WORK IN PROGRESS PentaMetricRS232SerialSpecWeb.doc

June 2011, Ralph Hiesey rev 15Jun11: added info on Ethernet/TCP-IP communication

How to access PentaMetric data:

via the RS232 port of the PM-100-C computer interface

or via Ethernet port of the PM-101-CE computer-TCP/IP interface.

Note: This information is intended for computer programmers that wish to control the PentaMetric using their own software program. Although the information here is mostly correct, it hasn't been thoroughly checked because of time constraints--therefore there are very likely to be some errors. Please call and let us know if something doesn't seem to be correct, or if certain parts are confusing so we can correct or improve this document. Bogart Engineering: 831 338-0616.

1. The first few pages of this document shows how to access data directly from the RS232 port, and applies directly to the PentaMetric PM-100-C interface. **If you wish to access data from the PentaMetric PM-101-CE interface with the TCP/ Ethernet connection**, then in addition you will also need the document which begins at the end of this document, beginning with page 11 which further explains how to access the TCP data. Once having gained such access, then the documentation immediately below will apply.

2. This document covers how to read the "real time" data in the PentaMetric (such as present "volts", "amps", "watt hour" etc.) and also how to change all of the "programmable data" that is used to control the operation of the PentaMetric.

3. There are also three "logged data" files that reside within the Pentametric. They are on the order of 4 kbyte each. By giving appropriate commands this data can be sent as serial output data with file lengths of a multiple of 256 bytes plus one byte checksum. The information for downloading and interpreting this data will be explained in separate documents. Presently, such documentation is now available on the **bogartengineering. com** website for intrepreting the "Periodic logged data". Look under "support/application notes/PentaMetric." For information on downloading the "Efficiency data" and "Discharge Profile" log data files contact Bogart Engineering.

4. For the RS232 communication data only: Communication is at 2400 baud, 8 bits, no parity. The RS232 RTS and CTS handshaking lines are not needed. Details on the TCP/IP communication are shown in appendix 1, page 11.

5 Communication of the "real time" and "programmed" data takes place by "reading" or "writing" data registers that reside in the PentaMetric "input unit".. Each type of data, for example "volts 1", "amp hours" or "charged voltage setpoint" may be read (or written to) by giving commands consisting of several sequential bytes of data:

How to read data: There are two ways to <u>read</u> data from the PentaMetric: a SHORT READ or LONG READ. The SHORT READ is used for accessing just a few bytes from the tables below used for reading real time data or programmed data (described in detail here). The LONG READ reads entire pages (of 256 bytes) so the data read has the number of bytes equal to a multiple of 256, used when downloading the logged data (described in detail in other documents.)**.**

SHORT READ: This following is used to read the "real time data" shown in tables 1 and the programmed data in Table 2. To READ a particular item in Table 1: Send the bytes in this order: First send hex 81 ("read" command) Then send the hex "address" as shown in Table 1 or 2. **Note that the table shows the address as a decimal number—this must be converted to a hex value for this command.** Next send the "no. bytes" (N) shown in the table. Finally send a one byte checksum. The checksum must be the hex number, which when added to the previous 3 bytes results in a sum with "FF" (hex) as the least significant byte. When this is done the Pentametric will respond in a few hundred milliseconds by sending the N data bytes, followed by the Checksum. The checksum is such that when the N+1 bytes are added the sum will have "FF: as the least significant byte. The data emerges with the LOWEST BYTE FIRST, with successive bytes, ending with the HIGHEST BYTE followed by the checksum.

Example: if you want to read **Average Battery1 volts** (see table1 below: under display D3) you would send the following 4 (hex) bytes 81, 03,02, 79. Each byte is a hex value from 0-ff. Note that the sum of these 4 bytes is "0FF". The PentaMetric would respond with 2 bytes of data followed by 1 byte of checksum. The LOW BYTE comes first followed by the HIGH BYTE, then the checksum. To confirm correct reception of the data you could add all (3) bytes together and see if they add up to a hex number with "ff" as the low byte.

Send: (all hex digits): 81,03,02,79. Response (for example): FA, 01, 04. The high byte is 01, the low byte FA. Checksum=04. So the voltage is 1FA= 506 decimal. Divide by 20 to get 25.3 volts. The sum of FA + 01 + 04=0ff, which confirms that the transmission is OK

LONG READ of 256 byte blocks: (For reference) This command is used for reading the logged data. It is shown for completeness but is used only when downloading the logged data, which this document doesn't explain. (We intend to describe this in a separate document.) This command accesses always a multiple of 256 bytes. The memory space is divided into 64 "pages" of 256 bytes each. But you can only read up to 4 pages in one read command (i.e, up to 1024 bytes at a time) To **READ** N **pages** (where N is between 1 and 4) starting with the base location of "page"=P (where P=0 through 3f hex) Send the following 4 (hex) bytes in sequence: C1(hex), P, N, X, where X is the checksum . The Pentametric then responds by sending the N*256 data bytes, followed by the Checksum, which is the hex number, which when added to all the bytes sent causes the least significant two bytes to be FF.

SHORT WRITE: Writing the data as shown in Table 2 (or even Table 1) is done as described here.

This command will write a maximum of 16 bytes of data at a time. To WRITE N bytes (up to 16 sequential bytes) for a particular item in Table 2 (or table 1): send N+4 bytes in this order: First send hex 01 ("write" command) Then send the hex "address" as shown in Table 2 (or 1): . Next send the "no. bytes" (N) shown in the table, then send each desired byte to write, beginning with the LOW BYTE and ending with the HIGH BYTE. Finally send a one byte checksum. The checksum must be the hex number, which when added to all the previous command bytes results in a sum with "FF" (hex) as the two least significant hex digits. When this is done the Pentametric then responds in a few hundred milliseconds by sending back the same checksum if it successfully received and wrote the data.

Example: To write the "Batt1 Capacity" equal to 1000 amp hours (see Table 2, "Batt1 Capacity") you would send the following 6 hex digits: 01,f2, 02, e8, 03, 1f. The decimal 1000 number is 03e8 hex, which is where the e8, 03 comes from. Note that the sum of all 6 bytes is 1ff. The PentaMetric will respond with the checksum originally sent (1f) if it receives the data OK. If you then read back the location it should contain the new data.

Table 1, below is the Display table: Data here is ordinarily only read, not written to. The **first column** shows the "display number" which is used to identify the type of data as described in the PentaMetric instructions. The **second column** shows the description of the data. The 5th column shows how to decode the data.

RESET commands:. A few of the display functions (in Table 1) have a "reset" command to easily allow the user to reset these values to 0. These are D13,14,15 (Amp hours), D16,D17 (cumulative amp hours), D20,21 (Watt hours), D24,25 Days since charged D26,27 (days since equalized). To "reset" these, write the data shown in the "Reset command" column to decimal location 39 (hex 27). (They could also be reset by just writing 0's to the address location, however.)

Example: To reset the "amp hours 1" to 0, note that the "Reset command" for "amp hours1" is 09. Write the data 09 to location hex 27 ,(one byte), as follows: 01,27,01,09,cd. The PentaMetric should respond with "cd". (Note that the "01" just before the 09 tells the PentaMetric that it is writing to just one location.)

TABLE 1.

IMPORTANT NOTE: When data is read from serial port, the LOW byte comes out first, and the HIGH byte comes out last.

Display number	Display Data Displayed name	decimal Address	No Bytes	Display format	Display units	Reset command
D1	**Battery1 Volts**	1	2	**FORMAT1:** Take the low 11 bits of (binary) data, divide it by (decimal) 20,	Volts	
D2	**Battery2 Volts:**	2	2	**FORMAT1**	Volts	
D3	**Average Battery 1 Volts**	3	2	**FORMAT1**	Volts	
D4	**Average Battery2 Volts**	4	2	**FORMAT1**	Volts	
D7	**Amps 1:**	5	3	**FORMAT2:** The 3 bytes of data consist of 24 bits of data which we label from B0 (LSB) to B23 (Hi bit)**. First** look at the highest bit (B23). **If B23 is high**, then COMPLEMENT all the bits B0-B22, and strip B23 off. (So you have 23 bits left.). **If B23 is low,** just leave the number as is. This is the binary data in 1/100 amp units: so next divide by 100 (decimal) to get amps. **Finally**, multiply the result by -1 for correct sign. **Note**: when using the 100 amp/100mV shunt it's OK to regard the lowest digit as significant. However when using the 500amp/50mV shunt you should only use the digits down to 1/10 amp, as the 1/100 amp digit is beyond the resolution of the PentaMetric.	Amps	
D8	**Amps 2**	6	3	**FORMAT2:**	Amps	
D9	**Amps 3:**	7	3	**FORMAT2.**	Amps	
D10	**Average Amp1s**	8	3	**FORMAT2.**	Amps	
D11	**Average Amp2**	9	3	**FORMAT2.**	Amps	
D12	**Average Amp3:**	10	3	**FORMAT2.**	Amps	
D13	**Amp Hours 1**	12	3	**FORMAT 3:** Same as **FORMAT2,** except disregard the **Note.**	Amp-hours	09
D14	**Amp Hours 2**	13	3	**FORMAT3**	Amp-hours	0a
D15	**Amp Hours 3**	15	4	**FORMAT4:** The4 bytes of data consist of 32 bits of data which we label from B0 (LSB) to B31 (Hi bit)**. First** look at the highest bit (B31). **If B31 is high**, then COMPLEMENT all the bits B0-B31,.. **If B31 is low** leave as is. **Next** strip off the low 7 bits: B0-B6, leaving 24 bits (B7-B30) This is the binary data in 1/100 amp hr units: so next divide by 100 (decimal) to get amp-hr..	Amp-hours	0b
D16	**Cumulative Amp Hours 1**	18	3	**FORMAT2B.** Same as **FORMAT2** except don't divide the result by 100..	Amp-hours	b0
D17	**Cumulative Amp Hours2**	19	3	**FORMAT2B.**	Amp-hours	b1
					Amp-hours	
D18	**Watts 1:**	23	3	**FORMAT2**	Watts	
D19	**Watts 2**	24	3	**FORMAT2**	Watts	
D20	**Watt Hours 1:**	**21**	4	**FORMAT5:** The4 bytes of data consist of 32 bits of data which we label from B0 (LSB) to B31 (Hi bit)**. First** look at the highest bit (B31). **If B31 is high**, then COMPLEMENT all the bits B0-B31,.. **If B31 is low** leave as is. This is the 31 bit binary data in 1/100 amp hr units: so next divide by 100 (decimal) to get watt-hr.. Finally multiply by -1 to get correct sign.	Watt-hours	11

D21	**Watt Hours 2**	22	4	**FORMAT5**	Watt-hours	12
D22	**Battery1 Percent Full**	26	1	**FORMAT6:** Just use the number "as is".	%	
D23	**Battery 2 Percent Full:**	27	1	**FORMAT6**	%	
D24	**Days since Battery 1 charged**	28	2	**FORMAT7.** Divide 2 byte result by 100 (decimal). Result has 1/100 day resolution.)	Days	19
D25	**Days since Battery 2 charged**	29	2	**FORMAT7**	Days	1a
D26	**Days since Battery 1 equalized**	30	2	**FORMAT7**	Days	1b
D27	**Days since Battery 2 equalized**	31	2	**FORMAT7**	Days	1c
D28	**Temperature**	25	1	**FORMAT8** 1 byte. Signed 1 byte number: 2's complement: ie fe= -2 , ff=-1, 0=0, 1=1, etc thru 127. 80(hex)= -128. So it can express number from -128 to +127.	Degrees C.	
D29thru34.	**Battery1 efficiency data**	218	12	Byte 0-1 is 1 cycle self disch current (amps): Take low 10 bits and divide by 100 to get value of amps. Sign is indicated by a "1" or "0" in bit 15. means -, 0 means +. Format is ±X.XXby 100 Byte 2 is 1 cycle batt efficeincy: like **FORMAT6** **Byte 3 is # cycles** Byte4-7, same for 4 cycle Byte8-11 same for 15 cycle		
Alarm data	**Battery alarm data**	38	2	Low byte has Battery 1 alarm data. High byte has battery 2 alarm data: Bit0=Batt low, Bit1=Bat charged, Bit2=Bat hi, Bit3=Tim to ch full Bit 4=Time to equalize		
D35thru40	**Battery 2 efficiency data**	215	12	Same as D29 thru 34		

MISC COMMAND DATA

/function	Loc	# bytes	command codes		
Misc command: write to this location to reset various registers	**39 (hex27)**	**1 byte**	**09 clears amp hr1. 0a clears amp hr2. 0c clears both. 19 resets "days since charged 1". 1A resets "days since charged 2" 11=reset watt hr 1. 12=reset watt hr 2. 13=reset both.**		

TABLE 2: PROGRAMMED DATA TABLE

Program number is the "P" number. Note that not all numbers will be used at first. Later they may be added.
Address and **No of bytes** are the numbers that are sent to the Pentametric to read the data.
Data Format points to the program that converts the received bytes to the screen display. It utilizes the data in "Display name".

Allowed Data Limits describes the lower and upper bound of acceptable data to store.

Description	Program number	(hex) Address1	No. of bytes 1	Address2	No of bytes2	**Data format:** To understand this: Refer to section 6A, PentaMetric instructions corresponding to program number	Allowed data limits)
	8 bit Integer	8 bit Integer	8 bit Integer	8 bit integer	8 bit integer		
Sw1-select	P1	0ff	5		0	**FormatA**: Refer to PentaMetric instructions, section 6 under P1-P5. Each of the 5 bytes specifies the "AD" number of the display items to be displayed, starting from byte1 to byte 5. Put 0 if it is desired to not show any data. For example, if the 5 bytes of data are 08,01,0,0, 0, switch 1 would display .Amps2 and Battery 1 volts. .	Limits of data: 0-28 (decimal). "0" means "don't display"
Sw2-select	P2	0fe	5		0	**FormatA**	
Sw3-select	P3	0fd	5		0	**FormatA**	
Sw4-select	P4	0fc	5		0	**FormatA**	
Sw5-select	P5	0fb	5		0	**FormatA**	
Batt2 Label	P6	e1	3		0	Bit 4 indicates label. bit 4= "1" is "Battery".Bit 4= 0 makes "Battery 1"	Bit 4 hi or low.
Amp1 thru 3 Labels	P7-9	e1	3		0	Byte1=label for amps 1. Byte2=label for amps 2. Byte3=label for amps3. The label is indicated by bits 0-3 as follows: 0=Amps#. 1=Solar 2=Wind 3=Hydro 4=Load 5=Battery 6=Battery 1.	For all 3 bytes: bits0-3 must be from 0 thru 6.
Shunt Select	P11-13	f6	3		0	Byte1=shunt for Amps 1. Byte2=shunt for Amps 2 Byte3=shunt for Amps3. Bit 4 hi=100A/100mV shunt Bit 4 lo=500A/50mV shunt.	For all 3 bytes, bit 4 either hi or low.
FirmwareVersionNo.	N/A	f7	1			Has version number: example: V1.2 is 12 decimal	0-255 (ver 0.0 – 25.5)
Batt1Capacity	P14	0f2	2		0	**FormatD:** Using 2 bytes, put capacity in amp hours 1-9999.	Limits of data: 0-9999. "0" indicates "no battery"
Batt2Capacity	P15	0f1	2		0	**FormatD:**	
Filter Time	P16	0f3	1		0	0=Time constant=0, 1:TC= .5 min, 2:TC=2min. 3=TC=8min, 4=32 min	Bits 0-1 can be number from 0-4.
Ch control(not used)	P17-20	0f0	7		0		

AlarmLevlBat1	P22-23	0ce	2		0	**Byte 1**: Bits 0-1 : Lo Batt alarm. Bits2-3:Hi batt alarm, bits 4-5: Batt charged, bits 6-7: **Byte 2**: Time to charge, bits0-1 For each of above: 0=alarm off. 1=visual alarm only, 3=visual/audible alarm.	see column to left.
AlarmLevlBat2	P24-25	0cd	2		0	**Byte 1**: Bits 0-1 : Lo Batt alarm. Bits2-3:Hi batt alarm, bits 4-5: Batt charged, bits 6-7: **Byte 2**: Time to charge, bits0-1 For each of above: 0=alarm off. 1=visual alarm only, 3=visual/audible alarm.	
Bat1LoAlarmSetpoint	P26	0ec	3		0	**FormatG:** Bytes1-2: Take low 10 bits. Divide by (decimal) 10 to get volts. Byte3=%, from 0-100 .	
Bat1HiAlarmSetpoint	P27	0ea	2		0	**FormatF:** Bytes1-2: Take low 10 bits. Divide by (decimal) 10 to get volts. :	
Bat2LoAlarmSetpt	P28	0eb	3		0	**FormatG:**	
Bat2HiAlarmSetpt	P29	0e9	2		0	**FormatF**	
Relay ON/OFF Setpt	P30-31	0d4	6		0	**Low3 bytes** Relay on; Bytes 1-2: voltage setting, byte3: %Full setting . **High 3 bytes:** Relay off; Bytes 1-2: voltage setting, byte3: %Full setting	
Batt1charged" criteria	P32	0e8	3		0	**FormatG:** Bytes1-2: Take low 10 bits. Divide by (decimal) 10 to get volts. Byte3=Amps, from 0-100?? .	Bytes 1-2
Batt2charged" criteria	P33	0e7	3		0	**FormatG**	
BattEfficy setpoints	P34-35	0e5	3		0	Low byte: %, next two bytes: amps expressed in 1/100 amp units.: 0-999 (decimal)	Amps: 0-999 decimal.
TimeBetween Equalize	P36	e3	1		0	Days 0-255. (0=off)	
TimeBetween Charge-	P37	e2	1		0	Days 0-255. (0=off)	
Day: 1/8 day units	P38	f9	2			To get day and time you need to read two different locations: The two bytes at f9 give the DAYS in 1/8 day units (0-65535). This gives the time within 1/8 day (=180 minutes). The remaining time (in minutes) is read in location 24, (0-179)and is added to other data to give date/time to 1 minute. See Note 5.	
Time: minutes 0-179	P38	24	1				

Time of One Periodic data measurement	P39	cf	3			Take byte 1, multiply by 180. Add this to byte 3. (Ignore byte 2) This gives number of minutes after midnight that measurement is taken. (a number from 0-1439)	
Periodic data: number of measurements/day	P40	d0	1			Each bit can be either 0 or 1. Calculate [(bit0 +1) * (bit1+1) * (bit2+1) * (bit3+1) * (bit4+1) * (2* bit5+1) * (2* bit6+1) * (4 * bit7+1)]. (The * represents multipication) . This is the number of times/day that the measurement will occur.	
Periodic data: items to log:	39-42	d2	2			Select measurement when bit is =1. **Byte 1:** Bit0=Amp hr 1. Bit1=AmpHr2. Bit2=Amp hr3. Bit 3=Watt Hr1 Bit4=WattHr2. Bit 5=Min/Max Temp. Bit6=Volts1 Bit 7=Amps1. **Byte 2**: Bit 0=Volts2. Bit1=Batt%Full	Change41-50
5%Data-Options	43	d1	1			Only bits 5 through 7 are used: But bits 0-4 must not be disturbed. So before writing bits 5-7 be sure to read the data and write back 0-4 the same way so as to leave them unchanged. Bit5=Record Batt 1 data: Bit 6=Record batt 2 data. : Bit7:=Record every 5% (otherwise every 10%) :	Change51-54
Backlight	44					Not yet implemented	
Erase periodic data	45					To erase: write hex 72. to location (hex)27, (or decimal 39.)	
Erase Batt discharge voltage profile data	46					To erase: write hex 82 to location (hex)27, (or decimal 39.)	
Erase battery 1 efficiency data	47					To erase: write hex 90 to location (hex)27, (or decimal 39.)	
Erase battery 2 efficiency data	48					To erase: write hex 91 to location (hex)27, (or decimal 39.)	
Erase & Initialize All programmed data	49					To erase: write hex a5 to location (hex)27, (or decimal 39.)	Change55-59???

NOTE 5: This gives a "relative present time" in "days and minutes". Using the PC computer's time system you can determine the "actual present date time". The "log data" times in the PentaMetric are stamped with the time according to this "relative time". By subtracting the "logged data" date/time from the date/time you read here you can determine how long ago a particular data was logged. This can be subtracted from the computer's time to determine its actual time/date.

.

(For reference) The following describe the 3 types of logged data which can be downloaded. The procedure for downloading and intrepreting the result is described in other documents.

(1)**The PERIODIC data**, consists of periodic recording of any or all of the following: AmpHours1, AmpHours2, AmpHours3, Watt-hr1, Watt-hr2, (Filtered)Volts1 and (FilteredAmps1). For stage 1, this just needs to download all the data from 300 hex to 1fff. This just requires that the PAGE data be retrieved, 50h starting from Page "0C" be loaded into a excel friendly file (7360d bytes) call **GetDataPages(0ch, 73h,M)** and then put data in a comma delimited file

(2) **The BATTERY DISCHARGE CURVE data** shows (filtered) volts and amps for each increment of 5% (or 10%) battery full. This is used to check batteries for possible capacity loss.This requires that the PAGE data be retrieved, 40h pages, starting from Page "80h" be loaded into a excel friendly file.(total 1000bytes-=4096 d bytes)Call **GetDataPages(80h,40h,M)** and then put data in a comma delimited file

(3)**The battery cycle efficiency data** tracks the time between successive "full charge" points, the amp-hours charging and amp hours discharging. This is used to check the batteries for their ability to retain charge. This requires that the PAGE data be retrieved, 40h pages, starting from Page "C0" be loaded into a excel friendly file.(total 1000bytes-=4096 d bytes)Call **GetDataPages(c0h,40h,M)** and then put data in a comma delimited file

Alarm data : . There are 10 different possible alarm conditions--5 for each battery. For each one there are 3 choices: 1.NO ALARM, 2.VISUAL ONLY ALARM, 3.AUDIBLE AND VISUAL. These can be read by reading location 25 (hex). See details below in Table 3 , under "Alarm Status"

Table 3

Description	Address	No bytes	detailed description	Where used
Amps and Battery 1 labels	e1	3	low byte: bits 0-3: Amps 1 label 0= "**Amps 1**" . 1= "**Solar**". 2= "**Wind**" 3= "**Hydro**" 4= "Load". 5= "Battery".6=Battery 1. Bit 4=Battery 1 label. When Bit4=0, "**Battery 1**". When Bit4=1, "**Battery**" Bits 3, 5-7 not used. Middle byte: bits 0-3: Amps 2 label 0= "**Amps 2**" . 1= "**Solar**". 2= "**Wind**" 3= "**Hydro**" 4= "Load". 5= "Battery".6=**Battery 2** High byte: bits 0-3: Amps 3 label 0= "**Amps3**" . 1= "**Solar**". 2= "**Wind**" 3= "**Hydro**" 4= "**Load**". 5= "**Battery**".6=**Battery**	When displaying
Alarm (enable) level	ce	4	Each alarm is specified as one of 3 levels: 0=**alarm off**, 1=**visual alarm only**. 2=**visual and audio alarm**. Low byte: Bits0-1 "**Low Batt 1 alarm**" Bits2-3: "**Battery 1 meets charged criteria**", Bits 4-5: "**high Batt 1 alarm**":. Bits 6-7: "**Time to Charge Batt 1 full**" 2nd byte: Bits0-1 "**Time to Equalize Batt 1**"" Bits2-7- not used (or actually, not needed: bit 7 is "relay on" bit) : 3rd byte: Bits0-1 "**Low Batt 2 alarm**" Bits2-3: "**Battery 2 meets charged criteria**", Bits 4-5: "**high Batt 2 alarm**":. Bits 6-7: "**Time to Charge Batt 2 full**" Highest byte: Bits0-1 "**Time to Equalize Batt 2**" Bits2-7- not used:	
Alarm status	25 (hex)	2	Low byte: Bit0: **Low Batt 1 alarm**. Bit1: **Battery 1 meets charged criteria** Bit2: "**high Batt 1 alarm**" Bit3: "**Time to Charge Batt 1 full**" Bit4: "**Time to Equalize Batt 1**" bits 5-6 not used. Bit7=relay on, but not used here. High byte: Bit0: **Low Batt 2 alarm**. Bit1: **Battery 2 meets charged criteria** Bit2: "**high Batt 2 alarm**" Bit3: "**Time to Charge Batt 2 full**" Bit4: "**Time to Equalize Batt 2**" bits 5-6 not used. Bit7=relay on, but not used here.	

Shunt type	f6	3	Low byte: bit 4: Shunt for **Amps1** channel: When 0=100A/100mV shunt. When 1= 500A/50mV shunt. Other bits not used. Middle byte: bit 4: Shunt for **Amps2** channel: When 0=100A/100mV shunt. When 1= 500A/50mV shunt. Other bits not used. Hi byte: bit 4: Shunt for **Amps3** channel: When 0=100A/100mV shunt. When 1= 500A/50mV shunt. Other bits not used.	
Minutes	24	1	Minutes 0-179 (to be added to days, below)	
Days	f9	2	1/8 day units.	

Appendix on next three pages: Additional information on how to access TCP/IP data when accessing data from the PentaMetric PM-101-CE Computer Interface

by John Hiesey

Connecting I/O using a TCP socket
To connect to the PentaMetric, open a TCP connection to the correct IP address and port number. If you are connecting from a local network, the PentaMetric also responds to NetBIOS name requests. The PentaMetric only accepts a connection from one client at a time.

When the Defaults Jumper (J3) on the Internet Interface is enabled (moved to the position closest to the Ethernet jack), the default settings are as follows:
The IP address (and gateway) is set by DHCP or to 169.254.1.1 if no DHCP server is on the network.
The Port number is set to 1701
The Subnet Mask is set to 255.255.0.0
The NetBIOS name is set to "PENTAMETRIC1"

The following describes the password protection which will eventually be enabled. For now, see the section "BETA operation without passwords" below.
As soon as the TCP connection is established, 9 bytes will be received. The first byte is the version number of the Internet Interface. The next 8 bytes represent a random number that is used as a "challenge" to verify the password. *Take these 8 bytes and concatenate these with the password (padded with 0's at the end to a total length of 16 bytes) and compute the SHA1 hash of that value. Send the first significant 8 bytes back to the PentaMetric*

For example, if the PentaMetric sends a random number of
52 1A DD 8C 26 97 C7 80 (in that order of transmission),
and the password is
41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 (this represents the ASCII string 'ABCDEFGHIJKLMNOP', in that order, and should be stored with the first byte at the lowest address in the PentaMetric memory),
the value that should be hashed is
52 1A DD 8C 26 97 C7 80 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50. When hashed, this gives a value of
1D C0 52 A6 CB A3 4D 41 88 DD 04 C1 07 3D 65 F5 81 88 EE C1
The value that should be sent back to the PentaMetric is then
1D C0 52 A6 CB A3 4D 41

If no password is in use, use a key of all 0 bytes. Once the encrypted value has been sent back to the PentaMetric, one byte will come back. A "0" byte means the password was correct, "1" (or anything else) means the password was wrong.

If the password was wrong, the PentaMetric will disconnect the TCP connection. In this case, the whole connection operation repeated for another try. If three incorrect guesses are made in a row, the PentaMetric will not send any more random number challenges for another minute, so the user will have to wait.

BETA operation without passwords
Technically most of the code for passwords is already enabled. In this beta mode, the PentaMetric will always send the version *0F* plus the eight bytes *52 1A DD 8C 26 97 C7 80*, and

the client computer should respond with the eight bytes *EE 28 DA 94 8B 0F 87 3A* as long as the password is still set to all 0 bytes. Then, the PentaMetric will reply with a byte of 0 to indicate the password was correct.

Once the connection is established and the password procedure has completed, communications continues as with the serial port, except that any command must be preceded by one additional "magic cookie" byte. The PentaMetric does not care what the value of this byte is, but it will send it back out as the first byte of the response to the command that it was associated with. This can be used to keep track of many commands sent at once, so that if the network has high latency any command can be positively identified with its response. For example, ten requests can be issued at once, where the initial "cookie" byte increases by one for each request. The client program can then match each response to the request that corresponds to it simply by checking the "cookie" byte. The cookie byte must be included in the calculation of checksums.

The PentaMetric expects no more than 2 seconds to elapse between receiving each byte of a transmission to it. Additionally, the PentaMetric closes the TCP connection if no data is sent for more than one minute. To prevent this timeout from occurring, just read any memory location (like Volts) periodically.

The following table shows the new configuration sections added for the TCP/IP settings. These can be read and written just like all of the rest of the data in the PentaMetric, with the exception that the password location, which will always read as all 0's and can only be written if the Password Jumper (J5) is enabled (moved to the position closes to the Ethernet jack).

2. New PentaMetric program locations

The following settings/addresses are new:

Address (hex)	Byte range	Description	Data format
90	1	Internet flags	Set the lowest bit (bit 0) to enable DHCP, otherwise clear the lowest bit. Leave the other bits clear, at least for now.
	2-5	IP address	These four bytes represent the four bytes that make up the IP address, with the byte that is typically written first being stored at the lowest memory location
	6-9	Gateway address	Same as IP address
	10	Subnet mask	This is an integer with maximum (theoretical) value of 32 which specifies the number of zeros on the end of the subnet mask, for instance, 255.255.192.0 is represented by 14, since the binary representation this subnet mask is 11111111.11111111.11000000.00000000, which is a bunch of ones followed by 14 zeros
	11-14	Primary DNS server address	Same as IP address
91	1-4	Secondary DNS server address	Same as IP address
	5-6	Port number	Holds the two-byte port number that the PentaMetric listens for TCP connections on with the low byte at the lower address
92	1-16	NetBIOS name	16 character (max) ASCII string, which is filled with zero bytes for characters the user leaves blank
93	1-16	Password	16 character (max) ASCII string, which is filled with zeros for characters the user leaves blank. *Reading this location will always give all 0's*, and write access to this location can be prevented by moving a jumper on the circuit board; if disabled, writes will do nothing.