NBSIR 81-2285

Using Microcomputers to Monitor the Field Performance of Residential Heat Pumps

Center for Building Technology
National Engineering Laboratory
U.S. Department of Commerce
National Bureau of Standards
Washington, DC 20234

June 1981

Sponsored by
'' S. Department of Energy
'00 Independence Avenue, S.W.
ıshington, DC 20585

NBSIR 81-2285

USING MICROCOMPUTERS TO MONITOR THE FIELD PERFORMANCE OF RESIDENTIAL HEAT PUMPS

Charles W. Hurley and George E. Kelly

Building Equipment Division

Paul A. Kopetka

Building Thermal Performance Division

Center for Building Technology
National Engineering Laboratory
U.S. Department of Commerce
National Bureau of Standards
Washington, DC 20234

June 1981

Sponsored by
U.S. Department of Energy
1000 Independence Avenue, S.W.
Washington, DC 20585

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, ***Secretary***

NATIONAL BUREAU OF STANDARDS, Ernest Ambler, ***Director***

ABSTRACT

Field data on the heating and cooling performance of residential heat pumps were gathered for the purpose of verifying and refining laboratory testing procedures. This report describes the procedures, instrumentation, and microprocessor-based data acquisition system (DAS) used for evaluating the field performance of three residential heat pumps located in the Washington, D.C. area. The instrumentation, signal conditioning unit and DAS are described in detail since the designs employed are applicable to future testing projects of this type in both small and large scale field studies.

To avoid the large capacities of the DAS and data reduction facility required for on-line monitoring, a strategy was developed which used the on-line microcomputer in the field to reduce and analyze the raw data and record the calculated results. This reduced the amount of recorded data to an acceptable level and thereby extended the time period between data collection.

This report discusses the selection of the heat pumps utilized in this field study and the design and selection of the instrumentation and DAS. The requirements for scanning data and recording the results are also discussed.

The basic equations and the software for processing the data at the field units and for reducing and editing the raw data disks at a central microcomputer are described. Examples of printouts taken directly at the field units and from the data disks are shown.

Key Words: Analog signal conditioning; data acquisition system; field data acquisition; field instrumentation; field performance of heat pumps; heat pumps; heat pump test methods; microcomputer.

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

DISCLAIMER

Certain commercial equipment and instrumentation and data acquisition systems are identified by name in this paper in order to adequately describe the capabilities and technical features of hardware used in the instrumentation system. In no case does such identification imply recommendation or endorsement by the National Bureau of Standards, nor does it imply that the material or equipment identified is necessarily the best available for the purpose.

1. INTRODUCTION

Methods of laboratory testing, rating and estimating the heating and cooling seasonal performance of heat pumps have been established in previous research performed at NBS[1,2]. This work formed the basis of test procedures for this type of consumer product, promulgated by the Department of Energy in December 1979[3]. The major objective of the research discussed in this report was to gather actual field data on the heating and cooling performance of residential heat pumps for the purpose of verifying and refining these procedures, and to develop and document methods for evaluating the dynamic performance of heat pumps installed in residences.

This report describes the development of the procedures, instrumentation, and microprocessor-based data acquisition system used for evaluating the field performance of three heat pumps located in the Washington, D.C. area. The instrumentation and data acquisition systems are described in detail, since the basic designs which were employed are applicable to future testing programs on the performance of central heating and cooling equipment in both small and large-scale field programs. A second report, describing the actual data obtained during this study, will be published after the experimental phase of the program is completed.

At the onset of this project, an estimate was made of the amount of data required to obtain the information needed for estimating the seasonal performance of heat pumps. It soon became obvious that the raw data would require a data storage system and a data reduction facility with capacities much larger than warranted by a project of this type. To avoid this problem, a strategy was developed which involved gathering of all necessary raw data, using a microcomputer to reduce and analyze it in the field, and then storing the calculated results. A large portion of the raw data was not directly recorded. However, the raw data scans from every tenth on/off cycle were saved. The latter provided some detailed performance data and served as a check on the calculated results. This shifted the data reduction task from a central facility to the data acquisition systems in the field and reduced the amount of data stored to an acceptable level. An added benefit of this approach was allowing the data from the field to be scanned quickly and any erroneous results, due to improper operation of either the heat pump or the data acquisition system, to be quickly detected. Prompt action could then be taken to correct the problem and to minimize the amount of good data lost.

The next subsection of this report, section 1.1, discusses the selection of the heat pumps utilized in this field study, giving a brief description of each unit. Section 2 discusses the design of the instrumentation and data acquisition systems, giving the design objectives, the requirements of scanning the units under test for data and recording the results, and the selection of the data acquisition system. Section 3 describes the actual measurements made on each unit and the instrumentation used. The signals from the instrumentation required amplification, attenuation, and/or offsetting to allow the data acquisition system to process and record the data with the maximum possible accuracy. Section 4 describes the signal conditioning unit which performed this task, together with registering the pulses from the pulsing type instruments, and continually monitored the unit for any change in the mode of operation. A more detailed description of the microcomputer used as a data acquisition system in this project is given in section 5. Some of the problems encountered in the field-installed systems are also discussed in this section. The equations used for processing the data in the field units are described in section 6. Software for collecting and processing the raw data at the field units and for reducing and editing the raw data disks at a central microcomputer are briefly described in section 7.

Examples of printouts made directly from the raw data disks are shown in Appendix A. Typical examples of data available from the units in the field using a portable terminal are illustrated in Appendix B. A complete printout of the software used by the microcomputers in the field units and by the central microcomputer to categorize and display the engineering data from the raw data disks is found in Appendix C.

1.1 SELECTION OF FIELD UNITS FOR TESTS

The heat pumps utilized in this field study were selected from those belonging to NBS employees who volunteered to participate in the program. The use of the homes of NBS employees reduced the administrative problems involved in getting access to the homes and assured that the instrumentation would be properly protected. Although over 25 volunteers responded to a notice posted in the weekly NBS "Technicalendar," only 7 met the following requirements for the program.

1. The heat pump model had to be available for procurement for additional laboratory testing.

2. The system must have been installed properly and be operating in a manner recommended by the manufacturer. (Units were excluded for mechanical and operational alterations, such as attempts to supplement heating with wood furnaces and completely disabling the auxiliary resistance heaters.)

3. There should be only a single heat pump in a dwelling, and humidifiers must not have been installed in the supply duct, since these things would make data analysis much more difficult. The dwelling should be located in the local area in order to avoid extra legal problems in the metering of electrical power lines, and excessive travel time and costs.

4. The unit must be an air-source heat pump and not a water-source heat pump.

5. There should be no indication that a volunteer would interfere with instrumentation and/or demand to see the data prior to the publication of the final report.

6. The unit must be located in quarters adequate for installation of the necessary instrumentation without major remodeling.

The final selection of the units to be instrumented was based upon a desire to obtain data on a variety of defrost control systems used by different manufacturers. A brief description of the three units selected and instrumented for the study is given below:

One unit was a 2 1/2 ton, air-source, unitary heat pump with an outdoor compressor/coil section and an indoor coil located in an air handler containing the auxiliary heater elements (15 kW). This unit utilized an air pressure differential defrost system. A pressure sensor was located in the area between the outdoor coil and the outdoor fan. When the static pressure difference between this area and the atmospheric pressure exceeds 0.5 inches (12.7 mm) of water, the defrost cycle is activated. In the defrost mode, the power to the outdoor fan is turned off and the refrigerant reversing valve is activated changing the unit into the cooling mode which allows the hot gases to pass through the outdoor coil for defrosting. The defrost cycle is terminated when a temperature-sensing bulb located near the outdoor coil indicates a liquid refrigerant temperature of 65°F (18°C). The differential pressure switch is designed to operate when approximately 85 percent of the coil is blocked by frost accumulation.

A second unit was an air-source, unitary heat pump consisting of an outdoor fan-coil section, an indoor compressor section, and an indoor fan-coil section containing the auxiliary electric heaters (15 kW). This unit was designed to operate with a three-ton cooling capacity and utilized a time/temperature defrost control system. In the heating mode, a defrost timer is actuated every 90 minutes of compressor on-time. If the defrost thermostat is closed (it closes when the temperature of the refrigerant in the outdoor coil reaches 27 ± 4°F [2.8 ± 2.2°C]) when the timer is actuated, the unit goes into a defrost mode until the defrost thermostat reaches 80 ± 6°F (26.7 ± 3.3°C), or for a maximum of 10 minutes. During the defrost mode, the outdoor fan stops, the reversing solenoid valve is energized to switch the refrigeration system into the cooling mode, and an auxiliary electric heating element is activated. At the end of the defrost period, the system returns to the normal heating mode of operation.

A third unit was a three ton, air-source, unitary heat pump with an outdoor section and an indoor section with auxiliary heaters (15 kW). This unit utilizes both the pressure differential across the outdoor coil and the refrigerant temperature to initiate the defrosting process. When frost and ice forms on the outdoor coil during the heating mode and the air flow is restricted, the pressure differential defrost switch closes. If this switch remains closed for 12 seconds (a delay to avoid the effect of wind gusts) and the liquid refrigerant leaving the outdoor coil is 39°F (3.9°C) or lower, the defrost relay is energized. This relay energizes the reversing valve to switch the unit to the cooling mode, stops the outdoor fan, and energizes the first stage of the auxiliary heaters. The defrost cycle is terminated when the liquid line temperature in the outdoor unit exceeds

75°F (23.9°C) or if it exceeds 45°F (7.2°C) for 5 minutes. The latter condition allows the defrost cycle to terminate if the wind velocity does not permit the liquid line temperature to reach 75°F (23.9°C).

2. DESIGN OF INSTRUMENTATION AND DATA ACQUISITION SYSTEMS

2.1 DESIGN OBJECTIVES

The design of the instrumentation and data acquisition system (DAS) installed in the field was based on the objectives established at the onset of this task including those listed below:

1. The instrumentation and the DAS systems were to be designed to fall within a minimal practical cost range.

2. The systems to be installed at each field site should allow the raw data to be recorded and analyzed, and the analyzed data recorded, with the precision required.

3. Any altering of the field units by the installation of the instrumentation must be small and have a negligible effect, if any, on the performance of the unit.

4. All measurements necessary to determine the input and output of each unit were to be made to allow the cyclic, daily and seasonal performance of the units to be determined.

5. The instrumentation and DAS must be capable of continually monitoring the mode of operation of the unit (i.e., on, off, or defrost). The actual time of each cycle, the compressor on-time, and defrost time must also be monitored and recorded.

6. The instrumentation and DAS must monitor and record all fundamental engineering measurements in such a manner that the malfunctioning of any primary data channel would be readily detected upon observation of the data. In addition, the design and installation of the instrumentation must be carried out in such a manner that maintenance, if any, is minimal.

7. Input data must include the indoor and outdoor ambient conditions to allow the output of the system to be accurately determined as a function of these variables.

2.2 DATA SCANNING AND RECORDING REQUIREMENTS

The data collected in laboratory tests of heat pumps[1,2], and in a study made to determine the field performance of a residential heat pump in the Washington, D.C. area[4], were reviewed to establish the desired periods of scanning the raw data and for recording the raw data and the calculated results. As a result of this review, a "cycle period" was defined as the shortest time period meeting <u>any one</u> of the following conditions:

1. The end of one compressor-off period to the end of the next compressor-off period. (i.e. - The start-up of the compressor to the start-up of the compressor for the following cycle.)

2. The end of one compressor-off period to the end of the next defrost period.

3. The end of one defrost period to the end of the next compressor-off period. (i.e. - The end of one defrost period to the start-up of the compressor for the next compressor-on period.)

4. The end of one defrost period to the end of the next defrost period.

These definitions provide adequate "cyclic data" to be obtained during cooling, heating and defrost modes of operation. During long heating periods, a defrost period will end a cycle and start another, avoiding extremely long cycle periods. In very mild weather when the unit may not be used for a long period, the energy consumed by the crankcase heater will result in a cycle with a very low C.O.P.

In addition, the profiles of the continuous laboratory data on heat pump performance indicated that the intervals of scanning the transducers and recording the desired information could be expanded as the cycle progressed without losing the desired accuracy. This allowed the total test period that could be recorded on the storage media to be expanded, and also reduced the amount of data processing required. In this project the intervals of scanning the desired data were established as listed below:

1. every 10 seconds for the first 120 seconds of a cycle,

2. every 30 seconds for the next 240 seconds,

3. every 60 seconds for the next 360 seconds, and

4. every 300 seconds until the compressor turns off.

The intervals of scanning start over again (i.e., were reset to no. 1 above) under any of the following conditions:

1. The compressor starts indicating a new cycle.

2. The beginning of a defrost period.

3. The end of a defrost period.

The data from each scan, as listed above, were used in accumulating the input and output of the heat pump for each cycle. The cyclic data were recorded for each cycle. However, the data for the individual scans were recorded every tenth cycle to provide additional space on the recording media for the cyclic data.

In addition to the scan data and cyclic data, every half hour of each day, a series of data were recorded to provide additional data for the outdoor environment, the status of the heat pump, and supplemental data for checking the functioning of the fundamental data channels of the instrumentation and DAS.

Daily performance data were also accumulated and printed out and recorded at midnight each day. The daily performance data represent a summation of the cyclic data for the day. The daily performance data also include the average outdoor dry bulb temperature and the average outdoor dew point temperatures for the day. These average values were taken from the 48 half hour data scans taken throughout the day.

Samples of computer printouts of scan, cyclic, half-hour, and daily data recordings made from the raw data recorded in the field are shown in Appendix A. These printouts were made using a central microcomputer located at NBS and programmed to extract the specific type of data desired. The software for performing this task will be discussed in section 7.19.

2.3 DATA ACQUISITION SYSTEM SELECTION

The method of recording the desired data introduced several problems -- the more serious problem being cost. Numerous types of data recording systems were investigated. The cost of any manufactured data acquisition system (DAS) that would meet the requirements previously outlined far exceeded the money allocated. Systems utilizing cassette tapes for recording were also found to be out of the desired price range and required other components (e.g., tape control devices, buffers, timers, etc.) which would have to be designed and assembled in-house. However, when the cost and capabilities of off-the-shelf components to produce a microcomputer system with a floppy disk drive and controller were investigated, the capabilities of such a unit were found to exceed the requirements. In addition, the cost was well within the acceptable range and the Cromemco Z-2D disk computer system was selected over other systems because of cost, availability, versatility, and previous NBS in-house experience.

This microcomputer-based DAS was capable of: 1) continuously scanning the mode of operation of the unit; 2) recording, on the disk, raw and reduced data at the desired time increments; 3) accepting any standard portable console operating with a standard serial RS-232 interface (for the purpose of monitoring, updating software, changing disks, repairing and checking the functioning of all input channels, etc.); 4) accepting any manufacturer's component card with the standard S-100 bus; and 5) accepting a "bootstrapping" program that reinitiates the operating program in the event of a power failure. The actual DAS system configuration is described in more detail in the following section.

2.4 BLOCK DIAGRAM OF INSTRUMENTATION AND DATA ACQUISITION SYSTEMS

A simplified block diagram of the instrumentation and DAS systems is shown in figure 1. A brief description of the components of the system follows. A more detailed description of the instrumentation is given in section 3, while the DAS is described in more detail in section 5.

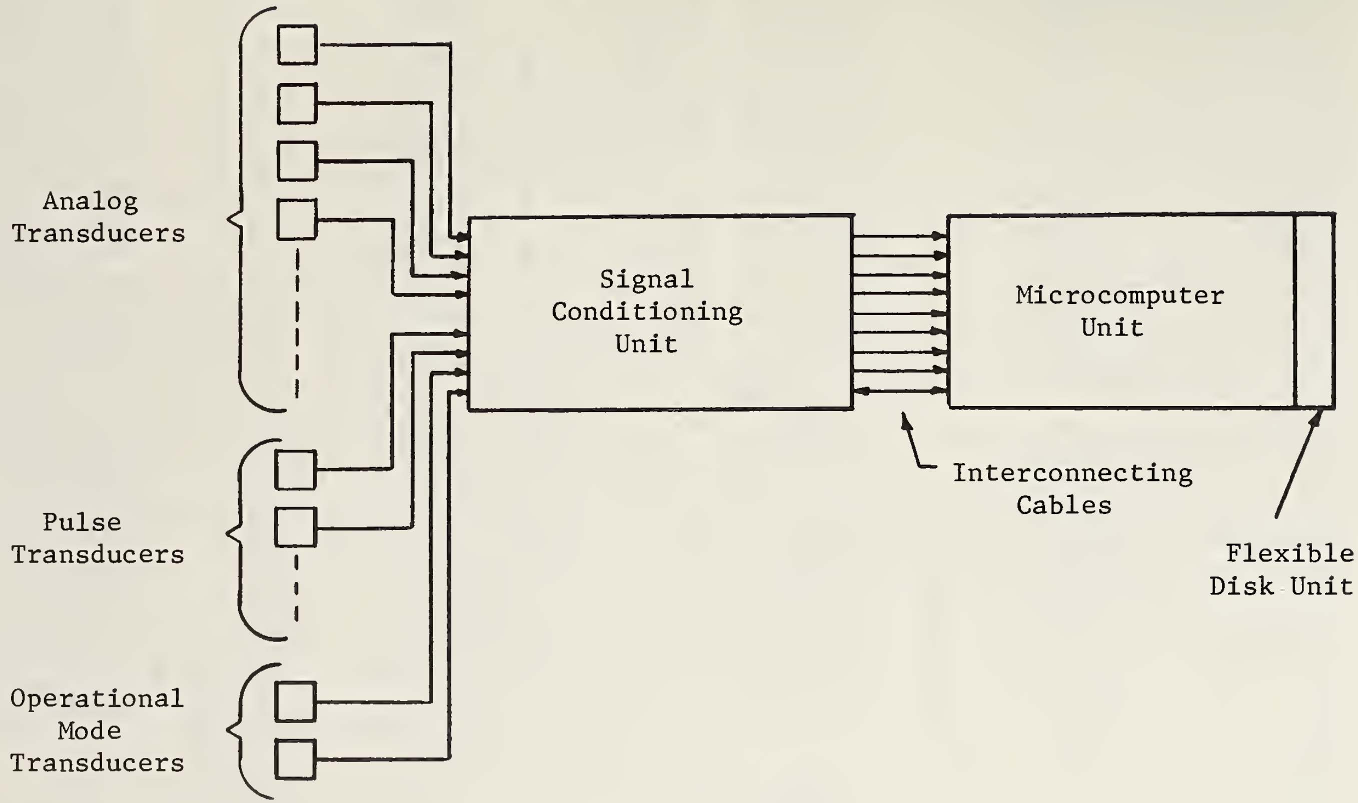


Figure 1. Simplified block diagram of instrumentation and data acquisition system used in heat pump field study

The analog transducers continually monitored temperatures, differential pressures, differential temperatures, dew points, and atmospheric pressure. Pulse-type transducers included the watt-hour meters and the reciprocating condensate pump metering the condensate from the indoor coil during the cooling season. The operation mode transducers continually monitored the compressor and outdoor fan operation and indicated the operational status of the heat pump, i.e., on, off or defrost.

The signal conditioning unit accepts the outputs of the various transducers and amplifies, attenuates, or otherwise conditions these signals to fall within the acceptance ranges of the microcomputer, while maintaining the maximum possible accuracy.

The microcomputer accepted the signals from the signal conditioning cards and made the required translations and calculations specified in the software. This unit also recorded the desired information on a single five inch floppy disk at the end of a scan, cycle, day, or 1/2 hour period, as the program directed. In addition, signals are sent back to the signal conditioning unit to reset the pulse-counting components at the end of a cycle. The microcomputer was accessible by connecting a portable terminal to monitor the functioning of the unit and all input channels. The disk was replaced periodically to avoid overflow and was capable of storing data for a period of 5 to 10 days, depending upon the activity of the unit.

2.5 ROLE OF A CENTRAL MICROCOMPUTER

A central microcomputer located at NBS was used to obtain the raw data from the floppy disks recorded in the field. The central computer was programmed to display on the monitor screen the desired data. This central computer was also equipped with a printer to provide hard copies of any desired data.

The central microcomputer is a Cromemco Z-2D disk computer systems with two 5 1/4-inch floppy disk drives, a 64K static memory, and a CPU board utilizing the Z-80 microprocessor. In addition to reducing the data recorded on the raw data disks printed in the field, this central unit allowed the raw data to be edited and the edited data recorded on a separate disk. This feature allowed the edited data to be limited to those desired for a specific analysis, thereby providing space for 5 to 10 disks of raw data on a single disk. Another primary feature of this editing stage is that the raw data disk was left unchanged for any further analysis. The software for reducing and editing the data from the raw data disks is described in sections 7.19 and 7.20.

3. MEASUREMENTS AND INSTRUMENTATION

3.1 AIR FLOW MEASUREMENT

Since these field tests were being performed on heat pumps in the homes of NBS employees and any modification to the unit required restoration, the use of nozzles, air straighteners with multiple pitot tubes, or other means of flow measurement which restricted air flow or required major modification to the ducts was not practical. Such techniques would not only increase the cost of this basic measurement, but would generally disrupt the normal air flow in the duct and increase the noise level of the system.

A pitot tube was located in the return duct at a position that would give the best possible flow conditions. Velocity measurements were made using an inclined manometer attached to pitot tube at the centers of 30-35 equal areas in the return duct. The pitot tube was mounted at the location in the duct which best approximated the average velocity of the total number of locations sampled. Since the speed of the indoor fan often changes between the heating and cooling modes of operation, another scan of the velocities in the return duct was made at the beginning of each season and the position of the pitot tube changed, if necessary.

One of the three field test units utilized two return ducts. In this case, the velocity profile was determined in each duct and a pitot tube was mounted at the appropriate point in each duct. Sections of tubing of equal length were brought from the static and velocity legs of each pitot tube to tees. The center outlet of the tees were then connected to the differential pressure transducer.

The use of the simple pitot tube, however, was complicated by the search for a reliable transducer to continually monitor the velocity pressure measured by the pitot tube. In general, the cost of the majority of transducers investigated exceeded the money allocated for all the instrumentation. However, a very stable and accurate device was found that was designed to operate from 0-10 inches (0-25.4 mm) of H_2O. The range originally selected for these field tests was from 0.01 to 0.20 inches (0.0254 to 0.508 mm) of H_2O. The specifications of the transducer located indicated a maximum error of $\pm$0.05% of the reading plus 0.005% of the full-scale reading. At 0.01 inches (0.0254 mm) this tolerance represented a maximum error of $\pm$0.00055 inches ($\pm$.0014 mm) of water, which was well within an acceptable range. Although the cost of these units exceeded that which had originally been allocated for this instrument, they were purchased because of the time and cost involved in designing and building a more suitable unit. The pressure transducers purchased were listed as Type 590D-10T-2Q1-VIX-4D manufactured by

Datametrics Inc.
340 Fordham Road
Wilmington, MA 01887.

These transducers were located in the field as close as possible to the pitot tube and in an inconspicuous place - such as on a board nailed across the two floor joists located above the pitot tube.

This transducer has an output of 0 to 10 VDC depending upon the differential pressure involved. Since the full-scale reading is for a differential pressure of 10 inches (254 mm) of H_2O, the output signal for the range encountered in the field was 0.01 to 0.20 volts. The input for this particular transducer is specified by the manufacturer to be +20 to 33 volts DC or 20 to 30 volts AC, 50-400 Hz. Neither of these voltage supplies were available from the DAS without adding other components. Since space in the DAS hindered the addition of other components, a simple bypass of a 12-volt regulator in the unit was made and the unit was powered directly from the 12-volt DC regulated supply in the DAS, after carefully reviewing the circuit diagram and consulting with an engineer at the manufacturer's plant.

The pressure transducers operated on the variable capacitance principle and utilized a well known bridge circuit and a balanced demodulator. As the ΔP changes, a diaphragm deflects and changes the capacitance which shifts the balance of the bridge. The resulting signal is demodulated and amplified to give a DC voltage directly proportional to the ΔP across the cell. All components of this transducer are shielded and the cell was corrosion resistant to all gases and liquid vapors found in air.

3.2 DEW POINT MEASUREMENT

The measurement of the quantity of moisture in the air passing over the indoor coil is needed to determine the mass flow rate of the air. The moisture in the air outside the residence is required to obtain correlations between the frosting of the outdoor coil and the ambient air conditions. Several types of moisture sensing devices were investigated. However, most of those found within a desirable cost range were found to be temperature dependent. Several had temperature sensors adjacent to the sensor and offered direct readouts with temperature compensation.

The characteristics of the Industrial Dew Point System offered by

Industrial Division
Yellow Springs Instrument Co., Inc.
Yellow Springs, OH 45387

appeared to meet the needs of this task. The Basic Dew Point Cell, Model 9400, and the Basic Transmitter (amplifier) Model No. 1394 were used. Dew point cells were located in the return duct and in an outdoor weather station mounted outside the residence from 8 to 12 feet (2.4 to 3.7 m) above the ground (see figure 4.6). The transmitters were mounted in the signal conditioning unit.

The output of these sensors is based on the crystal-liquid transition point of lithium chloride. The cell is constructed as a single unit approximately 3/8 inch (9.5 mm) in diameter and 3 3/4 inches (95.3 mm) in length. Bifilar heater electrodes were wound on a fiberglass wick surrounding a precision platinum resistance temperature detector (RTD). The wick was impregnated with LiCl, a hydroscopic salt, which becomes increasing conductive as it absorbs moisture from the surrounding atmosphere. When an AC voltage is applied to the bifilar heater windings, moisture evaporates from the wick until a heat-moisture equilibrium is reached. This temperature is sensed by the RTD whose output is related to the dew point of the air adjacent to the cell and is continually sensed by the signal conditioning unit.

In the event of a sudden change in the surrounding air or a power outage, the cell becomes excessively wet and the system automatically switches to the self-dry mode which de-energizes the bifilar windings and energizes the RTD to dry out the wick. When the wick has evaporated the excess moisture, the system automatically switches back to the normal mode of operation. This feature of the system was found to be of great value in the field tests where the equipment could not be under continuous surveillance.

When the dew-point cells were initially installed in the return duct of each system, they became excessively wet each time the indoor fan was energized. When this happened, the RTD often required a longer period to sufficiently dry out the cell and switch to normal operation than the total time the fan was operating. The manufacturer was consulted and suggested that the upstream outlets in the housing of the cell be shielded. After this minor modification was carried out, the units performed satisfactorily.

The dew point cells are designed to respond with a time constant of 0.9 minutes in still air. The response time in the return duct corresponded to a time constant of less than 1 minute after the upstream outlets were shielded. An initial series of calibration tests confirmed the specified 1 mV/°F (dew point temperature) output from the transmitter. However, minor offsets were required in the DC voltage output of the transmitter to reflect the correct value of the dew point.

3.3 DIRECT TEMPERATURE MEASUREMENTS

The temperatures of the air in the supply and return ducts and the outdoor air temperatures were continuously monitored. The return duct temperature was monitored to establish a base to be used in computing the temperature difference across the indoor coil. The latter was measured using a thermopile described in a following section. The actual temperature in both ducts also served as a check on the data recorded from the thermopile. These three temperatures were monitored using special thermistors designed to respond in a relatively linear fashion. The thermistors in the ducts were mounted on the supporting grid for the thermopile. The outdoor thermistor was mounted in the weather station outside the house. (See figure 9, page 30.)

A typical thermistor is designed using a metal oxide, and functions with a negative response to temperature. Generally, a typical thermistor is used in one leg of a Wheatstone bridge circuit. The bridge is arranged in such a manner that an increase in temperature, which results in a decrease in the resistance of that leg of the bridge containing the thermistor, will indicate a positive response. Unfortunately, the typical thermistor is nonlinear and the outputs require relatively complex mathematical expressions to relate the output signal to the correct temperature.

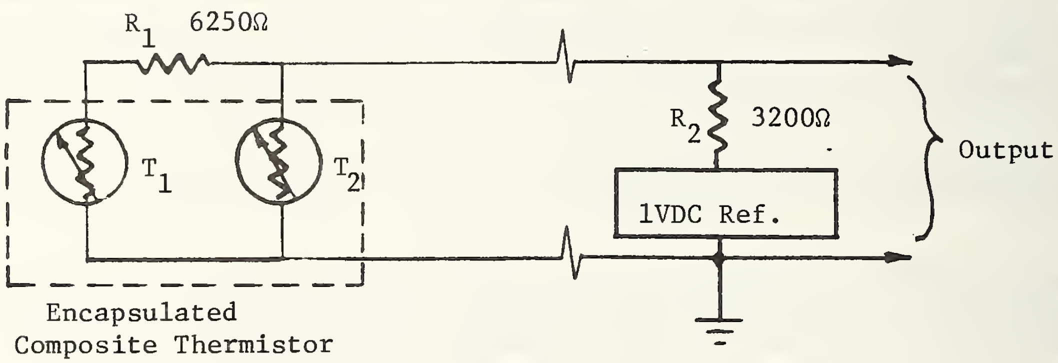
The composite thermistors used in this project were desiged with oxides of two metals, one mounted adjacent to the other, and encapsulated with three leads extending from the bead. Special metal film resistors are supplied with the units to complete the bridge circuit. By utilizing the different negative responses of the two oxides and inserting the corresponding resistors in the circuitry, the output was relatively linear. The manufacturer of this device was

Industrial Division
Yellow Springs Instrument Co.
Yellow Springs, OH 45387

and it was called a "Thermilinear Thermistor Network, YSI Part No. 44201."

The various components of the network are interchangeable with a tolerance of $\pm 0.27°F$ ($\pm 0.15°C$) from +32°F to 212°F (0°C to 100°C). The deviation from linearity was 0.36°F ($\pm 0.2°C$) over this entire range. These units can be used in a voltage mode or a resistance mode. To simplify the circuitry, these devices were used in the voltage mode for this project. The total cost of this entire "network" was only three times the cost of a typical nonshielded, nonlinear thermistor.

The circuit which is shown below had an output impedance of 10 MΩ.



The specifications state that the output signal E_{out}, is represented by the equation:

$$E_{out} = (+0.00297127\ E_{in})\ T + 0.03985\ E_{in}$$

where T is the temperature in °F and E_{in} was set equal to unity (1 volt).

Regardless, of its theoretical response, each thermistor was calibrated along with its conditioning circuit over the required range (or ranges) prior to installation in the field.

The response of these special thermistors is 10 seconds in still air, which is well within the required limits. To satisfy the manufacturer's specification that the resistor R_1 be mounted close to the encapsulated bead, the resistor and the ends of three leads of the thermistor were sealed in epoxy leaving the composite bead exposed without additional thermal mass. This resulted in a hermetically sealed transducer and resistor which were then mounted in the ducts and weather station. The resistor, R_2, which could be mounted up to 100 feet (30.5 m) from the thermisotr, was located in the signal conditioning unit, since the longest lead for this transducer was less than 50 feet (10.24 m) for any of the installations.

Thermistors are often abused when used in the voltage mode. The E_{in} voltage is applied without giving consideration to the self-heating errors that result. In the design of the circuitry for the thermistors used in this project, the value of 1 volt was selected to minimize any self-heating errors. The manufacturer's literature stated that a maximum of 2 volts could be applied without self-heating errors when the thermistor is immersed in stirred oil, where the dissipation constant is 8mW/°C. In air, the dissipation constant is less and the applied voltage must be reduced to avoid errors induced by self-heating. Heat sinks are often applied when using higher voltages; however, such techniques are not recommended when the medium of interest is air and a relatively short response time is important.

3.4 BAROMETRIC PRESSURE

The barometric pressure at each of the field installations was continuously monitored to compute the specific volume of the air passing over the coils and the humidity ratio at saturation. These quantities are directly related to the sensible heat released or absorbed by the indoor unit.

A suitable barometric pressure transducer and a transmitter were found available from:

Industrial Division
Yellow Springs Instrument Co., Inc.
Yellow Springs, OH 45387.

The YSI - Sostman Model 2014 transducer with a range of 27.0-31.5 inches (685.8-800.1 mm) of mercury was selected since it fell within the typical ranges of atmospheric pressures encountered in the area and altitudes of the field test units. A transmitter, Model 1354, was selected since the signal conditioning unit could conveniently house this single circuit board. The transmitter required a line voltage of 115 VAC and yields a linear output of 0-5 VDC that is proportional to the barometric pressure. The transducer monitors the barometric pressure by monitoring the motion of a diaphragm capsule. Pressure changes activate the diaphragm which moves a contact across a precision potentiometer. The specified accuracy of the potentiometer is $\pm 0.3\%$ of the range span. The resulting signal is amplified by the transmitter. Prior to field installation, each unit was calibrated using a mercury manometer.

It is of interest to note that the cost of the transmitter was less expensive than the transducer and could have been designed to be an integral part of the signal conditioning unit. However, the manufacturer's amplifier was purchased to reduce the complexity of the signal conditioning circuit. If a larger number of field units were to have been instrumented, the amplifier would probably have been included in the design of the signal conditioning unit in order to reduce the total cost of the instrumentation.

3.5 TEMPERATURE DIFFERENTIAL ACROSS THE INDOOR COIL

Thermopiles were used to determine the temperature change of the air as it passed through the heat exchanger of the indoor unit. In two of the three field units (nos. 1 and 2), the area of a supply and a return duct was divided into nine equal parts. A type T, copper-constantan thermocouple function (assembled in accordance with reference 5) was located in the center of each area. The supply and return thermocouple function were connected in series to form an 18 function thermopile. The section of the supply and return ducts in which the thermopile was mounted was carefully selected to avoid any direct radiation from the auxiliary heaters and to be in an area where the air flow was well mixed.

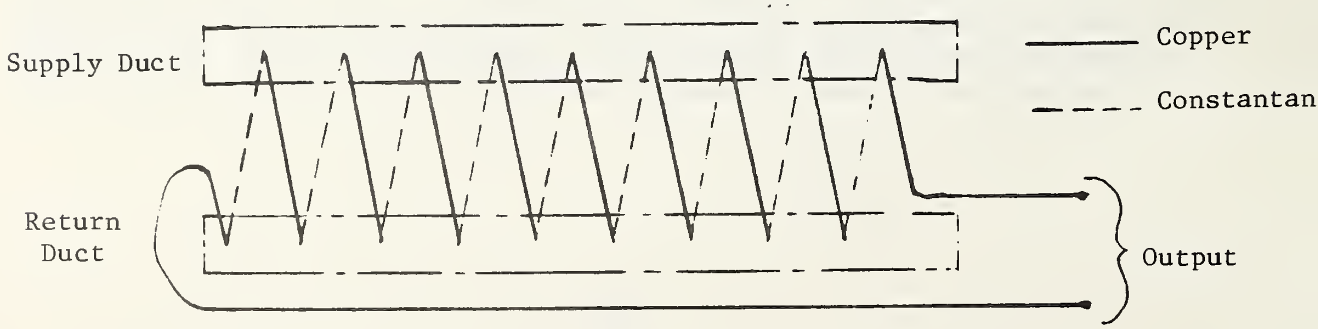
While these same procedures were followed in unit no. 3, a 40 function thermopile was used in this installation because of the relatively large area of the return plenum. In this unit, the air from the two return ducts joined a return plenum at the base of the indoor unit from opposite sides. The cross-sectional area in the plenum, above where the return ducts were attached, was divided into 20 equal areas and a thermocouple function placed at the center of each area. Twenty thermocouple functions were mounted in the major supply duct at centers of nine equal areas -- two thermocouple functions were placed at the center of each area with two additional functions located in the center of the duct. The supply and return thermocouple functions were connected in series to form a thermopile.

Although the temperature measurement procedure used for unit no. 3 was slightly different from the other units, numerous tests indicated that there were no significant measurement errors. A special transducer channel was also added to the instrumentation of unit no. 3 to continually monitor the temperature of the air in a smaller secondary supply duct. A review of the samples of recorded data for unit no. 3 indicated that the temperatures in the primary and secondary supply ducts were virtually the same under all operating conditions.

All thermocouples were mounted in the ducts with equal lengths of lead wire exposed to the air within the duct to avoid errors induced by the thermal conductivity of the thermocouple wire.

The output from each thermopile was checked at the maximum ΔT and at ΔT equal to zero for both the heating and cooling mode of operation. The procedure for changing the polarity and amplifiers when the heat pumps shifted from heating to cooling is described in the section on the signal conditioning unit.

A schematic diagram of the thermopile is shown below:



3.6 ELECTRICAL ENERGY INPUT MEASUREMENTS

The measurement of electrical energy consumption of the field test units was made using watt-hour meters with pulse initiators. The pulse initiator is activated by a gearing mechanism connected to the meter disk. The pulse initiator in the watt-hour meters selected had mercury-wetted bi-stable output relays which magnetically latch in one position and did not change state until the initiator circuitry "told" the relay to change state. The mercury-wetted contacts eliminate the majority of contact bounce, which is often a serious problem in pulse-counting circuits. Since the circuitry design for the signal conditioning unit was already available for open-close type pulse-counting mechanisms, the pulse initiator was used with a two-wire connection that employed the common and either of the other two contacts connections. Both 1 watt-hour/pulse and 20 watt-hour/pulse meters were used in this field study.

The outdoor unit at installations no. 1 and no. 3 and the outdoor and compressor units at installation no. 2 were wired on a separate circuit and the more sensitive meters (1 watt-hour/pulse) were employed. The indoor units were on two separate circuits for units no. 1 and no. 2 and on a single circuit for unit no. 3. Since these circuits were wired to carry the auxiliary heating and the indoor fan loads, the meters yielding 20 watt-hours/pulse were used to avoid excess overflow of the pulse-counting registers in the signal conditioning unit.

All wiring alterations, including the installation of the watt-hour meters were performed after permits were obtained from the local county office by a registered electrician. All work was inspected by the county electrical inspector and a seal of approval was placed on the service box at each of the three residences.

The signals from the pulse initiators were summed in the signal conditioning units and recorded at the end of each scan period. This is discussed in more detail in the section 4.2.

3.7 CONDENSATE METERING

The condensate from the indoor coils was measured during the cooling season to determine the latent energy output of the system for each cycle. Previous laboratory tests indicated that typical flow rates would vary from 0 to slightly over one gallon (0 to > .00379 m^3) per hour depending upon the temperature and moisture content of the air passing over the coils and the cooling capacity of the heat pump. The relatively small heads available at the condensate drains in the field did not permit a gravitational measurement system to be used. A search for an inexpensive, reliable, and accurate positive displacement pump led to the use of a solenoid metering pump manufactured by:

Valcor Engineering Corp.
365 Carnegie Ave.
Kenilworth, NY 07033.

The series 500 pump (with the Buna-N elastomer) was recommended by the manufacturer. A reservoir was built with a float that activates a timer connected to the small pump. The timer served two purposes -- it allowed the pulse rate to be adjusted from 3 to 120 cycles/minute, and, it protected the pump from overload since the maximum duty cycle of the pump was 50 percent with a maximum 6 seconds "on" time. Figure 2 is a photograph of the heat exchanger unit showing the reservoir under the return duct.

The original design was made to allow an upper float to activate the timer when the reservoir was full and the lower float to stop the pump when the reservoir was empty. A false "full" signal was sent to the timer at the end of each heat pump cycle period by a relay connected in the circuit. Two basic problems surfaced after these units were calibrated and installed in the field. First, the capacity of the reservoir was too large. During the periods of heavy cooling loads, the pump might have only five minutes to empty the reservoir while the compressor was off. The pump had been carefully calibrated to displace 1 ml of condensate per stroke and at 120 strokes per minute, the pump could only displace 600 ml during a 5 minute compressor-off period. The reservoir was, however, designed with a capacity of approximately 4 liters. This problem was solved by shorting out the upper reservoir float and letting the overriding signal from the lower float turn the pump on and off. This solution had one minor disadvantage. It reduced the accuracy of this measurement from < 1 percent to approximately 1.5 percent. However, its major advantage was that it allowed the pump to displace and monitor the condensate as it was released from the coil. This allowed the pump to perform the vast majority of its work while the heat pump was in operation and the "ticking" sound of the pump could not be heard by the people in their homes. Using the original design, several minor complaints had been voiced regarding the noise of the pump. Operating in the modified mode, this problem was eliminated and the pump, typically, displaced from 5 to 15 ml of condensate per "burst" of pulses, depending on the rate of condensate flow and the sensitivity of the lower float.

The second basic problem that surfaced was an unexpected reaction of the Buna-N "O" ring on the piston of the pump with the condensate. Within three days after the pumps were installed in the field, the calibrations of the pumps became erratic. When the pumps were examined, the "O" rings on the piston were found to be enlarged and a black deposit was found on the inner surface of the stainless steel cylinders of the pump. The manufacturer, when contacted, suggested that condensate was not "normal" water, but water that made an excellent electrolyte for the Buna-N to react with ions in the water from the copper fittings in our reservoir and the aluminum coil and fins of the indoor unit. The Buna-N rubber "O" rings were replaced with ethylene propylene "O" rings, the deposit removed from the cylinder of the pump and the pumps were recalibrated. No further problems of this kind were encountered with the new "O" rings.

After the above problems were solved, the simple solenoid pumps functioned very well. The displacement was checked as often as once each week until adequate confidence was obtained in the calibration of the pump.

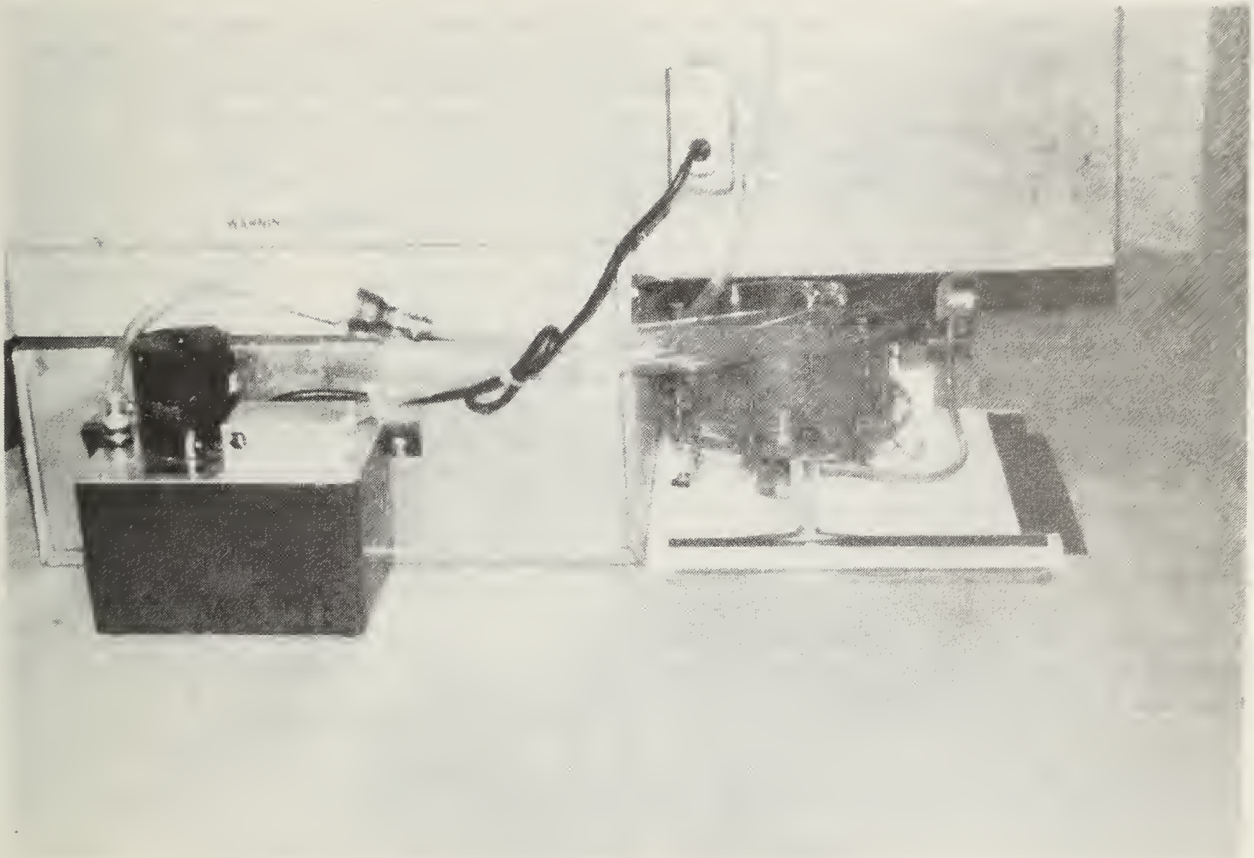


Figure 2. Condensate metering device shown under the return air duct on the right side of one of the field test units. The condensate was diverted into the metering reservoir and then forced by the positive displacement pump into the original sump pump installed by the heat pump contractor. The positive displacement pump can be seen on the right side of the metering reservoir.

The slight loss in accuracy that was observed when the pump was controlled only by the bottom float was related to the float displacing less than 1 ml before being turned off. That is, when the float started the timer, the pump would displace 1 ml per stroke until the float stopped the timer. However, occasionally the float would cut off the power to the pump while the pump was in the process of displacing its last ml of condensate for that "burst." If the float cut the power to the pump within a few msec after the pump was energized, the pump would not complete its stroke, but the stroke would be registered by the counting circuit. Fortunately, this did not occur very often and its effect is not believed to be significant.

Plastic tubing was placed in the systems to allow the condensate to overflow from the reservoirs into the original drains in the event of equipment failure. The lack of the head available in the condensate drains required a watertight system to be utilized. The reservoir was vented by an overflow line leading to the normal condensate drain. If the condensate pumping network should fail, the reservoir would fill and the condensate from the coils would be released through the overflow line. Although heads as low as 3 inches (76.2 mm) were encountered in the field to force the condensate out the overflow lines from the reservoir, each overflow system was tested and found to work satisfactorily.

The pump strokes were counted by the signal conditioning unit. In order to avoid typical problems caused by inductive circuits, an optical coupler was used which is described in the following section.

3.8 DETECTORS FOR THE MODE OF OPERATION OF THE HEAT PUMPS

The definitions of the cycle period and scan periods presented in section 2.2 indicate the need for the DAS to continuously monitor the mode of operation of the heat pump. In addition to the heating and cooling functions of the unit, which were detected by the DAS using the position of a manually controlled switch, there are three normal modes of operation: the "on" mode (compressor on, outdoor fan on); the "defrost" mode (compressor on, outdoor fan off); and the "off" mode (compressor off, outdoor fan off). To avoid any possible interference with the heat pump controls and to prevent possible errors in signal detection, isolation transformers were used for connection with the heat pump. One transformer was placed in parallel with the power lines to the compressor and a second transformer was placed in parallel with the power lines to the outdoor fan. In each case the transformer consisted of a 230 VAC primary coil and a 24 VAC secondary coil. These are relatively small transformers with an isolation resistance of 1500 VAC and an output rating of about 15 VA. With these transformers in place, adequate power for signals was available when the respective power lines were energized.

Since a DC signal was required by the signal conditioning unit, an optical coupler was used. A schematic diagram of the circuit is shown in figure 3. When the unit being monitored is energized, the secondary side of the isolation transformer yields 24 VAC. The current from this potential source is limited by the resistor R_1, resulting in a maximum or peak forward current of approximately 2 ma through the solid-state lamp in the opto-coupler. The light from this lamp energized the Darlington amplifier, which begins to discharge the capacitor, C_1, and, within approximately 50 msec, drops the potential at the signal point to less than 1.5 volts. During the reverse portion of the 60 Hz cycle, the reverse current passes through

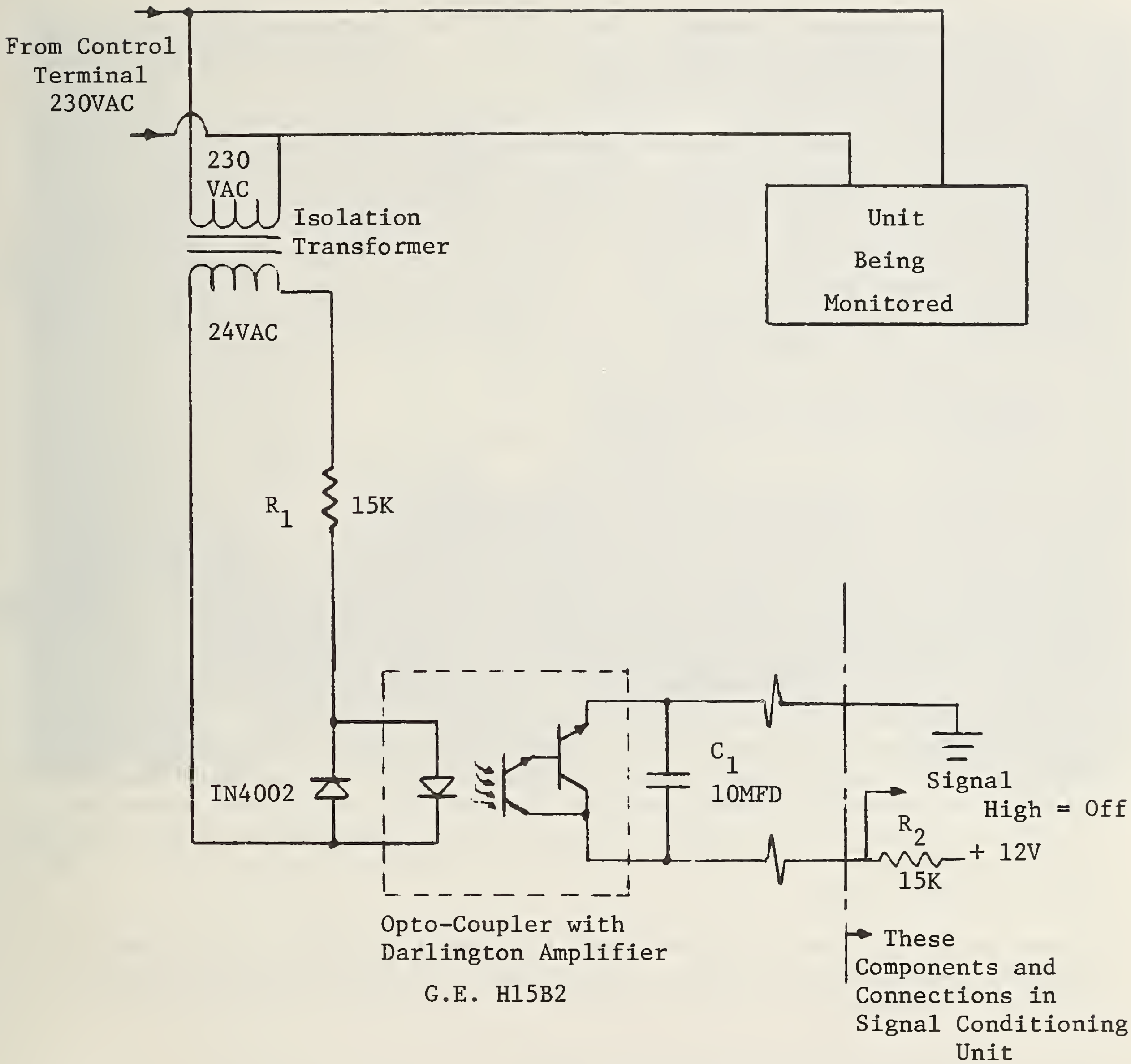


Figure 3. Opto-coupler circuit used in monitoring compressor and outdoor fan

the external diode and the reverse voltage across the solid-state light remains at approximately 0.5 V, which is well below the maximum rating. When the power line to the unit being monitored is de-energized, the potential at the signal point rises to +12 V within approximately 0.5 seconds. This provides a high signal level for the "off" mode and a low signal level for the "on" mode.

This same basic circuit was used in counting the number of times the condensate pump was activated. The only differences were that the isolation transformer was omitted and the value of R_1 was increased to 68K. This allows the opto-coupler and the reverse diode to be connected across the solenoid pump's 115 VAC power lines. The 68K resistance reduced the current and the reverse potential was limited by the diode. An isolation transformer was not required in this case since all components (except the pump itself) were located within a grounded enclosure.

4. DESCRIPTION OF SIGNAL CONDITIONING UNIT

The purpose of the signal conditioning unit, mentioned in section 2.4 and shown in the block diagram in figure 1, was to receive the signals from the various transducers and condition these signals to fall within ranges that are acceptable to the microcomputer and allow the signals to be processed and recorded with the maximum possible accuracy. The necessary power supplies, and the components of several transducers already described (e.g., transmitters for the dew-point sensors and transmitter for the barometric pressure transducer) are contained in this unit. However, the primary purpose of the unit is to house the specially designed printed circuit boards used to condition the signals from all the input channels (including the mode of operation) and present the conditioned signal to the microcomputer upon request. Figure 4 is a photograph of the electrical meters, the signal conditioning unit, and the microcomputer in one of the field installations.

There are three types of these specially designed printed circuit boards: analog, counting, and encoder. Each type is described briefly in the following sections.

4.1 ANALOG SIGNAL CONDITIONING CIRCUITS

The signals from all analog transducers were conditioned to fall within the input range of the A/D (analog to digital) converter in the micro-computer. The microcomputer requires all analog signals to be converted into digital signals for processing. The A/D converter's signal input range was from -2.56 to +2.54 volts and was divided into 256 increments (20 millivolts per division). The signals from all the analog-type transducers used required amplifying or attenuating, and offsetting to fall within this required range. The analog signal conditioning circuits were designed to yield an output voltage from -2.50 to +2.50 VDC over the expected range of the input signal. By using these limits, a 5-volt swing was available for the normal operating range of the transducer. When the output of the transducer was just outside the normal range, 60 millivolts or 3 increments were still available for recording and processing values below the normal range and 40-millivolts or 2 increments were available on the high end of the scale. By establishing limits in this fashion, a console

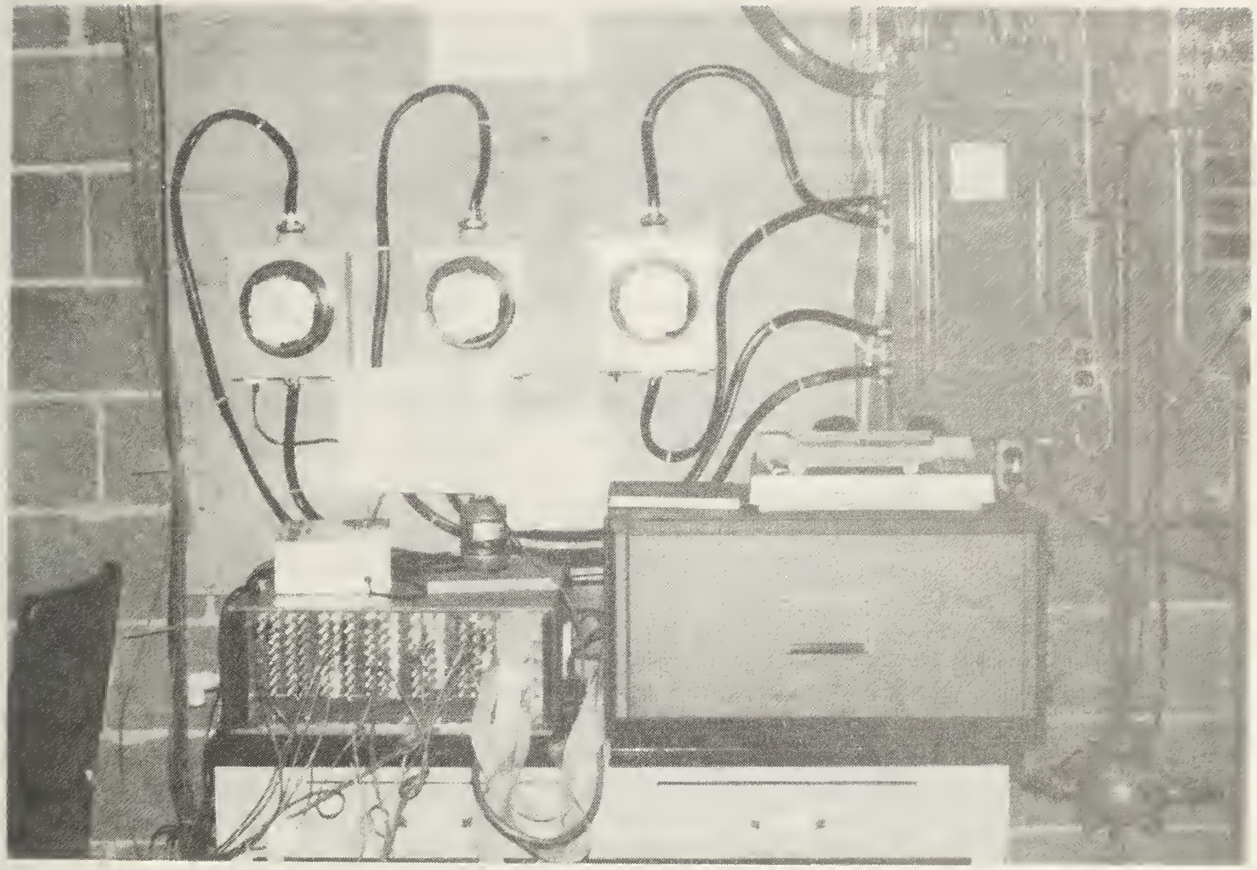


Figure 4. A photograph showing the signal conditioning unit on the left side of the supporting table with the microcomputer on the right. The floppy disk drive is located on the right side of the microcomputer and is not visible. A portable terminal is shown on the top of the microcomputer. The three pulsing watt-hour meters used for this project are mounted on the wall in the background. The barometric pressure transducer and the condensate pump control box are resting on the top of the signal conditioning unit.

printout of -2.56 volts or +2.54 volts generally indicated either a malfunctioning input channel or a channel which was not activated (such as the output from the pitot tube when the indoor fan was off).

An instrumentation amplifier AD521, made by Analog Devices, was selected for this task primarily because of its availability, wide range of gain, linearity, stability, high input impedance, and built-in output offset trim and gain trim features. The specifications for this amplifier are given in reference 6. Since only five basic conditioning circuits were required, the analog conditioning circuit board was designed to accept any one of the input signals and condition it in one of four different areas on each board. Table 1 lists each of the channels in the signal conditioning unit.

The five basic conditioning circuits and their applications are discussed in the following subsections.

4.1.1 AIR FLOW

As described previously, the velocity of the air flow was determined by a pitot tube mounted in the return air duct. The outputs of the pitot tube were connected by flexible tubing to a differential pressure transducer/amplifier. The output of this amplifier was from 0 to 10 VDC for differential pressure from 0 to 10 inches (0 to 254 mm) of H_2O. Since the range of interest in this study was from 0.01 to 0.20 inches (0.254 to 5.08 mm) of H_2O, the output from the differential pressure amplifier was approximately 10 to 200 millivolts. This signal required amplifying, offsetting, and, because of minor turbulence in the air stream, damping. The circuit which was designed to perform these functions is shown in figure 5. Each pitot tube, pressure differential amplifier, and analog signal conditioning circuit was calibrated as a matched set using a laboratory air duct and an inclined manometer prior to installation in the field. The gain trim and offset features of the AD521 allowed the output of each network to be adjusted to provide a signal from -2.50 V to +2.50 V over the velocity range of interest.

4.1.2 DIFFERENTIAL TEMPERATURE

The instrumentation amplifer (AD521) was used in a differential mode to amplify the "floating" output of the thermopile. Although this is a relatively simple application of the instrumentation amplifier, several requirements should be noted. First, a return path must be provided for the bias currents of the amplifier. In this design, 2MΩ resistors were placed at both inputs to provide this path[6]. Second, the amplitude of the differential input range is greater in the heating mode than in the cooling mode. To maintain maximum accuracy, two separate boards were assembled and calibrated. Each board had an instrumentation amplifier for the differential temperature circuit and the supply duct temperature, and the circuits were basically the same except for the gain resistor R_g. When the heating and cooling seasons changed, the boards were exchanged. The differential temperature circuits were designed for a 0 to 130°F (0 to 72.7°C) temperature differential in the heating season and a 0 to 50°F (0 to 27.7°C) temperature differential in the cooling season. In addition to exchanging the boards as the season changed, the polarity of the leads from the thermopile was reversed to maintain the same relative potential at the input pins of the instrumentation amplifer. The difference in the supply duct temperature circuit is discussed in the next subsection. The signal conditioning circuit for the thermopile is shown in figure 6. The values of the gain resistor R_g for the units using the 18-function thermopile was 510 ohms for the heating season board and 180 ohms for the cooling season board. The values of R_g for the heating and cooling boards for the unit with 40-function thermopile were 1100 ohms and 340 ohms, respectively.

4.1.3 DIRECT TEMPERATURE MEASUREMENT SIGNAL CONDITIONING

The characteristics of the linear thermistor network used for direct temperature measurements were described in section 3.3. The complete circuit, consisting of the thermistor "network" and the signal conditioning amplifier, is shown in figure 7.

The printed circuit board containing the thermopile conditioning circuit also contains one of the circuits shown in figure 7 for heat pumps no. 1 and 2, and two of these circuits for heat pump no. 3 since the latter has two supply-duct temperature transducers. As discussed previously, there are actually two "seasonal" printed circuit boards-- one designed to function with a temperature range from 50°F to 180°F (10°C to 82.2°C) for the supply ducts during the heating season and the other designed for a supply temperature range from 45°F to 95°F (72°C to 35°C) during the cooling season. The value of R_g was varied depending upon the gain required for the temperature range covered by the transducer.

A separate printed circuit board contained two additional temperature signal conditioning circuits to handle the return duct temperature, which ranges from 50°F to 95°F (10°C to 35°C), and the outdoor temperature which ranges from 0°F to 100°F (-17.8°C to 37.8°C).

The thermistor and the associated signal conditioning circuits were calibrated in the laboratory prior to installation in the field. The values of R_g were selected and the null and gain trim control potentiometers were adjusted to function within the intended range of the thermistor. The linearity of each unit was also checked and found to fall within the manufacturer's specifications.

A future modification that would improve the design of the temperature conditioning circuit would be to replace the divider used to obtain the 1 volt reference voltage with a voltage regulator similar to the LM317. This would establish a fixed reference voltage for these units and make it independent of possible shifts in the 12 V supply voltage and the current in the thermistor networks.

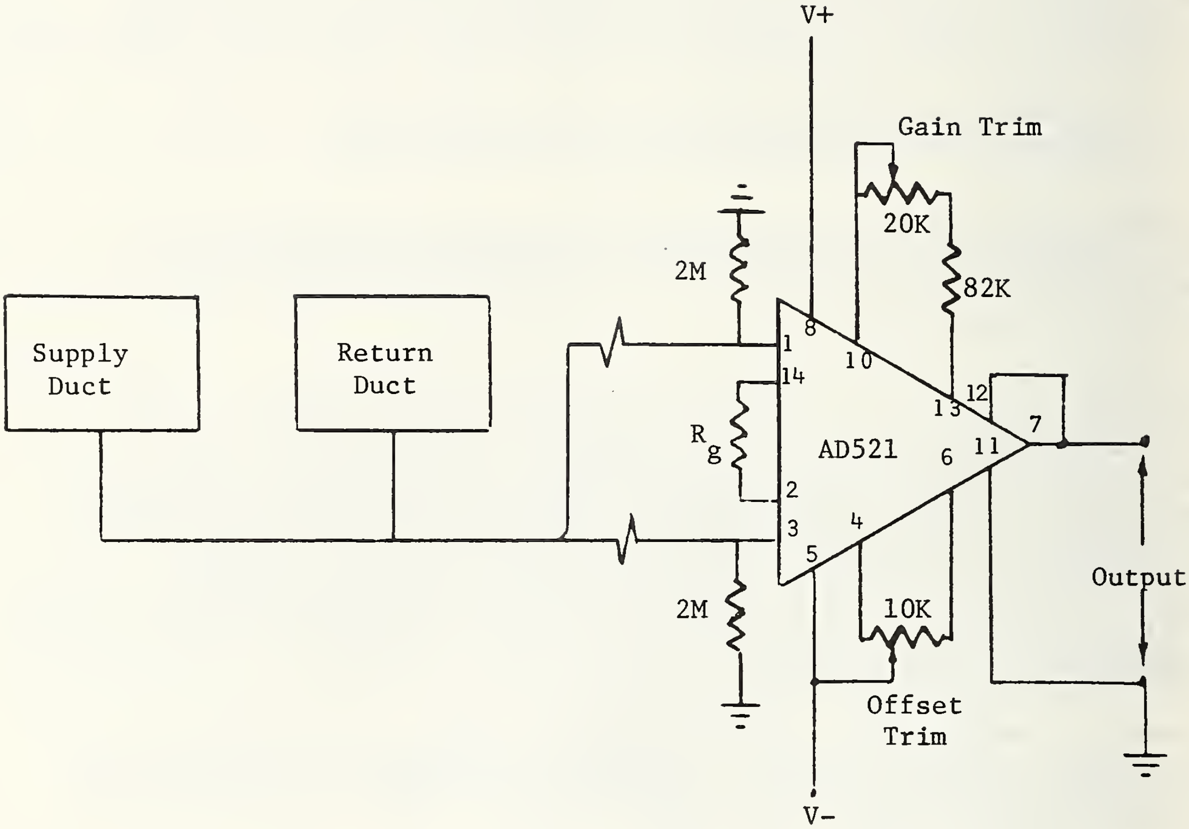


Figure 6. Thermopile signal conditioning circuit

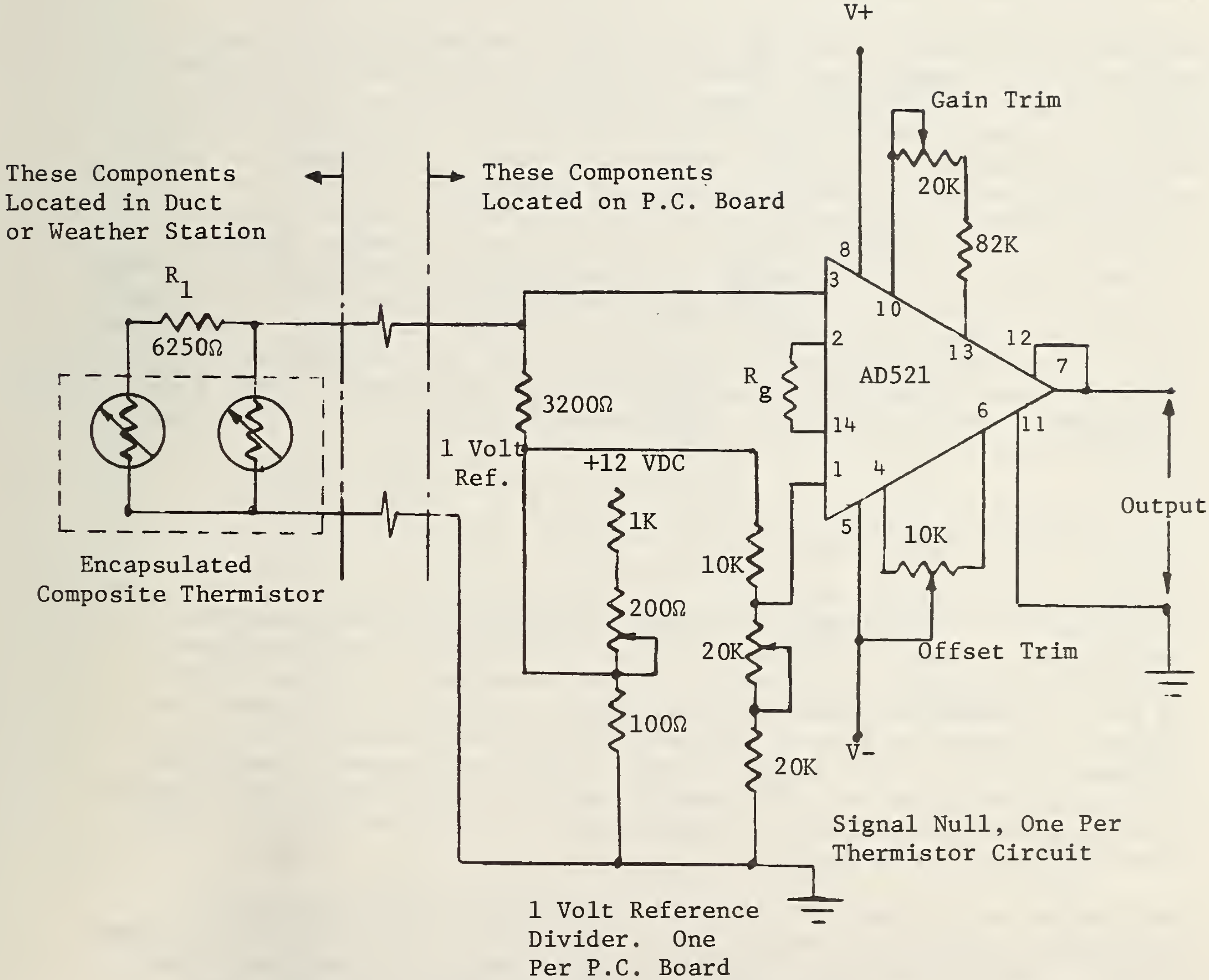


Figure 7. Direct temperature measurement
signal conditioning circuit

4.1.4 DEW-POINT MEASUREMENT SIGNAL CONDITIONING

The output signal of the dew point transmitters mounted within the signal conditioning unit was amplified and offset by the signal conditioning circuit shown in figure 8. The output of the transmitters was approximately 1 millivolt per °F dew point, as specified by the manufacturer.

One dew-point sensor was mounted in a weather station outside of each residence. The signal conditioning circuit for this sensor was designed and calibrated to produce an output voltage of -2.50 V to +2.50 V for a dew-point range of -40°F to 90°F (-40°C to 32.2°C). The second dew-point sensor was mounted in the return duct and its signal conditioning circuit was designed and calibrated to produce the same output voltage span for a dew-point range of 20°F to 90°F (-6.7°C to 32.2°C). Although a complete calibration over the entire range of each unit was not deemed necessary or practical, a sufficient number of dew-point levels were used to determine the slope and check the linearity of all sensors and conditioning circuits at dew-point temperatures above 35°F (1.7°C). The offset trim potentiometer adjacent to the instrumentation amplifier allowed the necessary offsets to be performed without additional balancing of components. Figure 9 is a photograph of the outdoor unit and the weather station at one of the field installations.

4.1.5 BAROMETRIC PRESSURE

The signal from the barometric pressure transducer and transmitter was conditioned as shown in figure 10. Since the output of the transmitter was 0 to 5 volts for the full range of the transducer, the input signal to the instrumentation amplifier was offset using the circuit shown. An alternative design to improve this offsetting circuit would be to use a voltage regulator on the negative input of the instrumentation amplifier and use the amplifier in a differential mode. This technique would require less components and retain the input potential within the +10 volt limits of the differential amplifier.

The simple, but precise, design of the barometric pressure transducer encourages combining the amplification circuit with the signal conditioning circuit. However, prior to the time of procurement, the characteristics of the transducer were undetermined and the transmitter was purchased primarily to protect the transducer from possible damage by the signal conditioning unit.

4.2 PULSE COUNTING CIRCUITS

The pulse counting circuits in the signal conditioning unit were designed to allow the pulses from the watt-hour meters and the condensate pump to be independently counted and to be transmitted to the microcomputer on demand. Provisions were made to reset the count on each counting circuit to zero by a reset signal from the microcomputer. A switch was also mounted on the board to allow the counters to be reset manually during testing of the printed circuit boards in the laboratory.

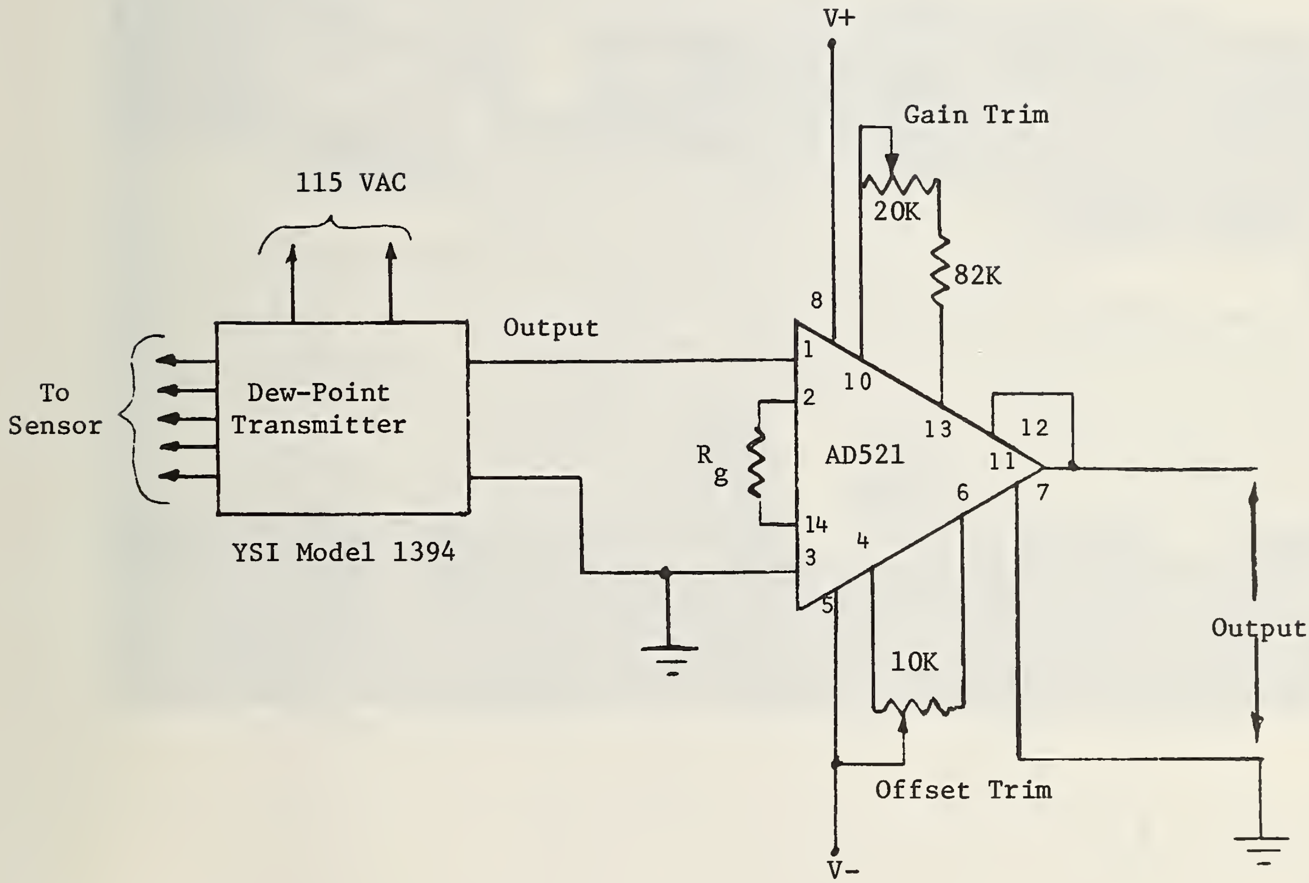


Figure 8. Dew-point measurement signal conditioning circuit

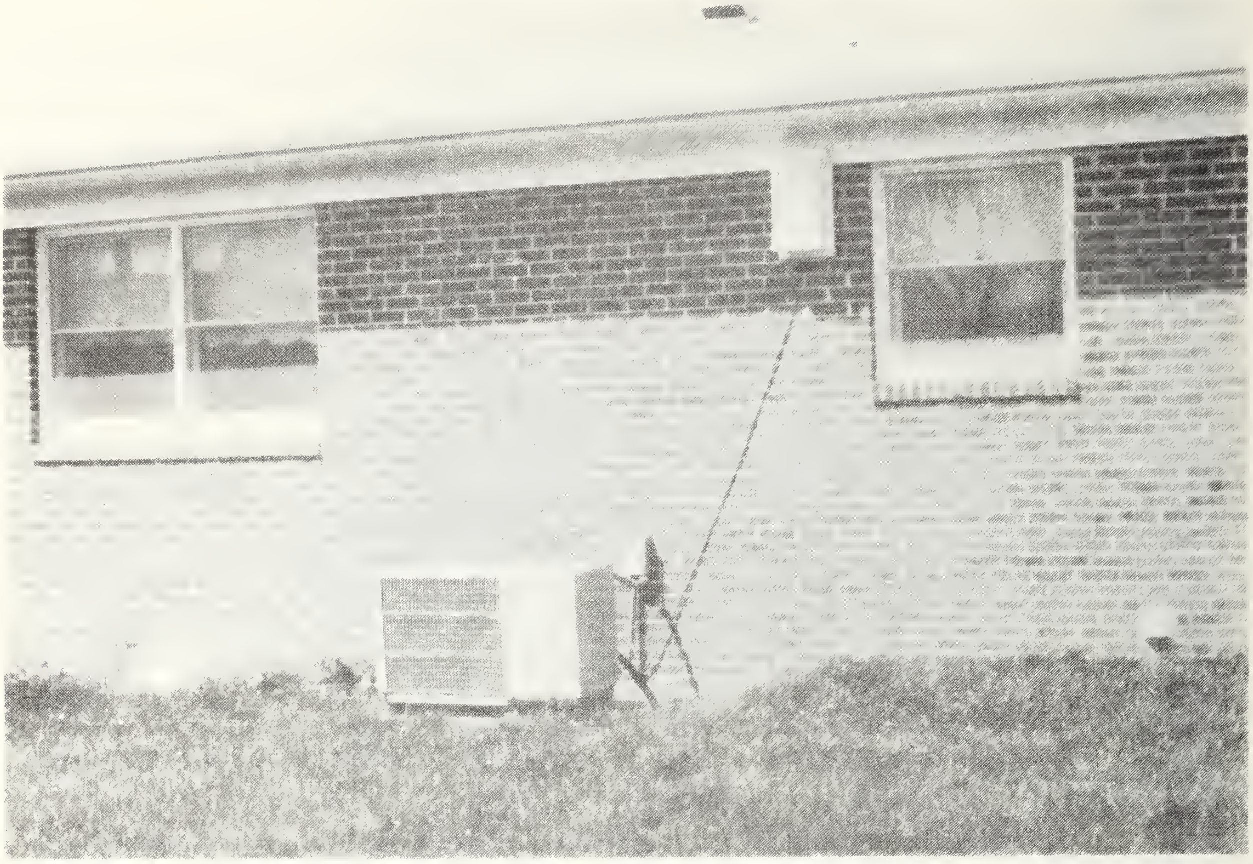


Figure 9. The outdoor unit at one of the field test sites. The weather station containing a dew point sensor and a thermistor is shown mounted on the fascia board on the house. The isolation transformers for energizing the opto-couplers monitoring the compressor and outdoor fan are mounted inside the outdoor unit's housing.

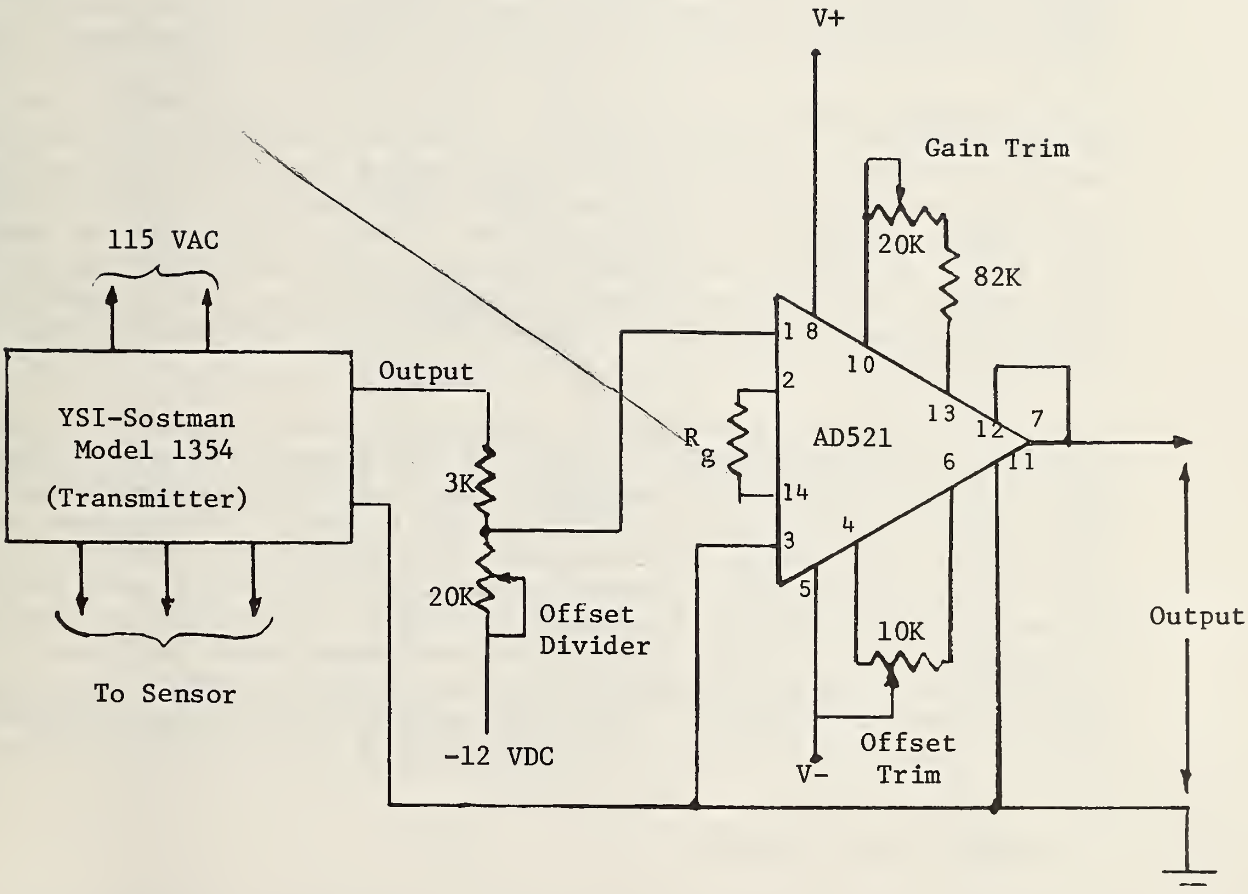


Figure 10. Barometric pressure signal conditioning circuit

A block diagram of the pulse-counting circuit is shown in figure 11. The contacts in the watt-hour meter or the transistors in the opto-couplers cause the amplifier to go into saturation and pulse the one-shot monostable multivibrator to generate a square pulse for a predetermined time period. This time period must be large enough to prevent contact bounce from causing more than one count to be registered per contact closure, but small enough to allow the fastest input pulse rate to be counted. This was not a serious conflict in the application of these circuits for this project, since the time period of the possible contact bounce was much shorter than the time between pulses from the watt-hour meters or condensate pump.

A CMOS 12-stage binary counter/divider receives the pulse from the one-shot multivibrator. In this project, the divider was not used in any of the pulse counting circuits and the pulse passed through the divider, which was preset at unity, and registered directly on the 12 bit binary counter. This counter is capable of registering up to 4095 pulses, with the 4096th pulse simply resetting the counter to zero. If the counter should be reset by this type of overflow during normal operation, the software in the microcomputer was designed to automatically add 4096 to the count recorded in memory.

At the end of each scan period, the binary count was recorded by the microcomputer. The microcomputer obtained the binary count through 12 parallel lines from the counter after passing through buffers whose purpose was to avoid disturbing the counter and provide the necessary driving power to register the count in the microcomputer. The counter was reset to zero by a reset pulse produced by the microcomputer at the end of each cycle. Further details of this circuit are not presented since numerous combinations of components can be used to function in the manner described. Each printed circuit board designed for this task contained two of the counting circuits shown in figure 11. Additional details on the 12-bit parallel transmission of the count to the microcomputer are discussed in section 5.4.

4.3 ENCODER CIRCUIT BOARD

When the signal conditioning unit was first designed, the number of interrupt signals from a heat pump which would be used in processing and recording the data was not known. Therefore, an encoder board was designed which allowed up to eight independent interrupt signals to be buffered into the microcomputer. However, the final design for the instrumentation and software allowed the microcomputer to function with only two interrupt signals. These signals, which were from the opto-couplers on the compressor and the outside fan, indicate the mode of operation of the unit (i.e., off, on, defrost). Since there were only two signals, they were simply passed directly through the encoder board to the microcomputer. A schematic of the encoder circuit board is shown in figure 12 since future studies might require the use of additional interrupt signals and the encoder board shown will allow up to six more circuits to be added without further design changes.

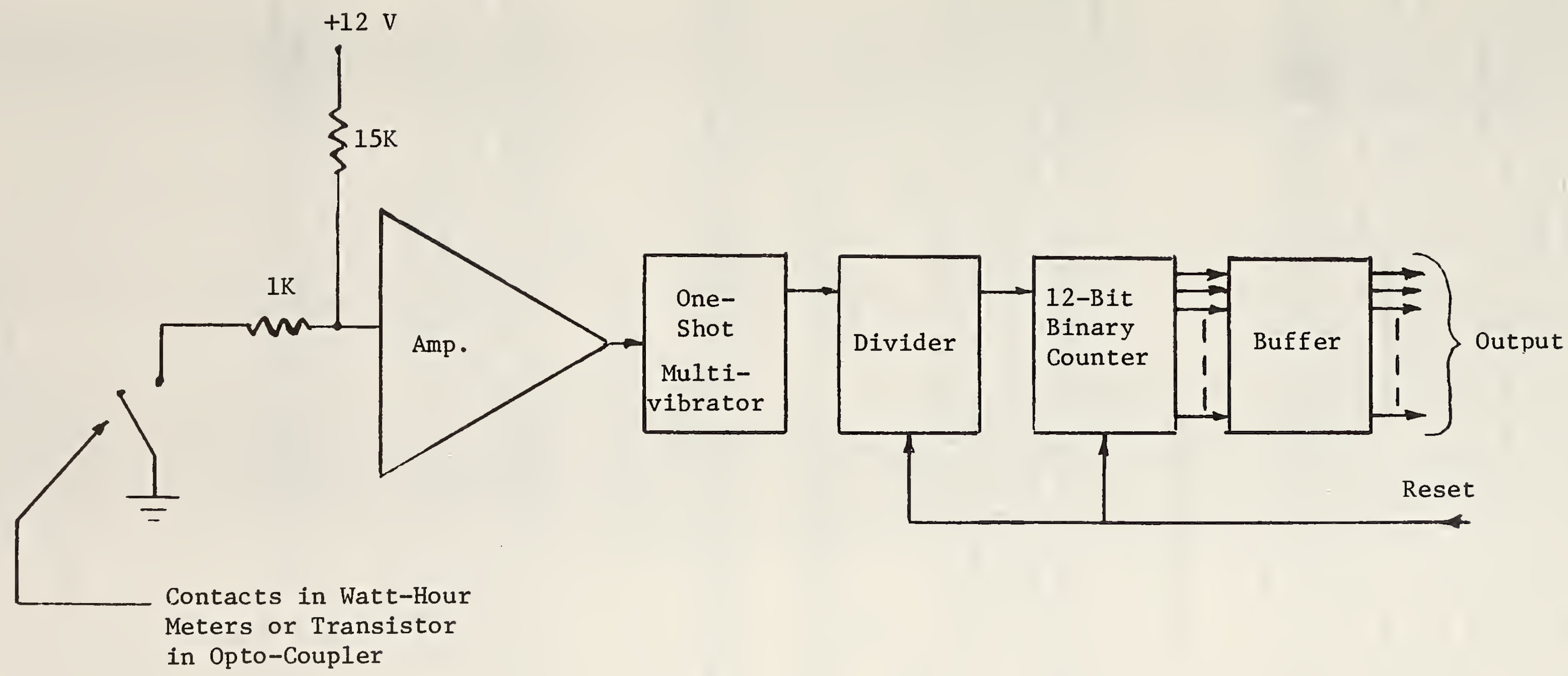


Figure 11. Block diagram of binary counting circuit

The encoder board shown in figure 12 receives any one of the eight external interrupt signals as shown and places the corresponding operational amplifier in saturation. The output of the operational amplifier passes through a buffer to the data bus of the microcomputer. This same signal is also routed to a parity-checking integrated circuit. When the parity of the input changes (even to odd or odd to even) the dual flip-flop circuit operates accordingly and a signal is generated by a NOR circuit which is routed to the interrupt terminal in the microprocessor, which then scans the data bus and takes the appropriate action. A signal is returned from the microprocessor to the voltage shifter which resets the dual flip-flops and awaits another change from the external interrupt signal generating sources. Up to eight interrupt signal sources can be monitored by this circuit.

5. DESCRIPTION OF THE MICROCOMPUTER

A brief description of the microcomputer used to monitor, process and record the desired data was given in section 2.3 where it was mentioned that the unit was capable of accepting any manufacturers' component card designed for the S-100 bus. A block diagram of the microcomputer used and its various components is shown in figure 13. In this section, the purpose and method of operation of the individual components in the microcomputer will be discussed.

5.1 HOUSING THE BASIC COMPONENTS AND "MOTHER" BOARD

The basic "black box" used in the field for processing and recording data is listed as a Cromemco Z-2D Disk Computer System manufactured by:

Cromemco Inc.
280 Bernardo Ave.
Mountain View, CA 94040

This unit consisted of an S-100 bus mother board capable of holding a total of 21 circuit boards. The power supplies, cooling fan, reset and power switches, single disk drive, 4-MHz CPU card (utilizing the Z-80A microprocessor), the disk controller board (capable of expansion and standard RS-232 serial interface), and other standard hardware items were obtained as part of this basic unit.

Six additional boards were procured separately and added to the basic unit. They are described in the following subsections. A photograph of the signal conditioning unit and the microcomputer at one of the field installations is shown in figure 4.

5.2 MEMORY BOARD

The memory board was purchased from:

Artec Electronics, Inc.
605 Old County Road
San Carlos, CA 94070.

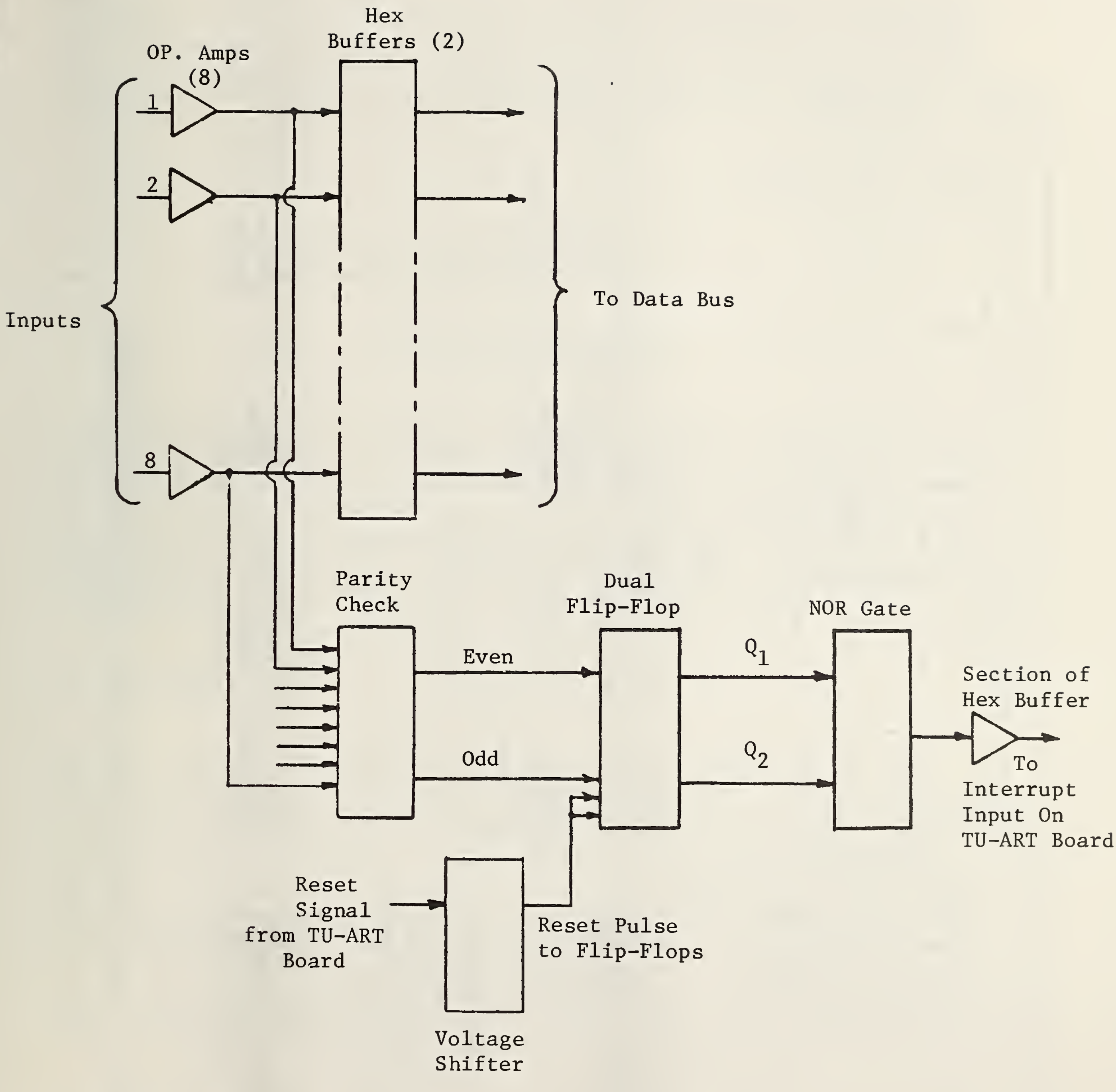


Figure 12. Diagram of encoder circuit board

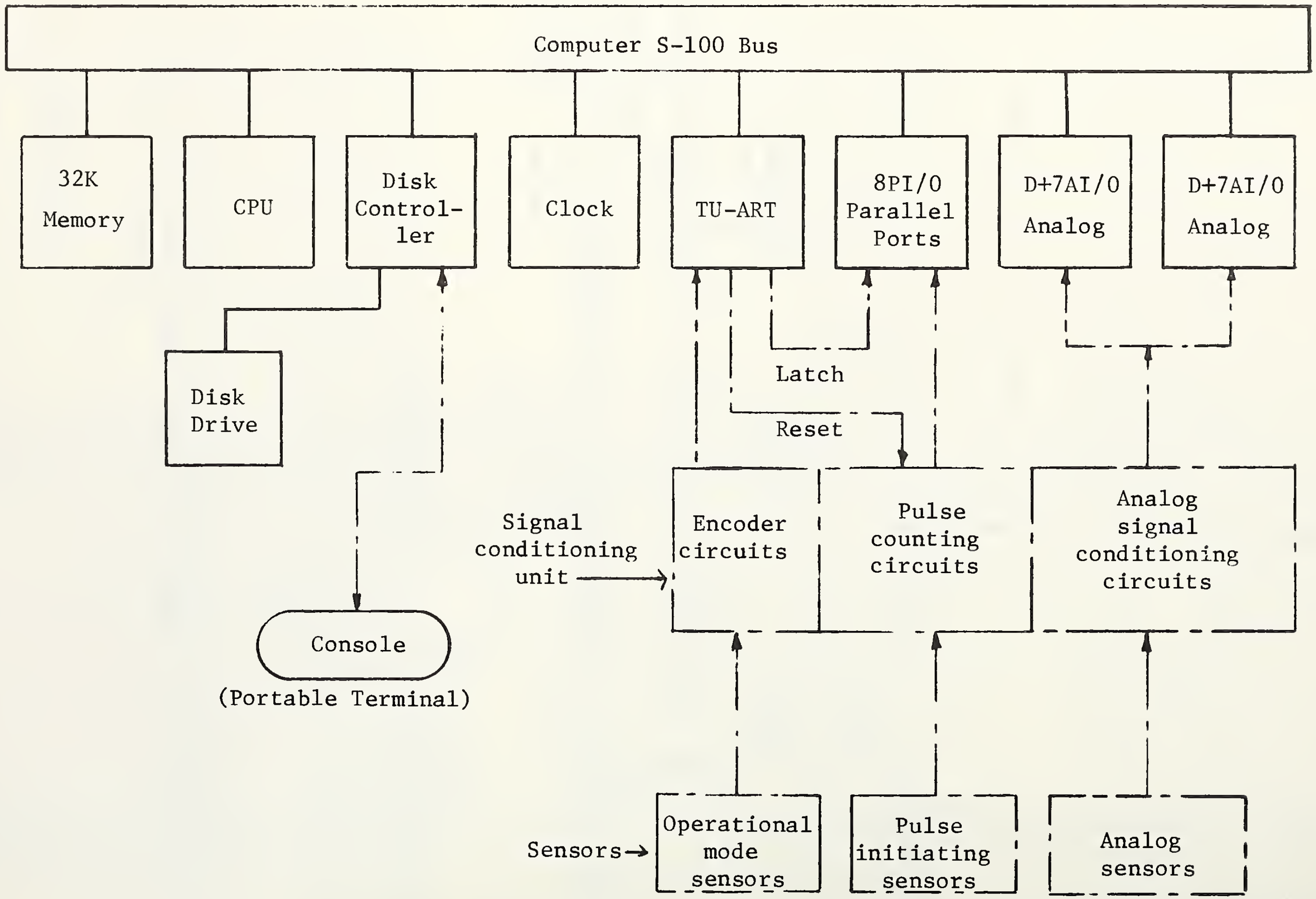


Figure 13. Block diagram of microcomputer data acquisition system.

These boards were purchased in kit form with the components for 8K bytes of memory. However, the boards were expandable to 32K bytes and the additional memory chips were ordered separately and installed. The 32K byte memory was necessary to meet the requirements of the software designed for this project. Each of these boards was tested after assembly and installation on the S-100 bus.

5.3 CLOCK

The "clock" board was manufactured by:

Mountain Hardware, Inc.
300 Harvey West Boulevard
Santa Cruz, CA 95060

This clock is capable of generating interrupt signals at intervals as fast as 100 micro-seconds. In this project, the clock was programmed to present the year, month, day, hour, minute and second, upon demand. This allowed all events to be recorded showing the time to the nearest second that the event occurred. The clock board is designed to hold a rechargeable 9-volt NiCad battery with a built-in recharging circuit. This battery is capable of keeping the clock running for a period of up to four days in the event of a power failure.

5.4 8PIO BOARD

The 8PIO board was manufactured by Cromemco and is capable of handling data from eight independent parallel input/output ports. These eight bi-directional, 8-bit ports can be used either singly or coupled together to form longer word lengths. In this project, four 12-bit words were required by the binary pulse-counting circuits. Therefore, one-and-one-half ports on this board were used for each pulse counting circuit. The software addressed these parallel I/O ports to obtain the proper data for processing in the microcomputer. Output data are automatically latched into an "OUT" port when the CPU executes an output instruction to the 8PIO, and the input data are latched into an 8PIO "IN" port by a positive-going signal transition on the "Latch Input" line. The latter is shown in the block diagram coming from the TU-ART board.

5.5 "TU-ART" BOARD

The Cromemco TU-ART (Twin Universal Asynchronous Receiver and Transmitter) board was capable of providing many types of direct input and output to and from the S-100 bus. The primary features of this board used in this project were providing the necessary "latch input" signal to the 8PIO board and the reset pulse at the end of each cycle for the four pulse counting circuits in the signal conditioning unit. In addition, this board provided access to the CPU for the mode-of-operation signals from the encoder board monitoring the compressor and outdoor fan opto-couplers. The numerous other features of this board are not used in this project and will not be described in this report.

5.6 ANALOG BOARDS

The last two boards on the right side of the block diagram in figure 13 were "D+7AI/O" multi-channel microcomputer analog interface boards manufactured by Cromemco. Each of these boards offers one 8-bit parallel I/o channel and seven channels of 8 bit analog-to-digital conversion of input data and seven channels of 8 bit digital-to-analog conversion of output analog data. The analog signal range of these boards is -2.56 volts to +2.54 volts in 20 millivolt increments for both the input and output. In this project, since only the input data were being recorded and/or processed, only the input feature of these boards was used. The maximum input bias current is 2 microamps and the input impedance is 20M. These features allowed the signal conditioning circuits to be designed without concern for load problems. Briefly, the signals from the analog channels are buffered on this board and the input multiplexing switch selects one of the channels. The signal on this channel is connected to one input of an analog comparator. The other input to this comparator is derived from the output of the D-to-A converter. A successive approximation shift register receives the output of the analog comparator and outputs a successively larger or smaller digital word to the D/A converter based on the output of the comparator. When the conversion is complete, the 8-bit output of the shift register is put on the CPU data input bus through the necessary drivers. The multiplexing switches to the next channel and this process is repeated. Since this project required a minimum of nine analog input channels, two analog boards were used.

5.7 EXPANSION CAPABILITIES

It should be noted that the DAS system recording field data was capable of some expansion without additional hardware. For example, only 9 of the 14 available analog input channels were being utilized and only 6 of the 8PIO channels were in use on the 8PIO boards. In addition, the two digital channels available on the analog boards were not being used. Also, six of the eight interrupt channels are available for use. Although these additional channels were available within the microcomputer, the software and the necessary signal conditioning, pulse-counting, or interrupt transducer circuits would be required. In general, the software would probably present the most difficult task in expanding the present system to utilize its full capabilities.

5.8 DATA RECORDED AT FIELD TEST UNITS

The periods of scanning and recording data were discussed in section 2.2. Examples of printouts taken directly from the raw data are shown in Appendix A. Examples of the printout received through a portable RS-232 console connected to a field unit as shown in figure 13 are shown in Appendix B. A brief explanation of each of these types of printouts is also given.

When the field units were first put on-line (March/April 1980), each data scan was being recorded on the disk. It was soon discovered that the capacity of the disk was too small to record data at this level of detail. At this point, the software was modified so that data from each scan were used in the calculations but only the scan data from every 10th cycle were recorded on the disk. (Data summarizing the performance of the heat pump during each cycle continued to be recorded as originally intended.) This modification allowed the five-inch disk to retain from 5 to 10 days of data, depending upon the activity of the heat pump. The cycles for which the scan data were recorded provide the detail data on temperature profiles and minute-by-minute operation that were needed for analyzing certain aspects of the dynamic performance of heat pumps.

The software also allowed the frequency of recording the scan data to be manually overridden through the portable console. This feature was extremely valuable in making tests and observing the reaction of all input channels during a field visit to check out both the instrumentation and the heat pump.

5.9 EFFECTS OF POWER FAILURES

The microcomputer utilized 32K of static memory. When the power is removed from this type of memory (due to an electrical storm, power outage, etc.) the entire contents of the memory is lost and only the data recorded on the disk can be retrieved. To avoid the effects of losing the operating programs used to direct the operation of the microcomputer, a "bootstrapping" program was added to the system in a PROM (programmable read-only memory). Since this memory is non-mobile, the bootstrap program allowed the operating program to be reloaded from the disk when a power failure was over and for the microprocessor to proceed in a normal fashion.

Although this bootstrapping method has been relatively successful in this project, it leaves several items to be desired. For example, if the power is lost for any reason, the bootstrapping procedure starts the instant the power is resumed. This procedure takes about 15 seconds. If the power is momentarily lost again within this period, the bootstrapping procedure is stopped and the operating programs are not loaded in. If this should occur, all data stored on the disk is normally lost if the file on the disk has not been closed.

The momentary outages from electrical storms frequently occur in short bursts and a situation such as described above is not uncommon. To avoid losing a disk of data, the field microcomputer is checked for the loss of the operational program prior to the removal of the disk. If the program has been lost, the reset switch is activated and the bootstrapping program is allowed to refresh the memory and allow the data file to be closed by the console, thereby retaining the data prior to the loss of power. However, no data will have been recorded for the period after the power outage.

To aovid this shortcoming, several steps were taken and additional ones were considered. One step considered was to modify the software, so that a file is closed at the end of each recording and is reopened to record additional data. This modification to the software will avoid the loss of data if a disk should accidentally be removed without closing the file. An additional step under considered was to incorporate a delay in the start of the bootstrapping program after a power outage. This would require an electro-mechanical device to be installed in the microcomputer to electrically activate the reset switch only after a short period (10-30 sec.) has elapsed from when the power is restored. This feature should overcome the effects of short bursts of outages, allow the power supply to charge its capacitors and supply the unit with the proper voltages, and allow the disk drive to reach its required speed before the bootstrapping program is executed. Another possibility considered was the installation of a "dead-man timer." The microcomputer would be required to reset this timer before it timed out. If it did not, the timer would reset the microcomputer and the operating programs would automatically be reloaded from the disk.

5.10 INTERFERENCE OF TELEVISION PROGRAMS

Any microprocessor will cause television interference in the lower frequency channels (channels 2 through 6) if it is close enough to the antenna of the TV set and is not completely shielded. If this happens, the picture is either superimposed by a variety of herringbone patterns (depending upon the activity of the microcomputer) or completely "washed out." If the antenna of the set is far enough (>30 feet) away from the microcomputer, this interference is negligible. The interference is caused by very weak high frequency radiation resulting from the fast switching and other activity taking place in the microcomputer. Shielding is effective only if the microcomputer is completely shielded for this high frequency radiation. This is a very difficult task since any joint or opening in the shield will amplify the radiation to escape unless the joint is electrically "sealed." See reference 8 for further details.

Fortunately, this is a relatively weak signal and the antennas are generally far enough away from the source to overcome these effects or at least minimize them. However, in one of the field test homes, the television room was directly above the microcomputer and the built-in antennas on the television set were being utilized. The majority of the attempts to shield the unit actually aggravated the situation because of radiation from the "edge" of an incomplete shield. The best solution to this problem would have been to supply the homeowner with an outdoor antenna. However, the cost, liability, and other administrative problems prevented this action. Several different types of television interference filters were obtained and attached to the set. Each type reduced the extent of the problem but none eliminated the problem completely. This problem was eventually "solved" by the combination of an interference filter, finding a specific location for the indoor antennas for each low frequency channel, and the willingness of the homeowner to accept a very weak herringbone pattern superimposed on the lower frequency channels.

6. EQUATIONS USED FOR PROCESSING THE DATA IN THE FIELD UNITS

The fundamental equations used in the software routines to convert the raw data to meaningful engineering quantities are discussed in the following subsections.

6.1 ANALOG DATA

In general, the analog data are processed by using the slope and offset of the first order equation established during the calibration of the transducers and signal conditioning circuits. Since the signal conditioning circuits were designed to function within the range of -2.50 volts to +2.50 volts, the simple equation:

$$X = (Vm + 2.50)a + b$$

was used, where

Vm = actual output voltage from the signal conditioning circuit,

a = slope of the straight line which is equal to the maximum value of the engineering units divided by 5,

b = offset of the straight line corresponding to the engineering value selected for the lower end of the voltage scale (i.e., engineering value at -2.50 volts),

X = the engineering value equivalent to a voltage of Vm from the signal conditioning/transducer circuit.

This basic equation required modification for the differential temperature measurement obtained using the thermopile since the output (of the thermopile) was not a straight line. The value of ΔT (degrees Fahrenheit) was obtained using the equation which follows:

During the heating season:

$$\Delta T = 44.4919\Delta mv - 0.98659(\Delta mv)^2$$

where $\Delta mv = 0.6134X - 1.5335$, and

X = the actual voltage output of the signal conditioning circuit.

During the cooling season:

$$\Delta T = 43.9624\Delta mv + 1.2911(\Delta mv)^2$$

where $\Delta mv = 0.2242X + 0.5605$, and

X = the actual output voltage of the signal conditioning circuit.

As discussed previously, the amplifier boards are changed as the season changes from heating to cooling and visa versa. The leads from the thermopile are also reversed to provide a positive output with the appropriate gain for the selected range of ΔT (see table 1).

6.2 DIGITAL DATA

As described in the preceding sections, the digital data were taken directly from the binary counting boards in the signal conditioning unit. The following constants are used for each digital counter:

Digital counter no. 1, (The watt-hour meter registering energy consumed by the compressor and outdoor fan.)

1 pulse = 1 watt-hour

Digital counter no. 2, (The watt-hour meter registering energy consumed by indoor fan and first stage of auxiliary heaters.)

1 pulse = 20 watt-hours

Digital counter no. 3, (The watt-hour meter registering energy consumed by second and third stages of auxiliary heaters.)

1 pulse = 20 watt-hours

Digital counter no. 4, (This counter registers the pulses or strokes of the condensate pump.)

1 pulse = 1 ml

6.3 DATA REDUCTION EQUATIONS

To conserve space on the raw data recording disk, many of the actual engineering quantities were processed by the microcomputer in the field to give the data of primary interest to the project. The equations used in the software routines are listed below:

1) $$V_n'' = \left(\frac{29.92}{P_b}\right) (0.02521)\ (460 + t_5)(1 + 1.6078\ W_s)$$

where V_n'' = specific volume of dry air (ft^3/lb_{ma}) at t_5 and P_b,

t_5 = temperature of air (°F) in the return duct (sensor no. 5),

P_b = barometric pressure (inches of mercury) (sensor no. 4), and the constant 0.02521 has the units of $ft^3/[(lb_{ma})\ (°R)]$.

2) $$\ln P_v^s = 15.4638 - \frac{7284}{T_{DP} + 392}$$

where P_v^s = partial pressure of saturated water vapor at T_{DP} (inches of H_g), and

T_{DP} = dew point in °F (sensor no. 2).

The above empirical forumula was taken from reference 7.

3) $W_s = 0.622\left(\dfrac{P_v^s}{P_b - P_v^s}\right)$,

where W_s = humidity ratio at the dew point of the air measured in the return air duct (lb. of water vapor/lb. of dry air),

P_b = barometric pressure (inches of mercury), and the constant 0.622 is the ratio of the molecular weights of water to the molecular weight of dry air

4) $\dot{V} = 1096\ A\sqrt{\dfrac{P_v V_n''}{1 + W_s}}$,

where $\dot{V}$ = volumetric flow rate in CFM in the return duct of the air/water vapor mixture at the temperature and pressure in the duct,

A = the cross sectional area in ft^2 of the return duct at the pitot tube and is a constant established for each field unit,

P_v = the velocity head in inches of H_2O obtained from the pitot tube - (sensor no. 1).

5) $Q_s = \dfrac{\dot{V}\ C_{pa}\ \Delta T\ \Delta t}{60\ V_n''}$,

where C_{pa} = $.240 + .444\ W_s$ is the specific heat of air-water mixture, $Btu/[(lb_{ma}) \cdot (°F)]$

ΔT = the temperature difference across unit in °F (sensor no. 7), and

Δt = the time period of interest in seconds.

6) Q_L = (2.334 Btu/ml) · (number of ml of condensate collected in a given period),

where Q_L = latent cooling done during the same period.

7. SOFTWARE

Figure 14 is a block diagram indicating the execution sequence and major program elements of the data acquisition and calculation program. The main routine "PUMP" is supported by a host of FORTRAN and assembly language subroutines, each handling a specific program task. Assembly language was used for some subroutines to facilitate handling of the interrupt requests on the CPU, the initialization of the computer hardware, and the monitoring of the mode-of-operation of the heat pump.

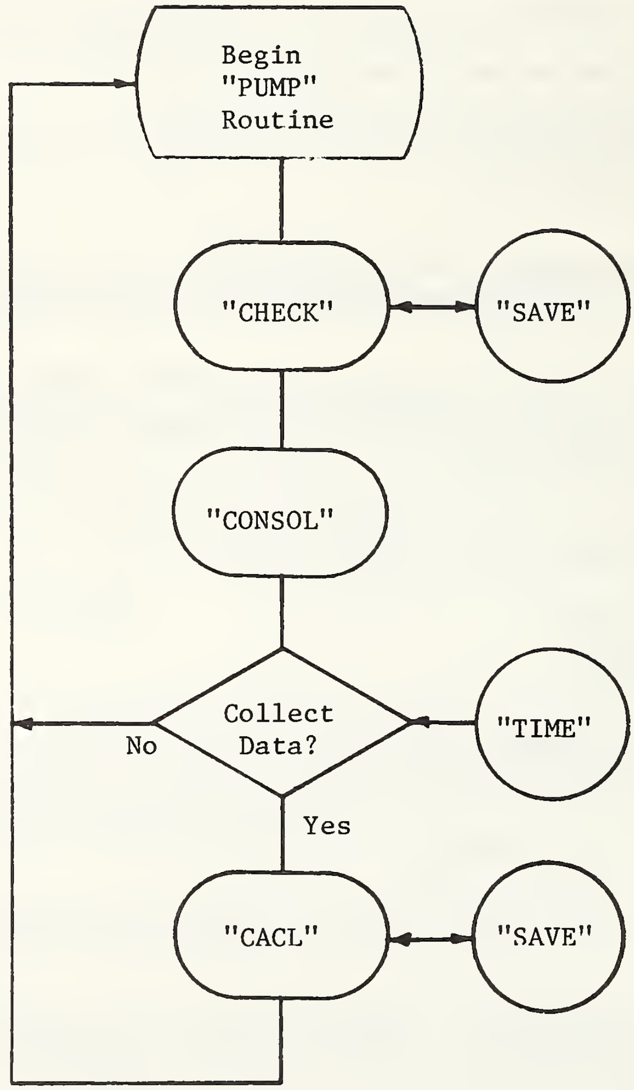


Figure 14. Simplified block diagram of data acquisition program execution sequence and major program elements.

The program execution follows basically a loop which consists of a number of calls to the supporting subroutines. The console status is continuously monitored for input and the time of day sampled to satisfy data computation and storage needs. In addition, the mode-of-operation of the heat pump system is queried to determine data scanning requirements.

Data common to many of the subroutines are made available in the program common areas. The reduced data are recorded on the floppy disk in files of 128 byte increments. Data are recorded on the disk in binary form and retrieved for further processing at a later time by the data listing or analysis programs. A brief description of the "PUMP" routine and the various subroutines is given in this section. Printouts of the software programs are shown in Appendix C.

7.1 MAIN ROUTINE "PUMP"

The software routine "PUMP" is the main or principal module of the data acquisition and calculation program. It is designed to initialize the computer system and the peripherals after a cold start, monitor the state of various program parameters, conduct heat pump system scan calculations, and handle the program termination.

The program begins execution by first initializing the computer and sensor interface hardware with calls to appropriate subroutines. The system data file, "PUMPDATA.DTA," is opened on the floppy disk and an attempt is made to read the first (header) record. If the first record is not present in the data file, as indicated by any end-of-file status being returned from the disk read, the disk is assumed to be a new one and is set up for the first data record. Numerous program variables, such as counters and heat pump system on-off time, are initialized to their appropriate values before beginning execution of the main program loop.

Statements in the main program loop are designed to allow the computer to continually monitor the heat pump status, (compressor on-off and defrost) and the portable console keyboard input when applicable. Various subroutines are called and executed depending on the status of the heat pump and console status parameters. When the heat pump status indicates the compressor has just turned on, the main program will call for sensors scans. Some scans are handled in the main program while others are performed in the supporting subroutines. Upon completion of the scan, the data are processed and the microcomputer will normally loop back and wait for the signal for the next scan, end-of-cycle indication, compressor turn-off, or program termination command status. When the compressor is de-energized, the computer will cease waiting for the start of sensor scanning, and jump to the beginning of the main program loop, and wait for the compressor to be re-energized.

Upon reception of a termination command from either the console or the file record counter indicating a full disk, the computer will close the data file after emptying the buffer and execute a "pause" command. Control of the computer system will then pass to the disk operating system (CDOS) after the user at the console enters a carriage return to bring the program out of the "pause" state.

7.2 SUBROUTINE "SYSTEM"

This is an assembly language subroutine and is designed to set up the computer hardware and CPU interrupts. The address of the interrupt service routine is stored in low memory. When an interrupt request has been received by the CPU, the low memory locations are inspected to obtain the address of the service routine. The appropriate interrupt/acknowledge mode is set by this subroutine. In addition to the above actions, the computer real-time clock and serial-parallel interface interrupts, and the corresponding acknowledgements are disabled.

7.3 SUBROUTINE "INIT"

This assembly language subroutine will initialize the cycle and/or daily summation data arrays, depending on the value of the variable "SET." When this parameter is equal to one, only cycle data will be initialized (all variable cycle data arrays are set equal to zero); when it is equal to two, the daily data arrays will be similarly reset, and finally, when it is equal to zero, the cycle as well as daily data will be reset. In addition, whenever the cycle data is reset, the digital counters in the signal conditioning unit are also reset.

7.4 SUBROUTINE "PSCAN"

This assembly language subroutine is designed to scan the eight 8-bit parallel ports (six of which are used in this project to handle the four 12-bit pulse counters) on the computer parallel interface board and stuff the raw data into four integer (16 bits or 2 bytes long) variables. This task is initiated by updating the hardware counters and then latching their respective outputs. The eight bytes of data are then read by the computer and stored in memory. Manipulation of the data to set up the four-element integer array is accomplished by shifting nibbles of data from one memory location, through CPU registers, to another memory location. Table 2 indicates the parallel port assignments and the corresponding byte and integer-array elements.

7.5 SUBROUTINE "ENERGY"

This FORTRAN language subroutine is responsible for monitoring and updating the electrical energy input and latent energy output of the heat pump system. This is accomplished by calling subroutine "PSCAN" to obtain the current values of the pulse counters for comparison with previously collected values. This is done to check for pulse counters exceeding their maximum capacity of 4095 counts. When over-ranging first occurs, indicated by value "DIG" being less than "LDIG," the quantity of 4096 will be added to "DIG" to make "DIG" properly larger than "LDIG." Next, the value of "LDIG" is replaced by "DIG." Subsequent comparisons of "DIG" and "LDIG" will then always yield "DIG" less than "LDIG," until the current cycle is completed and the pertinent hardware and software reset.

Table 2

Port assignments for the 8-port, multi-channel parallel interface (8PIO) board to allow 12-bit parallel interfacing.

Counter		8PIO Board Port Bit 0-7	8PIO Board Port Bit 8-11
1A (compressor + outdoor fan)	(DIG1)	Port 1	upper 1/2 Port 2
1B (indoor fan + aux. heat)	(DIG2)	Port 0	upper 1/2 Port 3
2A (auxiliary heat)	(DIG3)	Port 5	lower 1/2 Port 2
2B (condensate pump)	(DIG4)	Port 4	lower 1/2 Port 3

Note:

1) Counters 1A, 1B and 2A are pulsed by watt-hour meters. Counter 1A represents 1 watt-hour per pulse. Counters 1B and 2A represent 20 watt-hours per pulse. Counter 2B is pulsed by the condensate pump and represents 1 ml per pulse.

2) Port 6 is not used in this project.

3) Port 7 is used for coding the heat pump unit number and indicates a heating or cooling season.

Upon completion of the four counter comparisons, the three energy sums, the compressor and supplemental heater energy consumption, and the latent heat values are all updated.

7.6 SUBROUTINE "ENABL & DISABL"

The assembly language subroutines "ENABL" and "DISABL" simply enable and disable the interrupt capability of the computer system clock, respectively.

7.7 SUBROUTINE "TIME"

This assembly language subroutine queries the computer system clock and stores the time in a common data area. The time is collected only after the transition of the 100 μsec time digit. This is to ensure that no digits change during the collection and storage period. Since this routine was written in assembly language, speed is optimized and all the time components are stored in less than 100 μsec. Nine digits are reserved, which include the date and the time down to one second. Since the data input is in BCD format, the routine completes the data manipulation by masking off the high nibble of the data byte.

7.8 SUBROUTINE "MOTOR"

This simple assembly language subroutine turns the computer system floppy disk drive motors off. This routine is used to extend the life of the drives as well as the floppy disk magnetic media. This is accomplished through a system (CDOS, Cromemco disk operations system) call. The drive motors turn on automatically when the next "write" operation takes place.

7.9 SUBROUTINE "RESET"

This simple assembly language subroutine resets the disk operating system (the disks are logged off) and selects drive "A" as the default drive. This routine is utilized whenever the data disk is changed, and is called before the disk is removed. A new data disk is logged in the next time it is accessed.

7.10 SUBROUTINE "SETUP"

This assembly language subroutine initiates the analog data scans. The routine begins by first initializing the analog raw data area pointer. The reduced data array is then set to zero to allow subsequent data averaging. Status variables, one to indicate the number of analog scans, and another the scanning status, are reset. Finally, the analog scans are started by setting and enabling the one milli-second clock interrupt.

7.11 SUBROUTINE "ANALOG"

This FORTRAN language subroutine collects and averages the analog data from each analog channel as well as performs the preliminary data reduction. The data collection is initiated by calling subroutine "SETUP." The raw data, stored in array "ADATA," are averaged by monitoring the scanning progress through variable count. To increase program efficiency, data averaging progresses simultaneously with the raw data gathering. The count variable indicates the number of completed analog scans. A total of 33 analog scans of each channel, one milli-second apart, are performed. The one milli-second scan interval was chosen as a means of software filtering any 60-Hertz noise which may be present on the analog data signal inputs to the computer. The averaged data are processed with the appropriate factors that are related to the season (cooling or heating). The reduced data are stored in a real data array "AD2" for use by other software routines.

7.12 SUBROUTINE "CURTIM"

This FORTRAN language subroutine collects the current time of day and the current sequential day of the year. These are obtained by calling subroutine "TIME" and subsequently computing the time and day. The results are stored in real variable T and integer variable day.

7.13 SUBROUTINE "SAVE"

This FORTRAN language subroutine handles the storage of reduced data on the floppy disk. The data are buffered in memory before storage on the floppy disk. Two calls to this routine will result in one data record being stored in the data file on the floppy disk. This method was used to make efficient use of the available space on the disk, and reduce the rate of accesses to the disk. This routine is called to store four types of data; periodic scan data, cycle data, daily data and half hour data. Examples of each type of data recorded are listed in Appendix A. The data are stored on the disk utilizing two data arrays, byte array A and real variable array B. The data are written to the disk in binary form, totaling 128 bytes per record. A record counter is maintained by the program and stored on the first record of the data file. In addition to this variable, the heat pump unit number, the program version, the cycle storage indicator, and the number of power failures are stored on the first record. This record is read by the data acquisition program whenver a cold start is implemented (e.g., when the computer is manually reset or a power outage occurs). The first reocrd provides the program with the necessary information to continue.

7.14 SUBROUTINE "CALC"

This FORTRAN language subroutine reduces the data from the analog scan sequence. The routine begins by calling subroutine "ANALOG" to provide current analog data. The current time and heat pump system mode of operation are next checked. Finally, the required system variables, such as a flow rate, density, and humidity, are computed and stored in a common area called "RDATA."

7.15 SUBROUTINE "CHECK"

This FORTRAN language subroutine monitors the current time of day and specifically checks for any one-half hour (minutes equal to zero or thirty) or twenty-four hour (hour and minutes equal zero) times. Initially this routine will update the energy consumption variables with a call to subroutine "ENERGY." Following this, the time is sampled and examined for a half-hour time. If a half-hour increment is not detected, the routine exit sequence, consisting of a call to subroutine "CONSOLE," is executed. If a half-hour time is detected a check is made to see if data have already been processed for the same half-hour. If data have been processed the exit sequence is executed. It it has not been processed, the subroutine "CALC" is called and the average outdoor dry-bulb temperature and outdoor dew-point temperature are updated. Next, a check is made to see if the hour and minutes correspond to hour twenty-four. The exit sequence is executed if the twenty-four hour period is not present. Otherwise, the twenty-four-hour computations are completed. These involve calculating average temperatures, energy consumption and the daily coefficient of performance. The reduced data are then saved and the daily data arrays re-initialized. The routine finally terminates with the exit sequence previously defined.

7.16 SUBROUTINE "CONSOL"

This FORTRAN subroutine was designed to accept and execute the various program commands delivered from the portable computer system console used in the field. Six single-control character commands allow the user to examine the analog, digital, cycle and daily data, check and modify the program status parameters, change data disks, and terminate the program.

The program begins execution by checking the console status port to see if a command character has been entered. If a character is available it is read in and program execution continued. The character is next compared to the preprogrammed commands to test for a possible match. When a match is detected, a jump to the routine corresponding to the particular command is executed.

One or more routines, incorporated entirely within subroutine "CONSOL," are performed for each of the six console commands.

The first routine outputs the current time and date plus selected program variables, such as the unit number, interrupt indicators, and disk file status. This routine is executed when a "list" program status command (control L) has been entered as well as for all other requests, except the exit program and scan/save counter commands.

The exit program command (control E) causes the exit variable to be set. The computer will terminate execution of the data acquisition program when the exit status has been detected in the main program module "PUMP."

The new data disk routine is executed (control N) whenever the current data disk is replaced for a new one. This routine begins by closing the data file and then requests the user to remove the old data disk and insert the new one. When the computer receives a prompt from the user indicating that a new disk has been installed, the routine disables the interrupts and resets the disk operating system. This disk is checked to make sure it is a new one by opening the data file and attempting to read the file status of the first record. An end-of-file status returned from the read operation will signal to the routine that the disk is indeed a new one and operation of the program may continue. If the end of file status is not detected the routine will loop back and again ask the user to change the data disk.

The data dump routines, (control D and control T) simply output to the console the requested type of data, and then return program control to the calling program. An example of a printout from the console for the data dump routine is shown in Appendix B.

The cycle scan/counter routine (control V) is available so the user is able to alter the scan counter parameter. The computer will output the current value of the parameter and then request the desired number of cycles between scan recording. Upon reception of the new value from the console, the parameter is echoed back to the console to provide the user with a means of verifying the input. If the input was less than or equal to zero, the default value of ten is selected.

7.17 SUBROUTINE "CYCLE"

This routine is called to perform the heat pump cycle calculations, such as system time-on, energy, daily and cycle sum computations, when the following conditions occur:

a) a change in state of the heat pump defrost mode status (heating season only), or

b) the heat pump system is energized.

The routine will first check to see if the end-of-cycle is due to a change in the defrost status. If the defrost mode has ended, then the compressor time-on computation is modified accordingly. The defrost time-on computation is also made at this time. The routine continues by updating the various time parameters and time of day and date of the new cycle. The electrical energy consumption is next computed after a call to subroutine "ENERGY." Next, the daily data sums, sensible and latent heat, system time-on, electrical energy, and temperature averages, are performed. The cycle calculations, including the average temperature and cycle performance, are completed last. The data are saved on the disk with a call to subroutine "SAVE," and the cycle sum variables reset to zero.

7.18 SUBROUTINE "SERV"

This assembly language routine is designed to handle the interrupt requests to the CPU generated by the computer system clock, and to manage the complex timing of the analgo and digital scans.

The program handles two types of interrupts; those produced in the periodic mode, and those created when in the analog scan mode. A determination of the interrupt type is made immediately upon entering the service routine by inspecting the value of the analog scan counter. Should the value be equal to thirty-three then the periodic interrupt routine will be executed. A value of less than thirty-three indicates that analog scanning is in progress.

Heat pump status checks are made to determine the current compressor and defrost states. Should the compressor turn off between periodic interrupts (there is a maximum one-second delay before the condition is detected), a check will be made to see if the system was also in the defrost mode. If this is the case (a condition unlikely to occur unless the system was shut off manually from the thermostat), the defrost end of cycle status variable will be set. Otherwise, only the periodic interrupt status variable will be updated to indicate termination of the interrupts.

A compressor "on" condition will cause the program to jump to the defrost mode check routine. This routine will compare the previous state of the defrost mode with the current one. If a change in state has been detected, the defrost status variable is updated and a check is made to see if the system is currently in the defrost mode. If the system defrost is in operation, indicating that it had been initiated since the last periodic interrupt, the current time will be stored in the defrost start time array. Any change in the defrost state will cause the following variables to be updated:

a) The end of defrost cycle indicator will be set.

b) The defrost counter will be incremented.

c) The periodic interrupt counter will be reset.

d) The analog scan time zone indicator will be reset.

e) The analog divide-down counter will be set to one.

f) The interrupt divide-down counter will be set to one.

These variables are used to indicate to other programs and routines that an end-of-cycle has occurred and that the analog scan interval time base has been reset. If no change is detected in the defrost state, the program will jump to the periodic interrupt count routine.

The time between analog data scans is controlled by the interrupt count and divide-down routine. The principle interrupt frequency, one per second, is divided by ten to essentially provide one and ten second count for the program routines. When a ten second count is detected, the ten second counter and the analog scan interval tables are utilized to determine whether or not an analog scan should be made. If a data scan is to be made, a call to the analog scan set up routine is made. Otherwise, the one second periodic interrupts are continued.

7.19 PROGRAM "REDUCE"

This FORTRAN program is designed to run on a central microcomputer that is similar to, but remote from, the field units. It takes the data stored on the disks in the field, groups it into four categories and displays it on the console of central computer. The four categories are:

a) scan data

b) cycle data

c) daily data

d) 1/2 hour data

The central microcomputer will request from the user information on which of the four types of data are to be listed on the console. When the proper response has been received, the disk operating system will be reset. A header, obtained from the first record of the data file, will be listed on the console giving the following information:

a) The unit number of the field data acquisition system.

b) The total number of records in the data file.

c) The version number of the data acquisition and calculation program.

d) The number of recorded pieces of data of the type requested.

e) The number of power failures encountered by the computer system while in the field.

The requested data are then read from the disk and stored in memory. When the read operation is complete and all the requested data have been found or the memory buffer is full, a data header is printed on the console giving the time and date the data were stored on the disk and the mode of operation of the computer at that time. The data are then printed out in columns (there is a total of five columns across the page), with each column corresponding to a record of recorded data. In addition, a column is furnished to provide the data element number and abbreviated data name. When the listing operation is complete, the data file is closed and the computer will again request from the user the type of data to be displayed. Examples of printouts resulting from use of the program "REDUCE" are found in Appendix A.

7.20 SOFTWARE FOR EDITING FIELD DATA, "DCEP"

A review of raw data disks indicated the need for a method of editing the raw cyclic data on the central microcomputer prior to making seasonal performance calculations. Examples of the types of changes that were found necessary are listed below:

1) Eliminating those extremely short cycles which were generated by manipulation of the thermostat, power outages, or by the process of changing disks. These cycles involved compressor on-times of 5 to 60 seconds. When the disks are changed in the field, the person making the change is trained to check the functioning of the data channels by cycling the unit on and off. He does this by either generating false compressor on-off signals at the signal conditioning unit and/or occasionally turning the compressor on and off via the thermostat. Since the extremely short cycles generated by such actions are not directly related to the seasonal performance of the unit, it was desirable to remove these data from the information to be analyzed.

2) On several occasions, the quantity of 4096 would be added by the subroutine "ENERGY" for some unknown reason. When this happened on the condensate pump counter, the value of the latent heat output would be incorrect and result in an erroneous C.O.P. for that cycle. Likewise, when this action would take place on one of the digital counters monitoring the watt-hour meters, the C.O.P. for the cycle would be very low. This problem was primarily observed on field test unit no. 1. Replacement of the counter circuit cards on this unit reduced but did not eliminate the problem. The binary counter integrated circuit chips are extremely sensitive to external noise, and it was likely that the counters were occasionally dropping one of the least significant bits and the software was adding 4096 as directed. Since these errors are readily detected it was desirable to edit the data and retain the corrected cyclic data for further analysis.

To accomplish these tasks, FORTRAN program "DCEP" was developed to allow the cyclic data on the raw data disk to be examined and the edited version to be recorded on a separate disk. The original data disk was left unchanged. In addition, the daily and cyclic data are corrected, where necessary, and recorded on separate sections of the new disk. Since only cyclic and daily records are involved in this process, the edited data from 5 to 10 raw data disks could be stored on one new disk.

One additional step was taken by this program in the updating of the daily data. As noted in subroutine "INIT," the field units close out the daily data at midnight. This program corrects this by closing out the daily data at the end of the last cycle for that day and records the time corresponding to the end of that cycle as the end of the day. Likewise the compressor-on time and defrost time correspond to the actual sums of the cycles that occurred through the last cycle of the day.

8. SUMMARY

The objective of this paper was to document the techniques, instrumentation, data acquisition systems, and data reduction systems used in monitoring the field performance of three residential heat pumps located in the Washington, D.C. area. This information was documented because of its application to future testing projects of this type in both small- and large-scale laboratory and field studies.

Initial estimates of the quantity of data required to meet the objectives of this task indicated that the simple recording of raw data would require data storage systems in the field and a data reduction facility with capacities far beyond those warranted by a project of this type. To avoid this problem, a strategy was developed which utilized an on-line microcomputer at each field unit to gather the data, reduce and analyze the results, and record the calculated results on a floppy disk. The raw data scans were recorded for every tenth on/off cycle to provide detailed performance data and allowed the calculated results to be confirmed. This strategy reduced the storage capacities required and greatly simplified the task of further reduction of the recorded data by a microcomputer located at NBS. The use of the on-line microcomputers was found to be a cost-effective solution compared to other techniques explored during a search for the most practical means of data accumulation.

The basic design of the instrumentation and data acquisition systems was described first to give the reader a better understanding of the steps taken in designing the individual components of the systems. The individual sensors and transducers are identified. Those sensors developed in-house are also described.

The signal conditioning unit was described in detail. This unit not only received the signals from the various transducers and amplified, attenuated and/or offset them to meet the input characteristics of the microcomputer, but also allowed the microcomputer to process them with maximum accuracy. The pulse-counting circuits were also described.

The functioning of the components of the microcomputer was described, together with the overall operation of the unit. A brief description of the software-using assembly and a higher level language was also given.

The experience gained from this project and documented in this paper should be helpful in other studies, in both the laboratory and the field, requiring the collection of large quantities of raw data on the dynamic and seasonal performance of heating and cooling equipment.

REFERENCES

1. Kelly, G. E., and Parken, W. H., "Method of Testing, Rating and Estimating the Seasonal Performance of Central Air-Conditioners and Heat Pumps Operating in the Cooling Mode," National Bureau of Standards, NBSIR 77-1271 (April 1978).

2. Parken, W. H., Kelly, G. E., and Didion, D. A., "Method of Testing, Rating and Estimating the Heating Seasonal Performance of Heat Pumps," National Bureau of Standards, NBSIR 80-2002 (April 1980).

3. Federal Register, Part III, Department of Energy, Office of Conservation and Solar Energy, Test Procedures for Central Air Conditioners, Including Heat Pumps, Thursday, December 27, 1979.

4. Kelly, G. E., and Bean, J., "Dynamic Performance of a Residential Air-to-Air Heat Pump," National Bureau of Standards, NBS Building Science Series 93, March 1977.

5. ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.), "Standard Measurements Guide: Section on Temperature Measurements," ASHRAE Standard 41-66, Part 1, January 27, 1966.

6. "Orientation Instrumentation Amplifiers," pp. 105-112, Analog Devices, Inc., Data Acquisition Products Catalog, 1978, Route 1, Industrial Park, P.O. Box 280, Norwood, Massachusetts 02062.

7. Brokaw, R. S., "Calculation of Flue Losses for High-Efficiency Furnaces and Appliances," ASHRAE (American Society of Heating, Refrigeration and Air-Conditioning Engineers, Inc.) Journal, January 1979.

8. Ciarcia, S., "Electromagnetic Interference," Byte Publications Inc., New York, Vol. 6, No. 1, P 48-68, January 1981.

Appendix A

Typical examples of printouts from the raw data disk.

Examples of scan data recorded on raw data disk in the field and printed by the central processing unit.

		MODE = 0 DAY = 205 15:00:54	MODE = 0 DAY = 205 15:01:04	MODE = 0 DAY = 205 15:01:14	MODE = 0 DAY = 205 15:01:24	MODE = 0 DAY = 205 15:01:34
1	DT	15.24	15.47	15.64	15.73	15.74
2	FLOW	1258.	1230.	1245.	1283.	1264.
3	QS	56.04	55.67	56.97	59.04	58.20
4	TRET	76.41	76.34	76.41	76.47	76.32
5	TOUT	85.33	85.27	85.60	85.61	85.55
6	TSUP	60.36	59.81	59.73	59.59	59.42
7	DP	.3508E-01	.3358E-01	.3437E-01	.3653E-01	.3543E-01
8	TDPR	55.81	55.92	56.76	55.86	56.70
9	TDPO	62.85	62.80	62.71	63.13	63.20
10	PATM	29.52	29.52	29.52	29.52	29.52
11	TSUP	0.000	0.000	0.000	0.000	0.000
12	QSUM	495.9	551.6	608.6	667.6	725.8
13	DIG1	105.0	115.0	125.0	136.0	146.0

Glossary of Terms for Scan Data

MODE = 0 - mode of operation (see table 1)
DAY = 015 - Julian day of year
09:31:26 - hour, minute and second of the scan

1 DT = differential temperature across the unit (°F)
2 FLOW = volumetric flow rate in cfm in return duct
3 QS = sensible heat output of the unit for the current time increment (10 sec.)
4 TRET = temperature of air in return duct (°F)
5 TOUT = temperature of outdoor air (°F)
6 TSUP = temperature of air in supply duct (°F)
7 DP = differential pressure of the pitot tube (inches of water)
8 TDPR = dew point of air in the return duct (°F)
9 TDPO = dew point of outdoor air (°F)
10 PATM = atmospheric pressure (inches of H_g)
11 TSUP = temperature of air in secondary supply duct of unit no. 3 (°F) (if applicable)
12 QSUM = sensible heat output of the unit since the start of cycle (Btu)
13 DIG1 = digital count registered on sensor 9 (see table 1)

Examples of cycle data recorded on the raw data disk in the field and printed by the central processing unit.

		MODE = 0 DAY = 015 06:49:47	MODE = 0 DAY = 015 07:24:50	MODE = 0 DAY = 015 08:02:25	MODE = 0 DAY = 015 08:44:55	MODE = 0 DAY = 015 09:30:26
1	QS	.1871E+05	.1610E+05	.1804E+05	.1961E+05	.1975E+05
2	QL	0.000	0.000	0.000	0.000	0.000
3	ECMP	1919.	1670.	1800.	2043.	2196.
4	EFAN	306.8	268.7	288.1	325.8	349.0
5	EHET	1693.	1571.	1832.	1794.	1351.
6	DIG2	100.00	92.00	106.0	106.0	85.00
7	TRET	67.44	62.78	67.54	68.20	67.49
8	DIG3	0.000	0.000	0.000	0.000	0.000
9	TDPR	26.69	25.17	27.17	27.52	27.36
10	CTIM	2401.	2103.	2255.	2550.	2731.
11	DTIM	208.0	213.0	216.0	226.0	234.0
12	COP	1.399	1.344	1.348	1.380	1.485
13	TOUT	28.23	28.59	28.76	29.18	30.23

Glossary of Terms for Cycle Data (See Section 2.2)

MODE = 0 - mode of operation (see table 1)
DAY = 015 - Julian day of year
06:49:47 - hour, minute and second at end of cycle

1 QS = sensible heat output of unit during cycle (Btu)
2 QL = latent cooling of unit during cycle (Btu)
3 ECMP = electrical energy consumed by compressor, outdoor fan and heater (watt-hours)
4 EFAN = electrical energy consumed by indoor fan (watt-hours)
5 EHET = electrical energy consumed by auxiliary heaters (watt-hours)
6 DIG2 = digital count registered on sensor 10 for cycle (see table 1)
7 TRET = average temperature of air in return duct during cycle (°F)
8 DIG3 = digital count registered on sensor 11 for cycle (see table 1)
9 TDPR = average dew point of air in return duct during cycle (°F)
10 CTIM = compressor on-time for cycle (seconds)
11 DTIM = defrost time for cycle (seconds)
12 COP = coefficient of performance for cycle
13 TOUT = outdoor temperature at the end of cycle (°F)

Example of daily data recorded on raw data disk in the field and printed by the central processing unit.

		MODE = 0 DAY = 021 00:00:00	MODE = 0 DAY = 022 00:00:00
1	QS	.2426E+06	.4038E+06
2	QL	0.000	0.000
3	ECMP	.3867E+05	.5176E+05
4	EFAN	4436.	8130.
5	EHET	89.18	6998.
6	TOUT	43.33	33.11
7	TRET	69.24	65.72
8	TDPO	32.07	30.68
9	TDPR	32.95	33.33
10	CTIM	.3472E+05	.6363E+05
11	DTIM	0.000	4586.
12	COP	1.646	1.769
13	%CON	.4018	.7364

Glossary of Terms for Daily Data (See Section 2.2)

MODE = 0 – mode of operation at midnight (see table 1)
DAY = – Julian day of year
00:00:00 – hour, minute and second (midnight, day 014)

1 QS = total sensible heat output of unit for past 24 hours (Btu)
2 QL = total latent cooling of unit for past 24 hours (Btu)
3 ECMP = electrical energy consumed by compressor, outdoor fan and heater (watt-hours)
4 EFAN = electrical energy consumed by indoor fan (watt-hours)
5 EHET = electrical energy consumed by auxiliary heaters (watt-hours)
6 TOUT = average outdoor temperature computed from 1/2 hr. data (°F)
7 TRET = average temperature in return duct during normal* operation (°F)
8 TDPO = average outdoor dew point computed from 1/2 hr. data (°F)
9 TDPR = average dew point of air in return duct during normal* operation (°F)
10 CTIM = total compressor-on time (seconds)
11 DTIM = total defrost time (seconds)
12 COP = coefficient of performance for the past 24 hours
13 %CON = fraction of compressor-on time for the past 24 hours

---

* "normal" refers to the total time the compressor was on less the time the unit was in the defrost.

Examples of one half hour data recorded on the raw data disk in the field and printed by the central processing unit.

		MODE = 0 DAY = 013 11:30:00	MODE = 0 DAY = 013 12:00:00	MODE = 0 DAY = 013 12:30:00	MODE = 0 DAY = 013 13:00:00	MODE = 0 DAY = 013 13:30:00
1	TOUT	21. 10	21. 92	22. 87	23. 39	23. 41
2	TDPO	4. 799	4. 153	6. 895	6. 863	5. 776
3	PATM	29. 72	29. 70	29. 70	29. 68	29. 66
4	FLOW	1175.	1202.	1190.	1271.	1166.
5	DT	58. 37	17. 85	18. 94	58. 83	17. 85
6	TDPR	23. 63	22. 68	24. 11	24. 20	23. 31
7	TSUP	114. 9	86. 64	90. 24	116. 2	87. 09
8	DIG1	668. 0	2035.	3413.	4795.	6189.
9	DIG2	184. 0	443. 0	622. 0	701. 0	768. 0
10	DIG3	0. 000	0. 000	0. 000	0. 000	0. 000
11	DIG4	0. 000	0. 000	0. 000	0. 000	0. 000
12	TRET	70. 55	69. 92	71. 43	71. 08	69. 95
13	TSUP	130. 0	85. 97	89. 80	131. 8	87. 49

Glossary of Terms for One Half Hour Data (See Section 2.2)

MODE = 0 - mode of operation (see table 1)
DAY = 013 - Julian day of year
11:30:00 - hour, minute, second of data recorded

1 TOUT = outdoor temperature (°F)
2 TDPO = dew point of outdoor air (°F)
3 PATM = barometric pressure (inches of H_g)
4 FLOW = volumetric flow rate in return duct (CFM)
5 DT = differential temperature across the unit (°F)
6 TDPR = dew point of air in return duct (°F)
7 TSUP = temperature of air in supply duct (°F)
8 DIG1 = digital count registered on sensor 9 (see table 1)
9 DIG2 = digital count registered on sensor 10 (see table 1)
10 DIG3 = digital count registered on sensor 11 (see table 1)
11 DIG4 = digital count registered on sensor 12 (see table 1)
12 TRET = temperature of air in return duct (°F)
13 TSUP = temperature of air in secondary supply duct of unit 3 (°F)

Appendix B

Examples of the data available at the units in the field using a portable terminal. These data allowed any erroneous results from improper operation of the heat pump or the data acquisition system to be quickly detected.

Appendix B

Example of direct print-out from portable terminal connected to the field microcomputer. The values listed are direct read outs from the microcomputer at the time shown.

```
--------------------------------------------------------------------------------
    PROGRAM STATUS         15: 6:39 EST      1/16/81      PROGRAM VERSION 5.3
--------------------------------------------------------------------------------

                 UNIT=  3  RECORD=  0    POWER FAILURES=  0

                       INTERPT=  1   ZONE=  1   MODE=  0

                                  HEATING SEASON
    1    -.02              0.00  RHO  =  14.08          COMP   177   177.0
    2   -2.14  DP           .02  PVS  =  .2016          HEAT     1   20.00
    3   -1.44  TDPR       34.83  WS   =  .4261E-02      HEAT     0   0.000
    4     .25  TDPO       31.40  FLOW =  1157.          COND     0   0.000
    5     .42  PATM       29.63  CPA  =  .24189E+00
    6    -.08  TRET       71.80  QS   =  .27102E+03
    7    -.58  TOUT       38.36
    8     .20              0.00
    9    -.02              0.00
   10   -1.70  DT         21.53
   11    -.82  TSUP       93.57
   12    -.86  TSP3       92.66
   13   -2.03              0.00
   14   -2.12              0.00
   15    0.00              0.00
   16    0.00              0.00
```

Eng. Value
Pulse Counts
Pulse Counting Circuit
(See Glossary)

Computed Values
(See Glossary)

Engineering Values
(See Glossary)

Voltage (Vm) Output of Signal
Conditioning Circuit

DAS Number (See table 1)

Glossary of Terms Shown on Printout from Portable Terminal Connected to Field Unit

The engineering values are listed below:

DP = differential pressure of the pitot tube (inches of H_2O).

TDPR = dew point of air in the return duct (degrees Fahrenheit).

TDPO = dew point of outdoor air (degrees Fahrenheit).

PATM = atmospheric pressure (inches of H_g).

TRET = temperature of air in the return duct (degrees Fahrenheit).

TOUT = temperature of outdoor air (degrees Fahrenheit).

DT = differential temperature across the unit (degrees Fahrenheit).

TSUP = temperature of air in the supply duct (degrees Fahrenheit).

TSP3 = temperature of air in secondary supply duct of Unit 3 (degrees Fahrenheit). Note: The values listed apply only to Unit 3 and are not used in the calculations.

Computed Values:

RHO = specific volume of dry air in duct (ft^3/lb at the temperature in supply duct and the barometric pressure at the time shown).

PVS = partial pressure of saturated water vapor at TDPR.

WS = humidity ratio at TDPR (lb of water vapor/lb of dry air).

FLOW = volumetric flow rate in cfm in the return duct of the air/water vapor mixture at the temperature in the return duct.

CPA = specific heat of the air-water mixture, Btu/($lb_{mixture}$) · (°F).

QS = sensible heat output of the unit since the start of the cycle (Btu).

Pulse Counting Circuits:

COMP = electrical energy consumed by the compressor, outdoor fan and sump heater. The value listed in the first column is the pulse count for the current cycle. The value in the second column is the equivalent engineering value in watt-hours.

$HEAT_1$ = electrical energy consumed by the indoor fan and first stage of the auxiliary heaters. The first column is the pulse count; the second column is the equivalent engineering value in watt-hours.

$HEAT_2$ = electrical energy consumed by additional stages of auxiliary heaters. Note: The indoor fan and all stages of auxiliary heat on unit no. 3 were registered on $HEAT_1$.

COND = condensate metering pump. The pulse count is listed in the first column and the millileters of condensate displaced is shown in the second column.

Appendix C

A microcomputer printout of all software used in this task.

```
      PROGRAM PUMP
C     HEAT PUMP DATA AQUISITION PROGRAM          3/03/81
C
C     >>>>>>   LOGICAL UNIT NUMBER ASSIGNMENTS   <<<<<<
C
C       AS  5=CONSOLE
C       AS  9=REDUCED DATA FILE
C
C     >>>>>>   I/O PORT ASSIGNMENTS [DECIMAL,(HEX)]   <<<<<<
C
C      000 (00) - 015 (0F) = 4FDC  (CONSOLE DEVICE)
C      016 (10) - 031 (1F) = D+7AIO  (2 CARDS)
C      032 (20) - 047 (2F) = TU-ART PORT A
C      048 (30) - 052 (34) = 4FDC  (DISK CONTROLLER)
C      064 (40) -          = BANK SELECK PORT
C      080 (50) - 095 (5F) = PRI  (PRINTER INTERFACE)
C      128 (80) - 143 (8F) = TU-ART PORT B
C      160 (A0) - 167 (A7) = 8PIO  (PARALLEL INTERFACE)
C      192 (C0) - 207 (CF) = CLOCK
C
C          >>>>>>   VARIABLE DEFINITIONS   <<<<<<
C
C          COUNT              ANALOG SCAN COUNT (33 is maximum count)
C          STAT               INTERRUPT SCAN STATUS  (0=data scanning)
C          ICNT               PERIODIC INTERUPT COUNTER
C          NSCAN              NUMBER OF SCANS/CURRENT CYCLE
C          DEFRST             DEFROST STATUS  (1=defrost on : outside fan off)
C          NFRST              NUMBER OF DEFROST CYCLES PER DAY (24 hours)
C          MODE               FAN-DEFROST STATUS BYTE
C          PINT               PERIODIC INTERRUPT STATUS (1 = interrupt active)
C          ZONE               TIME ZONE STATUS  (0,1,2, or 3)
C          DIVIDE             TSCAN DIVIDE DOWN COUNTER
C          SET                INIT INDICATOR (0=all, 1=cycle, 2=daily)
C          COMP               COMPRESSOR STATUS (1 = off)
C          UNIT               FIELD UNIT DESIGNATION  (1,2, or 3)
C          AD                 ANALOG PORT AVERAGE VALUE (real number)
C          ADATA              ANALOG PORT SCAN VALUES (integer)
C          PDATA              PARALLEL PORT DATA (integer)
C          TDATA              CLOCK DATA
C          EXIT               PROGRAM EXIT STATUS (1 = exit program)
C          TYPE               DATA FILE RECORD TYPE
C                                   0 = scan data
C                                   1 = cycle data
C                                   2 = daily data
C                                   3 = 1/2 hour data
C          EOC                END OF CYCLE INDICATOR (1 = end of def. cycle)
C          TSTART             COMPRESSOR START TIME (seconds 0-86400)
C          TEND               COMPRESSOR SHUT OFF TIME (seconds 0-86400)
C          TLAST              TIME OF LAST (previous) SCAN
C          TDAY               JULIAN DAY CYCLE STARTS (compressor turns on)
C          EDAY               JULIAN DAY COMPRESSOR SHUTS OFF
C          LDAY               JULIAN DAY OF LAST (previous) SCAN
C          TIM1               DEFROST START TIME (time array)
C          VER                PROGRAM VERSION NUMBER
C          CYDIV              CYCLE DIVIDE NUMBER (i.e. 10)
C          NCYCLE             NUMBER OF CYCLES
C          POWER              NUMBER OF POWER FAILURES or COMPUTER RESETS
C
       INTEGER*1 ADATA,PDATA,COUNT,ICNT,NSCAN,ZONE,PINT
```

```
      INTEGER*1 DEFRST, MODE, STAT, TIM1, TDATA, ICON, DIVIDE
      INTEGER*1 UNIT, EXIT, COMP, SET, TYPE, EOC, NFRST
      INTEGER POWER, DIG, CYDIV, DAY, TDAY, EDAY
C
      COMMON /STATUS/STAT, ICNT, NSCAN, DEFRST, COMP, MODE, EOC, NFRST
     *              , PINT, ZONE, DIVIDE
      COMMON /PARELL/PDATA(8), DIG(4)
      COMMON /ATOD/ADATA(528), AD1(16), AD2(16), COUNT
      COMMON /DATE/TDATA(9)
      COMMON /SUM/TSUMC(4), TIMC(2), GSUMC(2), ESUMC(3), LDIG(4),
     *            TSUMD(4), TIMD(2), GSUMD(2), ESUMD(3), COP(2), SET
      COMMON /HOUSE/UNIT, AREA(3), FAN(3)
      COMMON /TIM/TIM1(9), TSTART, TEND, TLAST, TDAY, EDAY, LDAY
      COMMON /RDATA/RHO, PVS, WS, FLOW, CPA, GS, VER
      COMMON /MISC/CYDIV, POWER
C
      DATA AREA, FAN/1. 65, 1. 58, 1. 833, 514. 0, 648. 0, 460. 0/
C
C     -------------------------------------------------------------
C     >>>>>>   HARDWARE AND SOFTWARE INITIALIZATION ROUTINES   <<<<<<
C     -------------------------------------------------------------
C
      CALL SYSTEM
      CALL OPEN(9, 'PUMPDATADTA', 1)
C
      UNIT=INP(Z'A7'). AND. Z'03'
      IF(UNIT. EQ. 0) UNIT=1
C
      READ(9, END=11, REC=1) UNIT, NREC, VER, CYDIV, POWER
      POWER=POWER+1
      WRITE(9, REC=1) UNIT, NREC, VER, CYDIV, POWER
      NREC=NREC+1
C
12    CALL MOTOR
      SET=0
      CALL INIT
C
C     >>>>>>   TIME VARIABLES INTIALIZATION   <<<<<<
C
      CALL CURTIM(T, DAY)
C
      TSTART=T
      TEND=T
      EDAY=DAY
      TDAY=DAY
C
      NCYCLE=0
      CYDIV=10
      PINT=0
      NFRST=0
C
C     -----------------------------------------------------
C     >>>>>>   B E G I N   M A I N   P R O G R A M   <<<<<<
C     -----------------------------------------------------
C
3     COUNT=33
      ICNT=0
      NSCAN=0
      DEFRST=0
      EOC=0
```

```
C
C     >>>>>>   COMPRESSOR TURN-ON CHECK ROUTINE   <<<<<<
C
7     CALL CHECK(NREC,ZONE,NFRST,EXIT,PINT)
      IF(EXIT.EQ.1) GO TO 8
C
      COMP=INP(Z'24').AND.Z'01'
      IF(COMP.EQ.1) GO TO 7
C
      CALL CYCLE(NREC,EOC)
      NCYCLE=NCYCLE+1
      IF(NCYCLE.GT.CYDIV) NCYCLE=0
C
C     ----------------------------------
C     >>>>>>   NEW CYCLE STARTS   <<<<<<
C     ----------------------------------
C
14    ZONE=0
      DIVIDE=1
      STAT=0
C
C     >>>>>>   SYSTEM STATUS CHECK ROUTINE   <<<<<<
C
5     IF(EOC.NE.1) GO TO 13
      CALL CYCLE(NREC,EOC)
      NCYCLE=NCYCLE+1
      IF(NCYCLE.GT.CYDIV) NCYCLE=0
13    IF(COMP.EQ.1) GO TO 2
      IF(STAT.EQ.0) GO TO 2
C
      CALL CHECK(NREC,ZONE,NFRST,EXIT,PINT)
      IF(EXIT.EQ.1) GO TO 8
      GO TO 5
C
C     >>>>>>  ANALOG SCAN DATA REDUCTION ROUTINE  <<<<<<
C
2     CALL CALC(ZONE,MODE)
C
      CALL ENERGY
C
      QSUMC(1)=QSUMC(1)+QS
C
      CALL CURTIM(TT,MDAY)
      DT=TT-TLAST+(MDAY-LDAY)*86400.
      LDAY=MDAY
      TLAST=TT
C
      IF(NCYCLE.LT.CYDIV) GO TO 15
C
      TYPE=0
C
      CALL SAVE(TYPE,UNIT,NREC,NFRST)
C
15    IF(DEFRST.EQ.2) GO TO 23
C
      TSUMC(2)=TSUMC(2)+AD2(6)*DT
      TSUMC(4)=TSUMC(4)+AD2(3)*DT
C
23    IF(NREC.GE.440) GO TO 8
      IF(COMP.EQ.1) GO TO 4
```

```
      GO TO 5
C
C     >>>>>>   COMPRESSOR SHUT-OFF TIME ROUTINE   <<<<<<
C
4     CALL CURTIM(TEND,EDAY)
      GO TO 3
C
C     >>>>>>   NEW DATA DISK RECORD 1 SET-UP   <<<<<<
C
11    NREC=2
      POWER=0
      GO TO 12
C
C     >>>>>>   PROGRAM EXIT ROUTINE   <<<<<<
C
8     CALL DISABL
C
      TYPE=3
      CALL SAVE(TYPE,UNIT,NREC,NFRST)
      ENDFILE 9
C
      CALL MOTOR
C
      PAUSE P1
      PAUSE P2
C
      END
```

B.

```
C     ------------------------------------------------------------------
C     >>>>>>    HEAT PUMP DATA AQUISITION PROGRAM UTILITY LIBRARY   <<<<<<
C     ------------------------------------------------------------------
C
      SUBROUTINE CONSOL(UNIT,NREC,ZONE,EXIT,PINT)
C     CONSOLE PROCESSOR ROUTINE                         03/23/81
C
C
C                     INPUT CHARACTER DEFINITIONS
C
C           CONTROL  D (Z'04') = Data dump (analog & digital)
C           CONTROL  E (Z'05') = Exit from program
C           CONTROL  N (Z'0E') = New data disk
C           CONTROL  L (Z'0C') = List program status
C           CONTROL  T (Z'14') = Cycle & Daily data dump
C           CONTROL  V (Z'16') = Cycle scan save counter set
C
      INTEGER*1 UNIT,EXIT,K,MON,HR,MIN,SEC,ICON,SEASON
      INTEGER*1 MODE,SET,ZONE,PINT,COUNT,PDATA,ADATA
      INTEGER POWER,CYDIV,YR,DAY,DIG,RECORD
      DIMENSION VDIG(4),TITLE(16),E(4),P(6),R(6),STAGE(2)
C
      COMMON /ATOD/ADATA(528),AD1(16),AD2(16),COUNT
      COMMON /PARELL/PDATA(8),DIG(4)
      COMMON /SUM/TSUMC(4),TIMC(2),QSUMC(2),ESUMC(3),LDIG(4),
     -              TSUMD(4),TIMD(2),QSUMD(2),ESUMD(3),COP(2),SET
      COMMON /RDATA/RHO,PVS,WS,FLOW,CPA,QS,VER
      COMMON /MISC/CYDIV,POWER
C
      DATA P/'RHO','PVS','WS','FLOW','CPA','QS'/
      DATA STAGE/'HEAT','COOL'/
      DATA TITLE/' ','DP','TDPR','TDPO','PATM','TRET','TOUT',2*' ',
     -             'DT','TSUP','TSP3',4*' '/
      DATA VDIG/'COMP','HEAT','HEAT','COND'/
C
C     ---------------------------
C     PROGRAM VERSION NUMBER  5.5
C     ---------------------------
C
      VER=5.5
C
      EXIT=0
      ICON=INP(0).AND.Z'40'
      IF(ICON.EQ.0) RETURN
C
      SEASON=INP(Z'A7').AND.Z'80'
      SEASON=SEASON+1
      IF(SEASON.GT.1 .OR. SEASON.LE.0) SEASON=2
C
      ICON=INP(1).AND.Z'7F'
C
      IF(ICON.EQ.Z'04') GO TO 2
      IF(ICON.EQ.Z'05') GO TO 1
      IF(ICON.EQ.Z'0E') GO TO 2
      IF(ICON.EQ.Z'14') GO TO 2
      IF(ICON.EQ.Z'16') GO TO 8
      IF(ICON.NE.Z'0C') RETURN
C
C     >>>>>>   PROGRAM STATUS LIST ROUTINE   <<<<<<
```

```
C
2     CALL LINE
C
      CALL CLOCK(YR,MON,DAY,HR,MIN,SEC)
C
      WRITE(5,500) HR,MIN,SEC,MON,DAY,YR,VER
500   FORMAT(5X,'PROGRAM STATUS',8X,2(I2,':'),I2,' EST'
     *,4X,2(I2,'/'),I2,5X,'Program Version ',F3.1)
C
      CALL LINE
C
      RECORD=NREC-2
      MODE=INP(Z'24').AND.Z'03'
C
      WRITE(5,501) UNIT,RECORD,POWER
501   FORMAT(1X,//,15X,'    UNIT=',I3,'  RECORD=',I3,
     -'     POWER FAILURES=',I3)
C
      WRITE(5,505) PINT,ZONE,MODE
505   FORMAT(1X,//,24X,'INTERPT=',I3,'    ZONE=',I3,'    MODE=',I3)
C
      WRITE(5,506) STAGE(SEASON)
506   FORMAT(1X,//,33X,A4,'ING SEASON',//)
C
      IF(ICON.EQ.Z'04') GO TO 5
      IF(ICON.EQ.Z'14') GO TO 15
      IF(ICON.EQ.Z'OC') RETURN
C
C     >>>>>>   NEW DATA DISK LOG-IN ROUTINE   <<<<<<
C
4     ENDFILE 9
C
3     WRITE(5,502)
502   FORMAT(1X,//,10X,'Remove OLD data disk and insert NEW disk',/)
C
      WRITE(5,503)
503   FORMAT(1X,//,10X,'Is NEW data DISK inserted? ')
      READ(5,504) K
504   FORMAT(A1)
C
      CALL DISABL
C
      CALL RESET
C
      CALL OPEN(9,'PUMPDATADTA',1)
C
      READ(9,END=11,REC=1) UNIT,NREC
C
      GO TO 4
C
11    NREC=2
      POWER=0
      CALL SYSTEM
      IF(PINT.EQ.1) CALL ENABL
      RETURN
C
C     >>>>>>   REDUCED DATA OUTPUT ROUTINES   <<<<<<
C
5     CALL CALC(ZONE,MODE)
      CALL PSCAN
```

```
C
      R(1)=RHO
      R(2)=PVS
      R(3)=WS
      R(4)=FLOW
      R(5)=CPA
      R(6)=GS
C
      E(1)=DIG(1)
      E(2)=20.0*DIG(2)
      E(3)=20.0*DIG(3)
      E(4)=DIG(4)
C
      DO 10 I=1,16
      IF(I.GT.4) GO TO 6
      WRITE(5,507) I,AD1(I),TITLE(I),AD2(I),P(I),R(I),VDIG(I),DIG(I)
     -,E(I)
507   FORMAT('+',5X,I2,2(2X,F6.2,2X,A4),' = ',G10.4,4X,A4,2X,I4,2X
     -,G10.4,/)
      GO TO 10
C
6     IF(I.GT.6) GO TO 7
      WRITE(5,508) I,AD1(I),TITLE(I),AD2(I),P(I),R(I)
508   FORMAT('+',5X,I2,2(2X,F6.2,2X,A4),' = ',E11.5,/)
      GO TO 10
C
7     WRITE(5,509) I,AD1(I),TITLE(I),AD2(I)
509   FORMAT('+',5X,I2,2X,F6.2,2X,A4,2X,F6.2,/)
10    CONTINUE
      RETURN
C
C     >>>>>>   CYCLE AND DAILY DATA DUMP ROUTINE   <<<<<<
C
15    WRITE(5,510)
510   FORMAT(1X,//,33X,'CYCLE SUM DATA',//)
C
      DO 20 I=1,4
      IF(I.GT.2) GO TO 9
      WRITE(5,511) TSUMC(I),ESUMC(I),QSUMC(I),TIMC(I)
511   FORMAT('+',3X,'TSUM =',G10.3,' ESUM =',G10.3,' QSUM =',G10.3,
     -' TIMC =',G10.3,/)
      GO TO 20
C
9     IF(I.GT.3) GO TO 12
      WRITE(5,512) TSUMC(I),ESUMC(I)
512   FORMAT('+',3X,'TSUM =',G10.3,' ESUM =',G10.3,/)
      GO TO 20
C
12    WRITE(5,513) TSUMC(I)
513   FORMAT('+',3X,'TSUM =',G10.3,/)
20    CONTINUE
C
      WRITE(5,514)
514   FORMAT(1X,//,33X,'DAILY SUM DATA',//)
C
      DO 30 I=1,4
      IF(I.GT.2) GO TO 13
      WRITE(5,511) TSUMD(I),ESUMD(I),QSUMD(I),TIMD(I)
      GO TO 30
C
```

```
13    IF(I.GT.3) GO TO 14
      WRITE(5,512) TSUMD(I),ESUMD(I)
      GO TO 30
C
14    WRITE(5,513) TSUMD(I)
30    CONTINUE
      RETURN
C
C     >>>>>>   CYCLE SCAN DIVIDE NUMBER SET ROUTINE   <<<<<<
C
8     WRITE(5,517) CYDIV
517   FORMAT(1X,//,10X,'Current CYCLE divide number is = ',I3,//)
C
      WRITE(5,516)
516   FORMAT(11X,'Enter cycle divide number (every nTH) -> ')
19    READ(5,515,ERR=19) CYDIV
515   FORMAT(I3)
C
      IF(CYDIV.LE.0) CYDIV=10
      WRITE(5,517) CYDIV
      RETURN
C
1     EXIT=1
      RETURN
      END
C
C     ------------------------------------------------------------------------
C
      SUBROUTINE CYCLE(NREC,EOC)
C     ROUTINE PERFORMS HEAT PUMP CYCLE CALCULATIONS
C
      INTEGER*1 SET,PDATA,TIM1,MODE,UNIT,EOC,TYPE,TDATA
      INTEGER DAY,DDAY,TDAY,EDAY,DIG
C
      COMMON /PARELL/PDATA(8),DIG(4)
      COMMON /DATE/TDATA(9)
      COMMON /SUM/TSUMC(4),TIMC(2),QSUMC(2),ESUMC(3),LDIG(4),
     -            TSUMD(4),TIMD(2),QSUMD(2),ESUMD(3),COP(2),SET
      COMMON /HOUSE/UNIT,AREA(3),FAN(3)
      COMMON /TIM/TIM1(9),TSTART,TEND,TLAST,TDAY,EDAY,LDAY
C
      CALL CURTIM(T,DAY)
C
      SEASON=INP(Z'A7').AND.Z'80'
C
C     >>>>>>   TEST FOR END OF DEFROST CYCLE (EOC=1)   <<<<<<
C
      IF(EOC.EQ.0) GO TO 5
C
C     >>>>>>   END OF DEFROST CYCLE TIME-ON CALCULATION   <<<<<<
C
      T2=TIM1(1)+10*TIM1(2)+60*(TIM1(3)+10*TIM1(4))
      T2=T2+3600.*(TIM1(5)+10*TIM1(6))
      DDAY=TIM1(7)+10*TIM1(8)+100*TIM1(9)
C
      TIMC(1)=T-TSTART+(DAY-TDAY)*86400.
      TIMC(2)=TIMC(2)+T+(DAY-DDAY)*86400.-T2
      GO TO 6
C
5     TIMC(1)=TEND-TSTART+(EDAY-TDAY)*86400.
```

```
C
6     TSTART=T
      TDAY=DAY
      TLAST=TSTART
      LDAY=DAY
C
C     >>>>>>   ELECTRIC CONSUMPTION CALCULATION   <<<<<<
C
      CALL ENERGY
C
      ESUMC(2)=FAN(UNIT)*TIMC(1)/3600.
      ESUMC(3)=ESUMC(3)-ESUMC(2)
      IF(SEASON.NE.0 .OR. ESUMC(3).LT.0.0) ESUMC(3)=0.0
C
C     >>>>>>   UPDATE DAILY SUMS    <<<<<<
C
      DO 10 I=1,3
      IF(I.GT.2) GO TO 1
      QSUMD(I)=QSUMD(I)+QSUMC(I)
      TIMD(I)=TIMD(I)+TIMC(I)
1     ESUMD(I)=ESUMD(I)+ESUMC(I)
10    CONTINUE
C
      TSUMD(2)=TSUMD(2)+TSUMC(2)
      TSUMD(4)=TSUMD(4)+TSUMC(4)
C
C     >>>>>>   CYCLE CALCULATIONS   <<<<<<
C
      COP(1)=0.0
      NDUM=0
C
      DENOM=(ESUMC(1)+ESUMC(2)+ESUMC(3))*3.413
      IF(DENOM.EQ.0.00) GO TO 3
      COP(1)=(QSUMC(1)+QSUMC(2))/DENOM
C
3     DENOM=TIMC(1)-TIMC(2)
      IF(DENOM.EQ.0.00) GO TO 4
      TSUMC(2)=TSUMC(2)/DENOM
      TSUMC(4)=TSUMC(4)/DENOM
C
4     TYPE=1
C
      CALL SAVE(TYPE,UNIT,NREC,NDUM)
C
      EOC=0
      SET=1
      CALL INIT
C
      RETURN
      END
C
C     ------------------------------------------------------------------------
C
      SUBROUTINE CHECK(NREC,ZONE,NFRST,EXIT,PINT)
C     THIS ROUTINES CHECKS FOR 1/2 HR,DAILY, & CONSOLE REQUIREMENTS
C
      INTEGER*1 TDATA,UNIT,ZONE,TYPE,MODE,SET,NFRST,EXIT
      INTEGER*1 ADATA,COUNT,PINT,PDATA
      INTEGER HR,DAY,DIG
C
```

```
      COMMON /DATE/TDATA(9)
      COMMON /PARELL/PDATA(8),DIG(4)
      COMMON /ATOD/ADATA(528),AD1(16),AD2(16),COUNT
      COMMON /SUM/TSUMC(4),TIMC(2),QSUMC(2),ESUMC(3),LDIG(4),
     -             TSUMD(4),TIMD(2),QSUMD(2),ESUMD(3),COP(2),SET
      COMMON /HOUSE/UNIT,AREA(3),FAN(3)
C
      DATA NMIN,NHR,NDAY/3*1/
C
C     >>>>>>   ENERGY COUNT UPDATE   <<<<<<
C
      CALL ENERGY
C
C     >>>>>>   CHECK CURRENT TIME FOR HALF HOUR   <<<<<<
C
      CALL TIME
C
      MIN=TDATA(3)+TDATA(4)*10
      HR=TDATA(5)+TDATA(6)*10
      DAY=TDATA(7)+TDATA(8)*10+TDATA(9)*100
C
      IF(MIN.EQ.0 .OR. MIN.EQ.30) GO TO 1
      GO TO 3
C
C     >>>>>>   HALF HOUR CALCULATIONS   <<<<<<
C
1     IF(MIN.EQ.NMIN) GO TO 3
C
      NMIN=MIN
      NHR=HR
C
      CALL CALC(ZONE,MODE)
C
      TSUMD(1)=TSUMD(1)+AD2(7)
      TSUMD(3)=TSUMD(3)+AD2(4)
C
      TYPE=3
C
      CALL SAVE(TYPE,UNIT,NREC,NFRST)
C
      IF(MIN.EQ.0 .AND. HR.EQ.0) GO TO 2
      GO TO 3
C
C     >>>>>>   DAILY CALCULATIONS   <<<<<<
C
2     IF(DAY.EQ.NDAY) GO TO 3
      NDAY=DAY
C
      TSUMD(1)=TSUMD(1)/48.0
      TSUMD(3)=TSUMD(3)/48.0
C
      DENOM=TIMD(1)-TIMD(2)
      IF(DENOM.EQ.0.00) GO TO 5
      TSUMD(2)=TSUMD(2)/DENOM
      TSUMD(4)=TSUMD(4)/DENOM
C
5     COP(2)=0.00
      DENOM=(ESUMD(1)+ESUMD(2)+ESUMD(3))*3.413
      IF(DENOM.EQ.0.00) GO TO 4
      COP(2)=(QSUMD(1)+QSUMD(2))/DENOM
```

```
C
4       TYPE=2
C
        CALL SAVE(TYPE, UNIT, NREC, NFRST)
C
        NFRST=0
        SET=2
        CALL INIT
C
3       CALL CONSOL(UNIT, NREC, ZONE, EXIT, PINT)
C
        RETURN
        END
C
C       ------------------------------------------------------------------
C
        SUBROUTINE CALC(ZONE, MODE)
C       ROUTINE CALCULATES REDUCED DATA
C
        INTEGER*1 UNIT, TIM1, ZONE, MODE, ADATA, COUNT, M
        INTEGER DAY, TDAY, EDAY
C
        COMMON /ATOD/ADATA(528), AD1(16), AD2(16), COUNT
        COMMON /HOUSE/UNIT, AREA(3), FAN(3)
        COMMON /TIM/TIM1(9), TSTART, TEND, TLAST, TDAY, EDAY, LDAY
        COMMON /RDATA/RHO, PVS, WS, FLOW, CPA, QS, VER
C
        CALL ANALOG
C
C       >>>>>>   TIME SINCE LAST SCAN CALCULATION   <<<<<<
C
        CALL CURTIM(T, DAY)
C
        DT=T-TLAST+(DAY-LDAY)*86400.
C
C       >>>>>>   REDUCED DATA COMPUTATIONS   <<<<<<
C
        MODE=INP(Z'24').AND.Z'03'
C
        ARG=15.4638-7284.0/(AD2(3)+392.0)
        PVS=EXP(ARG)
C
        WS=0.00
        DENOM=AD2(5)-PVS
        IF(DENOM.EQ.0.00) GO TO 1
        WS=0.622*PVS/DENOM
C
        RHO=(29.92/AD2(5))*.02521*(459.6+AD2(6))*(1.+1.6078*WS)
C
1       FLOW=0.00
        IF(AD2(2).LE.0.01) GO TO 2
        ARG=AD2(2)*RHO/(1.0+WS)
        IF(ARG.LE.0.00) GO TO 2
        FLOW=1096.0*AREA(UNIT)*SQRT(ARG)
C
2       CPA=0.24+0.444*WS
        M=ZONE+1
C
        QS=0.00
        DENOM=60.0*RHO
```

```
      IF(DENOM.EQ.0.00) RETURN
      QS=FLOW*CPA*AD2(10)*DT/DENOM
C
      RETURN
      END
C
C     ------------------------------------------------------------------
C
      SUBROUTINE SAVE(TYPE,UNIT,NREC,NFRST)
C     ROUTINE BUFFERS DATA FOR SUBSEQUENT STORAGE ON DATA DISK
C
      INTEGER*1 TYPE,TDATA,MODE,UNIT,ADATA,COUNT,SET,A(24),NFRST
      INTEGER*1 PDATA
      INTEGER DIG,CYDIV,POWER
      DIMENSION B(26)
C
      COMMON /PARELL/PDATA(8),DIG(4)
      COMMON /ATOD/ADATA(528),AD1(16),AD2(16),COUNT
      COMMON /DATE/TDATA(9)
      COMMON /SUM/TSUMC(4),TIMC(2),QSUMC(2),ESUMC(3),LDIG(4),
     -            TSUMD(4),TIMD(2),QSUMD(2),ESUMD(3),COP(2),SET
      COMMON /RDATA/RHO,PVS,WS,FLOW,CPA,QS,VER
      COMMON /MISC/CYDIV,POWER
C
      DATA N/0/
C
      K1=N*12
      K2=N*13
C
      MODE=INP(Z'24').AND.Z'03'
C
      N=N+1
C
      A(K1+1)=TYPE
C
      DO 10 I=2,10
      L=K1+I
      A(L)=TDATA(I-1)
10    CONTINUE
C
      A(K1+11)=MODE
      A(K1+12)=NFRST
C
      GO TO (1,2,3),TYPE
C
C     >>>>>>   PERIODIC SCAN DATA ARRAY SET-UP   <<<<<<
C
      B(K2+1)=AD2(10)
      B(K2+2)=FLOW
      B(K2+3)=QS
      B(K2+4)=AD2(6)
      B(K2+5)=AD2(7)
      B(K2+6)=AD2(11)
      B(K2+7)=AD2(2)
      B(K2+8)=AD2(3)
      B(K2+9)=AD2(4)
      B(K2+10)=AD2(5)
      B(K2+11)=0.0
      IF(UNIT.EQ.3) B(K2+11)=AD2(12)
      B(K2+12)=QSUMC(1)
```

```
      B(K2+13)=DIG(1)
      GO TO 4
C
C     >>>>>>   CYCLE DATA ARRAY SET-UP   <<<<<<
C
1     B(K2+1)=QSUMC(1)
      B(K2+2)=QSUMC(2)
      B(K2+3)=ESUMC(1)
      B(K2+4)=ESUMC(2)
      B(K2+5)=ESUMC(3)
      B(K2+6)=DIG(2)
      B(K2+7)=TSUMC(2)
      B(K2+8)=DIG(3)
      B(K2+9)=TSUMC(4)
      B(K2+10)=TIMC(1)
      B(K2+11)=TIMC(2)
      B(K2+12)=COP(1)
      B(K2+13)=AD2(7)
      GO TO 4
C
C     >>>>>>   DAILY DATA ARRAY SET-UP   <<<<<<
C
2     B(K2+1)=QSUMD(1)
      B(K2+2)=QSUMD(2)
      B(K2+3)=ESUMD(1)
      B(K2+4)=ESUMD(2)
      B(K2+5)=ESUMD(3)
      B(K2+6)=TSUMD(1)
      B(K2+7)=TSUMD(2)
      B(K2+8)=TSUMD(3)
      B(K2+9)=TSUMD(4)
      B(K2+10)=TIMD(1)
      B(K2+11)=TIMD(2)
      B(K2+12)=COP(2)
      B(K2+13)=TIMD(1)/(3600.0*24.0)
      GO TO 4
C
C     >>>>>>   ONE-HALF HOUR DATA ARRAY SET-UP   <<<<<<
C
3     B(K2+1)=AD2(7)
      B(K2+2)=AD2(4)
      B(K2+3)=AD2(5)
      B(K2+4)=FLOW
      B(K2+5)=AD2(10)
      B(K2+6)=AD2(3)
      B(K2+7)=0.0
      IF(UNIT.EQ.3) B(K2+7)=AD2(12)
      B(K2+8)=DIG(1)
      B(K2+9)=DIG(2)
      B(K2+10)=DIG(3)
      B(K2+11)=DIG(4)
      B(K2+12)=AD2(6)
      B(K2+13)=AD2(11)
C
4     IF(N.EQ.1) RETURN
C
      WRITE(9,REC=NREC) (A(I),I=1,12),(B(I),I=1,13),(A(I),I=13,24)
     -,(B(I),I=14,26)
      WRITE(9,REC=1) UNIT,NREC,VER,CYDIV,POWER
C
```

```
      NREC=NREC+1
      N=0
C
      CALL MOTOR
C
      RETURN
      END
C
C     -------------------------------------------------------------------
C
      SUBROUTINE CURTIM(T,DAY)
C     THIS ROUTINE COMPUTES THE CURRENT TIME AND CURRENT DAY
C
C       >>>>>>   VARIABLE DEFINITIONS   <<<<<<
C
C           T        CURRENT TIME IN SECONDS
C           DAY      CURRENT JULIAN DAY
C           TDATA    TIME RAW DATA ARRAY
C
      INTEGER*1 TDATA
      INTEGER DAY
C
      COMMON /DATE/TDATA(9)
C
      CALL TIME
C
      T=TDATA(1)+10*TDATA(2)+60*(TDATA(3)+10*TDATA(4))
      T=T+3600.0*(TDATA(5)+10*TDATA(6))
C
      DAY=TDATA(7)+10*TDATA(8)+100*TDATA(9)
C
      RETURN
      END
C
C     -------------------------------------------------------------------
C
      SUBROUTINE ENERGY
C     ROUTINE COMPUTES THE ELECTRICAL AND LATENT ENERGY CONSUMED
C
C     THIS ROUTINE WILL UPDATE CYCLE ENERGY SUMS (ARRAY ESUMC)
C     EVERY TIME IT IS CALLED
C
C          >>>>>>   VARIABLE DEFINITIONS   <<<<<<
C
C       ESUMC(1)  WATT-HRS  Compressor enery consumption
C       ESUMC(3)  WATT-HRS  Suppl. heaters energy consumption
C       QSUMC(2)  WATT-HRS  Latent heat
C
      INTEGER*1 SET,PDATA
      INTEGER DIG
C
      COMMON /PARELL/PDATA(8),DIG(4)
      COMMON /SUM/TSUMC(4),TIMC(2),QSUMC(2),ESUMC(3),LDIG(4),
     -            TSUMD(4),TIMD(2),QSUMD(2),ESUMD(3),COP(2),SET
C
      CALL PSCAN
C
C     >>>>>>   COUNTER OVERRANGE,COMPENSATION, AND UPDATE ROUTINE   <<<<<<
C
      DO 10 I=1,4
```

```
1     IF(DIG(I).GE.LDIG(I)) GO TO 10
      DIG(I)=DIG(I)+4096
      GO TO 1
10    CONTINUE
C
C     >>>>>>   UPDATE CYCLE ENERGY SUMS   <<<<<<
C
      ESUMC(1)=ESUMC(1)+(DIG(1)-LDIG(1))
      QSUMC(2)=QSUMC(2)+2.334*(DIG(4)-LDIG(4))
      ESUMC(3)=ESUMC(3)+20.0*(DIG(2)+DIG(3)-LDIG(2)-LDIG(3))
C
      DO 20 I=1,4
      LDIG(I)=DIG(I)
20    CONTINUE
C
      RETURN
      END
```

B.

```
;        PROGRAM PARALLEL                  2/6/80
;        SCAN SUBROUTINE FOR CROMEMCO 8PIO
;
         ENTRY   PSCAN
;
         COM     PARELL
PDATA:   DS      8
DIG:     DS      8                ;INTEGER ARRAY DIG(4)
         REL
;
PSCAN:   PUSH    AF
         PUSH    BC
         PUSH    HL
;
         LD      A,0FFH
         OUT     84H,A            ;UPDATE PARALLEL PORTS...84H=TUART PORT B
;
         XOR     A                ;ZERO ACCUMULATOR
         OUT     84H,A            ;LATCH ALL 8PIO PORTS
;
         LD      HL,PDATA
         LD      B,8
         LD      C,0A0H           ;POINT TO FIRST PARALLEL PORT
;
PIN:     INI
         JR      Z,STUFF          ;EXIT IF DONE (8 DATA POINTS COLLECTED)
;
         INC     C                ;POINT TO NEXT PORT
         JR      PIN
;
;*********************************************************************
;        >>>>>>   INTEGER ARRAY DIG(4) SET-UP ROUTINE   <<<<<<
;*********************************************************************
;
STUFF:   LD      HL,PDATA         ;POINT TO DATA PORT 0
         LD      A,(HL)
         LD      (DIG+2),A        ;LOAD LOW BYTE OF DIG(2)
;
         INC     HL               ;POINT TO DATA PORT 1
         LD      A,(HL)
         LD      (DIG),A          ;LOAD LOW BYTE OF DIG(1)
;
         INC     HL               ;POINT TO DATA PORT 2
         XOR     A                ;ZERO ACCUMULATOR
         RRD                      ;ROTATE LOW NIBBLE OF DATA PORT 2 TO ACC.
         LD      (DIG+5),A        ;LOAD HIGH BYTE OF DIG(3)
;
         RRD                      ;ROTATE HIGH NIBBLE OF DATA PORT 2 TO ACC.
         LD      (DIG+1),A        ;LOAD HIGH BYTE OF DIG(1)
;
         INC     HL               ;POINT TO DATA PORT 3
         RRD                      ;ROTATE LOW NIBBLE OF DATA PORT 3 TO ACC.
         LD      (DIG+7),A        ;LOAD HIGH BYTE OF DIG(4)
;
         RRD                      ;ROTATE HIGH NIBBLE OF DATA PORT 3 TO ACC.
         LD      (DIG+3),A        ;LOAD HIGH BYTE OF DIG(2)
;
         INC     HL               ;POINT TO DATA PORT 4
```

```
        LD      A,(HL)
        LD      (DIG+6),A       ;LOAD LOW BYTE OF DIG(4)
;
        INC     HL              ;POINT TO DATA PORT 5
        LD      A,(HL)
        LD      (DIG+4),A       ;LOAD LOW BYTE OF DIG(3)
;
EXIT:   POP     HL
        POP     BC
        POP     AF
        RET
        END
```

B.

```
      SUBROUTINE ANALOG
C     DATA AQUISITION ROUTINE FOR CROMEMCO A/D CONVERTER CARD     5/27/80
C
C     >>>>>>    VARIABLE DEFINITIONS   <<<<<<
C
C       COUNT           ANALOG SCAN COUNT  (33=max)
C       ADATA           ANALOG PORT RAW DATA
C       AD1             ANALOG RAW DATA (volts)
C       AD2             ANALOG REDUCED DATA
C       M               SENSOR CALIBRATION SLOPES (Y=MX+B)
C       B               SENSOR CALIBRATION INTERCEPTS
C       SEASON          HEATING-COOLING SEASON INDICATOR (0=heating)
C
      INTEGER*1 COUNT, ADATA, SEASON
      REAL M(16)
      DIMENSION B(16)
C
      COMMON /ATOD/ADATA(528), AD1(16), AD2(16), COUNT
C
      DATA M/0., .038, 14., 26., .9, 9., 20., 0., 0., 1., 2*26., 4*0. /
      DATA B/0., .105, 55., 25., 29.25, 72.5, 50., 0., 0., 0., 2*115., 4*0. /
C
      CALL SETUP
C
      SEASON=INP(Z'A7').AND.Z'80'
      IF(SEASON.EQ.0) GO TO 1
C
      M(11)=10.0
      M(12)=10.0
      B(11)=70.0
      B(12)=70.0
      GO TO 3
C
1     M(11)=26.0
      M(12)=26.0
      B(11)=115.0
      B(12)=115.0
C
3     N=1
C
2     IF(N.GT.COUNT) GO TO 2
C
      DO 10 I=1,16
      J=I+(N-1)*16
      AD1(I)=AD1(I)+ADATA(J)
10    CONTINUE
C
      N=N+1
      IF(N.LT.34) GO TO 2
C
C     >>>>>>   REDUCED DATA CALCULATION ROUTINE   <<<<<<
C
      DO 20 I=1,16
      AD1(I)=AD1(I)*2.0/(100.0*(N-1))
      AD2(I)=AD1(I)*M(I)+B(I)
20    CONTINUE
C
      IF(SEASON.EQ.Z'80') GO TO 4
      DV=0.6134*AD1(10)+1.5335
```

```
      AD2(10)=44.4919*DV-0.98659*DV**2
      RETURN
C
4     DV=0.5605+0.2242*AD1(10)
      AD2(10)=43.9624*DV+1.2911*DV**2
C
      RETURN
      END
```

B.

```
C      ------------------------------------------------------------------------
C
C      >>>>>>   PERKIN ELMER CRT UTILITY PROGRAM LIBRAY   <<<<<<
C
C      ------------------------------------------------------------------------
C
       SUBROUTINE LINE
C      ROUTINE PUTS A LINE (DASHES) ON CRT          5/7/80
       INTEGER*1 L(80)
C
       DATA L/80*Z'2D'/
C
       WRITE(5,500) L
500    FORMAT(1X,80A1)
C
       RETURN
       END
C
C      ------------------------------------------------------------------------
C
       SUBROUTINE CLEAR
C
       INTEGER*1 L(22)
C
       DATA L/22*Z'00'/
C
       L(1)=Z'1B'
       L(2)=Z'4B'
C
       WRITE(5,500) L
500    FORMAT(1X,22A1)
C
       RETURN
       END
C
C      ------------------------------------------------------------------------
C
       SUBROUTINE CURSOR(X,Y)
       INTEGER*1 X,Y,L(6),NX,NY
C
       NY=Y+31
       NX=X+31
C
       L(1)=Z'1B'
       L(2)=Z'58'
       L(3)=NY
C
       L(4)=Z'1B'
       L(5)=Z'59'
       L(6)=NX
C
       WRITE(5,500) L
500    FORMAT(1X,6A1)
C
       RETURN
       END
C
C      ------------------------------------------------------------------------
C
```

```
      SUBROUTINE SKIP(DIR,NSKIP)
C
C     ROUTINE MOVES CRT CURSOR UP OR DOWN SCREEN
C
C        DIR = 0     MOVE CURSOR UP
C        DIR = 1     MOVE CURSOR DN
C      NSKIP =       NUMBER OF LINES TO BE SKIPPED
C
      INTEGER*1 DIR,NSKIP,L(2)
C
      L(1)=Z'1B'
      L(2)=Z'41'+DIR
C
      DO 10 I=1,NSKIP
      WRITE(5,500) L
500   FORMAT('+',2A1)
10    CONTINUE
C
      RETURN
      END
```

B.

```
      PROGRAM REDUCE
C     DATA REDUCTION PROGRAM FOR HEAT PUMP PROJECT      7/19/80
C
C     >>>>>>   LOGICAL UNIT NUMBER ASSIGNMENTS   <<<<<<
C
C        AS  5 = CONSOLE
C        AS  7 = PROGRAM MESSAGE FILE
C        AS  8 = DATA FILE  (PUMPDATA.DTA)
C
      INTEGER*1 TYPE,DTYPE(4,6),UNIT,NA(13),NB(13),NDATA(200,13)
      INTEGER CYDIV,POWER,TOTREC
      DIMENSION TITLE(13,4),RA(13),RB(13),RDATA(200,13)
C
      CALL OPEN(7,'DATANAMEMSG',0)
C
      DO 10 I=1,13
      READ(7,700,END=6) (TITLE(I,J),J=1,4)
700   FORMAT(4(1X,A4))
10    CONTINUE
C
      DO 50 I=1,4
      READ(7,701) (DTYPE(I,J),J=1,6)
701   FORMAT(1X,6A1)
50    CONTINUE
C
1     CALL CLEAR
C
      CALL LINE
C
      WRITE(5,509)
509   FORMAT(30X,'HEAT PUMP FIELD DATA')
C
      CALL LINE
C
      WRITE(5,500)
500   FORMAT(1X,///,20X,'DATA TYPE',/,10X,'0=SCAN',/,10X,'1=CYCLE',
     -/,10X,'2=DAILY',/,10X,'3=1/2 HOUR',//,5X,'Enter data type -> ')
      READ(5,501) TYPE
501   FORMAT(I1)
C
      IF(TYPE.GT.3) GO TO 1
C
      CALL RESET
C
      CALL OPEN(8,'PUMPDATADTA',1)
C
      READ(8,END=6,REC=1) UNIT,TOTREC,VER,CYDIV,POWER
C
      IF(VER.LT.0.00 .OR. VER.GT.9.90) VER=0.0
C
      NREC=1
2     N=1
C
3     NREC=NREC+1
      IF(N.GE.200) GO TO 4
C
      READ(8,END=4,REC=NREC) (NA(I),I=1,12),(RA(I),I=1,13)
     -,(NB(I),I=1,12),(RB(I),I=1,13)
C
```

```
      IF(NA(1).NE.TYPE) GO TO 5
C
      DO 20 I=1,13
      NDATA(N,I)=NA(I)
      RDATA(N,I)=RA(I)
20    CONTINUE
      N=N+1
C
5     IF(NB(1).NE.TYPE) GO TO 3
C
      DO 30 I=1,13
      NDATA(N,I)=NB(I)
      RDATA(N,I)=RB(I)
30    CONTINUE
      N=N+1
      GO TO 3
C
4     N=N-1
      IF(N.GT.200) N=200
C
      CALL LINE
C
      WRITE(5,504) UNIT,NREC,VER
504   FORMAT(3X,'UNIT =',I2,3X,'TOTAL # of records on FILE=',I4
     -,3X,'Aquisition program ver. = ',F3.1)
C
      CALL LINE
C
      WRITE(5,511) TYPE,(DTYPE(TYPE+1,J),J=1,6),(DTYPE(TYPE+1,J),
     -J=1,6),N
511   FORMAT(3X,'Data type =',I2,' (',6A1,' data)',3X,
     -'Number of ',6A1,' records on file=',I4)
C
      CALL LINE
C
      WRITE(5,512) CYDIV,POWER
512   FORMAT(3X,'Cycle divide number=',I3,3X,'Number of POWER ',
     -'FAILURES=',I3)
C
      CALL LINE
C
      IF(N.EQ.0) GO TO 7
C
C     >>>>>>   OUTPUT REDUCED DATA TO CONSOLE   <<<<<<
C
      DO 40 I1=1,N,5
      I2=I1+4
      IF(I2.GT.N) I2=N
      WRITE(5,503)
503   FORMAT(1X,//,10X)
C
      WRITE(5,505) (NDATA(I,11),I=I1,I2)
505   FORMAT(12X,5(1X,'MODE = ',I1,4X))
C
      WRITE(5,506) (NDATA(I,10),NDATA(I,9),NDATA(I,8),I=I1,I2)
506   FORMAT(12X,5(1X,'DAY = ',3I1,3X))
C
      WRITE(5,507) (NDATA(I,7),NDATA(I,6),NDATA(I,5),NDATA(I,4)
     -,NDATA(I,3),NDATA(I,2),I=I1,I2)
507   FORMAT(12X,5(1X,2I1,':',2I1,':',2I1,4X))
```

```
C
      WRITE(5,510)
510   FORMAT(10X)
C
      DO 40 J=1,13
      WRITE(5,502) J,TITLE(J,TYPE+1),(RDATA(I,J),I=I1,I2)
502   FORMAT(2X,I2,2X,A4,2X,5(G10.4,3X))
40    CONTINUE
C
      IF(N.GE.199) GO TO 2
7     ENDFILE 8
      GO TO 1
C
6     CALL CLEAR
      WRITE(5,508)
508   FORMAT(1X,//,10X,'ERROR!',//,10X,'Default drive set incorrectly OR
     - ',/,10X,'Data reduction disk in wrong drive',//)
C
      END
```

B.

```
;        PROGRAM SCAN              7/19/80
;        HEAT PUMP DATA AQUISITION SCAN ROUTINE
;
         ENTRY    SERV,DISABL,ENABL,TIME,SYSTEM,INIT,SETUP,MOTOR,RESET
;
         COM      ATOD
ADATA:   DS       528                 ;ANALOG RAW DATA ARRAY
AD1:     DS       64                  ;REDUCED DATA ANALOG DATA ARRAY
AD2:     DS       64                  ;REDUCED ANALOG DATA ARRAY
COUNT:   DS       1                   ;ANALOG SCAN COUNT (33 = max.)
;
         COM      STATUS
STAT:    DS       1                   ;SCAN OR NO-SCAN STATUS (1=scanning)
ICNT:    DS       1                   ;PERIODIC INTERRUPT COUNTER (max count=76)
NSCAN:   DS       1                   ;TOTAL SCANS PER CYCLE (255 MAX.)
DEFRST:  DS       1                   ;DEFROST STATUS BYTE; 2=DEFR. ON; fan off
COMP:    DS       1                   ;COMPRESSOR STATUS (0 = on)
MODE:    DS       1                   ;SYSTEM STATUS BIT; FAN AND DEFRST
EOC:     DS       1                   ;END OF CYCLE INDICATOR (1 = end of def. cycle)
NFRST:   DS       1                   ;NUMBER OF DEFROST CYCLES
PINT:    DS       1                   ;PERIODIC INTERRUPT STATUS (1 =interrupt active)
ZONE:    DS       1                   ;SCAN TIME ZONE INDICATOR
DIVIDE:  DS       1                   ;TSCAN DIVIDE DOWN COUNTER
;
         COM      DATE
TDATA:   DS       9                   ;TIME RAW DATA ARRAY
;
         COM      SUM
CYCLE:   DS       52                  ;CYCLE SUM DATA ARRAY
DAILY:   DS       44                  ;DAILY SUM DATA ARRAY
COP1:    DS       4                   ;CYCLE COP
COP2:    DS       4                   ;DAILY COP
SET:     DS       1                   ;SUM DATA INITIALIZATION INDICATOR
;
         COM      TIM
TIM1:    DS       9                   ;DEFROST START TIME
OTHER:   DS       18                  ;OTHER CYCLE TIME INFORMATION
;
         REL
;
SETPT:   DB       14,38,74,74         ;TIME ZONE SET POINTS
DIVCNT:  DB       1,3,6,30            ;TSCAN DIVIDE DOWN COUNTER VALUES
;
TENCNT:  DB       10                  ;ONE SECOND INTERRUPT DIVIDE DOWN COUNTER
DTAB:    DW       ADATA               ;ANALOG DATA TABLE ADDRESS POINTER
;
;****************************************************************
;        >>>>>>   INTERRUPT   SERVICE   ROUTINE   <<<<<<
;****************************************************************
;
;        ------------------------------------------------------------
;        >>> PERIODIC OR ANALOG INTERRUPT DETERMINATION ROUTINE <<<
;        ------------------------------------------------------------
;
SERV:    DI
         PUSH     AF
```

```
        PUSH    BC
        PUSH    DE
        PUSH    HL
;
        LD      A,(COUNT)
        CP      A,33            ;IS THIS A PERIODIC INTERRUPT?
        JR      Z,BEGIN         ;Z=PERIODIC INTERRUPT
;
;       ------------------------------------------
;       >>>>>>   ANALOG  DATA  INPUT  ROUTINE  <<<<<<
;       ------------------------------------------
;
ASCAN:  LD      HL,(DTAB)
        LD      B,16
        LD      C,10H
;
INP1:   INI
        JR      Z,EXIT1         ;EXIT IF ALL 16 PORT DATA COLLECTED
;
        INC     C
        JR      INP1            ;POINT TO NEXT ANALOG PORT
;
EXIT1:  LD      A,(COUNT)
        INC     A
        LD      (COUNT),A       ;UPDATE A/D DATA POINTS COUNTER
        CP      A,33            ;IS ACCUM. (COUNT) EQUAL TO 33?
        JR      Z,EXIT2         ;ALL DONE IF ZERO (COUNT=33)
;
        LD      (DTAB),HL       ;UPDATE DATA POINTER
;
EXIT3:  POP     HL
        POP     DE
        POP     BC
        POP     AF
        EI
        RET
;
;*******************************************************
;       >>>>>>   SCAN COMPLETION EXIT ROUTINE   <<<<<<
;*******************************************************
;
EXIT2:  XOR     A
        OUT     0CFH,A          ;DISABLE CLOCK INTERRUPTS
;
        INC     A               ;SET A=1
        LD      (PINT),A        ;SET INTERRUPT STATUS (1=interrupt active)
        LD      (STAT),A        ;SET SCAN STATUS (1=scanning completed)
;
        LD      HL,NSCAN        ;POINT TO ANALOG SCAN COUNTER
        INC     (HL)            ;INCREMENT ANALOG SCAN COUNTER
;
        LD      A,14H
        OUT     0CFH,A          ;SET INTERRUPT TO 1sec
        JR      EXIT3           ;EXIT AND ENABLE INTERRUPT
;
EXIT4:  POP     HL
        POP     DE
        POP     BC
        POP     AF
        RET
```

```
;
;*****************************************************************
;        >>>>>>   PERIODIC INTERRUPT SERVICE ROUTINE   <<<<<<
;*****************************************************************
;
;
;        ----------------------------------------
;        >>>   COMPRESSOR STATUS ROUTINE   <<<
;        ----------------------------------------
;
BEGIN:   IN      A,24H             ;INPUT COMPRESSOR-OUTDOOR FAN STATUS
         AND     A,1               ;MASK OFF ALL BUT COMPRESSOR BIT
         LD      (COMP),A          ;UPDATE COMPRESSOR STATUS
         JR      Z,DEF             ;Z=COMPRESSOR STILL ON : CONTINUE
;
;        >>> COMPRESSOR TURN-OFF <<<
;
         XOR     A
         LD      (PINT),A          ;RESET INTERRUPT STATUS (0=no interrupts)
;
         LD      A,(DEFRST)
         CP      A,2               ;WAS COMPRESSOR IN DEFRST MODE?
         JR      NZ,EXIT4          ;NZ=DEFROST OFF : THEN STOP EVERYYTHING
;
         LD      A,1
         LD      (EOC),A           ;SET EOC SINCE DEFROST HAS GONE OFF WITH COMP.
;
         LD      HL,NFRST          ;POINT TO DEFROST COUNTER
         INC     (HL)              ;INCREMENT DEFROST COUNTER
         JR      EXIT4             ;EVERYTHING OFF SO EXIT (STOP INTERRUPT)
;
;        -------------------------------------------------------
;        >>>   DEFROST MODE AND END OF CYCLE STATUS ROUTINE <<<
;        -------------------------------------------------------
;
DEF:     LD      A,(DEFRST)
         LD      B,A
         IN      A,24H             ;INPUT COMPRESSOR-FAN STATUS
         AND     A,02H             ;MASK OFF ALL BUT DEFRST
         CP      A,B               ;SEE IF CHANGE
         JR      Z,BEG2            ;Z=NO CHANGE
;
CHANG1:  LD      (DEFRST),A        ;SAVE NEW DEFRST VALUE
         OR      A
         JR      Z,CHANG3          ;Z=UNIT NOT IN DEFROST MODE (0=fan on)
;
;        >>>   START DEFROST MODE   <<<
;
         CALL    TIME
;
         LD      BC,9
         LD      HL,TDATA          ;POINT TO TIME DATA
         LD      DE,TIM1           ;POINT TO DEFROST START TIME ARRAY
         LDIR                      ;SET DEF. START ARRAY TO TIME
         JR      CHANG2
;
;        >>>   END OF DEFROST CYCLE   <<<
;
CHANG3:  LD      A,01H
         LD      (EOC),A           ;SET EOC (1 = end of defrost cycle)
```

```
;
        LD      HL,NFRST        ;POINT TO DEFROST COUNTER
        INC     (HL)            ;INCREMENT DEFROST COUNTER
;
CHANG2: XOR     A               ;ZERO ACCUMULATOR
        LD      (ICNT),A        ;RESET PERIODIC INTERRUPT COUNTER
        LD      (ZONE),A        ;RESET TIME ZONE
;
        INC     A               ;SET A=1
        LD      (DIVIDE),A      ;SET DIVIDE=1 TO START 10sec PERIODIC SCANS
        LD      (TENCNT),A      ;SET TENCNT TO INTIATE SCAN
;
;       ----------------------------------------------------------
;       >>>>>>   INTERRUPT DIVIDE DOWN AND CHECK ROUTINE   <<<<<<
;       ----------------------------------------------------------
;
BEG2:   LD      A,(TENCNT)
        DEC     A               ;DECREMENT DIVIDE DOWN COUNTER
        LD      (TENCNT),A      ;UPDATE DIVIDE DOWN COUNTER
        JP      NZ,EXIT3        ;NZ=CONTINUE 1sec INTERRUPTS
;
        LD      A,10
        LD      (TENCNT),A      ;INITIALIZE TENCNT TO 10 (divide by ten)
;
;       -------------------------------------------------------------
;       >>>>>>   PERIODIC INTERRUPT COUNT AND UPDATE ROUTINE   <<<<<<
;       -------------------------------------------------------------
;
        LD      HL,SETPT
        LD      BC,(ZONE)
        LD      B,0
        ADD     HL,BC           ;POINT TO CURRENT SET POINT
;
        LD      A,(ICNT)
        CP      A,(HL)
        JR      NZ,BEG1         ;SET POINT = # OF INTERRUPTS ? : NZ=NO
;
;       >>>  ZONE UPDATE ROUTINE  <<<
;
        LD      A,(ZONE)
        CP      A,3             ;IS ZONE = 3 ?
        JR      Z,BEG1          ;Z=ZONE IS EQUAL TO 3 (then do not increment)
        INC     A
        LD      (ZONE),A        ;UPDATE ZONE COUNTER
;
;       >>> INTERRUPT COUNTER UPDATE ROUTINE <<<
;
BEG1:   LD      A,(ICNT)
        CP      A,76            ;IS ICNT=76?
        JR      Z,BEG3          ;Z=ICNT IS EQUAL TO 76 (then do not inc.)
        INC     A
        LD      (ICNT),A        ;UPDATE ICNT
;
;       >>> ANALOG SCAN CHECK ROUTINE <<<
;
BEG3:   LD      A,(DIVIDE)
        DEC     A
        JP      Z,SETUP1        ;Z=ANALOG DATA SCAN REQUIRED
        LD      (DIVIDE),A      ;UPDATE DIVIDE COUNT
        JP      EXIT3           ;EXIT AND ENABLE INTERRUPT
```

```
;
;****************************************************************
;        >>>>>>   DATA AQUISITION UTILITY PROGRAMS   <<<<<<
;****************************************************************
;
;        ------------------------------------------
;        >>>   SET INTERRUPT COUNTER ROUTINE   <<<
;        ------------------------------------------
;
SETDIV: LD      HL,DIVCNT
        LD      BC,(ZONE)
        LD      B,0
        ADD     HL,BC           ;POINT TO DIVIDE DOWN NUMBER
        LD      A,(HL)
        LD      (DIVIDE),A      ;UPDATE DIVIDE COUNTER
        RET
;
;        ------------------------------------------------------
;        >>>   CPU INTERRUPT ENABLE AND DISABLE ROUTINES   <<<
;        ------------------------------------------------------
;
DISABL: DI
        PUSH    AF
;
        XOR     A
        OUT     0CFH,A          ;DISABLE CLOCK INTERRUPTS
;
        POP     AF
        RET
;
ENABL:  PUSH    AF
        LD      A,14H
        OUT     0CFH,A          ;START CLOCK INTERRUPTS (14H=1sec int.)
        POP     AF
        EI
        RET
;
;        ------------------------------------------
;        >>>   TIME AND DATE INPUT ROUTINE   <<<
;        ------------------------------------------
;
TIME:   PUSH    AF
        PUSH    BC
        PUSH    HL
;
        IN      A,0C0H
        AND     A,0FH
        CP      A,9             ;IS C0 = 9 ?
        JR      NZ,CLK5         ;NZ=C0 IS NOT EQUAL TO 9,CLK5
        LD      B,0             ;SET B=0 SINCE C0=9
        JR      CLK1
;
CLK5:   INC     A
        LD      B,A
;
CLK1:   IN      A,0C0H
        AND     A,0FH
        CP      A,B
        JR      NZ,CLK1         ;WAIT FOR 100usec CLOCK TRANSITION
;
```

```
        LD      B,9
        LD      HL,TDATA        ;POINT TO DATA AREA
        LD      C,0C4H          ;POINT TO SEC COUNTER
;
CLK2:   INI
        JR      Z,CLK3
;
        INC     C               ;POINT TO NEXT CLOCK PORT
        JR      CLK2
;
;       >>   MASK OFF HIGH NIBBLE OF CLOCK DATA  <<
;
CLK3:   LD      B,9
        LD      HL,TDATA
CLK4:   LD      A,0FH
        AND     A,(HL)
        LD      (HL),A
        INC     HL              ;POINT TO NEXT DATA ELEMENT
        DJNZ    CLK4            ;JUMP IF B<>0
;
        POP     HL
        POP     BC
        POP     AF
        RET
;
;       -----------------------------------------------------------
;       >>>   COMPUTER SYSTEM HARDWARE INITIALIZATION ROUTINE   <<<
;       -----------------------------------------------------------
;
SYSTEM: PUSH    AF
        PUSH    HL
;
        LD      HL,SERV         ;POINT TO INTERRUPTS SERVICE ROUTINE
        LD      (00F7H),HL      ;INIT INTERRUPT VECTOR LOCATION
;
        XOR     A               ;ZERO ACCUMULATOR
        LD      I,A
        IM2                     ;INT. MODE 2 (VECTOR ADDRESS=00F7H)
;
        OUT     0CFH,A          ;DISABLE ALL CLOCK INTERRUPTS
;
        OUT     03H,A           ;MASK CONSOLE PORT INTERRUPTS
        OUT     23H,A           ;MASK TUART PORT A INTERRUPTS
        OUT     83H,A           ;MASK TUART PORT B INTERRUPTS
;
        OUT     02H,A           ;DISABLE CONSOLE INT. ACK.
        OUT     22H,A           ;DISABLE TUART PORT A INT. ACK.
        OUT     82H,A           ;DISABLE TUART PORT B INT. ACK.
;
        POP     HL
        POP     AF
        RET
;
;       ---------------------------------------------------
;       >>>>>>   ANALOG  SCAN  SETUP  ROUTINE   <<<<<<
;       ---------------------------------------------------
;
SETUP:  DI
        PUSH    AF
        PUSH    BC
```

```
        PUSH    DE
        PUSH    HL
;
SETUP1: LD      HL,ADATA
        LD      (DTAB),HL       ;INITIALIZE DATA TABLE POINTER
;
        LD      BC,63
        LD      HL,AD1
        LD      DE,AD1+1
        LD      (HL),0
        LDIR                    ;INIT.  ANALOG DATA ARRAY AD TO ZERO
;
        CALL    SETDIV          ;INIT DIVIDE DOWN COUNTER
;
        XOR     A               ;ZERO ACCUMULATOR
        LD      (COUNT),A       ;INITIALIZE A/D  COUNT TO ZERO
;
        LD      (STAT),A        ;RESET STAT TO INDICATE SCANNING (STAT=0)
;
        OUT     0CFH,A          ;DISABLE ALL CLOCK INTERRUPTS
;
        LD      A,11H
        OUT     0CFH,A          ;ENABLE CLOCK INTERRUPT (11H=1 ms)
;
        JP      EXIT3           ;EXIT AND ENABLE INTERRUPT
;
;       ----------------------------------------------
;       >>>   CYCLE AND DAILY DATA INITIALIZATION   <<<
;       ----------------------------------------------
;
INIT:   PUSH    AF
        PUSH    BC
        PUSH    DE
        PUSH    HL
;
        LD      A,(SET)
        CP      A,2
        JR      Z,D1            ;Z=INIT DAILY DATA ONLY
;
        LD      HL,CYCLE
        LD      DE,CYCLE+1
        LD      BC,51
        LD      (HL),00H
        LDIR                    ;INITIALIZE DAILY DATA
;
        XOR     A
        OUT     24H,A           ;RESET DIGITAL COUNTERS
        LD      A,0FFH
        OUT     24H,A           ;ENABLE DIGITAL COUNTERS
;
        LD      A,(SET)
        CP      A,1
        JP      Z,EXIT4         ;Z=INIT CYCLE DATA ONLY  (1 = cycle data)
;
D1:     LD      HL,DAILY
        LD      DE,DAILY+1
        LD      BC,43
        LD      (HL),00H
        LDIR                    ;INITIALIZE DAILY DATA TO ZERO
;
```

```
        JP      EXIT4
;
;       ---------------------------------------------
;       >>>>>>   FLOPPY MOTOR SHUT-OFF ROUTINE   <<<<<<
;       ---------------------------------------------
;
MOTOR:  PUSH    BC
        LD      C,150
        CALL    0005H           ;CDOS SYSTEM CALL
        POP     BC
        RET
;
;       ----------------------------------
;       >>>>>>   CDOS RESET ROUTINE   <<<<<<
;       ----------------------------------
;
RESET:  PUSH    BC
        LD      C,13
        CALL    0005H           ;CDOS SYSTEM CALL
        POP     BC
        RET
        END
```

B.

```
C===========================================================
      PROGRAM DCEP
C===========================================================

      INTEGER*1 NS, NA, NB, TYPE, UNIT, UNIT1
      INTEGER POWER, CYDIV, TOTREC
      DIMENSION RA(13), RB(13), NA(13), NB(13)
      COMMON /A/ NS(5, 13), RS(5, 13), INUM, NDAYP, ISTART
      COMMON /C/ NN, UNIT, VER, CYDIV, NNREC, TYPE, NDISK, POWER
      COMMON /D/ NTITLE, III, IDS, SEC1, SECC1
      COMMON /E/ ISESON

      NREC=0
      NDISK=1
      WRITE(5, 1)
1     FORMAT(1X, ///, 2X, 'INPUT DATA TYPE', //, 15X, '1=CYCLE DATA', /, 15X,
     & '2=DAILY DATA', /)
      READ(5, 2)TYPE
2     FORMAT(1I1)

      WRITE(5, 11)
11    FORMAT(//, 2X, 'INPUT SEASON INFORMATION : ',
     &          /, 15X, '2 = HEATING SEASON',
     &          /, 15X, '5 = COOLING SEASON'/)
      READ(5, 12) ISESON
12    FORMAT(I1)

      NTITLE=1
      IF(TYPE. EQ. 2) NTITLE=2

      CALL OPEN(8, 'PUMPDATADTA', 1)
      CALL OPEN(9, 'PUMPDNEWDTA', 2)

      INUM=0
      DO 100 I=1, 13
      DO 100 J=1, 5
      NS(J, I)=0
      RS(J, I)=0. 0
100   CONTINUE

110   CONTINUE
      SECC1=-100. 0
      READ(8, END=220, REC=1) UNIT, TOTREC, VER, CYDIV, POWER
      READ(9, END=150, REC=1) UNIT1, NNREC, VER, CYDIV, POWER, NNN

      IF(UNIT. NE. UNIT1) WRITE(5, 3)
3     FORMAT(///, 9X, 'WRONG UNIT ---- CHECK DISK', ///)
      IF(UNIT. NE. UNIT1) GO TO 240
      IF(NNN. EQ. 0) GOTO 130

      READ(9, END=220, REC=NNREC) (NS(1, I), I=1, 12), (RS(1, I), I=1, 13)
      NNREC=NNREC-1
      INUM=1

      GO TO 140
130   CONTINUE
      READ(9, END=220, REC=NNREC) (NS(1, I), I=1, 12), (RS(1, I), I=1, 13)
     & , (NS(2, I), I=1, 12), (RS(2, I), I=1, 13)
```

```
      NNREC=NNREC-1
      INUM=2

140   CONTINUE
      IF(TYPE.EQ.1) GOTO 160
      TN=RS(INUM,10)-RS(INUM,11)
      RS(INUM,7)=TN*RS(INUM,7)
      RS(INUM,9)=TN*RS(INUM,9)
      ISTART=1
      NDAYP=100*NS(INUM,10)+10*NS(INUM,9)+NS(INUM,8)
      SEC1=3600.*(10.*NS(INUM,7)+NS(INUM,6))+60.*(10.*
     & NS(INUM,5)+NS(INUM,4))+10.*NS(INUM,3)+NS(INUM,2)
      IF(RS(INUM,13).EQ.0.) GOTO 160
      IF(RS(INUM,13).LT.1.) RS(INUM,13)=RS(INUM,10)/RS(INUM,13)
      GO TO 160

150   NNREC=1
      ISTART=0
      III=0

160   CONTINUE
      NREC=1
170   CONTINUE
      NREC=NREC+1

      DO 180 I=1,13
      NA(I)=0
      NB(I)=0
      RA(I)=0.0
      RB(I)=0.0
180   CONTINUE

      READ(8,END=230,REC=NREC) (NA(I),I=1,12),(RA(I),I=1,13),
     & (NB(I),I=1,12),(RB(I),I=1,13)

      IF(TYPE.NE.1) GOTO 185
      RA(ISESON)=RA(13)
      RB(ISESON)=RB(13)
185   IF(TYPE.EQ.1) GOTO 200
      IF(NA(1).NE.1) GOTO 190
      NA(1)=2
      CALL DDA(NA,RA,IEXIT)
      IF(IEXIT.EQ.1) GOTO 240

190   CONTINUE
      IF(NB(1).NE.1) GOTO 170
      NB(1)=2
      CALL DDA(NB,RB,IEXIT)
      IF(IEXIT.EQ.1) GOTO 240
      GOTO 170
200   CONTINUE

      IF(NA(1).NE.TYPE) GO TO 210
      CALL CTPT(NA,RA)

210   CONTINUE
      IF(NB(1).NE.TYPE) GO TO 170
      CALL CTPT(NB,RB)
      GO TO 170
```

```
220    CONTINUE
       WRITE(5,5)
5      FORMAT(//,30X,'ERROR !!',//,10X,
      & 'FILE 8 CANNOT BE FOUND IN DRIVE A OR IS EMPTY',//)
       GO TO 240

230    CONTINUE
       IF(INUM.EQ.0) GOTO 235
       IF(TYPE.EQ.1) GOTO 234
       IF(RS(INUM,13).EQ.0.) GOTO 233
       RS(INUM,13)=RS(INUM,10)/RS(INUM,13)
233    CALL CORECT(INUM,1)
234    CALL CHECK

235    ENDFILE 8
       CALL RESET
       WRITE(5,6)
6      FORMAT(//,5X,'INSERT NEW DISK :',//,10X,'INPUT  :'
      & ,'  1 -- NEW DATA DISK INSERTED ',/,18X,
      & ,'  0 -- END OF DATA INPUT',//)
       READ(5,7)NDISK
7      FORMAT(1I1)

       CALL OPEN(8,'PUMPDATADTA',1)
       IF(NN.NE.1) GO TO 237
       JJ=10
       CALL SAVE(JJ)

237    IF(NDISK.EQ.1) GOTO 110

240    CONTINUE
       STOP
       END


C-------------------------------------------------------------------
       SUBROUTINE DDA(NA,RA,IEXIT)
C-------------------------------------------------------------------

       INTEGER*1 NA,NS,UNIT,TYPE
       INTEGER POWER,CYDIV
       DIMENSION NA(13),RA(13)
       COMMON /A/ NS(5,13),RS(5,13),INUM,NDAYP,ISTART
       COMMON /C/ UNIT,VER,CYDIV,NNREC,TYPE,NDISK,POWER
       COMMON /D/ NTITLE,III,IDS,SEC1

       IEXIT=0
       NDAY=100*NA(10)+10*NA(9)+NA(8)
       IF(ISTART.EQ.1) GOTO 100
       INUM=1
       ISTART=1
       NDAYP=NDAY

100    CONTINUE
       IF(NDAY.GT.NDAYP) GOTO 140
       IF(NDAY.LT.NDAYP) GOTO 220

110    CONTINUE
       DO 120 I=1,13
```

```
      NS(INUM, I)=NA(I)
120   CONTINUE

      DO 130 I=1,5
      RS(INUM, I)=RS(INUM, I)+RA(I)
130   CONTINUE
      TN=RA(10)-RA(11)
      RS(INUM, 7)=RA(7)*TN+RS(INUM, 7)
      RS(INUM, 9)=RA(9)*TN+RS(INUM, 9)
      RS(INUM, 10)=RS(INUM, 10)+RA(10)
      RS(INUM, 11)=RS(INUM, 11)+RA(11)
      IF(III.EQ.0) SEC1=3600.*(NA(7)*10.+NA(6))+
     & 60.*(NA(5)*10.+NA(4))+NA(3)*10.+NA(2)
      IF(IDS.EQ.0.AND.SEC1.NE.0.) SEC1=SEC1-86400.
      SEC2=3600.*(10.*NA(7)+NA(6))+60.*(NA(5)*10.
     & +NA(4))+10.*NA(3)+NA(2)
      RS(INUM, 13)=RS(INUM, 13)+SEC2-SEC1
      IDS=1
      SEC1=SEC2
      III=1
      GOTO 230

140   CONTINUE
      IF(RS(INUM, 13).GT.1.) RS(INUM, 13)=
     & RS(INUM, 10)/RS(INUM, 13)
      NNDAY=NDAY-1
150   CONTINUE
      IF(NDAYP.EQ.NNDAY) GOTO 190
      CALL CORECT(INUM, 1)
      IF(INUM.EQ.5) CALL CHECK
      INUM=INUM+1
160   CONTINUE
      II=NNDAY-NDAYP

      IF(II.LE.0) GOTO 180
      NDAYP=NDAYP+1
      DO 170 I=1,13
      NS(INUM, I)=0
      RS(INUM, I)=0.0
170   CONTINUE
      NS(INUM, 1)=2

      NS(INUM, 10)=NDAYP/100
      NS(INUM, 9)=NDAYP/10-10*NS(INUM, 10)
      NS(INUM, 8)=NDAYP-10*NS(INUM, 9)-100*NS(INUM, 10)
      IF(INUM.EQ.5) CALL CHECK
      INUM=INUM+1
      GOTO 160
180   CONTINUE
      NDAYP=NDAYP+1
      GOTO 200
190   CONTINUE

      IDS=0
      CALL CORECT(INUM, 1)
      IF(INUM.EQ.5) CALL CHECK
      INUM=INUM+1
      NDAYP=NDAYP+1
200   CONTINUE
      DO 210 I=1,13
```

```
      RS(INUM,I)=0.0
210   CONTINUE
      GOTO 110

220   CONTINUE
      IEXIT=1
      WRITE(5,1)
1     FORMAT(////,10X,'THE CYCLE DATA IS NOT IN CORRECT ORDER',
     & ' FOR SUMMATION',//,15X,'CHECK THE DATA DISKS',///)
230   CONTINUE

      RETURN
      END


C-------------------------------------------------------------
      SUBROUTINE CTPT(N,R)
C-------------------------------------------------------------

      INTEGER*1 N,NS,TYPE,UNIT
      INTEGER POWER,CYDIV
      DIMENSION N(13),R(13)
      COMMON /A/ NS(5,13),RS(5,13),INUM,NDAYP,ISTART
      COMMON /C/ NN,UNIT,VER,CYDIV,NNREC,TYPE,NDISK,POWER
      COMMON /D/ NTITLE,III,IDS,SEC1,SECC1

      INUM=INUM+1
      DO 10 K=1,13
      NS(INUM,K)=N(K)
      RS(INUM,K)=R(K)
10    CONTINUE

      AD=100.*N(10)+10.*N(9)+N(8)
      SECC2=3600.*(10.*N(7)+N(6))+60.*(10.*N(5)+N(4))+10.*N(3)+N(2)
      IF(SECC1.LT.0.) GOTO 20
      SECC=SECC2-SECC1+86400.*(AD-ADP)
      RS(INUM,13)=RS(INUM,10)/SECC
20    CONTINUE

      ADP=AD
      SECC1=SECC2
      IF(INUM.EQ.5) CALL CHECK

      RETURN
      END



C-------------------------------------------------------------
      SUBROUTINE CHECK
C-------------------------------------------------------------

      INTEGER*1 N,TYPE,UNIT
      INTEGER CYDIV,POWER
      COMMON /A/ N(5,13),R(5,13),INUM,NDAYP,ISTART
      COMMON /C/ NN,UNIT,VER,CYDIV,NNREC,TYPE,NDISK,POWER
      COMMON /D/ NTITLE,III,IDS,SEC1
```

```
      CALL PRINT

100   WRITE(5,1)
1     FORMAT(5X,'WHICH DATA SET NEED CORRECTION? ',
     & 'INPUT ISET -- I5    &',/,5X,'NO. OF CORRECTIONS'
     & ,'   INPUT ICOR -- I5 (INPUT 0 FOR NO CORRECTION)'
     & ,/,5X,'FOR ERASING ALL DATA -- ICOR=100'/)

      READ(5,2) ISET,ICOR
2     FORMAT(2I5)
      IF(ISET.EQ.0.OR.ICOR.EQ.0) GO TO 140
      IF(ICOR.NE.100) GOTO 120
      DO 110 I=1,13
      R(ISET,I)=0.0
110   CONTINUE
      CALL PRINT
      GOTO 100
120   CONTINUE

      DO 130 I=1,ICOR
      WRITE(5,3)
3     FORMAT(//,5X,'INPUT COLUMN NO.  -- I3',/,
     &           5X,' &  NEW VALUE     -- F12.3 ',/)
      READ(5,4) IC,AA
4     FORMAT(I3,F12.3)
      R(ISET,IC)=AA
130   CONTINUE
      CALL CORECT(ISET)

      CALL PRINT
      GO TO 100


140   CONTINUE
      DO 150 II=1,INUM
      CALL SAVE(II)
150   CONTINUE
      INUM=0
      RETURN
      END


C-----------------------------------------------------------------
      SUBROUTINE SAVE(II)
C-----------------------------------------------------------------

      INTEGER*1 N,NA,UNIT,TYPE
      INTEGER CYDIV,POWER
      DIMENSION N(26),R(26)
      COMMON /A/ NA(5,13),RA(5,13),INUM,NDAYP,ISTART
      COMMON /C/ NN,UNIT,VER,CYDIV,NNREC,TYPE,NDISK,POWER
      COMMON /D/ NTITLE
      DATA NN/0/

      IF(II.EQ.10) GO TO 20
      NN=NN+1
      K=13*(NN-1)
      DO 10 I=1,13
      J=I+K
```

```
      N(J)=NA(II,I)
      R(J)=RA(II,I)
10    CONTINUE

      IF(NN.EQ.1) RETURN

20    CONTINUE
      NNREC=NNREC+1
      IF(NN.EQ.2) GO TO 30
      WRITE(9,REC=NNREC) (N(I),I=1,12),(R(I),I=1,13)
      GO TO 40
30    CONTINUE
      WRITE(9,REC=NNREC) (N(I),I=1,12),(R(I),I=1,13),
     & (N(I),I=14,25),(R(I),I=14,26)

      NN=0
40    CONTINUE
      WRITE(9,REC=1) UNIT,NNREC,VER,CYDIV,POWER,NN
      NN=0
      RETURN
      END


C-----------------------------------------------------------------
      SUBROUTINE CORECT(ISET,ID)
C-----------------------------------------------------------------

      INTEGER*1 N
      COMMON /A/N(5,13),R(5,13),INUM,NDAYP,ISTART
      COMMON /D/ NTITLE

      IF(NTITLE.NE.2.OR.ID.NE.1) GOTO 5
      T=R(ISET,10)-R(ISET,11)
      IF(T.LE.0.0) GOTO 5
      R(ISET,7)=R(ISET,7)/T
      R(ISET,9)=R(ISET,9)/T
5     CONTINUE
      E=(R(ISET,3)+R(ISET,4)+R(ISET,5))*3.413
      G=R(ISET,1)+R(ISET,2)
      IF(E.NE.0.0) GOTO 10
      R(ISET,12)=0.0
      GOTO 20
10    CONTINUE
      R(ISET,12)=G/E
20    CONTINUE

      RETURN
      END


C-----------------------------------------------------------------
      SUBROUTINE PRINT
C-----------------------------------------------------------------

      INTEGER*1 N,UNIT,TYPE
      INTEGER POWER,CYDIV
      DIMENSION TITLE(13,2)
      COMMON /A/ N(5,13),R(5,13),INUM,NDAYP,ISTART
```

```
      COMMON /C/ UNIT, VER, CYDIV, NNREC, TYPE, NDISK, POWER
      COMMON /D/ NTITLE
      COMMON /E/ ISESON
      DATA TITLE/'QS', 'QL', 'ECMP', 'EFAN', 'EHET', 'DIQ2', 'TRET',
     & 'DIQ3', 'TDPR', 'CTIM', 'DTIM', 'COP', '%CON', 'QS', 'QL',
     & 'ECMP', 'EFAN', 'EHET', 'TOUT', 'TRET', 'TDPO', 'TDPR', 'CTIM'
     & , 'DTIM', 'COP', '%CON'/
      DATA START/0.0/

      IF(NTITLE.EQ.2.OR.START.NE.0.0) GOTO 100
      TITLE(ISESON,1)=TITLE(6,2)
      START=1.0

100   WRITE(5,1) (N(I,10),N(I,9),N(I,8),I=1,INUM)
1     FORMAT(12X,5(1X,'DAY = ',3I1,3X))
      WRITE(5,2) (N(I,7),N(I,6),N(I,5),N(I,4),N(I,3),
     & N(I,2),I=1,INUM)
2     FORMAT(12X,5(1X,2I1,':',2I1,':',2I1,4X))
      WRITE(5,3)
3     FORMAT(/)

      DO 10 J=1,13
      WRITE(5,4) J,TITLE(J,NTITLE),(R(I,J),I=1,INUM)
4     FORMAT(2X,I2,2X,A4,2X,5(G10.4,3X))
10    CONTINUE

      RETURN
      END
```

B.

```
C ==============================================================
      PROGRAM DDC
C ==============================================================

      INTEGER*1 UNIT, UNIT1, N, NN
      DIMENSION N(12), NN(2, 12), RR(2, 13)
      COMMON /A/ IBDAY, NDAY, INUM, IEND

      WRITE(5, 1)
1     FORMAT(/, 5X, 'CHECK THE DISKS IN SLOTS', /, 5X,
     & 'INPUT 0 TO CONTINUE THE PROGRAM', //)
      READ(5, 2) IBEG
2     FORMAT(I1)

      CALL OPEN(9, 'PUMPDNEWDTA', 2)
      READ(9, END=850, REC=2) (N(I), I=1, 12)
      IBDAY=100*N(10)+10*N(9)+N(8)

100   ISTART=0
      NDC=0
      NREC=1
      INUM=0
      TOUT=0.0
      TDPO=0.0
      IEND=0

      CALL OPEN(8, 'PUMPDATADTA', 1)

      READ(8, END=700, REC=1) UNIT
      READ(9, END=700, REC=1) UNIT1
      IF(UNIT.NE.UNIT1) GOTO 800

200   CONTINUE

      NREC=NREC+1
      READ(8, END=600, REC=NREC) (NN(1, I), I=1, 12), (RR(1, I), I=1, 13)
     & , (NN(2, I), I=1, 12), (RR(2, I), I=1, 13)

      DO 500 I=1, 2
      IF(NN(I, 1).NE.3) GOTO 500
      NDC=100*NN(I, 10)+10*NN(I, 9)+NN(I, 8)
      IF(ISTART.EQ.0) NDAY=NDC
      MN=NDC-NDAY
      IF(MN) 900, 400, 300
300   CALL CHECK(TOUT, TDPO)
      TOUT=0.0
      TDPO=0.0
      INUM=0
      NDAY=NDC
400   TOUT=RR(I, 1)+TOUT
      TDPO=RR(I, 2)+TDPO
      INUM=INUM+1
      ISTART=1
500   CONTINUE

      GOTO 200
600   CONTINUE
      IEND=1
      CALL CHECK(TOUT, TDPO)
```

```
      ENDFILE 8
      CALL RESET
      WRITE(5,4)
4     FORMAT(//,5X,'INSERT NEW DISK :',//,10X,'INPUT  :'
     & ,'  1 -- NEW DATA DISK INSERTED ',/,18X,
     &  '  0 -- END OF DATA INPUT',//)
      READ(5,5) NDISK
5     FORMAT(I1)
      IF(NDISK.EQ.1) GOTO 100

      GOTO 900

700   CONTINUE
      WRITE(5,6)
6     FORMAT(//,30X,'ERROR!!',/,10X,'FILE 8 OR FILE 9 ',/,
     & 5X,'CANNOT BE FOUND OR FILE IS EMPTY',//)
      GOTO 900

800   CONTINUE
      WRITE(5,7)
7     FORMAT(//,30X,'ERROR!!',//,10X,'THE UNITS IN DRIVE',
     & ' A AND DRIVE B ',/,15X,'IS NOT CONSISTENT'//)
      GOTO 900

850   WRITE(5,8)
8     FORMAT(//,30X,'ERROR !!',//,10X,'PUMPDNEW.DTA CANNOT BE',
     & ' FOUND IN DRIVE B OR IS EMPTY',//)
900   STOP
      END


C -----------------------------------------------------------
      SUBROUTINE CHECK(TOUT,TDPO)
C -----------------------------------------------------------

      INTEGER*1 N
      DIMENSION N(2,12),R(2,13)
      COMMON /A/ IBDAY,NDAY,INUM,IEND

      II=NDAY-IBDAY+2
      J=1+II/2
      K=II-2*J+3

      READ(9,END=200,REC=J) (N(1,I),I=1,12),(R(1,I),I=1,13),
     & (N(2,I),I=1,12),(R(2,I),I=1,13)

      NO=N(K,12)*10+N(K,11)
      FNO=FLOAT(NO)
      R(K,6)=FNO*R(K,6)+TOUT
      R(K,8)=FNO*R(K,8)+TDPO

      INUM=INUM+NO
      FNUM=FLOAT(INUM)
      R(K,6)=R(K,6)/FNUM
      R(K,8)=R(K,8)/FNUM
      IF(IEND.NE.1) GOTO 100
      N(K,12)=INUM/10
      N(K,11)=INUM-10*N(K,12)
```

```
100   CONTINUE

      WRITE(9,REC=J) (N(1,I),I=1,12),(R(1,I),I=1,13),
     &  (N(2,I),I=1,12),(R(2,I),I=1,13)
      WRITE(5,1) NDAY,R(K,6),R(K,8),FNUM
1     FORMAT(/,5X,'DAY= ',I3,10X,'TOUT= ',F5.1,8X,'TDPO= '
     &  ,F5.1,/,5X,'NO. OF RECORDS (1/2 HR DATA) IN THIS',
     &  ,' DAY =',F5.0,//)

      GOTO 300
200   WRITE(5,2) J,K
2     FORMAT(/,5X,'CANNOT FIND THE FOLLOWING RECORD:',/,
     & 10X,'NREC= ',I4,5X,'SET= ',I2,//)
300   CONTINUE

      RETURN
      END
```

B.

Appendix D

SI Conversion

In view of the presently accepted practice of the building industry in the United States and the structure of the computer software used in this project, common U.S. units of measurement have been used throughout this report. In recognition of the United States as a signatory to the General Conference of Weights and Measures, which gave official status to the metric SI system of units in 1960, appropriate conversion factors have been provided in the table below. The reader interested in making further use of the coherent system of SI units is referred to: NBS SP330, 1972 Edition, "The International System of Units," or E380-72, ASTM Metric Practice Guide (American National Standard 2210.1).

Metric Conversion Factors

Length	1 inch (in) = 25.4 millimeters (mm) 1 foot (ft) = 0.3048 meter (m)
Area	1 ft^2 = 0.092903 m^2
Volume	1 ft^3 = 0.028317 m^3
Temperature	F = 9/5 C + 32
Temperature Interval	1 F = 5/9 C or K
Mass	1 pound (lb) = 0.453592 kilogram (kg)
Mass Per Unit Volume	1 lb/ft^3 = 16.0185 kg/m^3
Energy	1 Btu = 1.05506 kilojoules (kJ)
Specific Heat	1 Btu/[(lb) (°F)] = 4.1868 kJ/[(kg) (K)]

NBS-114A (REV. 2-80)

U.S. DEPT. OF COMM. **BIBLIOGRAPHIC DATA SHEET** *(See instructions)*	1. PUBLICATION OR REPORT NO. NBSIR 81-2285	2. Performing Organ. Report No.	3. Publication Date June 1981

4\. TITLE AND SUBTITLE

Using Microcomputers to Monitor the Field Performance of Residential Heat Pumps

5\. AUTHOR(S)

C. W. Hurley, P. A. Kopetka, G. E. Kelly

6\. PERFORMING ORGANIZATION *(If joint or other than NBS, see instructions)*

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7\. Contract/Grant No.

8\. Type of Report & Period Covered

9\. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS *(Street, City, State, ZIP)*

U.S. Department of Energy
1000 Independence Avenue, S.W.
Washington, D.C. 20585

10\. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11\. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)*

Field data on the heating and cooling performance of residential heat pumps were gathered for the purpose of verifying and refining laboratory testing procedures. This report describes the procedures, instrumentation, and microprocessor-based data acquisition system (DAS) used for evaluating the field performance of three residential heat pumps located in the Washington, D.C. area. The instrumentation, signal conditioning unit and DAS are described in detail since the designs employed are applicable to future testing projects of this type in both small and large scale field studies.

To avoid the large capacities of the DAS and data reduction facility required for on-line monitoring, a strategy was developed which used the on-line microcomputer in the field to reduce and analyze the raw data and record the calculated results. This reduced the amount of recorded data to an acceptable level and thereby extended the time period between data collection.

This report discusses the selection of the heat pumps utilized in this field study and the design and selection of the instrumentation and DAS. The requirements for scanning data and recording the results are also discussed.

The basic equations and the software for processing the data at the field units and for reducing and editing the raw data disks at a central microcomputer are described. Examples of printouts taken directly at the field units and from the data disks are shown.

12\. KEY WORDS *(Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)*

Analog signal conditioning; data acquisition system; field data acquisition; field instrumentation; field performance of heat pumps; heat pumps; heat pump test methods; microcomputer.

13\. AVAILABILITY

☒ Unlimited
☐ For Official Distribution. Do Not Release to NTIS
☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.
☐ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14\. NO. OF PRINTED PAGES
115

15\. Price
$11.00