4720 SEAbus Protocol Reference Manual

Version 1.3
Document Revision Date:
May, 1995

For technical support contact the Customer Support department at one of the following locations:

Revision History

The following versions of this document have been released:

1.0	May, 1993	Initial Revision: 4720 V1.11x support.
1.1	February, 1994	Support for 4720 V1.20x. New features: • Time of Use • Predicted demand • New Date/Time display format (Group Keys) • Scaleable Status Input counters • Additional setpoint action keys • High speed kVA, Phase reversal, voltage unbalance measurements • Extended waveform recorder. New waveform recorder download method • High speed snapshot log support • Enhanced trigger identifications for logs • Diagnostic class added, with support for programmable features.
1.2	December, 1994	Support for 4720 V1.30x New features: • Programmable communication cache • Programmable delay from RTS assertion to start of transmission added. • Master Ack/Nak with retransmit now supported • Configurable password protection • Daylight Savings Time • Two cycle waveform capture • Independent configuration parameters added for thermal, and sliding demand measurements. • High speed frequency measurement
1.3	May, 1995	Support for 4720 V1.40x with Multiport Communications Card (MPCC)

The information contained in this document is believed to be accurate at the time of its publication; however, Siemens assumes no responsibility for any errors which may appear here and reserves the right to make changes without notice.

TABLE OF CONTENTS

1 INTRODUCTION

The SEAbus Protocol is a master-slave packet-based asynchronous serial communications method. The remote device is a slave in this scheme and responds only to master requests.

This protocol is based on the abbreviated OSI model. The Open Systems Interconnect (OSI) model is a standard reference for communications systems.

1.1 OSI Model

The Open System Interconnect (OSI) reference model defines a standard for open communications. The OSI model defines a seven layer stack in which each layer is a different level of abstraction. An individual layer (layer N) uses the services of the next lower layer (layer N-1) and in turn provides services to the next higher layer (layer N+1).

Each layer defines a protocol with which it communicates to the equal layer of a remote communication process. This peer to peer communication does not take place directly, instead, the lower layer services are used to accomplish this task.

The seven layers of the OSI model are (from lowest to highest): physical, data-link, network, transport, session, presentation, application. The physical layer is responsible for the electrical characteristics of the communications method. The data-link layer handles the transmission and reception of error-free small byte streams called frames. The network layer is responsible for the proper routing of data. The transport layer hides the lower three (hardware-specific) layers from the higher layers. The session layer handles synchronisation and token management. The presentation layer is used for encoding and decoding data-structures. The application layer provides the interface to the application process.

1.2 OSI Model on 4720

The 4720 protocol is an abbreviated version of the standard OSI model. The abbreviated OSI model provides the physical layer, the data-link layer and the application layer. Its primary use is in the remote data-acquisition and control field.

The application layer provides an interface which uses key and value associative array structures. The data-link layer protocol uses multiple frames to transmit and receive application layer messages. The physical layer is RS-232/RS-485 half-duplex communications.

The presentation layer is not included because there is little complexity in the application layer. The session layer is not needed as there is only one session (not multiple). The transport layer is not needed since the complexity of the protocol is not very high. The network layer is not needed as there is no routing of frames. The RS-485 architecture has each device on a single loop, there are no sub-loops or separate loops.

1.3 Transaction Overview

This protocol interface organises information in the form of key and value pairs. These are discussed in detail in the next section.

A typical transaction with the 4720 will follow the following steps:

- the user process builds up one or more messages which include a command (such as read or write) and a list of keys and/or values;
- the messages are then sent to the application layer of the protocol along with the destination address;
- the application layer then sends the messages as a single block of data to the data-link layer;
- the data-link layer splits the data block into several smaller frames and transmits each of them over the network (the slave device can send an acknowledgement frame back to the master on receipt of each frame);
- each frame from the master device is assembled in the data-link layer on the slave device (the 4720);
- the assembled data block is passed to the application layer;
- the application layer signals the user process that a request has arrived;
- the user process then parses through the request and follows the commands indicated in the request and builds response messages;
- the response messages are then sent to the application layer;
- the application layer then sends the response messages to the data-link layer as a single data block;
- the data-link layer splits the data block into smaller frames and transmits them to the master station;
- the master station's data-link layer rebuilds the data block and passes it onto the application layer;
- the application layer then passes the response message to the user process, who then has the response to its initial request.

The process then repeats as often as the master station requires information from any 4720.

2 OBJECT KEY AND VALUE MAPPING

All information in a 4720 is stored and retrieved in data units called *objects*. An object is an instance of a *class*, which defines the properties of the object. Each object contains many *elements*, such as present value and set-up structure, and objects from the same class have the same elements.

For example, an Analog Input is an object class; it has elements which include a value and two labels. There are eight instances of the Analog Input class. Analog Input #1 is an instance of a Analog Input.

An element of an object is referenced by an *object key* and the values read from or written to the object element are called *object values*.

2.1 Object Key

The object key is a three-byte quantity which specifies a particular element of a single object. The key has two components: the object identifier and the object qualifier. The object identifier is a two-byte value which identifies the object class as well as the instance of that object. The object qualifier is a single-byte number which represents the particular element of the object.

The object identifier is made up of the following fields:

Field:	Class	Sub-class	Instance
Bits:	4	4	8

The following objects are defined:

Class	Sub-class	Meaning
0	0-4,6	Globals
1	0-F	I/O
2	0-3	Constants
3	-	Reserved
4	0-F	Variables
5	0-F	Timestamps
6	3,7,B	Variable Extension
7	3,7,B	Variable Extension timestamps
8	0-2	Control
9	-	Reserved
A	0-5	Logs
B	0-B	Time of use
C-D	-	Reserved
E	0-2	Diagnostic information
F	-	Reserved

The object qualifier describes the element of the particular object. The following table lists the possible object qualifiers:

Qualifier	Meaning
00	Present Value
01	Time *
02	Set-up Structure
03	Label #1 (Name or Active label)
04	Label #2 (Inactive label)
05	Log Record
06	Command
07	Object Specific Structure
08	Time of use (TOU) structure
09-0B	Reserved for future use
0C	Communication cache programming

* Not all objects support the time qualifier. To insure compatibility with recent and future additions to the protocol, the *Timestamp* Class should be used to read time values. The *Timestamp* Class (5) has the same sub-classes and instances as the Variable class (4). The Predicted Demand *Timestamp* Class (7) has the same sub-classes and instances as the Predicted Demand class (6).

2.2 Object Value

The object value depends on the object class and the object qualifier. The possible object value data-types are:

- **INT32, INT16, INT8**: signed integers, 4 bytes, 2 bytes, and 1 byte respectively.
- **UINT32, UINT16, UINT8**: unsigned integers, 4 bytes, 2 bytes, and 1 byte respectively.
- **STCOUNT**: status counter--status = 2 bytes high, counter = 2 bytes low order
- **CNTSTAT**: counter status--counter = 2 bytes high, status = 2 bytes low order
- **SCALED_UINT30**: Represents an unsigned scaled 30 bit integer. The two high order bits indicate the negative of the exponent (base 10). For example. 0x80000001 represents the value 1E-2.
- **STRING**: character string--1 bytes length, X bytes of character (last byte = 0)
- **TIME**: 8 bytes: 4 bytes integer seconds, 4 bytes integer microseconds. The seconds field represent the number of seconds since January 1, 1970, commonly referred to as "UNIX time".
- **STRUCT**: pre-defined structure--any length

In some cases, two or more object classes can have the same instances. Thus, a single object can appear to belong to two or more classes. Each class may have different elements associated with it, or a different interpretation of a given element, thus this expands the number of ways in which an object can be accessed. For example, the present value element of a instance of Digital Input can be a status or counter, or combination of the two, depending upon which class is used.

Three special object values are defined. These special values apply to all object types, except SCALED_UINT30. The special object values are:

- **INVALID (0x80000000)**: The object has an invalid value. Objects with an INVALID number will not be returned by the application layer.
- **INVALID_MX (0x80000001):** The object has an invalid maximum value.
- **INVALID_MN (0x7FFFFFFF)**: The object has an invalid minimum value.

Invalid maximum and minimum values are assigned to min/max objects when the min/max objects are cleared. This value will remain assigned to the object until a new maximum or minimum occurs. Invalid maximum or minimum values are also assigned to measurements that are not complete at the time the measurements are read. Sliding window and predicted demand measurements are incomplete during the first demand period after sliding demand measurements are cleared, or during the first demand period after power-up.

2.3 Object Classes

The following sections explain the object keys and values that exist in the 4720.

2.3.1 Global Class

The following global objects are defined:

Class	Sub-class	Instance	Meaning
0	0	00	Null Object Identifier
0	1	00-31	Global codes
0	2	00-25	Global counters
0	3	00-01	Group Key definitions
0	4	00-04	Global Time values
0	5	-	Reserved
0	6	00	Daylight Savings Time Table

2.3.1.1 Global Codes

The global codes are used in event log messages, to identify global event causes and effects. They have no readable or writeable elements; the following instances are possible:

Class	Sub-class	Instance	type	Meaning
0	1	00	effect	Meter Power (0=down, 1=up)
0	1	01	effect	Meter Firmware Change (firmware upgrade)
0	1	02	effect	Factory Reset (clearing of all parameters)
0	1	03	cause	Externally caused event
0	1	04	cause	Front panel caused event
0	1	05	cause	Communications caused event
0	1	06	effect	Set-up parameter(s) changed
0	1	07	effect	Control has been initiated
0	1	08	effect	Labels have been changed
0	1	09	effect	Logs have been changed
0	1	0A-0F	-	Reserved

0	1	10	effect	Internal non-volatile RAM failure
0	1	11	effect	Internal initialisation failure
0	1	12	effect	Internal Interrupt Service Routine failure
0	1	13	cause	Real-time clock battery low
0	1	14	cause	Non-volatile ram battery low
0	1	15	effect	Time-Of-Use (TOU) calendar corruption detected
0	1	16	effect	TOU calendar has been changed
0	1	17	effect	TOU profile has been changed
0	1	18	effect	TOU register configuration changed
0	1	19-1F	-	Reserved
0	1	20	effect	TOU register has been reset
0	1	21	effect	Time left
0	1	22-26	-	Reserved
0	1	27	effect	User configurable task has been changed
0	1	28	-	Reserved
0	1	29	effect	User selectable feature set has been changed
0	1	2A-2F	-	Reserved
0	1	30	effect	PML programmable feature set has been changed
0	1	31	effect	PML programmable hardware options changed
0	1	32	cause	Daylight Savings Time (DST) activated
0	1	33	effect	DST Set-up Changed
0	1	34	effect	DST time adjusted by

2.3.1.2 Global Counters

The global counters are used for retrieving status or for controlling groups of objects. Only the present value qualifier (0) is supported; it is of type STCOUNT. The following instances are possible:

Class	Sub-class	Instance	Meaning
0	2	00	Setpoint counter--any setpoint changed state
0	2	01-02	Reserved
0	2	03	Sliding Demand counter--any change in sliding demand set-up
0	2	04	Event Log counter
0	2	05	Snapshot Log counter--any new snapshot log
0	2	06	Min/max Log counter--any new min/max
0	2	07	Waveform Capture Log counter
0	2	08	Waveform Recorder Log counter
0	2	09-0F	Reserved
0	2	10	Hours counter
0	2	11	All min/max counter (not the Min/Max Log)
0	2	12	Real-time min/max counter
0	2	13	Real-time Demand min/max counter
0	2	14	Sliding Demand min/max counter
0	2	15	Harmonic V1 min/max counter
0	2	16	Harmonic V2 min/max counter
0	2	17	Harmonic V3 min/max counter
0	2	18	Harmonic VAUX min/max counter

0	2	19	Harmonic V1 Demand min/max counter
0	2	1A	Harmonic V2 Demand min/max counter
0	2	1B	Harmonic V3 Demand min/max counter
0	2	1C	Harmonic VAUX Demand min/max counter
0	2	1D	Harmonic I1 min/max counter
0	2	1E	Harmonic I2 min/max counter
0	2	1F	Harmonic I3 min/max counter
0	2	20	Harmonic I4 min/max counter
0	2	21	Harmonic I1 Demand min/max counter
0	2	22	Harmonic I2 Demand min/max counter
0	2	23	Harmonic I3 Demand min/max counter
0	2	24	Harmonic I4 Demand min/max counter
0	2	25	Predicted demand min/max counter
0	2	26	Time-of-Use (TOU) Calendar counter
0	2	27	TOU Profile counter
0	2	28	TOU Register counter
0	2	29	TOU rate counter
0	2	2A	Daylight Savings Time (DST) counter

Performing a read of these counters will return a value that indicates how many updates or changes have been made to the corresponding group.

Performing a write to the Hours counter will reset all of the hours (KWhrs, KVAhrs, etc.) values to zero (0). Performing a write to the min/max counters will reset the min/max readings of the particular group. The log counters cannot be reset by writing the present values, however some of them can be reset via other mechanisms.

2.3.1.3 Group Key Definitions

The group key definitions are used to program the front panel group keys of the 4720. The only qualifier supported is the Object Specific Structure (07) qualifier. This group key structure is described as follows:

Bytes	Meaning
2	Number of Group Key functions (0-20d)
40d	Sequence of 2-byte descriptors
2	Number of 2-byte object identifiers
212d	Sequence of 2-byte object identifiers

The 2-byte descriptors are made up of the following fields:

Display Format	Number of identifiers/display	Number of modes	Number of phases
MSB		LSB	
4 bits	4 bits	4 bits	4 bits

The possible display formats are:

Value	Meaning
0	Normal Display - LLLLLLLLLL=SDDDDDDD
1	Volt-amp Display - VVVV P AAAA SDDDDLLL
2	Normal PF Display - LLLLLLLLLL=GGDDDDD
3	Volt-amp PF Display - VVVV P AAAA GGDDDLLL
4	Hour Display - LLLLLLLL=SDDDDDDDDD
5	Percentage Display (harmonics)- LLLLLLLLLL= SDDDDD%
6	Triple Display - LLLLL DDDD DDDD DDDD
7	Label Display - LLLLLLLLLLLLLLLLLL
8	Date/Time Display - WWW MMM XX HH/MM/SS

The characters shown in the above table represent the display on the front panel. Each character represents a display alphanumeric. The L indicates the parameter's label; the S indicates a sign (- if negative, space if positive); the D indicates the value's digits; the V indicates a voltage parameter; the P indicates the phase indicator; the A indicates a current parameter; and the GG is LD for leading, LG for lagging and two spaces for neither. WWW indicates day of week (MON,TUE,WED,THU,FRI,SAT,SUN), MMM indicates the month (JAN,FEB,MAR,APR,MAY,JUN,JUL,AUG,SEP,OCT,NOV,DEC), XX indicates day of the month. HH indicates the hour of the current day (0-23), MM indicates the minute of the current hour (0-59) and SS indicate the number of seconds of the current minute (0-59). The Date/Time display format only functions with global time object 0400.

The group key structure is essentially a three dimensional array. Along the X axis is the function, along the Y axis is the mode and along the Z axis is the phase. Pressing the group key button (GROUP1 or GROUP2) moves the display along the X axis thereby changing the present function. Pressing the MODE button (GROUP1/GROUP2 + FUNCTION) moves the display along the Y axis thereby changing the present mode. Pressing the PHASE button traverses the Z axis thereby changing the present phase.

The number of group key functions in the data-structure defines the X axis length. The number of modes and number of phases define the Y and Z axis (respectively) for each function entry. All present locations wrap around to zero (0) when the length is exceeded. In addition, the present mode (Y) and present phase (Z) are set to zero (0) when the present function (X) is incremented.

The sequence of object identifiers is a flattened representation of this three-dimensional structure. The easiest explanation is by example. The example display operation is to show three function groups: Volts LN triple A/B/C; Amps Avg/A/B/C and demand Avg/A/B/C; and KWH Net. The following group definition would be used:

Bytes (dec)	Value (hex)	Meaning
0,1	0003	Number of Group Key functions (3)
2-41	----	Sequence of 2-byte descriptors, as follows:
2,3	6311	Triple display, 3 id's/display, 1 mode, 1 phase
4,5	0124	Normal display, 1 id/display, 2 modes, 4 phases
6,7	4111	Hour display, 1 id/display, 1 mode, 1 phase
8-41	*Don't care*	Unused
42,43	000C	Number of 2-byte object identifiers (12)
44-255	----	Sequence of 2-byte object identifiers, as follows:

44,45	4101	Volts LN phase A*
46,47	4102	Volts LN phase B*
48,49	4103	Volts LN phase C*
50,51	4108	Amps Avg*
52,53	4109	Amps Phase A*
54,55	410A	Amps Phase B*
56,57	410B	Amps Phase C*
58,59	4208	Amps Demand Avg*
60,61	4209	Amps Demand A*
62,63	420A	Amps Demand B*
64,65	420B	Amps Demand C*
66,67	4C10	KW Hours Net*
68-255	*Don't care*	Unused

the value shown here is explained in section 2.3.4 Variable Class.

Note: two-byte values are ordered as low-byte first. For example, bytes 00-01 contain 0003 in the order: 03, 00.

The 4720 uses the descriptor, present function, present mode and present phase to index into the correct object identifier in the sequence.

2.3.1.4 Global Time

The Global Time objects support the present value qualifier (0) as a INT32 and are defined as follows:

Class	Sub-class	Instance	Meaning
0	4	00	Time in seconds since January 1 1970 00:00:00
0	4	01	Time in micro-seconds since current second
0	4	02	Reserved
0	4	03	Alarm status indicator

The alarm status indicator provides a mechanism for quickly checking the status of the internal alarms (setpoints), as well as the digital inputs (status inputs). The alarm status object is defined as follows:

Bit Location (0 = LSB, 31 = MSB)	Function
0 - 16	Setpoint #1 to #17 1 = Active, 0 = Inactive
17 - 19	Reserved
20	Low battery warning
21 - 24	Status input #1 to #4 1 = Active, 0 = Inactive
25 - 31	Reserved

2.3.1.5 Daylight Savings Time (DST)

Subclass 6 contains the Daylight Savings Time object (0bject 0x0600). The Daylight Savings Time object provides a mechanism to adjust the 4720's internal time automatically to account for Daylight Savings Time (DST). The DST object consists of a table of time changes (4 maximum). Each time change contains the activation time, and the adjusted time. Upon reaching the activation time, the 4720's internal clock is set to the adjusted time.

The Daylight Savings Time (DST) objects support the set-up qualifier (2). The only DST object supported in instance 00, the Daylight Savings Time table. The DST table structure has the following components:

Bytes	Data-type	Meaning
2	INT16	Must be 0x0000
1	INT8	DST Set-up identifier
1	INT8	Number of time changes (N)
4	INT32	No time set window
8	STRUCT	Time changeover #1
8	STRUCT	Time changeover #2
8	STRUCT	Time changeover #3
8	STRUCT	Time changeover #4

The time changeover structure has the following components:

Bytes	Data-type	Meaning
4	INT32	Time of change in UNIX time
4	INT32	New time in UNIX time.

A maximum of four time changes can be programmed. If the number of time changed (N) indicated are less than four, only the first N time changes are used. The remained 4-N time changes should be set to 0. To disable automatic DST time changes, set the number of time changes to 0.

The *DST set-up identifier* identifies the DST table currently programmed in the meter. If the DST Set-up identifier matches the DST table programmed in the meter, the table will **NOT** be reprogrammed.

The *No time set window* controls the size of the window, around the time of changeover, when time set commands issued through communication are ignored. This value in this field represents the number of seconds before and after a time change during which time set commands are ignored.

The 4720 supports a resolution of 15 minutes for the Time of change.

2.3.2 I/O Class

The I/O objects have the following object identifiers:

Class	Sub-class	Instance	Meaning

1	0	00-05	Digital Inputs (Status Inputs)
1	1	00-02	Digital Outputs (Relays)
1	2	00-07	Analog Inputs (Voltage & Current Inputs)
1	3	00	Analog Outputs (IOUT)
1	4	00-05	Digital Inputs (Status Inputs) -- Status
1	5	00-02	Digital Outputs (Relays) -- Status
1	8	00-05	Digital Inputs (Status Inputs) -- Counter
1	9	00-02	Digital Outputs (Relays) -- Counter
1	C	00-05	Digital Inputs (Status Inputs) -- Preset/Reset
1	D	00-02	Digital Outputs (Relays) -- Reset
1	E	00-03	Digital Inputs (Status Inputs) -- Scale
1	F	00-03	Digital Inputs (Status Inputs) -- Rollover

The digital output instances 0 through 2 correspond to relays 1 through 3. The digital input instances are defined according to the table below:

Instance	Meaning
0	Status input 1
1	Status input 2
2	Status input 3
3	Status input 4
4	Logical OR of all status inputs
5	Logical NAND of all status inputs

The Digital Inputs and Outputs are broken up into several groups of subclasses. The straight Digital Inputs and Outputs (Sub-classes 0 and 1) have the following qualifiers:

Qualifier	Data-type	Description
Present Value	0	STCOUNT (status=on/off, counter=off→on transitions)
Label #1	3	Active label (STRING up to 20 characters)
Label #2	4	Inactive label (STRING up to 20 characters)

The Digital Input counts obtained from Sub-classes 0 and 1 represent the least significant word (16 bits) of the SCALED_UINT30 counters available through Sub-class 8.

Status and Counter subclasses (4,5,9) refer to objects that are subsets of the STCOUNT data-type. These objects can only be accessed via the Present Value qualifier (0). Their data-type is an INT32, of which the lower 2-bytes define the status or counter value. For the Relays, a write to the status object is used to force the relay state. The following write values are supported by the Relays:

Value	Description
0	Normal relay control
1	Force normally open relay closed
2	Force normally open relay open

The relay must be defined as *setpoint* triggered, otherwise the 4720 will ignore the command.

A read of the relay status object returns the status of the relay. The status of the relay should be interpreted as follows:

Bit	Description	Interpretation
0	RELAY ON	Current status of relay (0 = off, 1 = on)
1	RELAY FORCED	Manually forced relay (0 = normal, 1 = forced)
2	RELAY PULSED	Relay operation (0 = latched, 1 = pulsed)
3	RELAY ACTIVE	Status relay should be in under setpoint control

Digital Input Sub-classes E and F are used to define the scale and rollover value for each Digital Input Counter. These objects are of type SCALED_UINT30. Each positive transition the count will advance by the scaling value. Should the total count exceed the rollover value, the count will be set to 0 (exponent will be maintained).

The Status and Counter objects are also used to define setpoints and for the cause and effect fields of event records. See the section on setpoints and events for more information.

The Preset object identifiers for the Digital Inputs and Digital Outputs are used to preset the counter values. These objects support the Present Value Qualifier (0) and Command qualifier (6) which presets the counter to the SCALED_UINT30 value passed. If no value is passed, the counter will be cleared. Instance 4 can be used to Preset/Clear all of the counters. The reset object identifier will also be placed in the effect field of event records that are generated by the input counter(s) being Preset/Cleared.

The Analog Inputs and Analog Outputs presently only support the Present Value qualifier, which is of data-type of INT32. The Analog Inputs correspond to the raw A/D inputs of the eight voltage and current inputs. Generally, there is no reason to read these inputs, as the information is available elsewhere in processed form. The Analog Output is the IOUT output, which presently cannot be written.

2.3.3 Constant Class

The following constant objects are defined:

Class	Sub-class	Instance	Meaning
2	0	00-59	Device Configuration
2	1	00-80	Communication Port 1
2	2	00-80	Communication Port 2
2	3	00-80	Communication Port 3

2.3.3.1 Device Configuration

The device configuration constant objects are all of class=2 and sub-class=0. The device configuration object control the primary configuration of the 4720. The following object instances are defined:

Instance	Meaning
00	Device Type (4720d) (Read Only)
01	Firmware Revision (in hex, e.g. rev 1.1.1.1 = 1111h) (read-only)
02	Custom Version Designator (ASCII, read-only)
03	Serial Number (read-only)
04	Unit ID (1-9999)
05	Baudrate (0=300 1=1200 2=2400 3=4800 4=9600 5=19200 6=38400 7=57600 8=115200)
06	Communication Mode (0=RS-232 1=RS-485)
07	RTS line Mode (0=Active Low 1=Active High)
08	Display Time-out (0-999 minutes, 0=stay on)
09	Phase Labels (0=ABC 1=XYZ 2=RYB 3=RST)
0A	Numeric Format (0=1,234.5 1=1234,5)
0B	Use Ack/Nak handshaking (0=No, 1=Yes)
0C	Password (0-9999)
0D	Communication card type (0=None 1=ISOCOM 2=MPCC) (read-only)
0E	Communication card firmware revision (in hex, ex. V1.111 = 1111h) (read-only)
0F	Extended Front Panel (0=no, 1=yes--two outer keys for diags)
10	Volts Mode (0=4w-Wye 1=delta 2=single 3=demo 4=3w-Wye
11	Phase Rotation (0=ABC 1=ACB)
12	Standard Frequency (0=60 Hz 1=50 HZ)
13	Reserved
14	Volt Scale (1-999999)
15	Reserved
16	Current Scale (1-30000)
17	Reserved
18	Neutral Current Scale (1-9999)
19-1F	Reserved
20	Demand Period (0-99 minutes, 0=external sync) (see note 2)
21	Number of Demand Periods (1-15) (see note 2)
22	Prediction Base (0-99 %, 0=off)
23	Reserved
24	Sliding demand period (0-99 minutes, 0=Off)
25	Sliding demand synchronisation (0=Internal, 1=External)
26	Thermal demand period (0-99 minutes, 0=off)
27	Reserved
28	VAUX Full Scale (0-999999)
29	VAUX Zero Scale (-999999-999999)
2A-2B	Reserved
2C	IOUT Full Scale (0-999999)
2D	IOUT Zero Scale (-999999-999999)

2E	IOUT Key (see note 1)
2F	IOUT Range (0=0-20mA 1=4-20mA)
30	I^2T Amps #1 (0.0-1000.0 times tap current) (X 10)
31	I^2T Time #1 (0-10000 milliseconds)
32	I^2T Amps #2 (0.0-1000.0 times tap current) (X 10)
33	I^2T Time #2 (0-10000 milliseconds)
34	I^2T Amps #3 (0.0-1000.0 times tap current) (X 10)
35	I^2T Time #3 (0-10000 milliseconds)
36	I^2T Amps #4 (0.0-1000.0 times tap current) (X 10)
37	I^2T Time #4 (0-10000 milliseconds)
38	I^2T Amps #5 (0.0-1000.0 times tap current) (X 10)
39	I^2T Time #5 (0-10000 milliseconds)
3A	I^2T Amps #6 (0.0-1000.0 times tap current) (X 10)
3B	I^2T Time #6 (0-10000 milliseconds)
3C	I^2T Amps #7 (0.0-1000.0 times tap current) (X 10)
3D	I^2T Time #7 (0-10000 milliseconds)
3E	I^2T Amps #8 (0.0-1000.0 times tap current) (X 10)
3F	I^2T Time #8 (0-10000 milliseconds)
40	I^2T Tap (0-30000 Amps)
41	I^2T Number of points (0-8)
42	Depth of waveform recorder
43-47	Reserved
48	Log Status #1 (0=don't log Status Input #1 changes, 1=log changes)
49	Log Status #2 (0=don't log Status Input #2 changes, 1=log changes)
4A	Log Status #3 (0=don't log Status Input #3 changes, 1=log changes)
4B	Log Status #4 (0=don't log Status Input #4 changes, 1=log changes)
4C-4F	Reserved
50	Sliding Window #1 Key (4300-43FF)
51	Sliding Window #2 Key (4300-43FF)
52	Sliding Window #3 Key (4300-43FF)
53	Sliding Window #4 Key (4300-43FF)
54	Sliding Window #5 Key (4300-43FF)
55	Sliding Window #6 Key (4300-43FF)
56	Sliding Window #7 Key (4300-43FF)
57	Sliding Window #8 Key (4300-43FF)
58	Sliding Window #9 Key (4300-43FF)
59	Sliding Window #10 Key (4300-43FF)

The numbers in parentheses represent the range of the present value. The lowest and highest numbers are the minimum and maximum, respectively, of the set-up structure.

The waveform recorder depth controls the number of recorders that can be stored in non-volatile RAM. If the depth is reduced, the length of each recorder increases. The depth can be programmed to 1 (1 recorder of 36 cycles), 2 (2 recorders of 18 cycles), or 3 (3 recorders of 12 cycles).

The Sliding Window Keys are used to determine which of the Variable Class objects will have Sliding Window Demand and Predicted Sliding Demand calculated, since the 4720 will only

calculate Sliding Window Demand and Predicted Sliding Window Demand for 10 Variable Class Objects. The values for the Sliding Window Keys are the keys for the Variable Class Objects, of sub-class Sliding Window Demand Present.

The constant class supports the following qualifiers:

Qualifier	Data-type
Present Value (0)	INT32
Set-up Structure (2)	STRUCT 8-byte
Label #1 (3)	STRING

The Present Value is used to read and write 4720 configuration parameters. The set-up structure is used to do range checking of the values, and is read-only. It has the following elements:

Bytes	Meaning
4	Minimum value; INT32
4	Maximum value; INT32

The minimum and maximum values define the valid ranges that a constant object's Present Value may obtain.

Notes:

1. The IOUT key is used to determine which real time value is assigned to the current loop output. This key may have the following values:

IOUT Key	Parameter
0	Real time Volts Phase A
1	Real time Volts Phase B
2	Real time Volts Phase C
3	Real time Amps Phase A
4	Real time Amps Phase B
5	Real time Amps Phase C
6	Real time kW Phase A
7	Real time kW Phase B
8	Real time kW Phase C
9	Real time kVA Phase A
10	Real time kVA Phase B
11	Real time kVA Phase C
12	Real time kVAR Phase A
13	Real time kVAR Phase B
14	Real time kVAR Phase C
15	Real time Volts Average
16	Real time Current Average
17	Real time kW Total
18	Real time kVA Total
19	Real time kVAR Total

20	Real time PF Total
21	Real time Sliding window Parameter #1
22	Real time Sliding window Parameter #2
23	Real time Frequency (phase A)
24	Real time V Aux.
25	Real time Neutral Current

2. Independent Sliding and Thermal demand constants have been added in release V1.300. To maintain backwards compatibility, objects 0x2020 is still available. Values written to 0x2020 will result in the value of 0x2020 * 0x2021 automatically being written to the Thermal demand period object (0x2025). The sliding demand synchronisation is automatically set to EXTERNAL if a value of zero is written to object 0x2020.

2.3.3.2 Communication Ports A-C

Constant sub-object 01-03 contain the configuration for communication port A-C. The number of communication ports supported depends on the communication card installed in the 4720. The following communication cards are supported by V1.40x.

- ISOCOMM 2: Supports one RS-232/RS-485 communication port (Port A).
- MPCC: Supports one RS-232 (Port A), and two RS-485 communication ports (Port B & C)

The communications configuration objects support the Present Value (0x00) and Cache programming (0x0C) qualifier (only supported on instance 0x80). The following instances are supported by each communication port:

Instance	Protocol	Meaning
00	All	Active protocol (0=None, 1=PML 4720, 2=Modbus)
01	PML 4720	Unit ID (1-9999)
02	Modbus	Unit ID (1-254)
03	All	Baudrate (0=300, 1=1200, 2=2400, 3=4800, 4=9600, 5=19200, 6=38400, 7=57600, 8=115200)
04	All	Communications mode (0=RS-232, 1=RS-485)
05	All	RTS line mode (0=Active low, 1=Active high) (RS-485 only)
06	Modbus	Register size (0=16 bits, 1=32 bits)
07	Modbus	Return invalid objects (0=No, 1=Yes)
08	PML4720	Use Ack/Nak (0=No, 1=Yes)
09	All	Password protect (0=No, 1=Yes)
0A	All	Transmit delay (0-999 ms)
0B	All	Carrier Detect (0=No, 1=Yes) (MPCC Port A only)
0C-7F	-	Reserved
80	PML 4720	Communication cache message (see section 2.3.3.3)
81-FF	-	Reserved

Notes:

1. V1.40x supports the PML 4720, and Modicon Modbus protocols. The active protocol is selected through object 2X00h. Selecting protocol NONE disables the communication port.

2. Some instances are only supported by one specific protocol. Consult the protocol field for each instance.

3. The scope of this document includes only the PML 4720 protocol. To obtain documentation for other protocols, contact the nearest Power Measurement Customer Support department at one of the locations listed in the front of this document.

4. Use Ack/Nak enables Master Ack/Nak acknowledgements of the data link layer. This option is only available with the ISOCOM communications card. The MPCC communication card will automatically enable Ack/Nak on multiple frame packets.

5. Transmit delay controls the time between asserting the Request to Send (RTS) line and the start of transmission. Transmit delay can be configured from 0-999 ms. On RS-485 ports, this feature can be used to delay transmission for use in radio modem applications.

6. Password protect enables password protection on all write requests, and commands.

7. The communication cache message is not supported by the MPCC.

2.3.3.3 Programmable Communication Cache (not supported on MPCC)

The 4720, release V1.40X is equipped with one programmable communications cache on port one. The communication response time for cached messages is significantly faster than the response time for non-cached messages. Depending on the cached message size, an average reduction of 40-50% in response time can be expected.

The following rules have to be observed when programming the cache buffer:

- The cache request buffer has a maximum size of 1 frame. Multi-frame packets **cannot** be cached. The maximum length of the cached response **cannot** exceed 256 bytes, or 40 objects, whichever comes first.
- Not all messages are cacheable. Only messages with object qualifiers Present Value (0x00), Time (0x01) can be programmed in the cache.
- Cacheable messages are limited to messages having a fixed structure.
- A cache message identification number must be programmed with each cached message.
- The communication cache is only available for use with the ISOCOM communication card.

The communication cache supports only the Cache Programming (0x0C) qualifier. The cache programming structure contains the following elements:

Bytes	Data type	Meaning
1	UINT8	Cache ID number
1	UINT8	Reserved
N (max 256)	Application Layer message	Application Layer message

The cache identification number is used by the data link layer to identify cache requests. If the data link layer Cache ID matches the cache ID programmed in the cache buffer, a cache response will occur, otherwise the application layer message will be parsed normally.

The cache message is guaranteed to update once per second.

To avoid unnecessary loading on the 4720, the cache will place itself automatically in a dormant state when no cache request has been received for five minutes. The cache will be reawakened when a cache request is received.

Note: The communication cache message is erased when the meter's power is cycled.

2.3.4 Variable Class

The class component of variable object key is 4. The sub-class component describes the measurement method, and the instance specifies the particular measured parameter. The sub-classes are as follows:

Sub-class	Meaning
0	High-speed Present
1	Standard Present
2	Thermal Demand Present
3	Sliding Window Demand Present
4	High-speed Minimum
5	Standard Minimum
6	Thermal Demand Minimum
7	Sliding Window Demand Minimum
8	High-speed Maximum
9	Standard Maximum
A	Thermal Demand Maximum
B	Sliding Window Demand Maximum
C	Hours - Net (Import - Export)
D	Hours - Import
E	Hours - Export
F	Hours - Total (Import + Export)

Using these abbreviations:

HS = High-speed present/min/max
STD = Standard present/min/max
TD = Thermal demand present/min/max
SD = Sliding window demand present/min/max
PD = Predicted Sliding demand present/min/max
HRS = Hours

The following table describes the instance values and supported modes (HD=Harmonic Distortion):

Instance	Measurement	Supported Modes
00	Volts LN Average	HS STD TD SD PD
01	Volts LN Phase A	HS STD TD SD PD
02	Volts LN Phase B	HS STD TD SD PD
03	Volts LN Phase C	HS STD TD SD PD
04	Volts LL Average	HS STD TD SD PD
05	Volts LL Phase AB	HS STD TD SD PD
06	Volts LL Phase BC	HS STD TD SD PD
07	Volts LL Phase CA	HS STD TD SD PD
08	Amps Average	HS STD TD SD PD
09	Amps Phase A	HS STD TD SD PD
0A	Amps Phase B	HS STD TD SD PD
0B	Amps Phase C	HS STD TD SD PD
0C	Amps Neutral	HS STD TD SD PD
0D	Reserved	
0E	Volts Imbalance (0-100)	HS STD TD SD PD
0F	Amps Imbalance (0-100)	STD TD SD PD
10	kW Total	HS STD TD SD PD HRS
11	kW Phase A	HS STD TD SD PD
12	kW Phase B	HS STD TD SD PD
13	kW Phase C	HS STD TD SD PD
14	kVAR Total	STD TD SD PD HRS
15	kVAR Phase A	STD TD SD PD
16	kVAR Phase B	STD TD SD PD
17	kVAR Phase C	STD TD SD PD
18	kVA Total	HS STD TD SD PD HRS
19	kVA Phase A	HS STD TD SD PD
1A	kVA Phase B	HS STD TD SD PD
1B	kVA Phase C	HS STD TD SD PD
1C	PF Total	STD TD SD PD
1D	PF Phase A	STD TD SD PD
1E	PF Phase B	STD TD SD PD
1F	PF Phase C	STD TD SD PD
20	Frequency	HS STD TD SD PD
21-23	Reserved	
24	Phase Reversal (0 or 1)	HS STD
25-27	Reserved	
28	VAUX	STD TD SD PD
29-2F	Reserved	
30	I2T Avg. (0 = Off, 1= On)	HS
31	I2T Phase A (0=Off, 1=On)	HS
32	I2T Phase B (0=Off, 1=On)	HS

33	I2T Phase C (0=Off, 1=On)	HS
34-67	Reserved	
68	V1 HD - K-Factor	STD TD SD PD
69	V2 HD - K-Factor	STD TD SD PD
6A	V3 HD - K-Factor	STD TD SD PD
6B	VAUX HD - K-Factor	STD TD SD PD
6C	I1 HD - K-Factor	STD TD SD PD
6D	I2 HD - K-Factor	STD TD SD PD
6E	I3 HD - K-Factor	STD TD SD PD
6F	I4 HD - K-Factor	STD TD SD PD
70	V1 HD - Total Odd	STD TD SD PD
71	V2 HD - Total Odd	STD TD SD PD
72	V3 HD - Total Odd	STD TD SD PD
73	VAUX HD - Total Odd	STD TD SD PD
74	I1 HD - Total Odd	STD TD SD PD
75	I2 HD - Total Odd	STD TD SD PD
76	I3 HD - Total Odd	STD TD SD PD
77	I4 HD - Total Odd	STD TD SD PD
78	V1 HD - Total Even	STD TD SD PD
79	V2 HD - Total Even	STD TD SD PD
7A	V3 HD - Total Even	STD TD SD PD
7B	VAUX HD - Total Even	STD TD SD PD
7C	I1 HD - Total Even	STD TD SD PD
7D	I2 HD - Total Even	STD TD SD PD
7E	I3 HD - Total Even	STD TD SD PD
7F	I4 HD - Total Even	STD TD SD PD
80	V1 HD - Total	STD TD SD PD
81	V2 HD - Total	STD TD SD PD
82	V3 HD - Total	STD TD SD PD
83	VAUX HD - Total	STD TD SD PD
84	I1 HD - Total	STD TD SD PD
85	I2 HD - Total	STD TD SD PD
86	I3 HD - Total	STD TD SD PD
87	I4 HD - Total	STD TD SD PD
88	V1 HD - Harmonic #1	STD TD SD PD
89	V2 HD - Harmonic #1	STD TD SD PD
8A	V3 HD - Harmonic #1	STD TD SD PD
8B	VAUX HD - Harmonic #1	STD TD SD PD
8C	I1 HD - Harmonic #1	STD TD SD PD
8D	I2 HD - Harmonic #1	STD TD SD PD
8E	I3 HD - Harmonic #1	STD TD SD PD
8F	I4 HD - Harmonic #1	STD TD SD PD
90	V1 HD - Harmonic #2	STD TD SD PD
91	V2 HD - Harmonic #2	STD TD SD PD
92	V3 HD - Harmonic #2	STD TD SD PD
93	VAUX HD - Harmonic #2	STD TD SD PD
94	I1 HD - Harmonic #2	STD TD SD PD

95	I2 HD - Harmonic #2	STD TD SD PD
96	I3 HD - Harmonic #2	STD TD SD PD
97	I4 HD - Harmonic #2	STD TD SD PD
98	V1 HD - Harmonic #3	STD TD SD PD
99	V2 HD - Harmonic #3	STD TD SD PD
9A	V3 HD - Harmonic #3	STD TD SD PD
9B	VAUX HD - Harmonic #3	STD TD SD PD
9C	I1 HD - Harmonic #3	STD TD SD PD
9D	I2 HD - Harmonic #3	STD TD SD PD
9E	I3 HD - Harmonic #3	STD TD SD PD
9F	I4 HD - Harmonic #3	STD TD SD PD
A0	V1 HD - Harmonic #4	STD TD SD PD
A1	V2 HD - Harmonic #4	STD TD SD PD
A2	V3 HD - Harmonic #4	STD TD SD PD
A3	VAUX HD - Harmonic #4	STD TD SD PD
A4	I1 HD - Harmonic #4	STD TD SD PD
A5	I2 HD - Harmonic #4	STD TD SD PD
A6	I3 HD - Harmonic #4	STD TD SD PD
A7	I4 HD - Harmonic #4	STD TD SD PD
A8	V1 HD - Harmonic #5	STD TD SD PD
A9	V2 HD - Harmonic #5	STD TD SD PD
AA	V3 HD - Harmonic #5	STD TD SD PD
AB	VAUX HD - Harmonic #5	STD TD SD PD
AC	I1 HD - Harmonic #5	STD TD SD PD
AD	I2 HD - Harmonic #5	STD TD SD PD
AE	I3 HD - Harmonic #5	STD TD SD PD
AF	I4 HD - Harmonic #5	STD TD SD PD
B0	V1 HD - Harmonic #6	STD TD SD PD
B1	V2 HD - Harmonic #6	STD TD SD PD
B2	V3 HD - Harmonic #6	STD TD SD PD
B3	VAUX HD - Harmonic #6	STD TD SD PD
B4	I1 HD - Harmonic #6	STD TD SD PD
B5	I2 HD - Harmonic #6	STD TD SD PD
B6	I3 HD - Harmonic #6	STD TD SD PD
B7	I4 HD - Harmonic #6	STD TD SD PD
B8	V1 HD - Harmonic #7	STD TD SD PD
B9	V2 HD - Harmonic #7	STD TD SD PD
BA	V3 HD - Harmonic #7	STD TD SD PD
BB	VAUX HD - Harmonic #7	STD TD SD PD
BC	I1 HD - Harmonic #7	STD TD SD PD
BD	I2 HD - Harmonic #7	STD TD SD PD
BE	I3 HD - Harmonic #7	STD TD SD PD
BF	I4 HD - Harmonic #7	STD TD SD PD
C0	V1 HD - Harmonic #8	STD TD SD PD
C1	V2 HD - Harmonic #8	STD TD SD PD
C2	V3 HD - Harmonic #8	STD TD SD PD
C3	VAUX HD - Harmonic #8	STD TD SD PD

C4	I1 HD - Harmonic #8	STD TD SD PD
C5	I2 HD - Harmonic #8	STD TD SD PD
C6	I3 HD - Harmonic #8	STD TD SD PD
C7	I4 HD - Harmonic #8	STD TD SD PD
C8	V1 HD - Harmonic #9	STD TD SD PD
C9	V2 HD - Harmonic #9	STD TD SD PD
CA	V3 HD - Harmonic #9	STD TD SD PD
CB	VAUX HD - Harmonic #9	STD TD SD PD
CC	I1 HD - Harmonic #9	STD TD SD PD
CD	I2 HD - Harmonic #9	STD TD SD PD
CE	I3 HD - Harmonic #9	STD TD SD PD
CF	I4 HD - Harmonic #9	STD TD SD PD
D0	V1 HD - Harmonic #10	STD TD SD PD
D1	V2 HD - Harmonic #10	STD TD SD PD
D2	V3 HD - Harmonic #10	STD TD SD PD
D3	VAUX HD - Harmonic #10	STD TD SD PD
D4	I1 HD - Harmonic #10	STD TD SD PD
D5	I2 HD - Harmonic #10	STD TD SD PD
D6	I3 HD - Harmonic #10	STD TD SD PD
D7	I4 HD - Harmonic #10	STD TD SD PD
D8	V1 HD - Harmonic #11	STD TD SD PD
D9	V2 HD - Harmonic #11	STD TD SD PD
DA	V3 HD - Harmonic #11	STD TD SD PD
DB	VAUX HD - Harmonic #11	STD TD SD PD
DC	I1 HD - Harmonic #11	STD TD SD PD
DD	I2 HD - Harmonic #11	STD TD SD PD
DE	I3 HD - Harmonic #11	STD TD SD PD
DF	I4 HD - Harmonic #11	STD TD SD PD
E0	V1 HD - Harmonic #12	STD TD SD PD
E1	V2 HD - Harmonic #12	STD TD SD PD
E2	V3 HD - Harmonic #12	STD TD SD PD
E3	VAUX HD - Harmonic #12	STD TD SD PD
E4	I1 HD - Harmonic #12	STD TD SD PD
E5	I2 HD - Harmonic #12	STD TD SD PD
E6	I3 HD - Harmonic #12	STD TD SD PD
E7	I4 HD - Harmonic #12	STD TD SD PD
E8	V1 HD - Harmonic #13	STD TD SD PD
E9	V2 HD - Harmonic #13	STD TD SD PD
EA	V3 HD - Harmonic #13	STD TD SD PD
EB	VAUX HD - Harmonic #13	STD TD SD PD
EC	I1 HD - Harmonic #13	STD TD SD PD
ED	I2 HD - Harmonic #13	STD TD SD PD
EE	I3 HD - Harmonic #13	STD TD SD PD
EF	I4 HD - Harmonic #13	STD TD SD PD
F0	V1 HD - Harmonic #14	STD TD SD PD
F1	V2 HD - Harmonic #14	STD TD SD PD
F2	V3 HD - Harmonic #14	STD TD SD PD

F3	VAUX HD - Harmonic #14	STD TD SD PD
F4	I1 HD - Harmonic #14	STD TD SD PD
F5	I2 HD - Harmonic #14	STD TD SD PD
F6	I3 HD - Harmonic #14	STD TD SD PD
F7	I4 HD - Harmonic #14	STD TD SD PD
F8	V1 HD - Harmonic #15	STD TD SD PD
F9	V2 HD - Harmonic #15	STD TD SD PD
FA	V3 HD - Harmonic #15	STD TD SD PD
FB	VAUX HD - Harmonic #15	STD TD SD PD
FC	I1 HD - Harmonic #15	STD TD SD PD
FD	I2 HD - Harmonic #15	STD TD SD PD
FE	I3 HD - Harmonic #15	STD TD SD PD
FF	I4 HD - Harmonic #15	STD TD SD PD

2.3.5 Variable Extension Class

The Variable Extension class is an extension of the Variable class. The class component of the Variable Extension class object key is 6. The sub-class component describes the measurement method, and the instance specifies the particular measured parameter. The sub-classes supported by this class are listed below.

Sub-class	Meaning
0-2	Reserved
3	Predicted Sliding Window Demand Present
4-6	Reserved
7	Predicted Sliding Window Demand Minimum
8-A	Reserved
B	Predicted Sliding Window Demand Maximum
C-F	Reserved

Notes:

1. The valid voltage readings depend on the Voltage mode parameter (Constant Class). For high-speed voltages, only the line-neutral readings are valid in the 4W-WYE, 3W-WYE, SINGLE and DEMO modes; only the line-line readings are valid in DELTA mode. For standard voltages, line-line readings are valid in all modes; line-neutral readings are valid in 4W-WYE, 3W-WYE, SINGLE and DEMO modes.

2. Any Phase C readings are invalid in Volts mode of SINGLE. Also, line-line voltage of BC is invalid.

If an invalid object is requested, the 4720 will NAK the entire data request

3. Harmonic values are passed as times 10 (including K-Factor). Frequency is passed as times 100. For example, 60.00 Hz is passed as 6000.

4. PF is passed as a value between 0 and 200. This value is interpreted as follows:

PF	Meaning
0-100	Power Factor = PF (lagging)
100-200	Power Factor = 200 - PF (leading)

The 4720 interprets an inductive load as having a lagging power factor and a positive reactive power value (see the installation manual for more details).

5. Sliding window demand modes are only valid if they have been programmed in the Constant Class objects for Sliding Window Demand. Any parameter supporting the standard mode (type STD) are possible candidates for Sliding Window Demand.

6. Predicted Sliding window demand modes are only valid if they have been programmed in the Constant Class objects for Sliding Window Demand. Any parameter supporting the standard mode (type STD) are possible candidates for Predicted Sliding Window Demand.

Variable classes have the following qualifiers:

Qualifier	Data-type
Data (0)	INT32 (may represent times 10 or times 100 value)
Time (1)	timestamp for minimum and maximum measurement types
Label #1 (3)	character string (*Currently not supported*)

2.3.6 Control Class

Control classes are class number 8 and include the following sub-classes:

Sub-class	Instance	Meaning
0	00-05	High-speed setpoints #1 to #6
1	00-0A	Standard setpoints #1 to #11
2	00-02	Relay Pulse Controller #1 to #3

2.3.6.1 Setpoints

The High-speed setpoints and Standard setpoints have the following components:

Qualifier	Data-type
Data (0)	CNTSTAT
Set-up (2)	STRUCT
Label #1 (3)	STRING
Label #2 (4)	STRING

The set-up structure for setpoints has the following arrangement:

Bytes	Data-type	Meaning
2	integer	flags
2	integer	trigger key
4	INT32	high limit
4	INT32	low limit
2	integer	operate time delay
2	integer	release time delay
2	integer	action key #1
2	integer	action key #2

The flags specify the type of the setpoint. Only the lower 2 bits of the flags field are used. The remaining bits should be masked off before use. The lower 2 bits have the following meaning:

Bits	Meaning
00	under positive
01	over positive
10	under negative
11	over negative

The limit fields for the setpoint structure must always be positive numbers. Thus, to specify a setpoint for a negative value, the *under negative* or *over negative* type must be used. In practice, the only time this is done is to specify an over power setpoint for kW export or KVAR export (since exported power has a negative sign), in which case the *over negative* type is used.

The trigger key can be any object in the IO Class (1), Variable Class (4) and Predicted Demand Class (6). For the standard setpoints, all of the Variable Class objects marked "STD" can be used as trigger keys. For the high speed setpoint, all of the Variable Class objects marked "HS" can be used.

The limit fields specify where the setpoint is activated and where its is released. For an over setpoint, the setpoint is activated on the high limit and released on the low limit, and vice versa for an under setpoint. The limit field values for frequency objects are x100 and the limit field values for harmonic distortion objects (including K-Factor) are x10.

The delay fields specify how long the 4720 will wait to activate or release a setpoint once the pertinent limit has been reached. For standard setpoints, the value for this field is specified in seconds. For high speed setpoints, it is specified in cycles.

The action keys specify the instance number for an object to perform an action on. The following action keys are possible:

Action Key	Setpoint Supported	Meaning
0	-	No action
0211	STD	Clear all preset min/max logs
0212	STD	Clear real time preset min/max log
0213	STD	Clear RT TD preset min/max log
0214	STD	Clear SD preset min/max log

0215	STD	Clear HD V1 preset min/max log
0216	STD	Clear HD V2 preset min/max log
0217	STD	Clear HD V3 preset min/max log
0218	STD	Clear HD VX preset min/max log
0219	STD	Clear HD V1 TD preset min/max log
021A	STD	Clear HD V2 TD preset min/max log
021B	STD	Clear HD V3 TD preset min/max log
021C	STD	Clear HD VX TD preset min/max log
021D	STD	Clear HD I1 preset min/max log
021E	STD	Clear HD I2 preset min/max log
021F	STD	Clear HD I3 preset min/max log
0220	STD	Clear HD I4 preset min/max log
0221	STD	Clear HD I1 TD preset min/max log
0222	STD	Clear HD I2 TD preset min/max log
0223	STD	Clear HD I3 TD preset min/max log
0224	STD	Clear HD I4 TD preset min/max log
0225	STD	Clear PD preset min/max log
1000-1004	STD HS	Clear digital input counter 0-3 (Status input counter 1-4), 4=ALL
1100-1102	STD HS	Operate Relay #1 to 3
1C00-1C04	STD HS	same as 1000-1004
A100-A107	STD HS (*)	Snapshot #1 to 8
A200-A215	STD	Reset prog. min/max log #1 to 16
A400-A407	STD HS	Waveform Capture channels #1 to 8
A500	STD HS	Waveform Recorder

Note: Action keys marked with STD are supported by Standard Setpoints (1-11), action keys marked with HS are supported by High Speed Setpoints (1-6). High Speed Setpoints can only trigger the High Speed Snapshot Log (Snapshot log #8).

There are a few additional idiosyncrasies associated with using setpoints:

1. For kW, kVAR, kW demand and kVAR demand, the 4720 only supports over power import (type *over positive*) and over power export (type *over negative*).

2. For kVA and kVA demand, the 4720 only supports over power (type *over positive*). No distinction is made between import and export.

3. For voltage and current unbalance, only an *over positive* type setpoint is supported. The limits can range from 0 to 100 (percent).

4. For power factor parameters, only an under power factor setpoint is supported. A setpoint type of *under positive* is used for under power factor lagging. In this case the limit values can range from 0 to 100, with the low limit used to trigger the setpoint. A setpoint type of *over positive* is used for under power factor leading. In this case, the limit values can range from 100 to 200, which represent power factors from 1.00 to 0.00 leading. The low limit is still used to trigger the setpoint, although it will be numerically higher than the high limit.

5. To trigger a setpoint for phase reversal, an *over positive* setpoint should be used, with the high limit set to 0 and the low limit set to 1.

6. For the status input setpoints, to trigger on a single input going ON, use an *over positive* setpoint. To trigger on a single input going OFF, use an *under positive* setpoint. To trigger on any status input going ON (object 1404) or any status input going OFF (object 1405), use an *over positive* setpoint. For all status setpoints, the high limit must be set to 0 and the low limit must be set to 1.

7. For the input counters, only an *over positive* setpoint is supported.

2.3.6.2 Relay Pulse Controller

The relay pulse controller has the following qualifiers:

Qualifier	Data-type
Data (0)	STCOUNT
Set-up (2)	STRUCT

The set-up structure has the following components:

Bytes	Data-type	Meaning
2	integer	mode
2	integer	value

The mode is one of:

Mode	Meaning
0	Relay is setpoint triggered
1	Relay is kWh pulsed
2	Relay is kVARh pulsed
3	Relay is kVAh pulsed
4	Relay is kWh-F pulsed
5	Relay is kWh-R pulsed
6	Relay is kVARh-F pulsed
7	Relay is kVARh-R pulsed

If the mode is setpoint triggered, then the value represents the on time of the relay in seconds. A pulse time of 0 indicates a latched condition. If the mode is for one of the hour pulsing types, then the value indicates the number of khours per pulse.

2.3.7 Log Class

The following log sub-classes are supported:

Class	Sub-class	Instance	Meaning
A	0	00	Event Log
A	1	00-07	Snapshot Logs #1 to #8
A	2	00-0F	Min/max Logs #1 to #16
A	3	-	reserved
A	4	00-07	Waveform Capture Log
A	5	00,10-17	Waveform Recorder Log

Log classes have the following qualifiers:

Qualifier	Data-type
Data (0)	STCOUNT
Set-up (2)	STRUCT
Log Record (5)	STRUCT

The Data (0) element contains two-bytes that indicate the depth of the log and two-bytes that represent the current record pointer value. The most significant two bytes are the depth and the least significant two bytes are the counter value. In most cases, the current record pointer value points to the end of the log.

Before reading log records from any log, the host must first determine the current record pointer and the size of the log. Any attempt to read outside the current boundaries of the log can cause erratic behaviour on the 4720.

When reading log records from any of the logs, it is necessary to follow the log object key with a four byte Extended Key. The first two bytes of the extended key specify the number of records to read, and the second two bytes specify the starting record. When the device responds, it will always send the extended key back, immediately preceding the records, even if the message does not request keys with the response. See example 3 for more details.

2.3.7.1 Event Log Sub-class

The event log currently has no set-up structure. The structure of an event log record is:

Bytes	Data-type	Meaning
8	TIME	time of event record
2	integer	cause key
4	INT32	cause value
2	integer	effect key
4	INT32	effect value

The *cause key* identifies the reason that the event occurred. The *effect key* identifies the type of event that occurred. For example, if an overvoltage condition on phase A activates standard setpoint #1, the cause key would be equal to 4101h. The effect key would be equal to 8100h. If the setpoint has an action key to trigger a waveform recorder, then a second event would be

generated with a cause key of 8100h and an effect key of A500h. If there is a second action key defined, then a third event will be generated.

Values for the cause key can include any of the setpoint trigger keys, in addition to the setpoint keys themselves. Several of the global class objects can also be event causes. The effect key can be a setpoint key, any of the setpoint action keys, or a global class key.

The *cause value* is the present value of the cause key's object, at the time event occurred. The data type is determined by the present value data type of the cause key. For example, it can be a numeric value to represent the trip current for an overcurrent setpoint, or a Boolean value, 0 or 1, to represent OFF or ON, in the case of a status input or a setpoint changing state.

The *effect value* is the present value that the effect key's object takes on, as a result of the event occurring. The type is also dependent upon the present value type for the effect object.

The event log pointer, returned by the Present Value Qualifier, points to 1 record past the end of the event log. The host must not try to read past the event log pointer minus one. The event log pointer always increments with each new event, although the number of records in the log is never more than 100. The log rolls over at 65565. The event log cannot be reset.

2.3.7.2 Snapshot Log Sub-class

The snapshot log set-up structure is as follows:

Bytes	Data-type	Meaning
4	INT32	trigger type
4	INT32	percent space used (0 to 100)
4	INT32	percent space remaining (0 to 100)
8	TIME	time interval
8	TIME	time offset
4	INT32	number of keys (0 to 12)
24	array of integer	list of keys for snapshots
4	INT32	operation type
8	TIME	time duration

The snapshot trigger type can have the following values:

Value	Meaning
0	unused
1	standard log, interval driven
2	standard log, triggered by standard setpoints
3	high speed log, manually triggerable.
4	high speed log, triggered by high speed setpoints

The *percent space used* parameter is a value from 0 to 100, which must be a multiple of 2 and, if non-zero, must be at least equal to 4. It defines how much of the available snapshot record space will be made available to the particular log. The *percent space remaining* parameter is a read only parameter, which is calculated by the 4720, each time that a snapshot set-up structure

is written. The total of the percent space used parameters for all snapshots must not exceed 100.

The *time interval* field specifies how often to take the snapshot. For standard snapshot logs (trigger type 1-2), the interval is expressed in seconds. For the high speed snapshot log (trigger type 3-4), the interval is expressed is cycles. The microseconds part of the field is not used. The maximum frequency for standard snapshot logs is once per second. The maximum frequency for the high speed snapshot log is once per two cycles.

The *time offset* field specifies where in the interval the snapshot should be taken. Normally, the 4720 will line the intervals up on minute, hour, or day boundaries, if possible, so to take a snapshot once per hour, at 15 minutes past the hour, the time interval field would be set to 3600 and the time offset field would be set to 900. The time offset field does not apply to the high speed snapshot log, and should be set to 0.

Up to 12 keys can be specified for each snapshot log. Standard snapshot logs support any object that supports the present data qualifier. The high speed snapshot log supports all instances of the Variable Class (4), sub-class 0 (High Speed). In addition, the high speed snapshot log also supports all instances in IO Class (1), Sub-classes 0,4,8 (digital inputs).

The *operation type* in conjunction with *trigger type* control the operation of the snapshot logs. The operation types functionality is controlled by individual bits within the *operation type* field. Bit 0 and 3 are read only and are asserted automatically by the 4720.

The following *operation types* are supported with the indicated *trigger types:*

Bit	Indication	Trigger Type
0	Log status (1 = Active, 0 = Inactive)	1-4 (Read only)
1	Log gated by setpoint	2,4
2	Log stop when full	3,4
3	Log oneshot	2,4 (Read only)
4	Log timed	3,4
5	Log stopped	3,4
6	Log compressed	1,2
5-31	Reserved	

Log status indicates the current status of the highspeed log. This field is a read only field.

Log gated by setpoint configures a log to activate when a setpoint triggers the log. The log automatically deactivates when the setpoint condition is cleared.

Log stop when full allows the highspeed log to run until all the non-volatile storage allocated to the log has been filled.

Log oneshot indicates that only one snapshot is taken each time the log is triggered. This is a read only field, automatically set when the snapshot interval is set to 0.

Log stopped provides a mechanism to halt a highspeed snapshot log. The *log stopped* bit is set automatically when the highspeed snapshot log stop conditions are met. The *log stopped* bit is cleared when the highspeed snapshot log is rearmed.

Log compressed enables real-time data compression of the snapshot log. The compression routine used employs a “lossy” compression algorithm, resulting in a maximum compression/decompression error of 0.05% (of reading). Log capacity can be increased by as much as 50% using real-time data compression. Data compression is not performed on non-compressible parameters, such as energy accumulators and status input counters.

The *time duration* parameter specifies the amount of time that the highspeed log is running (in cycles). If the time duration exceeds the storage capacity of the log, the log will wrap around. The time duration field is only used with operation type "Log Timed". For all other operation types, this field is ignored.

The snapshot log pointer, returned by the Present Value Qualifier, points to the end of the snapshot log. The host must not try to read past the snapshot log pointer. The snapshot log pointer always increments with each new snapshot, although the maximum number of records in the log is fixed by the percent space used parameter. The log rolls over at 65565. The snapshot log can only be reset by writing the set-up structure. Writing the set-up structure always clears the log.

Manually triggerable logs can be triggered using the Command qualifier (6) with the key of the log to be triggered (Currently, only the high speed snapshot log can be manually triggered). If the command value is non-zero, the log is triggered. If the command value is zero, the log is rearmed

The snapshot log record structure is as follows:

Bytes	Data-type	Meaning
8	TIME	time of snapshot
4∗N	INT32	values for snapshot (N=number of keys)

Note: To use the high speed snapshot log, the high speed feature Object must be programmed correctly. If the high speed snapshot is active, snapshot log #8 becomes a high speed snapshot log.

2.3.7.3 Min/max Log Sub-class

The min/max log set-up structure is as follows:

Bytes	Data-type	Meaning
4	INT32	number of keys (0 to 16)
32	array of integer	list of keys for min/max (first is trigger)

The min/max log record structure is as follows:

Bytes	Data-type	Meaning
8	TIME	time of min
8	TIME	time of max
8∗N	INT32	values for min/max (N=number of keys), 4 bytes min value, 4 bytes max value

2.3.7.4 Waveform Capture Sub-class

The waveform capture class has no set-up structure. There is only one waveform capture log, which holds a single waveform. However, there are eight keys that can be used to reference the log, each associated with one of the eight analog channels.

To capture one of the channels, the Command qualifier (6) is used, with one of the keys shown below. If the command value is non-zero, the analog channel is captured. If the command value is zero, the log is reset.

Analog channel	Key (one cycle)	Key (two cycles)
Phase A Voltage	A400	A410
Phase A Current	A401	A411
Phase B Voltage	A402	A412
Phase B Current	A403	A413
Phase C Voltage	A404	A414
Phase C Current	A405	A415
V aux.	A406	A416
Neutral Current	A407	A417

Before a channel can be captured, the log must be empty. To read the contents of the log, the Present Value qualifier is used, with Key A400. None of the other keys can read the contents of the log. As with the other logs, the Present Value qualifier returns the length of the log and the log pointer. The length will be zero or one. Resetting the log sets the length to zero. When a new waveform is captured, the length is set to one. The log pointer is the number to read when reading the log record.

Each log record consists of one (V1.20X or prior), or two (V1.30X or later) waveform cycles. Log records can be read as one cycle or two cycles (V1.30X or later). Keys A400-A407 return a log record containing one cycle. Keys A410-A417 return a log record containing two cycles.

The log record structure is as follows:

Bytes	Data-type	Meaning
4	INT32	number of channels (N=1)
4	INT32	samples per channel (M=128)
4	INT32	sampling frequency (HZ)
4	INT32	trigger type (see note)
8	TIME	time of trigger
8	TIME	time of record
N∗(2M+16)	STRUCT	waveform structure (N=number of channels)

The waveform structure looks like:

Bytes	Data-type	Meaning
4	INT32	input object identifier (an Analog Input)
4	INT32	multiplier
4	INT32	divisor
4	INT32	offset
2∗N	integer array	array of waveform values (N=number of samples)

To calculate actual RMS volts or current, apply the multiplier M, divisor D and offset F to each waveform array element as follows:

$$y = (x + F) * M / D$$

This will scale each waveform point to its proper value.

Note: The trigger type indicates the source of the trigger. The low order word (16 bits) of the trigger type indicates whether the trigger was manual (1) or setpoint (0) initiated. For setpoint initiated triggers, the high order word of the trigger type represents the setpoint number which caused the trigger. (1-11 = standard setpoints 1-11, 12-17 = high speed setpoints 1-6).

2.3.7.5 Waveform Recorder Sub-class

The waveform recorder class has no set-up structure.

To trigger the waveform recorder, the Command Qualifier (6) is used, with one of the keys listed in the Waveform Capture Sub-Class section. Unlike the waveform capture, where the key indicates the channel to capture, any key used with the waveform recorder will trigger all channels of the waveform recorder. No mechanism is available to halt channels independently. If the command value is non-zero, the currently running waveform recorder is stopped. If additional waveform recorders are empty, the next waveform recorder is automatically started. If the command value is zero, the log is reset. (all recorders are cleared, the first recorders is started).

The number of recorders that can be stored in the 4720 onboard non-volatile ram is controlled by the Waveform Recorder Depth object (Constant Class). If the depth is set to 1, 1 recorder (8 channels, each containing 36 cycles) can be stored. With a depth of 2, 2 recorders (8 channels, each containing 18 cycles) can be stored. With a depth of 3, 3 recorders (8 channels, each

containing 12 cycles) can be stored. To read the contents of the log, the Present Value qualifier is used, with Key A500. None of the other keys can read the contents of the log. The length will indicate the number of recorders saved in memory (0,1-3). Resetting the log sets the length to zero. When a recorder is triggered (halted), the length and log pointer is incremented by one. The log pointer is the number to read when reading the log recorder.

For example: The waveform recorder is configured as 3 recorders of 12 cycles. Three triggers occur, leaving the length set to 3, the log pointer set to 3. To read each of the saved recorders, three separate requests are used. The first request will be for log recorder #1, the second for log recorder #2, the last request will be for recorder #3.

To maintain compatibility with previous versions of the 4720 SEAbus Protocol, two methods are available to read the log. If keys A500-A507 are used, all 8 channels will returned. The last (closest to trigger point) 12 cycles of each recorder will be returned. If keys A510-A517 are used, only the channel indicated by the key will be returned. The number of cycles returned will depend on the configuration of the recorder.

The log recorder structure and waveform structure is identical to that of the Waveform Capture Sub-class, with the following exceptions: the number of channels can by either 1 or 8, and the samples per channel will depend on the configuration and whether 1 or 8 channels are requested (See discussion above).

Note: The older method of download all channels at once is limited to the last 12 cycles of each recorder due to bandwidth limitations. It is recommended that all future implementation of the 4720 protocol employ the by channel method of downloading. The older method is not guaranteed to be supported in future versions of the 4720 SEAbus Protocol.

2.3.8 Time of Use Class

Time of Use allows the user to measure the energy and peak demand consumed, based on the day and time of day. Time of use is implemented through a two year programmable calendar. Each day in the two year calendar can be programmed to 1 out of 16 profiles. Each profile consist of a maximum of 8 tariff changes. Tariff changes within each profile can be programmed in 15 minute increments. Each tariff change lists the tariff switched to at the indicate time. A maximum of 10 different tariffs are available. For each tariff (10), three energy and three peak demand registers are available. Therefore, a total of 60 objects are available. The energy and peak demand registers are programmable.

The time of use class consists of the following sub-classes:

Class	Sub-class	Instance	Meaning
B	0	00-09	Energy register #1
B	1	00-09	Energy register #2
B	2	00-09	Energy register #3
B	3	00-09	Peak demand register #1
B	4	00-09	Peak demand register #2
B	5	00-09	Peak demand register #3
B	6	00-0E	Miscellaneous
B	7	00	Calendar configuration
B	8	00-0F	Profile configuration

B	9	00-08	Reset registers
B	A	-	Reserved
B	B	00-04	Automatic reset triggers

2.3.8.1 Energy register sub-classes

Each energy sub-class consists of 10 instances of type INT32. Instance 00-09 correspond to Tariff 1-10 respectively. The only qualifier supported is the Present Value Qualifier (0).

2.3.8.2 Peak demand sub-classes

Each peak demand sub-class consists of 10 instances corresponding to tariff 1-10 respectively. The Present Value (0) and Time (1) qualifiers are supported. The Present Value qualifier will return type INT32, the Time Qualifier (1) returns the time at which the peak demand occurred.

2.3.8.3 Miscellaneous sub-class

The miscellaneous sub-class supports only the Present Value Qualifier (0). The sub-class consists of the following instances. Instances marked READ-ONLY can not be modified.

Instance	Meaning
00	Active tariff (READ-ONLY)
01	Energy register #1 configuration
02	Energy register #2 configuration
03	Energy register #3 configuration
04	Peak demand register #1 configuration
05	Peak demand register #2 configuration
06	Peak demand register #3 configuration
07-08	Reserved
09	Calendar CRC-16 checksum (READ-ONLY)
0A	Profile CRC-16 checksum (READ-ONLY)
0B	Reserved
0C	Active profile (READ-ONLY)
0D	Penalty rate
0E	TOU memory status (READ-ONLY)

Notes:

1. Energy registers can be configured as follows:

Value	Measurement
00	kWh NET
01	kWh IMPORTED
02	kWh EXPORTED
03	kWh TOTAL
04-06	Reserved
07	kVAh TOTAL
08	kVARh NET
09	kVARh IMPORTED
0A	kVARh EXPORTED
0B	kVARh TOTAL
0C-0F	Reserved

2. Peak demand registers can be configured as follows:

Value	Measurement
00	kW TOTAL SD
01	Reserved
02	kW TOTAL TD
03	Reserved
04	kVA TOTAL SD
05	Reserved
06	kVA TOTAL TD
07	Reserved
08	kVAR TOTAL SD
09	Reserved
0A	kVAR TOTAL TD
0B	Reserved
0C	AMP AVG. SD
0D	Reserved
0E	AMP AVG. TD
0F	Reserved

3. The calendar CRC-16 value represents the calendar checksum calculated during the previous TOU memory check. The TOU memory check is performed every 12 hours. The calendar CRC may be used to verify integrity of the calendar. If the calendar integrity is compromised, the 4720 will automatically generate a warning in the EVENT LOG. The profile CRC-16 value represents the checksum calculated on the compressed profiles during the last TOU memory check. This checksum is used to verify TOU profile integrity.

4. The penalty tariff is the tariff that takes effect when status input 3 becomes active. A penalty tariff of -1 disables this feature.

5. The TOU memory status instance indicates calendar and profile integrity failure conditions. The instance is of type INT32, bit mapped as follows:

Bit	Meaning
0	Calendar integrity lost
1	Profile integrity lost
2-31	Reserved

2.3.8.4 Calendar sub-class

The calendar sub-class contains the two year calendar. The calendar controls the active profile for each day. The calendar class supports only the Time of Use (TOU) Structure Qualifier (8).

The calendar is a 732 element array of four-bit integers. Each four bit element of the structure represents a single day. With 732 elements in the calendar structure, the overall calendar length is two years. The value stored for each day presents the daily profile (0-15). The profile structure is discussed in section 2.3.8.5.

The calendar is of a wrap around nature. At the end of two years, the calendar wraps around from element 732 to element 1. The wrap around nature means that a calendar must be send to the 4720 at least once every two years to avoid incorrect data.

The first element of the calendar is January 1 of an odd numbered year. The last element of the calendar corresponds to December 31 of an even numbered year. Space is left in the calendar for February 29. If the current year is not a leap year, the 4720 will automatically skip over the February 29 element.

For example, a date of February 3, 1995 will be located at element 34 (31 + 3) of the calendar. Similarly, April 12, 1996 will be located at element 469 (366 + 31 + 29 + 31 + 12).

2.3.8.5 Profile configuration sub-class

The profile configuration sub-class contains instances 00-0F representing the 16 daily profiles. The profiles can be read and modified using the TOU Structure Qualifier (8). Each profile is of type *Profile Structure*.

The *profile structure* consists of an array of *Changeover structures*, with a leading byte indicating the number of *Changeover structures* present within the *profile structure*.

Each *Changeover structure* contains two bytes of information. The first byte represents the time of the tariff changeover. The time is represented by multiples of 15 minutes since midnight. For example, a changeover time of 8:30 am will be represented by a value of 34. The second element of the structure contains the Tariff number (0-9) to change to at the changeover time.

2.3.8.6 Reset register sub-class

The Reset register sub-class provides instances to reset each energy and peak demand register. To reset a register, the Command Qualifier (6) is used, with one of the keys shown below. If the command value is zero, the register is reset for all tariffs.

Key	Action
B900	Reset Energy Register 1
B901	Reset Energy Register 2
B902	Reset Energy Register 3
B903	Reset Peak Demand Register 1
B904	Reset Peak Demand Register 2
B905	Reset Peak Demand Register 3
B906	Reset all energy registers
B907	Reset all peak demand registers
B908	Reset all energy and demand registers

Note: No mechanism is available to reset registers for individual tariffs.

2.3.8.7 Automatic triggers sub-class

The trigger sub-class provides a means of automatically triggering a reset of the TOU registers. The triggers can be defined as setpoint source keys, allowing automatic clearing of TOU register, min/max log or triggering of any other action key. The trigger sub-class supports the Present Value (0) qualifier. The following instances are supported:

Class	Sub-class	Instance	Meaning
B	B	00	New hour
B	B	01	New day
B	B	02	New week
B	B	03	New month
B	B	04	New year

Notes:

1. The above instance obtain a value of 1 when a new hour, day, week, month or year is detected. The instances only remain active for a maximum period of 2 seconds, upon which they return to their inactive (0) state. If the above triggers are used with setpoints, time delay to operate must be set to 0.

2. New hour activates at the top of the hour, new day at midnight, new week at midnight on Saturday, new month at midnight on the last day of the month, new year at midnight, January 31.

2.3.9 Diagnostic Information Class

The Diagnostic Information Class provides communication and battery diagnostic information. Communication diagnostic information can assist third party developers in implementing the 4720 SEAbus Protocol. All information is read-only. The sub-classes are as follows:

Sub-class	Meaning
0	Communication Diagnostics
1	Battery Diagnostics
2	Programmable Features
3-F	Reserved

2.3.9.1 Communication Diagnostics Sub-class

The communication diagnostics sub-class supports the following READ-ONLY instances:

Instance	Object name	Measurement
00	RX FRAMES	4720 Frames Received (device ID not checked)
01	TX FRAMES	4720 Frames Transmitted
02	NO RESPONSE	# of instances when App. layer did not respond
03	BAD CHECKSUM	# of frames with bad CRC-16 checksum
04	INCOMPLETE	# of incomplete frames
05	WATCHDOG	# of times communication watchdog expired
06	BYTE ERROR	Framing/Overrun errors at physical layer
07	OVERRUN	Application layer overrun errors
08-0F	Reserved	
10	CACHE ID	Cache ID of programmed cache message (0=Empty)
11	CACHE STATUS	Current status of the communication cache

The cache status object provides a means of querying the current status of the communication cache. The cache status object is of type INT32, bit mapped as follows:

Bit	Field	Meaning
0	CACHE READY	Cache is being updated once per second
1	CACHE FIRST PASS	Cache has just been programmed, still needs to be updated
2	CACHE DORMANT	Cache is dormant, cache is NOT being updated.
3	CACHE WIPE	Cache erase command. Asserting this bit will erase the cache message
4-6	Reserved	
7	CACHE PROGRAMMED	Cache has been programmed with a valid message
7-31	Reserved	

Notes:

1. The communication watchdog expires when no 4720 format frame has been detected on the bus for a period in excess of 5 seconds. When the watchdog expires, communications is reset to a default state. All pending transactions will be cleared.

2. All instances in the communication sub-class increment until reset. A reset of the instances can be performed by reprogramming of any communication related parameter (baud rate, RTS level).

3. All instances in the communications sub-class may be viewed on the front panel of the device. To view the information, enter programming mode on the front panel and enable the EXTENDED field (Diagnostic menu). With the EXTENDED field enabled, the user can, while in normal display mode, enter the diagnostic display mode by pressing the outer two buttons of the front panel simultaneously. Use the phase button to advances through all instances of the sub-class.

4. The communication cache message may be cleared by asserting the CACHE WIPE bit when writing to the cache status object.

2.3.9.2 Battery Diagnostics Sub-class

The battery diagnostics sub-class supports the following instances:

Instance	Object name	Meaning
00-03	Reserved	
04	RTC BATTERY	Real-time clock battery life left in % (0-100)
05	RAM BATTERY	Ram battery life left in % (0-100)
06-FF	Reserved	

2.3.9.3 Features Sub-class

The Features Sub-class provides objects identifying the features and options installed on the device. In addition, a programmable feature object is available, allowing the user to select between features installed on the meter. All instances in this sub-class are of type INT32, with bit mapped representations. The following instances are supported:

Instance	Meaning
00	User programmable features
01	Software features installed on device
02	Hardware options installed on device
04-FF	Reserved

The user programmable features object is bit mapped as follows:

Bit	Permissions	Meaning
0	READ	Automatic battery warning and battery life indications: Should battery life fall below 10%, weekly warnings will be generated in the EVENT LOG.
1	READ	Predicted Sliding Window Demand
2	READ	K-Factor calculated on all 8 input channels.
3	READ/WRITE	Time overcurrent
4	READ/WRITE	High speed snapshot log. Snapshot log #8 becomes a high speed snapshot log (min 2 cycle update rate)
5	READ	Reserved
6	READ/WRITE	Thermal demand characteristics (0=exponential, 1=log base 10)
5-31	READ	Reserved

Notes:

1. Only one high speed feature can be active. Select between Time overcurrent and High speed snapshot log. If both Time overcurrent and High speed snapshot log are enabled, the 4720 will default to Time overcurrent. The meter is shipped from the factory with Time overcurrent enabled.

2. Only fields marked with WRITE permission can be modified. Fields marked only with READ permission will be set automatically by the 4720.

The software feature object indicates the software feature installed on the 4720. This object is READ-ONLY. The object type is INT32, bit mapped as follows:

Bit	Meaning
0	Time-of-use (1=installed, 0=not installed)
1	Scaleable status input counters
2	1ms time-sync ability. All devices on the same RS-485 bus equipped with this feature will be time synchronised to 1ms (typical).
3-4	Reserved
5	Extended waveform recorder. Provides programmable depth recorder.
6	Setpoint triggerable waveform capture
7	2-cycle waveform capture. Two cycles * 128 samples/cycle
8	Daylight Savings Time (DST). Automatically adjust time for DST
9-30	Reserved
31	User configurable task supported

The hardware options object provides a means of identifying the voltage, current and power supply options installed on the 4720. This object is READ-ONLY. The object type is INT32, bit mapped as follows:

Bit	Meaning	Ordering option
0	**CURRENT INPUT:** 5.000 Amps AC full scale	Standard
1	**CURRENT INPUT**: 1.000 Amp AC full scale	-1AMP
2	**CURRENT OVERRANGE:** XAMPS (200%)	-XAMPS
3	**CURRENT OVERRANGE:** YAMPS (500%)	-YAMPS
4	**CURRENT OVERRANGE:** ZAMPS (1000%)	-ZAMPS
5-6	Reserved	
7	**VOLTAGE INPUT:** 120 VAC full scale	Standard
8	**VOLTAGE INPUT:** 277 VAC full scale	-277
9	**VOLTAGE INPUT:** 347 VAC full scale	-347
10-12	**STATUS INPUT EXCITATION**: (0=Internal, 1=External)	0 = -EES 1 = Standard
13	Reserved	
14	**POWER SUPPLY:** 90-270 VAC or VDC	Standard
15	**POWER SUPPLY:** 20-60 VDC	Not available
16-19	Reserved	
20	**OPERATING TEMP:** 0 = 0 to +50 °C, 1 = -20 to +70 °C	0 = Standard 1 = -XTEMP
21	**RELAYS:** 0 = Form C dry contact, 1 = SPST solid state	0 = Standard 1 = -SSR
22-31	Reserved	

3 APPLICATION LAYER

The application layer provides services to allow applications to communicate with each other. In this case, the master station makes a request to a slave who, in turn, sends a response back to the master.

The format of the application layer message is described in the following section. Some example transactions are presented.

3.1 Message Format

The application packet is made up of one or more messages, each of which can have only 1 qualifier associated with it. Each additional message is appended after the previous message.

The format of the message is as follows:

Field:	Function	RR Code	Reserved	Qualifier	Length	Password	Data
Bytes:	1	1	1	1	2	2*	N

*Presence depends on Function and RR Code.

NOTE: all two-byte and four-byte integer values are passed in low-byte first order.

3.1.1 Message Function

The function byte defines the format of the rest of the message. It has the following bits defined:

Bits	Meaning
7	Last message: 0=no, 1=yes
6-4	Reserved(0)
3	Values present: 0=no, 1=yes
2	Keys present: 0=no, 1=yes
1	Read/Write: 0=read, 1=write
0	Request/Response: 0=response, 1=request

3.1.2 Response/Request Code (RRCode)

The response/request byte has a different meaning depending on whether the function byte defines a request or a response message.

For a request message, the following bit fields are defined:

Bits	Meaning
7-5	Reserved(0)
4	Password included: 0=no, 1=yes
3	Respond with values: 0=no, 1=yes
2	Respond with keys: 0=no, 1=yes
1-0	Reserved(0)

For a response message, the following bit fields are defined:

Bits	Meaning
7-1	Reserved(0)
0	Acknak: 0=nak, 1=ack

3.1.3 Message Qualifier

The qualifier byte defines which value is desired or included for all the keys in the message. The qualifier is conceptually the third byte of each key within the message.

3.1.4 Message Length

The length word defines the number of bytes in the entire message, from the function byte to the last byte of the data field.

3.1.5 Message Password

The password is a two-byte field that can only be passed in a request type message (see Message Function). The password included bit of the Request Code indicates the presence of this field.

The password is used to allow write access to parameters of a higher security level. Higher security objects include all configuration objects (Constant Class), as well as the Relay Control objects.

The message password is only required when the 4720 has been configured to operate in PASSWORD PROTECT mode. PASSWORD PROTECT mode can be enabled from the front panel (communications menu). The default mode is PASSWORD PROTECT=NO.

3.1.6 Message Data

The message data field consists of a sequence of keys or key/value pairs. In general the qualifier field and the key will define the size and interpretation of the value field.

For a LogRecord qualifier, there are four additional bytes directly after the two-byte key called the Extended-Key. The first two bytes refer to the number of log records requested, and the next two bytes specify the start counter.

3.2 Example Transactions

The following examples may provide additional help in understanding this protocol. The first example shows a set-up parameter change, the second is a request for some real-time data and the third example reads some snapshot log records.

NOTE: in the following examples, 2-byte values and 4-bytes values are read in low-order byte first (Little Endian format). For example: Bytes 04-05 (hex) may contain the value 000C; byte 04 will contain 0C (low-order byte) and byte 05 will contain 00 (high-order byte).

3.2.1 Example 1 -- Set-up Change

In this example, the volt scale, amp scale and i4 scale parameters are read and modified. First a read request is built up to find out the present values (all numbers are in hex):

Bytes	Value	Meaning
00	85	Message Function - last, no values, has keys, read, request
01	0C	Request Code - no password, send values, send keys
02	00	Reserved Byte
03	00	Qualifier - Present Value
04-05	000C	Length - # bytes in the message (12)
06-07	2014	Key - Voltage Primary Scale
08-09	2016	Key - Current Primary Scale
0A-0B	2018	Key - Neutral Current Primary Scale

The response from the 4720 is then:

Bytes	Value	Meaning
00	8C	Message Function - last, has values, has keys, read, response
01	01	Response Code - ack
02	00	Reserved
03	00	Qualifier - Present Value
04-05	0018	Length - # bytes in this message (24)
06-07	2014	Key - Voltage Primary Scale
08-0B	000004B0	Value - 1200
0C-0D	2016	Key - Current Primary Scale
0E-11	00001388	Value - 5000
12-13	2018	Key - Neutral Current Primary Scale
14-17	00001388	Value - 5000

Once the present values are read, the new values are written using the following write request:

Bytes	Value	Meaning
00	8F	Message Function - last, has values, has keys, write, request
01	0C	Request Code - no password, send values, send keys
02	00	Reserved
03	00	Qualifier - Present Value
04-05	0018	Length - # bytes in this message (24)
06-07	2014	Key - Voltage Primary Scale
08-0B	00003840	Value - 14400
0C-0D	2016	Key - Current Primary Scale
0E-11	000003E8	Value - 1000
12-13	2018	Key - Neutral Current Primary Scale
14-17	000005DC	Value - 1500

The response from the 4720 is:

Bytes	Value	Meaning
00	8E	Message Function - last, has values, has keys, write, response
01	0C	Response Code - ack
02	00	Reserved
03	00	Qualifier - Present Value
04-05	0018	Length - # bytes in this message (24)
06-07	2014	Key - Voltage Primary Scale
08-0B	00003840	Value - 14400
0C-0D	2016	Key - Current Primary Scale
0E-11	000003E8	Value - 1000
12-13	2018	Key - Neutral Current Primary Scale
14-17	000005DC	Value - 1500

*Note: It is **not** necessary to read a value before writing to that value.*

3.2.2 Example 2 -- Real-time Request

This example is a multiple message packet that will read some real-time values and read a snapshot counter. The requested real-time values are Volts LN Avg., Volts LL Avg., Amps Avg., kW Total, kVAR Total, kVA Total, kW Total Demand, kVAR Total Demand, kWh Net, kVARh Net. The request looks like:

Bytes	Value	Meaning
00	05	Message Function - not last, no values, has keys, read, request
01	0C	Request Code - no password, send values, send keys
02	00	Reserved Byte
03	00	Qualifier - Present Value
04-05	001A	Length - # bytes in the message (26)
06-07	4100	Key - Volts LN Avg.
08-09	4104	Key - Volts LL Avg.
0A-0B	4108	Key - Amps Avg.

0C-0D	4110	Key - kW Total
0E-0F	4114	Key - kVAR Total
10-11	4118	Key - kVA Total
12-13	4210	Key - kW Demand Total
14-15	4214	Key - kVAR Demand Total
16-17	4C10	Key - kWh Net
18-19	4C14	Key - kVARh Net
1A	85	Message Function - last, no values, has keys, read, request
1B	0C	Request Code - no password, send values, send keys
1C	00	Reserved Byte
1D	00	Qualifier - Present Value
1E-1F	0008	Length - # bytes in the message (8)
20-21	A100	Key - Snapshot Log #1

The response to this request is:

Bytes	Value	Meaning
00	0C	Message Function - not last, has values, has keys, read, response
01	01	Response Code - ack
02	00	Reserved Byte
03	00	Qualifier - Present Value
04-05	0042	Length - # bytes in the message (66)
06-07	4100	Key - Volts LN Avg.
08-0B	000039E1	Value - 14817
0C-0D	4104	Key - Volts LL Avg.
0E-11	00006440	Value - 25664
12-13	4108	Key - Amps Avg.
14-17	0000015C	Value - 348
18-19	4110	Key - kW Total
1A-1D	00003381	Value - 13185
1E-1F	4114	Key - kVAR Total
20-23	00001F91	Value - 8081
24-25	4118	Key - kVA Total
26-29	00003C68	Value - 15464
2A-2B	4210	Key - kW Demand Total
2C-2F	00002E44	Value - 11912
30-31	4214	Key - kVAR Demand Total
32-35	00001E20	Value - 7712
36-37	4C10	Key - kWh Net
38-3B	0058D6A5	Value - 5822117
3C-3D	4C14	Key - kVARh Net
3E-41	9928059F	Value - 2622879
42	8C	Message Function - last, has values, has keys, read, response
43	01	Response Code - ack
44	00	Reserved Byte
45	00	Qualifier - Present Value
46-47	000C	Length - # bytes in the message (12)

48-49	A100	Key - Snapshot Log #1
4A-4D	00080007	Value - current record #8, log depth is 7

This is a common request type where the first message would be initiated by some application and the snapshot counter request would be tagged on by a process responsible for managing logs.

3.2.3 Example 3 -- Snapshot Log Update

This example show how a log is uploaded. The log counter retrieved in the last example shows the status of snapshot #1. Normally, this value would be compared to the value stored in the master station. If the values are different, the master station can then make a request such as:

Bytes	Value	Meaning
00	85	Message Function - last, no values, has keys, read, request
01	0C	Request Code - no password, send values, send keys
02	00	Reserved Byte
03	05	Qualifier - Log Record
04-05	000C	Length - # bytes in the message (12)
06-07	A100	Key - Snapshot Log #1
08-0B	00050002	Extended Key - read 2 records starting at record #5

The response from the 4720 would be:

Bytes	Value	Meaning
00	8C	Message Function - last, has values, has keys, read, response
01	01	Response Code - ack
02	00	Reserved Byte
03	05	Qualifier - Log Record
04-05	0038	Length - # bytes in the message (56)
06-07	A100	Key - Snapshot Log #1
08-0B	00050002	Extended Key - read 2 records starting at record #5
0C-0F	00000003	Number of elements per log record
10-13	2B69F0D4	Log #5: TimeStamp.seconds - Jan 1 1993 03:07:00
14-17	00000000	TimeStamp.microseconds - 0
18-1B	00008D90	Parameter #1 - (kW Demand Total) 36240
1C-1F	00005469	Parameter #2 - (kVAR Demand Total) 21609
20-23	0000A502	Parameter #3 - (kVA Demand Total) 42242
24-27	2B69F110	Log #6: TimeStamp.seconds - Jan 1 1993 03:08:00
28-2B	00000000	TimeStamp.microseconds - 0
2C-2F	00008FE6	Parameter #1 - (kW Demand Total) 36838
30-33	00005294	Parameter #2 - (kVAR Demand Total) 21140
34-37	0000A623	Parameter #3 - (kVA Demand Total) 42531

The parameter values are always INT32 (4 bytes). The keys that these values refer to are stored in the snapshot set-up structure, which must be known beforehand by the requesting station.

4 DATA-LINK LAYER

The data-link layer is responsible for the transmission and reception of error-free frames. A frame is a collection of bytes in a predefined sequence. The frame of this protocol is backward compatible with the existing Power Measurement Ltd. products.

4.1 Frame Format

A two-byte value is represented as low-byte first and high-byte second. For example, the number 1234 (04D2h) would be ordered as: 210 (D2h) followed by a 4 (04h).

Bytes and bits that are marked as reserved may be used in the future and must be set to 0.

The format of a frame is as follows:

Field:	Sync	Fmt	Cntl	Len	SRC	DST	Tran	Cache ID	Data	CRC
Bytes:	1	1	1	1	2	2	1	1	N	2

The frame fields are described as follows.

4.1.1 Frame Sync

The frame sync byte is the value 14h (20 decimal). This is true for both transmission to the master and transmission to the slave.

4.1.2 Frame Format (Fmt)

The frame format byte is the value CDh (205 decimal). This byte defines the type of the frame as 4720-type frame. Previous PML protocols used the value FDh (253 decimal) as the frame format.

4.1.3 Frame Control (Cntl)

The frame control byte is a made up of the following bit fields:

Bit	Meaning
7	Direction: 0=Master to Slave, 1=Slave to Master
6	Frame Type: 0=Data, 1=Ack/nak
5	if Frame Type = Data: 0=Ack/nak disable, 1=Ack/nak enable if Frame Type = Ack/nak: 0=Nak, 1=Ack
4-0	Reserved

Note: The 4720 can be programmed to set bit 5 to 1 for a data frame. In other words, the 4720 can be configured to ask for acknowledgement of data frames from the master station. In most cases, acknowledgement frames are not necessary. If an application layer message is broken up into multiple packets, it may be more efficient to have each packet acknowledged, when data error rates are high. Full Ack, retransmission on Nak are supported. To enable master station acknowledgements, enable the "Use Ack/Nak" feature, selectable from the front panel (communications menu).

4.1.4 Frame Length

The frame length byte describes the length frame. This byte is the sum of the Frame Source bytes, the Frame Destination bytes, the Transaction byte, the reserved byte and the data field plus 1. The effective length of the frame from the Sync byte up to and including the CRC bytes is the frame length plus 5. This length is compatible with the previous PML protocols.

4.1.5 Source Address (SRC)

The source address defines the address of the frame initiator. The format of this address is as follows:

Bits	Meaning
15	Reserved(0)
14-0	Device ID

The address of 0 is reserved. The device ID has the range of 1 to 32767. Master stations can be of this range. 4720s are usually in the range 1 to 9999.

4.1.6 Destination Address (DST)

The destination address defines the address of the frame responder. The format of this address is as follows:

Bits	Meaning
15	Multicast
14-0	ID

When the multicast bit is a 0 the ID field represents the Device ID as used in the source address. When the multicast bit is set, the ID field represents a Family ID. Multicast frames are broadcast to all members of a family. The 4720 will not transmit a response to any multicast frame.

The following family ID's are supported on the 4720, version V1.200 or later.

Family	Family ID
All families	7FFFh
Time-of-use equipped meter family	0001h

The address of 0 is reserved. The device ID has the range of 1 to 32767. Master stations can be of this range. 4720s are usually in the range 1 to 9999.

4.1.7 Transaction Code (Tran)

The transaction code field is one byte in length and divided into the following bit fields:

Bits	Meaning
7	Reserved(0)
6	First Frame
5-0	Counter

The first frame bit is set to 1 for the first frame in a multi-frame transaction. The counter counts down from the first frame to the last frame which is 0. A single frame transaction has the first frame bit set to 1 and the counter set to 0.

4.1.8 Cache ID

A non-zero cache ID identifies the request as a request for a cached response. A cache response will occur if the cache ID number matches the cache ID number of the message programmed previously into the 4720 communication cache. When the cache ID requested does not match the cache ID of the stored cache message, the application layer will be parsed normally.

Notes: The communication cache is automatically reset when power to the 4720 is cycled. To activate, the cache must be reprogrammed.

The communication cache is guaranteed to update once per second. Requests for a cached message more than once per second will result in the same message being returned.

The communication cache is automatically disabled if no cache request has been received for a period of five minutes. Should a cached response arrive after the cache has become dormant, the message will be parsed normally, while the cache is reawakened.

4.1.9 Frame Data

The data field is passed *as is* to and from the upper layer (application layer).

4.1.10 Frame CRC

The CRC field is a two-byte field using CRC-16 computation on all bytes starting with the frame control byte (the sync and format bytes are skipped) and ending with the last byte of the data field. The polynomial used for this calculation is $x^{16} + x^{15} + x^2 + 1$.

4.1.10.1 CRC Calculation

This section describes the procedure for obtaining the CRC-16 value. The relevant bytes in the frame are considered a serial stream of binary data (ones and zeros). The 16-bit checksum is obtained by multiplying the serial data stream by 2^{16} and then dividing by the generator polynomial ($x^{16} + x^{16} + x^2 + 1$), which is expressed in binary as 11000000000000101. The

quotient is ignored and the 16-bit remainder is then the CRC-16 value. In calculating the CRC-16, all arithmetic operations (additions and subtractions) are performed using *modulo two* or *exclusive or (XOR)* arithmetic.

The following steps can be followed for generating the CRC-16 checksum:

1) Form a new polynomial by dropping the most significant bit of the generator polynomial and reversing the bit sequence. This results in 1010000000000001 or A001 hex.

2) Load a 16-bit register with the initial value of all one's or FFFF hex.

3) XOR the first data byte with the low-order byte of the 16-bit register, storing the results in the 16-bit register.

4) Shift the 16-bit register one bit to the right. If the shifted out bit is a 1 goto step 5, otherwise goto step 6.

5) XOR the 16-bit register with the new generator polynomial and store the result in the 16-bit register.

6) Repeat step 4 until eight shifts have been performed.

7) XOR the next data byte with the low-order byte of the 16-bit register, storing the results in the 16-bit register.

8) Repeat steps 4 through 7 until all bytes of the packet have been XORed with the 16-bit register and shifted eight times.

9) The contents of the 16-bit register is the CRC-16.

The following is an example of a CRC calculation on the following bytes: 63 90 BE (hex):

Step	Byte	Action	Register	#bits	shift bit
2		Initial Value	1111 1111 1111 1111		
	1	Load first byte	0000 0000 0110 0011		
3		Exclusive OR	1111 1111 1001 1100		
4		Shift 1 bit right	0111 1111 1100 1110	1	0
4		Shift 1 bit right	0011 1111 1110 0111	2	0
4		Shift 1 bit right	0001 1111 1111 0011	3	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1011 1111 1111 0010		
4		Shift 1 bit right	0101 1111 1111 1001	4	0
4		Shift 1 bit right	0010 1111 1111 1100	5	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1000 1111 1111 1101		
4		Shift 1 bit right	0100 0111 1111 1110	6	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1110 0111 1111 1111		
4		Shift 1 bit right	0111 0011 1111 1111	7	1

		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1101 0011 1111 1110		
4		Shift 1 bit right	0110 1001 1111 1111	8	1
	2	Load next byte	0000 0000 1001 0000		
7		Exclusive OR	0110 1001 0110 1111		
4		Shift 1 bit right	0011 0100 1011 0111	1	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1001 0100 1011 0110		
4		Shift 1 bit right	0100 1010 0101 1011	2	0
4		Shift 1 bit right	0010 0101 0010 1101	3	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1000 0101 0010 1100		
4		Shift 1 bit right	0100 0010 1001 0110	4	0
4		Shift 1 bit right	0010 0001 0100 1011	5	0
4		Shift 1 bit right	0001 0000 1010 0101	6	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1011 0000 1010 0100		
4		Shift 1 bit right	0101 1000 0101 0010	7	0
4		Shift 1 bit right	0010 1100 0010 1001	8	1
	3	Load next byte	0000 0000 1011 1110		
7		Exclusive OR	0010 1100 1001 0111		
4		Shift 1 bit right	0001 0110 0100 1011	1	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1011 0110 0100 1010		
4		Shift 1 bit right	0101 1011 0010 0101	2	0
4		Shift 1 bit right	0010 1101 1001 0010	3	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1000 1101 1001 0011		
4		Shift 1 bit right	0100 0110 1100 1001	4	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1110 0110 1100 1000		
4		Shift 1 bit right	0111 0011 0110 0100	5	0
4		Shift 1 bit right	0011 1001 1011 0010	6	0
4		Shift 1 bit right	0001 1100 1101 1001	7	0
4		Shift 1 bit right	0000 1110 0110 1100	8	1
		Polynomial	1010 0000 0000 0001		
5		Exclusive OR	1010 1110 0110 1101		
9		CRC-16 Value	1010 1110 0110 1101		

The final CRC-16 value is then AE6D in hex.

4.2 Frame Timing Considerations

The following timing constraints should be adhered to for proper serial communications with 4720s.

The response time is the time between the end of the last Master station request frame and the first frame of the Slave response. The following tables shows the minimum, maximum and typical response times for cached and non-cached transactions.

Cached message	
Response Time Min	2 byte periods
Response Time Max	300 ms
Response Typical	25 ms

Non-cached message	
Response Time Min	2 byte periods
Response Time Max	900 ms
Response Typical (small)	130 ms
Response Typical (large)	400 ms

The response time depends somewhat on the complexity of the response message. A small transaction would consist of one or two response frames, and a large transactions would consist of more than six or so response frames.

Delay time is the time between the end of the last frame of the Slave response and the start of the first frame of the next Master request. Release V1.2XX requires this time to be at least 250 ms. V1.3XX requires this time to be at least 2 byte periods (2 ms @ 9600 baud).

4.4 Sample Transaction

The following example transaction shows the method in which the data-link layer frames operate. The master station, unit 42, is sending a 100 byte application layer request to the remote device, unit 100. The remote device responds with a 400 byte application layer response. The master is configured to request acknowledgement frames, and the remote device is configured to not ask for acknowledgement frames.

The C1 and C2 bytes of the frames represent the CRC-16 field. The C1 byte is the low order byte, and the C2 is the high order byte.

Master 42 (2Ah)

From Application Layer--100 byte data request to device 100 (64h)

Frame 1 transmit: Master-Slave, Data-frame, Enable AckNak, First & Last frame: 14 CD 20 6B 2A 00 64 00 40 00 [100 bytes AL data] C1 C2

Frame 2 receive.

Frame 3 receive (240 bytes).

Frame 4 receive (160 bytes).

To Application Layer--400 byte data response from device 100 (64h).

Slave 100 (64h)

Frame 1 receive.

Frame 2 transmit: Slave-Master, Acknak-frame, Ack'd, First & last frame: 14 CD E0 07 64 00 2A 00 40 00 C1 C2

To Application Layer--100 byte data request from device 42 (2Ah).

From Application Layer--400 byte data response to device 42 (2Ah).

Frame 3 transmit: Slave-Master, Data-frame, Disable AckNak, First of 2 frames: 14 CD 80 xx 64 00 2A 00 41 00 [240 bytes AL data] C1 C2

Frame 4 transmit: Slave-Master, Data-frame, Disable AckNak, First of 2 frames: 14 CD 80 xx 64 00 2A 00 41 00 [160 bytes AL data] C1 C2

5 PHYSICAL LAYER

The physical layer of the 4720 protocol is either RS-232 or RS-485. Both of these standards are asynchronous serial communications standards. RS-232 supports only a single master and a single slave. It supports full-duplex communications which is used only in half-duplex mode. RS-485 is a half-duplex bus communications standard which supports up to 32 slave devices on a single loop.

The serial data stream on the RS-232 or RS-485 line is in the following format: no parity, eight data bits and one stop bit (10 bits per byte).

5.1 Single Device (RS-232C)

The RS-232C specification supports a single device at bits rates up to 20 kbits/second with a cable up to 15 meters (50 feet). A binary 1 is indicated by a voltage less than -3 V, and a binary 0 is indicated by a voltage greater than +4 V.

The 4720 supports a maximum bit rate of 19200 bits/second. The 4720 also supports the following signal lines:

Transmit - data output from the 4720;
Receive - data input to the 4720;
Signal Ground - common for data lines.

An additional signal line provides a Request to Send (RTS) output, but the corresponding Clear to Send (CTS) input is not used. The RTS line is asserted at least 10 milliseconds before transmission begins, and deasserted after transmission is complete.

The RS-232C standard provides for full-duplex communications. The 4720 uses only half-duplex communications.

5.2 Multiple Devices (RS-485)

The RS-485 standard allows up to 32 devices on a single bus loop at speeds up to 1 Mbits/second over a total cable length of 120 meters (4000 feet). This is a half-duplex two-wire balanced transmission method.

The 4720 is limited to 19200 bits/second on an RS-485 loop.

5.3 Network Considerations

There are a few physical limitations and recommendations for operation on an RS-232C or RS-485 communication line.

It is recommended that Master station frame requests be prefaced with one or two bytes to ensure the state of the RS-485 bus before the frame begins. The 4720 prefaces all frames with two AAh bytes (the bit pattern of 10101010 allows easy baudrate synchronisation as well as stabilising the bus). An additional byte of 55h is transmitted following the frame in order to allow the RTS line to be deasserted at the proper time. Neither the prefaced bytes or the following bytes are considered part of the frame.

The allowed time between bytes in a single frame is the interbyte time. This interbyte time must be less then one byte time as shown in the following table:

Baudrate	Interbyte Time
1200	<8.3 ms
2400	<4.2 ms
4800	<2.1 ms
9600	<1.0 ms
19200	<0.5 ms