                C      RETURNED, THE FILE ALREADY
                C      EXISTS. IN THIS CASE
                C      THUMB THROUGH THE FILE TO FIND
                C      THE END-OF-FILE MARKER, NOTE
                C      THE LENGTH, CLOSE THE FILE,
                C      REOPEN THE FILE, AND POSITION
                C      THE SEQUENTIAL ACCESS POINTER
                C      IMMEDIATLY BEFORE THE EOF.
                C
                C      IF A NON-ZERO STATUS WORD IS
                C      RETURNED THEN THE FILE DOES
                C      NOT EXIST, A FILE IS THEN
                C      CREATED AND A MESSAGE TO THAT
                C      IS LOGGED ON THE OPERATORS
                C      CONSOLE.
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
   000AA8I             CODE = 0
   000AB0I             CALL OPENW(3,IFILE,4,0,0,STAT)
                C
   000AFCI             IF(STAT.EQ.0)GOTO 1
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C      CREATE A DATA LOGGING FILE, AND
                C      POST A MESSAGE TO THIS EFFECT
                C      ON THE SYSTEM OPERATOR CONSOL
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
   000B12I             WRITE(6,1010)
   000B2CI             CALL CFILW(IFILE,2,80,1,1,00,00,STAT)
   000B90I             CALL OPENW(3,IFILE,4,0,0,STAT)
   000BDCI             GOTO 11
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C      POSITION THE FILE POINTER TO OVER-
                C      WRITE THE EOF MARKER, THEREBY
                C      APPENDING THE NEW DATA TO THE END
                C      OF THIS EXISTING DATA LOGGING FILE.
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
   000BE2I      1      DO 2 I=1,1000
   000BEAI      2      READ(3,1006,END=3)
   000C1FI      3      CALL CLOSE(3,STAT)
   000C40I             CALL OPENW(3,IFILE,4,0,0,STAT)
   000C8CI             DO 4 J=1,I-1
   000CA2I      4      READ(3,1006)
```

FORTRAN VITD: LICENSED RESTRICTED RIGHTS AS STATED IN LICENSE L-0174

```
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C      THE INTERACTION WITH THE REMOTE
                C      MICROPROCESSOR PROCEEDES AS
                C      INDICATED BELOW WITH THE 'CODE'
                C      VALUES ASSOCIATED.
                C
                C      1 - CLEAR THE MODEMS AUTO DIAL
                C          CIRCUITS
                C
                C      2 - DIAL THE MICRO PHONE
                C
                C      3 - WAIT FOR THE MICRO PROMPT
                C          CHARACTER, '.'
                C
                C      4 - SEND THE MICRO A 'D' AND
                C          ACCEPT DATA FROM THE MICRO
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C
     000CD4I    11     CODE = 1
     000CDCI    5      CALL SYSIO(PBLK,FCO,2,'<SOH>',5,0)
     000D50I           CALL SYSIO(PBLK,FCI,2,RESP,1,0)
     000D9AI           IF(RESP.NE.'B') GOTO 5
                C
                C
                C
                C
     000DC6I           CODE = 2
     000DCFI           CALL SYSIO(PBLK,FCO,2,DIAL,10,0)
     000E10I           CALL SYSIO(PBLK,FCI,2,RESP,1,0)
     000E5AI           IF (RESP.NE.'A') GOTO 5
                C
                C
                C
                C
     000E86I           CODE = 3
     000E8FI    6      CALL SYSIO(PBLK,FCI,2,RESP,1,0)
     000ED4I           IF (RESP.NE.'.') GOTO 6
                C
                C
                C
                C
     000F02I           CODE = 4
     000F0AI           CALL SYSIO(PBLK,FCO,2,'D',1,0)
                C
     000F7CI           CRFLAG=0
     000F84I           DO 7 I=1,2300
     000F8CI           CALL SYSIO(PBLK,FCI,2,IDATA(I),1,0)
     000FECI           IF ((IDATA(I).EQ.'*') .AND.
                      1    ( CRFLAG  .EQ.  1)) GOTO 8
     001044I           CRFLAG = 0
     00104CI    7      IF (IDATA(I).EQ.CR) CRFLAG = 1
                C
                C
```

FORTRAN VIID: LICENSED RESTRICTED RIGHTS AS STATED IN LICENSE L-0174

```
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C      AFTER THE SUCCESSFUL TRANSFER
                C      OF DATA FROM THE MICRO TO LOCAL
                C      MEMORY, THE DESIRED ERROR DETECTION
                C      ALGORITHIMS CAN BE EXECUTED. THIS
                C      VERSION DOES NOT CHECK DATA
                C      INTEGRITY, IT ACCEPTS THE DATA THEN
                C      STORES IT ON DISK FOR SUBSEQUENT
                C      PROCESSING.
                C
                C      THE FOLLOWING CODE INITIALIZES
                C      POINTER 'I' TO THE BEGINING OF A
                C      DISPLAY LINE AND ADVANCES POINTER
                C      'IP' UNTIL THE END OF A DISPLAY
                C      LINE IS ENCOUNTERED. THAT LINE IS
                C      THEN OUTPUT TO THE DISK FILE, 'I'
                C      IS BUMPED UP TO THE POSITION 'IP'
                C      IS POINTING TO AND 'IP' IS ADVANCED
                C      AGAIN. THIS CONTINUES UNTIL THE
                C      ENTIRE DATA BLOCK HAS BEEN SAVED
                C      ON DISK.
                C
                C      AT THE COMPLETION OF THIS PROCESS
                C      SUBROUTINE 'TERM'INATE IS CALLED
                C      WITH A STATUS 'CODE' OF FIVE (5)
                C      INDICATING A NORMAL EXIT IS TO
                C      BE TAKEN.
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
     0010ACI    8      CALL CARCON(3,0)
     0010DAI           CODE = 5
     0010E0I           J=I-1
     0010EEI           IP=0
                C
     0010F4I    9      I=IP+1
                C
     001104I    10     IP=IP+1
     001112I           IF (IDATA(IP).NE.CR) GOTO 10
     001156I           IL=IP-I+1
     00116AI           CALL SYSIO(PBLK,FCO,3,IDATA(I),IL,STAT)
     00119AI           IF (IP.LT.J) GOTO 9
     0011CEI           CALL TERM
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
     0011D6I    1006   FORMAT(A4)
     0011E0I    1010   FORMAT(" ***** NEW FILE *****")
                C
                C
                C
                C
                C
     001200I           END
WARNING 5    9  **************************************************************
          >>> VARIABLE NOT INITIALIZED IN PROGRAM
```

```
FORTRAN-VIID R04-00

FORTRAN VIID: LICENSED RESTRICTED RIGHTS AS STATED IN LICENSE L-0174

               SYMBOL/LABEL =  IDATA

NO  ERRORS:F7D  R04-00      MAINPROG    .MAIN      08/31/81  19:03:23  TABLE
STATEMENT BUFFER:   20 LINES/1321 BYTES    STACK SPACE: 152 WORDS
```

SUBROUTINE TERM

Purpose:	To terminate execution of task 'PHONE'. If an abnormal termiation request occurs, a message to that effect is logged.
Support Modules:	FORTRAN Run Time Library
Support Files:	Logging file assigned to Logical Unit 3
Start Sequence:	Execution initiated by either task 'START' or 'PHONE'.
Termination Sequence:	N/A

```
FORTRAN-VIID R04-00

FORTRAN VIID: LICENSED RESTRICTED RIGHTS AS STATED IN LICENSE L-0174

                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
                C     ----     TERM.FTN V2.1     ----
                C
                C      THIS SUBROUTINE IS LINKED INTO TASK
                C      'PHONE', AND CONTROLS THE ORDERLY
                C      TERMINATION OF THAT TASK. IT MAY BE
                C      ENVOKED EITHER AS A RESULT OF A SUB-
                C      ROUTINE CALL FROM 'PHONE' OR BY A
                C      TASK TRAP FROM 'START'. THE ONLY
                C      WAY TO DETERMINE THE SOURCE IS TO
                C      EXAMINE THE VARIABLE 'CODE'. THE
                C      APPROPRIATE ACTION IS TAKEN ONCE THE
                C      SOURCE IS DETERMINIED.
                C
                CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                C
    000000I            SUBROUTINE TERM
    000004I            IMPLICIT INTEGER (A-Z)
    000004I            COMMON CODE
    000004I            DIMENSION TIME(2)
    000004I            CHARACTER MSG(4)*24
                C
    000004I            DATA MSG(1) /'  MODEM NOT POWERED UP   '/
    000004I            DATA MSG(2) /'    PHONE OUT OF ORDER   '/
    000004I            DATA MSG(3) /' MICRO DOES NOT ANSWER   '/
    000004I            DATA MSG(4) /'TIME OUT - NO EOF RCVED '/
                C
    000004I            IF (CODE.EQ.0.OR.CODE.GT.4) GOTO 110
                C
    00009AI            CALL ICLOCK(1,TIME)
                C
    0000BCI            CALL CARCON(3,0)
    0000E8I            WRITE(3,1000) MSG(CODE),TIME
                C
    000130I     110    ENDFILE 3
    000148I            CALL CLOSE(2,STAT)
    00016CI            CALL CLOSE(3,STAT)
    000190I            CALL EXIT(CODE)
                C
                C
                C
    0001A8I     1000   FORMAT (///,X,10('<'),X,24A,
                     1        X,2A4,X,10('>'),///)
                C
                C
                C
                C
    0001D2I            END

NO  ERRORS:F7D  R04-00      SUBROUTINE  TERM       08/31/81  19:00:13  TABLE
STATEMENT BUFFER:   20 LINES/1321 BYTES    STACK SPACE: 110 WORDS
```

COMMAND SUBSTITUTION SYSTEM (CSS) FILES

1. FORTRAN compilation.

2. General task link file used for task 'START'.

3. Task link file used for task 'PHONE'.

```
*
*   F7C.CSS - FORTRAN VII COMPILE
*
*   @1 = O OR D FOR OPTIMIZING OR DEVELOPMENT COMPILER(DEFAULT D)
*   @2 = SOURCE FILE
*   @3 = OBJECT OUTPUT (DEFAULT @2.OBJ)
*   @4 = LISTING FILE (DEFAULT PR:)
*   @5 = START OPTIONS
*   @6 = LOAD SIZE (DEFAULT 100K FOR O,20K FOR D)
*
$IFNN @1
F7@1 @2,@3,@4,@5,@6
$ENDC
$IFNU @1
F7D @2,@3,@4,@5,@6
$ENDC
$EXIT
```

```
LO .BG,LINK.TSK
TA .BG
$BUILD WALK.LCM
ESTABLISH TASK
OPTION FL,CON,COM,UN,DFL,NRES,PRIOR=(100,100),LU=50,WORK=(1000,20000)
INCLUDE @1.OBJ
LIBRARY F7RTL.OBJ/S
MAP ,ADDRESS
BUILD @1.TSK
END
$ENDB
ST,COMMAND=WALK.LCM,LOG=CON:
$EXIT
```

Georgia Institute of Technology
Engineering Experiment Station
Atlanta, Georgia 30332

ELECTRICAL POWER AND STEAM CONSUMPTION METERING TECHNIQUES

Technical Report For

Project A-2891-000

By:

Dr. James A. Mahaffey

Submitted to:

Naval Construction Battalion Center
Port Hueneme, California 93043

Delivery Order:

N68305-81-F-0026

Under:

BOA #F33657-80-G-0077

ABSTRACT

Methods for measuring energy usage and energy usage rates for electric and steam systems are surveyed, with emphasis given to sensors that can be economically and reliably interfaced to microprocessor-based facility management systems. For electricity demand measurements, a standard wattmeter equipped with a digital impulse generator is recommended. Of many possible steam flow measurement techniques, an insertion turbine meter is recommended, with appropriate temperature and pressure compensation sensors.

Table of Contents

List of Tables

List of Illustrations

1.0 Introduction

There are hundreds of known methods for measuring the energy transferred by electricity and steam, most of which are impractical or outmoded for microcomputer-based information systems. The purpose of this report is to present details of only the most workable metering techniques, as extracted from current technology. Criteria for the inclusion of a metering technique in this report are:

1. The energy meter system must produce an electrical signal, which is either digital in character or digitizable. This requirement excludes direct-reading gauges.
2. The metering system must be currently available on the domestic commercial market. This requirement excludes all archaic instruments, and many European and Japanese varieties, which may be incompatible with domestic power systems.
3. The metering system must have sufficient dynamic range to service the needs of an office building. This requirement excludes bench-top instrumentation, and fixed instruments intended for use only in single-family dwellings.

The metering techniques examined in this report are to be used in facilities management systems (FMS) composed of computer networks. Data from the electric and steam metering systems can be processed in real-time, to yield total energy consumed over a time period, instantaneous power demand, and average power demand over a time period. This information can then be used by an FMS to automatically optimize energy usage, and reduce peak demands.

Two types of electric meters are discussed, one analog and one digital. Although the digital meter is superior on several counts, the analog meter is included in the discussion as a comparison. Digital electric energy transducers, or "pulse initiators," are currently available as add-on units for existing, motor-driven watt meters. These units typically have a single moving part, operate under hermetic seal, and have expected operating lifetimes in excess of 15 years.

Steam metering using insertion turbine instruments is discussed. Although steam flow measurements using a turbine give digital data, the metering of steam energy is complicated by the need for temperature and pressure compensation. Temperature and pressure measurements are taken from separate analog instruments, and must be integrated into a resultant energy measurement using steam table hookups. Complete steam tables can be voluminous, requiring the entire memory capacity of a typical microcomputer architecture, but by taking into account the narrow range of states for heating-plant steam, a subset of the complete steam tables can be efficiently stored and accessed by a microprocessor node in the FMS.

2.0 Electric Metering

For billing purposes, electric utilities use the integrated quantities kilowatt-hours and kilowatt demand. The kilowatt is a unit of power, or the rate at which energy is used; kilowatt-hour is a unit of energy used; and demand is defined as the kilowatt load averaged over a specified interval of time. The interval over which demand is measured may be any selected period, but 15 and 30 minutes are power industry standards.

In alternating-current (AC) circuits, power in kilowatts is the product of the electromotive force (volts), the current (amperes), and the power factor (percent). The power factor is a correction for an AC power measurement, taking into account the reactance of an AC circuit, and the resultant phase difference between the current and electromotive force. The product of the voltage and the component of the current that is in phase with the voltage (actual current corrected by the power factor) is the active, or real power, expressed mathematically as

$$P_{REAL} = EI \cos \theta, \tag{1}$$

where E is electromotive force, I is current, and θ is the angle of phase lag or lead.

The simple product of volts times current gives the apparent power, or volt-amps. An apparent power measurement is consistently an overestimation of power load in an alternating-current circuit, by an indeterminant factor.

The difference between apparent and real power is a significant factor in the design of a computer-readable power meter. A method of building a non-intrusive digital power meter is to measure a circuit voltage with a digital voltmeter, giving a root-mean-squared (RMS) reading of the alternating voltage. The RMS current can be measured with a similar digital instrument using an inductive pickup, and the product of the two quantities is the apparent power. This elementary measurement suffers from the power factor error. To compensate for the power factor using strictly digital electronic means would be a more complex undertaking, involving digitizing the 60 cycle voltage and current waveforms, and calculating the phase angle from the zero-crossing difference between the two curves. Voltage, current and power curves are plotted in figure 1, for three special cases showing potential error conditions in apparent-power measurements: current in phase with voltage, current leading voltage by 30°, and current lagging voltage by 30°.

Industry-standard watthour-meters (or "wattmeters") are motor-driven, electromechanical devices connected to building power lines via current transformers. They are usually owned and maintained by power utilities. Power factor compensation is included in energy measurements taken from an electromechanical wattmeter, because the torque developed at the motor armature is always proportional to the product of the instantaneous values of current and voltage. Angular velocity of the armature is directly proportional to instananeous real power in the meter circuit. The power measurement sensed by the turning motor armature is usually integrated into an energy readout using a gear train, driving decade dials.

Potential sources of error in a wattmeter are changes in drag, or resistance, in the gear train, and thermal expansion of the aluminum-disk armature. These conditions are minimized by sealing the unit under a glass dome. Generally, electric energy metering devices are accurate to within $\pm 0.25\%$ total registered watthours per 10,000 hours.

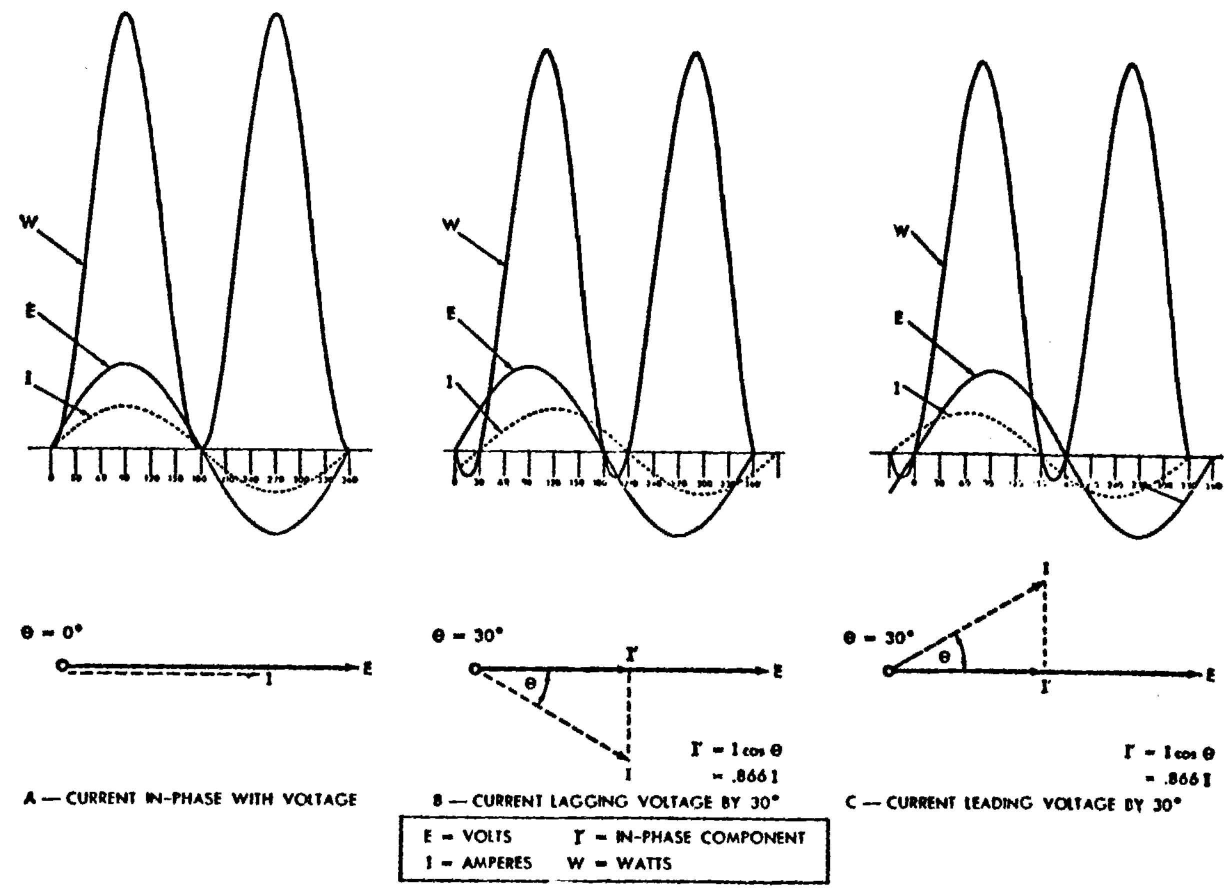


Figure 1. Voltage, Current, and Power Curves, for Three Special Cases

Industrial, commercial, and a growing number of domestic electric utility customers are billed on a demand basis. The reasons for demand-weighted charges instead of straight energy usage charges are straightforward. Utilities seek to recover capital investment in distributing equipment, and a consumer requiring large amounts of power over a short time period should therefore pay for the transformers necessary to deliver this energy surge. Many utilities set the demand billing rate for an entire year using a peak demand occurring during a single 15-minute demand averaging period. Minimizing such surges with automated energy management can therefore have great rewards.

Utilities install demand meters of the following types:

1. Up to 50 kw demand, indicating demand registers (analog indicator).
2. Up to 200 kw demand, graphic demand meters (digital indicator).
3. Above 200 kw demand, strip-chart or printing demand meters.

Although multi-stator watthour meters have been made to measure demand on multiple circuits or polyphase circuits, it is usually recommended that multi-circuit demands be totalized from separate meters on each circuit or phase.

A. Block Interval Meters

A widely-used method of demand metering is the block interval, so-called because a kilowatt hour measurement is totalized over a block of time, typically a 15-minute period.

The most flexible and easily computerized block interval metering technique may be impulse totalization. An impulse generator can be retrofitted to an existing electromechanical watthour meter. These devices cause relay contacts to close at a rate proportional to the load on the meter. Defined kilowatt demand is derived from relay closure pulses using the expression

$$\text{KW demand} = \frac{nC}{T} \qquad (2)$$

where n is the number of pulses occurring over the counting interval T (hours), and C is the constant number of watthours per pulse (demand multiplier), also designated KWC.

There are two types of pulse generators in common use: the contact device and the impulse device.

1. Contact Devices

The contact device is entirely mechanical, consisting of a multi-lobed cam, connected into the watthour meter gear train. The cam actuates one or more contact sets. All components of the device are mounted on a single backing plate, which bolts into the meter mechanism under the glass dome. There are three contact configurations available, as shown in figure 2. The variety of contact types services different electromechanical kilowatt demand totalizer configurations. Slow-break contacts, for example, are used for solenoid-operated totalizers, quick-break for motor operated units. If a reset signal is required by the totalizer after each kilowatt hour increment signal from the contact unit, then a three-wire, or SPDT, configuration is used. Two pulses are then generated for each kilowatt hour increment, but only one is counted. Table 1 shows the contact configuration currently available in demand meter systems from General Electric. The table is taken from the General Electric document GET-3048C,

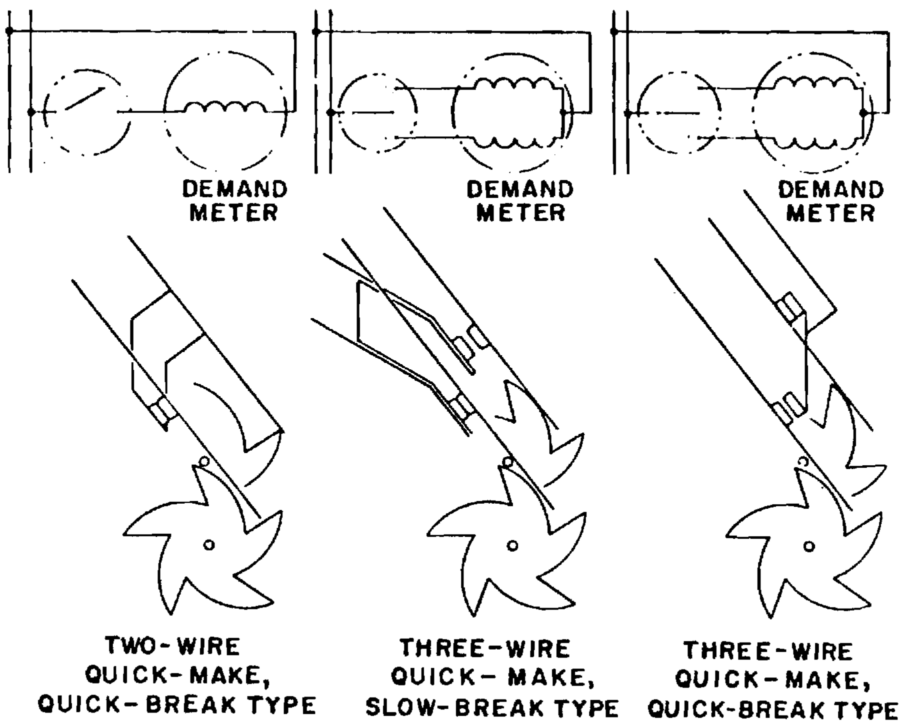


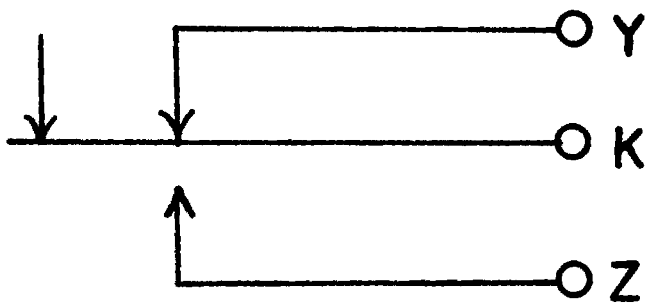
Figure 2. Contact Device Configurations

Device Type	Description	Contacts	Bridging	Contact Bounce	Contact Type
D-5 D-30-I D-30-V	3-Wire QM, QB, Make-Before-Break	SPDT Form D	Yes	Yes	Hard
D-13 D-30-IS D-30-VS	3-Wire SB, QM, Break-Before-Make	SPDT Form D (1)	No (2)	Yes	Hard
D-41	3-Wire QB, QM, Relay Output	SPDT Form C	No	Yes	Hard
D-51	3-Wire QB, QM, Mercury Wetted Relay	SPDT Form C	No	No	Mercury Wetted
S-6	3-Wire QB, QM, Break-Before-Make	SPDT Form C	No	Yes	Hard
S-3	3-Wire QB, QM	SPDT Form C or D (3)	(3)	Yes	Hard
MR-3	3-Wire QB, QM, Break-Before-Make	SPDT Form C	No	No	Mercury Wetted

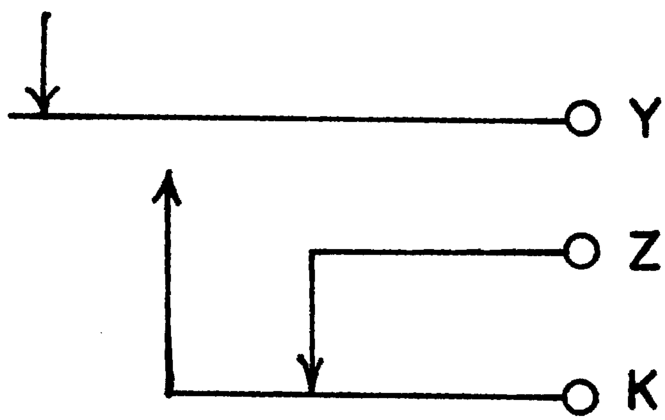
Notes:
1. It is possible for both contacts to remain open if the meter should stop during transfer time.
2. If not properly adjusted, contacts could be Form D bridging.
3. The S-3 relay follows the contact device from which it is operated (Form C or D).

QM = Quick-Make, QB = Quick-Break, SB = Slow Break

Table 1a. General Electric Contact Devices



FORM C



FORM D

Table 1b. Contact Type Definitions

"How to Select Contact Devices and Impulse Generators."

The contact closures from a pulse generator can be interfaced to a microcomputer through a single digital interface channel. The central pole of the contact switch, or its equivalent in a repeater relay, can be grounded with respect to the microcomputer, with a stationary pole connected to a logical "1" state through a current-limiting resistor. Figure 3 is a schematic of this interface, which is the most elementary of computer interfaces.

Figure 3 shows no connection to the Z contact on the cam-operated switch. Normally, this contact carries the indicating pulse, while the Y contact carries a reset pulse for a solenoid-operated counter. In this case, either contact can be used in redundant, cross-checking circuits.

Only one more bit of information in addition to the pulse data is needed to measure a wattmeter load using a contact device with a microcomputer: the demand multiplier (KWC). The value of the demand multiplier depends upon the type of contact device used and its position in the meter gear train. Most contact devices are driven by a separate worm on the meter shaft, and there are 360 possible gear ratios using available worm and sector combinations on GE meters. Tables for Equivalent Cam Teeth (E_QT), given the number of points on a cam and the gear ratio from the meter rotor to the cam, are given in Appendix A. Necessary formulae for computing KWC are listed in table 2.

The disadvantages of contact devices have limited their installation in the newer, more sensitive watthour meters. The newer meters produce lower torque per kilowatt, and are thus affected by the mechanical friction introducted by the cam-operated switch. There is also a need for a greater pulse rate, which involves using shallow cam lobes and lighter switch

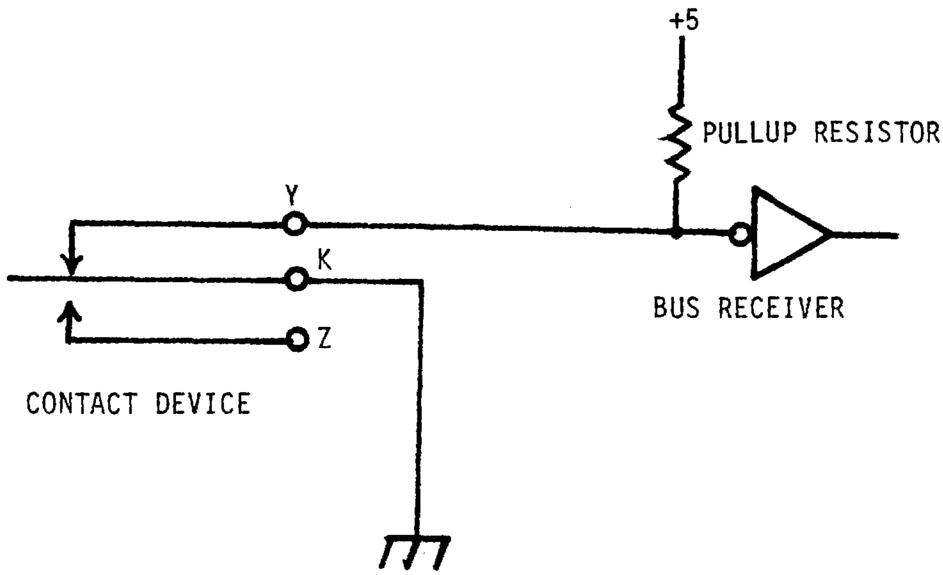


Figure 3. Sense-Line Interface

Condition	Contact Devices D-5, D-12, D-13	Contact Devices D-20, D-30	Impulse Generators D-41, D-51
	Formulae to Calculate E_QT, C/R, and R/I		
1. 3-wire contact device, SPDT	$E_QT = \frac{1}{2} \times \frac{PK_h \times \text{teeth WW}}{1{,}000 \times T \times KWC}$	$C/R = \frac{PK_h}{1{,}000 \times T \times KWC}$	$\frac{R}{I} = \frac{1{,}000 \times T \times KWC}{PK_h}$
2. 3-wire contact device, used as a SPST	$E_QT = \frac{PK_h \times \text{teeth WW}}{1{,}000 \times T \times KWC}$	$C/R = \frac{PK_h \times 2}{1{,}000 \times T \times KWC}$	$\frac{R}{I} = \frac{1{,}000 \times T \times KWC}{PK_h \times 2}$
3. 2-wire contact SPST	$E_QT = \frac{PK_h \times \text{teeth WW}}{1{,}000 \times T \times KWC}$	$C/R = \frac{PK_h}{1{,}000 \times T \times KWC}$	$\frac{R}{I} = \frac{1{,}000 \times T \times KWC}{PK_h \times 2}$
	Formulae to Calculate KWC		
1. 3-wire contact device, SPDT	$KWC = \frac{1}{2} \times \frac{PK_h \times \text{teeth WW}}{1{,}000 \times T \times E_QT}$	$KWC = \frac{PK_h}{1{,}000 \times T \times C/R}$	$KWC = \frac{R}{I} \times \frac{PK_h}{1{,}000 \times T}$
2. 3-wire contact device, used as a SPST	$KWC = \frac{PK_h \times \text{teeth WW}}{1{,}000 \times T \times E_QT}$	$KWC = \frac{PK_h \times 2}{1{,}000 \times T \times C/R}$	$KWC = \frac{R}{I} \times \frac{PK_h \times 2}{1{,}000 \times T}$
3. 2-wire contact SPST	$KWC = \frac{PK_h \times \text{teeth WW}}{1{,}000 \times T \times E_QT}$	$KWC = \frac{PK_h}{1{,}000 \times T \times C/R}$	$KWC = \frac{R}{I} \times \frac{PK_h \times 2}{1{,}000 \times T}$

Table 2. Application Formulae, from GET-3048C

E_QT = Equivalent (cam) teeth

= cam points x worm pitch x $\frac{\text{Rev cam}}{\text{Rev WW}}$

WW = worm wheel

C/R = Contacts per revolution of watthour meter disk

R/I = Revolutions of disk per impulse

= $\frac{\text{watthours per impulse}}{\text{watthours per disk revolution}}$

K_h = Watthour constant for self-contained meters.
For transformer-rated meters,
PK_h = K_h (secondary) x CT ratio x PT ratio.

CT = current temperature

PT = power temperature

teeth WW = teeth on worm wheel (100 for all meters except V-2, -3, -6 and -8, which have 50)

T = demand time interval in hours
(0.25, 0.50, or 1.00)

KWC = kilowatt constant, or demand multiplier

Legend for Table 2.

contact springs. These modifications have proven to increase the switch failure rate, and the associated high maintenance problems. Maximum pulse rates for GE contact devices are listed in table 3.

From table 2a is it noted that non-mercury-wetted switch contacts experience bounce. High-speed logic circuits are sensitive to contact bounce, being able to resolve a train of pulses from a single rebounding switch closure. This problem can be handled in the software of the microprocessor analyzing the pulse data, by discarding any pulse data occurring during a specified fraction of a second after an initial pulse is received.

2. Impulse Devices

The disadvantages of contact devices are absent in the current impulse generator designs. An impulse generator consists of a revolving shutter disk operated by the standard watthour meter geartrain. The shutter disk interrupts a beam of light, activating a photocell. Figure 4 shows the basic shutter configuration, and the placement of photocell and light source. The device is essentially frictionless, and it gives the same switch-output as a contact device, through a mercury-wetted relay. The bounceless contact-closure data from the impulse device is processed using the same techniques used for the contact device. The impulse devices are, in fact, designed to interface to older pulse totalizing equipment built for use with contact devices. The impulse device modules can be retrofitted to most watthour meter families.

Common impulse device types built by GE are the D-41, D-51, and D-52. Specifications for these representative units are given in table 4. Note

Demand Meters	Suggested Contact Devices* With Meters Having Ball or Pivot Bearings	With Meters Having Magnetic Suspension	Max. No. of Impulses Per Interval Demand Meter Can Record	Every Other Impulse Counts	Every Impulse Counts
BR-1, -2	D-5	D-30, D-41	200		X
G-9 Form W	D-5, D-13	D-30, D-41, D-51	100	X	
G-9 Form W-2	D-5, D-13	D-30, D-41, D-51	200	X	
Form SD-2	D-12	D-20	200		X
GM-10 Form W-1	D-5, D-13	D-30, D-41, D-51	100	X	
GM-10 Form W-2	D-5, D-13	D-30, D-41, D-51	200	X	
Form SD-2	D-12	D-20	200		X
MD-3, -5 Form M-30, Form M-51	D-5	D-30, D-41, D-51	CL. 2 Scale: 200 (15 min)		X
			400 (30 min)		X
			800 (60 min)		X
Form R	D-5	D-30, D-41, D-51	–		X
PD-5, -7	D-5, D-13	D-30, D-41, D-51	300		X
PD-55-F, -57-F	D-5, D-13	D-30, D-41, D-43, D-51	999		X
Totalizers					
MD-3, -5	D-5, D-30	D-30, D-41 D-51	1 per 1.5 sec		X
DT-3	D-5, D-30	D-30, D-41 D-51	1 per 1.5 sec		X
SST-1	D-5, D-30	D-30, D-41, D-42, D-51	3 per sec		X
SST-2	D-5, D-30	D-30, D-41, D-51	3 per sec		X

* Types DS-19 to D3-44 watthour meters, which use Types D-5, D-12 and D-13 contact devices, can be provided with conversion kits, permitting the use of Types D-20, -30, -41, and -51 devices. Types V-4-A, V-7-A, V-10-A, V-16-A, VS-3 and VS-4 meters also can utilize conversion kits, permitting the use of Types D-20, -30 and -41 devices.

Table 3. Maximum impulse rates for contact devices retrofitted to watthour meters, types DS-19 to D3-44, from GE document GET-3048C

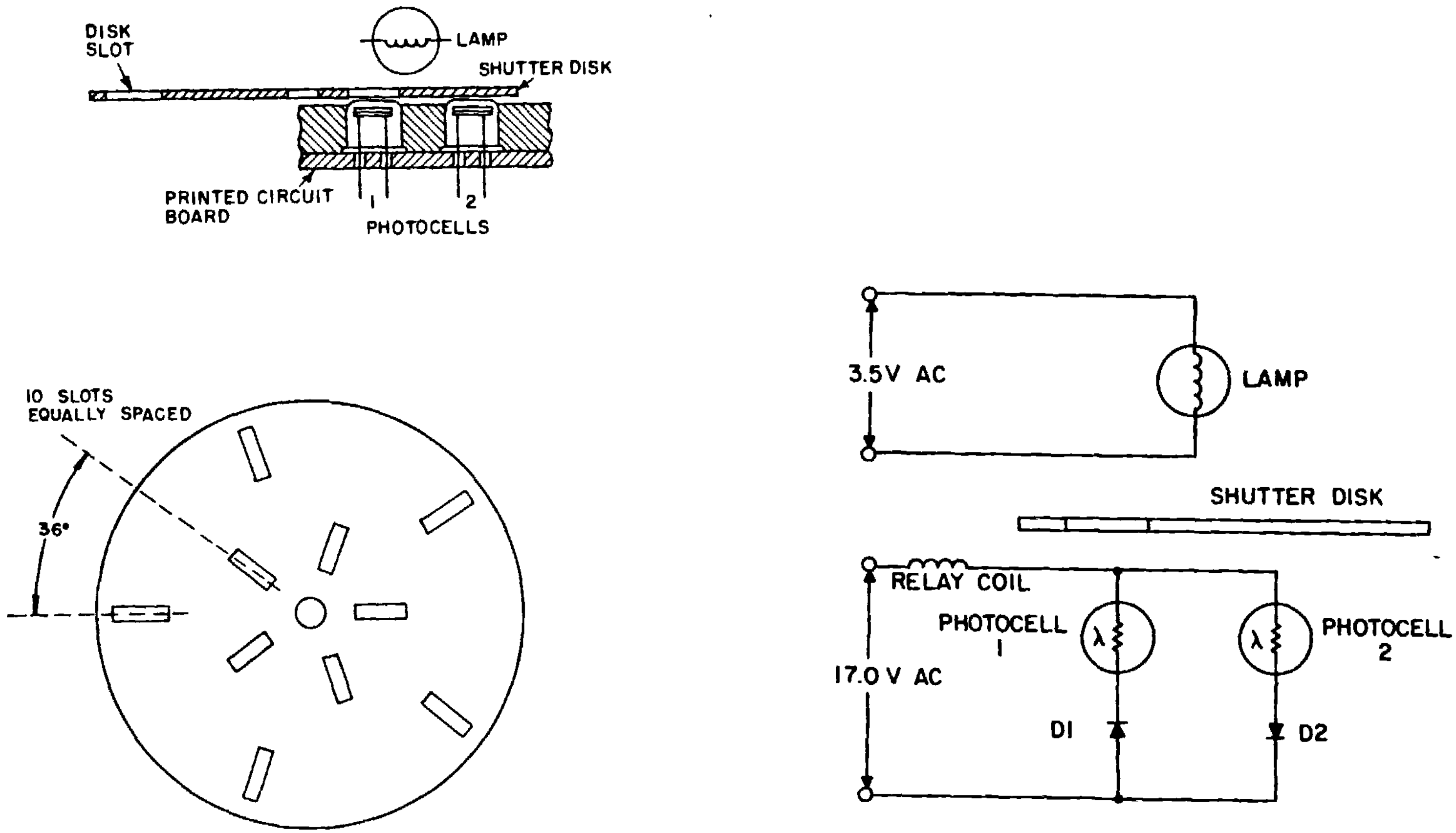


Figure 4. Impulse Device Shutter Configuration

	D-41	D-51	D-52
Meters used on	VW-60 DSW-50 DSW-60	V-60 DS-50 DS-60	VM-60 DSM-50 DSM-60
Allowable operating voltage deviation (at 120 volts)	+/-8%	+/-18.3%	+/-18.3%
Operating temp. range	$-4^{o}F$ to $+150^{o}F$	$-22^{o}F$ to $+150^{o}F$	$-22^{o}F$ to $+150^{o}F$
Maximum pulse rate	3/sec	4/sec	4/sec
Input power burden	3 VA	2 VA	2 VA
Expected lamp life	10 years	15 years	15 years
Output relay type	Mercury Wetted	Mercury Wetted	Mercury Wetted
Expected relay life	10^9 operations	10^9 operations	10^9 operations
Contact type*	3-wire MBB	3-wire MBB	3-wire MBB

MBB = make-before-break

*Form C, Table 1b

Table 4. Specifications for the D-41, D-51, and D-52 Impulse Devices.

that the maximum pulse rates given in this table are at least twice, and possibly three to four times the maximum pulse rate used by electric utilities for demand billing or power usage surveys. The meter can be pushed to higher pulse rates, but the extended maximum becomes temperature and voltage dependent. The extended maxima for D-51 and D-52 impulse devices are tabulated in table 5.

The output relay for the D-41 device is sealed in a box, mounted outboard to the meter and providing a Form C (table 1b) contact configuration through an industrial screw-connector socket. The D-52 and D-51 devices use a miniaturized relay, contained inside the meter.

As in the case of the contact devices, there are packaged formulae for calculating the number of pulses per kilowatt hour. Basically, the kilowatt constant is calculated using the formula

$$KWC = \frac{\text{estimated power load (kilowatts)}}{\text{desired no. pulses per demand interval}}, \tag{3}$$

which gives an approximate value for KWC to be used in the selection of an impulse device and associated gear train. The number of revolutions of the meter rotor per pulse are then calculated using the formula

$$\frac{R}{I} = \frac{1000 \times T \times KWC}{PK_h}, \tag{4}$$

where R = meter rotor revolutions,

I = pulses from the output relay,

T = demand interval, in hours,

PK_h = primary watthours per revolution of the meter rotor, and

	98 Volts			120 Volts			132 Volts		
Meter Ambient Temperature	-22	+72	+150	-22	+72	+150	-22	+72	+150
Pulses/sec	7	7	7	7	13	13	4	14	14

Table 5. Extended Maximum Pulse Rating for the GE D-51 and D-52 Impulse Devices.

KWC = demand multiplier, a kilowatt constant.

A typical impulse device type has one shutter disk, with ten slots, producing one pulse per slot. Shutters with two, four, six, or eight slots are available.

New demand meter installations are likely to involve the current state-of-the-art impulse devices, of which the GE D-72 is an example. This device is a modification of the D-52 type, described in table 4. Differences between the D-52 and D-72 devices are:

(1) The D-52 illumines the photocells with a single incandescent lamp. The D-72 uses two, parallel light-emitting diodes (LEDs).

(2) The D-72 has a 39 ohm current-limiting resistor connected in series with the illuminator circuit.

Table 6 lists GE documents useful in the design of a digital power demand system.

3. Remote Interfacing

For applications requiring a distance between the block interval meter and the FMS microprocessor of over 50 feet, a voltage higher than the logical +5 volts must be used at the relay contacts. When a higher voltage is used, the contacts must be protected from arcing with an RC network.

A resistor and capacitor, in series, are connected across the relay contacts. Suggested values are calculated using the equations,

$$R = \frac{E}{10I^{(1 + 50/E)}}, \text{ and} \tag{5}$$

$$C = \frac{I^2}{10} \text{ mfd}, \tag{6}$$

1. GET-3048C, "How to Select Contact Devices and Impulse Generators"
2. GET-2327C, "Manual of Demand Meters"
3. GEH-2767C, "Type D-41 Impulse Generator for Demand Metering"
4. GEH-2781B, "Type D-51 Pulse Initiator for Demand Metering"
5. GEH-2786-1, "Type D-72 Pulse Initiator for Demand Metering"
6. GEH-2786, "Type D-52 Pulse Generator for Demand Metering"
7. GEH-2251, "Demand Meters, Types G-9, GS-9, and GS-12"
8. GEH-2764F, "Types PD-55-F and PD-57-F Printing Demand Meters"
9. GEH-224, "Instructions for Types D-5 and D-13 Contact Devices"
10. GEH-2754, "Instructions for Type D-30 Contact Device"
11. GEH-2782, "Instruction for Types C-14 and C-16 Contact-Making Clocks"
12. GEH-1050, "MD Totalzers"
13. GET-813, "How to Test and Adjust GE AC Watthour Meters"
14. GET-1840, "Manual of Watthour Meters"
15. GET-2669, "Guide for Installing GE Watthour Meters"
16. GET-1905, "Application of Watthour Meters"
17. GET-3068, "Demand Metering, Local and Remote"

Table 6. GE Power Demand Documents.

where R(min) = 0.5 ohm,

C(min) = 0.001 mfd,

I = steady-state current at time of circuit opening,

and E = open-circuit voltage.

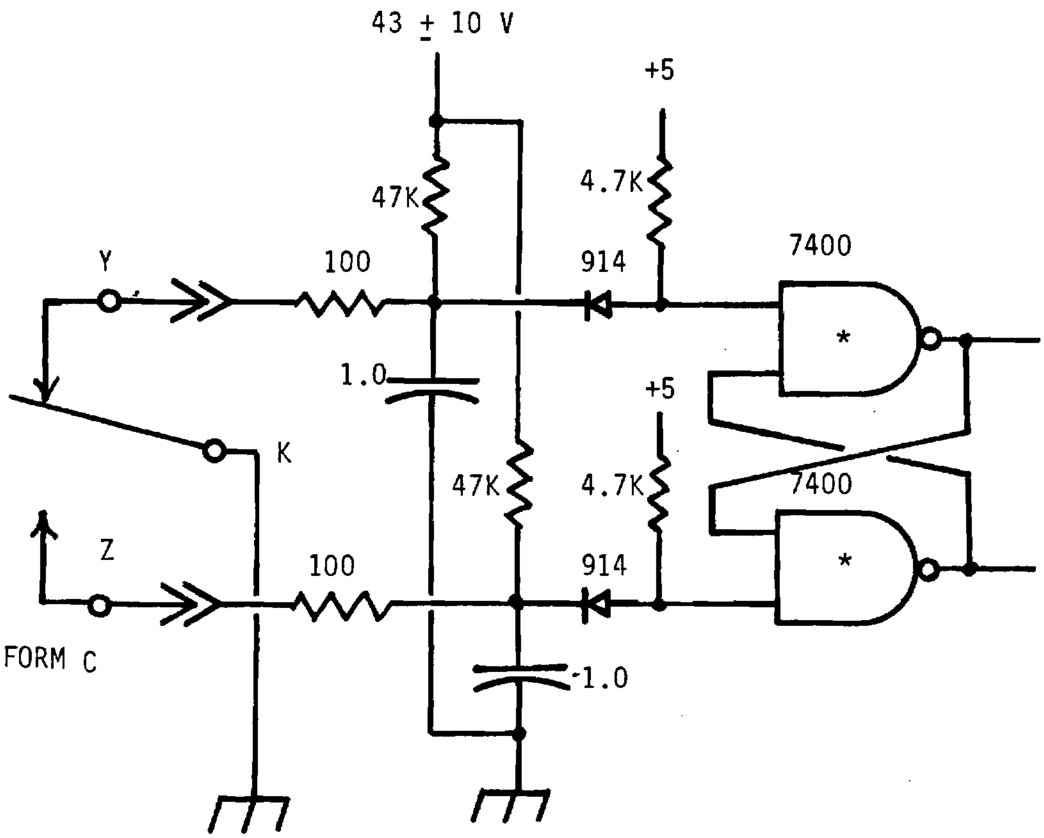
The interface circuit in figure 3 is shown in figure 5 with the RC modification for 50 volts on the contacts, plus overvoltage protection for the receiving integrated circuits.

B. Analog Watt Transducers

Totally electric transducers are available to provide a DC current output proportional to an AC power input. Voltage and current in a power line are sampled separately, and compensation for the varying phase difference between these two quantities is provided. Watt transducers also have temperature drift compensation.

Interface to the FMS microprocessor can be provided by an analog-to-digital converter (ADC), with a power-utility-compatible demand reading taken by sampling the wattage output from the transducer at a nominal interval (once per second), and averaging over a 15-minute period. The analog output from the transducer can be voltage or current, depending upon the distance over which the data are to be accessed.

The large number of watt transducer manufacturers in the U.S. (39) is shown in table 7. Most units are physically similar, outline dimensions for a typical unit are shown in figure 6. The common watt transducer is usually enclosed in a steel box, with input/output connections on terminal strips on the top, and calibration through small (0.125 inches) access ports. The largest example found in this investigation, a Rochester



* interface to microcomputer port

Figure 5. Remote Interface

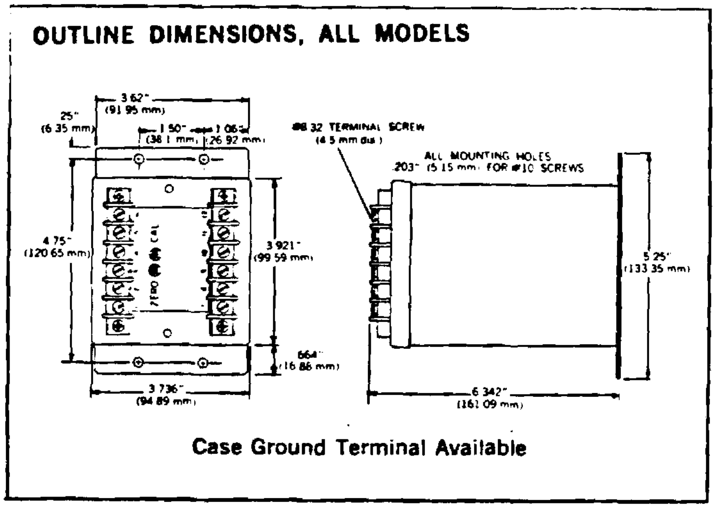


Figure 6. Watt Transducer Enclosure

1. American Aerospace Controls, Inc., 570 Smith Street, Farmingdale, New York 11735, (516) 694-5100

2. Hi-G, Inc., Electronic Products, 580 Spring Street, Windsor Locks, Connecticut 06096, (203) 623-2481

3. North Hills Electrics, Inc., Alexander Place, Glen Cove, New York 11542, (516) 671-5700

4. Ohio Semitronics, Inc., 1205 Chesapeake Avenue, Columbus, Ohio 43212, (614) 486-9561

5. Kratos, Inc., Industrial Products, 403 South Raymond Avenue, Pasadena, California 91109, (213) 449-3090

6. Action Instruments Company, Inc., 8601 Aero Drive, San Diego, California 92123, (714) 279-5726

7. Active Control Instrumentation, P.O. Box 243, Key Largo, Florida 33037, (305) 451-1000

8. AEMC Corporation, North American Distributor for Chauvin Arnoux, 729G Boylston Street, Boston, Massachusetts 02116, (617) 266-8506

9. Allegany Technology, Inc., 143 Offutt Street, Cumberland, Maryland 21502, (301) 722-7330

10. American Laser Systems, Inc., 106 James Fowler Road, Goleta, California 93017, (805) 967-0423

11. Analog Precision, Inc., 1620 North Park Avenue, Tuscon, Arizona 85719, (602) 622-1344

12. Ark Electric Products, Inc., PMD, 325 West Hibiscus Boulevard, Melbourne, Florida 32901, (305) 724-5260

13. F. W. Bell, Inc., 6120 Hanging Mass Road, Orlando, Florida 32807, (305) 678-6900

14. Bourns Instruments, Inc., 6135 Magnolia Avenue, Riverside, California 92506, (714) 781-5148

15. Bristol Babcock Division, Acco Industries, Inc., 40 Bristol Street, Waterburg, Connecticut 06708, (203) 575-3000

16. Caltemp Instruments, Inc., 2275 Meyers Avenue, Escondido, California 92025, (714) 743-2000

Table 7. Domestic Manufacturers and Distributors of Watt Transducers

17. Computer Instruments Corp., Columbia Components, 100 Madison Avenue, Hempstead, New York 11550, (516) 483-8200

18. Electrical Instruments Service, Inc., Greiback Instruments Division, 25 Dock Street, Mount Vernon, New York 10550, (914) 699-9717

19. Electric Wattmeter Co., 2275 6th Street, Sarasota, Florida 33577, (813) 953-5355

20. Fenwal Electrics, 63 Fountain, Framingham, Massachusetts 01701, (617) 872-8841

21. Fisher & Porter Co., 48 Warminster Road, Warminster, Pennsylvania 18974, (215) 674-6000

22. General Electric Company, Industrial Products, 40 Federal Street, Lynn, Massachusetts 01910, (617) 594-0100

23. Kimberly James, Inc., Box D7 Reese Avenue, Newtown Square, Pennsylvania 19073, (215) 353-2828

24. H. Levinson Co., Bert Electric Division, 601 Amora North, Seattle, Washington 98109, (206) 285-3622

25. Magnetic Labs, Inc., Power Concepts, Route 81 Exit 68, Hallstead, Pennsylvania 18822, (717) 879-2114

26. New England Instrument Co., Inc., Kendall Lane, Natwick, Massachusetts 01760, (617) 873-9711

27. Parko Electric Co., Inc., 16722 Milliken Avenue, Irvine, California 92714, (714) 549-8301

28. PET, Inc., 227 Allegheny Avenue, Oakmont, Pennsylvania 15139

29. Rochester Instruments Systems, Inc., 255 North Union Street, Rochester, New York 14605, (716) 263-7700

30. Ronan Engineering Co., 21200 Oxnard Street, Woodland Hills, California 91367, (213) 883-5211

31. Sangamo Weston, Inc., Weston Instruments, 614 Frelinghuysen, Newark, New Jersey 07114, (201) 242-2600

32. Scientific Columbus, Inc., Esterline Corporation, 1900 Arlingate Lane, Columbus, Ohio 43228, (614) 274-7160

33. Sanareline Controls, 525 Boston Post Road, Milford, Connecticut 06460, (203) 878-4619

Table 7 Continued

34. Sunshine Scientific, Inc., 1810 Grant Avenue, Philadelphia, Pennsylvania 19115, (215) 673-5600

35. Trans Data, Inc., P.O. Box 518, Dublin, Ohio 43017, (614) 889-0350

36. Vacuum Products, Inc., 25 Dock Street, Mount Vernon, New York 10550, (914) 664-7330

37. Vernitron Corporation, Vernitech Division, 300 Marcus Boulevard, Deer Park, New York 11729, (516) 586-5100

38. Westinghouse Electric Corporation, 95 Orange Street, Newark, New Jersey 07101, (201) 465-2302

39. Yokozawa Corporation of America, 2 Dart Road, Shenandoah, Georgia 30205, (404) 253-7000

Table 7 Continued

Instrument Systems PCE-30, weighs less than four pounds and can continuously sample a 4.2 kilowatt power line.

There are two types of analog transducers to consider, distinguished by the mode of interface to the power lines - induction-coil pickups and Hall-effect pickups. The signals from these two types of sensors are conditioned in similar ways, with conditioning features dependent on the manufacturer and the application of the transducer.

1. Induction-Coil Pickup

Major functional blocks of an induction-coil watt transducer are diagrammed in figure 7. The potential, or voltage, sense is picked from power transmission lines by the potential transformers (PT), connected in parallel with the power load circuit. Operating power (less than 0.5 watts) is also taken from this circuit. The current is sensed by the magnetic field it produces around a current-carrying conductor in the power load circuit. There are basically two sensor types used for direct inductive pickup - a solenoid that encircles the current carrier, through which the conductor must be threaded, and a split ferro-magnetic core used to impress the primary magnetic field on a secondary transformer winding. The advantage to the split core is that it can be easily retrofitted to an existing conductor. The core comes apart, and is reassembled around the conductor. The current and potential are multiplied together by an analog circuit. The output of the multiplier is a DC signal, proportional to the real power, plus a ripple voltage whose amplitude is proportional to the apparent power component. This component is stripped off by a low-pass filter, and the resultant DC signal is conditioned for transmission to an ADC by a final amplifier stage. Figure 8 is a schematic of a representative watt transducer, by Rochester Instrument Systems, Inc.

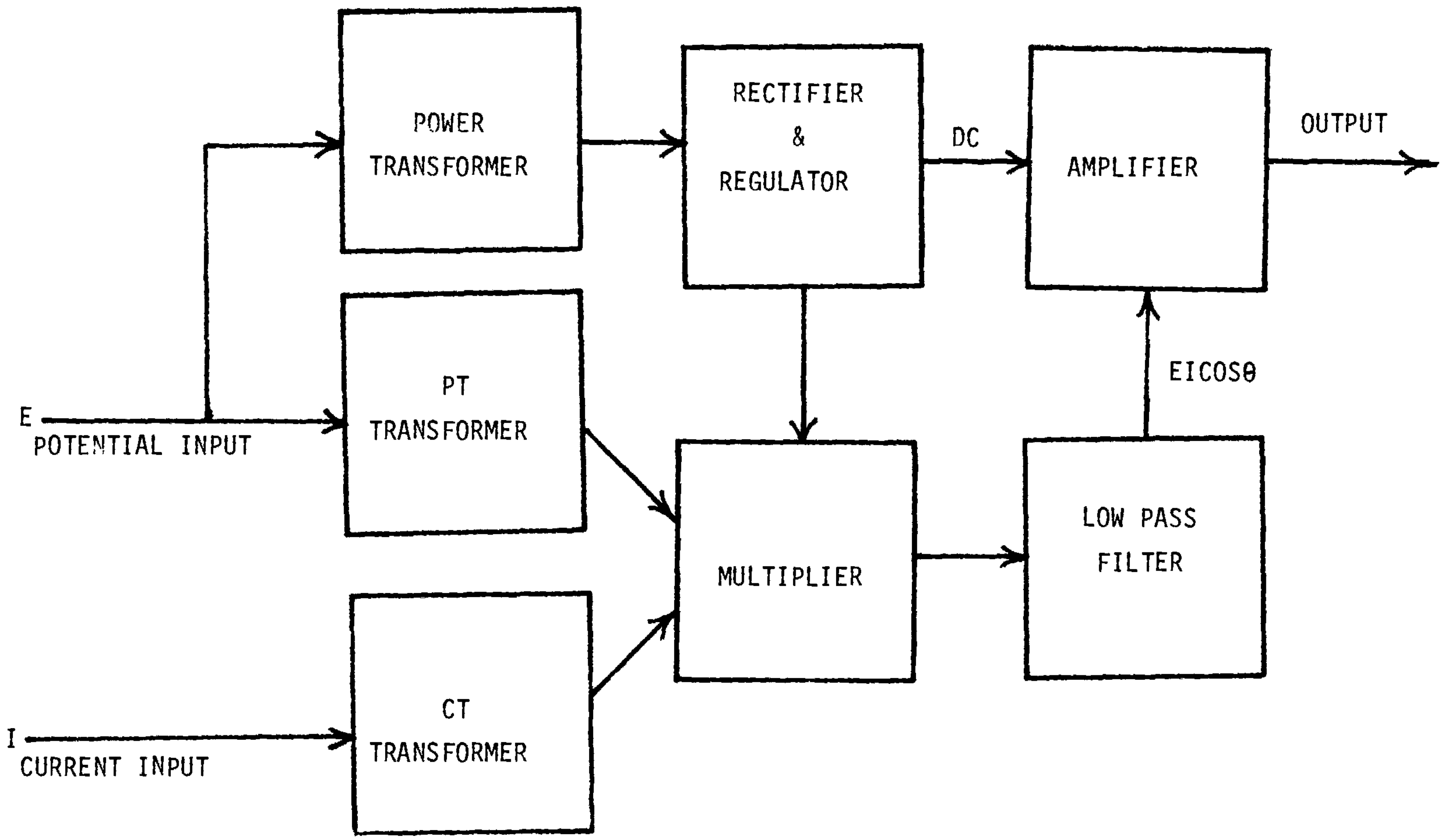


Figure 7. Watt Transducer, Functional Block Diagram

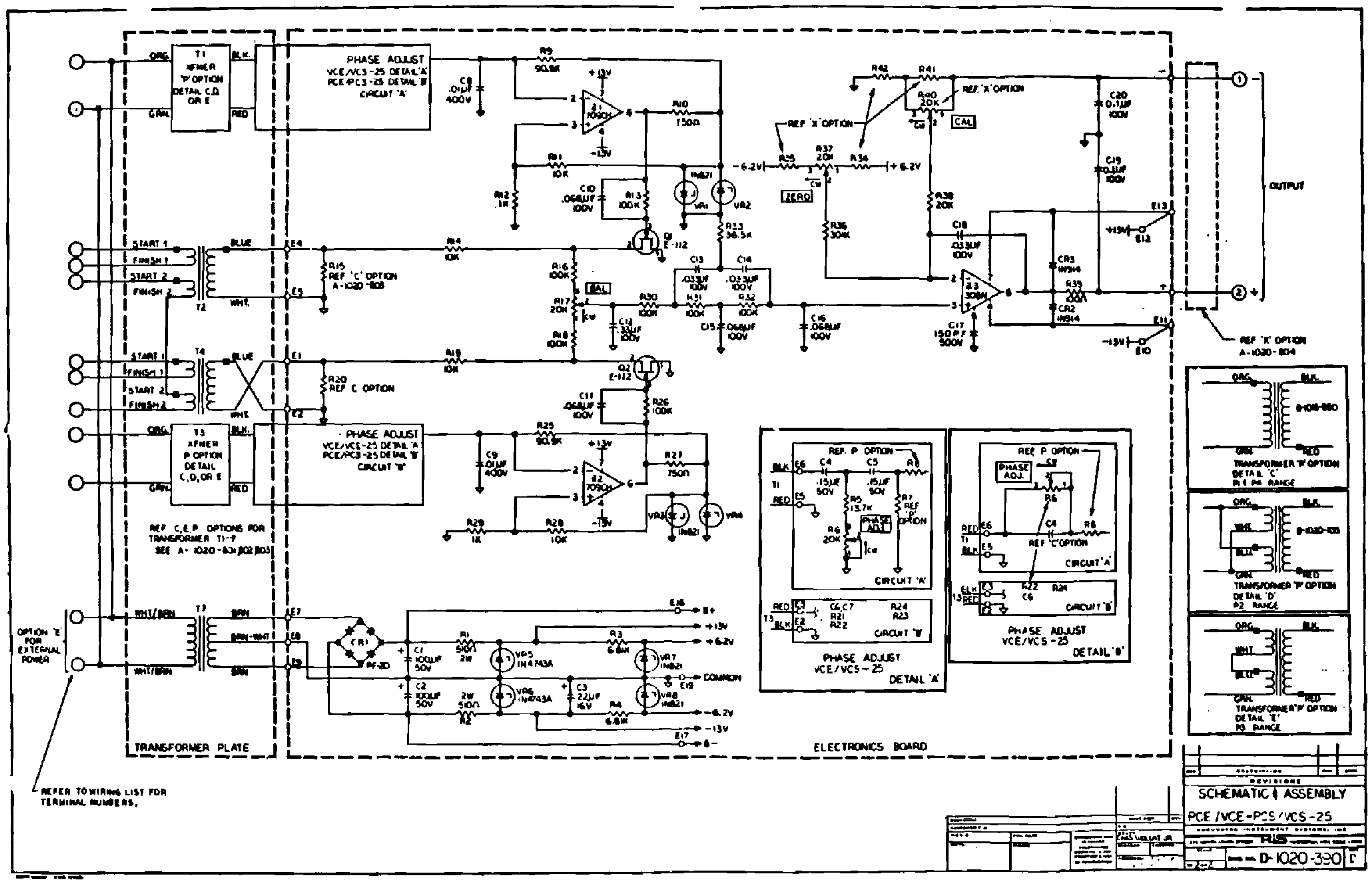


Figure 8. RIS Watt Transducer Schematic

2. Hall-Effect Pickup

Hall-effect devices are an alternative to direct-inductive current sensing. The hall sensor is a semiconductor device, typically made of indium arsenide, indium antimonide, or silicon. A current is passed through the semiconducting crystal. When the device is placed in a magnetic field, a potential difference is generated between the two opposed edges of the semiconductor in the direction mutually perpendicular to both the field and the crystal. The magnitude of this potential difference is proportional to the magnitude of the magnetic field through the device.

Linearity, noise characteristics, and reliability of Hall sensors are good, but they suffer from all of the disadvantages of semiconductor devices used in heavy industrial applications. Operating high-temperature range is typically limited to 85^{o}C, and the temperature coefficient ranges from -0.005 to -0.25% $V_H/^{o}C$, where V_H is the Hall potential. The temperature coefficient can be compensated for using a thermistor temperature sensor, but it is interesting to note that many power meters are advertised as having "no Hall-effect devices; no temperature problem."

Hall-effect current probes are often used to detect over-current conditions in electrical circuits, as a simple threshhold device.

The disadvantages of the analog transducer are common to most analog instrumentation: to send the signal over a significant distance requires a very high-quality transmission line, free of ground-loop problems. Digitization should occur as close to the transducer as possible.

C. Summary

Load-sensing equipment instaled by a power utility in a building is usually a chart-recording device located at the sense point, a central bank of chart recorders fed by remote sensors in the same building, or local sensors feeding a remote location for mechanical or computer tabulation. The sensors provide data in the form of a switch closure at a specific interval of power consumption, which ca be used to calculate peak demand.

The utility-provided sensor is usually an accessory device attached to the watt-hour totalizing meter used to measure integral power consumption. The type of switch-closure scheme depends basically on the age of the installation. The oldest types are mechanicaly-driven spring contacts. Later types use a meter-driven shutter to pulse a relay through a light pickup. If a new installation is requested of a utility, the device will likely be the newest version of the light-operated transducer. All types are upwardly compatible.

Analog power meters give a direct reading of instantaneous power demand, and are most often used by the consumer to monitor machinery and building circuits in an effort to damp-out demand peaks. These devices commonly drive panel meters or pen-chart recorders. In Japan and Europe, solid-state meters using direct inductive pickups, as well as solid-state watt hour meters, are being phased in by many utilities. Experiments in remote peak demand sensing in Europe have proven the feasibility of automated power billing. Analog signals from the demand sensor are usually digitized and transmitted over conventional telephone lines.

D. Conclusions

The impulse-device block interval meter has advantages over any of the dozens of analog meters available:

1. The impulse device requires no calibration; the analog meters require initial calibration and periodic recalibration.
2. Although it is composed of moving parts, the impulse device is electronically simpler than the analog devices, and could therefore be more reliable.
3. Reliability figures are available for impulse devices from over 30 years of use in the field, with thousands of units. The analog meter is a comparatively new device.
4. The impulse device requires only a poor-quality transmission line to the FMS, as:
   a. The signal is digital,
   b. The logic level can be high (0 - 50 V), and
   c. The data rate is very low.

   The analog device requires higher-quality lines.
5. The impulse device requires the simplest digital interface: a single sense line. The analog device requires a more complex interface: an analog-to-digital converter.

Neither method of demand digitization, using impulse or analog devices, requires unusual memory reach or calculations for a microprocessor.

3.0 Steam Metering

To measure the energy consumed by a system from a circulating fluid, three quantities must be known:

1. Temperature of the fluid before it enters the energy consumer (heating plant);
2. Temperature of the fluid as it leaves the energy consumer; and
3. Volumetric flow rate of the fluid.through the energy consumer.

The power consumed by the system, or heat loss rate of the fluid, is then calculated using the equation:

$$Q = KV(\Delta t), \quad (7)$$

where Q is the loss rate, in Btu/hr, K is a constant of proportionality, V is the volumetric flow rate of the fluid in cubic feet per minute (CFM), and Δt is the difference in temperature between the hot and cold legs of the system. Figure 9 is a schematic diagram of an energy metering system for a fluidic loop.

Equation 7 assumes an incompressible fluid (water). The same basic calculation holds for compressible fluid systems (gases), but the volumetric flow rate must be corrected for differences in fluid densities due to pressure and temperature changes. For perfect gases, a relatively simple algebraic model (gas law) can be used to adjust the flow rate, given temperature and pressure. For an imperfect gas (steam) the correction factor is best found using data taken from a three-dimensional "surface-graph," arrived at by empirical means (steam tables).

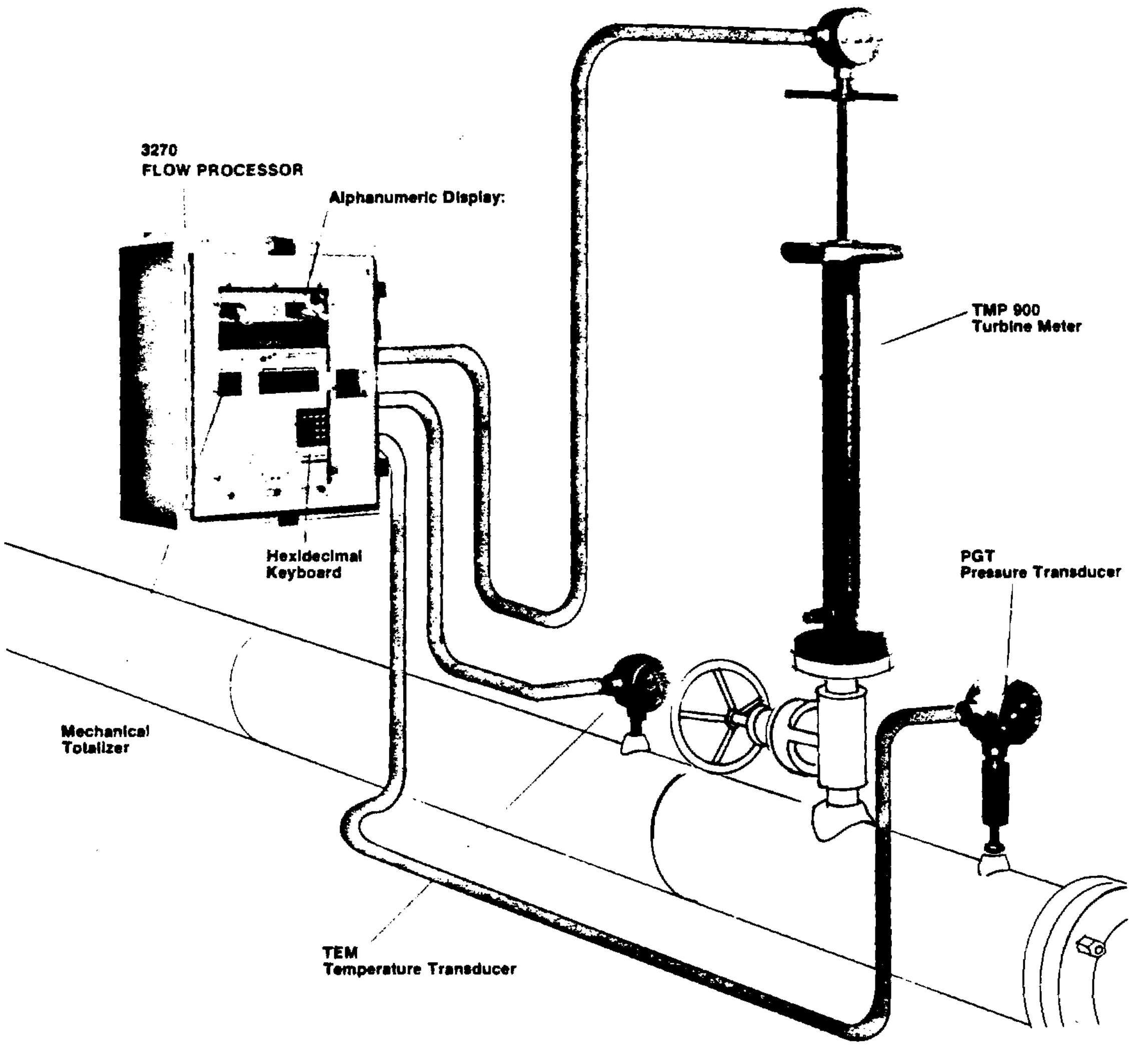


Figure 9. Energy Metering System for a Fluidic Loop

A point of controversy in the measurement of energy consumed in a steam-heating system is the selection of the steam flowmeter, used to measure the valumetric flow rate.

There are at least ten major flowmeter types commercially available, as described briefly in tables 8 through 17 (from Chilton's Instruments and Control systems). A survey of the data shows that of these ten major types, three types and four subtypes are suitable for steam flow measurement:

1. Variable Area (Rotometer),
2. Turbine,
3. Target,
4. Differential Pressure, Orifice,
5. Differential Pressure, Venturi,
6. Differential Pressure, Flow Nozzles and Tubes, and
7. Oscillatory, Vortex Shedding (Bluff Body).

Of these types, the differential pressure meter, the vortex shedding meter, and the turbine meter are possibly the most applicable to the FMS application.

A. Differential Pressure Meters

A differential pressure device generally consists of a constriction, either an orifice plate or a venturi throat, mounted in a straight section of steam pipe. The orifice plate is simply a meter disk with an aperture of known diameter drilled through its center. The flow rate of steam may be derived from the pressure drop across the constriction, using the equation

Service:	Liquids and gases, including steam
Design Pressure:	Up to 350 psig (glass); to 720 psig (metal)
Design Temperature:	Up to $400^{o}F$ (glass); to $1,000^{o}F$ (metal)
Flow Range:	Liquids 0.01 cc/min to 4,000 gpm; gases 0.3 cc/min to 13,003 CFM
Scale:	Linear or logarithmic
Signal:	Visual
Accuracy:	+/-0.5% of rate to +/-10% of full scale, depending on type, size and calibration
Rangeability:	5:1 to 12:1
End Connections:	Female pipe threaded or flanged
Sizes:	Up to 3 inches
Advantages:	Inexpensive, somewhat self-cleaning; viscosity variation not critical; direct indicating; no power required; can be direct-mass device; minimum piping requirements
Limitations:	Requires accessories for data transmission; must be vertically mounted; gas use requires minimum backpressure

Table 8. Variable Area (Rotometer) Flowmeter Characteristics

Service:	Clean liquids and gases, including steam
Design Pressure:	Up to 3,000 psig
Design Temperature:	$-450^{o}F$ to $500^{o}F$
Flow Range:	0.001 through 40,000 gpm liquids, to 10,000 SCFH gases
Scale:	Linear when Reynolds number is 10,000 or higher
Signal:	Frequency
Accuracy:	+/-0.25% of rate liquids; +/-1% of rate gas
Rangeability:	10:1 to 50:1
End Connections:	Flanged or threaded
Sizes:	Up to 24 inches (sampling types available)
Advantages:	One of the most accurate meters; good operating range; easy-to-install and maintain; very low rate designs available; small in size; lightweight
Limitations:	Sensitive to increasing viscosity; avoid use where liquid state may change; gas versions require care when used in varying flow rate applications

Table 9. Turbine Flowmeter Characteristics

Service:	Electrically conductive liquids or slurries
Design Pressure:	Up to 1,500 psig
Design Temperature:	Up to 360^{o}F
Flow Range:	0.01 through 500,000 gpm
Scale:	Linear
Signal:	Analog electronic; digital
Accuracy:	+/-0.5% of rate to +/-2% of full scale
Rangeability:	10:1 for any span (30:1 of span adjustment plus 10:1 range for any given span setting)
End Connections:	Flanged
Sizes:	0.1 inches to 96 inches (sampling type available)
Advantages:	Unaffected by change in fluid density, viscosity, zero head loss; bi-directional; no flow obstruction; easy to respond
Limitations:	Relatively expensive; always requires AC or DC power; only works with liquids whose conductivity is typically five microsiemen or higher; larger sizes big and heavy

Table 10. Magnetic Flowmeter Characteristics

Service:	Clean liquids and gases
Design Pressure:	Up to 1,400 psig for liquid or gas
Design Temperature:	Up to 600°F liquids; up to 250°F gas
Flow Range:	0.01 to 9,000 gpm liquid; 0 to 100,000 SCFH gas
Scale:	Linear
Signal:	Frequency
Accuracy:	+/-0.5% of rate liquid; +/-1% full scale gas
Rangeability:	Typically 10:1
End Connections:	Flanged or threaded
Sizes:	Up to 12 inches
Advantages:	Ideal for viscous liquids; good for custody transfers, batching, blending; requires no electrical power
Limitations:	Subject to mechanical wear; requires periodic proving; sensitive to dirt and may require upstream filters; larger sizes are excessive in size and weight, may require special care in installation

Table 11. Positive Displacement Flowmeter Characteristics

Service:	Relatively clear liquids
Design Pressure:	Through pipe - determined by pipe limits; clamp-on - no limit
Design Temperature:	-300°F to +500°F
Flow Velocity:	Typically to 40 ft/sec
Scale:	Linear
Signal:	Analog or digital
Accuracy:	+/-1% of rate
Rangeability:	-40 ft/sec to +40 ft/sec
End Connections:	Flanged (clamp-on design available)
Sizes:	1/2 inch and up
Advantages:	No flow obstruction; can be bidirectional; use with practically any relatively clean liquid
Limitations:	Must have uniform flow profile; liquids; must be relatively clean

Table 12a. Ultrasonic Flowmeter Characteristics (Transmit Time)

Service:	Liquids with entrained or suspended solids
Design Pressure:	No limit
Design Temperature:	$-300^{o}F$ to $+250^{o}F$
Flow Velocity:	To 50 ft/sec
Scale:	Linear
Signal:	Analog or digital
Accuracy:	Typically +/-3% of full scale
Rangeability:	-40 ft/sec to +40 ft/sec
End Connections:	Clamp-on
Sizes:	1/2 inch and up
Advantages:	Can handle inorganic slurries and aerated liquids; can be installed without process shutdown
Limitations:	Not suitable for clean liquids; requires minimum straight run

Table 12b. Ultrasonic Flowmeter Characteristics (Doppler Clamp-on)

Service:	Liquids and gases, including steam
Design Pressure:	Determined by transmitter
Design Temperature:	-20^{o}F to 250^{o}F
Flow Range:	From 0.1 cc/min up, or gas equivalent
Scale:	Square root
Signal:	Analog electronic or pneumatic
Accuracy:	+/-1% of max flow
Rangeability:	4:1
End Connections:	Mounts between flanges
Sizes:	Determined by pipe size
Advantages:	Easily installed; uses one transmitter regardless of pipe size; low cost; wide variety of types and materials available; easily changed capacity
Limitations:	Not good for slurries, dirty fluids; corrosive fluids

Table 13a. Differential Pressure Flowmeter Characteristics (Orifice)

Service:	Liquids and gases, including steam
Design Pressure:	Determined by transmitter
Design Temperature:	Determined by transmitter
Flow Range:	From 5 gpm liquid; 20 SCFM gas; determined by pipe size
Scale:	Square root
Signal:	Analog electronic or pneumatic
Accuracy:	+/-0.5% to 3% of rate
Rangeability:	5:1
End Connections:	Flanged
Sizes:	3 inches to 72 inches
Advantages:	Good accuracy; low permanent loss; good for slurries and dirty fluids; uses one transmitter regardless of pipe size
Limitations:	Most expensive differential pressure producer; generally limited to air and water

Table 13b. Differential Pressure Flowmeter Characteristics (Venturi)

Service:	Liquids and gases, including steam
Design Pressure:	Determined by transmitter
Design Temperature:	Generally determined by transmitter
Flow Range:	From 5 gpm liquid; from 20 SCFM gas equivalent
Scale:	Square root
Signal:	Analog electronic or pneumatic
Accuracy:	+/-0.5% to 3% of rate
Rangeability:	5:1
End Connections:	Flanged or mounted between flanges
Sizes:	3 inches to 48 inches
Advantages:	Economical, low permanent loss; uses one transmitter regardless of pipe size
Limitations:	May not be good for dirty fluids; generally limited to air and water

Table 13c. Differential Pressure Flowmeter Characteristics (Flow Nozzles and Tubes)

Service:	Liquids and gases
Design Pressure:	Determined by transmitter
Design Temperature:	Determined by transmitter
Flow Range:	Determined by pipe size
Scale:	Square root
Signal:	Analog electronic or pneumatic
Accuracy:	+/-5% to +/-10% of full scale
Rangeability:	3:1
End Connections:	Insert probe
Sizes:	Unlimited probe length
Advantages:	Very low cost; uses one transmitter regardless of pipe size
Limitations:	Does not sample full stream; limited accuracy

Table 13d. Differential Pressure Flowmeter Characteristics (Pitot)

Service:	Liquids and gases
Design Pressure:	Determined by transmitter
Design Temperature:	Determined by transmitter
Flow Range:	Determined by pipe size
Scale:	Square root
Signal:	Analog electronic or pneumatic
Accuracy:	+/-5% to +/-10% of full scale
Rangeability:	3:1
End Connections:	Mounts at pipe elbow
Sizes:	Determined by pipe size
Advantages:	Very economical; easily installed; uses one transmitter regardless of pipe size
Limitations:	Not good for low velocity; requires long, straight pipe runs upstream and downstream of elbow; not as accurate as other types

Table 13e. Differential Pressure Flowmeter Characteristics (Elbow)

Service:	Liquids and gases, including steam
Design Pressure:	Up to 1,440 psig
Design Temperature:	Typically 350^{O}F
Flow Range:	3 to 5,000 gpm liquid; 10,000,000 SCFH gases
Scale:	Linear
Signal:	Frequency
Accuracy:	+/-1% of rate to +/-1% of full scale
Rangeability:	8:1 to 15:1
End Connections:	Flanged or threaded
Sizes:	Up to 12 inches (sampline types available)
Advantages:	No moving parts; suitable for wide variety of liquids; excellent combination of price and performance
Limitations:	Straight piping required; sensitive to increasing viscosity; Reynolds number must be at least 10,000 for linear performance

Table 14a. Oscillatory Flowmeter Characteristics (Vortex Shedding, Bluff Body)

Service:	Liquids
Design Pressure:	Up to 600 psig
Design Temperature:	$0^{o}F$ to $250^{o}F$
Flow Range:	1 to 1,000 gpm
Scale:	Linear
Signal:	Analog electronic or pneumatic pulse
Accuracy:	+/-1% of rate
Rangeability:	Typically 30:1
End Connections:	Mounts between flanges
Sizes:	1 inch to 4 inches
Advantages:	No moving parts; suitable for wide variety of liquids; excellent combination of price and performance
Limitations:	Straight piping required; sensitive to increasing viscosity; minimum operating Reynolds number is 3,000

Table 14b. Oscillatory Flowmeter Characteristics (Fluidic, Coanda Effect)

Service:	Gases
Design Pressure:	Up to 1,400 psig
Design Temperature:	-100^{o}F to 350^{o}F
Flow Range:	Up to 10,000,000 SCFH
Scale:	Linear
Signal:	Frequency
Accuracy:	+/-1.25% of rate
Rangeability:	8:1 to 25:1
End Connections:	Flanged
Sizes:	1 inch to 6 inches
Advantages:	No moving parts; ideal for corrosive and difficult gases
Limitations:	Expensive; has minimum combination of flow rate and operating pressure

Table 14c. Oscillatory Flowmeter Characteristics (Vortex Precession)

Service:	Liquids in open channels
Flow Range:	From 1/2 gpm up
Scale:	Proportional to the measured head to the 3/2 power for rectangular and trapezoidal weirs and Parshall flumes; proportional to the measured head to the 5/2 power for V-notch weirs
Accuracy:	3% to 5% full scale
Rangeability:	75:1 rectangular, trapezoidal weirs, Parshall flumes; 500:1 V-notch weirs
Sizes:	From 1 inch up
Advantages:	Ideal for water and waste water flows; very slight head loss; low cost
Limitations:	Not very accurate

Table 15. Weir and Flume Flowmeter Characteristics

Service:	Liquids and gases
Design Pressure:	Up to 1,500 psig
Design Temperature:	-60°F to 165°F
Flow Range:	Up to 30,000 pph liquid; 6,000 pph gas
Scale:	Linear
Signal:	Analog electronic or frequency
Accuracy:	+/-1% full scale to +/-1% of rate
Rangeability:	10:1
End Connections:	Flanged or threaded
Sizes:	Up to 6 inches
Advantages:	Measures mass directly
Limitations:	Expensive; complex

Table 16. Mass Flowmeter Characteristics

Service:	Liquids and gases, including steam
Design Pressure:	Up to 10,000 psig for liquids and gases
Design Temperature:	Up to 750^{o}F liquids and gases
Flow Range:	0.07 gpm up liquid; 0.35 CFM up gas
Scale:	Square root or linear
Signal:	Analog electronic or pneumatic
Accuracy:	+/-.5% to +/-5% of full scale
Rangeability:	10:1 for any span
End Connections:	Flanged, threaded, flared tube
Sizes:	Up to 8 inches (sampling types available)
Advantages:	No moving parts; relatively inexpensive; good for hot, tarry, sediment-bearing fluids
Limitations:	Need 20 diameters upstream and 10 diameters downstream of straight pipe to maintain accuracy; reading is percent of scale; limited range

Table 17. Target Flowmeter Characteristics

$$M = 358.93CYd^2F_a \sqrt{ph_w} / \sqrt{1-\beta^4} \qquad (8)$$

where

M = mass flowrate, lb/hr,

C = coefficient of discharge for the orifice,

Y = fluid expansion factor,

d = diameter of orifice, inches,

F_a = area factor for thermal expansion of the orifice,

p = density of steam entering the meter, lb/ft,

h_w = differential pressure, inches of water at 68^0F, and

β = d/D, where D is the inside diameter of the pipe.

The differential pressure, h_w, is measured with two pressure sensors, one on either end of the constriction. Many types of sensing elements have been used in such systems, including everything from glass-tube manometers to modern solid-state sensors. Much experience and knowledge has accumulated in this area, and the sensor hardware types are well developed. The lack of moving parts is considered to be a plus for this flowmeter type.

A disadvantage to the differential pressure meters is a lack of accuracy for low flow rates. Most industrial differential pressure meters are accuracy rated at their full scale, where the pressure difference is 10 or 20 psi. At the low end of the readable scale the pressure difference can be as low as a few inches of water. Pressure losses are typically lower for nozzles than for orifice plates, as shown in the values of the coefficient of discharge, C. C rages from about 0.60 to 0.65 for orifices and from 0.93 to 1.0 for nozzles. The best full-scale accuracy obtainable is approximately 0.2 to 0.5 percent of flow rate, while 1.0 to 2.0 percent and upward are usual, depending upon the calibration method used.

A more subtle difficulty with differential pressure schemes is the range of dependencies. The properties of the steam (wet, dry), flow conditions, and piping geometry all affect the flowrate measurement. The system must therefore be directly calibrated at installation. As the system ages, the accuracy will drift and degrade unless the system is periodically recalibrated, or if a relatively sophisticated process computer can predict the degredation and compensate accordingly.

Two analog channels must be read by a computer processing the differential pressure - derived flowrate, one for each pressure sensor.

B. Vortex Shedding Meters

This relatively new metering technique measures the oscillatory vortex shedding frequency for steam flow around an object in the flow stream. Output of this device is a digital pulse string, the frequency of which is related to the steam velocity by the function

$$fD/V = O(DV\rho/\mu), \qquad (9)$$

where

fD/V = Strouhal number,

$DV\rho/\mu$ = pipe Reynolds number,

V = steam velocity,

D = characteristic linear dimension of the meter element,

ρ = vortex shedding frequency, and

μ = steam absolute viscosity.

Advantages of using the vortex shedding instrument in FMS installations include:

1. The accuracy is essentially independent of flowrate.
2. The output signal is digital, simplifying a microcomputer interface.
3. Vortex units do not drift or degrade when used with clear fluids.
4. No recalibrations are necessary.

Performance documentation is sparse for this device, and it lacks the rich experience of the differential pressure flowmeters, but it should be carefully considered for use in an FMS.

C. Turbine Meters

The turbine meter consists of a free-running, bladed turbine rotor held in the flow stream. Fluid flow through the turbine causes the rotor to rotate at an angular velocity proportional to the linear velocity of the fluid. Rotation of the turbine is detected electronically, with either magnetic, radio-frequency, or capacitive, non-instrusive pickups.

There are two major subtypes of turbine meters: the full bore meter, and the insertion meter. Full-bore meters occupy the entire inside diameter of a process pipe at the sampling point, with a full-diameter turbine rotor. Actual average velocity of the fluid is measured, with excellent linearity. (Local velocity of a fluid depends upon radial position in the pipe.) An insertion meter samples a strategic location in a pipe with a turbine that is small compared with the pipe diameter. The turbine, typically one-inch diameter, is located at a critical location in the flow profile, at the end of a stalk inserted into the pipe. A representative turbine insertion meter is the TMP-900, built by Engineering

Measurements Company, Inc., Longmont, Colorado. This unit is shown schematically in figure 10.

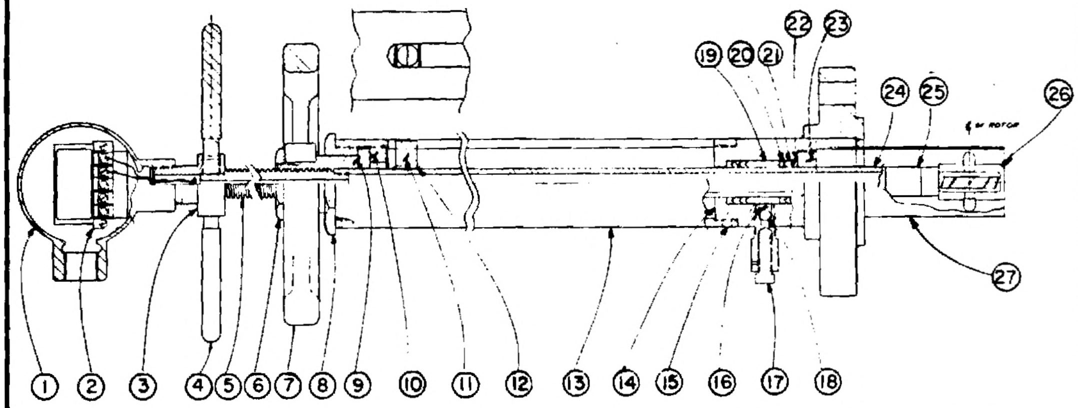
Rotation of the turbine is detected with a magnetic pickup. A built-in preamplifier provides a logical, square-wave output. Advantages to using the insertion turbine meter over other applicable flow-measurement methods are summarized as follows:

1. Extreme range of flow rates are measurable; typically better than other flowmeter types by a factor of ten.
2. High accuracy, especially at low flow rates.
3. Linear output, requiring a minimum of digital processing. Flow velocity is determined using the turbine with the equation;

$$V = \frac{f_p}{K_f} \tag{8}$$

   where f_p is the rotor pulse frequency in hertz, K_f is the probe velocity constant, in pulses per foot, and V is the flow velocity in feet/second.
4. Signal is digital pulses, and the logic level can be high (+24V). High-quality signal lines are not necessary.
5. The insertion meter is less expensive than the full-bore type.
6. The insertion meter can be retrofit to existing pipes without costly re-piping.
7. The insertion meter can be withdrawn from or inserted into an active steam pipe. This feature simplifies periodic or catastrophic maintenance procedures.

Major disadvantages are associated with the relative fragileness of the turbine rotor. The rotor is mounted on jewelled bearings, which cannot



1. Junction Box
2. Pre-Amplifier/Transmitter
3. Junction Mount
4. Orientation Lever
5. Upper Stem
6. Hand Wheel Nut
7. Hand Wheel
8. Gland Nut
9. Yoke Sleeve
10. Yoke Sleeve Stop
11. Depth Bar
12. Tru-Arc Circlip
13. Bonnet
14. Housing and Flange Weldment
15. Set Screw
16. Roll Pin
17. Packing Screw
18. Steel Check Ball
19. Plastic Packing
20. Chevron Ring
21. Chevron Ring
22. Chevron Ring
23. Packing Nut
24. Lower Stem
25. Pickup Housing
26. Rotor Housing
27. Protective Shroud

Figure 10. The TMP-900 Turbine Meter

withstand shock. Rough installation can be fatal. Bearing life must be considered. Specifications for the Model TMP-900 turbine meter, Engineering Measurements Company, Inc., are listed in table 18.

Steam flow computers are available as line-items from companies such as Engineering Measurements. Temperature and pressure transducers as used in these systems are available as companion items to turbine meters, such as the TMP-900. Specifications for the TEM series 30 temperature transducers are listed in table 19, and specifications for the PGT series 20 pressure transducer are in table 20.

Temperature and pressure readings are transmitted as analog currents (current loop is used for long-range capability), and must be converted to digital accordingly.

D. Summary

Choices for a steam metering system in an FMS thus distills down to three metering schemes: the differential pressure meter, the vortex shedding meter, and the turbine meter. Advantages, disadvantages, and features of these three types are compiled in summary table 21.

E. Conclusions

Steam flow measuring is a well-developed technology. Little new work is required to design a digital system which records net steam energy flow, provided detailed data and methods are readily available. Such a system can, in fact, be bought on an off-the-shelf basis. Unfortunately, steam table algorithms used in some commercial units are considered proprietary, and therefore the research necessary for a new design of a computer-driven meter system could be a costly duplication of effort.

Insertion Length:	21 inches
Pressure Rating:	Flange rating in PSI
Seal:	Packing gland
Mount:	2" MNPT or flanged
Material:	Wetted parts 316 stainless steal Body parts 1018 carbon steel
Temperatures:	Wetted parts 750°F maximum Seal 450°F
Rotor Velocity:	Model 100, 3 - 500 FPS Model 150, 2 - 100 FPS Model 175, 1 - 75 FPS
Flow Limit:	2 - 500 ft/sec
Linearity:	Upper 5:1 range, +/-0.5% Normal 10:1 range, +/-1.0%
Repeatability:	+/-0.25%
Pressure Drop:	Negligable
Output:	10 V p-p square wave
Power:	15 - 40 VAC, or 15 - 50 VDC

Table 18. TMP-900 Turbine Meter Specifications

Full-Scale Temperature:	800°F
Major Dimensions:	3-7/8" x 5-1/2", exposed
Accuracy:	+/-0.1%
Repeatability:	+/-0.001%/year
Material:	100 ohm Platinum RTD 314 stainless thermowell
Pressure:	Unprotected, 500 PSI
Fitting:	1/2 inch NPT Male
Interchangeability:	+/-0.6% full scale, +/-1°F
Output:	2-wire current loop; 4/20 ma
Source Voltage:	15 - 60 VDC
Stability:	+/-0.005%/°C (50 ppm/°C)

Table 19. TEM Series 30 Temperature Transducer Specifications

Full-Scale Pressure:	Model	PSIG	PSIA
	PGT 20-0-15	--	15
	PGT 20-0-30	30	30
	PGT 20-0-50	50	50
	PGT 20-0-100	100	100
	PGT 20-0-150	150	150
	PGT 20-0-200	200	200
	PGT 20-0-250	250	
	PGT 20-0-300	300	
	PGT 20-0-500	500	
	PGT 20-0-1000	1,000	
	PGT 20-0-5000	5,000	
Major Dimensions:	3-7/8" x 8-5/16", exposed		
Accuracy:	+/-0.2% full scale		
Overpressure Tolerance:	2 x full scale		
Fitting:	1/4 inch NPT female		
Output:	2-wire current loop, 4-20 ma standard		
Source Voltage:	15 - 60 VDC		
Operating Temperature:	-40^{o}F to +185^{o}F		
Stability:	+/-0.005%/^{o}C		

Table 20. PGT Series 20 Pressure Transducer Specifications

	Pressure Differential	Vortex Shedding	Turbine
Design Temperature	$-20^{o}F$ to $250^{o}F$	$350^{o}F$	$< 500^{o}F$
Flow Range	> 0.1cc/min	$< 10^{7}$SCFH	$< 10^{4}$SCFH
Scale	Square Root	Linear	Linear when Re $\geq 10^{4}$
Signal	Analog	Digital	Digital
Full Scale Accuracy	$\pm 1\%$	$\pm 1\%$	$\pm 1\%$
Maximum Rangeability	4:1	15:1	50:1
Common Failure Modes	Pressure Sensor Failure	Thermocouple Heater Burnout	Bearing Failure
Number of Moving Parts	0	0	1
Accuracy Drift Modes	Aperture Erosion, Sensor Fatigue	No Serious Drift	Bearing Friction Buildup, Turbine Blade Erosion
Catastrophic Failure	Unlikely	Unlikely	Inevitable
Calibration	Periodic	Installation	Periodic

Table 21. Pressure Differential, Vortex Shedding, and Turbine Meter Performance Summary

Of the three metering techniques surveyed here, vortex shedding may be the most attractive, but the limited depth of this report precludes endorsement of any one method.

The range of steam states available in a building heat system is actually a small subset of the full steam tables. Steam characteristic curves can be stored as polynomials, or as tables. Neither method involves unusual memory reach or calculational requirements.

References

1. International Conference on Metering and Apparatus for Modern Electricity Supply Tariffs, Institution of Electrical Engineers, Kent, England, 1967.

2. International Conference on Metering, Apparatus, and Tariffs for Electricity Supply, Institution of Electrical Engineers, London, England, 1977.

3. International Conference on Metering, Apparatus, and Tariffs for Electricity Supply, Institution of Electrical Engineers, London, England, 1972.

4. G. F. Tagg, Electrical Measuring Instruments, John Wiley and Sons, New York, 1952.

5. H. Buckingham, E. M. Price, Principles of Electrical Measurements, English Universities Press, London, England, 1966.

6. "How to Select Contact Devices and Impulse Generators," GET-3048C, General Electric, Somersworth, New Hampshire, 1971.

7. "Manual of Demand Meters," GET-2327C, General Electric, Somersworth, New Hampshire, 1968.

8. "Instructions, Types PD-55-F and PD-57F Printing Demand Meters," GEH-2764F, General Electric, Somersworth, New Hampshire.

9. "Instructions, Demand Meters, Types G9, G5-9, and G5-12," GEH-225T, General Electric, Somersworth, New Hampshire, 1971.

10. "Instructions, Type D-52 Pulse Generator for Demand Metering," GEH-2786, General Electric, Somersworth, New Hampshire, 1972.

11. "Type D-72 Pulse Initiator for Demand Metering," GEH-2786-1, General Electric, Somersworth, New Hampshire, 1976.

12. "Type D-51 Pulse Initiator for Demand Metering," GEH-2781-B, General Electric, Somersworth, New Hampshire, 1977.

13. "Type D-41 Impulse Generator for Demand Metering," GEH-2767C, General Electric, Somersworth, New Hampshire, 1970.

14. "Instructions, Type SST-3 Solid-State Totalizer," GEH-2784B, General Electric, Somersworth, New Hampshire, 1975.

15. "Product Data Bulletin, Series PCE and VCE Electronic Watt and VAR Transducer for Three-Phase and Single-Phase Systems," TD-1003-B, RIS Rochester, New York, 1977.

16. "A Condensed Guide to RIS Power Transducers," TD-1000-B, RIS, Rochester, New York, 1978.

17. "RIS Series PCE and VCE Watt and VAR Transducers: Basic Operation, Application, and Calibration," P-10, RIS, Rochester, New York, 1976.

18. "Series CCC, VCC, and VCX Current and Voltage Transducers," TD-1001-B, RIS, Rochester, New York, 1977.

19. "Calibration Procedures for CCC-1B Series," A-1017-445 Rev. A, RIS, Rochester, New York, 1977.

20. "Calibration Procedure for PCE, PCM, PPCM, VCE, VCM, PVCM, PCS, and VCS Series," A-1020-120, RIS, Rochester, New York, 1978.

21. "Model SC-1356 Linear Integration; Model SC-1398 Square Root Integrator," 1356/58, RIS, Rochester, New York, 1979.

22. Model TME 1180, 1182, 1184, 1186 Turbine Meter Electronics Instruction Manual: Volume One: Installation/Operation, EMCO, Longmont, California, 1979.

23. Turbine Meter Probe Model TMP-900 Instruction Manual, EMCO, Longmont, California, 1979.

24. "Dimensional Information for Hot Tapping," Topaz, Houston, Texas, 1973.

25. C. E. Miller, "On the Application and Performance of Insertion Turbine Meters for Steam FLow Measurement," IDHA Annual Conference, Pittsburgh, Pennsylvania, 1978.

26. "Industrial Turbinr Meters," TMI Series 80-100, EMCO, Longmont, California.

27. "Pressure Gauge Transmitter," PGT Series 20, EMCO, Longmont, California.

28. "Temperature Transmitter," TEM Series 30, EMCO, Longmont, California.

29. "Heat Flow Computer," FCH Series 1170, EMCO, Longmont, California.

30. "Turbine Meter Electronics," TME Series 1180, EMCO, Longmont, California.

31. "Multi-Channel Flow Processor," Earl F. Scott Co., Atlanta, Georgia, 1980.

32. "Steam Flow Computer," EMS Series 3270, EMCO, Longmont, California.

33. "Insertion Turbine Meter General Specifications," Earl F. Scott Co., Atlanta, Georgia, 1979.

34. "Turbine Meter Probe," TMP-Series 600, EMCO, Longmont, California.

35. "Turbine Meter Probes," TMP, EMCO, Longmont, California.

36. "Turbine Flowmeter for Steam," TMP-Series 5, EMCO, Longmont, California.

37. J. W. Hayes, J. J. Rusnak, "Use Meters to Put Teeth into Your Conservation Programs," Conference on Industrial Conservation Technology, Houston, Texas, 1980.

38. "Energy Management," EMCO, Longmont, California.

39. Robert P. Martin, "Flow, Velocity, Pressure, and its Measurement," Hy-Cal Engineering, Sante Fe Springs, California, 1981.

40. "Large Series Flow Meters," UFI, Hazel Park, Michigan, 1979.

41. "Energy and Process Thermocouples," Gordon Temperature Measurement, Richmond, Illinois, 1978.

42. "Flow Meters," Data Industrial, Pacasset, Massachusetts.

43. "Controlatron Series 450 Clamp-On Energy Flow Meter," Controlatron, Hauppauge, New York, 1980.

44. "Controlatron System 480 Ultrasonic Flowmeter," Controlatron, Hauppauge, New York, 1980.

45. "Controlatron Autajust 180 Doppler Flowmeter," Controlatron, Hauppauge, New York, 1980.

46. "Yewflow Vortex Flowmeter," Yokozawa Electric Works, Tokyo, Japan, 1980.

47. "7000 Series BTU Measurement System," Hersey, Spartanburg, South Carolina, 1978.

48. "Silicon Diaphragm Pressure Transducers," Foxboro, San Jose, California, 1980.

49. "Standard Line Turbine Flowmeters," Flow Technology, Phoenix, Arizona.

50. "Omniflow Turbine Flow Transducers," Flow Technology, Phoenix, Arizona.

51. "Electronic Readant Equipment and Accessories for Use with Turbine Flowmeters," Flow Technology, Phoeniz, Arizona.

52. "Pressure/Position Transducers and Transmitters," Bourns, Riverside, California, 1977.

53. "Chilton's Instruments and Control Systems," Chilton, Radnor, Pennsylvania, 1980.

APPENDIX A

Equivalent Teeth Tables, for gear train selections when fitting block-interval demand devices to wattmeters. From GET-3048C.

EQUIVALENT TEETH

The table is prepared primarily for contact devices driven from a separate worm on the meter shaft; that is, contact devices for a-c watthour meters. The pitch of this separate worm may be single, double, triple, or quadruple.

Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *	Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *	Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *
0.15	1	1	20:3	1-1/7	1	1	7:8	3.2	4	1	5:4
0.16	1	1	25:4	1.2	2	1	5:3	3.24	2	1	50:81
0.162	1	1	105:17	1.25	5	1	4:1	3-1/3	5	1	3:2
1/6	1	1	6:1	1.26	7	1	50:9	3.36	7	2	25:6
0.18	1	1	50:9	1.28	4	1	25:8	3.375	6	1	16:9
0.2	1	1	5:1	1-1/3	2	1	3:2	3-3/7	3	1	7:8
0.225	1	1	40:9	1.35	6	1	40:9	3.5	7	1	2:1
0.24	1	1	25:6	1.4	7	1	5:1	3-5/9	3	1	27:32
0.25	1	1	4:1	1.44	6	1	25:6	3.6	4	1	10:9
0.2688	1	1	93:25	1.5	2	1	4:3	3.75	5	1	4:3
0.27	1	1	100:27	1.575	7	1	40:9	3.78	7	3	50:9
0.3	1	1	10:3	1.6	2	1	5:4	3.84	4	1	25:24
0.32	1	1	25:8	1.62	1	1	50:81	3.9375	7	1	16:9
1/3	1	1	3:1	1-2/3	5	1	3:1	4.0	4	1	----
0.36	1	1	25:9	1.68	7	1	25:6	4.05	6	3	40:9
0.4	1	1	5:2	1.6875	3	1	16:9	4-1/6	5	1	6:5
0.45	2	1	40:9	1-5/7	2	1	7:6	4.2	7	1	5:3
0.48	2	1	25:6	1.75	7	1	4:1	4.32	6	2	25:9
0.5	1	1	2:1	1.8	2	1	10:9	4.4	2	1	5:11
0.525	3	1	40:7	1.875	2	1	16:15	4.5	6	1	4:3
0.54	3	1	50:9	1.92	2	1	25:24	4-4/7	4	1	7:8
0.5625	1	1	16:9	2.0	2	1	----	4-2/3	7	1	3:2
0.6	1	1	5:3	2.025	3	3	40:9	4.68	6	1	50:39
0.66	3	1	50:11	2.1	7	1	10:3	4.6875	5	1	16:15
2/3	1	1	3:2	2.16	2	1	25:27	4.725	7	3	40:9
0.675	3	1	40:9	2.2	1	1	5:11	4.8	6	1	5:4
0.7	7	1	10:1	2.25	3	1	4:3	4.86	3	1	50:81
0.72	3	1	25:6	2-2/7	2	1	7:8	5.0	5	1	----
0.75	1	1	4:3	2-1/3	7	1	3:1	5.04	7	2	25:9
0.8	1	1	5:4	2.4	3	1	5:4	5.0625	3	3	16:9
5/6	5	1	6:1	2.5	5	1	2:1	5.25	7	1	4:3
0.84	3	1	25:7	2.52	7	1	25:9	5-1/3	4	2	3:2
0.9	1	1	10:9	2-2/3	4	1	3:2	5.4	6	1	10:9
0.9375	1	1	16:15	2.7	3	1	10:9	5.6	7	1	5:4
0.96	1	1	25:24	2.8	7	1	5:2	5.625	6	1	16:15
1.0	1	1	----	2.8125	3	1	16:15	5-5/7	5	1	7:8
1.05	1	1	20:21	2.88	3	1	25:24	5.76	6	1	25:24
1-1/15	1	1	15:16	3.0	3	1	----	6.0	6	1	----
1.08	1	1	25:27	3.125	5	1	8:5	6.075	9	3	40:9
1.125	1	1	8:9	3.15	7	2	40:9	6.25	5	1	4:5

*Gear Ratio = $\frac{\text{Rev WW}}{\text{Rev Cam}}$. If no ratio is listed, use 1:1 in formulas.

EQUIVALENT TEETH

Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *	Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *	Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *
6.3	7	1	10:9	11-2/3	7	1	3:5	21	7	3	----
6.4	4	2	5:4	11.8125	7	3	16:9	21-1/3	4	4	3:4
6.48	4	1	50:81	12	6	2	----	21.6	6	4	10:9
6.5625	7	1	16:15	12.5	5	2	4:5	22	5	2	5:11
6.6	3	1	5:11	12.6	7	2	10:9	22.4	7	2	5:8
6-2/3	5	2	3:2	12.8	4	2	5:8	22.5	5	4	8:9
6.72	7	1	25:24	12.96	6	2	25:27	22.68	7	2	50:81
6.75	3	2	8:9	13.125	7	2	16:15	22-6/7	5	4	7:8
6-6/7	6	1	7:8	13.2	6	1	5:11	23.04	6	4	25:24
7.0	7	1	----	13-1/3	5	2	3:4	23-1/3	7	2	3:5
7.2	4	2	10:9	13.44	7	2	25:24	23.625	7	3	8:9
7.5	5	2	4:3	13.5	5	3	10:9	24	6	4	----
7.56	7	1	25:27	13-5/7	6	2	7:8	24.3	5	3	50:81
7.68	4	2	25:24	14	7	2	----	24.48	6	4	50:51
7.875	7	1	8:9	14.0625	5	3	16:15	25	5	2	2:5
8.0	4	2	----	14.4	6	2	5:6	25.2	7	4	10:9
8.1	3	3	10:9	14.58	3	3	50:81	25.3125	9	3	16:15
8-1/3	5	1	3:5	15	5	3	----	25.6	8	2	5:8
8.4	7	2	5:3	15.12	7	2	25:27	25.92	6	3	25:36
8.4375	3	3	16:15	15.1875	9	3	16:9	26.25	7	4	16:15
8-8/15	4	2	15:16	15.36	4	4	25:24	26.4	6	2	5:11
8.64	3	3	25:24	15.4	7	1	5:11	26-2/3	5	4	3:4
8.75	7	1	4:5	15.75	7	2	8:9	26.88	7	4	25:24
8.8	4	1	5:11	16	4	4	----	27	6	4	8:9
9.0	3	3	----	16.2	6	3	10:9	27-3/7	6	4	7:8
9-1/7	4	2	7:8	16-2/3	5	2	3:5	28	7	4	----
9-1/3	7	1	3:4	16.8	7	2	5:6	28.125	10	3	16:15
9.375	5	2	16:15	16.875	5	3	8:9	28.8	6	2	5:12
9.45	7	1	20:27	17-1/7	5	3	7:8	29.16	6	3	50:81
9.6	5	2	25:24	17.28	6	3	25:24	30	5	2	1:3
9.72	6	1	50:81	17.5	7	2	4:5	30.24	7	3	25:36
10	5	2	----	17.6	4	2	5:11	30.375	9	3	8:9
10.08	7	1	25:36	18	6	3	----	30.72	8	4	25:24
10.125	3	3	8:9	18-2/7	4	4	7:8	30.8	7	2	5:11
10-2/7	3	3	7:8	18-2/3	7	2	3:4	30-6/7	9	3	7:8
10.5	7	2	4:3	18.75	5	4	16:15	31.25	5	1	4:25
10-2/3	4	2	3:4	18.9	7	3	10:9	31.5	7	4	8:9
10.8	6	2	10:9	19.2	5	4	25:24	32	6	4	3:4
11	5	1	5:11	19.44	6	2	50:81	32.4	5	4	50:81
11.025	7	3	40:21	19.6875	7	3	16:15	33	5	3	5:11
11.2	7	2	5:4	19.8	3	3	5:11	33-1/3	5	4	3:5
11.25	5	2	8:9	20	5	4	----	33.6	7	2	5:12
11.34	7	1	50:81	20.16	7	3	25:24	33.75	9	3	4:5
11-3/7	5	2	7:8	20.25	9	2	8:9	34.2	7	3	50:81
11.52	6	2	25:24	20-4/7	6	3	7:8	34.132	8	4	15:16

*Gear Ratio = $\frac{\text{Rev WW}}{\text{Rev Cam}}$. If no ratio is listed, use 1:1 in formulas.

EQUIVALENT TEETH

Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *	Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *	Equivalent Cam Teeth (E_QT)	Points On Cam	Pitch Of Worm	Gear Ratio *
34-2/7	10	3	7:8	53-1/3	5	4	3:8	96	6	4	1:4
34.56	6	4	25:36	53.76	7	2	25:96	100	5	4	1:5
35	7	2	2:5	54	6	3	1:3	103.68	9	3	25:96
35.2	4	4	5:11	56	7	4	1:2	105	7	3	1:5
36	6	2	1:3	57.6	6	4	5:12	106-2/3	10	4	3:8
36-4/7	8	4	7:8	58.32	9	4	50:81	107.52	7	4	25:96
37-1/3	7	4	3:4	59.4	9	3	5:11	108	9	3	1:4
37.5	5	3	2:5	60	6	4	2:5	112	7	4	1:4
37.8	9	4	20:21	61.44	4	4	25:96	115.2	10	3	25:96
38.4	4	4	5:12	61.6	7	4	5:11	120	6	4	1:5
38.88	6	4	50:81	63	7	3	1:3	122.88	8	4	25:96
39.6	6	3	5:11	64	6	4	3:8	126	7	4	2:9
40	6	4	3:5	64.8	9	3	5:12	128	8	4	1:4
40.32	7	4	25:36	66	10	3	5:11	129.6	9	4	5:18
40.5	9	4	8:9	66-2/3	10	4	3:5	135	9	3	1:5
41-1/7	9	4	7:8	67.2	7	4	5:12	138.24	9	4	25:96
42	7	2	1:3	67.5	9	3	2:5	140	7	4	1:5
42-2/3	4	4	3:8	69.12	6	3	25:96	144	6	4	1:6
43.2	6	3	5:12	70	7	4	2:5	150	10	3	1:5
43.74	9	3	50:81	70.4	8	4	5:11	153.6	10	4	25:96
44	5	4	5:11	72	6	4	1:3	160	8	4	1:5
45	6	3	2:5	74-2/3	7	4	3:8	162	9	4	2:9
45.36	7	4	50:81	75	5	3	1:5	168	7	4	1:6
45-5/7	10	4	7:8	75.6	7	3	5:18	180	10	4	2:9
46.08	6	2	25:96	76.8	5	4	25:96	192	8	4	1:6
46.2	7	3	5:11	79.2	9	4	5:11	200	10	4	1:5
46-2/3	7	4	3:5	80	5	4	1:4	216	9	4	1:6
48	6	4	1:2	80.64	7	3	25:96	240	10	4	1:6
48.6	10	3	50:81	81	9	3	1:3				
50	5	4	2:5	84	7	4	1:3				
50.4	7	3	5:12	85-1/3	8	4	3:8				
51.2	8	4	5:8	86.4	9	4	5:12				
51.84	8	4	50:81	68	10	4	5:11				
52.5	7	3	2:5	90	6	3	1:5				
52.8	6	4	5:11	92.16	6	4	25:96				

*Gear Ratio = $\frac{\text{Rev WW}}{\text{Rev Cam}}$. If no ratio is listed, use 1:1 in formulas.