EPA/600/R-10/110
October 2010

AUTOMATED LONG-TERM REMOTE MONITORING OF SEDIMENT-WATER INTERFACIAL FLUX

AUTOMATED LONG-TERM REMOTE MONITORING

OF

SEDIMENT-WATER INTERFACIAL FLUX

By

Bob K. Lien[1]

Carl G. Enfield[2]

[1]U.S. EPA National Risk Management Research Laboratory

Office of Research and Development

U.S. Environmental Protection Agency

Cincinnati, OH 45268

[2]Pegasus Technical Services,

Cincinnati, OH 45219

Notice

The U. S. Environmental Protection Agency through its Office of Research and Development funded the research described here. This project was conducted under an approved Quality Assurance Project Plan (743-Q2-1). This report has been subjected to the Agency's peer review and has been approved for publication as an EPA document. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

The authors wish to acknowledge George Enfield (Pegasus Technical Service Inc.), Patrick Clark and Robert Ford (USEPA/ORD) for their assistance in the field activities. Kim McClellan (USEPA/ORD) is acknowledged for providing editorial review of this report. Steve Acree (USEPA/ORD-Ada) and Robert Ford (USEPA/ORD) are acknowledged for providing Red Cove staff gauge data and technical review of this report.

Foreword

The U.S. Environmental Protection Agency (EPA) is charged by Congress with protecting the Nation's land, air, and water resources. Under a mandate of national environmental laws, the Agency strives to formulate and implement actions leading to a compatible balance between human activities and the ability of natural systems to support and nurture life. To meet this mandate, EPA's research program is providing data and technical support for solving environmental problems today and building a science knowledge base necessary to manage our ecological resources wisely, understand how pollutants affect our health, and prevent or reduce environmental risks in the future.

The National Risk Management Research Laboratory (NRMRL) is the Agency's center for investigation of technological and management approaches for preventing and reducing risks from pollution that threaten human health and the environment. The focus of the Laboratory's research program is on methods and their cost-effectiveness for prevention and control of pollution to air, land, water, and subsurface resources; protection of water quality in public water systems; remediation of contaminated sites, sediments and groundwater; prevention and control of indoor air pollution; and restoration of ecosystems. NRMRL collaborates with both public and private sector partners to foster technologies that reduce the cost of compliance and to anticipate emerging problems. NRMRL's research provides solutions to environmental problems by: developing and promoting technologies that protect and improve the environment; advancing scientific and engineering information to support regulatory and policy decisions; and providing the technical support and information transfer to ensure implementation of environmental regulations and strategies at the national, state, and community levels.

This publication has been produced as part of the Laboratory's strategic long-term research plan. It is published and made available by EPA's Office of Research and Development to assist the user community and to link researchers with their clients.

Sally Gutierrez, Director
National Risk Management Research Laboratory

Abstract

Advective flux across the sediment-water interface is temporally and spatially heterogeneous in nature. For contaminated sediment sites, monitoring spatial as well as temporal variation of advective flux is of importance to proper risk management. This project was conducted to develop a ruggedized advective flux meter capable of unattended long-term remote monitoring of the sediment-water interfacial flux. The flux meter, based on the heat-pulse technique, is capable of quantitatively measuring bi-directional seepage flux from 0 to 8 cm/day with the current configuration. The system has automatic data acquisition, real time data display, and signal display/analysis capability. A remotely controlled monitoring module was included to permit system monitoring and modification over the internet. The instrument has undergone several calibrations to establish relations between the flow rate and the heat-pulse travel time. For very low flow rates, an alternate approach was adopted based on the ratio of temperature rises. Using this approach, it is possible to accurately interpret flow rates down to zero flow conditions. Depending on the magnitude, flow rate can be derived as a function of peak temperature arrival time or as a ratio of rise in temperature on both sides of the heater. Using these two methods, reliable measurements can be made over a wide bi-directional range of flows. In the field operation, the flow across the sediment-water interface is isolated by a dome chamber and funneled through a flow sensor where the flow rate is measured. The advective flux through the sediment-water interface, in terms of the vertical Darcy velocity, is calculated by dividing the flow rate through the flow sensor by the cross-section area of the dome.

The flux meter underwent a continuous long-term field test from June 2008 to April 2009 in the Red Cove area of Plow Shop Pond, Fort Devens, Massachusetts. Two flux meters were placed side by side; the unattended operation ran automatically with a self-sufficient power supply system. The measurement frequency was changeable through the wireless network. Seepage flux data was available for download after each measurement cycle. All of the data suggest very low flow and the majority of the data suggest an upward flow direction from the groundwater to the surface water. The average advective flux, in terms of Darcy velocity, for flux meter unit SN300 and SN303 was 0.44 ± 0.34 mm/day and 0.76 ± 0.33 mm/day, respectively. Calculated seepage flux based on hydraulic conductivity and gradient data suggests the average flux was 1.12 ± 0.71 mm/day. During the winter months when the surface water was frozen, the data superficially suggests there was significant rapid oscillatory pumping of the ice-covered water during this period of time.

Table of Contents

List of Figures

1 Introduction

Advective flux (also referred to as seepage flux) at the groundwater-surface water interaction zone is the magnitude and direction of water movement across a sediment-water interface. The convention is that positive advective flux values indicate flow of groundwater to the surface water and negative flux values indicate flow from the surface water to the groundwater. The seepage flux is measured in terms of volume per unit area per unit time ($L^3/L^2/T$), or simply translates to unit length per unit time (L/T).

There are many indirect and direct methods of estimating advective flux at the groundwater-surface water interaction zone (Kalbus, Reinstorf et al. 2006). The indirect methods include hydraulic conductivity and hydraulic gradient, heat tracer, hydrographic analysis, artificial and/or environmental tracers, or hydrogeological mapping. The most common direct method is the use of seepage flux meters. The basic concept of the flux meter is to cover and isolate an area of the sediment-water interface with a submerged chamber opened at the bottom to measure the volume of water exchange between the groundwater and the surface water over an interval of time. The classic flux meter design of Lee (1977) consists of a cutoff 55 gallon (200L) drum inserted into the sediment. A stopper with a tube is inserted into a port on the top of the drum and a plastic bag is attached to the tube with rubber bands. The time interval when the bag is connected and subsequently disconnected is recorded, as well as the change in the volume of water in the bag. The basic Lee-type flux meter has its shortcomings (Schincariol and McNeil 2002), and many factors could affect the accuracy of the flux measurements (Murdoch and Kelly 2003). One of the major disadvangates of the Lee-type flux meter is that it only measures the accumulated seepage over the deployed time period, which does not provide any data on temporal changes of seepage during that time period.

Over the years, various modifications have been made to the basic design of Lee's flux meter. The same principle of an inverted chamber to isolate and direct seepage flux remains. However, instead of using a plastic bag or similar collection device, different instruments are used to continually measure the time-series change in flow automatically. From the literature, examples of these modified devices include the ultrasonic meter (Paulsen, Smith et al. 2001), the heat-pulse meter (Taniguchi and Fukuo 1993; Lien 2006; Smith, Herne et al. 2009), the continuous heat-type meter (Taniguchi and Iwakawa 2001), the dye-dilution meter (Sholkovitz, Herbold et al. 2003) and the electromagnetic meter (Rosenberry and Morin 2004). The range of measurable flow rates varies with each device. Most of the devices were deployed for periods of days or weeks in various subsequent field studies (Taniguchi, Burnett et al. 2003; Taniguchi, Burnett et al. 2006; Taniguchi, Ishitobi et al. 2007; Swarzenski and Izbicki 2009; Mwashote, Burnett et al. 2010), with the exception of (Fritz, Mendoza et al. 2009) who deployed an ultrasonic meter in a river for a period of two months. None of these devices were specifically developed to include remote data transmittal capability. Although the above mentioned heat-pulse meters are similar in that they all employ heat-pulse technique, Lien's and Smith's heat-pulse meters are also capable of measuring bi-directional water flow at the groundwater-surface water interaction zone.

Temporal hydrological variation exists in nature. For reliable environmental risk management, it is essential to collect and analyze repeated measurements to evaluate changes in conditions. Knowledge of the variation in seepage flux is invaluable to proper remedial design at a contaminated sediment site. The need to develop a flux meter that is sustainable to extensive environmental exposure is therefore warranted. This document presents the development and field testing of a ruggedized advective flux meter for measurement of advective water flux across the sediment-water interface. Evolved from a previously developed prototype by Lien (2006), the flux meter was enhanced and configured for long-term deployment and remote data transmittal capabilities.

1.1 Background

Previously, a prototype bi-directional advective flux meter for measuring water flux across the sediment-water interface was developed by U.S. EPA and field tested at multiple sites (Lien 2006). The prototype consists of a dome chamber opened at the base to isolate and direct flow through a flow sensor for measurement. The flow sensor employs a principle of heat-pulse technique (Campbell, Calissendorff et al. 1991; Choi, Lee et al. 1991; Taniguchi and Fukuo 1993; Paillet, Crowder et al. 1996; Ren, Kluitenberg et al. 2000; Mortensen, Hopmans et al. 2006) derived from the effect of flowing water velocity on heat flow propagation along a flow tube. When a temperature gradient exists along the flow path, the heat will be transported in the flowing water by advection and conduction. The advective component of the heat flow is related to the flow velocity of the fluid. Empirical relations between flow rate and heat-pulse peak arrival time were established; flow rate could be derived as a function of peak temperature arrival time, or as a function of first temporal moment of the heat-pulse. The complete prototype system was also equipped with automatic data acquisition and signal display/analysis capability. The prototype flux meter has undergone field tests at three different settings. The first test site was a shallow turbulent stream at Santo Domingo, Nicaragua. The second test site was at Grand Calumet River, Hammond, Indiana, which is a slow moving river with fine texture organic rich sediment. And the third test site was at Lake Hartwell, South Carolina which is a deeper water reservoir with sandy sediment. The field deployments were successful in that the flux meter was operational and bi-directional flow measurement capability was demonstrated. Nevertheless, ideas for further improvement emerged from lessons learned from earlier work.

The early prototype flux meter had notable limitations:

1. The prototype required constant physical presence of an operator and is not suitable for unattended long-term monitoring from a remote location;
2. In fast moving water bodies, differential pressures were observed between inside and outside the dome chamber, but could not be recorded in real-time to correct for the Bernoulli Effect;
3. It is necessary to separate gas flow from liquid flow due to differences in thermal conductivity. The prototype had potential problems with gas venting configuration. The prototype used a vent tube to the surface, which was successful when the surface water was calm and with little change in the surface water elevation. However, when the surface water is dynamic unintended flow could be induced;
4. The prototype flux meter was not built to sustain lengthy field deployment.

1.2 Scope and Objectives

The scope of this project is to aim at improving the limitations and shortcomings of the prototype flux meter and further enhance its capability for long-term operation. The objectives of this project include:

- Redesign and ruggedize the flux meter mechanicals for long-term deployment;
- Redesign an imbedded low power data collection system for unattended operation, and capable of being queried and modified through the internet;
- Redesign circuitry to incorporate differential pressure measurements for use in determining differences in pressure between the inside and the outside of the dome to correct for the Bernoulli Effect;
- Redesign signal conditioning at the data collection system for optional pressure transducers (up to four with 0-10 VDC output and 15 VDC excitation) and temperature transducers (10kΩ thermistors up to two);
- Redesign gas venting mechanism to minimize the interference of gas ebullition on flow measurements;
- Demonstrate the automated long-term seepage flux measurement at a field location.

2 Flux Meter Design

The design of the meter incorporates the same principle of the previous prototype with improvements to ensure good operational integrity over an extended period of time. Significant modifications were made to address power consumption, remote data transmittal, and long-term environmental exposure issues. The meter was designed to withstand up to 30 feet (9.1 m) of hydraulic pressure. The *in situ* components of the meter consist of a cylinder that when pushed into the sediment isolated 2,922.5 cm^2 of interfacial sediment area, a dome attached to the cylinder with a water-tight seal, and a flow tube that allows water to freely flow between inside the dome and the surface water at the rate equal to the specific discharge rate across the sediment-water interface.

2.1 Basic Concept

The principles of operation of a groundwater/surface water flow meter were originally presented by (Zuber 1970). If a known area of the groundwater/surface water interface is covered by a chamber which is open at the bottom and has a pipe outlet near the top (Figure 1), the following expression can be written to express the flux (q) of water normal to a plane passing through the open bottom of the container:

$$q = , \frac{a}{A} \qquad [1]$$

where

- v is the velocity of the water in the measuring pipe
- A is the area covered by the chamber
- a is the cross sectional area of the measuring pipe.

Two assumptions must be satisfied for the validity of Equation [1]. The first assumption requires that area A be tightly covered by the instrument, so the flow through area A is equal to the flow through the measuring pipe. The second assumption requires a negligible loss of potential in the pipe compared with the loss along the edge of the cylinder which is tightly pressed into the bed material. In other words, the presence of the measuring instrument should not disturb natural flow lines. Taniguchi and Fukuo (1993) concluded that inaccuracy of the flow measurement caused by pressure drop in the pipe could be minimized by using a large diameter pipe (i.e. ≥ 0.9 cm ID).

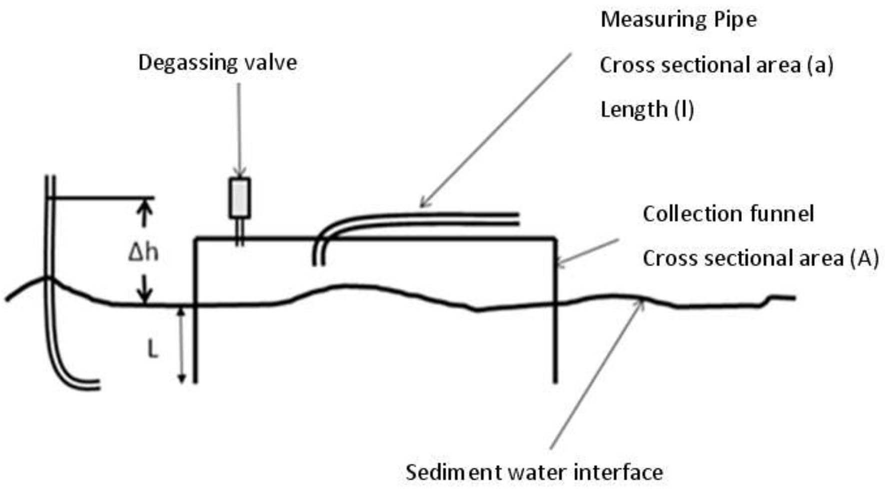


Figure 1 Schematic drawing of the measurement system.

A heat-pulse flow sensor is incorporated along the flow tube to measure advective flow rate. Flow through an isolated section of the sediment-water interface is captured and directed through the flow sensor. As shown in a conceptual drawing in Figure 2, a typical field installation would include a data collection/controller system, internet interface, underwater instrumentation and power source. The device is capable of minimizing interference of gas ebullition on flow measurements by installing an automatic degassing valve (Plast-O-matic Automatic Vent Valve DGV050V-PV[1]) on the dome. The top of the dome is slightly convex allowing any gas generated from the sediments to escape through the degassing valve rather than into the flow tube.

A differential pressure transducer (Setra Systems 23010R5PB2FEB ±0.5PSID[1]) is incorporated to measure the difference in pressure between the inside and the outside of the dome. In high velocity streams, the orientation of the discharge can create pressure gradients (the Bernoulli Effect) which would potentially influence flow through the device (Shinn, Reich et al. 2002; Rosenberry 2008). During installation, when significant stream flows are present, the dome should be rotated to an optimal orientation where this pressure difference is minimal. Alternatively, the differential pressure transducer can also be used to determine the difference in hydraulic head from two piezometers that are installed at different elevations in the sediment formation.

[1] Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

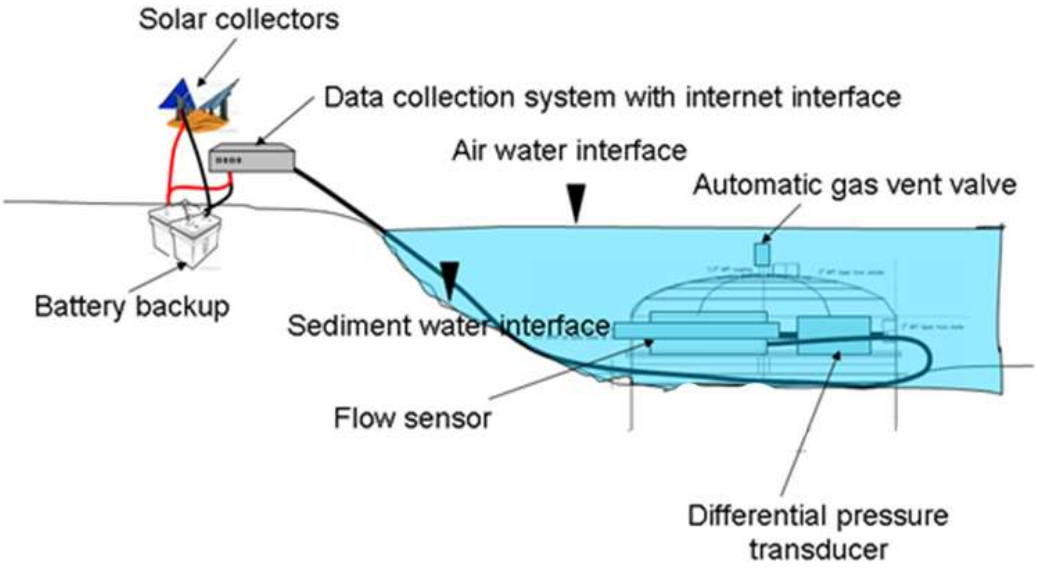


Figure 2 Field implementation of the advective flux meter.

2.2 Design of Measurement System

Power for the system operation was provided by a 300 W solar panel with regulator and battery backup provided by Sun Wise Systems. A pulse of heat is injected into in the center of the flow sensor using a resistive heater (Figure 3). The heater is a section of 3/8" stainless steel pipe with a 5 mm stainless steel tubing helix fused to the pipe and filled with the resistive heating element. The migration of this heat-pulse is monitored using four thermistors (General Electric BR14KA1003K 10K ohm ± 10% @ 25°C housed in 20 GA 304 Stainless Steel thin wall hypo tubing, closed end). Two of the thermistors are placed on each side of the heater to measure the bi-directional movement of the heat-pulse. A fifth thermistor is attached to the surface of the heater to monitor the temperature rise and fall of a heating cycle. A reference thermistor measures the temperature of the flow case. The five thermistors shown in Figure 3 are identified as T2, T3, T4, T5 and T6, with T2 being closest to the cable end of the device. This nomenclature is used throughout this report to identify the individual thermistors. Not shown in Figure 3 are T1 (a reference thermistor imbedded in the case of the flux meter) and T7 (a thermistor outside the dome used to measure the ambient water temperature). For the three remaining channels on the 10-channel counter, Channel 8 is the differential pressure transducer, Channel 9 and 10 can be connected to either pressure transducers or thermistors that are connected above the water surface.

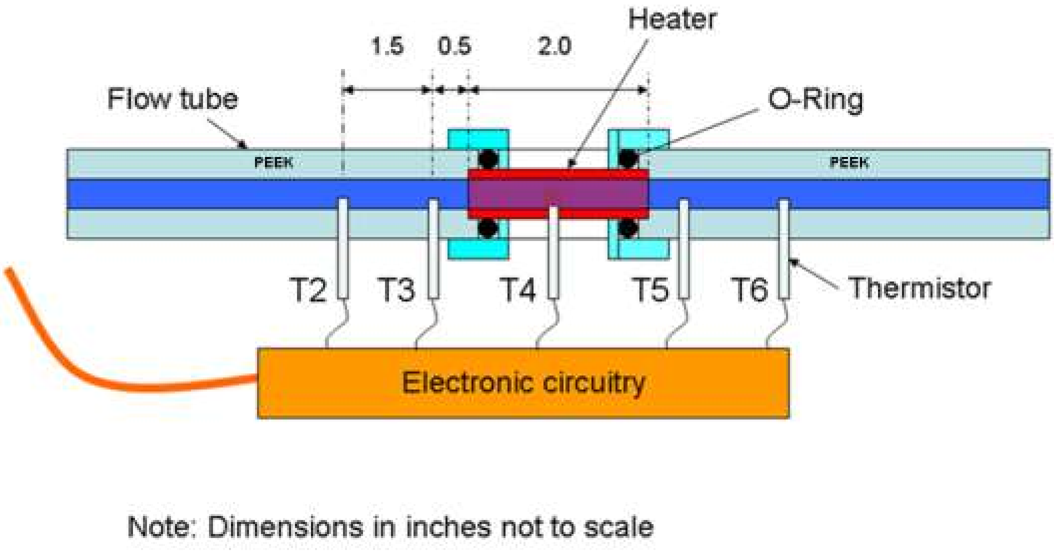


Figure 3 Conceptual drawing of the flow sensor. The flow tube is constructed with Polyetheretherketone (PEEK) material. Note that T4 is on the outside surface of the heater not in contact with water.

The system is controlled by an embedded single board PC with Windows XP as the operating system (WinSystems PPM-GX500-G). Software installed on the single board computer included Symantec PC Anywhere to provide a user interface between the remote and local computers, and Tethys Automation Anywhere to automate the remote boot and program initialization sequence, as well as a Visual Basic code used to operate the device. The application that operated the device was written in Visual Basic. The code is exhibited in Appendix A. A cellular modem (Janus Terminus GSM/GPRS Terminal) was used to provide the field modem for the T-Mobile virtual private network (VPN). This allowed remote communication with the field instrument from an office environment. The single board computer was interfaced with the sensors using a Measurement Computing USB 4303 10 channel counter. A counter system was used to overcome difficulties in transmitting low voltage analog signals over the long cables.

All of the analog signals needed to be converted to variable frequency digital pulse strings for recording. A simplified block diagram of the temperature measurement and signal conversion system is shown in Figure 4. There are five different functional blocks in the analog to the digital circuitry. Regulated constant sources (Burr Brown REF200) feed a fixed 100 µA current into each of two thermistors used to measure temperature. The voltage generated is proportional to the temperature at the thermistors reactive element. The reference thermistor (Figure 4) is imbedded into the case of the flow sensor. The two voltages are subtracted with the difference amplifier and the voltage difference is converted to a frequency for transmission to the frequency counter. A circuit diagram is shown in Appendix B and the printed circuit board layout is shown in Appendix C. To minimize errors caused by temperature induced buoyancy gradients, the flow sensor is installed level and the amount of energy injected into the water is small. The above described circuit board provides eight channels of data and requires one input power voltage (24 VDC) and two 5 VDC control signals. The USB 4303 used as the interface between the signal data and the embedded computer has ten channels available. An additional circuit board was constructed to support the USB 4303 and provide up to three additional signals. Two of the additional signals are designed for the pressure transducers, which are excited by a 15 VDC power supply, and one thermistor supplied a 100 µA current source. The same voltage to frequency circuitry was used as discussed above. The board layout is shown in Appendix D. Both circuit boards are shown in Figure 5 as they were configured for the work at Fort Devens. Some of the components on the circuit boards are labeled. In Figure 5, one of the circuit boards containing the USB 4303 is not fully populated with components. These components were not installed, because they were not needed at Fort Devens, and the power consumption needed to be reduced.

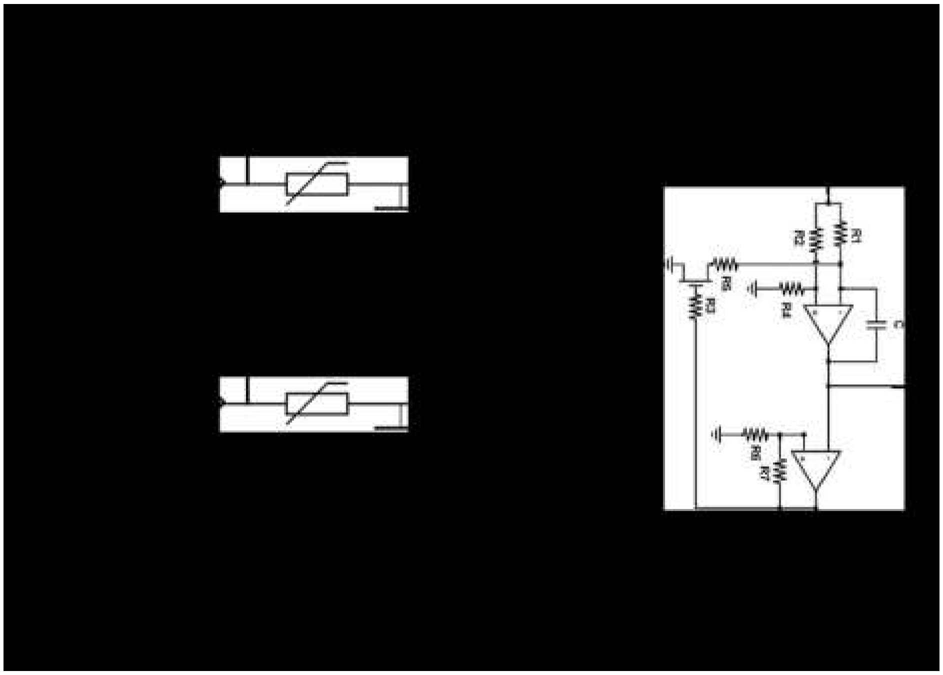


Figure 4 Block diagram of the measurement system.

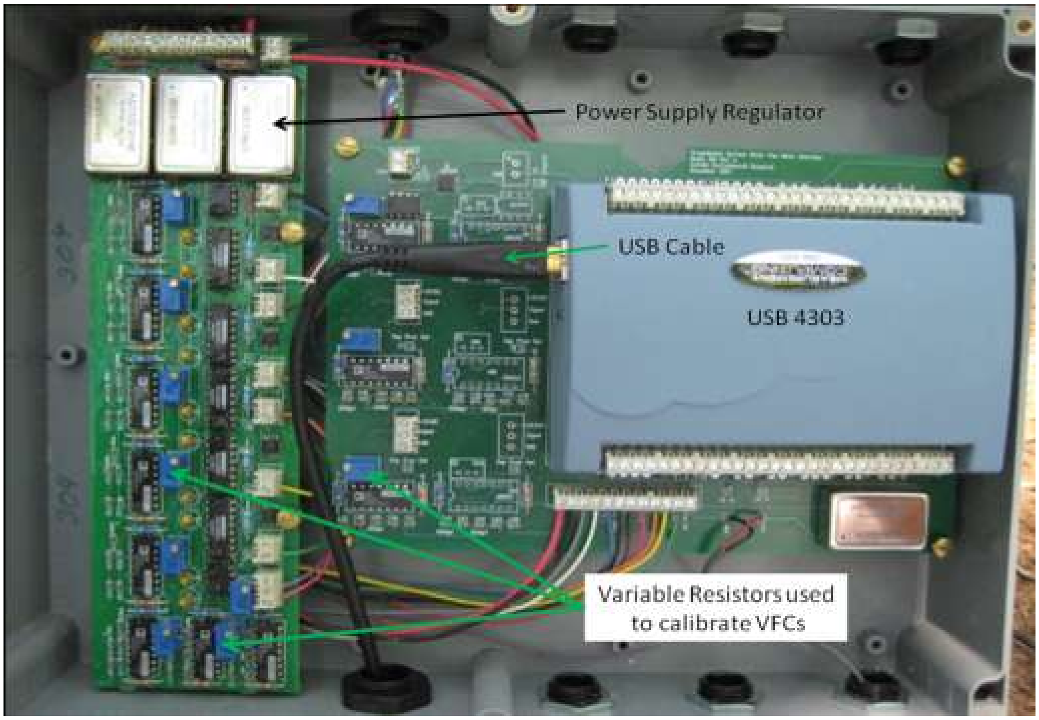


Figure 5 Signal conditioning and data interface boards as used in the field.

2.3 Flux Meter Mechanicals

The flow sensor is shown in Figure 6. The flow tube is constructed with Polyetheretherketone (PEEK) material. The flow sensor is encased in a 304 Stainless Steel cylindrical flow tube case shown in Figure 6A. In addition to containing the flow tube, the flow case provides a water tight environment for the thermistor that measures the ambient water temperature and the connections for the differential pressure transducer. These items are shown in the photo in Figure 6A. The end plates drawings of the flow tube case are shown in Figure 6B and C. The flow tube was held in the case using 1" 316 Stainless Steel compression fittings. The end plates of the flow tube case were sealed with O-rings. The differential pressure transducer was encased in PVC. The two halves of the differential pressure transducer case were sealed with a neoprene gasket. In the photo of Figure 6A, each pressure test port of the flow tube case and the differential pressure transducer case is pointed out by an arrow. Both cases were pressure-tested by immersing in water and pressurizing the case to 20 psig. The seals were considered adequate if no air bubbles were observed.

The dome chamber consists of two separable pieces constructed out of 304 Stainless Steel. The bottom cylinder is pressed into the sediment formation during installation in the field and an upper dome houses the flow tube case and other components. The bottom cylinder isolates 2,922.5 cm^2 of interfacial sediment area when pushed into the sediments. The dome chamber also serves as an amplifier to the generally low seepage flow rates. A photo of the two parts separated is shown in Figure 7A. Between the two halves there is a closed cell PVC gasket providing a water tight seal. The two halves are held together by lever clamps. A flange is welded around the lower part that serves as a gauge for installing the unit to a consistent depth of 15 cm. The radius of the bottom cylinder is 30.5 cm. There are cross supports and a drive index pin in the center for use in stiff sediments. The upper part of the dome has several ports to connect the required fittings (Figure 7D). The 1/4" NPT fitting in the top is for the vent degassing valve. The 2" NPT port in the top serves as an air relief port used during installation to allow flow through the dome or a location where a standard addition can be performed. On the left side of the figure there are three ports. The 1" NPT fitting is for the flow tube. The 1/4" NPSF fitting is for the thermistor that measures the ambient temperature. The 1/4" NPT fitting is for a pressure port for the differential pressure transducer. On the right side of the figure there are two more ports. The 2" NPT fitting is for the cable going to the surface and the 1/4" NPT fitting not labeled is an additional pressure port for the differential pressure transducer. The unit is shown assembled in Figure 7B and C. Note that the flow tube is positioned inside the dome for maximum protection from any potential outside destruction.

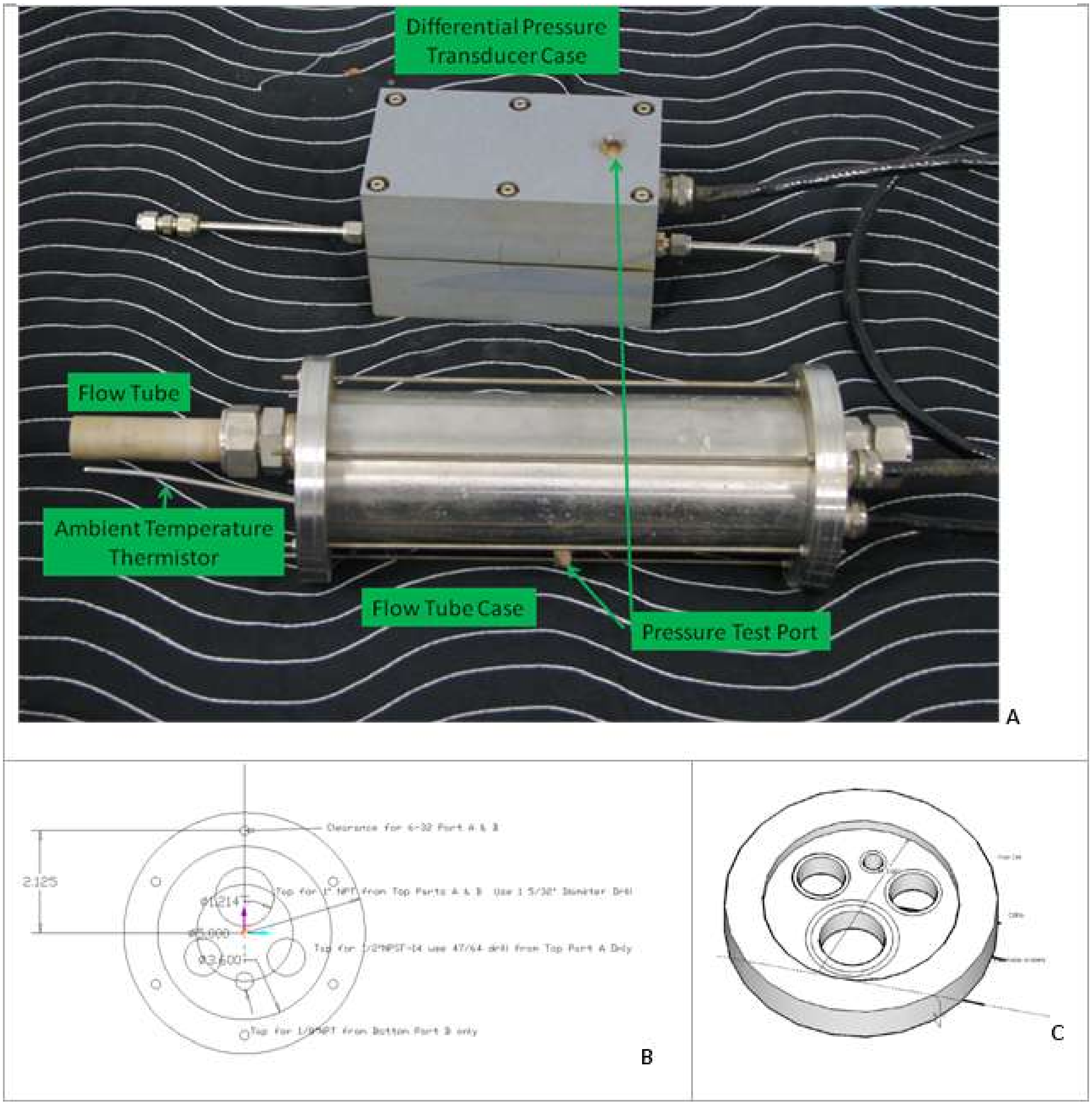

Differential Pressure Transducer Case
Flow Tube
Ambient Temperature Thermistor
Flow Tube Case
Pressure Test Port
A
Clearance for 6-32 Port A & B
2.125
Ø1.214
Top for 1" NPT from Top Ports A & B Use 1 5/32" Diameter Drill
Ø5.000
Ø3.600
Top for 1/2"NPST-14 use 47/64 drill from Top Port A Only
Top for 1/8"NPT from Bottom Port B only
B
C

Figure 6 Flow tube and differential pressure transducer with drawings of flow tube case end plates.

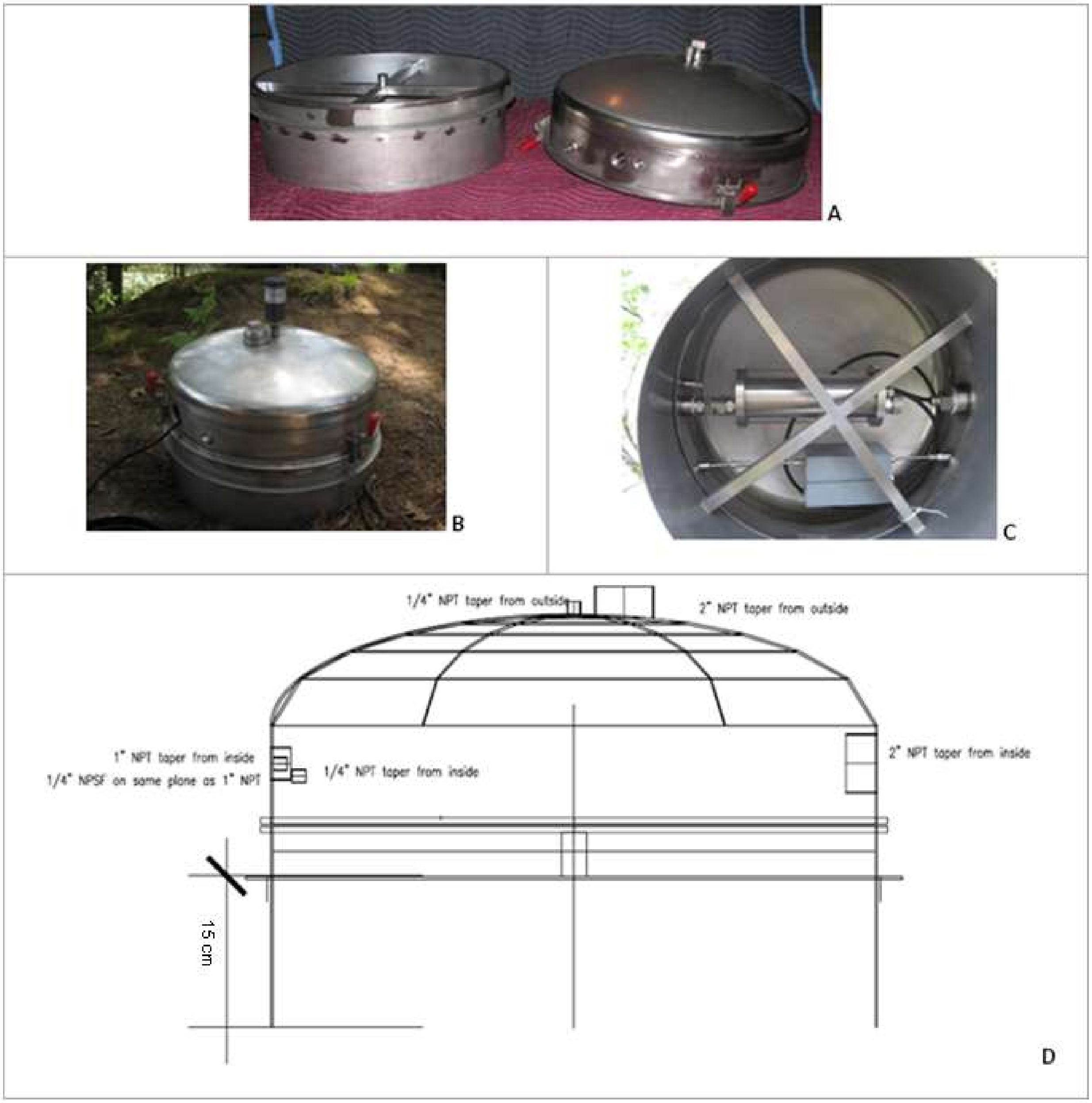

A
B
C
1/4" NPT taper from outside
2" NPT taper from outside
1" NPT taper from inside
1/4" NPSF on same plane as 1" NPT
1/4" NPT taper from inside
2" NPT taper from inside
15 cm
D

Figure 7 Flow meter Dome Chamber. (A) Photo of dome halves separated; (B) Photo of dome as it would be installed in the field; (C) Assembled unit with the flow tube and the differential pressure transducer installed inside the chamber (shown from the bottom); (D) Drawing of the dome chamber. A flange is welded around the bottom cylinder that serves as a gauge for installing the unit to a consistent depth of 15 cm.

3 Theory of Operation

The theory of heat flow has been a subject of investigation for centuries. Numerous papers and books have been written on the subject. Probably the most comprehensive book on the subject within the last century was Conduction of Heat in Solids written by H.S. Carslaw and J.C. Jaeger originally published in 1946. When the developed theories are applied to physical problems it is often necessary to approximate either the initial or boundary conditions. For a simple solution that can describe the thermograph, one would either want an instantaneous pulse of heat, a step of heat, or a square wave pulse. None of the normally assumed initial conditions is a reasonable description of the heat input produced by our heater. As a result, the shape of the thermograph observed in the physical system is inconsistent with the shape produced by the common the mathematical developments. To overcome this problem, Taniguchi and Fukuo, (1993) proposed using the time the peak temperature is detected at the thermistors to determine the velocity of the heat. They took the derivative of an analytical solution with ideal boundary conditions. The peak temperature occurs when the derivative equals zero. Their development yields the equation:

$$U^2 = \frac{x^2 - 2\alpha t_{\max}}{t_{\max}^{\ 2}} \qquad [2]$$

where U is the velocity, t_{max} is the peak temperature arrival time measured by the thermistor at distance x from the heat source, and α is the thermal conductivity of water divided by the specific heat and density. The water velocity is proportional but not equal to the thermal velocity. If one can assume there is no thermal gradient in the radial direction (instantaneous heat transfer in the radial direction) and there is no heat loss from the pipe, then the thermal velocity will be slower and directionally proportional to the water velocity. The (Taniguchi and Fukuo 1993) technique is suitable for measuring higher flow rates when the advective process dominates. It uses the difference in peak arrival times between thermistors to estimate flow rates. The precise distance between the thermistors must be known.

Lien (2006) proposed using moment analysis to determine the travel time of the heat-pulse. If there are no other external inputs, the first moment of the thermograph is the heat-pulse peak arrival time (t_P). The time of travel of the heat-pulse between the center of the generated heat-pulse and a measuring thermistor is equal to the distance between the points divided by the sum of the diffusive flux and the advective flux, plus the time required for the heat to travel from the heater to the water. Heat flow is controlled by two primary components. At low flow rates the predominant component is heat conduction through the media. At high flow rates the primary component is advective flow of the water. The empirical function which describes the flow of heat though the flow sensor has the form:

$$Q = \frac{C_1 - C_3 \, t_P}{t_P + C_2} \qquad [3]$$

where Q is the volumetric flow rate in the pipe and C_1, C_2, C_3 are calibration constants optimized by mathematical fitting of data. The calibration constants are presumed to be dependent on the physical geometry and material properties of each individual device.

4 Laboratory Calibrations

There are three aspects of the instrument calibration. In the flux meter, thermistors are used to measure temperature. Thermistors are non linear temperature dependent resistors and need to be calibrated when precise measurements are needed. Since the heat-pulse thermograph is defined by a series of data points with small temperature changes, all of the thermistors in the flux meter need to be calibrated so small temperature changes can be detected. Thermistors are analog devices and the signals must be "conditioned" before they can be converted to digital pulse streams. The signal conditioning circuitry needs to be calibrated because of the differences in components used in the construction of the circuitry. Finally, the entire flow sensor needs to be calibrated because of the variability in the machining and construction of the device.

4.1 Thermistor Calibration

Thermistors are temperature dependent resistors frequently used to measure temperature. All of the thermistors used are commercially manufactured in stainless steel jackets designed for continuous emersion in water. The nominal resistance, of all the thermistors, is 10,000 ohms at 25^{o}C. Calibration is performed by measuring the resistance of the thermistor as a function of temperature over the range of temperatures anticipated under field conditions (0 – 40^{o}C). This is accomplished by placing the thermistor along with a reference thermometer (Traceable Digital Thermometer Model 4000CC or equivalent) in an aluminum block (Figure 8), and placing the entire block in a temperature controlled water bath. The resistance of the thermistor is

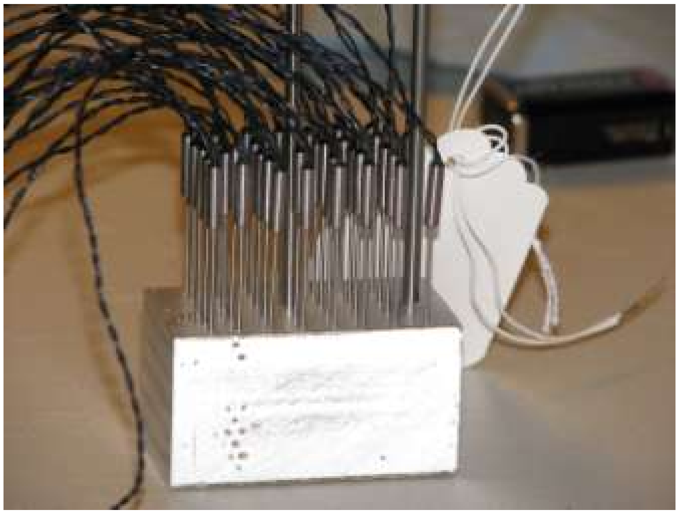


Figure 8 Thermistors in an aluminum block used for calibration.

measured with a four-digit digital multimeter (Fluke Model 85). A minimum of eight data points are collected over a temperature range of 0 – 40^{o}C. The least-squares method is then used to fit the data to the Steinhart-Hart equation (Steinhart and Hart 1968):

$$\frac{1}{T} = A + B\ln(\rho + C[\ln(\rho]^{2} \qquad [4]$$

where

- *T* is the temperature (in degree Kelvin)
- ρ is the resistance at T (in Ohms)
- *A*, *B*, and *C* are the Steinhart-Hart coefficients which vary depending on the type and model of thermistor and the temperature range of interest.

The Steinhart-Hart coefficients for each thermistor are stored along with the serial number of the thermistor. The calibration is considered successful when the mean squared error of the measurements is less than or equal to 1.0E-11, and the difference between measured and calculated temperature does not exceed 0.5^{o}C. When the mean squared error of the thermistor is less than 1.0E-11, the overall reproducibility is better than ± 0.2^{o}C. Calibration is performed in batches. The thermistors are then grouped or matched based on the resistance value at 25^{o}C. All thermistors used in each flux meter are from a same group.

All of the thermistors were calibrated as described above. Most of the thermistors followed the expected response curve within the accuracy of the reference thermometer. Typical calibration results are shown in Figure 9. The blue-diamond marks the measured temperature using the reference thermometer versus estimated temperature using the thermistor. The red-cross marks the error (secondary Y-axis) associated with each estimated temperature in degrees C. The error appears to be small but random. In this particular example, the root mean squared error (RMSE) of the estimator was 4.5E-3^{o}C. There were no problems with the Steinhart-Hart function. Calibration curves were, however, significantly different. The thermistors were grouped such that the calculated resistances at 25^{o}C of each calibration curves were as close to the same value as possible. Thermistors within the same group were used in the same flux meter.

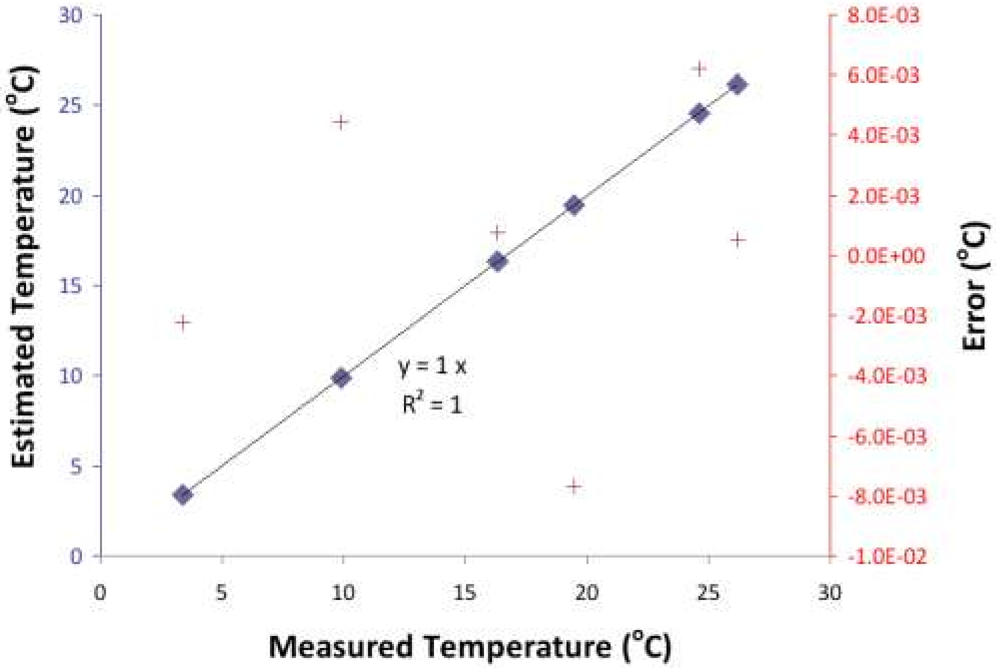


Figure 9 Example results of a typical thermistor calibration. The blue-diamond marks the measured versus the estimated temperature. The red-cross marks the error (on secondary Y-axis) associated with each estimated temperature. In this particular example, the root mean squared error of the estimator was 4.5E-3°C.

4.2 Circuitry Description and Calibration

The circuit diagram for the flux meter sensor, not including the data logger, is shown in Appendix B. To measure the reference temperature, a non-adjustable precision reference current source supplies a constant current (100 µA half of Burr Brown REF 200, U15) to the thermistor connected to J2. This thermistor is in series with the 1.82 kΩ resistor (R32) the produced voltage is buffered, gain of 1, by an operational amplifier (Burr Brown OPA134, U19) to reduce the impedance and minimize stray current sinks. The output voltage from the OPA134, pin 6, is proportional to a reference temperature (the temperature of the flow case) and used as a reference for all of the temperature measurements in the flow path. The reference temperature voltage is converted to a frequency by a voltage to frequency converter U2 (an Analog Devices AD650), the output of U2 is connected to connector J1 pin 5 for connection to the cable going to the interface board. Temperatures along the flow path are measurements of difference. A measured temperature is subtracted from the reference temperature and the difference is amplified by an instrumentation amplifier. For example the thermistor nearest the cable end of the flow sensor is connected to J3 and terminal 2 is connected to the negative input of the instrumentation amplifier U3 to make a temperature measurement in the flow tube. The reference voltage (which is proportional to the case temperature) is applied to the positive input of one of the instrumentation amplifiers (Burr Brown INA 103). The output of the instrumentation amplifier is connected to the input of a voltage to frequency converter (VFC) (Analog Devices AD650, U3). The voltage to frequency converter circuit is designed to produce 100,000 pulses per second with a 10 VDC input. The linearity of the circuit with the components used is specified by the manufacturer. The linearity was verified during the design phase but is not re-checked during calibration. The output of the VFC is transmitted by cable to the computer interface (J1).

The VFCs require calibration. The process used for VFC U2 is to substitute a decade resistance substitution box in place of the Thermistor at Jack J2 (see Appendix B for parts layout). The resistance, approximately 8180 ohms, is adjusted until 1 VDC is produced at the output (pin 6) of the OPA134 (U19) then R1 is adjusted until 10,000 Hz is produced at pin 8 of the AD650 (U2). Once calibrated the adjusting screw on R1 is secured using a small drop of fingernail polish.

A similar approach is used on the circuitry for the temperature outside the dome connected to J8 (see Appendix B). The decade resistance substitution box is connected to J8. The resistance should be approximately 10kΩ. The decade resistance substitution box is adjusted until 1VDC is obtained at pin 6 of the OPA 134, U20. Then adjust R22 until 10,000 Hz is obtained at pin 8 of the AD650, U11.

The thermistors connected to J3 through J7 are measured on a differential circuit (the voltage generated by the thermistors connected to J3 through J7 is sequentially subtracted from the reference voltage from OPA134, U19 and then multiplied by ten using an instrumentation amplifier). Thermistors decrease in resistance with an increase in temperature. The thermistors connected to J3 through J7 are at or above the temperature of the reference thermistor connected to J2. Therefore, the output from the instrumentation amplifier will always be positive. To calibrate the voltage to frequency converters, connect a resistance substitution box to J2, as before, and adjust the output of the OPA 134, U19 to 1 VDC. Short the pins of J3 and adjust R7 for 100,000 Hz at pin 8 of the AD650, U6. Repeat the process for the remaining four temperature detectors.

The last input on the circuit board is the differential pressure transducer. Supply a known reference voltage between 1 and 10 VDC to pin 2, and adjust R25 until the output from the AD650, pin 8, of U12 is equal to 10,000 Hz per volt DC. Calibration is considered successful when all adjustable frequencies are within 0.5% of the target test voltage or frequency.

On the circuit board there is also a connection for the heater and two digital control lines. One digital control line operates the heater and the other control line operates the power supplies used for the remainder of the circuitry. The power supplies are turned off between sampling events to preserve power.

4.3 Flow Sensor Calibration

The flow sensors were calibrated by pumping water through each unit at 7 different flow rates bi-directionally from 0.27 to 16.7 ml/min (equivalent to Darcy velocities of 0.13 to 8.22 cm/day in our current flux meter configuration). Calibrations were performed at room temperature. The flow sensor was placed in a tank of water to minimize short term temperature fluctuation but no attempt was made to maintain a constant temperature during the calibration process. The measurements were repeated at least five times at each flow rate to develop the calibration curves. Data was then fitted to Equation 3 using the least-squares method. For the purpose of distinguishing flow direction in the flow tube, flow vector is assigned. Flow toward T5 and T6 direction is considered positive while flow toward T3 and T2 direction is considered negative flow. In the field the convention is to define positive flow as upward groundwater discharge to surface water and negative flow as downward flow from surface water to groundwater.

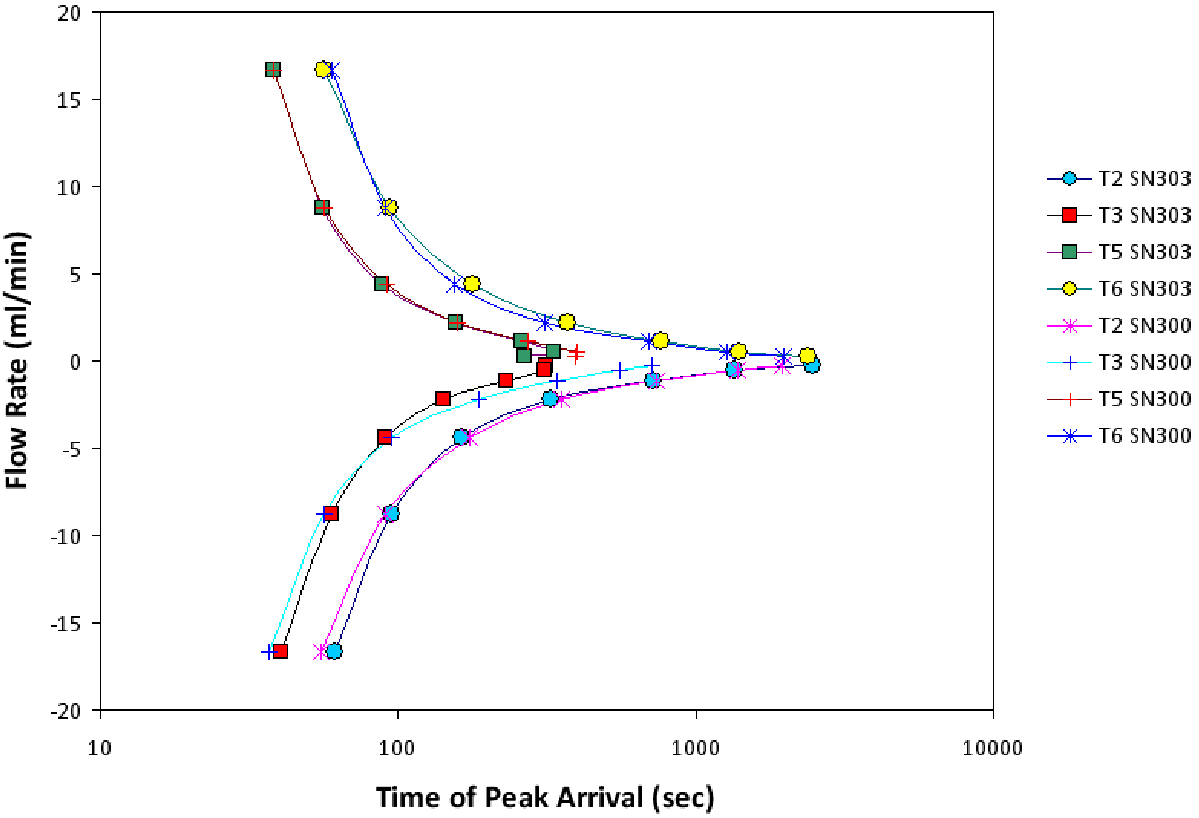


Figure 10 Comparative plot of calibration curves for flow sensor unit SN300 and SN303. For the purpose of distinguishing flow direction, flow toward T5 and T6 is considered positive flow while flow toward T3 and T2 is considered negative flow.

Figure 10 shows the calibration curves for flow sensor unit SN300 and SN303 (see Appendix E for the calibration constants). The calibration curves of the two flow sensors are inherently similar because of the similarity in their geometry and material used. To find out the difference of the two flow sensors, a series of peak arrival times were applied to each calibration to calculate flow rates. The results indicate the mean absolute error (MAE) between the two sensors was 0.29 ml/min. The data points in Figure 10 represent the average of n replicate measurements at the same flow rate, with n typically equal or greater than five. Taking all replicate measurements into account the average coefficient of variation was 0.038 for SN300 and 0.037 for SN303. Figure 11 compares the estimated flow rate versus the measured flow rate for T2 and T6 (the two outer thermistors) using Equation 3. It was perceptible that the estimated and measured flow rates correlated very well. The error associated with each estimated flow rate was shown on the secondary Y-axis. The root mean squared error (RMSE)

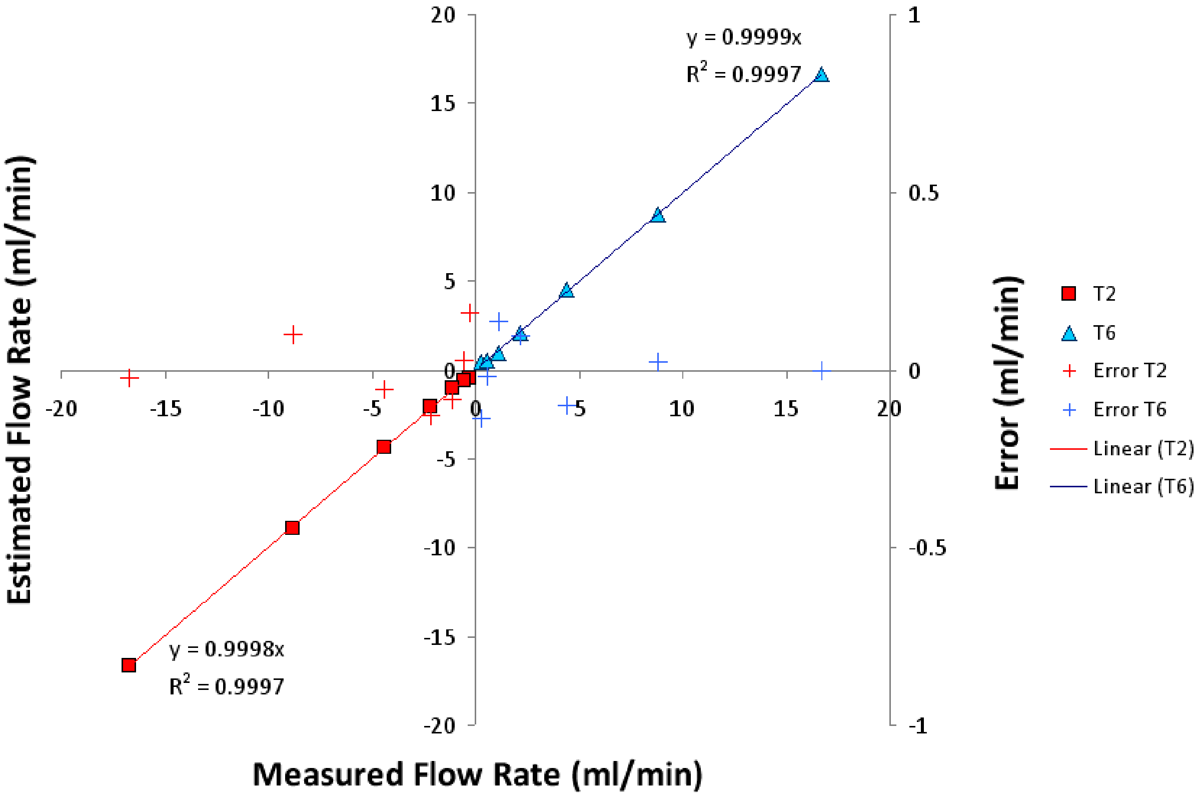


Figure 11 Comparison between measured and estimated flows for T2 and T6 using Equation 3. The error associated with each estimated flow rate is shown on the secondary Y-axis. The RMSE of T2 and T6 flow estimation was 0.095 and 0.090 ml/min, respectively.

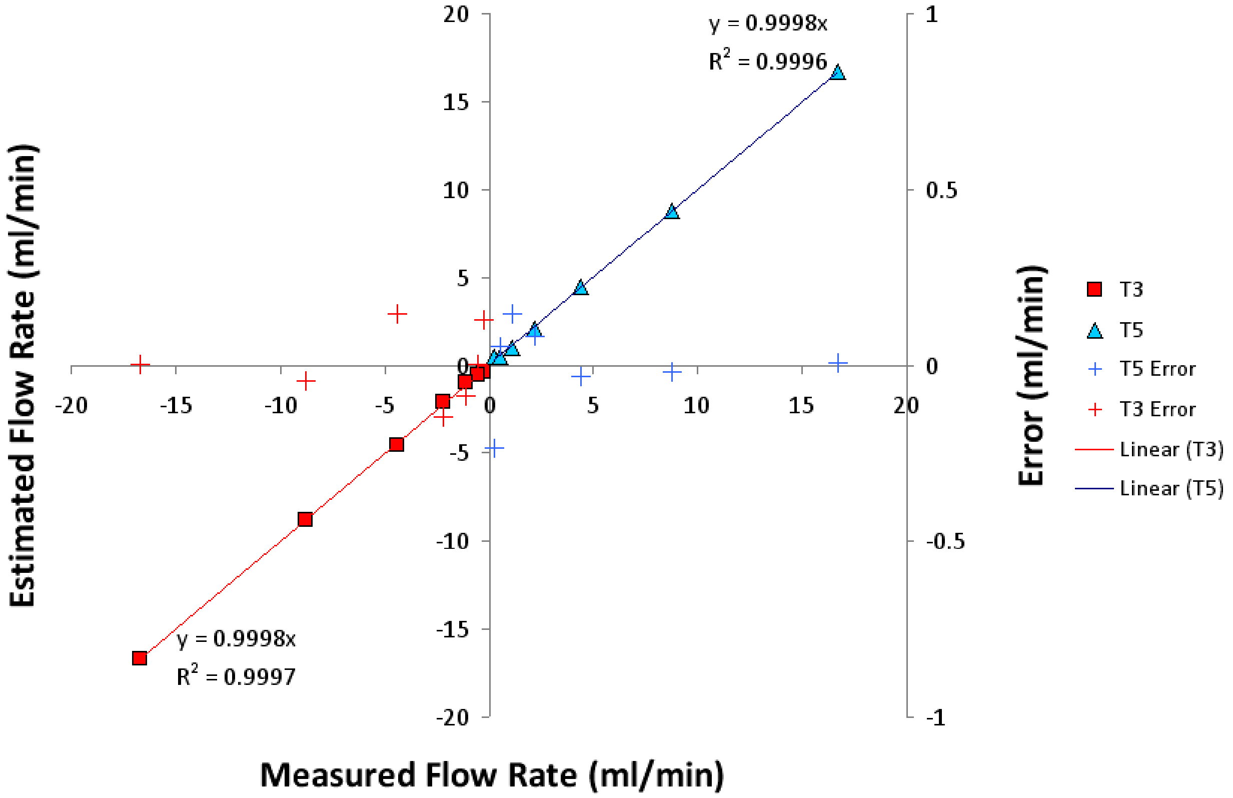


Figure 12 Comparison between measured and estimated flow for T3 and T5 using Equation 3. The error associated with each estimated flow rate is shown on the secondary Y-axis. The RMSE of T3 and T5 flow estimation was 0.099 and 0.112 ml/min, respectively.

of T2 was 0.095 ml/min and the RMSE of T6 was 0.090 ml/min. The magnitude of error appeared to be greater at lower flow rates. In a similar manner, Figure 12 compares the measured flow rate and the estimated flow rate for T3 and T5, which are the thermistors closest to the heater. Similarly, the magnitude of error appeared to be greater at lower flow rates. The RMSE of T3 was 0.099 ml/min and the RMSE for T5 was 0.112 ml/min.

Figure 13 shows how to look at the same error data by plotting the MAE as a function of the absolute value of the flow rate. The magnitude of error emerged to be greater at lower flow rates. One potential explanation is based on the principals of operation. There are two components to the heat flow. Advective flow dominates the time of travel of the heat-pulse at high flow rates and conductive or diffusive flow dominates the time of travel of the heat-pulse at low flows. One might conclude that the majority of the error associated with the calibration data presented here is in the estimation of the conductive flow. Another potential explanation of greater errors under low flow situations is the high uncertainty in determining when the peak temperature arrives. As the flow velocity slows down, the thermograph spreads and looses its sharp peak. Lack of a sharp peak increases the uncertainty in determining the peak arrival time. It appears that Figure 13 represents the behavior one normally encounters with analytical measurements that have a lower limit of detection. Typically for the analytical measurements, the error increases as the measured parameter approaches the limit of detection of the analytical device.

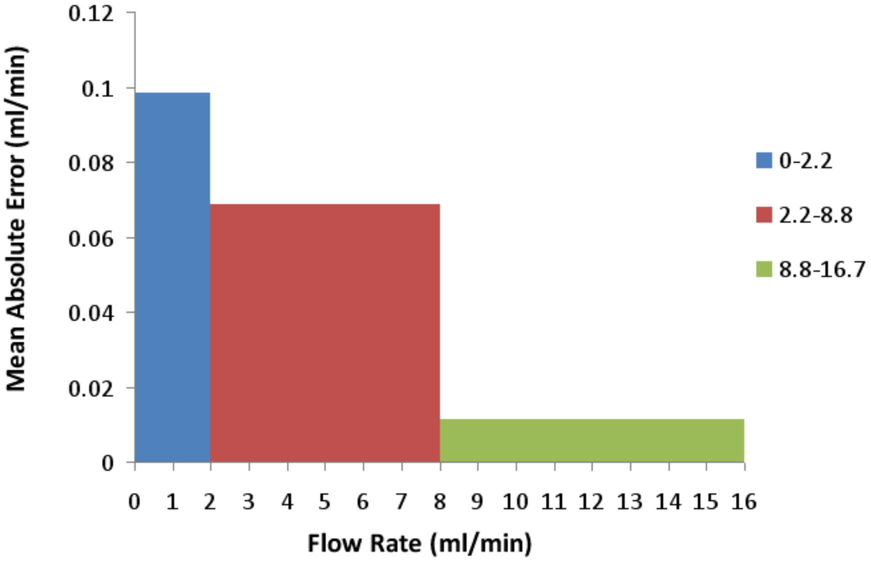


Figure 13 Mean absolute error in defining flow rate as a function of peak temperature arrival time.

5 Extended Time laboratory Study

A laboratory study was performed to evaluate the stability of the measurements over an extended period of time. Flow sensor SN302, without an external temperature detector or differential pressure transducer, was used to determine the repeatability of the measurements. The pump supplied a constant flow through the sensor. Measurements were repeated 296 times at a flow rate of 2.6 ml/min equivalent to Darcy velocities of 1.28 cm/day in our current configuration. There was no attempt made to control the ambient temperature beyond the controlled room temperature in the laboratory. Figure 14 demonstrates the long-term measurement repeatability in the laboratory. The mean travel time to the first thermistor down gradient from the heater (T5) was 104.8 seconds with a coefficient of variation of 0.013. The mean travel time to the second thermistor down gradient from the heater (T6) was 188.3 seconds with a coefficient of variation of 0.022. Based on the calibration curve these travel times would be equivalent to Darcy velocities of 1.36 and 1.24 cm/day with a coefficient of variation of 0.2. The error is less than 6.5% compared to the known pumping rate. As mentioned above the error increased with lower flow rates. The errors observed here are not different than those observed during calibration.

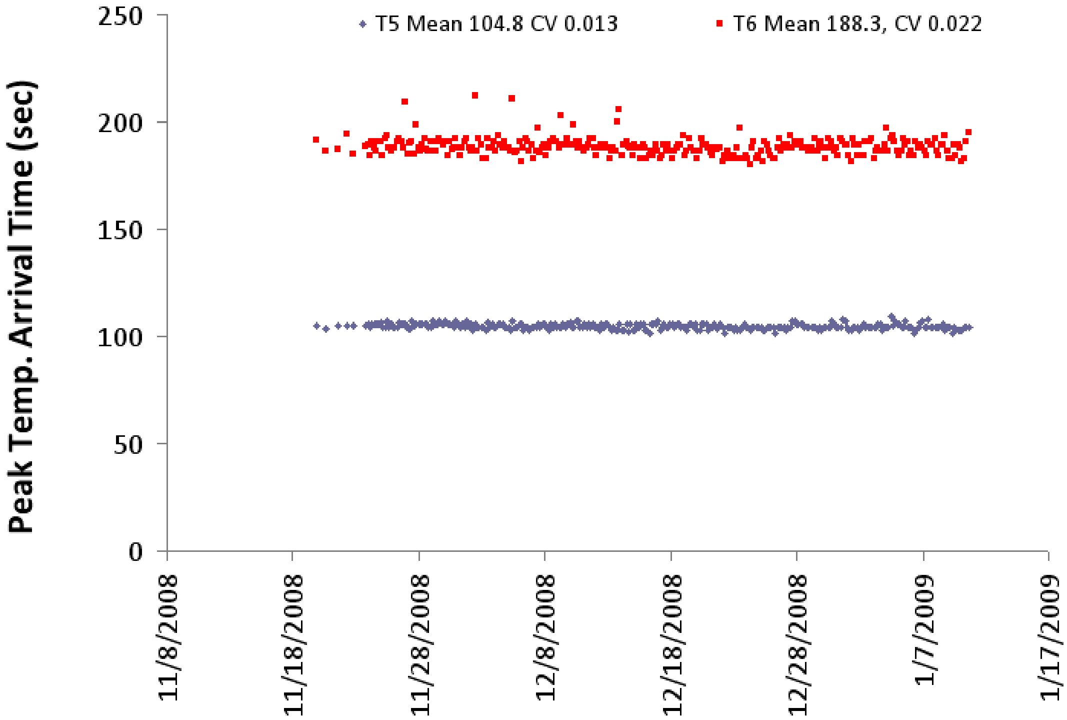


Figure 14 Demonstration of long-term measurement repeatability in the laboratory.

6 Field Deployment

Field testing of the long-term flux meter operation was carried out continuously from June 12, 2008 through April 7, 2009. The site selected was located in the Red Cove area of Plow Shop Pond adjacent to the Shepley's Hill Landfill Superfund site at Fort Devens, Massachusetts (Figure 15). Detailed site information was described in the report by Ford, Acree et al. (2009); as noted, groundwater was discharging to Red Cove under most conditions based on the potentiometric surface data and seepage flux measurements with the prototype flux meters. Note that the flux measurements within this report are specific to the area within Red Cove in which the automated flux meters were installed. The studies presented in this report were not designed to estimate seepage flux at the scale of Red Cove.

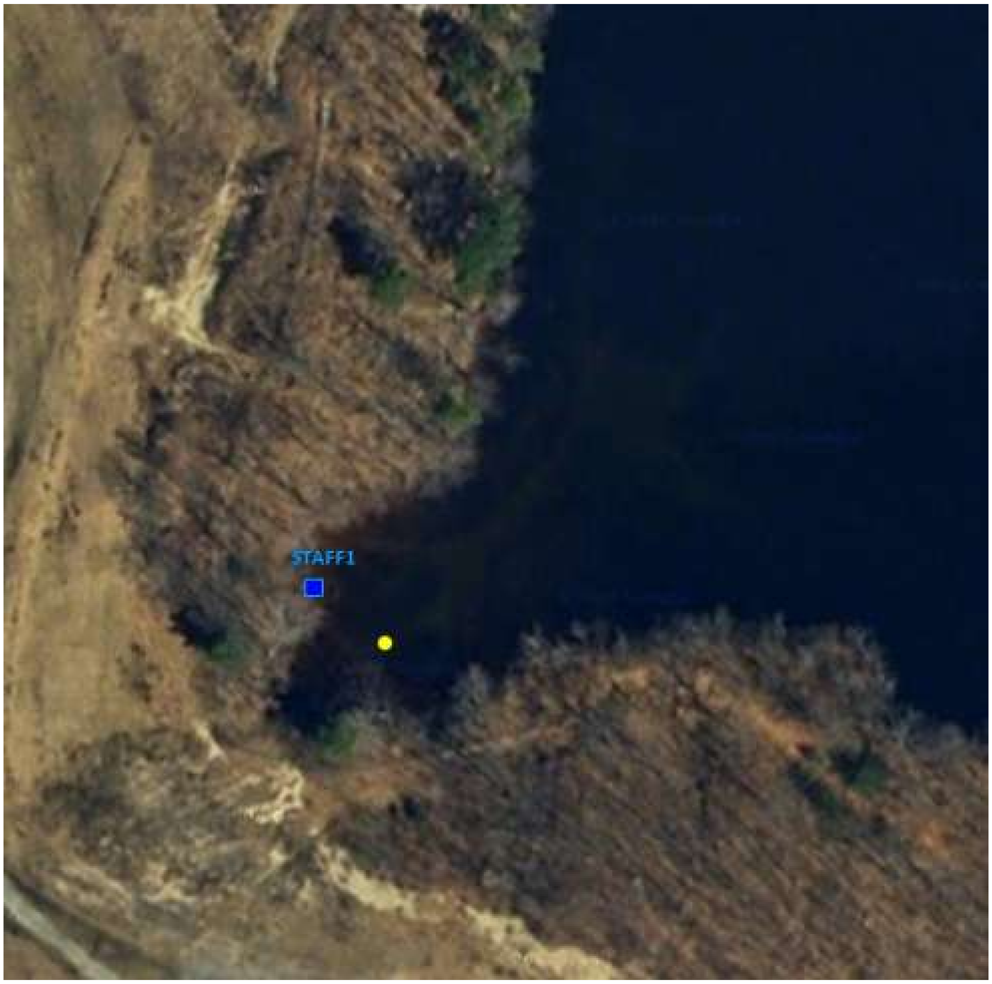


Figure 15 Aerial photo of the Red Cove area of Plow Shop Pond, Fort Devens, Massachusetts. The yellow dots mark the location of the instruments; the blue square marks the location of STAFF1 Red Cove staff gauge.

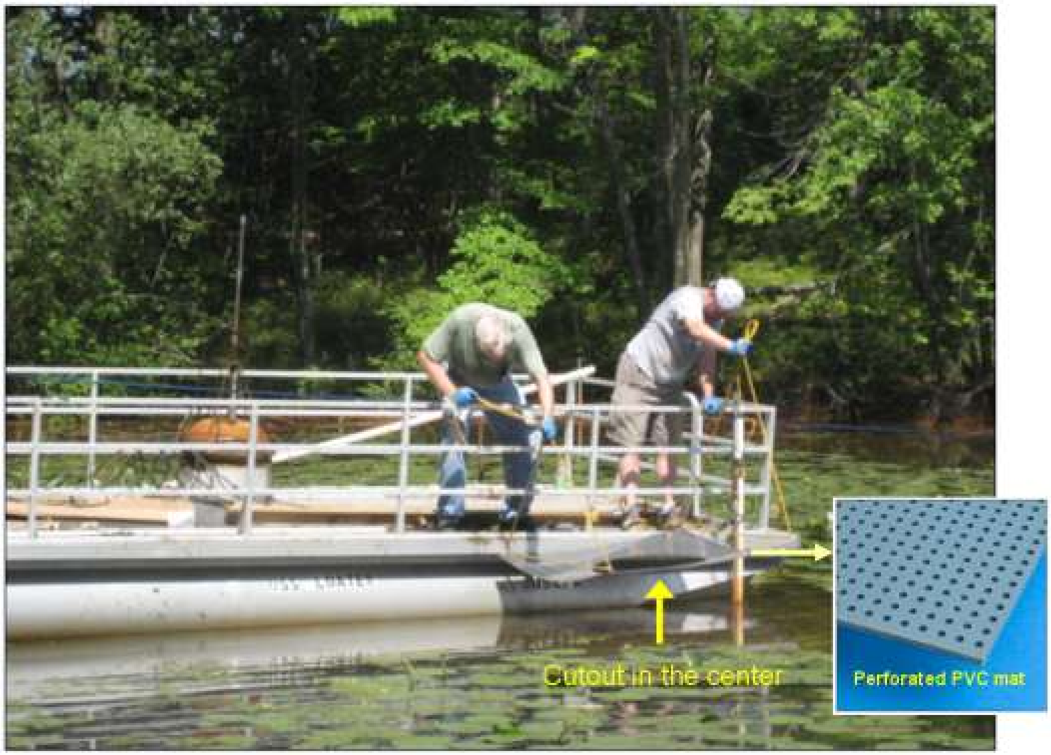
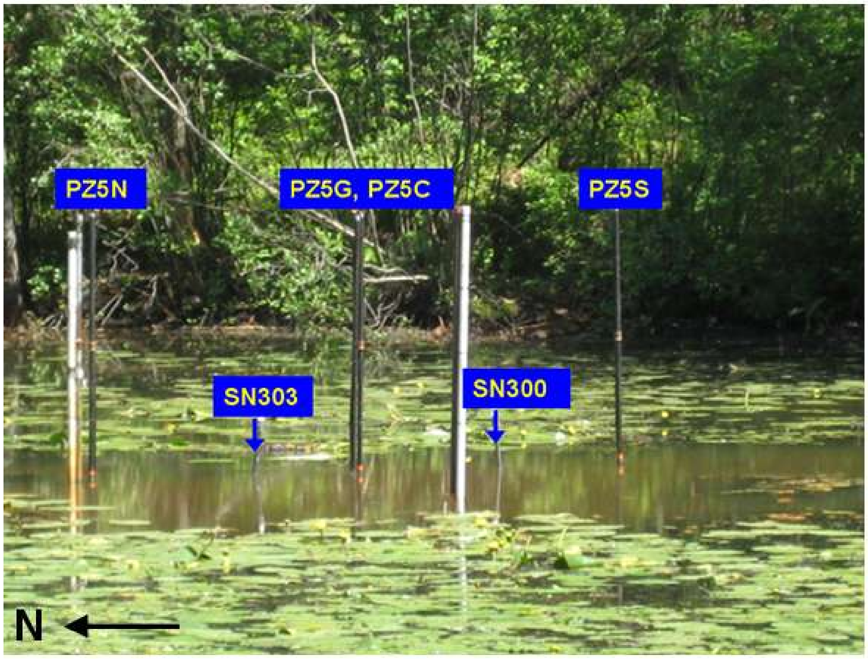


Figure 16 Placement of a perforated PVC mat prior to the flux meter installation. The perforated mat (4’ x 4’ in size, 1/4” thick with 1/4” hole-diameter) was placed on the sediment with a cutout in the center for the flux meter.

Sediments in this portion of the pond have a thick organic layer of material developed from falling leaves and branches. The surface sediment has very little load bearing capacity. To support the weight of the instrument, a perforated PVC mat (4’x 4’, 1/4” thick, 1/4” hole-diameter, 1/2” centers straight rows) was placed on the sediment with a cutout in the center for the flux meter (Figure 16). A perforated mat was used rather than a solid mat to minimize divergence of the water flow lines around the mat that could impact the measurements. While the mat design was configured to minimize impact to sediment conditions, it should be noted that the extent of the sediment compaction due to the placement of the mat and how it could be a factor in the observed discharge rates at this location was not evaluated as part of this study. Two flux meter units (SN300 & SN303) along with four piezometers (PZ5S, PZ5C, PZ5G & PZ5N) were installed in Red Cove as shown in Figure 17. The flux meters were deployed side by side for comparison. The meters were placed on separate pads but the pads were placed adjacent each other resulting in a center to center spacing of approximately 4 feet (1.2 m). The flux meter easily penetrated the sediment and was supported by the flange on the cylinder. The Southern flux meter was SN300 and the Northern meter was SN303. The three piezometers were placed along the west side of the pads and labeled PZ5N, PZ5C and PZ5S for the North, Center, and South piezometer, respectively. All three piezometers were installed at a 4 ft (1.2 m) vertical depth below the sediment. Staff gauge well PZ5G (bundled with PZ5C) was placed above the sediment surface to measure water stage data. Water lilies covered most of the cove at the time of field installation; the diver had to clear out the underwater vegetation before installing the flux meter units. The installation took place during the turnover of the reservoir and the diver reported near zero visibility underwater. There was no significant horizontal flow in this water body; the Bernoulli Effect was considered negligible. Thus there was no need to use the differential pressure transducer to assist in the placement of the dome. The water depth at the time of installation was 3.46 ft (105.5 cm). The instrument installation was completed on June 12, 2008 and the first data was collected at 9:28 PM Greenwich Mean Time (GMT). Monitoring continued until April 7, 2009 at 2:13AM GMT.



PZ5N [+42°33'16.80", -71°35'41.34"]
SN303 [+42°33'16.74", -71°35'41.10"]
PZ5G, PZ5C [+42°33'16.68", -71°35'40.86"]
SN300 [+42°33'16.56", -71°35'41.34"]
PZ5S [+42°33'16.56", -71°35'41.10"]

Figure 17 Deployed locations [latitude, longitude] of the flux meter and piezometer. Water lily covered most of the cove at the time of installation.

Daily meteorological data for the meteorological station at the Fitchburg Municipal Airport were obtained from the National Climatic Data Center of the National Oceanic and Atmospheric Administration. It is noted that the station is located approximately twelve miles from the study site. Although the exact magnitude of meteorological data at the Fitchburg station may be different from that in the Red Cove, the meteorological patterns should be sufficiently similar to represent the conditions at the site. Figure 18 shows the time trend (June 2008 - April 2009) of the water temperature measured at the SN300 flux meter location and the average air temperature at Red Cove. The majority of the time the water and air temperature at the site was below 10^{o}C during the sampling period, and was way below room temperature range during the calibration of the flow sensor.

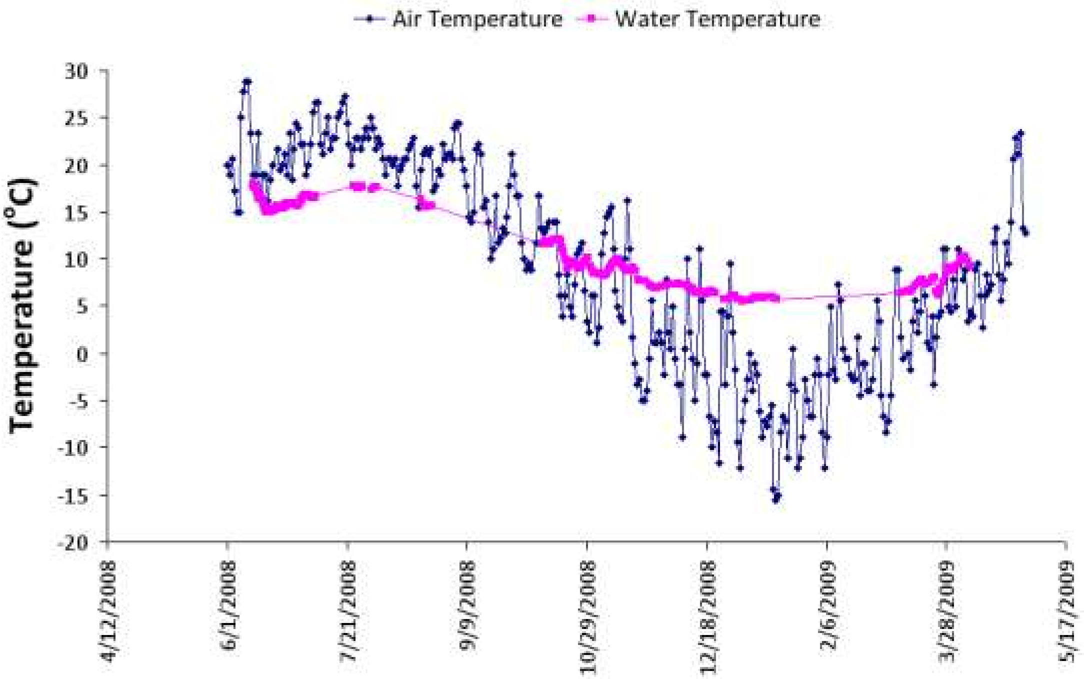


Figure 18 Time trend of water temperature and average air temperature at the Red Cove of Plow Shop Pond, Fort Devens, Massachusetts. Air temperature data were obtained from the Fitchburg Municipal Airport meteorological station, National Climatic Data Center of the National Oceanic and Atmospheric Administration.

6.1 Solar Power System

There was no power source available at the site. Therefore, a solar power system was installed to provide continuous electrical power for the long-term flux meter operation (Figure 19). The solar panel was installed uphill, south of Red Cove, for a better sunlight exposure. The solar system was designed for 300W peak power production. The average solar hours in the area is about 4.2 (Marion and Wilcox 1992), which should yield an annual average production of approximately 52W. The system would consume an average of 22W, if the flux meter measurement operated 4 times a day. The battery backup was designed to maintain power for up to 10 days. Initially there was no monitoring of the power system status after installation. In early July and late August, the voltage regulator/battery charger abruptly failed. After a second failure of the regulator, it was decided to begin monitoring the battery voltage. In Figure 20, a seven point moving average of the battery voltage is plotted along with a seven point moving average of the sky cover. A moving average was used to smooth out a significant amount of apparent randomness in the data. On the sky cover scale a zero is full sun and a 10 is 100% cloud cover. Although logic would suggest that when the sky cover went up the battery voltage might go down, there does not appear to be much of a correlation between the two. One possible explanation might be related to when the battery voltage was measured and how

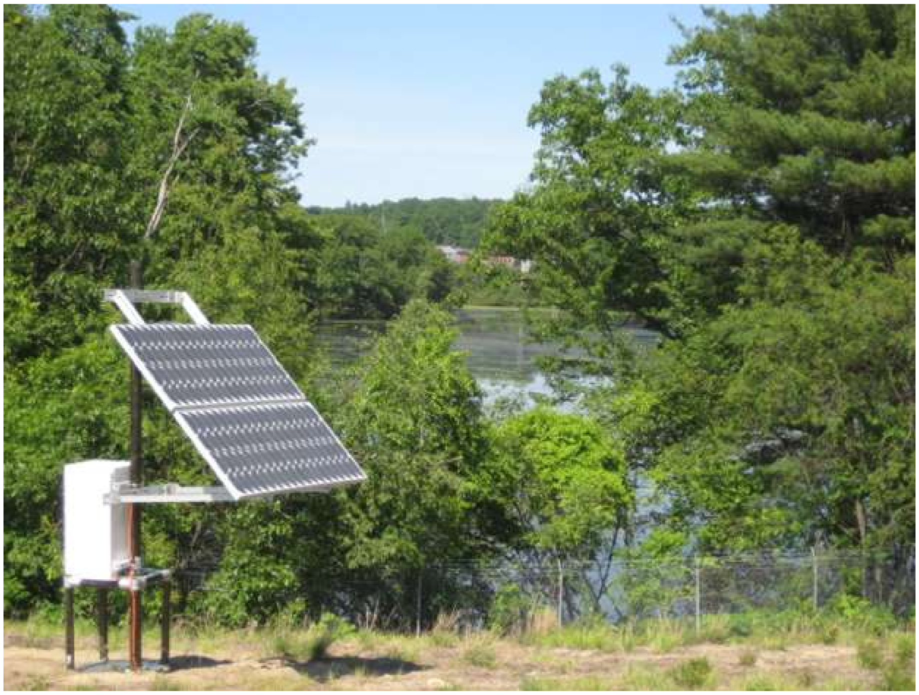


Figure 19 Solar power system was installed uphill south of Red Cove.

bright the sun was shining at the time of the measurement. The battery voltage was measured once per sampling cycle. The sampling frequency was not constant throughout the experiment. The sampling frequency was reduced during the winter to minimize power consumption. It should be noted that after the battery voltage was monitored there were no more problems with the regulator and the solar panel performed as expected. From a qualitative point of view the power produced was somewhat less than anticipated. While no setbacks were encountered after resolving the problem with the regulator, it was felt that the sampling frequency needed to be reduced to reduce power demand during the winter months.

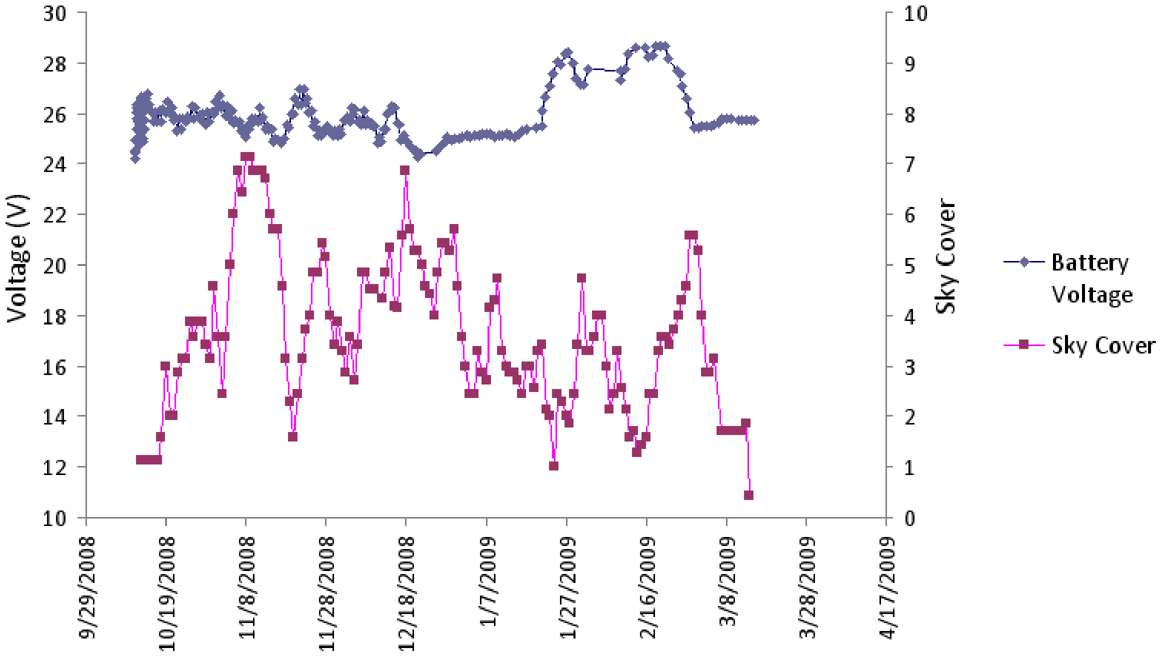


Figure 20 Battery Voltage Compared to Sky Cover. Sky cover data was obtained from the Fitchburg Municipal Airport meteorological station, National Climatic Data Center of the National Oceanic and Atmospheric Administration.

6.2 Heater Response

For reliable results the heater should produce a consistent amount of heat at the time of each measurement cycle. In the laboratory this was not an issue, because a regulated power supply was used rather than a battery, and very little variability was noticed in the measured peak temperature of the heater. In the field, the irregular battery voltage seemed to affect the consistency of the rise in the temperature of the heater. From the data collected between 10/10/2008 and 4/7/2009 (the battery voltage data collection did not begin until October 2008), there is a weak but apparent correlation (coefficient of correlation = 0.34) between the rise in temperature of the heater and the battery voltage (see Figure 21). In Figure 21, the correlation was influenced by a small fraction (11%) of data (associated with random dates over the measurement period), which appear to have a distinctive lower rise in temperature. The trends in the two groups of data appear to be similar. If we regard the data with a distinctive lower rise in temperature as outliers, the coefficient of correlation improves significantly to 0.82. This proves that the consistency of the rise in the temperature of the heater was compromised by the voltage irregularity. Although there was no indication that the correlation had a bad effect on the Red Cove data quality (based upon evaluating the shape of the thermograph as well as the peak arrival time), the power supplied to the heater should be regulated in future designs to prevent any doubts in the field.

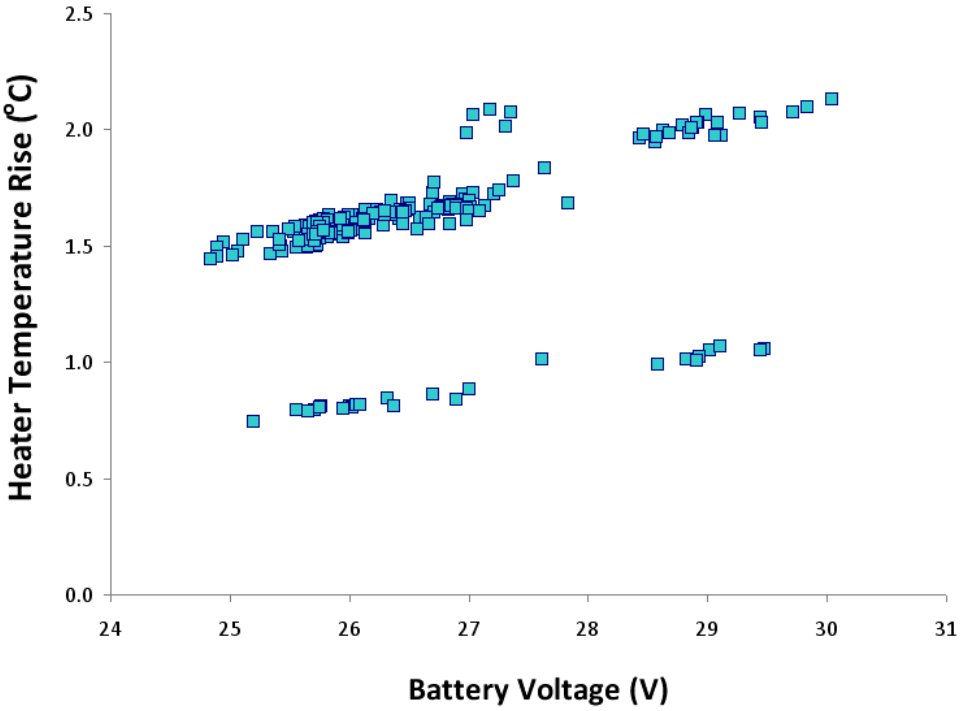


Figure 21 Heater Temperature Rise Versus Battery Voltage.

6.3 Pressure Transducer Response

In the field portion of the research at Red Cove, an attempt was made to monitor the groundwater-surface water flux. There are several approaches that can be used to estimate this flux. If one knows the hydraulic conductivity of the sediment one could monitor the vertical hydraulic gradient and calculate the flux. In a similar manner if you know the flux and vertical hydraulic gradient you can calculate the hydraulic conductivity. At Red Cove, piezometers were installed at four feet below the sediment-water interface. Pressure transducers (SETRA 5241-010wd-G-W1-2C-300-F) were installed in the piezometers and gauging station at common depths to record the vertical hydraulic heads electronically. Due to the faulty voltage regulator of the solar panel, the flux meter system was out of commission from 7/9/2008 to 7/24/2008 and from 8/26/2008 to 10/9/2008. There was no piezometric data available during these time periods. In both incidents the system regained its full functionality after the faulty voltage regulator was replaced.

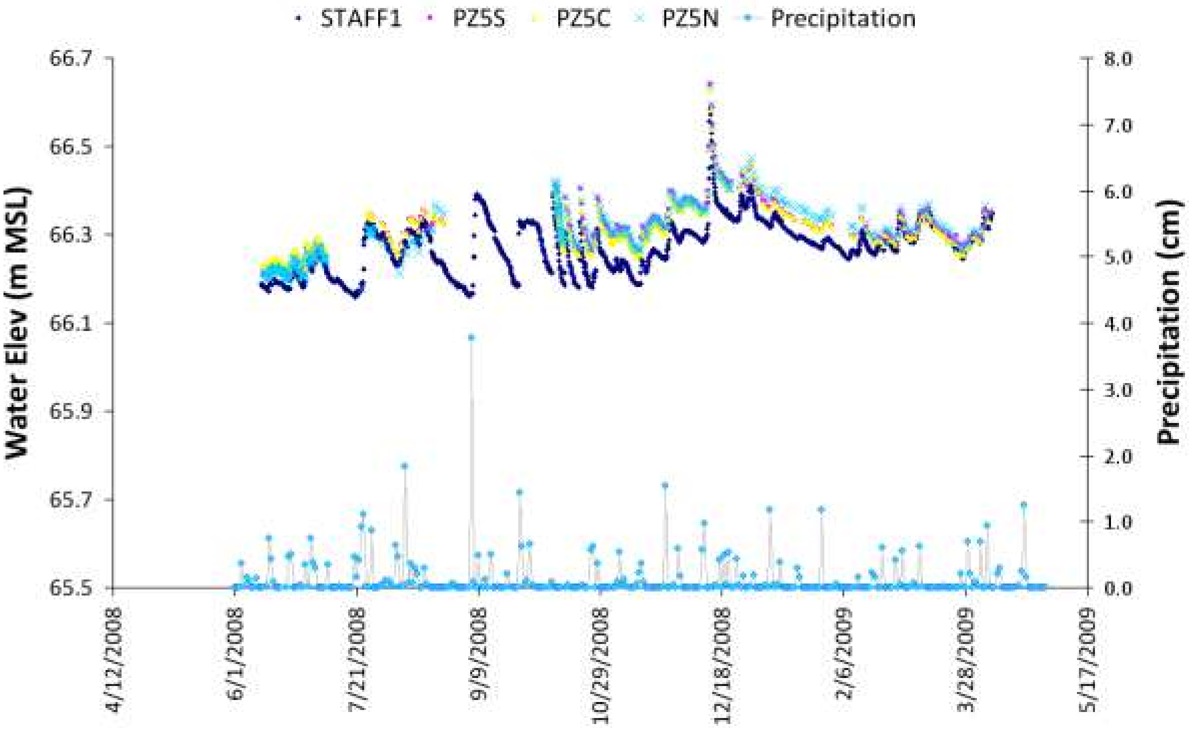


Figure 22 Comparative plots of water elevations (primary Y-axis) at piezometers PZ5S, PZ5C, PZ5N and Red Cove STAFF1 gauge station and time-series precipitation (secondary Y-axis). The daily precipitation data of Red Cove area was obtained from the Fitchburg Municipal Airport meteorological station, National Climatic Data Center of the National Oceanic and Atmospheric Administration.

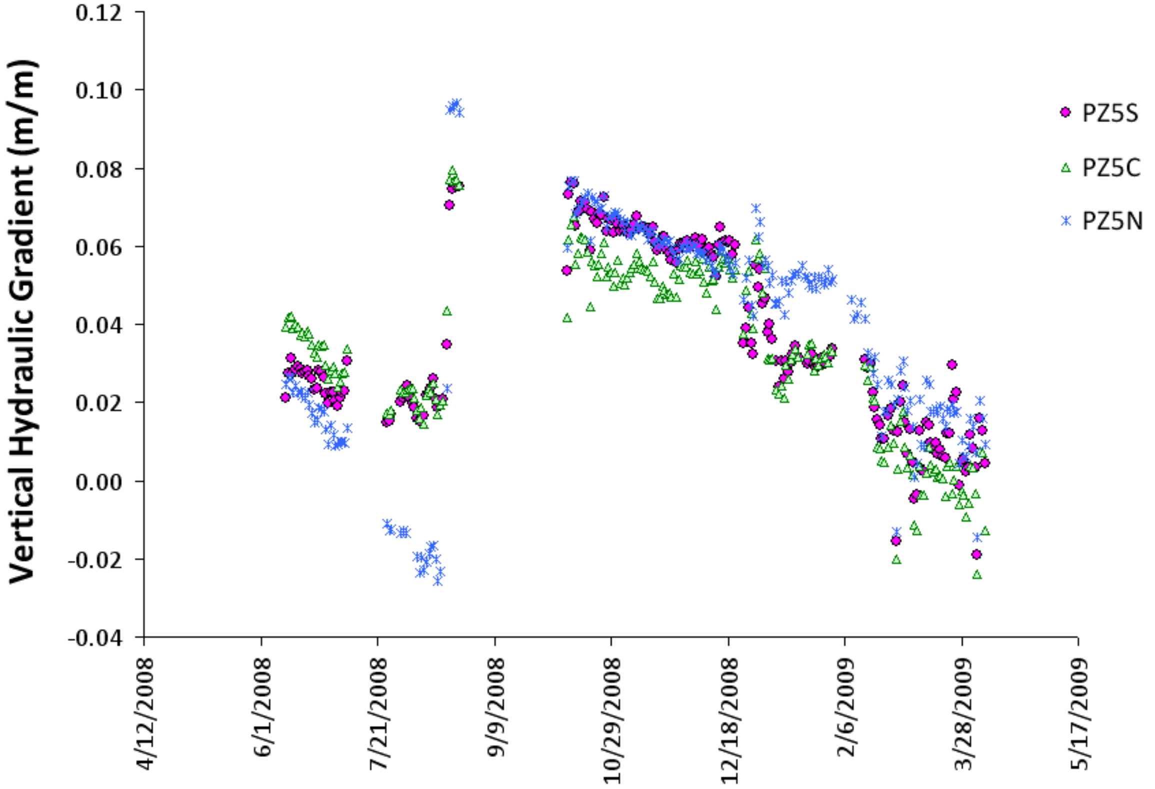


Figure 23 Temporal vertical hydraulic gradients at each piezometer location. Temporal variation of the three piezometers appeared to follow similar trend. The gradient peaked in August 2008 and then gradually descended over time.

In the field the vertical hydraulic gradient was manually determined three times, once on 6/13/2008, once on 8/20/2008 and once on 4/7/2009. The vertical hydraulic gradient is determined by the static elevation in the piezometer minus the static elevation of the water level in the pond, divided by the vertical depth of piezometer screen below sediment. For PZ5S, the vertical hydraulic gradients were 0.030, 0.033, and 0.030, respectively. For PZ5C, the vertical hydraulic gradients were 0.044, 0.041, and 0.040, respectively. For PZ5N the the vertical hydraulic gradients were 0.028, 0.024, and 0.026, respectively. Each time the manual measurements were made the pressure transducers readings were calibrated accordingly. Based on the pressure transducer data and field observation, it was determined the PZ5G staff gauge had physically slipped downward several feet into the sediment during winter time, which caused the water stage data to drift upward gradually over time. The exact time and rate of the occurrence were unverifiable. As a result, the PZ5G gauge data was proven unreliable and was excluded for any further use. Instead, the surface water elevation data obtained at the Red Cove STAFF1 gauge station were used. The water elevation of the piezometer was determined by the static head in the piezometer plus the surface water elevation at the Red Cove STAFF1 gauge. Figure 22 shows comparative plots of water elevations at piezometers PZ5S, PZ5C and PZ5N and STAFF1 gauge and time-series precipitation (secondary Y-axis). The daily precipitation data of Red Cove area was obtained from the Fitchburg Municipal Airport meteorological station, National Climatic Data Center of the National Oceanic and Atmospheric Administration. Temporal and spatial variations were evident as shown in Figure 22. Vertical hydraulic gradients were calculated for each piezometer location based on the water elevation data and the vertical depth of the piezometer below the sedimets. All three piezometers were installed at a vertical depth of 4 feet (1.22 m) below the sediment. Figure 23 shows the majority of the time the direction of the vertical hydraulic gradients were upward and their magnitude fluctuated daily. There is no verifiable reason why the vertical hydraulic gradient for PZ5N during the time span between 7/21/2008 and 9/9/2008 diverges so much from the other two piezometers. During the other times, all three piezometers seem to show consistent result.

Due to natural heterogeneity, it is not unexpected to see spatial variability in the piezometer data. Figure 23 illustrates minor spatial heterogeneity does exist in the vertical hydraulic gradient. Fifty two percent (52%) of the daily piezometric data have a spatial coefficient of variation smaller than 25%. Overall, 79% of the data have a daily spatial coefficient of variation smaller than 50%. Temporal variation appeared to follow similar trend for the piezometers. The vertical hydraulic gradient descended initially, ascended rapidly, peaked in August 2008, and then gradually descended over time. The gradient climax in August could be a cumulative hydrological response to the continual rainfall event from July 31 to August 15, 2008 (Figure 22) with a total precipitation of 10.8 cm. The apparent precipitation event from September 6 to September 14, 2008 yielded 12.6 cm (Figure 22). Because of the system stoppage, as mentioned above, there was no piezometric data recorded during that time. The hydrological response at this time was not verifiable. Nonetheless, Figure 22 clearly shows noticeable correlation between the rise in the surface water elevation and the precipitation event.

6.4 Flux Meter Response

Unexpectedly, the data collected from the field test was not consistent with the calibrations. The initial crude observation indicated that the advective flux values were extremely low. But many of the observed peak temperature arrival times in the field were longer than the peak temperature arrival times of the zero advection (or zero flow) condition observed during the laboratory test. This suggested that the calibration curves could not adequately describe the field data when the flow was extremely low. In the literature, the influence of very low temperature on the seepage meter measurements has been observed (Mwashote, Burnett et al. 2010). It was suspected that this temperature effect might have also played a role in our field measurements, given the fact that the average water temperature was 7.8 ± 5.1^{o}C during the field test, which was much lower than the room temperature condition during calibration.

After completing the field deployment, the flow sensors were retrieved and tested in the laboratory to investigate the phenomenon. The test was conducted at zero advection while the temperature of the water bath, where the flow sensor was placed was allowed to drift. Ice was used to cool the temperature of the water bath initially, and then the water bath was allowed to gradually return toward room temperature over time. Numerous cycles of measurements at

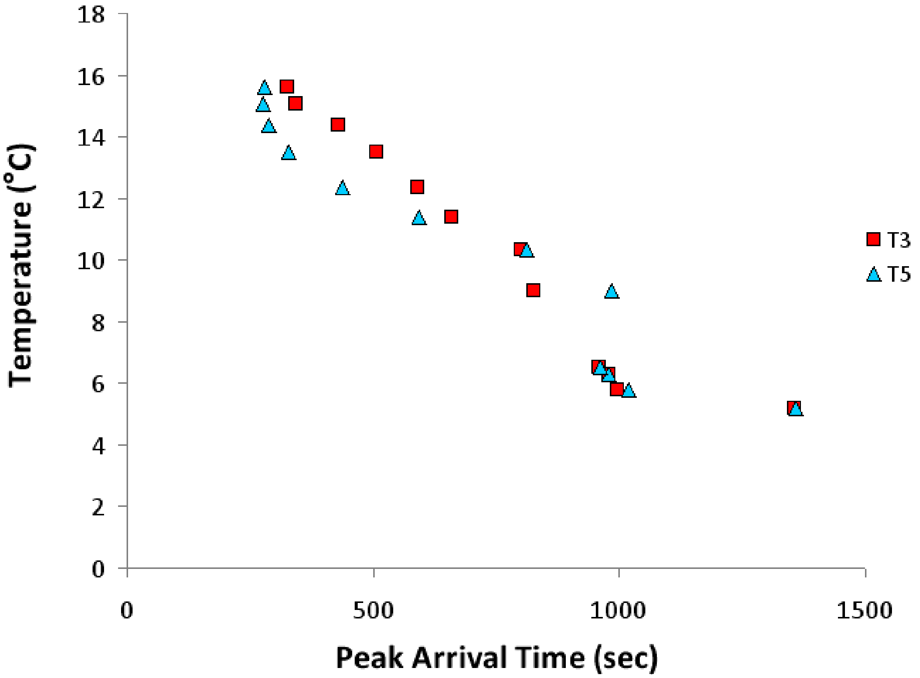


Figure 24 Effect of ambient temperature on the movement of heat-pulse in zero flow condition showing increasing peak arrival time with decreasing temperature.

zero flow were repeated. The temperature of the water bath and the peak arrival time at each cycle were recorded. In Figure 24, preliminary results show the effect of ambient temperature on the travel time of a heat-pulse under a zero flow condition. It is evident that the heat-pulse movement becomes sluggish as the water temperature decreases. Although it is reasonable to believe the temperature effect would become much less significant when advection dominates, this effect could have profound influence on field measurements when advection is low. Figure 24 illustrates the need to understand flow sensor performance based on knowledge of the temperature-dependent thermal conductivity of water and the PEEK material.

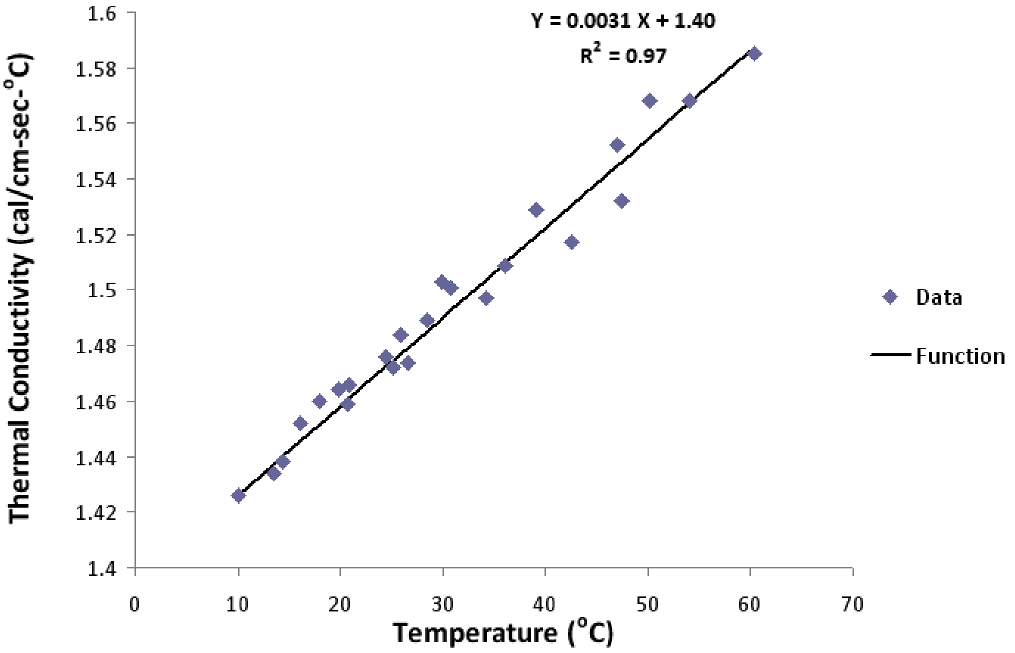


Figure 25 Martin and Lang's Thermal conductivity of water as a function of temperature.

Martin and Lang (1933) measured the thermal properties of water and demonstrated the thermal conductivity is temperature dependent (Figure 25). Lower thermal conductivity at lower temperatures is consistent with longer travel times at lower temperatures. Martin and Lang's data is almost linear over the temperature range evaluated, which shows decreasing thermal conductivity with decreasing temperature. This is consistent with our observations of approximately linear data points in Figure 24 with increasing peak temperature arrival time when the ambient water temperature decreases. However, the thermal properties of the PEEK material used to construct the flow tube are not the same as water. Local thermal equilibrium between the liquid and solid phases may not always be maintained. Runyan and Jones (2008) observed temperature-dependent thermal conductivity of PEEK at very low temperatures between 0.3 and 4 degree Kelvin. But at the present time, the literature source(s) for the temperature-dependent thermal conductivity of PEEK in the temperature range of this study have not been identified. Therefore, we can not use Martin and Lang's data alone to forecast the flow sensors response. Instead, an alternative approach will be used to establish an empirical relationship between the variables in the heat-pulse transfer.

6.4.1 Alternative Approach

At the location of the automated flux meter installation within Red Cove, the flow was limited to very low rates. While these low flow rates may not be significant from a total volume of flow, they may be important from an ecological standpoint for the sediment's flora and fauna, if the flow contains toxic chemicals. As mentioned above, the errors appear to increase as the flow rate decreases. At very low flows the temperature peak spreads over time and it becomes difficult to accurately determine when the peak occurs. As a result, the estimated flow rates become unreliable at low flows. To improve the resolution, an alternative method similar to the approach proposed by (Wang, Ochsner et al. 2002; Ochsner, Horton et al. 2005) for soil water flux determination was considered. At very low flows heat can be detected on both sides of the heater. At zero flow, if the flow tube is constructed symmetrically, the peak temperature that is observed should be the same at equal distances from the heater center. Likewise, if there is a very small flow in one direction the higher peak will occur in the direction of the flow. For flow rates where the heat can accurately be detected on both sides of the heater, one can estimate both the magnitude and direction of the flow from the ratio of the temperature of equally spaced temperature detectors. The form of the equation describing flow (Q) is:

$$Q = \beta \ \mathrm{n}\left(\frac{T_H}{T_L} \right) \qquad [5]$$

where β is a coefficient and T_H and T_L are the downstream and upstream temperature rises. Note that (Wang, Ochsner et al. 2002; Ochsner, Horton et al. 2005) define the term β as a function of the thermal conductivity of the porous media, the heat capacity of water and the distances from the heater to the upstream and the downstream temperature sensors. The heater and the temperature sensors were three parallel needles aligned in a commom plane, which is different from the configuration of our heat-pulse device. Because of the different boundry condition, we can not use the same expression in (Wang, Ochsner et al. 2002; Ochsner, Horton et al. 2005) to define β. Instead, a mathematical fitting of data was used to establish the empirical expression of β.

Intuitively, if there is no water flow, the temperature rise in each of the two temperature detectors spaced at an equal distance from the heat source should be the same. Likewise, if there is a very small amount of flow, the convection of heat by the flowing water resulted in a larger temperature increase downstream from the heat source than upstream from the heat source. With this concept in mind, the ratio of the rise in temperature downstream to the rise in temperature upstream as a function of flow rate was evaluated. The natural log of the ratio appears to be directly proportional to the flow rate at a constant temperature. In Figure 26, the response is shown for the data collected at two different temperatures. The temperature effect is significant. As the flow rate approaches zero, the importance of the thermal conductivity dominate over the advective processes. Based on the work of Martin and Lang it is assumed the coefficient β in Equation 5 is linearly correlated to the temperature. Using the data in Figure 26, we determined by mathematical fitting of the data, the linearly temperature dependent $\beta = 0.0301 + 0.0287\ T$, where T is the temperature in Celsius. Figure 27 shows a comparative plot of measured flow rate versus temperature corrected estimated flow rate, when attempting to estimate flow rates up to 0.3 ml/min through the flow tube over a temperature range of 5 to 15°C. The estimated flow rate follows $Q = (0.0301 + 0.0287T) \ln (T_H/T_L)$, where the flow rate Q is in ml/min and the ratio T_H/T_L is the temperature rise of the downstream thermistor divided by the temperature rise of the upstream thermistor. T is the ambient water temperature in Celsius. The flow direction is determined by the thermistor with the higher temperature rise. The RMSE of the estimated flow rate was 0.013 ml/min, equivalent to a Darcy velocity of 0.006 cm/day in our current dome cylinder configuration. This is adequate for the measurements being made. It is known that over the flow range studied, temperature affects are significant, and the lower the flow rate the more significant the effect. Equation 5 should only be applied to low flow situations where advective heat flow no longer dominates. For the data set collected at Red Cove, Equation 5 modified with a thermally dependent coefficient has provided a valid way for interpreting the data. The empirical equation is assumed universally applicable to both flow sensors for the reason of their near identical geometry in the construction.

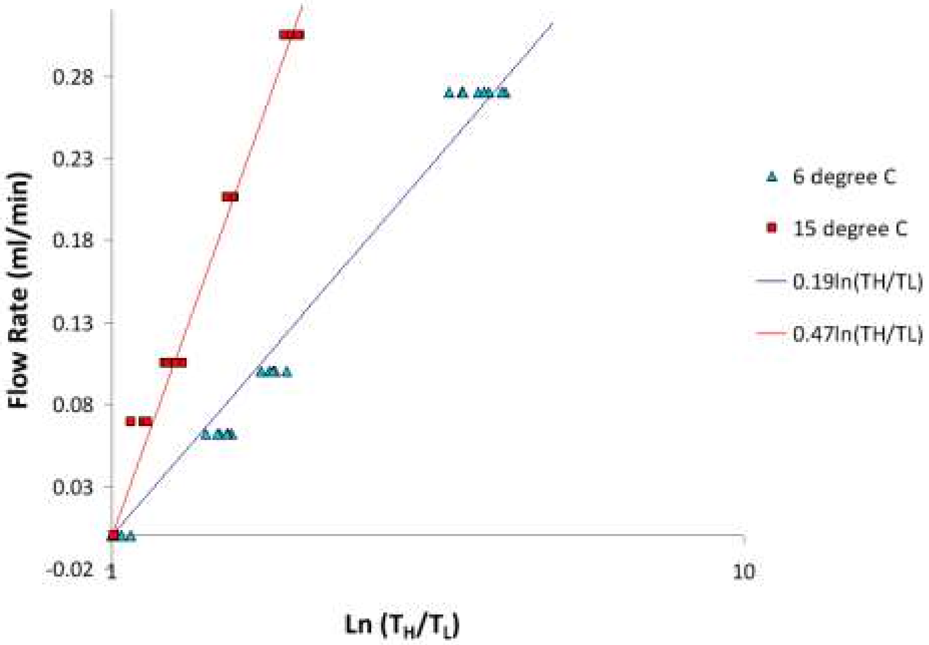


Figure 26 Ambient temperature effect on estimated flow rate.

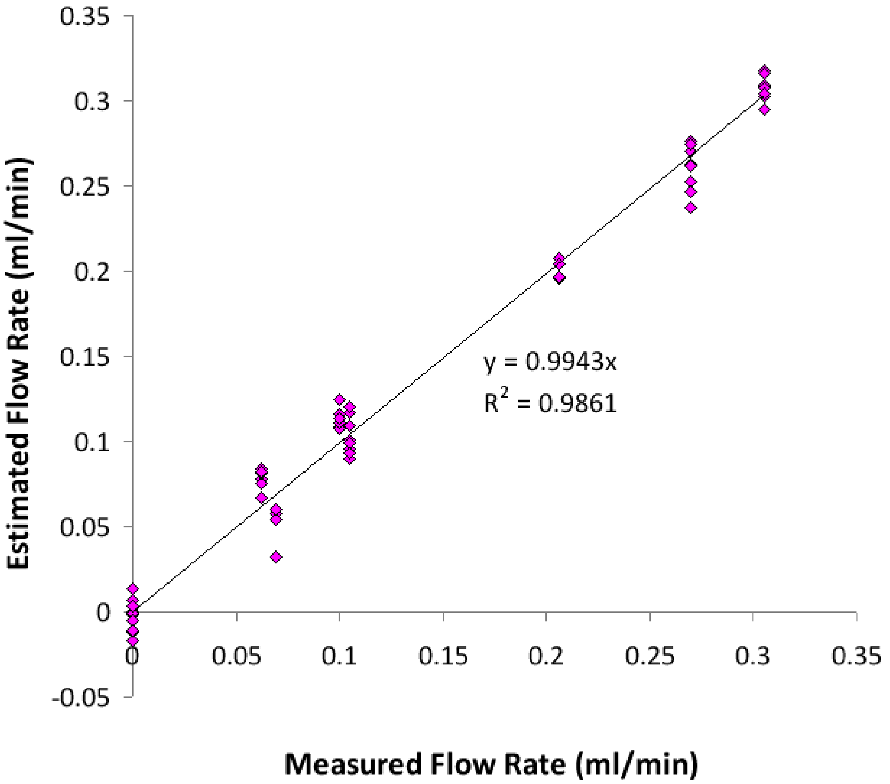


Figure 27 Comparative plot of measured flow rate versus temperature corrected estimated flow rate. The RMSE of the estimated flow rate was 0.013 ml/min.

6.4.2 Flux Meter Results

Monitoring was continuous at various frequencies over the time period except when there were problems with the solar panel's voltage regulator. Sampling frequency was reprogrammable from any remote location with Internet access. Data were downloaded daily for analysis. During the winter months when the water surface was ice-covered, data could not be interpreted with certainty.

Temporal advective flux data for SN300 and SN303 excluding the period of January 17 - March 8, 2009 is shown in Figure 28. The daily sampling frequency was not constant; each data point represents a daily average value. Piezometric data was collected once per measurement cycle. The field data shows a significant amount of scattering, which was not seen under controlled laboratory conditions of constant flow. Data in Figure 23 suggests that there were fluctuations in the vertical hydraulic gradients, which could be the cause of the variability observed in the measured seepage flux.

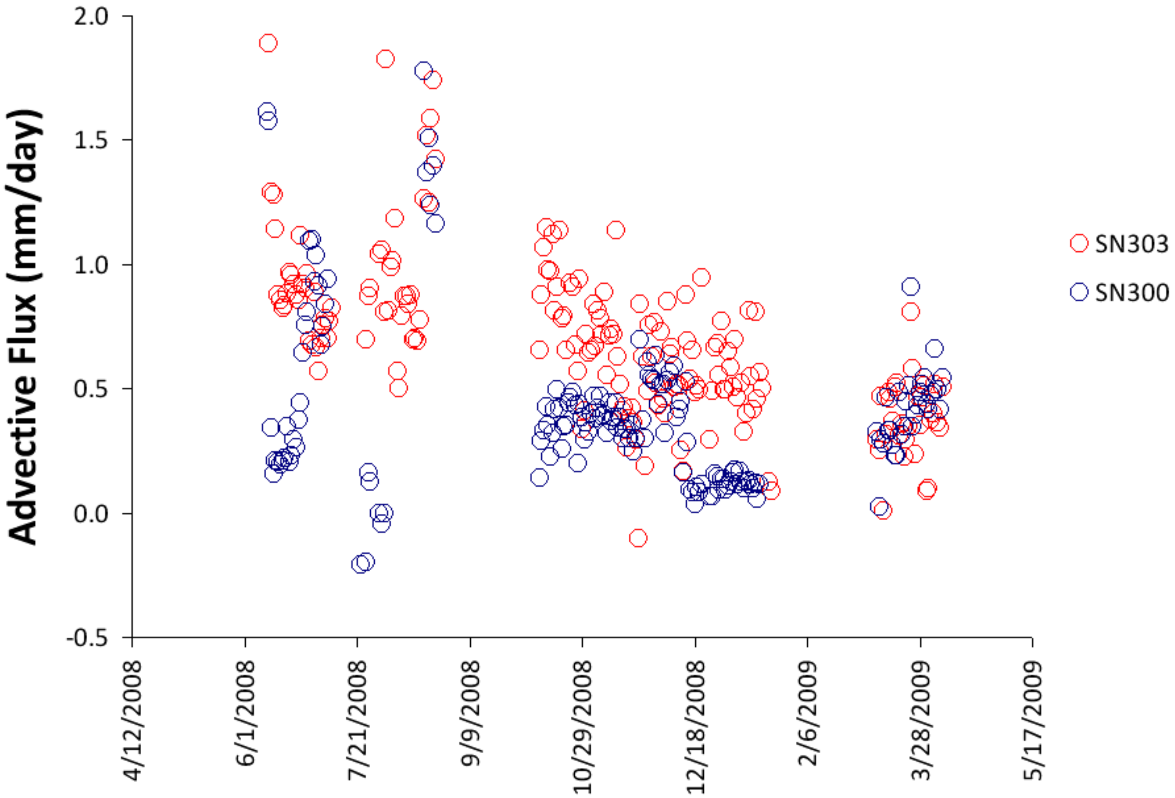


Figure 28 Daily average advective flux data at the Red Cove of Plow Shop Pond, Fort Devens, Massachusetts. The data were corrected according to the ambient water temperature.

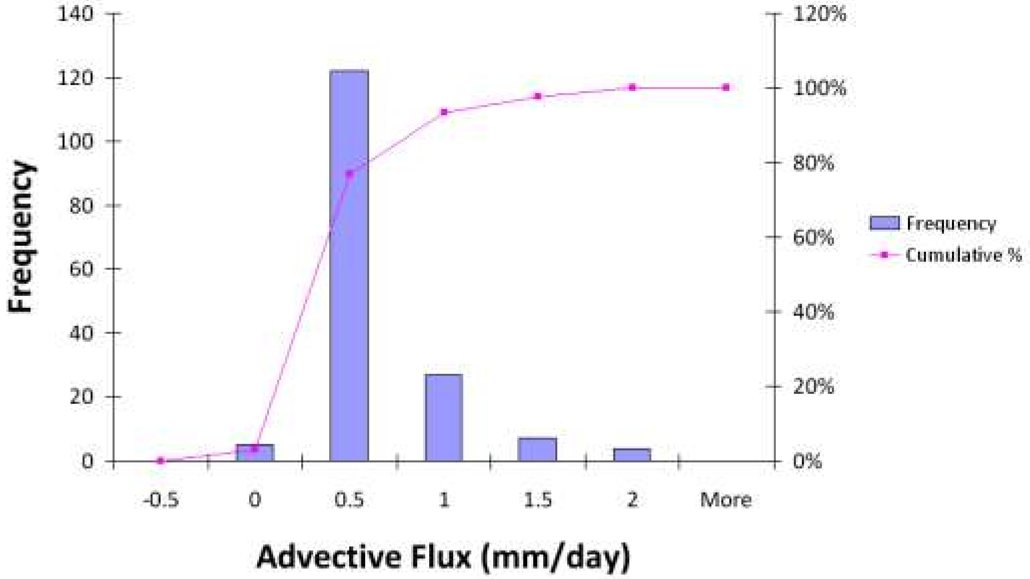


Figure 29 Histogram of advective flux measured at Red Cove flux meter location SN300.

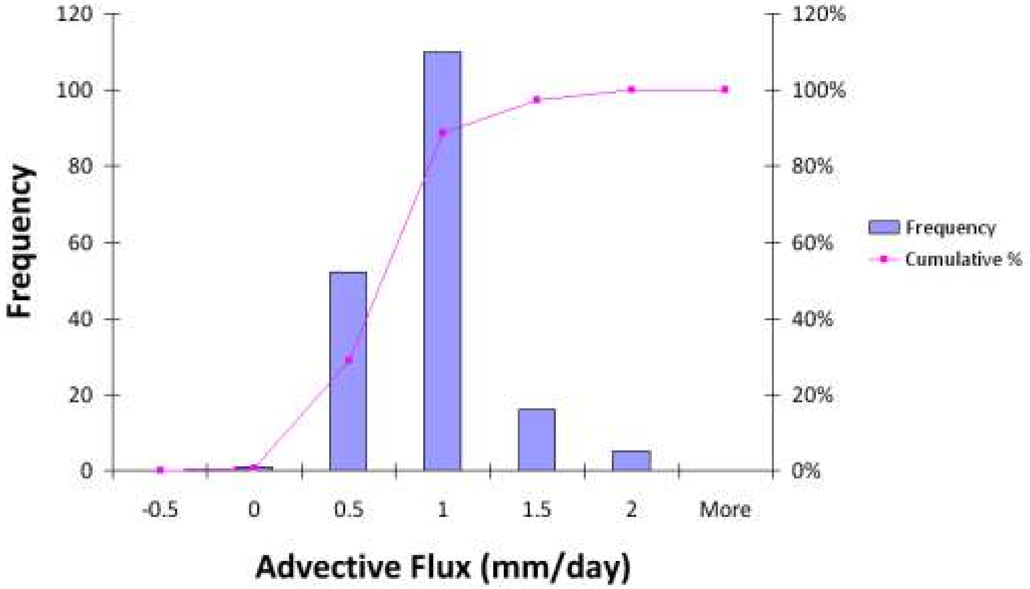


Figure 30 Histogram of advective flux measured at Red Cove flux meter location SN303.

Due to the natural heterogeneity, it is expected that the vertical hydraulic gradient at the flux meter locations were, to some extent, different from the piezometer locations. The gradients at piezometers were low with sporadic negative values at times. All of the data suggest low flow and the majority of the data suggest an upward flow direction from the groundwater to the surface water. Data also superficially suggests a trend that the flow magnitude was declining over time. The average advective flux, in terms of Darcy velocity, for SN300 was 0.44 ± 0.34 mm/day (median = 0.36 mm/day; minimum = -0.34 mm/day; maximum = 1.81 mm/day). Figure 29 is the histogram of the flux measured at SN300. It is evident that the advective flux was extremely low with 77% of the measurements less than 0.5 mm/day and 93% less than 1 mm/day. The average advective flux for SN303 was 0.76 ± 0.33 mm/day (median = 0.77 mm/day; minimum = -0.45 mm/day; maximum = 1.89 mm/day). Figure 30 is the histogram of the advective flux measured at SN303. Flow measured at SN303 was also extremely low but slightly higher than flow measured at SN300, with 89% less than 1mm/day and 97% less than 1.5 mm/day. The data scattered appears to be less in the spring than during the entire time. The average spring (3/9/2009 – 4/7/2009) advective flux for SN300 and SN303 was 0.42 ± 0.16 mm/day and 0.38 ± 0.16 mm/day, respectively. Spatial heterogeneity in the advective flux does exist, eventhough the two flux meters were placed only 1.2 m apart center to center.

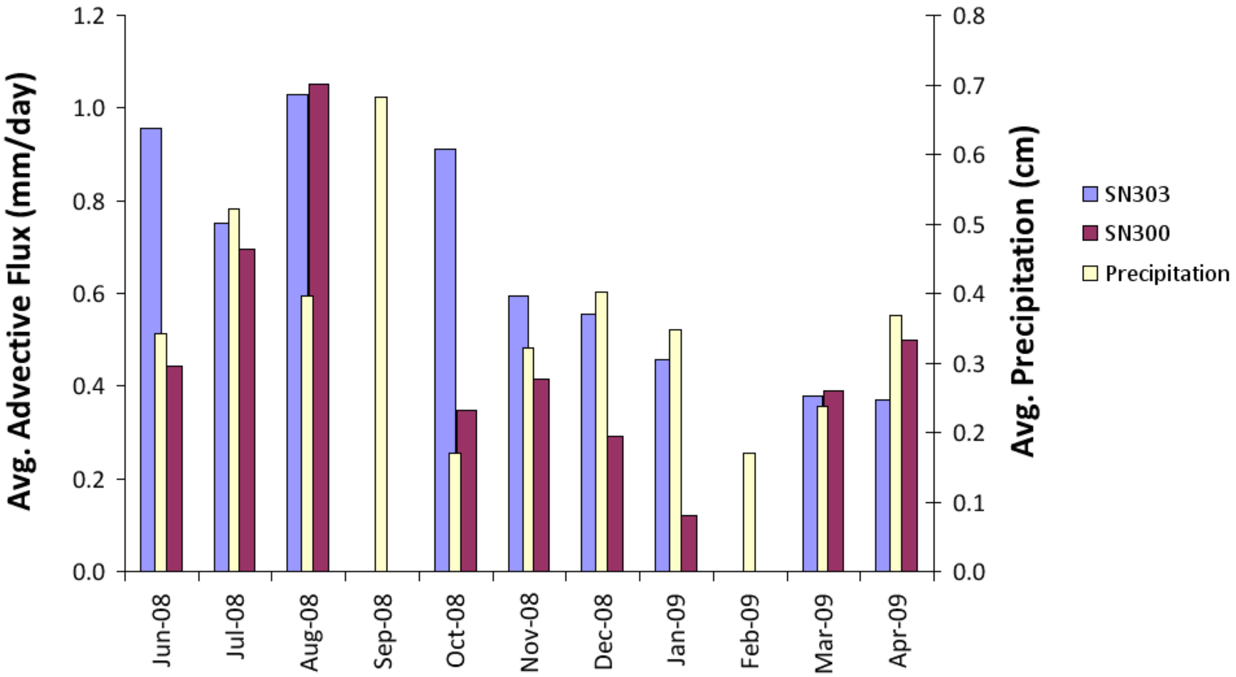


Figure 31 Monthly average advective flux and precipitation.

Figure 31 illustrates the relationship between monthly average advective flux (primary Y-axis) and precipitation (secondary Y-axis). From observations, there appears to be a qualitative association between the flux and the precipitation. Flux was higher during the summer and early fall when precipitation was also relatively higher. Flux was lower during winter and early spring when precipitation was lower. The average precipitation in September 2008 was elevated, but unfortunately no flux data was available during that time due to the faulty solar power system. The elevated flux values in October could very well be the result of the large precipitation in September.

There were problems analyzing the winter data, from January 17, 2009 to March 8, 2009, when uncharacteristic thermographs resembling wave-driven flow were noticed. Both flux meters experienced the same phenomenon. The wave influenced artifact was similar to the thermographs observed by Lien (2006) at Lake Hartwell. It was noticed that the surface water of Red Cove was covered with ice during the winter months. Figure 32A shows a representative data set for the winter months. Thermograph of the heater and the two thermistors T3 and T5 nearest to the heater are shown. Figure 32A suggests that the flow in the flow tube was changing direction back and forth; the thermograph of T3 and T5 appears to be conversing with each other indicating continuous alternation in flow direction. The oscillation frequency was about 3.5 minutes. It was suspected this oscillating pattern is comparable to the effect of wave-induced flow observed by (Smith, Herne et al. 2009). Although with waves this is intuitive, when the surface is frozen there should not be any waves. Nevertheless, if there were oscillating instability of the overlaying ice mass it could potentially create a pumping effect in the ice-covered water akin to waves. The data was unexpected and we did not collect pressure gradient data near the sediment interface to substantiate alternation of the pressure gradients to be the cause of the oscillating flow. Uncertainty of meaningful data interpretation could be attributed to the winter data (from January 17, 2009 to March 8, 2009) being omitted from the data analysis. Figure 32B is the type of smooth thermographs observed after the ice was thawed and is typical of much of the remainder of the data, which proves the uncharacteristic winter data was not caused by any instrument malfunction. It is safe to conclude, the flux meters were functional throughout the field deployment.

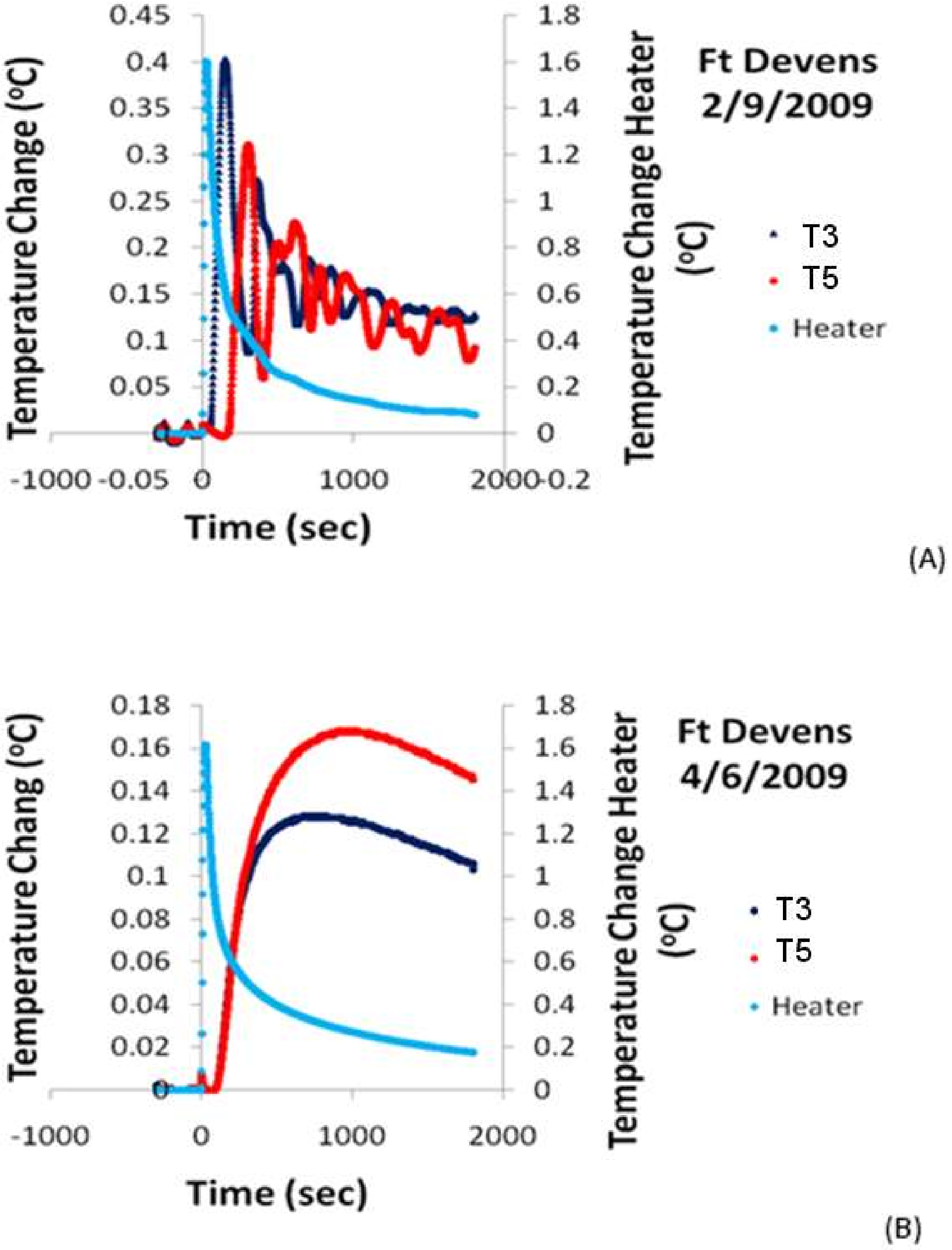


Figure 32 Apparent flow reversals during measurement cycle. Winter processed data (A), spring processed data (B).

6.5 Flux Estimation using Darcy's Law

Another way of estimating seepage flux is to apply Darcy's law (Hubbert 1956) using hydraulic conductivity and piezometric data. Darcy's law is expressed in the following equation:

$$q = \frac{Q}{A} = K\frac{\Delta H}{L} \qquad \textbf{[6]}$$

where q is flux (L/T), Q is flow rate (L^3/T), A is area (L^2), K is hydraulic conductivity (L/T), and (ΔH/L) is the vertical hydraulic gradient. To obtain the hydraulic conductivity, a falling-head permeameter test was conducted on April 8, 2009 adopting the same approach by (Chen 2004; Genereux, Leahy et al. 2008). A permeameter was installed closely adjacent to the flux meters. The permeameter was installed on April 7, 2009, which allowed the disturbance of the formation to settle before taking measurements the next day. Figure 33 is a schematic of the field permeameter test. The test involves inserting a pipe (permeameter) vertically into the sediments, filling the pipe with water, and measuring the rate at which the water level inside the pipe falls; hydraulic conductivity of the section of sediments isolated is then calculated from this rate.

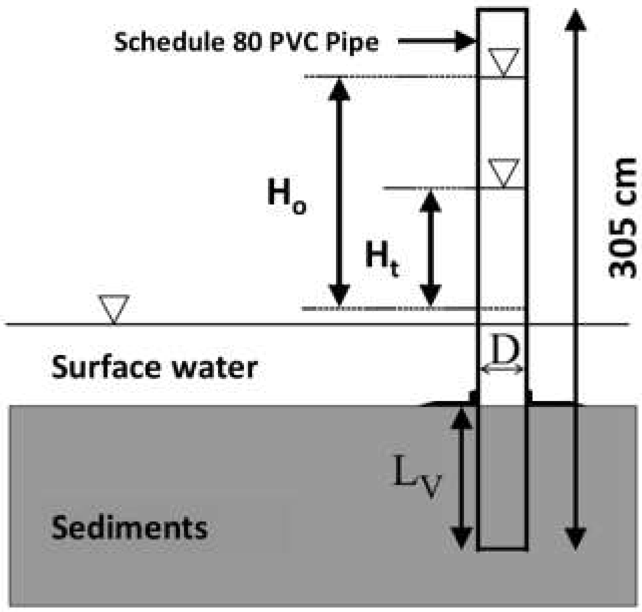


Figure 33 Falling-head permeameter test to measure sediment hydraulic conductivity.

Chen (2004) proposed a simplified equation in the form of Equation 7 to estimate vertical hydraulic conductivity K_V:

$$K_V = \frac{L_V}{(t - t_0)} \ln\left(\frac{H_0}{H_t}\right) \qquad \textbf{[7]}$$

where L_V is the length of the sediment column; H_o and H_t are the hydraulic heads inside the pipe corresponding to time t_o and t. Chen (2000) suggested the error in estimate is less than 5% when $L_V/D > 5$ with D denoting the inner diameter of the pipe. The error can be reduced by increasing the ratio of L_V/D. Equation 7 could be derived in the form of Equation 8 (Genereux, Leahy et al. 2008). Hydraulic conductivity K_V is calculated from the slope of the best fit line.

$$\ln H_t = -\frac{K_V}{L_V} t + \ln H_0 \qquad \textbf{[8]}$$

Our permeameter is a schedule 80 PVC pipe 14.4 cm in inner diameter and 305 cm in length. The bottom edge of the pipe was shaped in an outward bevel to help minimize sediment compaction during installation. The permeameter was driven 91.4 cm into the sediment with a hammer. Figure 34 shows the results of a falling-head permeameter test. Hydraulic conductivity was calculated from the slope of the trend line. The results show at this particular location that the hydraulic conductivity is 0.031 m/day, which is a typical value for silt.

Temporal advective flux was then calculated based on the hydraulic conductivity and the average vertical hydraulic gradient of the three piezometers. Similar to the measured values the calculated flux values were small. The average calculated flux was 1.12 ± 0.71 mm/day (median = 1.15 mm/day). Comparison between the measured and calculated flux is shown in Figure 35. Although the permeameter and flux meters were closely adjacent they were not at the exact same location. Therefore, spatial variability of the flux value should be anticipated. Despite the inherent spatial heterogeneity the calculated and measured flux were convincingly comparable, which also confirmed the validity of the two methods. Results from both approaches suggest a very low flow, and the majority of the data suggests an upward flow direction from the groundwater to the surface water. The magnitude of flow compare favorably in the early summer of 2008 and in the spring of 2009. During the fall and winter months the calculated flux values were higher than the measured values, but nevertheless remain in the same order of magnitude as the measured values.

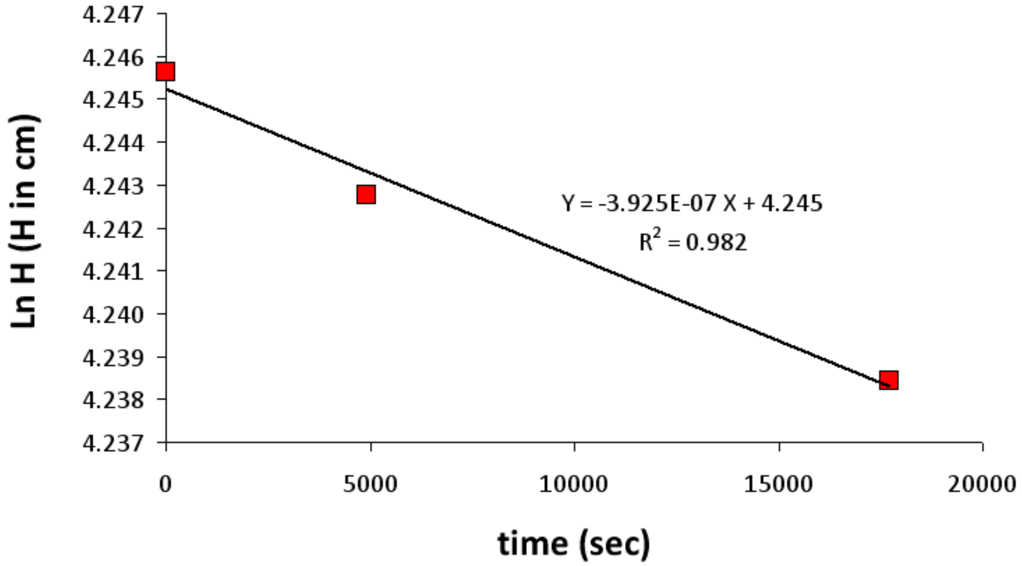


Figure 34 Falling head permeameter test at the Red Cove of Plow Shop Pond, Fort Devens, Massachusetts. Hydraulic conductivity is proportional to the slope of the best fit line.

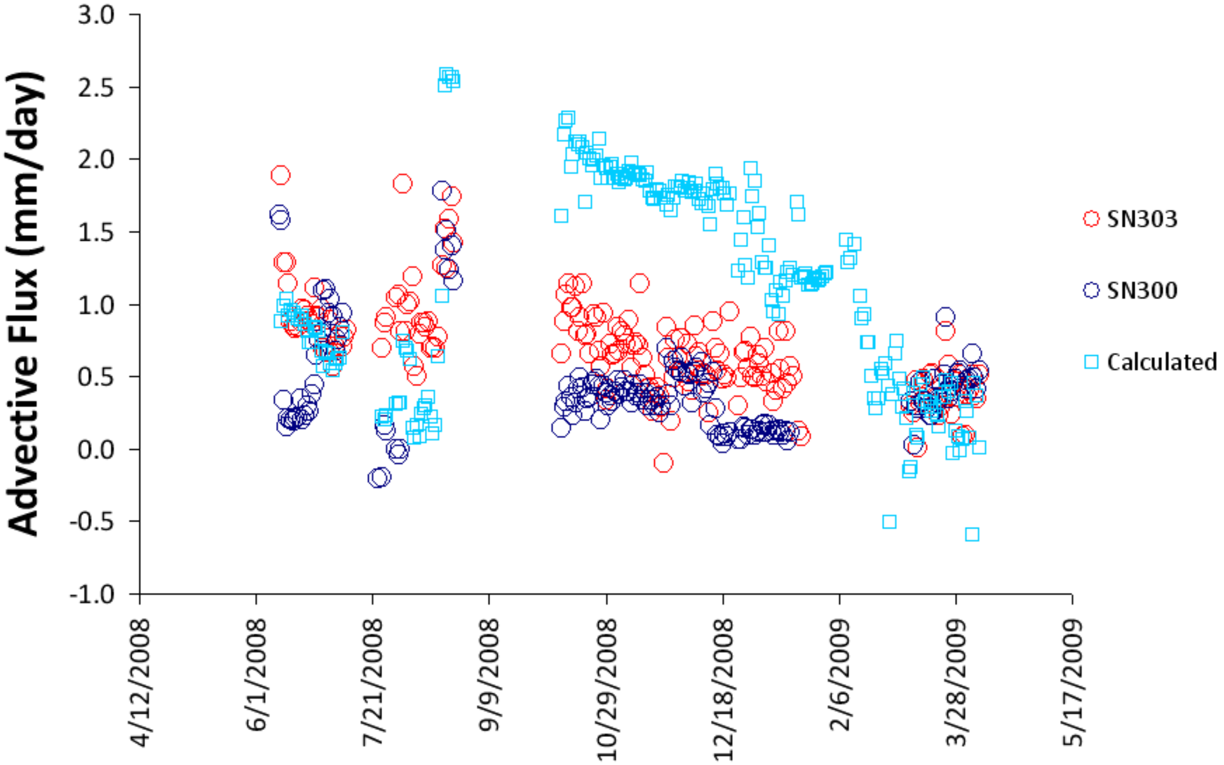


Figure 35 Comparison between measured and calculated advective flux. The calculated flux was based on hydraulic conductivity and vertical hydraulic gradient measurements.

7 Summary and Discussion

In Situ long-term monitoring of seepage flux across the sediment-water interface is necessary, for two reasons. Seepage flux is temporally heterogeneous in nature, it is central to collect and analyze repeated measurements to evaluate changes in condition, and the long-term monitoring provides a key to evaluating *in situ* performance of the flux meter. For contaminated sediment sites, monitoring temporal (as well as spatial) variation of advective flux is essential to proper risk management. This report presents the development and field testing of a ruggedized advective flux meter capable of unattended remote monitoring of sediment-water interfacial flux. The flow sensor employs a heat-pulse technique; the bi-directional feature of the flux meter is realized through the temperature measuring capability on either side of the flow tube. In the field operation, flow across the sediment-water interface is funneled through a dome to a flow tube; the rate of water flowing through the flow tube is measured. The advective flux, in terms of a vertical Darcy velocity, is calculated by dividing the flow rate through the flow sensor by the cross-sectional area of the dome. The dome serves as an amplifier to the generally low Darcy flow rates. A remote controlled monitoring system is included to permit system monitoring and modification over the internet. The sampling frequency is changeable through the wireless network. Data is downloadable from any remote location with Internet access after each measurement cycle.

The system has automatic data acquisition, real time data display, and signal display/analysis capability. The instrument has undergone several calibrations to establish relations between flow rate and heat-pulse travel time. Under typical condition when advection dominates, time of peak temperature arrival is inversely propotional to the flow rate. For very low flow rates when advection no longer dominates, the spread of the heat-pulse made it difficult to determine precisely when the peak arrived. A sensible approach using temperature rise ratio was adopted to provide a creditable alternative for determining flow rate. Using this approach it is possible to accurately interpret flow rates down to zero flow conditions. Depending on the magnitude, flow rate can be derived as a function of peak temperature arrival time or as a ratio in the rise of temperature. Using these two methods, reliable measurements can be made over a wide bi-directional range of flow rates.

The extended laboratory study of flux meter performance demonstrated the ability to repeat measurements accurately. In the field, unattended remote monitoring of seepage flux measurement was demonstrated at Fort Devens. The automated operation ran continuously for ten months with a self-sufficient power supply system. The only problem encountered in the field was with a faulty solar panel voltage regulator. The system performed well after the glitches were resolved. Two flux meters were placed side by side in shallow water. They successfully monitored the water flux from June 2008 to April 2009. The flux meter is capable of quantitatively measuring very low flow rates. All of the data suggest very low flow at the flux meter locations and the majority of the data suggest an upward flow direction from the groundwater to the surface water. The average advective flux for flux meter unit SN300 and SN303 was 0.44 ± 0.34 mm/day and 0.76 ± 0.33 mm/day, respectively. The results show that temporal flow variation exists over the study period and the flow at the location of the flux meter was very low. Hydraulic conductivity and piezometric measurements were not part of the flux meter. They were intended to provide supporting evidence for the anticipated variability in the flux measurements. Calculated seepage flux based on hydraulic conductivity and gradient data suggests the average flux was 1.12 ± 0.71 mm/day. Note that these measurements are not reflective of measured fluxes in other locations of the cove receiving contaminated groundwater discharge. The studies presented in this report were not designed to estimate seepage flux at the scale of Red Cove.

The flux meter works well when the water surface is reasonably still. The apparent oscillating flow during the winter months is perplexing. It was suspected that wave actions due to oscillating disturbances in ice-covered water were the cause of the flow oscillation during the winter period. More field studies are needed to determine if the observation made during this study is a common occurrence, and to determine the cause of the observation. At the present time there are no reliable data interpretation methods for handling flow oscillation or wave action. Efforts need to be made in developing techniques that will permit interpreting data that is generated during apparent flow oscillation. Measuring the differential pressure across the sediment-water interface may provide data needed to assist in the interpretation of the flow behavior. The apparent oscillating flow is potentially significant. It would potentially allow a major cyclic fluid exchange between the sediment and the overlaying water affecting the biogeochemical processes in permeable beds (Precht and Huettel 2003).

A sharp peak is essential when the analysis is based on the time of the peak arrival. It is difficult to determine the precise time of the peak arrival when the heat-pulse spreads in a low flow condition. A few modifications could be made to improve the data interpretation. An additional temperature sensor could be installed closer to the heat source. This would reduce the time available for the heat-pulse to spread before it was detected. The length of the heater could be reduced making the heat source more of a point source. The material of the heater could be changed to one with a high thermal conductivity and low heat capacity. The duration of the heat-pulse could be shortened focusing the energy closer to an instantaneous point source. Sufficient heat is needed to be able to easily detect the change in temperature, but the amount of heat injected needs to remain small to minimize the buoyancy effects caused by the heating of the water.

Thermistors were used to measure temperature in this device. They provided excellent resolution and accuracy. However, there was a considerable amount of variability among the thermistors. The time and labor involved in generating individual calibration curves for each thermistor was costly. It would be desirable to find an interchangeable temperature detector that is sufficiently small and have the required sensibility that could be used to replace the thermistors. Thermocouples are interchangeable, but have low sensitivity and did not offer a proper signal-to-noise ratio needed for this application. A suitable replacement has not been identified at this point in time.

8 References

Campbell, G. S., C. Calissendorff, et al. (1991). "Probe for measuring soil specific heat using heat-pulse method." Soil Sci Am J **55**: 291-293.

Chen, X. (2004). "Streambed Hydraulic Conductivity for Rivers in South-Central Nebraska." Journal of the American Water Resources Association (JAWRA) **40**(3): 561-573.

Choi, H.-m., K.-B. Lee, et al. (1991). "Measurement of gas flowrate by using a heat pulse." Flow Measurement and Instrumentation **2**(3): 188.

Ford, R. G., S. Acree, et al. (2009). "Devens 2008 Monitoring Update: Arsenic Fate, Transport and Stability Study, Groundwater, Surface Water, Soil and Sediment Investigation, Fort Devens Superfund Site, Devens, Massachusetts." EPA/600/R-09/064.

Fritz, B. G., D. P. Mendoza, et al. (2009). "Development of an electronic seepage chamber for extended use in a river." Ground water **47**(1): 136-140.

Genereux, D. P., S. Leahy, et al. (2008). "Spatial and temporal variability of streambed hydraulic conductivity in West Bear Creek, North Carolina, USA." Journal of Hydrology **358**: 332-353.

Hubbert, M. K. (1956). "Darcy's law and the field equations of the flow of underground fluids." Am. Inst. Min. Met. Petl. Eng. Trans. **207**: 222-239.

Kalbus, E., F. Reinstorf, et al. (2006). "Measuring methods for groundwater,surface water and their interactions: a review" Hydrol. Earth Syst. Sci. Discuss. **3**: 1809-1850.

Lee, D. R. (1977). "Device for Measuring Seepage Flux in Lakes and Estuaries." Limnology and Oceanography **22**(1): 140-147.

Lien, B. K. (2006). "Development and Demonstration of a Bi-directional Advective Flux Meter for Sediment-Water Interface." EPA/600/R-06/122.

Marion, W. and S. Wilcox. (1992). "Solar Radiation Data Manual for Flat-Plate and Concentrating Collectors." from http://rredc.nrel.gov/solar/pubs/redbook/.

Martin, L. H. and K. C. Lang (1933). "The Thermal Conductivity of Water." Proc. Phys. Soc. **45**(4): 523-529.

Mortensen, A. P., J. W. Hopmans, et al. (2006). "Multi-functional heat pulse probe measurements of coupled vadose zone flow and transport." Advances in Water Resources **29**(2): 250.

Murdoch, L. C. and S. E. Kelly (2003). "Factors affecting the performance of conventional seepage meters." Water Resources Research **39**(6): SWC 2-1 : 2-10.

Mwashote, B. M., W. C. Burnett, et al. (2010). "Calibration and use of continuous heat-type automated seepage meters for submarine groundwater discharge measurements." Estuarine, Coastal and Shelf Science **87**: 1-10.

Ochsner, T. E., R. Horton, et al. (2005). "Evaluation of the Heat Pulse Ratio Method for Measuring Soil Water Flux." Soil Sci. Soc. Am. J. **69**: 757-765.

Paillet, F., R. Crowder, et al. (1996). "High-Resolution Flowmeter Logging Applications with the Heat-Pulse Flowmeter." Journal of Environmental and Engineering Geophysics **1**(1): 1-11.

Paulsen, R. J., C. F. Smith, et al. (2001). "Development and evaluation of an ultrasonic ground water seepage meter." Ground Water **39**(6): 904-911.

Precht, E. and M. Huettel (2003). "Advective pore-water exchange driven by surface gravity waves and its ecological implications." Limnol. Oceanogr. **48**(4): 1674-1684.

Ren, T., G. J. Kluitenberg, et al. (2000). "Determining soil water flux and pore water velocity by a heat pulse technique." Soil Sci Am J **64**: 552-560.

Rosenberry, D. O. (2008). "A seepage meter designed for use in flowing water." Journal of Hydrology **359**(1-2): 118-130.

Rosenberry, D. O. and R. H. Morin (2004). "Use of an electromagnetic seepage meter to investigate temporal variability in lake seepage." Ground Water **42**(1): 68-77.

Runyan, M. C. and W. C. Jones (2008). "Thermal conductivity of thermally-isolating polymeric and composite structural support materials between 0.3 and 4 K." Cryogenics **48**(9-10): 448-454.

Schincariol, R. A. and J. D. McNeil (2002). "Errors with small volume elastic seepage meter bags." Ground Water **40**(6): 649-651.

Shinn, E. A., C. D. Reich, et al. (2002). "Seepage Meters and Bernoulli's Revenge." Estuaries and Coasts **25**(1): 126-132.

Sholkovitz, E., C. Herbold, et al. (2003). "An automated dye-dilution based seepage meter for the time-series measurement of submarine groundwater discharge." Limnology and Oceanography-Methods **1**: 16-28.

Smith, A. J., D. E. Herne, et al. (2009). "Wave effects on submarine groundwater seepage measurement." Advances in Water Resources **32**(6): 820-833.

Steinhart, J. S. and S. R. Hart (1968). "Calibration curves for thermistors." Deep Sea Res. **15**: 497-503.

Swarzenski, P. W. and J. A. Izbicki (2009). "Coastal groundwater dynamics off Santa Barbara, California: Combining geochemical tracers, electromagnetic seepmeters, and electrical resistivity." Estuarine Coastal and Shelf Science **83**(1): 77-89.

Taniguchi, M., W. C. Burnett, et al. (2006). "Submarine groundwater discharge measured by seepage meters in Sicilian coastal waters." Continental Shelf Research **26**(7): 835-842.

Taniguchi, M., W. C. Burnett, et al. (2003). "Spatial and temporal distributions of submarine groundwater discharge rates obtained from various types of seepage meters at a site in the Northeastern Gulf of Mexico." Biogeochemistry **66**(1-2): 35-53.

Taniguchi, M. and Y. Fukuo (1993). "Continuous measurements of ground-water seepage using an automatic seepage meter." Ground water **31**(4): 675-679.

Taniguchi, M., T. Ishitobi, et al. (2007). "Evaluating ground water-sea water interactions via resistivity and seepage meters." Ground Water **45**(6): 729-735.

Taniguchi, M. and H. Iwakawa (2001). "Measurements of submarine groundwater discharge rates by a continuous heat—type automated seepage meter in Osaka Bay, Japan." Groundwater Hydrology **42**(271-277).

Wang, Q., T. E. Ochsner, et al. (2002). "Mathematical analysis of heat pulse signals for soil water flux determination." Water Resour. Res. **38**(6): 1091.

Zuber, A. (1970). "Method for determine leakage velocities through the bottom of reservoirs." Isotope Hydrology: 761-771.

Appendix A Visual Basic Code

frmDataAcquisition

```
‘Option Explicit ' last revision 5/19/08 by Carl Enfield
' Purpose To measure temperatures, differential temperatures and differential pressures
' Scans a range of frequencies Input Channels, stores the data in an array
  ' and output the data to ASCII files.

'send digital output to switch heater ON/OFF
'sends digital signal to control the power supply on the flux meter
'the temperature measurements are made with thermistors.  The thermistors
'  are supplied a constant current of 100 uA.  For channels 1 and 7 the impedance
'  is changed by supplying the voltage to the positive input of an operational
'  amplifier which serves as a buffer amplifier. The output from this amplifier from channel 1 is
'  used as a reference For the other temperature measurements
'  the voltage signals are input to the negative input of instrumentation amplifiers

'channel 1: reference junction temperature a 1.82K ohm 1% resistor is in series with this thermistor
'channel 2 - 3: flow temperatures beginning from cable end
'channel 4: heater temperature CA adhesive is initially used to attach the thermistor to the heater.
   'Epoxy (JB weld) is then applied for added strength.
'channel 5-6: flow temperature
'channel 7: temperature outside the dome
'channel 8: Differential pressure transducer
'channels 2 through 6 are subtracted from channel 1 and have a gain of ten all other channels have
   'a gain of one.

Const BoardNum1% = 0        ' Board number Site 1
Const BoardNum2% = 1       ' Board number Site 2
'Dim BoardNum2%

Const FirstPoint& = 0      ' set first element in buffer to transfer to array
Const Warmup = 12        ' Time in minutes after activating circuit board power before measurements are initiated
Const GateInterval& = 100   ' Number of milliseconds used to count pulses and determine frequency
Const GateInterval1& = 1000 ' Number of milliseconds used to count pulses and determine frequency used on ref
temperature only
'Dim BoardNum%
Dim ActualCounts%(10)
Public baseline     'time used to collect temperature data prior to injecting heat
Dim CBCount%(10)
Dim ChipNum%
Dim counter%, rounds, RateCount
Dim DataValue%            ' digital value to write to Auxport
Dim Dir1 As String * 32     ' Directory for data storage for Site 1
Dim Dir2 As String * 32     ' Directory for data storage for Site 2
'Dim DirA As String * 32
Dim DomeDia               ' diameter of dome in centimeters
Dim fname As String * 50    ' file name for raw data with ext .dat
Dim fnameS As String * 50 ' file name for supplemental raw data with ext .dat
Dim fnameP As String * 50   ' file name for processed output data with extension .csv
Dim fname1 As String * 50   ' file name for processed relative change in temperature
Dim fname2 As String * 50   ' file name for data converted to temperature in deg C
Dim FOutDivider&
Dim Freq&(10)
Public heater      'time in seconds heater is turned on
Dim i, k, j, N As Integer                  ' counter
Dim MemHandle&             ' define a variable to contain the handle for
                        ' memory allocated by Windows through cbWinBufAlloc%()
Public pause       ' time in seconds before next cycle of data collection allos heat to dissipate
Public CycleTime
Dim SampleTime, startpause
Dim site1 As String * 64 ' a descriptive string describing the site 1
Dim site2 As String * 64 ' a descriptive string describing the site 2

Dim Source&
Public tail        ' time in seconds experimental data is collected
Dim CallStop, mytime
Dim finish, timepast, t, timezero
Dim sum(8) As Double
Dim ULStat As Long               'universal library statistic return
Dim SerialNo
Dim SigSource(10)
Dim textline
Dim Pass%
Dim PassWarm As Long                    ' number of passes through pause cycle
Dim Resistor(), c10(), c20(), c30(), cv0(), c11(), c21(), _
      c31(), cv1(), c13(), c23(), c33(), cv3(), c14(), c24(), _
      c34(), cv4(), SerNo(), Tslope(), Tint(), TSn1(), ch1a(), _
      ch1b(), ch1c(), TSn2(), ch2a(), ch2b(), ch2c(), TSn3(), _
      ch3a(), ch3b(), ch3c(), TSn4(), ch4a(), Ch4b(), Ch4c(), _
      TSn5(), ch5a(), Ch5b(), Ch5c(), TSn6(), Ch6a(), Ch6b(), _
      Ch6c(), TSn7(), Ch7a(), Ch7b(), Ch7c()


Private Sub CmdClose_Click()
   Call PowerOffHeatOff(BoardNum1%)
   If Check2 Then
      'MsgBox ("Box Checked")
      Call PowerOffHeatOff(BoardNum2%)
   End If
   Unload Me
End Sub
Private Sub PowerOffHeatOff(BoardNum%)
'*******
' Turn off heater turn off power supply
'*******

  ULStat = cbDBitOut(BoardNum%, AUXPORT, 0, 0) ' turn off power supply flux meter
  If ULStat <> 0 Then Stop
  ULStat = cbDBitOut(BoardNum%, AUXPORT, 1, 0) ' turn off heater
  If ULStat <> 0 Then Stop
  ULStat = cbDBitOut(BoardNum%, AUXPORT, 2, 0) ' turn off power auxiliary sensors
  If ULStat <> 0 Then Stop
'   MsgBox (BoardNum%)
```

```
End Sub
Private Sub PowerOnHeatOff(BoardNum%)

'*******
' Turn off heater turn on power supply
'*******

   ULStat = cbDBitOut(BoardNum%, AUXPORT, 0, 1) ' turn on power supply
   If ULStat <> 0 Then Stop
   ULStat = cbDBitOut(BoardNum%, AUXPORT, 1, 0) ' turn off heater
   If ULStat <> 0 Then Stop
   ULStat = cbDBitOut(BoardNum%, AUXPORT, 2, 1) ' turn on power supply auxillary sensors
   If ULStat <> 0 Then Stop

'   MsgBox (BoardNum%)
End Sub

Private Sub cmdStart_Click()
'*********
' Collect Site Specific Screen Data
'*********
site1 = CStr(txtSite1.Text)
site2 = CStr(txtSite2.Text)
SerialNo = CStr(txtSerNo.Text)
SerialNo2 = CStr(txtSerNo2.Text)
baseline = CDbl(txtBaseline.Text)       ' time to collect baseline data (min)
heater = CDbl(txtHeater.Text)           ' time for heater to be turned on (sec)
tail = CDbl(txtTail.Text)               ' time to collect data after heater is turned off (min)
DomeDia = CStr(txtDDia.Text)
'MsgBox ("Read screen data")
Dir1 = "C:\Temp\Site1"
Dir2 = "C:\Temp\Site2"
'fname = Format$(Now, "yyyy") + ".txt"
'myFile = Dir("C:\\Temp\*.*")
'MsgBox ("looked for file in C:\\Temp")
'If (myFile = "") Then
'    ChDir (Dir1)
'    Open fname For Output As #1
'    Write #1, Format("test")
'    Close #1
'    MkDir "C:\\Temp\"
'    ChDir ("C:\\Temp\")
'    Open fname For Output As #1
'    Write #1, Format("test")
'    Close #1
'End If
'myFile = Dir("C:\\Temp\Site1\*.*")
'MsgBox ("looked for file in C:\\Temp\Site1")
'If (myFile = "") Then
'    MkDir "C:\\Temp\Site1"
'    ChDir (Dir1)
'    Open fname For Output As #1
'    Write #1, Format("test")
'    Close #1
'End If
'myFile = Dir("C:\\Temp\Site2\*.*")
'MsgBox ("looked for file in C:\\Temp\Site2")
'If (myFile = "") Then
'    MkDir "C:\\Temp\Site2"
'    ChDir (Dir2)
'    Open fname For Output As #1
'    Write #1, Format("test")
'    Close #1
'End If
Pass% = 1
        lblTime.Caption = Now    'display time
    Label12.Caption = Pass%
Dir1 = "C:\Temp\Site1"
Dir2 = "C:\Temp\Site2"
    ChDir (Dir1)

    fnameP = Format$(Now, "mmddhh") + "p.CSV"
    Open fnameP For Output As #10
        Write #10, Format("DateTime"), Format("baseline"), Format("heater"), Format("Tail"), Format("Serial No."), _
            Format("Dome Dia"), Format("Site")

        Write #10, Format(Now), Format(baseline, "#0.00"), _
            Format(heater, "#0.00"), Format(tail, "#0.00"), Format(SerialNo, "#0"), _
            Format(DomeDia, "#0.00"), Format(site1)
        Write #10, Format("DateTime"), Format("Peak T 2"), Format("Peak T 3"), Format("Peak T 5"), Format("Peak T 
6"), _
            Format("Moment 2"), Format("Moment 3"), Format("Moment 5"), Format("Moment 6"), Format("Ref 
Temp"), _
            Format("Sup Temp"), Format("Diff Pres"), Format("Ch 9"), Format("Ch 10"), _
            Format("PeakRefTemp"), Format("PeakTemp2"), Format("PeakTemp3"), Format("PeakTemp4"), 
Format("PeakTemp5"), _
            Format("PeakTemp6")


    Close #10
If Check2 Then
        ChDir (Dir2)
        fnameP = Format$(Now, "mmddhh") + "p.CSV"
        Open fnameP For Output As #10
            Write #10, Format("DateTime"), Format("baseline"), Format("heater"), Format("Tail"), Format("Serial No."), 
_
                Format("Dome Dia"), Format("Site")

            Write #10, Format(Now), Format(baseline, "#0.00"), _
                Format(heater, "#0.00"), Format(tail, "#0.00"), Format(SerialNo2, "#0"), _
                Format(DomeDia, "#0.00"), Format(site2)
            Write #10, Format("DateTime"), Format("Peak T 2"), Format("Peak T 3"), Format("Peak T 5"), Format("Peak 
T 6"), _
                Format("Moment 2"), Format("Moment 3"), Format("Moment 5"), Format("Moment 6"), Format("Ref 
Temp"), _
```

```
            Format("Sup Temp"), Format("Diff Pres"), Format("Ch 9"), Format("Ch 10"), _
          Format("PeakRefTemp"), Format("PeakTemp2"), Format("PeakTemp3"), Format("PeakTemp4"),
Format("PeakTemp5"), _
          Format("PeakTemp6")

       Close #10
End If
Do While CallStop <> -1         'main loop
   Label12.Caption = Pass%

'*********
' Re-read timing values each pass so system can be modified during run
'*********

baseline = CDbl(txtBaseline.Text)       ' time to collect baseline data (min)
heater = CDbl(txtHeater.Text)          ' time for heater to be turned on (sec)
CycleTime = CDbl(txtCycleTime.Text)     'total time for a cycle in hours
tail = CDbl(txtTail.Text)              ' time to collect data after heater is turned off (min)
SampleTime = baseline * 60 + heater + tail * 60 'total sampling time in seconds
pause = CycleTime * 60 - (SampleTime / 60) - Warmup ' time in minutes
   If Check2 Then pause = CycleTime * 60 - (2 * (SampleTime / 60)) - 2 * Warmup
   If pause < 0 Then MsgBox ("Insufient Pause time. Pause in minutes = " & pause)
'MsgBox ("pause= " & pause)
'   lblSiteNum.Caption = 1
   Refresh
   Call Warm_Up(BoardNum1%)
   Call DataAcquisition(1, Dir1, site1, BoardNum1%, Pass%, fnameP, SerialNo, baseline, heater, CycleTime, tail,
SampleTime, DomeDia) ' Collect Data Set for Site 1
'   MsgBox (tail)
   Call PlotData(fname, fnameP, fnameS, tail, 1, Dir1, Pass%, 1) ' Plot Site 1 Data
   If Check2 Then
'       LblSiteNo.Caption = 2
      Refresh
      Call Warm_Up(BoardNum2%)
      Call DataAcquisition(2, Dir2, site2, BoardNum2%, Pass%, fnameP, SerialNo2, baseline, heater, CycleTime, tail,
SampleTime, DomeDia) ' Collect Data Set for Site 1
      Call PlotData(fname, fnameP, fnameS, tail, 2, Dir2, Pass%, 2) ' Plot Site 2 Data

   End If
   Pass% = Pass% + 1
   If Pass% > 100 Then
      Pass% = 1
      ChDir (Dir1)
      fnameP = Format$(Now, "mmddhh") + "p.CSV"
      Open fnameP For Output As #10
         Write #10, Format("DateTime"), Format("baseline"), Format("heater"), Format("Tail"), Format("Serial No."),
_
            Format("Dome Dia"), Format("Site")

         Write #10, Format(Now), Format(baseline, "#0.00"), _
            Format(heater, "#0.00"), Format(tail, "#0.00"), Format(SerialNo, "#0"), _
            Format(DomeDia, "#0.00"), Format(site)
      Write #10, Format("DateTime"), Format("Peak T 2"), Format("Peak T 3"), Format("Peak T 5"), Format("Peak
T 6"), _
        Format("Moment 2"), Format("Moment 3"), Format("Moment 5"), Format("Moment 6"), Format("Ref
Temp"), _
        Format("Sup Temp"), Format("Diff Pres"), Format("Ch 9"), Format("Ch 10"), _
        Format("PeakRefTemp"), Format("PeakTemp2"), Format("PeakTemp3"), Format("PeakTemp4"), _
        Format("PeakTemp5"), Format("PeakTemp6")

    Close #10
    If Check2 Then
      ChDir (Dir2)
      fnameP = Format$(Now, "mmddhh") + "p.CSV"
      Open fnameP For Output As #10
        Write #10, Format("DateTime"), Format("baseline"), Format("heater"), Format("Tail"), Format("Serial
No."), Format("Dome Dia"), Format("Site")

        Write #10, Format(Now), Format(baseline, "#0.00"), _
          Format(heater, "#0.00"), Format(tail, "#0.00"), Format(SerialNo2, "#0"), _
          Format(DomeDia, "#0.00"), Format(site2)

        Write #10, Format("DateTime"), Format("Peak T 2"), Format("Peak T 3"), Format("Peak T 5"),
Format("Peak T 6"), _
          Format("Moment 2"), Format("Moment 3"), Format("Moment 5"), Format("Moment 6"), Format("Ref
Temp"), _
          Format("Sup Temp"), Format("Diff Pres"), Format("Ch 9"), Format("Ch 10"), _
          Format("PeakRefTemp"), Format("PeakTemp2"), Format("PeakTemp3"), Format("PeakTemp4"), _
          Format("PeakTemp5"), Format("PeakTemp6")

      Close #10
    End If
  End If
  txtCycleTime.BackColor = &HFF00&
  Refresh
  Call PowerOffHeatOff(BoardNum1%)
  If Check2 Then
    'MsgBox ("power off heat off")
    Call PowerOffHeatOff(BoardNum2%)
  End If
  t = 0
  timezero = Timer
'   MsgBox ("pause =" & pause)
  Do While t < pause * 60     'Loop 14

    If Timer < timezero Then   'if cross midnight
      t = 86400 - timezero + Timer
    Else
      t = Timer - timezero
    End If
    lblNxtCycTime.Caption = Format(((pause - t / 60) + Warmup), "#0.0")
    lblTime.Caption = Now    'display time
    DoEvents    'Yield to other processes
    If CallStop = -1 Then GoTo 44
  Loop    'Loop 14
```

```
   '    Pass% = 1
   txtCycleTime.BackColor = &HFFFFFF
Loop  'main



44 txtBaseline.BackColor = &HFFFFFF
  txtHeater.BackColor = &HFFFFFF
  txtTail.BackColor = &HFFFFFF
  txtCycleTime.BackColor = &HFFFFFF
  cmdStart.Enabled = -1
  CallStop = 0
  GoTo 45


'Loop
45 End Sub
Private Sub Warm_Up(BoardNum%)
   Call PowerOnHeatOff(BoardNum%)
'    timezero_main = Timer
   '*********
   ' Warm up required before data valid.  20 min determined adequate to obtain reasonably stable values
   '*********
   txtBaseline.BackColor = &HFFFF&
   Screen.MousePointer = 11    'hourglass pointer
   Refresh
   t = 0
   timezero = Timer
   PassWarm = 0
   Do While t < Warmup * 60
      PassWarm = PassWarm + 1
      '******
      'calculate accmulated time
      '******
      DoEvents
      If Timer < timezero Then   'if cross midnight
         t = 86400 - timezero + Timer
      Else
         t = Timer - timezero
      End If
      If PassWarm = 64000 Then  ' reduces the number of times display is refreshed
         lblNxtCycTime.Caption = Format((Warmup - (t / 60)), "#0.0") ' warm up time remaining
         Refresh
         PassWarm = 0
      End If
      DoEvents
   Loop

End Sub
Private Sub DataAcquisition(site#, DirA, siteA, BoardNum%, Pass%, fnameP, SerialNo, baseline, heater, CycleTime,
tail, SampleTime, DomeDia)
'Dim DirA As String * 32
ChDir (DirA)
```

```
'*********
' Initialize signal sources
' There are five inputs on each chip so use chip 2 for the second 5 signal sources
'*********
SigSource(1) = CTRINPUT1       ' Input one chip 1
SigSource(2) = CTRINPUT2       ' Input two chip 1
SigSource(3) = CTRINPUT3       ' Input three chip 1
SigSource(4) = CTRINPUT4       ' Input four chip 1
SigSource(5) = CTRINPUT5       ' Input Five chip 1
SigSource(6) = CTRINPUT6       ' Input six chip 2
SigSource(7) = CTRINPUT7       ' Input 7 chip 2
SigSource(8) = CTRINPUT8       ' Input 8 chip 2
SigSource(9) = CTRINPUT9       ' Input 9 chip 2
SigSource(10) = CTRINPUT10      ' Input 10 chip 2

'*******
' Collect Screen Input Data
'*******
MemHandle& = cbWinBufAlloc(100)
If MemHandle& = 0 Then Stop

'******
'write baseline, heat-pulse duration, SerialNo, and Site description raw data file #1
'******
Date = Now
Time = Now
fnameS = Format$(Date, "mmdd") + Format$(Time, "hh") + "S" + ".dat"
fname = Format$(Date, "mmdd") + Format$(Time, "hh") + ".dat"
'fname1 = Format$(Date, "mmdd") + Format$(Time, "hh") + "1.dat"
'fname2 = Format$(Date, "mmdd") + Format$(Time, "hh") + "2.dat"
Open fname For Output As #1
  Write #1, Format(Now), Format(baseline, "#0.00"), _
    Format(heater, "#0.00"), Format(tail, "#0.00"), Format(SerialNo, "#0"), _
    Format(DomeDia, "#0.00"), Format(siteA)
Close #1
  ULStat = cbWinBufFree(MemHandle&)      ' Free up memory for use by
  If ULStat <> 0 Then Stop               ' other programs

'*******
' Initialize settings required to scan thermistors
' Chanels 1 through 6 are scanned.
'*******

MemHandle& = cbWinBufAlloc(100)
If MemHandle& = 0 Then Stop

  Screen.MousePointer = 0     'Default Pointer
  Refresh
  Date = Now
  Time = Now
  lblSiteNum.Caption = site#
  Refresh
```

```
    timezero = Timer
    t = 0
    k = 0
    Do While t < SampleTime - 1     'Loop 11
        'calculate accmulated time
        If Timer < timezero Then   'if cross midnight
            t = 86400 - timezero + Timer
        Else
            t = Timer - timezero
        End If
        '*******
        'control when to activate digital output for remote power supply and heater
        '*******
        If t <= (baseline * 60) Then DataValue% = 0: txtBaseline.BackColor = &HFF00& '
        If t > (baseline * 60) Then txtBaseline.BackColor = &HFFFFFF
        If (t > (baseline * 60) And t <= (baseline * 60 + heater)) Then DataValue% = 1: txtBaseline.BackColor = _
            &HFFFFFF: txtHeater.BackColor = &HFF00&
        If t > (baseline * 60 + heater) Then DataValue% = 0: txtHeater.BackColor = &HFFFFFF: txtTail.BackColor =
&HFF00&
        '*******
        'digital out
        '*******
        ULStat = cbDBitOut(BoardNum%, AUXPORT, 1, DataValue%)
        If ULStat <> 0 Then Stop
        DataSet (BoardNum%)    ' Collect a set of data
        Open fname For Append As #1
            'write data in frequency
            Write #1, Format(Now), Format(t, "#0.00000"), _
                Format(Freq&(1), "#0"), Format(Freq&(2), "#0"), _
                Format(Freq&(3), "#0"), Format(Freq&(4), "#0"), _
                Format(Freq&(5), "#0"), Format(Freq&(6), "#0")
        Close #1
        '*******
        'display data in frequency
        '*******
        For i = 1 To 6
            lblADData(i).Caption = Format(Freq&(i), "#0")
        Next i
        lblTime.Caption = Now    'display time
        lblRemainTime.Caption = Format((SampleTime - t) / 60, "#0.0")
        Refresh

        DoEvents
        If CallStop = -1 Then GoTo 44
    Loop     'Loop 11

            '48 Close #1
    DataSet2 (BoardNum%)       'collect suplimental data
    For i = 6 To 10
         lblADData(i).Caption = Format(Freq&(i), "#0")
    Next i
    Open fnameS For Output As #9
        Write #9, Format(Now), _
```

```
         Format(Freq&(7), "#0"), Format(Freq&(8), "#0"), Format(Freq&(9), "#0"), _
         Format(Freq&(10), "#0"), Format(SerialNo, "#0"), Format(site)
   Close #9

   txtTail.BackColor = &HFFFFFF

44 txtBaseline.BackColor = &HFFFFFF
  txtHeater.BackColor = &HFFFFFF
  txtTail.BackColor = &HFFFFFF
  txtCycleTime.BackColor = &HFFFFFF
  cmdStart.Enabled = -1
  CallStop = 0
  GoTo 45
45 End Sub
Private Sub CmdStop_Click()
   CallStop = -1
   Call PowerOffHeatOff(BoardNum1%)
   If Check2 Then Call PowerOffHeatOff(BoardNum2%)
Unload Me

End Sub

Private Sub cmdStopConvert_Click()
   'MsgBox ("cmdStop")
   Call PowerOffHeatOff(BoardNum1%)
   'MsgBox ("Board 1 off")
   If Check2 Then
      Call PowerOffHeatOff(BoardNum2%)
   'MsgBox ("Board 2 off")
   End If
   ULStat = cbWinBufFree(MemHandle&)      ' Free up memory for use by
                                  ' other programs
'   If ULStat <> 0 Then Stop

   End
   Unload Me

End Sub


Private Sub Form_Load()

  ' declare revision level of Universal Library

  ULStat = cbDeclareRevision(CURRENTREVNUM)

  ' Initiate error handling
  '  activating error handling will trap errors like
  '  bad channel numbers and non-configured conditions.

  '  Parameters:
  '    PRINTALL    :all warnings and errors encountered will be printed
  '    DONTSTOP    :if an error is encountered, the program will not stop,
  '              errors must be handled locally

  ULStat = cbErrHandling(PRINTALL, DONTSTOP)
  If ULStat <> 0 Then Stop

  ' If cbErrHandling% is set for STOPALL or STOPFATAL during the program
  ' design stage, Visual Basic will be unloaded when an error is encountered.
  ' We suggest trapping errors locally until the program is ready for compiling
  ' to avoid losing unsaved data during program design.  This can be done by
  ' setting cbErrHandling options as above and checking the value of ULStat%
  ' after a call to the library. If it is not equal to 0, an error has occurred.


End Sub

Private Sub Initialize_Click()
   Call PowerOffHeatOff(BoardNum1%)
   If Check2 Then Call PowerOffHeatOff(BoardNum2%)
End Sub
Private Sub Init1UL(BoardNum%)
  ' declare revision level of Universal Library

  ULStat& = cbDeclareRevision(CURRENTREVNUM)

  ' Initiate error handling
  '  activating error handling will trap errors like
  '  bad channel numbers and non-configured conditions.
  '  Parameters:
  '    PRINTALL    :all warnings and errors encountered will be printed
  '    DONTSTOP    :if an error is encountered, the program will not stop,
  '              errors must be handled locally

  ULStat& = cbErrHandling(DONTPRINT, DONTSTOP)
  If ULStat& <> 0 Then Stop

  ' If cbErrHandling& is set for STOPALL or STOPFATAL during the program
  ' design stage, Visual Basic will be unloaded when an error is encountered.
  ' We suggest trapping errors locally until the program is ready for compiling
  ' to avoid losing unsaved data during program design.  This can be done by
  ' setting cbErrHandling options as above and checking the value of ULStat%
  ' after a call to the library. If it is not equal to 0, an error has occurred.


  ' Initialize the board level features
  '  Parameters:
  '    BoardNum&    :the number used by CBCONFIG to describe this board
  '    ChipNum&    :chip to be initialized (1 for CTR5, 1 or 2 for CTR10)
  '    FOutDivider&:the F-Out divider (0-15)
  '    Source&     :the signal source for F-Out
  '    Compare1&   :status of comparator 1
  '    Compare2&   :status of comparator 2
  '    TimeOfDay&  :time of day mode control
  ChipNum% = 1
  FOutDivider& = 1         ' sets up OSC OUT for 10kHz signal which can
  Source& = FREQ1          ' be used as frequency source for this example
  Compare1& = Disabled
  Compare2& = Disabled
  TimeOfDay& = Disabled

  ULStat& = cbC9513Init(BoardNum%, ChipNum%, FOutDivider&, Source&, Compare1&, Compare2&,
TimeOfDay&)
  If ULStat& <> 0 Then Stop
End Sub

Private Sub Init2UL(BoardNum%)

  ' Initialize the board level features
  '  Parameters:
  '    BoardNum&    :the number used by CBCONFIG to describe this board
  '    ChipNum&    :chip to be initialized (1 for CTR5, 1 or 2 for CTR10)
  '    FOutDivider&:the F-Out divider (0-15)
  '    Source&     :the signal source for F-Out
  '    Compare1&   :status of comparator 1
  '    Compare2&   :status of comparator 2
  '    TimeOfDay&  :time of day mode control
  ChipNum% = 2
'   FOutDivider& = 1         ' sets up OSC OUT for 10kHz signal which can
'   Source& = FREQ1          ' be used as frequency source for this example
'   Compare1& = Disabled
'   Compare2& = Disabled
'   TimeOfDay& = Disabled

  ULStat& = cbC9513Init(BoardNum%, ChipNum%, FOutDivider&, Source&, Compare1&, Compare2&,
TimeOfDay&)
  If ULStat& <> 0 Then Stop
End Sub
Private Sub DataSet(BoardNum%)
  Init1UL (BoardNum%)
  '**********
  ' First channel, the reference temperature, is counted longer to improve resolution see GateInterval1& versus
GateInterval&
  '   Resolution is directly dependent on GateInterval
  '**********
  ULStat& = cbCFreqIn(BoardNum%, SigSource(1), GateInterval1&, CBCount%(1), Freq&(1))
     If ULStat& <> 0 Then
        ULStat& = cbCFreqIn(BoardNum%, SigSource(1), GateInterval1&, CBCount%(1), Freq&(1))
        If ULStat& <> 0 Then MsgBox ("ULStat& = " & ULStat&)
     End If

     If CBCount%(1) < 0 Then
        ActualCounts%(1) = 65535 + CBCount%(1)
     Else
        ActualCounts%(1) = CBCount%(1)
     End If
     If ActualCounts%(1) <= 0 Then MsgBox ("ActualCount(1) = " & ActualCounts%(1))
   For i = 2 To 5
```

```
    ULStat& = cbCFreqIn(BoardNum%, SigSource(i), GateInterval&, CBCount%(i), Freq&(i))
    If ULStat& <> 0 Then
      ULStat& = cbCFreqIn(BoardNum%, SigSource(i), GateInterval&, CBCount%(i), Freq&(i))
      If ULStat& <> 0 Then MsgBox ("ULStat& = " & ULStat&)
    End If

    If CBCount%(i) < 0 Then
        ActualCounts%(i) = 65535 + CBCount%(i)
'      MsgBox (CBCount%(i))
    Else
      ActualCounts%(i) = CBCount%(i)
    End If
    If ActualCounts%(i) <= 0 Then MsgBox ("ActualCount(i) = " & ActualCounts%(i))

999 Next i
  Init2UL (BoardNum%)
  For i = 6 To 6
    ULStat& = cbCFreqIn(BoardNum%, SigSource(i), GateInterval&, CBCount%(i), Freq&(i))
    If ULStat& <> 0 Then
      ULStat& = cbCFreqIn(BoardNum%, SigSource(i), GateInterval&, CBCount%(i), Freq&(i))
      If ULStat& <> 0 Then MsgBox ("ULStat& = " & ULStat&)
    End If

    If CBCount%(i) < 0 Then
      ActualCounts%(i) = 65535 + CBCount%(i)
'      MsgBox (CBCount%(i))
    Else
      ActualCounts%(i) = CBCount%(i)
    End If
  Next i

End Sub

Private Sub DataSet2(BoardNum%)
  Init2UL (BoardNum%)
  For i = 7 To 10
    ULStat& = cbCFreqIn(BoardNum%, SigSource(i), GateInterval&, CBCount%(i), Freq&(i)) ' short gate interval

    If ULStat& <> 0 Then
      ULStat& = cbCFreqIn(BoardNum%, SigSource(i), GateInterval&, CBCount%(i), Freq&(i))
      If ULStat& <> 0 Then MsgBox ("ULStat& = " & ULStat&)
    End If

    If CBCount%(i) < 0 Then
    MsgBox (CBCount%(i))
      ActualCounts%(i) = 65535 + CBCount%(i)
    Else
      ActualCounts%(i) = CBCount%(i)
    End If
  Next i

End Sub
```

```
Private Sub PlotData(fname, fnameP, fnameS, tail, SiteNo, DirA, Pass%, SiteNum)
'   Screen.MousePointer = 11
  Dim textline
  Dim time0() As String
  Dim time1 As String 'used to store time file created needed to determine age of file and be compatable with
older data

   Dim chb()   'dynamic array to hold analog byte data stored on .dat file
   Dim ch() As Double   'dynamic array to hold double precision data from .dat file
   Dim tb()           ' dynamic array to hold elapsed time data in byte format
   Dim t()  As Double  'dynamic array to hold elapsed time data
  Dim ct As Integer    '        , pt_odd, pt_half As Integer
  Dim tmax(6) As Double
  Dim tempmax(6) As Double
  Dim bline
  Dim heater
  Dim SerNumber
'   Dim tail
  Dim N, j, i, l, m, g, k, z, ii, jj, kk As Integer

  Dim CalResistor
  Dim Resistor()
  Dim DomeDia
  Dim site As String * 64
   Dim Q2, Q3, Q5, Q6
   Dim V2, V3, V5, V6
   Dim dat_type As Boolean
  Dim sumx(6) As Double
  Dim sumy(6) As Double
  Dim sumxx2(6) As Double
  Dim sumxy(6) As Double
  Dim sumxx(6) As Double
'   Dim sumy1(6) As Double
  Dim avgx(6) As Double
  Dim avgy(6) As Double
  Dim slope(6) As Double
  Dim sInt(6) As Double
  Dim avgx1(6) As Double
  Dim avgy1(6) As Double
   Dim xxx
LblSiteNo.Caption = SiteNum
'******
' Import calibration data
'******

ChDir ("C:\Program Files\FluxMeter\")

'******
' determine the number of calibration curves in the data table
'******
Open "Calibration_Data(2).csv" For Input As #6
   l = 0
   Do While Not EOF(6)
```

```
      Line Input #6, textline
      l = l + 1    'l is the number of calibration curves
   Loop
Close #6
      'MsgBox (l)
'********
'Allocate space to hold arrays place calibration data into arrays
'********

ReDim Resistor(l), c10(l), c20(l), c30(l), cv0(l), c11(l), c21(l), _
   c31(l), cv1(l), c13(l), c23(l), c33(l), cv3(l), c14(l), c24(l), _
   c34(l), cv4(l), SerNo(l), Tslope(l), Tint(l), TSn1(l), ch1a(l), _
   ch1b(l), ch1c(l), TSn2(l), ch2a(l), ch2b(l), ch2c(l), TSn3(l), _
   ch3a(l), ch3b(l), ch3c(l), TSn4(l), ch4a(l), Ch4b(l), Ch4c(l), _
   TSn5(l), ch5a(l), Ch5b(l), Ch5c(l), TSn6(l), Ch6a(l), Ch6b(l), _
   Ch6c(l), TSn7(l), Ch7a(l), Ch7b(l), Ch7c(l)
'*******
' Import calibration data
'*******
Open "Calibration_Data(2).csv" For Input As #7
  l = 1
  Do While Not EOF(7)
      '   MsgBox (l)
      Input #7, Resistor(l), c10(l), c20(l), c30(l), cv0(l), c11(l), c21(l), _
         c31(l), cv1(l), c13(l), c23(l), c33(l), cv3(l), c14(l), c24(l), _
         c34(l), cv4(l), SerNo(l), Tslope(l), Tint(l), TSn1(l), ch1a(l), _
         ch1b(l), ch1c(l), TSn2(l), ch2a(l), ch2b(l), ch2c(l), TSn3(l), _
         ch3a(l), ch3b(l), ch3c(l), TSn4(l), ch4a(l), Ch4b(l), Ch4c(l), _
         TSn5(l), ch5a(l), Ch5b(l), Ch5c(l), TSn6(l), Ch6a(l), Ch6b(l), _
         Ch6c(l), TSn7(l), Ch7a(l), Ch7b(l), Ch7c(l)
      l = l + 1
   Loop
      '   MsgBox (l)
Close #7
LblSiteNo.Caption = SiteNo
lblSamRnd.Caption = Pass%
'*********
'Determine the number of records in the data file
'*********
ChDir (DirA)
   'MsgBox (CurDir)
  'MsgBox (fname)
Open fname For Input As #1
   N1 = 0
   N = 0
   Do While Not EOF(1)
      Line Input #1, textline
      N = N + 1    'n is the number of records or values first redord is header not data
      N1 = N1 + 1  'N1 is the number of records or values do not reset this counter first recored is header not data
   Loop
   N = N - 1
   N1 = N1 - 1
```

```
Close #1
If (N1 - 20 < 0) Then
   MsgBox "Not enough data points.  Please check Initial Parameters Initial read data."
   GoTo 44
End If
'********
'Allocate space to hold arrays
'********
j = 6       'j is the number of channels of data in the raw data file

ReDim time0(N + 1) As String '+1?
ReDim t(N) As Double
ReDim tb(N)
ReDim ch(j, N1) As Double
ReDim chb(j, N1)

Open fname For Input As #5
   j = 0
   k = 0
   Do While Not EOF(5)
      If j = 0 Then
         Input #5, time0(j), bline, heater, xxx, SerNumber, DomeDia, site   ' xxx = tail skiped
      End If
      j = j + 1
      k = k + 1
      Input #5, time0(j), tb(j), chb(1, j), chb(2, j), chb(3, j), chb(4, j), chb(5, j), chb(6, j)
   Loop
Close #5
For i = 1 To 6
   For ii = 1 To k
      ch(i, ii) = CDbl(chb(i, ii))    'convert byte data to double precision
   Next ii
Next i
For ii = 1 To k
   t(ii) = CDbl(tb(ii))    ' convert byte data to double precision
Next ii
      '   lblSiteD.Caption = site
      ' MsgBox (" number of data records = " & j)
      '13  j = j - 1
'********
' Find serial Number
'********
For i = 2 To l - 1
   If SerNumber - SerNo(i) = 0 Then GoTo 100
Next i
If i = l Then SerNo(i - 1) = "default"  ' Serial number not detected
i = i - 1
      '100       lblSerNo.Caption = SerNo(i)
100   ii = i ' ii is used as the record number of the for the calibration data
      '   MsgBox (ii)
'******
'   Convert Data to temperatures
'******
```

```
Dim temp     '
ReDim Res(6, N1)

For i = 1 To N1    'N1 is the number of data records

    Res(1, i) = ch(1, i)
'    MsgBox (Res(1, i))
    Res(2, i) = Res(1, i) - (ch(2, i) / 10)
    Res(3, i) = Res(1, i) - (ch(3, i) / 10)
    Res(4, i) = Res(1, i) - (ch(4, i) / 10)
    Res(5, i) = Res(1, i) - (ch(5, i) / 10)
    Res(6, i) = Res(1, i) - (ch(6, i) / 10)
    Res(1, i) = ch(1, i) - 1820                 '1820 is the series resistance on the circuit board added to Channel 1
Next i

'  Now convert to temperature and return as the original variable
For i = 1 To N1

    If Res(1, i) > 0 Then
    temp = ch1a(ii) + ch1b(ii) * Log(Res(1, i) / 10000)
    temp = temp + ch1c(ii) * ((Log(Res(1, i) / 10000)) ^ 3)
    ch(1, i) = (1 / temp) - 273.15             ' 273.15 convert from Kelvin to Celsus
    Else: ch(1, i) = 0
    End If

    If Res(2, i) > 0 Then
    temp = ch2a(ii) + ch2b(ii) * Log(Res(2, i) / 10000)
    temp = temp + ch2c(ii) * ((Log(Res(2, i) / 10000)) ^ 3)
    ch(2, i) = (1 / temp) - 273.15
    Else: ch(2, i) = 0
    End If

    If Res(3, i) > 0 Then
    temp = ch3a(ii) + ch3b(ii) * Log(Res(3, i) / 10000)
    temp = temp + ch3c(ii) * ((Log(Res(3, i) / 10000)) ^ 3)
    ch(3, i) = (1 / temp) - 273.15
    Else: ch(3, i) = 0
    End If

    If Res(4, i) > 0 Then
    temp = ch4a(ii) + Ch4b(ii) * Log(Res(4, i) / 10000)
    temp = temp + Ch4c(ii) * ((Log(Res(4, i) / 10000)) ^ 3)
    ch(4, i) = (1 / temp) - 273.15
    Else: ch(4, i) = 0
    End If

    If Res(5, i) > 0 Then
    temp = ch5a(ii) + Ch5b(ii) * Log(Res(5, i) / 10000)
    temp = temp + Ch5c(ii) * ((Log(Res(5, i) / 10000)) ^ 3)
    ch(5, i) = (1 / temp) - 273.15
    Else: ch(5, i) = 0
    End If
```

```
   If Res(6, i) > 0 Then
   temp = Ch6a(ii) + Ch6b(ii) * Log(Res(6, i) / 10000)
   temp = temp + Ch6c(ii) * ((Log(Res(6, i) / 10000)) ^ 3)
   ch(6, i) = (1 / temp) - 273.15
   Else: ch(6, i) = 0
   End If

Next i
'******
' Convert temperatures from actual temperature in deg C to differential temperature in Deg C
'******
For i = 1 To N1
   For jj = 2 To 6
      ch(jj, i) = ch(jj, i) - ch(1, i)
   Next jj
Next i

'MsgBox (ch(1, 1))
'MsgBox (ch(2, 1))
'Open fname2 For Output As #11   'file for temperatures in deg C
'    For i = 1 To N
'       Write #11, Format(t(i), "#0.0000"), Format(ch(1, i), "#0.0000"), _
'          Format(ch(2, i), "#0.0000"), Format(ch(3, i), "#0.0000"), Format(ch(4, i), "#0.0000"), _
'          Format(ch(5, i), "#0.0000"), Format(ch(6, i), "#0.0000")
'    Next i
'Close #11

'******
' Determine if sufficient data for possible analysis
'******
If (t(j - 1) - bline * 60 < 0) Then
   MsgBox "Not enough data points.  Please check Initial Parameters."
   GoTo 44

End If

'******
' Analyze base line data and determine slope and intercept of the base line
'******
jj = 0
Dim offset
offset = bline * 60 + ((5 * GateInterval& / 1000) + GateInterval1& / 1000) ' Note:
         '   GateInterval is in milliseconds calculations are in seconds
'MsgBox ("offset = " & offset)
   Do
      jj = jj + 1     '  jj is sample number before heater is turned on
   Loop Until (t(jj) - offset) >= 0
      jj = jj - 2

         'MsgBox ("sample number before heater turned on = " & jj)
         'calculate sums used to calculate baseline averages, slopes, and intercepts
      For j = 1 To 6
         sumx(j) = 0
```

```
       sumy(j) = 0
       sumxy(j) = 0
       sumxx(j) = 0
    Next j
       '   Determind average values
    For ct = 1 To jj
       For j = 1 To 6
          sumx(j) = sumx(j) + t(ct)
          sumy(j) = sumy(j) + ch(j, ct)
       Next j
    Next ct

    For j = 1 To 6
       avgx(j) = sumx(j) / jj
       avgy(j) = sumy(j) / jj
    Next j
          'MsgBox ("avgx(4) =" & avgx(4))
          'MsgBox (avgx(2))
          'MsgBox ("avgy(4)= " & avgy(4))
    Dim val1 As Double
    Dim val2 As Double
    For ct = 1 To jj
       For j = 1 To 6
          val1 = (t(ct) - avgx(j)) * (ch(j, ct) - avgy(j))
          val2 = (t(ct) - avgx(j)) ^ 2
          sumxy(j) = sumxy(j) + val1
          sumxx(j) = sumxx(j) + val2
       Next j
    Next ct
    '*********
    '  Calculate Slopes and Intercepts
    '*********
    For j = 1 To 6
       slope(j) = sumxy(j) / sumxx(j)
       sInt(j) = avgy(j) - (slope(j) * avgx(j))
    Next j
       '*********
       '  Correct for baseline drift
       '*********
    For i = 1 To N
       For j = 1 To 6
          ch(j, i) = ch(j, i) - (slope(j) * t(i) + sInt(j))
        Next j
    Next i

'   Open fname1 For Output As #11   'file with change in temperature adjusted for base line slope
'      For i = 1 To N
'         Write #11, Format(t(i), "#0.00"), Format(ch(1, i), "#0.00"), _
'            Format(ch(2, i), "#0.00"), Format(ch(3, i), "#0.00"), Format(ch(4, i), "#0.00"), _
'            Format(ch(5, i), "#0.00"), Format(ch(6, i), "#0.00")
'      Next i
'   Close #11
```

```
'********
' find time at maximum signal
' the time is calculated as max time - base line time - (pulse time/2)
' This will be one of the metrics evaluated to determine travel time
'********
Dim jmax(6) As Integer  'sample number at max temperature

For i = 1 To 6
   tempmax(i) = ch(i, jj + 1)  ' only consider samples collected after baseline data
Next i
For i = 1 To 6
   For j = jj + 2 To N
      If (ch(i, j) - tempmax(i) > 0) Then
         jmax(i) = j
         tempmax(i) = ch(i, j)
      End If
   Next j
Next i
tmax(1) = t(jmax(1)) + GateInterval1& / 2000
For i = 2 To 6
   If tempmax(i) < 0.075 Then  ' 0.075 considered minimum peak to be used eliminates some of the noise
      tmax(i) = bline * 60 + (heater / 2)

   Else: tmax(i) = t(jmax(i)) + (GateInterval& / 1000 * (i - 1)) + GateInterval& / 2000 + GateInterval1& / 1000
   End If
Next i
      'MsgBox ("tempmax(4) = " & tempmax(4))
      'MsgBox ("tmax(4) = " & tmax(4))

'********
'Begin Moment analysis - Data will be extrapolated based on a lograthmetic
' function based on a sum of squares fit to the data between 80% of tempmax
' and next occurance of <= 0.04 deg C or the end of file which ever comes first
' first determine sample number correlated to 80%of tempmax to perform moment analysis
' there should be only one peak and it should occure in the first 40% of the data set
' if peak temperature occures after the 40% of the data points do not make moment analysis
'********
Dim jbegin(6) As Integer   'jbegin is the sample number for the first sample to use in generating fit
Dim jend(6) As Integer     'jend is the last sample to use
For i = 2 To 6
   If (jmax(i) + 20 - N1) > 0 Then     'See if there is sufficient data to calculate moments
      GoTo 200
   End If

   j = jmax(i)

   Do Until ((tempmax(i) * 0.8) - ch(i, j) > 0) 'determine sample number for first sample
                             ' to use in interpolating and extrapolating
                             ' moments 80% of peak was qualitatively
                             ' estimated as being sufficient to be away
                             ' from the curvature at the peek
      j = j + 1
      If (j - N = 0) Then
```

```
               jbegin(i) = j
               jend(i) = j
               GoTo 90
            End If
            '      jend(i) = jend(i) + 1
         Loop
         jbegin(i) = j
         If (jbegin(i) + 20 - N1) > 0 Then
            GoTo 90
         End If

         j = j + 1
         If j >= N1 Then GoTo 89
         Do Until (0.05 - ch(i, j) > 0) 'determine last sample number considered valid for
                           'interpolating or extrapolating moments noise was measured
                           'at +- 0.005 deg C.  data has already been adjusted for baseline
                           'offset. Resolution is 0.005 deg C.

            j = j + 1
            If j >= N1 Then GoTo 89
         Loop
89         jend(i) = j - 1
90      Next i
      '********
      ' Fit log transformed data to a streight line first determine if there is enough data
      ' there should be a minimum of minobs observations to calculate the extrapolation function
      '********
      Dim sumyy(6)
      Dim avgxx(6)
      Dim avgyy(6)
      Dim Mslope(6) As Double
      Dim Mint(6) As Double
      Dim minobs As Integer   'minimum number of observations between jbegin(i) and jend(i)
                  'to permit moment analysis 20 is considered minimum
      Dim minsignal As Single 'minimum peak signal in volts for analysis
      minobs = 20
      minsignal = 0.05

      For i = 2 To 6  'initialize values
         sumxx(i) = 0
         sumyy(i) = 0
         sumxy(i) = 0
         avgxx(i) = 0
         avgyy(i) = 0
         Mslope(i) = 0
         Mint(i) = 0
      Next i
      For i = 2 To 6
         If ((jend(i) - jbegin(i) - minobs) > 0) And (tempmax(i) - minsignal > 0) Then
            For j = jbegin(i) To (jend(i) - 1)
               kk = j + 1
               avgxx(i) = avgxx(i) + t(kk)
               If ch(i, kk) < 0 Then ch(i, j) = 1E-50
         avgyy(i) = avgyy(i) + Log(ch(i, kk))
      Next j
      avgxx(i) = avgxx(i) / (jend(i) - jbegin(i))
      avgyy(i) = avgyy(i) / (jend(i) - jbegin(i))
   End If
Next i
For i = 2 To 6
   If ((jend(i) - jbegin(i) - minobs) > 0) And (tempmax(i) - minsignal > 0) Then
      For j = jbegin(i) To (jend(i) - 1)
         kk = j + 1
         If ch(i, kk) < 0 Then ch(i, kk) = 1E-50
         sumxx(i) = sumxx(i) + (t(kk) - avgxx(i)) ^ 2
         sumyy(i) = sumyy(i) + (Log(ch(i, kk)) - avgyy(i)) ^ 2
         sumxy(i) = sumxy(i) + (t(kk) - avgxx(i)) * (Log(ch(i, kk)) - avgyy(i))
      Next j
      Mslope(i) = sumxy(i) / sumxx(i)
      Mint(i) = avgyy(i) - (Mslope(i) * avgxx(i))
   End If
Next i
'*********
'Calculate zero and first moments
'*********
Dim Momzero(6)
Dim Mom1(6)
Dim c1, c2
Dim T1, T2
Dim timecor 'time correction such that zero time is at the end of base line measurements
Dim jstart 'sample number to begin moment calculations this is to remove switching noise
timecor = bline * 60
For i = 2 To 6
   If ((jend(i) - jbegin(i) - minobs) < 0) Or (tempmax(i) - minsignal < 0 Or Mslope(i) > 0) Then
      Mom1(i) = "N.A."
         '        GoTo 300
   Else
      Momzero(i) = 0
      Mom1(i) = 0
      ch(i, 0) = 0
      t(0) = 0
      '*********
      '  Start moment analysis beginning with sample after heat-pulse
      '  Correct time scale to begin half way through the heat-pulse in moment calculations
      '  First step is to determine sample number after end of heat-pulse
      '*********
      j = 1
      Do Until (t(j) - bline * 60 - heater / 2) > 0
         j = j + 1
      Loop
      jstart = j
      For j = jstart To jbegin(i)        'first part of curve
         Momzero(i) = Momzero(i) + ((t(j) - t(j - 1)) * (ch(i, j) + ch(i, j - 1))) / 2
         Mom1(i) = Mom1(i) + (1 / 6) * ((ch(i, j - 1) * (t(j - 1) - timecor) + ch(i, j) * _
            (t(j) - timecor)) * (t(j) - t(j - 1)) + (ch(i, j - 1) _
            + ch(i, j)) * ((t(j) - timecor) * (t(j) - timecor) - (t(j - 1) - timecor) * _
            (t(j - 1) - timecor)))
        Next j
        'end of first part of curve begin analysis for the curve fit portion of the curve
        c1 = ch(i, j)
        T1 = t(j)
        j = 1
        Do Until (c1 - 0.001 * tempmax(i) < 0)
          T2 = T1 + 1
          c2 = Exp(Mslope(i) * (T2) + Mint(i))
          Momzero(i) = Momzero(i) + (c1 + c2) / 2
          Mom1(i) = Mom1(i) + (1 / 6) * ((c1 * (T1 - timecor) + c2 * (T2 - timecor)) * _
            (T2 - T1) + _
            (c1 + c2) * ((T2 - timecor) * (T2 - timecor) - (T1 - timecor) * (T1 - timecor)))
          T1 = T2
          c1 = c2
        Loop
      End If

300   Next i
    Dim Mon2, Mom3, Mom4, Mom5, Mom6
        'MsgBox (Mom1(1))

    If ((jend(2) - jbegin(2)) > minobs) And (tempmax(2) - minsignal > 0 And IsNumeric(Mom1(2)) _
      And Momzero(2) > 0 And IsNumeric(Momzero(2))) Then
      '   lblMom2.Caption = Format(Mom1(2) / Momzero(2), "#0.0")
      Mom2 = Mom1(2) / Momzero(2)
    Else
        '   lblMom2.Caption = Format("N.A.")
      Mom2 = 0
    End If
    If ((jend(3) - jbegin(3)) > minobs) And (Not Not tempmax(3) - minsignal > 0 And IsNumeric(Mom1(3)) _
      And Momzero(3) > 0 And IsNumeric(Momzero(3))) Then
        '   lblMom3.Caption = Format(Mom1(3) / Momzero(3), "#0.0")
      Mom3 = Mom1(3) / Momzero(3)
    Else
        '   lblMom3.Caption = Format("N.A.")
      Mom3 = 0
    End If
    If ((jend(5) - jbegin(5)) > minobs) And (tempmax(5) - minsignal > 0 And IsNumeric(Mom1(4)) _
      And Momzero(5) > 0 And IsNumeric(Momzero(5))) Then
        '   lblMom5.Caption = Format(Mom1(5) / Momzero(5), "#0.0")
      Mom5 = Mom1(5) / Momzero(5)
    Else
        '   lblMom5.Caption = Format("N.A.")
      Mom5 = 0
    End If
    If ((jend(6) - jbegin(6)) > minobs) And (tempmax(6) - minsignal > 0 And IsNumeric(Mom1(6)) _
      And Momzero(6) > 0 And IsNumeric(Momzero(6))) Then
        '   lblMom6.Caption = Format(Mom1(6) / Momzero(6), "#0.0")
      Mom6 = Mom1(6) / Momzero(6)
    Else
        '   lblMom6.Caption = Format("N.A.")
      Mom6 = 0
    End If

    '*********
    ' calculate flows
    '*********
    Dim CL  'Confidence limit
    CL = 2.326
      '  Channel 2
    If ((jend(2) - jbegin(2) > minobs) And (tempmax(2) - minsignal > 0) And _
        (tmax(2) - bline * 60 - 1.5 * heater > 0) And IsNumeric(Mom1(2))) Then
'       Mom1(2) = (Mom1(2) / Momzero(2))
'       Q2 = (c10(ii) - (c30(ii) * Mom1(2))) / (Mom1(2) + c20(ii))
      Q2 = (c10(ii) - (c30(ii) * tmax(2) - (bline * 60 + heater / 2))) / _
        (tmax(2) - (bline * 60 + heater / 2) + c20(ii))
        'MsgBox (Mom1(1))
        'MsgBox (c10(ii))
    Else
      Q2 = 0
    End If
      '  Channel 3
    If (jend(3) - jbegin(3) > minobs) And (tempmax(3) - minsignal > 0) And (tmax(3) - bline * 60 - 1.5 * heater > 0 _
        And IsNumeric(Mom1(3))) Then
'       Mom1(3) = (Mom1(3) / Momzero(3))
'       Q3 = (c11(ii) - c31(ii) * Mom1(3)) / (Mom1(3) + c21(ii))
      Q3 = (c11(ii) - c31(ii) * tmax(3) - (bline * 60 + heater / 2)) / (tmax(3) - (bline * 60 + heater / 2) + c21(ii))
    Else
      Q3 = 0
    End If
      '  Channel 5
    If (jend(5) - jbegin(5) > minobs) And (tempmax(5) - minsignal > 0) And _
        (tmax(5) - bline * 60 - 2 * heater > 0 And IsNumeric(Mom1(5))) Then
'       Mom1(5) = (Mom1(5) / Momzero(5))
'       Q5 = (c13(ii) - c33(ii) * Mom1(5)) / (Mom1(5) + c23(ii))
      Q5 = (c13(ii) - c33(ii) * tmax(5) - (bline * 60 + heater / 2)) / (tmax(5) - (bline * 60 + heater / 2) + c23(ii))
        'MsgBox (c13(ii))
        'MsgBox (c33(ii))
        'MsgBox (c23(ii))
        '     lblQ3.Caption = Format(Q3, "#0.000")
    Else
      Q5 = 0
    End If
      '  Channel 6 Captions
    If (jend(6) - jbegin(6) > minobs) And (tempmax(6) - minsignal > 0) And _
        (tmax(6) - bline * 60 - 1.5 * heater > 0 And IsNumeric(Mom1(6))) Then
'       Mom1(6) = (Mom1(6) / Momzero(6))
'       Q6 = (c14(ii) - c34(ii) * Mom1(6)) / (Mom1(6) + c24(ii))
      Q6 = (c14(ii) - c34(ii) * tmax(6) - (bline * 60 + heater / 2)) / (tmax(5) - (bline * 60 + heater / 2) + c24(ii))
        'MsgBox (Q4)
    Else
      Q6 = 0
    End If
    '*********
    ' Calculate Darcy velocity
    '*********
    If DomeDia > 0.1 Then   'Data exists to calculate flux
      If Q2 <> 0 Then
        V2 = Q2 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      Else
        V2 = 0
      End If
      If Q3 <> 0 Then
        V3 = Q3 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      Else
        V1 = 0
      End If
      If Q5 <> 0 Then
        V5 = Q5 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      Else
        V5 = 0
      End If
      If Q6 <> 0 Then
        V6 = Q6 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      Else
        V6 = 0
      End If
    End If
      'MsgBox ("check3")

    '********
    ' set up arrays for ploting
    '********

200   Dim Graph() As Single
    Dim x As Integer
    Dim myXarray() As Double        'Time
    Dim myYArray0() As Double
    Dim myYArray1() As Double
    Dim myYArray2() As Double
    Dim myYArray3() As Double
    Dim myYArray4() As Double
    Dim myYArray5() As Double

    ReDim myXarray(N)
    ReDim myYArray0(N)
    ReDim myYArray1(N)
    ReDim myYArray2(N)
    ReDim myYArray3(N)
    ReDim myYArray4(N)
    ReDim myYArray5(N)

    '*******
    'Generate some x y data.
    '*******

    myXarray(0) = t(1)
    myYArray0(0) = ch(1, 1)
```

```
   'MsgBox (ch(2, 1))
myYArray1(0) = ch(2, 1)
myYArray2(0) = ch(3, 1)
myYArray3(0) = ch(4, 1)
myYArray4(0) = ch(5, 1)
myYArray5(0) = ch(6, 1)

   'MsgBox N
For x = 1 To N - 1

   myXarray(x) = t(x) - bline * 60    'value for X-axis
   myYArray0(x) = ch(1, x)         'value for Y-axis Ref Temp
   myYArray1(x) = ch(2, x)          ' Ch 2 cable end temp
   myYArray2(x) = ch(3, x)          'Ch 3 temp
   myYArray3(x) = ch(4, x)          ' Heater Temp
   myYArray4(x) = ch(5, x)
   myYArray5(x) = ch(6, x)          'Discharge end temp

Next x


With TChart1
   .AddSeries scPoint
   .Series(0).AddArray UBound(myYArray0), myYArray0(), myXarray() ' ref temp
   .Series(0).XValues.TempValue = True
   .Series(1).AddArray UBound(myYArray1), myYArray1(), myXarray()  ' Cable end Temp
   .Series(2).AddArray UBound(myYArray2), myYArray2(), myXarray()
   .Series(3).AddArray UBound(myYArray3), myYArray3(), myXarray()  ' Heater
   .Series(4).AddArray UBound(myYArray4), myYArray4(), myXarray()
   .Series(5).AddArray UBound(myYArray5), myYArray5(), myXarray()
End With
Refresh
DoEvents
   'MsgBox ("check 4")
   '   Screen.MousePointer = 1
'**********
'Convert Suplimential Data
'**********
Dim time01
Dim ch7, ch8, ch9, ch19
Open fnameS For Input As #11
   Input #11, time01, ch7, ch8, ch9, ch10, SerialNo, site
Close #11
   ' MsgBox (ch7)

'**********
' Convert Channel 7 to temperature
'**********
If ch7 > 0 Then
   temp = Ch7a(ii) + Ch7b(ii) * Log(ch7) / 10000
   temp = temp + Ch7c(ii) * ((Log(ch7)) / 10000 ^ 3)
   ch7 = (1 / temp) - 273.15            ' 273.15 convert from Kelvin to Celsus
Else: ch7 = 0
```

```
    End If


    '***********
    ' write output file with processed data
    '***********
    'ChDir Dir1
    Open fnameP For Append As #8
      Write #8, Format$(time0(1)), Format(tmax(2) - (bline * 60 + heater / 2), "#0.00"), Format(tmax(3) - (bline *
60 + heater / 2), "#0.00"), _
        Format(tmax(5) - (bline * 60 + heater / 2), "#0.00"), Format(tmax(6) - (bline * 60 + heater / 2), "#0.00"), _
        Format(Mom1(2), "#0.00"), Format(Mom1(3), "#0.00"), _
        Format(Mom1(5), "#0.00"), Format(Mom1(6), "#0.00"), _
        Format(avgy(1), "#0.00"), Format(ch7, "#0.00"), Format(ch8 / 10000, "#0.00"), _
        Format(ch9 / 10000, "#0.00"), Format(ch10 / 10000, "#0.00"), _
        Format(tempmax(1), "#0.00"), Format(tempmax(2), "#0.00"), Format(tempmax(3), "#0.00"), _
        Format(tempmax(4), "#0.00"), Format(tempmax(5), "#0.00"), Format(tempmax(6), "#0.00")
    Close #8
    Refresh
    DoEvents

    '********
    'TChart3 Peak Arrival Times
    '********
    'Determine number of records No header in file
    '******
    ' determine the number of lines in the data table
    '******
    Open fnameP For Input As #6
      l = 0
      Do While Not EOF(6)
        Line Input #6, textline
        l = l + 1    'l is the number of lines of data
      Loop
    Close #6
    Dim tmax2(), tmax3(), tmax5(), tmax6(), Mom2d(), Mom3d(), Mom5d(), Mom6d(), Reft(), SupT(), DifP(),
chan9(), _
      chan10(), txt(34)
    Dim DaTime() As Date
    ReDim tmax2(l), tmax3(l), tmax5(l), tmax6(l), Mom2d(l), Mom3d(l), Mom5d(l), Mom6d(l), Reft(l), SupT(l),
DifP(l), chan9(l), chan10(l) ' read in data
    ReDim DaTime(l)
    i = 4
    Open fnameP For Input As #6
      Do Until i = l + 1

        If i = 4 Then
          Input #6, txt(1), txt(2), txt(3), txt(4), txt(5), txt(6), txt(7), txt(8), txt(9), txt(10), txt(11), txt(12), _
          txt(13), txt(14), txt(15), txt(16), txt(17), txt(18), txt(19), txt(20), txt(21), _
          txt(22), txt(23), txt(24), txt(25), txt(26), txt(27), txt(28), txt(29), _
          txt(30), txt(31), txt(32), txt(33), txt(34), _
          time0(i - 4), tmax2(i - 4), tmax3(i - 4), tmax5(i - 4), tmax6(i - 4), Mom2d(i - 4), Mom3d(i - 4), _
          Mom5d(i - 4), Mom6d(i - 4), Reft(i - 4), SupT(i - 4), DifP(i - 4), chan9(i - 4), chan10(i - 4), _
      tempmax(1), tempmax(2), tempmax(3), tempmax(4), tempmax(5), tempmax(6)
    ElseIf i > 4 Then
      Input #6, time0(i - 4), tmax2(i - 4), tmax3(i - 4), tmax5(i - 4), tmax6(i - 4), Mom2d(i - 4), Mom3d(i - 4), _
      Mom5d(i - 4), Mom6d(i - 4), Reft(i - 4), SupT(i - 4), DifP(i - 4), chan9(i - 4), chan10(i - 4), _
      tempmax(1), tempmax(2), tempmax(3), tempmax(4), tempmax(5), tempmax(6)
    End If
    DaTime(i - 4) = time0(i - 4)
    i = i + 1
  Loop
Close #6
l = i - 4

'   MsgBox (l)
'   MsgBox (DaTime(1))
'   MsgBox (tmax6(1))
'********
' set up arrays for ploting
'********
Dim Graph3() As Single

Dim myXarray2() As Double
Dim myYArray20() As Double      ' tmax2 time to peak temp
Dim myYArray21() As Double      ' tmax3
Dim myYArray22() As Double      ' tmax5
Dim myYArray23() As Double      ' tmax6
Dim myYArray24() As Double      ' Mom2d first temporal moment
Dim myYArray25() As Double      ' Mom3d
Dim myYArray26() As Double      ' Mom5d
Dim myYArray27() As Double      ' Mom6d
Dim myYArray28() As Double      ' Reft Reference temperature
Dim myYArray29() As Double      ' SupT  Extra temperature measurement Outside/Inside Dome
Dim myYArray210() As Double      ' DifP  Differential pressure
Dim myYArray211() As Double      ' chan9
Dim myYArray212() As Double      ' chan10

ReDim myXarray2(l)
ReDim myYArray20(l)
ReDim myYArray21(l)
ReDim myYArray22(l)
ReDim myYArray23(l)
ReDim myYArray24(l)
ReDim myYArray25(l)
ReDim myYArray26(l)
ReDim myYArray27(l)
ReDim myYArray28(l)
ReDim myYArray29(l)
ReDim myYArray210(l)
ReDim myYArray211(l)
ReDim myYArray212(l)

'*******
'Generate some x y data.
'*******
```

```
myXarray2(0) = DaTime(0)
myYArray20(0) = tmax2(0)
  'MsgBox (ch(2, 1))
myYArray21(0) = tmax3(0)
myYArray22(0) = tmax5(0)
myYArray23(0) = tmax6(0)
If IsNumeric(Mom2d(0)) Then
  myYArray24(0) = Mom2d(0)
Else: myYArray24(0) = Empty
End If
If IsNumeric(Mom3d(0)) Then
  myYArray25(0) = Mom3d(0)
Else: myYArray25(0) = Empty
End If
If IsNumeric(Mom5d(0)) Then
  myYArray26(0) = Mom5d(0)
Else: myYArray26(0) = Empty
End If
If IsNumeric(Mom6d(0)) Then
  myYArray27(0) = Mom6d(0)
Else: myYArray27(0) = Empty
End If
myYArray28(0) = Reft(0)
myYArray29(0) = SupT(0)
myYArray210(0) = DifP(0)
myYArray211(0) = chan9(0)
myYArray212(0) = chan10(0)

  'MsgBox N
For i = 0 To l

  myXarray2(i) = DaTime(i) 'value for X-axis
  myYArray20(i) = tmax2(i)       'time to peek channel 2
  myYArray21(i) = tmax3(i)        '
  myYArray22(i) = tmax5(i)        '
  myYArray23(i) = tmax6(i)         '
  If IsNumeric(Mom2d(i)) Then
    myYArray24(i) = Mom2d(i)
  Else: myYArray24(i) = Empty
  End If
  If IsNumeric(Mom3d(i)) Then
    myYArray25(i) = Mom3d(i)
  Else: myYArray25(i) = Empty
  End If
  If IsNumeric(Mom5d(i)) Then
    myYArray26(i) = Mom5d(i)
  Else: myYArray26(i) = Empty
  End If
  If IsNumeric(Mom6d(i)) Then
    myYArray27(i) = Mom6d(i)
  Else: myYArray27(i) = Empty
  End If
```

```
    myYArray28(i) = Reft(i)
    myYArray29(i) = SupT(i)
    myYArray210(i) = DifP(i)
    myYArray211(i) = chan9(i)
    myYArray212(i) = chan10(i)

Next i


With TChart3
    .AddSeries scPoint
    .Series(0).AddArray UBound(myYArray20), myYArray20(), myXarray2() '
    .Series(0).XValues.DateTime = True
    .Series(1).AddArray UBound(myYArray21), myYArray21(), myXarray2()  '
    .Series(1).XValues.DateTime = True
    .Series(2).AddArray UBound(myYArray22), myYArray22(), myXarray2()
    .Series(2).XValues.DateTime = True
    .Series(3).AddArray UBound(myYArray23), myYArray23(), myXarray2()
    .Series(3).XValues.DateTime = True
End With
Refresh
DoEvents
With TChart4
    .AddSeries scPoint
    .Series(0).AddArray UBound(myYArray24), myYArray24(), myXarray2() '
    .Series(0).XValues.DateTime = True
    .Series(1).AddArray UBound(myYArray25), myYArray25(), myXarray2()  '
    .Series(1).XValues.DateTime = True
    .Series(2).AddArray UBound(myYArray26), myYArray26(), myXarray2()
    .Series(2).XValues.DateTime = True
    .Series(3).AddArray UBound(myYArray27), myYArray27(), myXarray2()
    .Series(3).XValues.DateTime = True
End With
Refresh
DoEvents
With TChart5
    .AddSeries scPoint
    .Series(0).AddArray UBound(myYArray28), myYArray28(), myXarray2() '
    .Series(0).XValues.DateTime = True
    .Series(1).AddArray UBound(myYArray29), myYArray29(), myXarray2()  '
    .Series(1).XValues.DateTime = True
    .Series(2).AddArray UBound(myYArray210), myYArray210(), myXarray2()
    .Series(2).XValues.DateTime = True
End With
Refresh
DoEvents
With TChart6
    .AddSeries scPoint
    .Series(0).AddArray UBound(myYArray211), myYArray211(), myXarray2() '
    .Series(0).XValues.DateTime = True
    .Series(1).AddArray UBound(myYArray212), myYArray212(), myXarray2()  '
    .Series(1).XValues.DateTime = True
End With
```

```
       Refresh
       DoEvents
       GoTo 45
44     Unload Me
'   Screen.MousePointer = 1

45 End Sub
```

frmDataAnalysis

```
' last revised 04/20/05 by Carl Enfield

Private Sub Dir2_Change()
   File2.Path = Dir2
End Sub

Private Sub File2_Click()
  Dim textline
  Dim dt, arm, y1, y2 As Double
  Dim time0() As String
  Dim time1 As String
  Dim ch1(), ch2(), ch3(), ch4(), ch5(), ch6(), ch7(), ch8()
  Dim t()
  Dim a1, a2, a3, a4, a5, a6 As Double
  Dim m1, m2, m3, m4, m5, m6 As Double
  Dim c1() As Double
  Dim c2() As Double
  Dim c3() As Double
  Dim c4() As Double
  Dim c5() As Double
  Dim c6() As Double
  Dim x1() As Double
  Dim x2() As Double
  Dim x3() As Double
  Dim x4() As Double
  Dim x5() As Double
  Dim x6() As Double
  Dim ct, ipt, pt_odd, pt_half As Integer
  Dim b1, s1 As Single
  Dim b2, s2 As Single
  Dim b3, s3 As Single
  Dim b4, s4 As Single
  Dim b5, s5 As Single
  Dim b6, s6 As Single
  Dim d1, d2, d3, d4, d5, d6 As Double
  Dim tmax1, tmax2, tmax3, tmax4, tmax5, tmax6 As Double
  Dim x1max, x2max, x3max, x4max, x5max, x6max
  Dim pt
  Dim baseline$, heater

  pt = CDbl(txtPtavg.Text)  'points average
  If pt = 0 Then MsgBox "Invalid Points Average Number.": GoTo 44

  pt_odd = pt Mod 2
  If pt_odd <> 1 Then MsgBox "Please enter an odd number for points average.": GoTo 44

  pt_half = pt / 2 - 0.5

  baseline$ = 10#
  heater = 5#
  heater = heater + baseline$

  'ChDir "c:\temp\"
  ChDir Dir2
  Open File2 For Input As #1

  N = 0
  Do While Not EOF(1)
  Line Input #1, textline
  N = N + 1
  Loop

  Close #1

  k = N - pt + 1
  g = k + pt_half
  ReDim x1(g), x2(g), x3(g), x4(g), x5(g), x6(g)

  ReDim time0(N + 1) As String
  ReDim ch1(N), ch2(N), ch3(N), ch4(N), ch5(N), ch6(N), t(N), ch7(N), ch8(N)
  ReDim c1(N), c2(N), c3(N), c4(N), c5(N), c6(N)

  Open File2 For Input As #3
  Input #3, time1
  Close #3

  If (Year(time1)) > 2004 Then GoTo 3
  If (Year(time1) >= 2004 And Month(time1) > 4) Then GoTo 3
  If Month(time1) = 4 And Day(time1) >= 26 Then GoTo 3 Else GoTo 4

3  Open File2 For Input As #2
  j = 0
  N = 0
  Do While Not EOF(2)
   If j = 0 Then
   Input #2, time0(j), baseline, heater
   End If
  j = j + 1
  N = N + 1
  Input #2, time0(j), t(j), ch1(j), ch2(j), ch3(j), ch4(j), ch5(j), ch6(j), ch7(N), ch8(N)
  'baseline = 10 sec.
  'heater = 3 sec.
  'If (t(j) > 10 And t(j) <= 13) Then GoTo 13 'skip data collected while heater was on

  Loop
  GoTo 5

4  Open File2 For Input As #2
  j = 0
  Do While Not EOF(2)
  j = j + 1
14 Input #2, time0(j), t(j), ch1(j), ch2(j), ch3(j), ch4(j), ch5(j), ch6(j), ch7(j), ch8(j)
  'If t(j) > 10 And t(j) < 14 Then GoTo 14 'skip data collected while heater was on
  Loop
5  Close #2

  If t(j) <= baseline$ Then MsgBox "Not enough data points.  Please check file.": GoTo 44

  ct = 1
  Do While t(ct) <= baseline$
  s1 = s1 + ch1(ct)
  s2 = s2 + ch2(ct)
  s3 = s3 + ch3(ct)
  s4 = s4 + ch4(ct)
  s5 = s5 + ch5(ct)
  's6 = s6 + ch6(ct)
  ct = ct + 1
  Loop

  b1 = s1 / (ct - 1)
  b2 = s2 / (ct - 1)
  b3 = s3 / (ct - 1)
  b4 = s4 / (ct - 1)
  b5 = s5 / (ct - 1)
  'b6 = s6 / (ct - 1)

  For i = 1 To j
  d1 = ch1(i) - b1
  c1(i) = d1
  d2 = ch2(i) - b2
  c2(i) = d2
  d3 = ch3(i) - b3
  c3(i) = d3
  d4 = ch4(i) - b4
  c4(i) = d4
  d5 = ch5(i) - b5
  c5(i) = d5
'   d6 = ch6(i) - b6
'   c6(i) = d6
  Next i

  'take pt points average
  For m = pt_half + 1 To k + pt_half
   x1(m) = 0
   x2(m) = 0
   x3(m) = 0
   x4(m) = 0
   x5(m) = 0
'    x6(m) = 0
      For z = m - pt_half To m + pt_half
          x1(m) = x1(m) + c1(z)
          x2(m) = x2(m) + c2(z)
          x3(m) = x3(m) + c3(z)
          x4(m) = x4(m) + c4(z)
          x5(m) = x5(m) + c5(z)
'          x6(m) = x6(m) + c6(Z)
```

```
        Next z
     x1(m) = x1(m) / pt
     x2(m) = x2(m) / pt
     x3(m) = x3(m) / pt
     x4(m) = x4(m) / pt
     x5(m) = x5(m) / pt
'     x6(m) = x6(m) / pt
    Next m

    'find t at maximum signal
    x1max = x1(1)
    x2max = x2(1)
    x3max = x3(1)
    x4max = x4(1)
    x5max = x5(1)
'    x6max = x6(1)
    tmax1 = t(1)
    tmax2 = t(1)
    tmax3 = t(1)
    tmax4 = t(1)
    tmax5 = t(1)
'    tmax6 = t(1)

    For i = 1 + pt_half To k + pt_half
    If (x1(i) > x1max) Then
    x1max = x1(i)
    tmax1 = t(i)
    End If
    If (x2(i) > x2max) Then
    x2max = x2(i)
    tmax2 = t(i)
    End If
    If (x3(i) > x3max) Then
    x3max = x3(i)
    tmax3 = t(i)
    End If
    If (x4(i) > x4max) Then
    x4max = x4(i)
    tmax4 = t(i)
    End If
    If (x5(i) > x5max) Then
    x5max = x5(i)
    tmax5 = t(i)
    End If
'    If (x6(i) > x6max) Then
'    x6max = x6(i)
'    tmax6 = t(i)
'    End If
    Next i

     a1 = 0
     m1 = 0
```

```
For i = 1 To k - 1
 y1 = x1(i)
 y2 = x1(i + 1)
 dt = t(i + 1) - t(i)
 a1 = a1 + ((y1 + y2) / 2) * dt
 arm = (t(i) * 1 + t(i + 1) * 1) / 2 'moment arm
 m1 = m1 + (((y1 + y2) / 2) * dt) * arm
Next i

 a2 = 0
 m2 = 0
For i = 1 To k - 1
 y1 = x2(i)
 y2 = x2(i + 1)
 dt = t(i + 1) - t(i)
 a2 = a2 + ((y1 + y2) / 2) * dt
 arm = (t(i) * 1 + t(i + 1) * 1) / 2
 m2 = m2 + (((y1 + y2) / 2) * dt) * arm
Next i

 a3 = 0
 m3 = 0
For i = 1 To k - 1
 y1 = x3(i)
 y2 = x3(i + 1)
 dt = t(i + 1) - t(i)
 a3 = a3 + ((y1 + y2) / 2) * dt
 arm = (t(i) * 1 + t(i + 1) * 1) / 2
 m3 = m3 + (((y1 + y2) / 2) * dt) * arm
Next i

 a4 = 0
 m4 = 0
For i = 1 To k - 1
 y1 = x4(i)
 y2 = x4(i + 1)
 dt = t(i + 1) - t(i)
 a4 = a4 + ((y1 + y2) / 2) * dt
 arm = (t(i) * 1 + t(i + 1) * 1) / 2
 m4 = m4 + (((y1 + y2) / 2) * dt) * arm
Next i


 a5 = 0
 m5 = 0
For i = 1 To k - 1
 y1 = x5(i)
 y2 = x5(i + 1)
 dt = t(i + 1) - t(i)
 a5 = a5 + ((y1 + y2) / 2) * dt
 arm = (t(i) * 1 + t(i + 1) * 1) / 2
 m5 = m5 + (((y1 + y2) / 2) * dt) * arm
Next i
```

```
'    a6 = 0
'    m6 = 0
'   For i = 1 To k - 1
'    y1 = x6(i)
'    y2 = x6(i + 1)
'    dt = t(i + 1) - t(i)
'    a6 = a6 + ((y1 + y2) / 2) * dt
'    arm = (t(i) * 1 + t(i + 1) * 1) / 2
'    m6 = m6 + (((y1 + y2) / 2) * dt) * arm
'   Next i

   Text1.Text = Format(a1, "#.00") 'zero moment
   Text2.Text = Format(a2, "#.00")
   Text3.Text = Format(a3, "#.00")
   Text4.Text = Format(a4, "#.00")
   Text5.Text = Format(a5, "#.00")
'   Text6.Text = Format(a6, "#.00")
   Text7.Text = Format(m1 / a1, "#.00") 'normalized first moment
   Text8.Text = Format(m2 / a2, "#.00")
   Text9.Text = Format(m3 / a3, "#.00")
   Text10.Text = Format(m4 / a4, "#.00")
   Text11.Text = Format(m5 / a5, "#.00")
'   Text12.Text = Format(m6 / a6, "#.00")
   Text13.Text = Format(tmax1, "#.00")  'tmax
   Text14.Text = Format(tmax2, "#.00")
   Text15.Text = Format(tmax3, "#.00")
   Text16.Text = Format(tmax4, "#.00")
   Text17.Text = Format(tmax5, "#.00")
'   Text18.Text = Format(tmax6, "#.00")

44 End Sub

Private Sub CmdClose_Click()
    Unload Me
End Sub

Private Sub Form_Load()
   Dir2.Path = "c:\temp"
   File2.Path = Dir2
End Sub
```

frmDifferentialPres

```
'Last Revision 11/26/07 by Carl Enfield

' The purpose is to display differential pressure between the surrounding water and
'      inside the dome.

'Const BoardNum% = 0        ' Board number
Const FirstPoint& = 0      ' set first element in buffer to transfer to array
Const GateInterval1& = 500 ' Number of milliseconds used to count pulses and determine frequency used on ref temperature only

Dim CBCount%(10)
Dim ChipNum%
Dim counter%, rounds, RateCount
Dim DataValue%            ' digital value to write to Auxport
Dim DomeDia               ' diameter of dome in centimeters
Dim fnamePres As String * 50    ' file neame for raw data with ext .dat
Dim FOutDivider&
Dim Freq&(10)
Dim i, k, j, N As Integer                  ' counter
Dim MemHandle&            ' define a variable to contain the handle for
                     ' memory allocated by Windows through cbWinBufAlloc%()
Public CycleTime
Dim SampleTime, startpause
Dim SigSource(8)
Dim ULStat As Long              'universal library styatistic return
Dim site As String * 64 ' a descriptive string describing the site
Dim CallStop, mytime
Dim ActualCounts%(8)
Dim Logi1, Logi2 As Boolean


Private Sub cmdStart_Click()
'********
' Determine which device Site 1 or Site 2
'********
If Check1 And Check2 Then
   MsgBox ("Check only one box")
   Call CmdStop_Click
End If
Logi1 = Check1
Logi2 = Check2
'MsgBox (Logi1)
'MsgBox (Logi2)
'MsgBox (Logi1 + Logi2)
If Logi1 + Logi2 = 0 Then
   MsgBox ("Check one site")
   Call CmdStop_Click
End If
If Check1 Then BoardNum% = 0
If Check2 Then BoardNum% = 1
```

```
'*******
' Turn off heater turn on power supply
'*******

  ULStat = cbDBitOut(BoardNum%, AUXPORT, 0, 1) ' turn on power supply
  If ULStat <> 0 Then Stop
  ULStat = cbDBitOut(BoardNum%, AUXPORT, 1, 0) ' turn off heater
  If ULStat <> 0 Then Stop
'   MsgBox (BoardNum%)

'*********
' Initialize signal sources
' There are five inputs on each chip so use chip 2 for the second 5 signal sources
'*********
SigSource(8) = CTRINPUT8       ' Input 8 chip 2 Pressure transducer

'*******
' Collect Screen Input Data
'*******
MemHandle& = cbWinBufAlloc(100)
If MemHandle& = 0 Then Stop

site = CStr(txtSite.Text)          ' site description
''******
'write baseline, heat-pulse duration, SerialNo, and Site description raw data file #1
'******
Date = Now
Time = Now
fnamePres = "c:\temp\" + Format$(Date, "mmm dd ") + Format$(Time, "hh mm ss") + "Pres" + ".dat"
Open fnamePres For Output As #1
   Write #1, Format(Now), Format(site)
Close #1

MemHandle& = cbWinBufAlloc(100)
If MemHandle& = 0 Then Stop

   ULStat = cbDBitOut(BoardNum%, AUXPORT, 0, 1) ' turn on power supply
   If ULStat <> 0 Then Stop
   Screen.MousePointer = 0     'Default Pointer
   Refresh
   Date = Now
   Time = Now
   timezero = Timer
   t = 0
   k = 0
Do While CallStop <> -1         'main loop
      'calculate accmulated time
      If Timer < timezero Then   'if cross midnight
         t = 86400 - timezero + Timer
      Else
         t = Timer - timezero
      End If
      '*******
```

```
    'control when to activate digital output for rempte power supply and heater
    '*******
    DataSet     ' Collect a data point
    Open fnamePres For Append As #1
      'write data in frequency
      Write #1, Format(Now), Format(t, "#0.00000"), _
        Format(Freq&(8), "#0")
    Close #1
    '*******
    'display data in frequency
    '*******

      lblADData.Caption = Format(Freq&(8), "#0")
      PlotData (fnamePres)
    Refresh

    DoEvents

    If CallStop = -1 Then GoTo 44
  Loop     'Loop 11
44  End Sub
Private Sub CmdStop_Click()
  CallStop = -1
  ULStat = cbDBitOut(0, AUXPORT, 0, 0) ' turn off power supply
  If ULStat <> 0 Then Stop
  ULStat = cbDBitOut(0, AUXPORT, 1, 0) ' turn off heater
  If ULStat <> 0 Then Stop
  ULStat = cbDBitOut(1, AUXPORT, 0, 0) ' turn off power supply
  If ULStat <> 0 Then Stop
  ULStat = cbDBitOut(1, AUXPORT, 1, 0) ' turn off heater
  If ULStat <> 0 Then Stop
  Unload Me
End Sub


Private Sub DataSet()
  Init1UL
  Init2UL
  i = 8
    ULStat& = cbCFreqIn(BoardNum%, SigSource(i), GateInterval1&, CBCount%(i), Freq&(i))
    If ULStat& <> 0 Then Stop

    If CBCount%(i) < 0 Then
      ActualCounts%(i) = 65535 + CBCount%(i)
    Else
      ActualCounts%(i) = CBCount%(i)
    End If


End Sub
Private Sub Init1UL()
  ' declare revision level of Universal Library
  ULStat& = cbDeclareRevision(CURRENTREVNUM)

  ' Initiate error handling
  '  activating error handling will trap errors like
  '  bad channel numbers and non-configured conditions.
  '  Parameters:
  '    PRINTALL    :all warnings and errors encountered will be printed
  '    DONTSTOP    :if an error is encountered, the program will not stop,
  '                errors must be handled locally

  ULStat& = cbErrHandling(PRINTALL, DONTSTOP)
  If ULStat& <> 0 Then Stop

  ' If cbErrHandling& is set for STOPALL or STOPFATAL during the program
  ' design stage, Visual Basic will be unloaded when an error is encountered.
  ' We suggest trapping errors locally until the program is ready for compiling
  ' to avoid losing unsaved data during program design.  This can be done by
  ' setting cbErrHandling options as above and checking the value of ULStat%
  ' after a call to the library. If it is not equal to 0, an error has occurred.


  ' Initialize the board level features
  '  Parameters:
  '    BoardNum&    :the number used by CBCONFIG to describe this board
  '    ChipNum&    :chip to be initialized (1 for CTR5, 1 or 2 for CTR10)
  '    FOutDivider&:the F-Out divider (0-15)
  '    Source&     :the signal source for F-Out
  '    Compare1&   :status of comparator 1
  '    Compare2&   :status of comparator 2
  '    TimeOfDay&  :time of day mode control
  ChipNum% = 1
  FOutDivider& = 1        ' sets up OSC OUT for 10kHz signal which can
  Source& = FREQ1         ' be used as frequency source for this example
  Compare1& = Disabled
  Compare2& = Disabled
  TimeOfDay& = Disabled

  ULStat& = cbC9513Init(BoardNum%, ChipNum%, FOutDivider&, Source&, Compare1&, Compare2&,
TimeOfDay&)
  If ULStat& <> 0 Then Stop
End Sub
Private Sub Init2UL()

  ' Initialize the board level features
  '  Parameters:
  '    BoardNum&    :the number used by CBCONFIG to describe this board
  '    ChipNum&    :chip to be initialized (1 for CTR5, 1 or 2 for CTR10)
  '    FOutDivider&:the F-Out divider (0-15)
  '    Source&     :the signal source for F-Out
  '    Compare1&   :status of comparator 1
  '    Compare2&   :status of comparator 2
  '    TimeOfDay&  :time of day mode control
  ChipNum% = 2
```

```
'   FOutDivider& = 1         ' sets up OSC OUT for 10kHz signal which can
'   Source& = FREQ1          ' be used as frequency source for this example
'   Compare1& = Disabled
'   Compare2& = Disabled
'   TimeOfDay& = Disabled
  Source& = FREQ1          ' be used as frequency source for this example

  ULStat& = cbC9513Init(BoardNum%, ChipNum%, FOutDivider&, Source&, Compare1&, Compare2&,
TimeOfDay&)
  If ULStat& <> 0 Then Stop
End Sub
Private Sub PlotData(fnamePres)
'   Screen.MousePointer = 11
  Dim textline
  Dim time0() As String
  Dim time1 As String 'used to store time file created needed to determine age of file and be compatable with
older data

   Dim chb()   'dynamic array to hold analog byte data stored on .dat file
   Dim ch() As Double   'dynamic array to hold double precision data from .dat file
   Dim tb()           ' dynamic array to hold elapsed time data in byte format
   Dim t()  As Double  'dynamic array to hold elapsed time data
  Dim ct As Integer    '        , pt_odd, pt_half As Integer
  Dim tmax(6) As Double
  Dim tempmax(6) As Double
  Dim SerNumber
'   Dim tail
  Dim N, j, i, l, m, g, k, z, ii, jj, kk As Integer

  Dim site As String * 64
   Dim dat_type As Boolean
'*********
'Determine the number of records in the data file
'*********
ChDir ("C:\Temp")

Open fnamePres For Input As #1
   N1 = 0
   N = 0
   Do While Not EOF(1)
      Line Input #1, textline
      N = N + 1    'n is the number of records or values first redord is header not data
      N1 = N1 + 1  'N1 is the number of records or values do not reset this counter first recored is header not data
   Loop
   N = N - 1
   N1 = N1 - 1

Close #1
'********
'Allocate space to hold arrays
'********
ReDim time0(N + 1) As String '+1?
ReDim t(N) As Double
ReDim tb(N)
ReDim ch(N1) As Double
ReDim chb(N1)

Open fnamePres For Input As #5
   j = 0
   k = 0
   Do While Not EOF(5)
      If j = 0 Then
         Input #5, time0(j), site   ' xxx = tail skiped
      End If
      j = j + 1
      k = k + 1
      Input #5, time0(j), tb(j), chb(j)
   Loop
Close #5
For ii = 1 To k
   ch(ii) = CDbl(chb(ii))    'convert byte data to double precision
Next ii
For ii = 1 To k
   t(ii) = CDbl(tb(ii))    ' convert byte data to double precision
Next ii

      '********
      ' set up arrays for ploting
      '********

200    Dim Graph() As Single
      Dim x As Integer
      Dim myXarray() As Double         'Time
      Dim myYArray0() As Double

      ReDim myXarray(N)
      ReDim myYArray0(N)

      '*******
      'Generate some x y data.
      '*******

      myXarray(0) = t(1)
      myYArray0(0) = ch(1)

         'MsgBox N
      For x = 1 To N - 1

         myXarray(x) = t(x)             'value for X-axis
         myYArray0(x) = ch(x)         'value for Y-axis Ref Temp

      Next x


      With TChart1
         .AddSeries scPoint
      .Series(0).AddArray UBound(myYArray0), myYArray0(), myXarray() ' ref temp
      .Series(0).XValues.TempValue = True
    End With
    Refresh
    DoEvents
      'MsgBox ("check 4")
      '  Screen.MousePointer = 1

45 End Sub
```

frmMain

```
Option Explicit

Private Sub MDIForm_Load()
   frmMain.Show
   frmMain.AutoShowChildren = True
   frmDataAcquisition.Show
   'frmDataDisplay.Show
   'frmDifferentialPres.Show
   frmDataAcquisition.Initialize = True

End Sub

Private Sub mnuFOpen_Click()

   dlgCommon.FILTER = "All Files (*.*)|*.*|Data Files(*.dat)|*.dat"
   dlgCommon.ShowOpen
   On Error GoTo errorhandler
      If FileLen(dlgCommon.Filename) > 65000 Then
      MsgBox "The file is too large to open."
      Exit Sub
      End If
   'frmDataDisplay.Show
   frmDifferentialPres.Show
   OpenFile (dlgCommon.Filename)
errorhandler:     Exit Sub
End Sub

Private Sub mnuHAbout_Click()
   MsgBox "Flux Meter Data Acquisition Program" _
   & vbCrLf _
   & vbCrLf & "Beta Version 5.01" _
   & vbCrLf _
   & vbCrLf & "Carl G Enfield" _
   & vbCrLf & "Enfield Environmental Research" _
   & vbCrLf & "5017 Park Ridge Ct" _
   & vbCrLf & "West Chester, OH 45069, USA" _
   & vbCrLf _
   & vbCrLf & "Bob K Lien" _
   & vbCrLf & "U.S. EPA" _
   & vbCrLf & "NRMRL" _
   & vbCrLf & "Cincinnati, Ohio" _
   , vbInformation
End Sub


Private Sub mnuMAInScan_Click()
   frmDataAcquisition.Show
   frmDataAcquisition.SetFocus
End Sub
```

```
Private Sub mnuMReference_Click()
   MsgBox "Measurement Computing. Universal Driver Library: User's Guide" _
   & vbCrLf
End Sub

Private Sub mnuTAnalysis_Click()

   frmDataAnalysis.Show
End Sub

Private Sub mnuTDisplay_Click()
   frmSignalDisplay.Show
End Sub


Private Sub mnuTPres_Click()

    frmDifferentialPres.Show
End Sub

Private Sub mnuWArrangeIcons_Click()
   Me.Arrange vbArrangeIcons
End Sub

Private Sub mnuWCascade_Click()
   Me.Arrange vbCascade
End Sub

Private Sub mnuWTileHorizontal_Click()
   Me.Arrange vbTileHorizontal
End Sub

Private Sub mnuWTileVertical_Click()
   Me.Arrange vbTileVertical
End Sub


Private Sub mnuFClose_Click()
   On Error GoTo errorhandler
   ActiveForm.CmdClose.Value = True
errorhandler:    Exit Sub
End Sub

Private Sub mnuFExit_Click()
   Unload Me
End Sub

Private Sub mnuFPrint_Click()

   Dim NumCopies, i
   dlgCommon.CancelError = True
   On Error GoTo errorhandler
   If ActiveForm Is Nothing Then Exit Sub
   dlgCommon.Flags = 0
   dlgCommon.ShowPrinter

   NumCopies = dlgCommon.Copies
   For i = 1 To NumCopies
      Screen.ActiveForm.PrintForm
   Next i

errorhandler:     Exit Sub
End Sub

Private Sub mnuFPrintSetup_Click()
   dlgCommon.CancelError = False
   dlgCommon.Flags = cdlPDPrintSetup
   dlgCommon.ShowPrinter
End Sub

Sub OpenFile(Filename)

    Dim fIndex As Integer

   On Error Resume Next
   ' Open the selected file.
   Open Filename For Input As #1
   If Err Then
      MsgBox "Can't open file: " + Filename
      Exit Sub
   End If
   ' Change the mouse pointer to an hourglass.
   Screen.MousePointer = 11

   ' Change the form's caption and display the new text.
   'frmDataDisplay.Caption = UCase(FileName)
   'frmDataDisplay.txtNote.Text = StrConv(InputB(LOF(1), 1), vbUnicode)

   Close #1
   ' Reset the mouse pointer.
   Screen.MousePointer = 0
End Sub
```

frmSignalDisplay

```
'last revised 05/19/08 by Carl Enfield
Option Explicit
Private Sub CmdClose_Click()
   Unload Me
End Sub

Private Sub Dir1_Change()
   File1.Path = Dir1
End Sub

Private Sub Form_Load()
  Dir1.Path = "c:\temp"
  File1.Path = Dir1
End Sub

Private Sub Command1_Click() '  Display Information

  Screen.MousePointer = 11
  Dim textline
  Dim time0() As String
  Dim timea As String
  Dim ch7, ch8, ch9, ch10 As Double
  Dim time1 As String 'used to store time file created needed to determine age of file and be compatable with
older data
   Dim chb()   'dynamic array to hold analog byte data stored on .dat file
   Dim ch() As Double    'dynamic array to hold analog data stored on .dat file
   Dim t() As Double   'dynamic array to hold elapsed time data
   Dim tb()    'dynamic array to hold byte elapsed time data
  Dim ct As Integer    '        , pt_odd, pt_half As Integer
  Dim tmax(6) As Double
  Dim tempmax(6)
  Dim bline
  Dim heater
  Dim SerNumber
  Dim tail
  Dim N, N1, j, i, l, m, g, k, z, ii, jj, kk As Integer
   Const GateInterval& = 200
   Const GateInterval1& = 1000
  Dim CalResistor
  Dim Resistor()
  Dim DomeDia
  Dim site As String * 64
   Dim Q0, Q1, Q3, Q4, Q0min, Q1min, Q3min, Q4min, Q0max, Q1max, Q3max, Q4max
   Dim V0, V1, V3, V4
   Dim dat_type As Boolean
   Dim sumx(6) As Double
   Dim sumy(6) As Double
   Dim sumxx(6) As Double
   Dim sumxy(6) As Double
   Dim avgx(6) As Double
```

```
    Dim avgy(6) As Double
    Dim slope(6) As Double
    Dim sInt(6) As Double
    Dim File2 As String

   ChDir Dir1
'**********
' Initialize all of the calculated outputs other than peak arrival time to "N.A."
'       to make sure only recomputed values are displayed  if this isn't done
'       it is possibe to display a value from a previous data set
'**********
        lblMom2.Caption = Format("N.A.")
        lblMom3.Caption = Format("N.A.")
        lblMom5.Caption = Format("N.A.")
        lblMom6.Caption = Format("N.A.")
        Q0 = "N.A."
        Q0min = "N.A."
        Q0max = "N.A."
        lblQ0.Caption = Format(Q0)
        lblQ0min.Caption = Format(Q0min)
        lblQ0max.Caption = Format(Q0max)
        Q1 = "N.A."
        Q1min = "N.A."
        Q1max = "N.A."
        lblQ1.Caption = Format(Q1)
        lblQ1min.Caption = Format(Q1min)
        lblQ1max.Caption = Format(Q1max)
        Q3 = "N.A."
        Q3min = "N.A."
        Q3max = "N.A."
        lblQ3.Caption = Format(Q3)
        lblQ3min.Caption = Format(Q3min)
        lblQ3max.Caption = Format(Q3max)
        Q4 = "N.A."
        Q4min = "N.A."
        Q4max = "N.A."
        lblQ4.Caption = Format(Q4)
        lblQ4min.Caption = Format(Q4min)
        lblQ4max.Caption = Format(Q4max)
    V0 = "N.A."
    V1 = "N.A."
    V3 = "N.A."
    V4 = "N.A."
    lblV0.Caption = Format(V0)
    lblV1.Caption = Format(V1)
    lblV3.Caption = Format(V3)
    lblV4.Caption = Format(V4)

'******
' Import calibration data
'******
ChDir ("C:\Program Files\FluxMeter\")
'******
```

```
' determine the number of calibration curves in the data table
'******
Open "Calibration_Data(2).csv" For Input As #6
   l = 0
   Do While Not EOF(6)
      Line Input #6, textline
      l = l + 1    'l is the number of calibration curves
   Loop
Close #6
'MsgBox (l)
'********
'Allocate space to hold arrays place calibration data into arrays
'********

ReDim Resistor(l), c10(l), c20(l), c30(l), cv0(l), c11(l), c21(l), _
   c31(l), cv1(l), c13(l), c23(l), c33(l), cv3(l), c14(l), c24(l), _
   c34(l), cv4(l), SerNo(l), Tslope(l), Tint(l), TSn1(l), ch1a(l), _
   ch1b(l), ch1c(l), TSn2(l), ch2a(l), ch2b(l), ch2c(l), TSn3(l), _
   ch3a(l), ch3b(l), ch3c(l), TSn4(l), ch4a(l), Ch4b(l), Ch4c(l), _
   TSn5(l), ch5a(l), Ch5b(l), Ch5c(l), TSn6(l), Ch6a(l), Ch6b(l), _
   Ch6c(l), TSn7(l), Ch7a(l), Ch7b(l), Ch7c(l)
'*******
' Import calibration data
'*******
Open "Calibration_Data(2).csv" For Input As #7
   l = 1
   Do While Not EOF(7)
      '    MsgBox (l)
      Input #7, Resistor(l), c10(l), c20(l), c30(l), cv0(l), c11(l), c21(l), _
         c31(l), cv1(l), c13(l), c23(l), c33(l), cv3(l), c14(l), c24(l), _
         c34(l), cv4(l), SerNo(l), Tslope(l), Tint(l), TSn1(l), ch1a(l), _
         ch1b(l), ch1c(l), TSn2(l), ch2a(l), ch2b(l), ch2c(l), TSn3(l), _
         ch3a(l), ch3b(l), ch3c(l), TSn4(l), ch4a(l), Ch4b(l), Ch4c(l), _
         TSn5(l), ch5a(l), Ch5b(l), Ch5c(l), TSn6(l), Ch6a(l), Ch6b(l), _
         Ch6c(l), TSn7(l), Ch7a(l), Ch7b(l), Ch7c(l)
      l = l + 1
   Loop
      '    MsgBox (l)
Close #7
ChDir Dir1
  '*********
  'Determine the number of records in the data file
  '*********

Open File1 For Input As #1
   N1 = 0
   N = 0
   Do While Not EOF(1)
      Line Input #1, textline
      N = N + 1    'n is the number of records or values
      N1 = N1 + 1  'N1 is the number of records or values do not reset this counter
   Loop
   N = N - 1
   N1 = N1 - 1
Close #1
'   MsgBox ("Number of points = " & N1)
If (N1 - 20 < 0) Then
   MsgBox "Not enough data points.  Please check file."
   GoTo 44
End If
  '********
  'Allocate space to hold arrays
  '********
j = 6       'j is the number of channels of data in the raw data file

ReDim time0(N + 1) As String '+1?
ReDim t(N1)
ReDim tb(N1)
ReDim ch(j, N1) As Double
ReDim chb(j, N1)

Open File1 For Input As #5
   j = 0
   N = 0
   Do While Not EOF(5)
      If j = 0 Then
         Input #5, time0(j), bline, heater, tail, SerNumber, DomeDia, site
      End If
      j = j + 1
      N = N + 1
      Input #5, time0(j), tb(j), chb(1, j), chb(2, j), chb(3, j), chb(4, j), chb(5, j), chb(6, j)
      t(j) = CDbl(tb(j))
      ch(1, j) = CDbl(chb(1, j))
      ch(2, j) = CDbl(chb(2, j))
      ch(3, j) = CDbl(chb(3, j))
      ch(4, j) = CDbl(chb(4, j))
      ch(5, j) = CDbl(chb(5, j))
      ch(6, j) = CDbl(chb(6, j))
      'MsgBox ("t(j) = " & t(j))
   Loop
Close #5

'For i = 1 To 6
'   For ii = 1 To k
'      ch(i, ii) = CDbl(chb(i, ii))    'convert byte data to double precision
'      MsgBox ("ch(i,ii) = " & ch(i, ii))
'   Next ii
'Next i
'For ii = 1 To k
'   t(ii) = CDbl(tb(ii))    ' convert byte data to double precision
'   MsgBox ("t(ii) = " & t(ii))
'Next ii
lblSiteD.Caption = site
      'MsgBox (ch(1, 1))
'13  j = j - 1
   '********
   ' Find serial Number
   '********
   For i = 2 To l - 1
      If SerNumber - SerNo(i) = 0 Then GoTo 100
   Next i
   If i = l Then SerNo(i - 1) = "default"  ' Serial number not detected
   i = i - 1
100 lblSerNo.Caption = SerNo(i)
   ii = i ' ii is used as the record number of the for the calibration data
'   MsgBox (ii)
'******
'  Convert Data to temperatures
'******
Dim temp     '
ReDim Res(6, N1)
For i = 1 To N1   'N1 is the number of data records
   Res(1, i) = ch(1, i)
   Res(2, i) = Res(1, i) - (ch(2, i) / 10)
   Res(3, i) = Res(1, i) - (ch(3, i) / 10)
   Res(4, i) = Res(1, i) - (ch(4, i) / 10)
   Res(5, i) = Res(1, i) - (ch(5, i) / 10)
   Res(6, i) = Res(1, i) - (ch(6, i) / 10)
   Res(1, i) = ch(1, i) - 1820
Next i
'  Now convert to temperature and return as the original variable
For i = 1 To N1

   If Res(1, i) > 0 Then
   temp = ch1a(ii) + ch1b(ii) * Log(Res(1, i) / 10000)
   temp = temp + ch1c(ii) * ((Log(Res(1, i) / 10000)) ^ 3)
   ch(1, i) = (1 / temp) - 273.15             ' 273.15 convert from Kelvin to Celsus
   Else: ch(1, i) = 0
   End If

   If Res(2, i) > 0 Then
   temp = ch2a(ii) + ch2b(ii) * Log(Res(2, i) / 10000)
   temp = temp + ch2c(ii) * ((Log(Res(2, i) / 10000)) ^ 3)
   ch(2, i) = (1 / temp) - 273.15
   Else: ch(2, i) = 0
   End If

   If Res(3, i) > 0 Then
   temp = ch3a(ii) + ch3b(ii) * Log(Res(3, i) / 10000)
   temp = temp + ch3c(ii) * ((Log(Res(3, i) / 10000)) ^ 3)
   ch(3, i) = (1 / temp) - 273.15
   Else: ch(3, i) = 0
   End If

   If Res(4, i) > 0 Then
   temp = ch4a(ii) + Ch4b(ii) * Log(Res(4, i) / 10000)
   temp = temp + Ch4c(ii) * ((Log(Res(4, i) / 10000)) ^ 3)
   ch(4, i) = (1 / temp) - 273.15
   Else: ch(4, i) = 0
   End If

   If Res(5, i) > 0 Then
   temp = ch5a(ii) + Ch5b(ii) * Log(Res(5, i) / 10000)
   temp = temp + Ch5c(ii) * ((Log(Res(5, i) / 10000)) ^ 3)
   ch(5, i) = (1 / temp) - 273.15
   Else: ch(5, i) = 0
   End If

   If Res(6, i) > 0 Then
   temp = Ch6a(ii) + Ch6b(ii) * Log(Res(6, i) / 10000)
   temp = temp + Ch6c(ii) * ((Log(Res(6, i) / 10000)) ^ 3)
   ch(6, i) = (1 / temp) - 273.15
   Else: ch(6, i) = 0
   End If


Next i
'*********
' get supplemental data
'*********
  File2 = Replace(File1, ".dat", "S.dat")
'  MsgBox (File2)
Open File2 For Input As #1
      Input #1, timea, ch7, ch8, ch9, ch10
Close #1
   If ch7 > 0 Then
      temp = Ch7a(ii) + Ch7b(ii) * Log(ch7 / 10000)
      temp = temp + Ch7c(ii) * ((Log(ch7 / 10000)) ^ 3)
      ch7 = (1 / temp) - 273.15            ' 273.15 convert from Kelvin to Celsus
   Else: ch7 = 0
   End If
'********
' Convert suplimental frequencies from ch8 through ch 10 to voltages
'********
ch8 = ch8 / 10000
ch9 = ch9 / 10000
ch10 = ch10 / 10000
'******
' Determine if sufficient data for possible analysis
'******

14  If (t(N1 - 1) - bline * 60 < 0) Then
      MsgBox ("t(N1-1) = " & t(N1 - 1))
      MsgBox "Not enough data points.  Check Initial Parameters."
      GoTo 44
   End If
'******
' Convert temperatures from actual temperature in deg C to differential temperature in Deg C
'******
For i = 1 To N1
   For jj = 2 To 6
      ch(jj, i) = ch(jj, i) - ch(1, i)
   Next jj
Next i

'******
' Analyze base line data and determine slope and intercept of the base line
'******
jj = 0
Dim offset
offset = bline * 60 + ((5 * GateInterval& / 1000) + GateInterval1& / 1000) ' Note:
         '   GateInterval is in milliseconds calculations are in seconds
Do
   jj = jj + 1     '  jj is sample number before heater is turned on
Loop Until (t(jj) - offset) >= 0
   jj = jj - 2

      'calculate sums used to calculate baseline averages, slopes, and intercepts
   For j = 1 To 6
      sumx(j) = 0
      sumy(j) = 0
      sumxy(j) = 0
      sumxx(j) = 0
   Next j
         '    Determind average values
   For ct = 1 To jj
      For j = 1 To 6
         sumx(j) = sumx(j) + t(ct)
         sumy(j) = sumy(j) + ch(j, ct)
      Next j
   Next ct
      '   ct = ct + 1
      '   Loop Until (t(ct) - bline * 60) > 0

   For j = 1 To 6
      avgx(j) = sumx(j) / jj
      avgy(j) = sumy(j) / jj
   Next j
   Dim val1 As Double
   Dim val2 As Double

For ct = 1 To jj
   For j = 1 To 6
      val1 = (t(ct) - avgx(j)) * (ch(j, ct) - avgy(j))
      val2 = (t(ct) - avgx(j)) ^ 2
      sumxy(j) = sumxy(j) + val1
      sumxx(j) = sumxx(j) + val2
   Next j
Next ct
'*********
'  Calculate Slopes and Intercepts
'*********
For j = 1 To 6
   slope(j) = sumxy(j) / sumxx(j)
   sInt(j) = avgy(j) - (slope(j) * avgx(j))
```

```
Next j
     lblTemp.Caption = Format(avgy(1), "#0.0")
     LblBLSlope.Caption = Format(slope(1) * 3600, "0.0E-00")
     LblBLInt.Caption = Format(sInt(1), "#0.0")
      '*********
      'Correct for baseline drift
      '*********
 For i = 1 To N
    For j = 1 To 6
      ch(j, i) = ch(j, i) - (slope(j) * t(i) + sInt(j))
    Next j
 Next i

'********
' find time at maximum signal
' the time is calculated as max time - base line time - (pulse time/2)
' This will be one of the metrics evaluated to determine travel time
'********
Dim jmax(6) As Integer  'sample number at max temperature

  For i = 1 To 6
    tempmax(i) = ch(i, jj + 1)
  Next i
  For i = 1 To 6
    For j = jj + 2 To N
      If (ch(i, j) - tempmax(i) > 0) Then
        tempmax(i) = ch(i, j)
        jmax(i) = j
      End If
    Next j
  Next i
    tmax(1) = t(jmax(1)) + GateInterval1& / 2000
  For i = 2 To 6
      If tempmax(i) < 0.075 Then  ' 0.075 considered minimum peak to be used eliminates some of the noise
        tmax(i) = bline * 60 + (heater / 2)

      Else: tmax(i) = t(jmax(i)) + (GateInterval& / 1000 * (i - 1)) + GateInterval& / 2000 + GateInterval1& / 1000
      End If
  Next i

' display the peak arrival time
     lbltmax1.Caption = Format(tmax(2) - (bline * 60 + heater / 2), "#0.00")
     lbltmax2.Caption = Format(tmax(3) - (bline * 60 + heater / 2), "#0.00")
     lbltmax3.Caption = Format(tmax(4) - (bline * 60 + heater / 2), "#0.00")
     lbltmax4.Caption = Format(tmax(5) - (bline * 60 + heater / 2), "#0.00")
     lbltmax5.Caption = Format(tmax(6) - (bline * 60 + heater / 2), "#0.00")
'********
'Begin Moment analysis - Data will be extrapolated based on a lograthmetic
' function based on a sum of squares fit to the data between 80% of tempmax
' and next occurance of <= 0.04 deg C or the end of file which ever comes first
' first determine sample number correlated to 80%of tempmax to perform moment analysis
' there should be only one peak and it should occure in the first 40% of the data set
' if peak temperature occures after the 40% of the data points do not make moment analysis
```

```
'********
Dim jbegin(6) As Integer   'jbegin is the sample number for the first sample to use in generating fit
Dim jend(6) As Integer     'jend is the last sample to use
    If ((0.4 * N1) - jmax(1) < 0) And ((0.4 * N1) - jmax(2) < 0) And ((0.4 * N1) - jmax(3) < 0) _
      And ((0.4 * N1) - jmax(4) < 0) And ((0.4 * N1) - jmax(5) < 0) Then
      lblMom2.Caption = Format("N.A.")
      lblMom3.Caption = Format("N.A.")
      lblMom5.Caption = Format("N.A.")
      lblMom6.Caption = Format("N.A.")
      GoTo 200
    End If
  For i = 2 To 6
      If (jmax(i) + 20 - N1) > 0 Then     'See if there is sufficient data to calculate moments
        GoTo 90 ' changed from 200 2/24/08
      End If
    j = jmax(i)

  Do Until ((tempmax(i) * 0.8) - ch(i, j) > 0) 'determine sample number for first sample
                           ' to use in enterpolating and extrapolating
                           ' moments 80% of peak was qualitatively
                           ' estimated as being sufficient to be away
                           ' from the curvature at the peek
    j = j + 1
    If (j - N = 0) Then
      jbegin(i) = j
      jend(i) = j
      GoTo 90
    End If
'      jend(i) = jend(i) + 1
  Loop
    jbegin(i) = j
      If (jbegin(i) + 20 - N1) > 0 Then
        GoTo 90
      End If
    j = j + 1
    If j >= N1 Then GoTo 89
  Do Until (0.05 - ch(i, j) > 0) 'determine last sample number considered valid for
                     'interpolating or extrapolating moments noise was measured
                     'at +- 2 mv.  data has already been adjusted for baseline
                     'offset

    j = j + 1
    If j >= N1 Then GoTo 89
  Loop
89     jend(i) = j - 1
90   Next i
'********
' Fit log transformed data to a streight line first determine if there is enough data
' there should be a minimum of minobs observations to calculate the extrapolation function
'********
Dim sumyy(6)
Dim avgxx(6)
Dim avgyy(6)
Dim Mslope(6) As Double
Dim Mint(6) As Double
Dim minobs As Integer   'minimum number of observations between jbegin(i) and jend(i)
             'to permit moment analysis 20 is considered minimm
Dim minsignal As Single 'minimum peak signal in volts for analysis
minobs = 20
minsignal = 0.05

For i = 2 To 6  'initialize values
  sumxx(i) = 0
  sumyy(i) = 0
  sumxy(i) = 0
  avgxx(i) = 0
  avgyy(i) = 0
  Mslope(i) = 0
  Mint(i) = 0
Next i
For i = 2 To 6
 If ((jend(i) - jbegin(i) - minobs) > 0) And (tempmax(i) - minsignal > 0) Then
      For j = jbegin(i) To (jend(i) - 1)
        kk = j + 1
        avgxx(i) = avgxx(i) + t(kk)
        If ch(i, kk) < 0 Then ch(i, kk) = 1E-50
        avgyy(i) = avgyy(i) + Log(ch(i, kk))
      Next j
      avgxx(i) = avgxx(i) / (jend(i) - jbegin(i))
      avgyy(i) = avgyy(i) / (jend(i) - jbegin(i))
 End If
Next i
For i = 2 To 6
 If ((jend(i) - jbegin(i) - minobs) > 0) And (tempmax(i) - minsignal > 0) Then
      For j = jbegin(i) To (jend(i) - 1)
        kk = j + 1
        If ch(i, kk) < 0 Then ch(i, kk) = 1E-50
        sumxx(i) = sumxx(i) + (t(j) - avgxx(i)) ^ 2
        sumyy(i) = sumyy(i) + (Log(ch(i, kk)) - avgyy(i)) ^ 2
        sumxy(i) = sumxy(i) + (t(kk) - avgxx(i)) * (Log(ch(i, kk)) - avgyy(i))
      Next j
    Mslope(i) = sumxy(i) / sumxx(i)
    Mint(i) = avgyy(i) - (Mslope(i) * avgxx(i))
 End If
Next i
'*********
'Calculate zero and first moments
'*********
Dim Momzero(6)
Dim Mom1(6)
Dim c1, c2
Dim T1, T2
Dim timecor 'time correction such that zero time is at the end of base line measurements
Dim jstart 'sample number to begin moment calculations this is to remove switching noise
timecor = bline * 60
For i = 2 To 6
```

```
  If ((jend(i) - jbegin(i) - minobs) < 0) Or (tempmax(i) - minsignal < 0 Or Mslope(i) > 0) Then
    Mom1(i) = "N.A."
'    GoTo 300
  Else
    Momzero(i) = 0
    Mom1(i) = 0
    ch(i, 0) = 0
    t(0) = 0
'*********
'   Start moment analysis beginning with sample after heat-pulse
'   Correct time scale to begin half way through the heat-pulse in moment calculations
'   First step is to determine sample number after end of heat-pulse
'*********
    j = 1
Do Until (t(j) - bline * 60 - heater / 2) > 0
    j = j + 1
Loop
    jstart = j
    For j = jstart To jbegin(i)          'first part of curve
        Momzero(i) = Momzero(i) + ((t(j) - t(j - 1)) * (ch(i, j) + ch(i, j - 1))) / 2
        Mom1(i) = Mom1(i) + (1 / 6) * ((ch(i, j - 1) * (t(j - 1) - timecor) + ch(i, j) * _
            (t(j) - timecor)) * (t(j) - t(j - 1)) + (ch(i, j - 1) _
            + ch(i, j)) * ((t(j) - timecor) * (t(j) - timecor) - (t(j - 1) - timecor) * _
            (t(j - 1) - timecor)))
    Next j
            'end of first part of curve begin analysis for the curve fit portion of the curve
        c1 = ch(i, j)
        T1 = t(j)
        j = 1
    Do Until (c1 - 0.001 * tempmax(i) < 0)
        T2 = T1 + 1
        c2 = Exp(Mslope(i) * (T2) + Mint(i))
        Momzero(i) = Momzero(i) + (c1 + c2) / 2
        Mom1(i) = Mom1(i) + (1 / 6) * ((c1 * (T1 - timecor) + c2 * (T2 - timecor)) * _
            (T2 - T1) + _
            (c1 + c2) * ((T2 - timecor) * (T2 - timecor) - (T1 - timecor) * (T1 - timecor)))
        T1 = T2
        c1 = c2
    Loop
  End If

300 Next i

'MsgBox (Mom1(1))

  If ((jend(2) - jbegin(2)) > minobs) And (tempmax(2) - minsignal > 0 And IsNumeric(Mom1(2))) Then
    lblMom2.Caption = Format(Mom1(2) / Momzero(2), "#0.0")
  Else
    lblMom2.Caption = Format("N.A.")
  End If
  If ((jend(3) - jbegin(3)) > minobs) And (Not Not tempmax(3) - minsignal > 0 And IsNumeric(Mom1(3))) Then
    lblMom3.Caption = Format(Mom1(3) / Momzero(3), "#0.0")
  Else
  lblMom3.Caption = Format("N.A.")
 End If
 If ((jend(5) - jbegin(5)) > minobs) And (tempmax(5) - minsignal > 0 And IsNumeric(Mom1(5))) Then
  lblMom5.Caption = Format(Mom1(5) / Momzero(5), "#0.0")
 Else
  lblMom5.Caption = Format("N.A.")
 End If
 If ((jend(6) - jbegin(6)) > minobs) And (tempmax(6) - minsignal > 0 And IsNumeric(Mom1(6))) Then
  lblMom6.Caption = Format(Mom1(6) / Momzero(6), "#0.0")
 Else
  lblMom6.Caption = Format("N.A.")
 End If

'*********
' generate labels for flow
'*********
Dim CL
Dim maxtime
Dim CLV
CL = 2.326
'  Channel 2 Captions
  If ((jend(2) - jbegin(2) > minobs) And (tempmax(2) - minsignal > 0) And _
      (tmax(2) - bline * 60 - 1.5 * heater > 0) And IsNumeric(Mom1(2))) Then
      maxtime = tmax(2) - (bline * 60 + heater / 2)
      CLV = CL * cv0(ii)
    Q0 = (c10(ii) - (c30(ii) * maxtime)) / (maxtime + c20(ii))
    Q0min = (c10(ii) - (c30(ii) * maxtime * (1 + CLV))) / _
      (maxtime * (1 + CLV) + c20(ii))
    Q0max = (c10(ii) - (c30(ii) * maxtime * (1 - CLV))) / _
      (maxtime * (1 - CLV) + c20(ii))
    lblQ0.Caption = Format(Q0, "#0.00")
    lblQ0max.Caption = Format(Q0max, "#0.00")
    lblQ0min.Caption = Format(Q0min, "#0.00")
  Else
    Q0 = "N.A."
    Q0min = "N.A."
    Q0max = "N.A."
    lblQ0.Caption = Format(Q0)
    lblQ0min.Caption = Format(Q0min)
    lblQ0max.Caption = Format(Q0max)
  End If
'  Channel 3 Captions
  If (jend(3) - jbegin(3) > minobs) And (tempmax(3) - minsignal > 0) And (tmax(3) - bline * 60 - 1.5 * heater > 0 _
      And IsNumeric(Mom1(3))) Then
      maxtime = tmax(3) - (bline * 60 + heater / 2)
      CLV = CL * cv1(ii)
    Q1 = (c11(ii) - (c31(ii) * (maxtime))) / (maxtime + c21(ii))
    Q1min = (c11(ii) - (c31(ii) * maxtime * (1 + (CLV)))) / _
      (maxtime * (1 + CLV) + c21(ii))
    Q1max = (c11(ii) - c31(ii) * (maxtime) * (1 - (CLV))) / _
      (maxtime * (1 - CLV) + c21(ii))
    lblQ1.Caption = Format(Q1, "#0.00")
    lblQ1max.Caption = Format(Q1max, "#0.00")
```

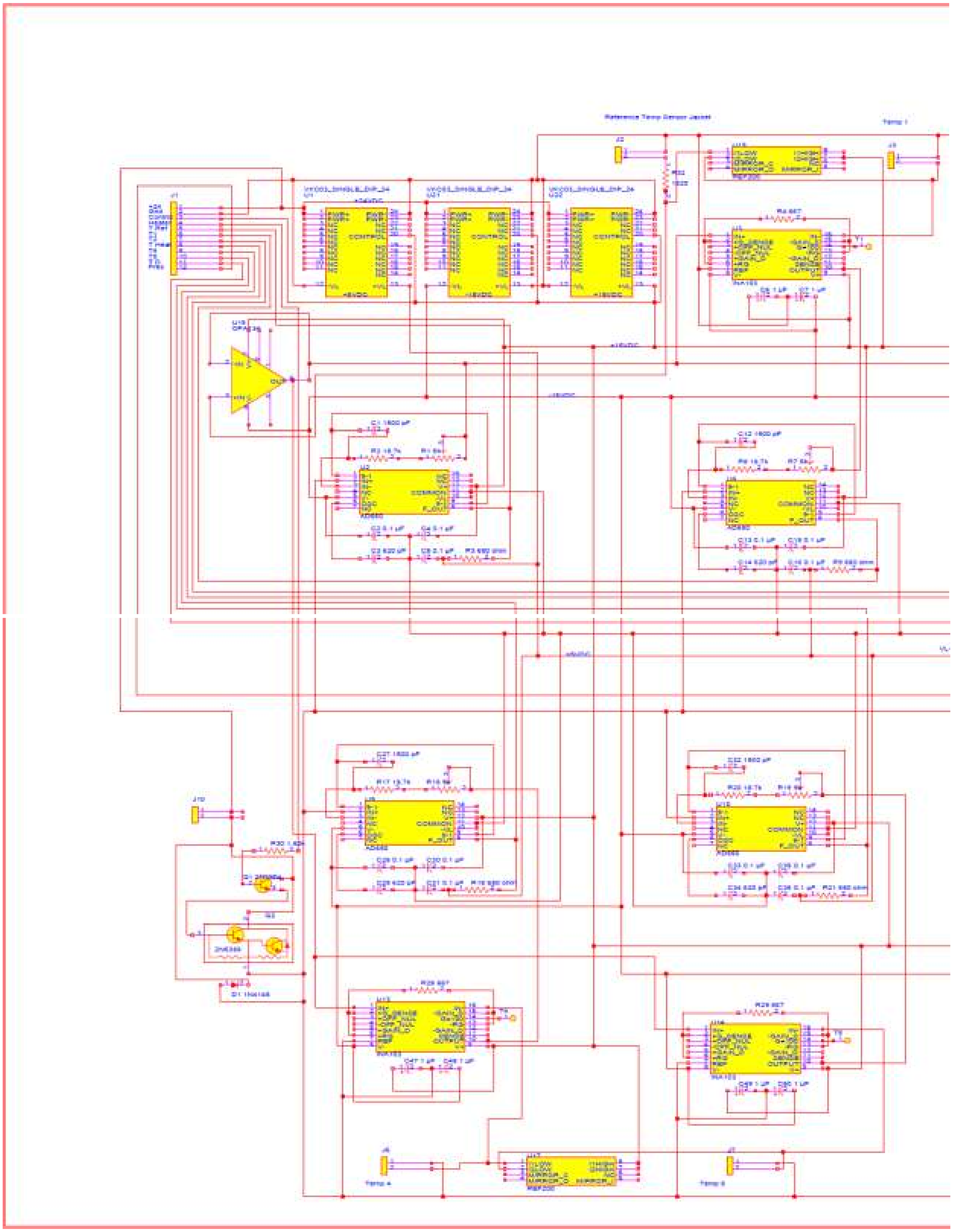
```
        lblQ1min.Caption = Format(Q1min, "#0.00")
    Else
        Q1 = "N.A."
        Q1min = "N.A."
        Q1max = "N.A."
        lblQ1.Caption = Format(Q1)
        lblQ1min.Caption = Format(Q1min)
        lblQ1max.Caption = Format(Q1max)
    End If
'   Channel 5 Captions
    If (jend(5) - jbegin(5) > minobs) And (tempmax(5) - minsignal > 0) And _
            (tmax(5) - bline * 60 - 2 * heater > 0 And IsNumeric(Mom1(5))) Then
            maxtime = tmax(5) - (bline * 60 + heater / 2)
            CLV = CL * cv3(ii)
        Q3 = (c13(ii) - (c33(ii) * maxtime)) / (maxtime + c23(ii))
        Q3min = (c13(ii) - c33(ii) * maxtime * (1 + CLV)) / _
            (maxtime * (1 + CLV) + c23(ii))
        Q3max = (c13(ii) - c33(ii) * maxtime * (1 - CLV)) / _
            (maxtime * (1 - CLV) + c23(ii))
        lblQ3.Caption = Format(Q3, "#0.00")
        lblQ3max.Caption = Format(Q3max, "#0.00")
        lblQ3min.Caption = Format(Q3min, "#0.00")
    Else
        Q3 = "N.A."
        Q3min = "N.A."
        Q3max = "N.A."
        lblQ3.Caption = Format(Q3)
        lblQ3min.Caption = Format(Q3min)
        lblQ3max.Caption = Format(Q3max)
    End If
'   Channel 6 Captions
    If (jend(6) - jbegin(6) > minobs) And (tempmax(6) - minsignal > 0) And _
            (tmax(6) - bline * 60 - 1.5 * heater > 0 And IsNumeric(Mom1(6))) Then
            maxtime = tmax(6) - (bline * 60 + heater / 2)
            CLV = CL * cv4(ii)
        Q4 = (c14(ii) - (c34(ii) * maxtime)) / (maxtime + c24(ii))
        Q4min = (c14(ii) - c34(ii) * (maxtime) * (1 + CLV)) / _
            (maxtime * (1 + CLV) + c24(ii))
        Q4max = (c14(ii) - c34(ii) * (maxtime) * (1 - CLV)) / _
            (maxtime * (1 - CLV) + c24(ii))
        lblQ4.Caption = Format(Q4, "#0.00")
        lblQ4max.Caption = Format(Q4max, "#0.00")
        lblQ4min.Caption = Format(Q4min, "#0.00")
    Else
        Q4 = "N.A."
        Q4min = "N.A."
        Q4max = "N.A."
        lblQ4.Caption = Format(Q4)
        lblQ4min.Caption = Format(Q4min)
        lblQ4max.Caption = Format(Q4max)
    End If
'*********
' Generate Darcy velocity lables
```

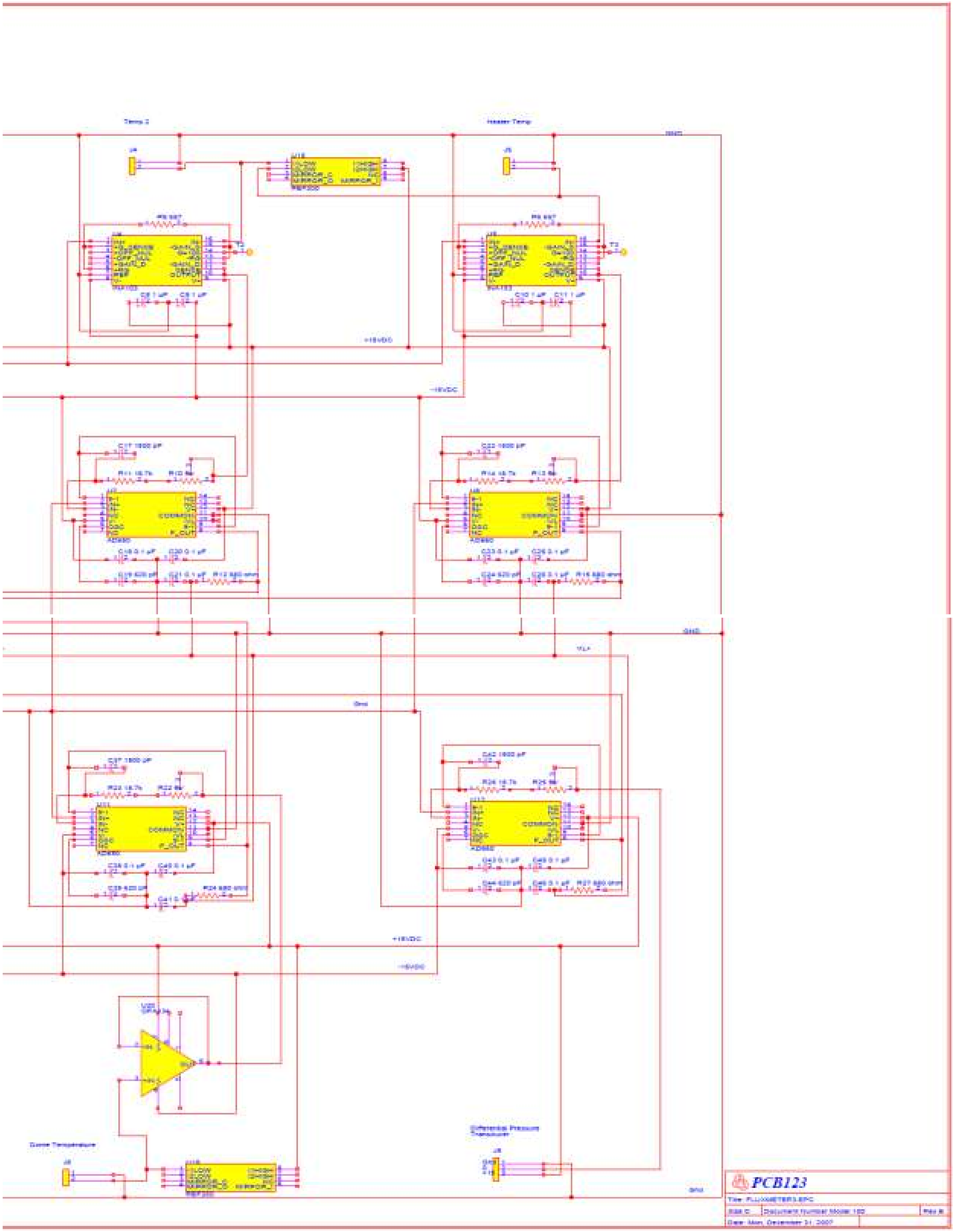
```
'*********
If DomeDia > 0.1 Then   'Data exists to calculate flux
   If Q0 <> "N.A." Then
      V0 = Q0 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      lblV0.Caption = Format(V0, "#0.0")
   Else
      V0 = "N.A."
      lblV0.Caption = Format(V0)
   End If
   If Q1 <> "N.A." Then
      V1 = Q1 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      lblV1.Caption = Format(V1, "#0.0")
   Else
      V1 = "N.A."
      lblV1.Caption = Format(V1)
   End If
   If Q3 <> "N.A." Then
      V3 = Q3 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      lblV3.Caption = Format(V3, "#0.0")
   Else
      V3 = "N.A."
      lblV3.Caption = Format(V3)
   End If
   If Q4 <> "N.A." Then
      V4 = Q4 * 60 * 24 / (3.1415 * DomeDia * DomeDia / 4)
      lblV4.Caption = Format(V4, "#0.0")
   Else
      V4 = "N.A."
      lblV4.Caption = Format(V4)
   End If
Else
   V0 = "N.A."
   V1 = "N.A."
   V3 = "N.A."
   V4 = "N.A."
   lblV0.Caption = Format(V0)
   lblV1.Caption = Format(V1)
   lblV3.Caption = Format(V3)
   lblV4.Caption = Format(V4)
End If
'MsgBox ("check3")

 '********
 ' set up arrays for ploting
 '********

200   Dim Graph() As Single
  Dim x As Integer
  Dim myXarray() As Double        'Time
  Dim myYArray0() As Double
  Dim myYArray1() As Double
  Dim myYArray2() As Double
  Dim myYArray3() As Double
  Dim myYArray4() As Double
  Dim myYArray5() As Double

  ReDim myXarray(N)
  ReDim myYArray0(N)
  ReDim myYArray1(N)
  ReDim myYArray2(N)
  ReDim myYArray3(N)
  ReDim myYArray4(N)
  ReDim myYArray5(N)

'*******
'Generate some x y data.
'*******

  myXarray(0) = t(1)
  myYArray0(0) = ch(1, 1)
'MsgBox (ch(2, 1))
   myYArray1(0) = ch(2, 1)
  myYArray2(0) = ch(3, 1)
  myYArray3(0) = ch(4, 1)
  myYArray4(0) = ch(5, 1)
  myYArray5(0) = ch(6, 1)

'MsgBox N
  For x = 1 To N - 1

    myXarray(x) = t(x) - bline * 60    'value for X-axis
    myYArray0(x) = ch(1, x)         'value for Y-axis Ref Temp
    myYArray1(x) = ch(2, x)          ' Ch 2 cable end temp
    myYArray2(x) = ch(3, x)          'Ch 3 temp
    myYArray3(x) = ch(4, x)          ' Heater Temp
    myYArray4(x) = ch(5, x)
    myYArray5(x) = ch(6, x)          'Discharge end temp

  Next x


  With TChart1
   .AddSeries scPoint
   .Series(0).AddArray UBound(myYArray0), myYArray0(), myXarray() ' ref temp
   .Series(0).XValues.TempValue = True
   .Series(1).AddArray UBound(myYArray1), myYArray1(), myXarray()  ' Cable end TEp
   .Series(2).AddArray UBound(myYArray2), myYArray2(), myXarray()
   .Series(3).AddArray UBound(myYArray3), myYArray3(), myXarray()  ' Heater
   .Series(4).AddArray UBound(myYArray4), myYArray4(), myXarray()
   .Series(5).AddArray UBound(myYArray5), myYArray5(), myXarray()
  End With
'MsgBox ("check 4")

  Screen.MousePointer = 1
'***********
' write output file with processed data
```

```
'***********
ChDir Dir1
Dim fname2 ' file name for output data
'MsgBox ("time0(0) = " & time0(0))
  fname2 = Format$(time0(0), "mm dd hh mm") + "a.CSV"
  Open fname2 For Output As #8
    Write #8, Format("baseline"), Format("heater"), Format("Serial No."), Format("Dome D"), Format("site")
    Write #8, Format(bline, "#0.0"), _
      Format(heater, "#0.0"), Format(SerNumber, "#0"), _
      Format(DomeDia, "#0.0000"), Format(site)
    Write #8, Format("Channel No"), Format("Flow Rate ml/min"), Format("Lower 98% Limit"), _
      Format("Upper 98%Limit"), Format("Darcy Velocity cm/day")
    Write #8, Format("2"), Format(Q0, "#0.00"), Format(Q0min, "#0.00"), Format(Q0max, "#0.00"), Format(V0,
"#0.00")
    Write #8, Format("3"), Format(Q1, "#0.00"), Format(Q1min, "#0.00"), Format(Q1max, "#0.00"), Format(V1,
"#0.00")
    Write #8, Format("5"), Format(Q3, "#0.00"), Format(Q3min, "#0.00"), Format(Q3max, "#0.00"), Format(V3,
"#0.00")
    Write #8, Format("6"), Format(Q4, "#0.00"), Format(Q4min, "#0.00"), Format(Q4max, "#0.00"), Format(V4,
"#0.00")

    Write #8, Format("Peak arrival time in seconds from cable end")
    Write #8, Format(tmax(2) - (bline * 60 + heater / 2), "#0.0"), Format(tmax(3) - (bline * 60 + heater / 2), "#0.0"),
_
      Format(tmax(4) - (bline * 60 + heater / 2), "#0.0"), Format(tmax(5) - (bline * 60 + heater / 2), "#0.0"), _
      Format(tmax(6) - (bline * 60 + heater / 2), "#0.0")
    Write #8, Format("Supplemental Data Temperature Outside Dome Ch7, Ch8, Ch9, Ch10")
    Write #8, Format("Ch7 looks for calibration coefficients while Ch8 - ch10 are equivalent voltages")
    Write #8, Format("Ch7"), Format("Ch8"), Format("Ch9"), Format("Ch10")
    Write #8, Format(ch7, "#0.0"), Format(ch8, "#0.0"), Format(ch9, "#0.0"), Format(ch10, "#0.0")
    Write #8, Format("Maximum differential temperature or reference temperature c")
    Write #8, Format("Ch 1 Ref"), Format("Ch 2"), Format("Ch 3"), Format("Ch 4"), Format("Ch 5"), _
      Format("Ch 6"), Format("Ref Temp C")
    Write #8, Format(tempmax(1), "#0.00"), Format(tempmax(2), "#0.00"), Format(tempmax(3), _
      "#0.00"), Format(tempmax(4), "#0.00"), Format(tempmax(5), "#0.00"), Format(tempmax(6), "#0.00"), _
      Format(avgy(1), "#0.")
    Write #8, Format("Normalized First Moments Calculated beginning at the end of the base line samples")
    If ((jend(2) - jbegin(2)) > minobs) And (tempmax(2) - minsignal > 0 And _
      Mom1(2) <> 0 And IsNumeric(Mom1(2))) Then
        Write #8, Format("Ch 2"), Format(Mom1(2) / Momzero(2), "#0.0")
    Else
        Write #8, Format("Ch 2"), Format("N.A.")
    End If
    If ((jend(3) - jbegin(3)) > minobs) And (tempmax(3) - minsignal > 0 And _
      Mom1(3) <> 0 And IsNumeric(Mom1(3))) Then
        Write #8, Format("Ch 3"), Format(Mom1(3) / Momzero(3), "#0.0")
    Else
      Write #8, Format("Ch 3"), Format("N.A.")
    End If
    If ((jend(4) - jbegin(4)) > minobs) And (tempmax(4) - minsignal > 0 And _
      Mom1(4) <> 0 And IsNumeric(Mom1(4))) Then
        Write #8, Format("Ch 4"), Format(Mom1(4) / Momzero(4), "#0.0")
    Else
```

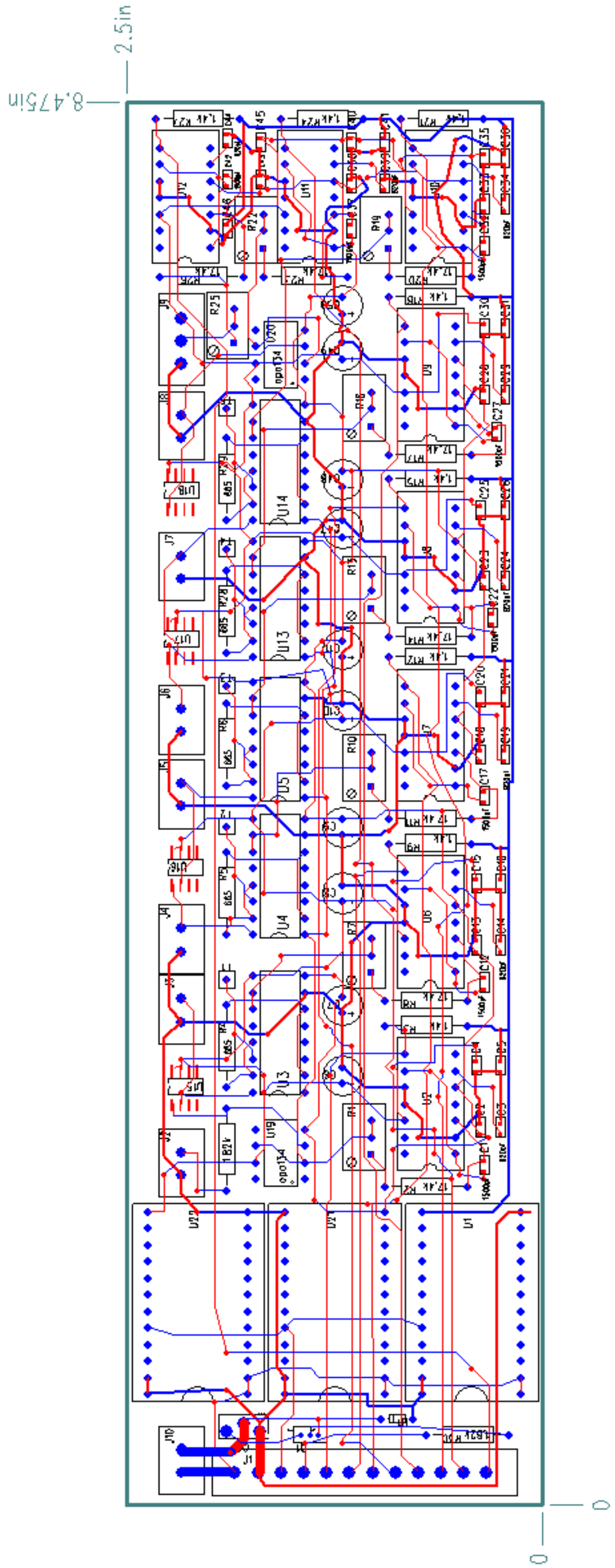
```
      Write #8, Format("Ch 4"), Format("N.A.")
    End If
    If ((jend(5) - jbegin(5)) > minobs) And (tempmax(5) - minsignal > 0 And _
      Mom1(5) <> 0 And IsNumeric(Mom1(5))) Then
        Write #8, Format("Ch 5"), Format(Mom1(5) / Momzero(5), "#0.0")
    Else
      Write #8, Format("Ch 5"), Format("N.A.")
    End If
    If ((jend(6) - jbegin(6)) > minobs) And (tempmax(6) - minsignal > 0 And _
      Mom1(6) <> 0 And IsNumeric(Mom1(6))) Then
        Write #8, Format("Ch 6"), Format(Mom1(6) / Momzero(6), "#0.0")
    Else
      Write #8, Format("Ch 6"), Format("N.A.")
    End If
    Write #8, Format("Times in seconds from beginning of heat-pulse and Rel Temperatures in Deg C")
    Write #8, Format("E.T."), Format("T-Ref-1"), Format("T-2"), Format("T-3"), Format("T-4"), Format("T-5"),
Format("T-6")

  For x = 1 To N - 1
    Write #8, Format(myXarray(x), "#0.000"), Format(myYArray0(x), "#0.000"), Format(myYArray1(x), "#0.000"), _
      Format(myYArray2(x), "#0.000"), Format(myYArray3(x), "#0.000"), Format(myYArray4(x), "#0.000"), _
      Format(myYArray5(x), "#0.000")
  Next x

  Close
  Refresh
  DoEvents
 GoTo 45
44 Unload Me
  Screen.MousePointer = 1

45    End Sub



Private Sub TeeCommander2_OnEditedChart()

End Sub

Private Sub Mom4_Click()
End Sub
```

Appendix B Circuit Diagram of Measurement System

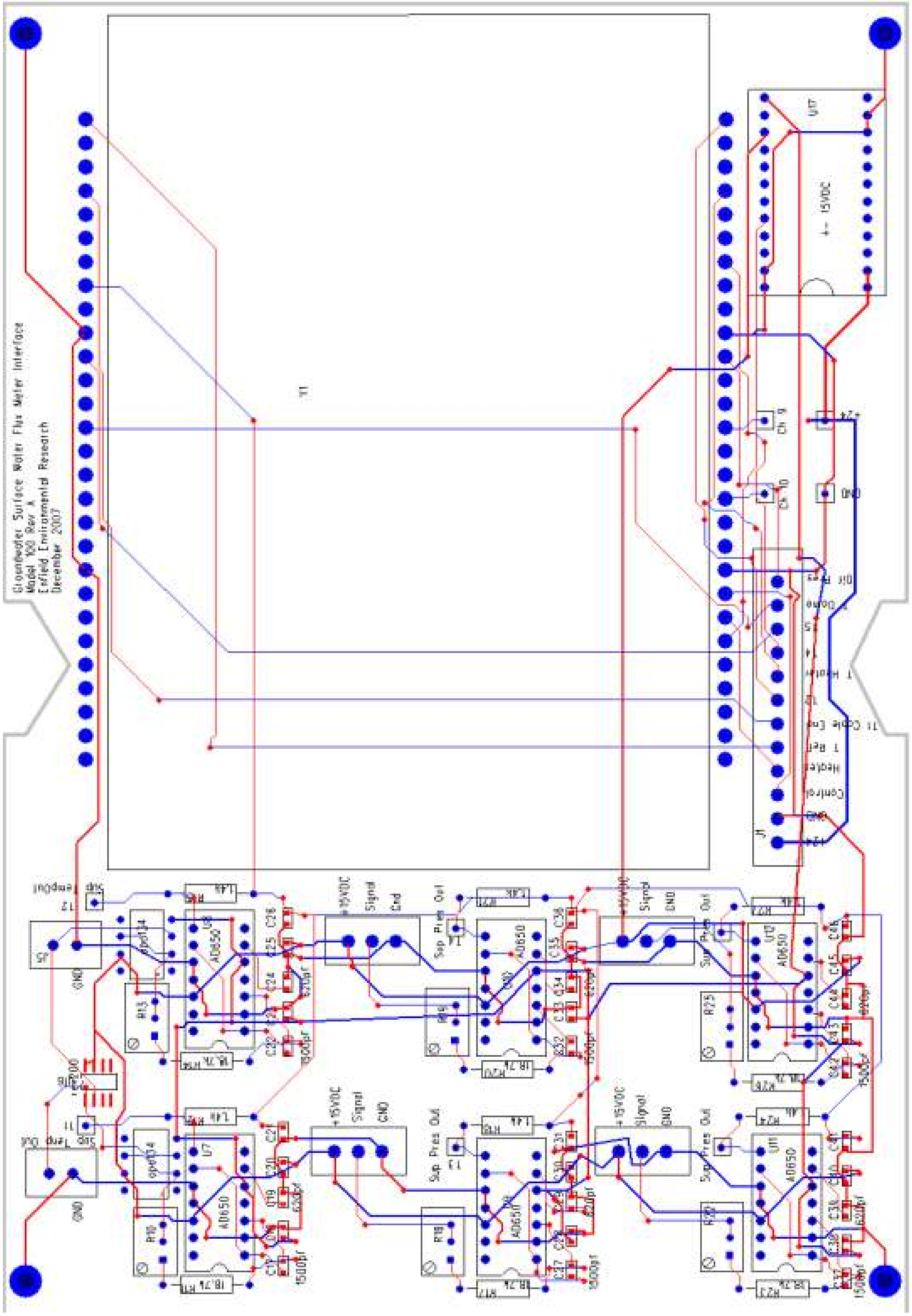



Temp 2
Header Temp
GND
J4
J5
U16
REF200
INA103
C8 1 µF
C9 1 µF
C10 1 µF
C11 1 µF
+15VDC
-15VDC
C17 1500 pF
C22 1500 pF
R11 16.7k
R10 5k
R14 16.7k
R13 5k
AD650
C18 0.1 µF
C20 0.1 µF
C19 620 pF
C21 0.1 µF
R12 680 ohm
C23 0.1 µF
C26 0.1 µF
C24 620 pF
C25 0.1 µF
R15 680 ohm
GND
VL+
Gnd
C37 1500 pF
R23 16.7k
R22 5k
C42 1500 pF
R26 16.7k
R25 5k
AD650
C38 0.1 µF
C40 0.1 µF
C39 620 pF
R24 680 ohm
C43 0.1 µF
C45 0.1 µF
C44 620 pF
C46 0.1 µF
R27 680 ohm
+15VDC
-15VDC
U20
OP
Dome Temperature
J8
Differential Pressure Transducer
J9
U18
REF200
Gnd
PCB123
Title: FLUXMETERS.EPC
Size C
Document Number Model 100
Rev B
Date: Mon, December 31, 2007

Appendix C Signal Conditioning Circuit Board Layout



Appendix D Interface Circuit Board Layout



Appendix E Calibration constants of flow sensor SN300, SN303

		C_1	C_2	C_3
SN300	**T2**	-674.98	-14.27	0.081
	T3	-380.23	-14.19	-0.15
	T5	370.28	-17.20	0.46
	T6	562.70	-25.58	-0.12
		C_1	C_2	C_3
SN303	**T2**	-603.84	-24.61	0.11
	T3	-420.85	-17.45	-0.96
	T5	379.77	-17.03	0.69
	T6	695.78	-14.35	-0.10