BALDOR®
MOTORS AND DRIVES

Baldor Binary Protocol

BBP 1.03
Command Set Specification

Programmers Reference Manual

Rev D 1/99 MN???

Baldor Binary Protocol

Command Set Specification

Author	John Marshall
Protocol Version	BBP 1.03
Document Revision	C
Revision Date	1/14/99 Tom Yohanan
Filename	BBP command Set.doc
Location	

Table of Contents

1 Protocol DESCRIPTION 1

1.1 Document Scope 1

1.2 Protocol Overview 1

1.2.1 Physical Layer Description 1

1.2.2 Data Link Layer Description 1

1.2.3 Application Layer Description 2

2 Application Layer specification 1

2.1 Data Types 1

2.2 Command Packet 2

2.3 Response Packet 2

2.3.1 Transaction Status Values: 2

3 Transaction Specification 1

3.1 How To Read The Transaction Specification 1

3.1.1 Transaction Number (T#) 1

3.1.2 Name 1

3.1.3 Type 1

3.1.4 Data Field 2

3.1.5 Class 2

3.1.6 Description 2

3.2 Transaction Specification Table 3

3.2.1 5 - CommandMode 12

3.2.2 23 - InternalValue 13

3.2.3 31 - TerminalStrip 16

3.2.4 36 - 37 – OptionId# 16

3.2.5 41 - WatchdogTime 17

3.2.6 42 - CommandBuffer 18

3.2.7 43 - ExecuteBuffer 18

3.2.8 44 - Execute On Trigger 18

3.2.9 ParameterDetails 19

4 Specification Revision Table 1

1 PROTOCOL DESCRIPTION

1.1 Document Scope

This document specifies the Application layer of the Baldor Binary Protocol (BBP.) The Application layer contains the command set (also known as the Transaction table) of the protocol. The command set defines the instructions, data requirements, and syntax for communicating with the control product.

1.2 Protocol Overview

The Baldor Binary Protocol (BBP) is a high speed, binary format, transaction based, master / slave protocol designed for communications between Baldor control products and external devices such as host computers, keypads, option cards, man machine interfaces and factory bus gateways.

The protocol uses three layers of the seven-layer OSI model, (commonly known as the simplified OSI model.) These layers are the:

- Physical Layer
- Data Link Layer
- Application Layer

1.2.1 Physical Layer Description

The BBP is a half-duplex protocol designed to be used across multiple physical layers including but not limited to:

- Serial RS485, RS422, RS232. There are no inherent limitations on baud rates.
- Direct parallel connections (Application Layer only).

1.2.2 Data Link Layer Description

The Data Link layer handles the framing, addressing and error detection of Application Layer 'messages' across an asynchronous serial link. The Data Link layer is not concerned with the content or form of the messages it is transferring. The Data Link layer is fully specified in a separate document entitled "BBP Half-Duplex Data Link Layer Specification."

This layer is not normally used for communication across local parallel connections such as between a control and a locally installed option card.

The Data Link layer is designed primarily for master / slave multi-drop applications. In a master / slave application, the protocol allows a host computer system (master) to control up to 31 controls (slaves) (RS485 limitation). The slaves do not originate messages, they can only respond when polled by the master. The host (master) must wait for an acknowledgment to a telegram before addressing another control on the link.

1.2.3 Application Layer Description

The Application Layer is the message portion of the telegram. The message is made up of the actual commands (or transactions), and the corresponding data. There are two types of Application Layer messages: commands and responses.

2 APPLICATION LAYER SPECIFICATION

The Application Layer is transaction-based. The host (the master) communicates with the motor control (the slave) using a set of predetermined transactions or commands. Each Application Layer message contains a single transaction 'packet'.

There are two transaction packet types: Command and Response. Command transaction packets originate at the host, and request the control to 'do something'. Response transaction packets originate from the control and reply to commands.

In a transaction-based protocol the data field length and format is not rigidly defined. It is up to the individual transactions to define their command data requirements and response data formats.

Command packets are divided into a transaction number field and optional data field(s). Response packets are divided into a transaction number (mirrors the command transaction number), a status field, and an optional data field(s). The transaction field is an USINT data type. The maximum number of transactions is 256. The last five transaction numbers, 250-255, are reserved for 'extended transactions' to allow for future expansion beyond 256. The status field is an USINT data type. The status field is used to report the execution status of the transaction. The data fields can be any combination of the data types specified in this section, with a maximum data length of 64 bytes (defined in Data Link Layer specification.) The total application layer message length is 66 bytes (including transaction number and status.)

2.1 Data Types

The elementary data types used in this specification are given in the following table.

Keyword	**Size byte**	**Description**	**Range**	
			Minimum	**Maximum**
BOOL	1	Boolean (USINT)	0 = FALSE	1 = TRUE
SINT	1	Short Integer	-128	127
INT	2	Integer	-32768	32767
DINT	4	Double Integer	-2^{31}	$2^{31}-1$
USINT	1	Unsigned Short Integer	0	255
UINT	2	Unsigned Integer	0	65535
UDINT	4	Unsigned Double Integer	0	$2^{32}-1$
STRING	1	Character String (1 byte per character)		
BYTE	1	bit string - 8 bits	Values are bN-1 bN-2 … b2 b1 b0 where N is the number of bits in the bit string. Each value is represented as 0 or 1, corresponding to boolean FALSE or TRUE, respectively.	
WORD	2	bit string - 16 bits		
DWORD	4	bit string - 32 bits		

2.2 Command Packet

(USINT) TRANSACTION	DATA 1	DATA 2	DATA 3	. . .

Data Field

Contains commands from the host (master) to the control (slave.) The transaction field is the corresponding transaction number. The data field contains the value(s) to be passed to the transaction.

2.3 Response Packet

(USINT) TRANSACTION	(USINT) STATUS	DATA 1	DATA 2	DATA 3	. . .

Data Field

Contains replies from control (slave) to host (master.) The transaction field is the transaction number of the command that is being responded to. The status field contains the error status of the last command. If a command transaction that requests data is valid, the status byte will contain a 0 for ‘OK’, and the requested data field (if any) will follow. If a command was invalid, or not processed for some reason, an error code will be contained in the status byte. If the status indicates an ‘out of bounds’ condition, the new modified data value will returned, for all other error conditions no data fields will follow.

2.3.1 Transaction Status Values:

0 =	Command executed properly (no error.)
1 =	Invalid (or unsupported) transaction number.
2 =	Invalid data field length for transaction.
3 =	Data value out of range for transaction. (Value rejected.)
4 =	Data value out of bounds for transaction. (Value modified by control.)
5 =	Control fault condition prevented execution of command.
6 =	Control status/mode condition prevented execution of command.
7 =	Block transfer value not accepted.
8 =	End of block reached.

3 TRANSACTION SPECIFICATION

This section contains a detailed list of the transactions currently supported by the protocol. The list includes the transaction number, name, type description, and a detailed specification of the required and returned data.

Note: Some transactions are not supported by all control types. Also some controls require variations in commanded data. Where these exceptions exist, they will be identified in the text.

3.1 How To Read The Transaction Specification

The transaction table provides quick access to relevant information about each transaction. When necessary a transaction will be explained in more detail in the sections that follow.

3.1.1 Transaction Number (T#)

The transaction number is the ID, in decimal, of the command. As mentioned in Section 3, the maximum number of transactions is 256 (with 250 - 255 being reserved for future expansion.)

3.1.2 Name

The ‘Name’ field refers to a ‘C’ style variable for function names associated with the transaction. Use of these names is not necessary to interface with the transaction. These names may be used, however, in present and future software drivers and libraries provided by Baldor. When used in conjunction with such software tools, the transaction name is case sensitive.

3.1.3 Type

There are three basic transaction types: Set, Get, and those which do both: Set/Get.

- ‘Set’ transactions are used to change internal values, or execute one-time (non-modal) commands. As a general rule most ‘set’ transactions pass data to the control, but do not return any. Most execution ‘set’ commands do not pass or return data.
- ‘Get’ transactions are used to retrieve internal values or control conditions. Most ‘get’ transactions return data but do not pass data.
- ‘Set/Get’ transactions do both functions. Usually these transaction always return data, but only accept or pass data when a ‘set’ or change function is occurring. When no data is passed, the ‘Set/Get’ functions as a ‘Get’ or read-only transaction.

When a transaction does not fit these general rules, both passed and returned data fields will be clearly specified.

3.1.4 Data Field

The Data Field defines the data transfer requirements of the application layer message. This field describes the data using the ‘data type’ defined in Section 2.1. Commas separate individual elements of data.

As previously discussed in Section 3, there are two types of application layer packets: Command and Response. Command packets ALWAYS contain a transaction number (USINT). Response packets ALWAYS contain a transaction number (USINT) and a status (USINT). The transaction specification is only concerned with the data field portion of these messages. The transaction number and status are assumed to be present, and are not shown in the specification.

In transactions that fit the basic Set, Get, and Set/Get definitions, only a single data field is described in the specification. In these cases it is assumed that a ‘Set’ transaction has only Command packet data. A ‘Get’ has only Response packet data. And a ‘Set/Get’ has the same data in the Command and Response packet, unless it is being used to ‘Get’ only, in which case there is no Command data.

Transactions are not required to conform to these basic rules. When such exceptions exist, both their Command and Response data fields will be described in detail. The command data field is preceded by a **C:**, the response data field is preceded by a **R:** for identification purposes only.

In some cases variable names are given in the data field specification. These names are used to describe multiple elements of a common data type. These names are not required for use, but may be included in present and future software drivers and libraries provided by Baldor. (When used with such tools, variable names are case sensitive.)

3.1.5 Class

The class field indicates the product family letter designator. V = Vector , H = harmonized etc.

3.1.6 Description

The description field gives information regarding the use of the transaction. When possible the data range, scale, units, etc. are also given. When it is not possible to fully describe the transaction in the table, or when other information such as a state diagram or event matrix must be given, further information will be included in sections following the transaction table. An asterisk is used to indicate default power up values where applicable.

3.2 Transaction Specification Table

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
0	Null	Set	Ö	Ö	Ö	NONE	All	No action. This can be used to reset the watchdog timer, or as a placeholder in conjunction with a global ExecuteBuffer transaction.
1	RunCmd	Set/ Get	Ö	Ö	Ö	USINT	All	Network run / stop command. 0 = Stop (refer to stop mode parameter) 1 = Fwd 2 = Rev 3 = Bipolar* Actual motor direction is returned in MotorDirection. In fwd or rev, only the absolute value of the command references (speed, torque) are used. In bipolar run, the signed reference values control the direction. These commands are only valid when CtrlSource = network control.
2	RunInhibit	Set/ Get	Ö	Ö	Ö	BOOL	All	Commands a stop regardless of the command source (network, local or remote.) 1 = Stop 0 = No action*
3	CtrlSource	Set/ Get	Ö	Ö	Ö	USINT	All	Source of actual Run/Stop command 0 = Keypad (local) 1 = Terminal strip (remote) 2 = Control from network
4	ControlState	Get		Ö	Ö	USINT	All	0 = Not Ready (no main power) 1 = Ready (disabled) 2 = Enabled 3 = Stopping 4 = Faulted

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
5	CommandMode	Set/ Get	Ö	Ö	Ö	USINT	All See sec.	Command Mode 0 = None (Disables)* 1 = Torque CMD selected source 2 = Torque CMD network 3 = Speed CMD selected source 4 = Speed CMD network 5 = Orient 6 = Position CMD ABS 7 = Position CMD INC 8 = Position Tracking 9 = Position CMD external 10 = Home 11 = Process Torque 12 = Process Speed 13 = Auto Tune Refer to the command mode section for complete operational description.
6	HzSpeedRef	Set/ Get	Ö			INT	I	Hz Speed Reference. Units: .1 Hz (one decimal place)
7	SpeedRef	Set/ Get		Ö	Ö	INT	All	Speed reference (Standard Resolution) Units: RPM
8	SpeedRefHigh	Set/ Get		Ö	Ö	DINT	D,E,S, V	Speed reference (High Resolution) Units: 1/256 RPM The middle 16bits mirror SpeedRef. Not supported by all product classes.
9	TorqueRef	Set/ Get		Ö	Ö	INT	D,S,V	Torque reference (Current) Scaling: ±15bit (32767) = programmed current limit.
10	ProcessRef	Set/ Get				INT	All	Process Control Reference Setpoint Scaling: +/-14bit (16384) = full scale.
11	ProcessFeedback	Set/ Get				INT	All	Process Control Feedback Scaling: +/-14bit (16384) = full scale.
12	PositionRef	Set/ Get		Ö	Ö	DINT	D*,S,V	Position Reference Scaling = quadrature counts (4xfeedback counts per rev.)

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
13	PositionSpeed	Set/ Get		Ö	Ö	INT	D*,S,V	Positioning Speed Reference Max speed used for positioning commands. Also referred to as feed rate or target velocity. Units: RPM
14	PositionFeedFwd	Set/ Get				INT	D*,S,V	Position Tracking Feedforward Optional commanded velocity used to reduce error in tracking command.
15	Position	Set/ Get		Ö	Ö	DINT	D*,S,V	Position counter Scaling = quadrature counts.
16	HomeOffset	Set/ Get				DINT	D*,S,V	Home position offset The captured home position plus the offset equals the target position.
17	CurrentActual	Get		Ö	Ö	INT	E,I,S, V	Actual motor phase current Units: 100mA RMS Note: calculated on inverter.
18	SpeedActual	Get	Ö	Ö	Ö	INT	All	Actual motor speed (absolute value.) (Approximated in some products.) Units: RPM
19	FrequencyActual	Get		Ö	Ö	INT	E,I,S, V	Actual motor frequency Units .1 Hz (one decimal place)
20	PowerActual	Get		Ö	Ö	INT	All	Actual output power Units: Watts
21	InputVoltage	Get				INT	All	Input line voltage Units: Volts RMS
22	OutputVoltage	Get		Ö	Ö	INT	All	Motor phase voltage (commanded) Units: Volts RMS
23	InternalValue	Get		Ö	Ö	**C:** USINT VariableIndex **R:** INT InternalValue	See sec 3.2.2	Request internal value Used for software debugging. Not intended for customer use.
24	MotorDirection	Get		Ö	Ö	BOOL	All	0 = Fwd 1 = Rev Actual in position feedback products, commanded in others.
25	ZeroSpeed	Get		Ö	Ö	BOOL	D,E,S, V	1 = At zero 0 = Not at zero
26	AtSpeed	Get		Ö	Ö	BOOL	All	1 = At commanded speed 0 = Not at speed

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
27	Warning	Get				BOOL	All	1 = Warning 0 = No warnings present
28	AtPosition	Get		Ö	Ö	BOOL	D*,S,V	1 = At position 0 = Not at position
29	AtSetpoint	Get				BOOL	All	1 = At setpoint 0 = Not at setpoint
30	AtSetSpeed	Get		Ö	Ö	BOOL	All	1 = At set speed 0 = Not at set speed
31	TerminalStrip	Get		Ö	Ö	WORD	All	Digital I/O status word refer to specific section for bit description.
32	SoftwareVersion	Get		Ö	Ö	STRING	All	Control software version (16 characters maximum.)
33	SoftwareRevision	Get	Ö	Ö	Ö	UINT	All	Control software revision number. For example S15-4.03 is returned as 403. Note: for custom software revisions only the core or standard revision is returned.
34	ProductSeries	Get	Ö	Ö	Ö	UINT	All	Product Series For example a Series 15H returns: 15.
35	ProductClass	Get	Ö	Ö	Ö	USINT	All	5 = V, 0=H, 1=J
36	OptionId1	Get		Ö	Ö	USINT	All	Option ID1 0 = Not installed (See table)
37	OptionId2	Get		Ö	Ö	USINT	All	Option ID2 0 = Not installed (See table)
38	RunTime	Get		Ö	Ö	UDINT	All	Total time power has been applied. Units: seconds.
39	TableSelect	Set/ Get		Ö	Ö	USINT	D,E,S, V	Parameter table select Range 0 - 3 Note: DDC only supports 2 tables. *Can only be changed when under network control.*
40	AccDecGroup	Set/ Get		Ö	Ö	USINT	All	Accel / decel group select Range 1 – 2 *Can only be changed when under network control.*
41	WatchdogTime	Set/ Get		Ö	Ö	UINT	All	Network watchdog timer Units: 10mS 0 = disable 20mS minimum 60S (6000) maximum Note: resolution varies among product classes.

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
42	CommandBuffer	Set				Buffered transaction packet	All	Command buffer Refer to specific section for full description.
43	ExecuteBuffer	Set				**C:** BOOL **R:** Buffered transaction response data	All	1 = Execute command buffer 0 = No action Refer to specific section for full description.
44	ExecuteOnTrigger	Set/ Get				USINT	All	Execute on trigger 0 = Disarmed (initial condition) 1 = Armed 2 = Executed (disarmed)
45	FaultStatus	Get		Ö	Ö	USINT	All	Request current fault condition 0 = No fault condition present 1-xx = Current fault code Refer to individual control manuals for description of codes.
46	FaultRst	Set		Ö	Ö	BOOL	All	1 = Execute fault reset 0 = No action Clears any active fault condition. Operation resumes at previous command.
47	FaultLog	Get		Ö	Ö	**C:** USINT FaultLogIndex **R:** UINT FaultCode, UDINT Time Stamp	All	Requests the FaultCode and TimeStamp for the given index. The fault log holds the last 31 fault conditions (1 being most recent and 32 the oldest.) The log will return a 0 for the code and time stamp if the specified index is empty. Time stamp is given in seconds.
48	FaultCodeText	Get		Ö	Ö	**C:** USINT Fault Code **R:** STRING Fault Text	All	Returns the text string associated with the FaultCode. The maximum number of characters is 16.
49	ForceFault	Set		Ö	Ö	BOOL	All	0->1 = Force Network Fault 0 = No action
50	SecurityStatus	Get		Ö	Ö	USINT	All	Requests present parameter security status. 0 = Network security disabled 1 = Network security unlocked 2 = Network security locked

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
51	SecurityLock	Set		Ö	Ö	INT or NONE	All	Unlocks or locks network parameter security. Passing the valid SecurityCode unlocks parameter access. Any other value (including NONE) locks parameter access. Note: valid only if network or total security is enabled.
52	CalcPresets	Set		Ö	Ö	BOOL	D,E,S,V	This transaction is used during setup to calculate initial values for tuning and performance parameters based on motor nameplate values. Note: this command is not valid for all product classes. 1 = Execute preset calculation 0 = No action
53	AutoTuneMode	Set/Get		Ö	Ö	USINT	D,E,S,V	0 = No test or cancel active test (N>0) = Run test #N. (Refer to control manual) Note: CommandMode must be set to Auto Tune. Not valid for all product classes.
54	AutoTuneData	Get				INT	D,E,S,V	Returns data related to the active running test. If test has finished, the last recorded value will be returned.
55	AutoTuneStatus	Get		Ö	Ö	USINT	D,E,S,V	Returns result of previously executed test. 0 = Test failed 1 = Test Passed 2 = Test currently running
56	BlockStructure	Get	Ö	Ö	Ö	USINT Level1Max, Level2Max, Level3Max	All	Returns the number of blocks on each programming level. Assumes a max of three programming levels.
57	BlockDetail	Get	Ö	Ö	Ö	**C:** USINT Level, Block **R:** USINT MaxParams, STRING BlockName	All	Returns the number of parameters in the block and the BlockName (16 characters max.)

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
58	BlockParamDetail	Get	Ö	Ö	Ö	**C:** USINT Level, Block, Index **R:** INT Pnum, Pvalue, Pmin, Pmax, Pdlft, Pprec, Ptype, STRING Pname, Punits	All	Returns full parameter detail information for the parameter specified at the given Level, Block and Index.
59	ParameterDetail	Get	Ö	Ö	Ö	**C:** INT Pnum **R:** INT Pnum, Pvalue, Pmin, Pmax, Pdlft, Pprec, Ptype, STRING Pname, Punits	All	Returns full parameter detail information for the given Pnum.
60	ParameterList	Get		Ö	Ö	**C:** INT Pnum, ListIndex **R:** STRING ListText	All	Returns the enumerated list string for the given parameter number and list index. Max number of characters is 16. If ListIndex exceeds the number of elements an 'end of block' status will be returned. (Note: Pmax should be used to determine the end of the list.)
61	ParameterValue	Set/ Get	Ö	Ö	Ö	**C:** INT Pnum , Pvalue **R:** INT Pvalue (Excluding Pvalue from CMD indicates Request only.)	All	Change / request value of specified user parameter. Returned value will give actual, after any bounds checking. Refer to individual control manual for detailed parameter description.

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
62	BatchSend	Get		Ö	Ö	**C:** Int GroupNumber **R:** INT $Pnum_N$, $Pvalue_N$, … N=16	All	Batch transfer that returns raw parameters (data only) from control to host. Up to 16 parameters are sent at a time. Last group will be truncated if necessary. Group numbers start from 0. If the GroupNumber exceeds the number of blocks an 'end of block' status will be returned. The control must be disabled.
63	BatchRcv	Set		Ö	Ö	INT $Pnum_N$, $Pvalue_N$, … N=16	All	Block transfer that accepts raw parameter (data only) from host to control. Up to 16 parameters may be sent at a time. Parameters may be sent in any order. The control must be disabled.
64	FactorySettings	Set		Ö	Ö	BOOL	All	1 = Reset all parameters to initial (factory) condition. 0 = No action
65	StatusStructure	Get		Ö	Ö	SINT StatusMax, DiagsMax, OtherMax	All	Returns number of screens in each status level.
66	StatusDetail	Get		Ö	Ö	**C:** SINT Level, Index **R:** UINT Type. (if Type = standard) INT Value, UINT Prec, STRING Name, Units (If Type = custom) STRING Line1, Line2	All	Type = 1 for standard (keypad overlays STOP,FWD,REV and LOCAL, REMOTE etc.) Type = 2 for Custom: drive formats and sends both lines of text. Notes: Level 1, Index 0 returns the startup screen (custom type.) Level 3, Index 0 gives the level 3 menu text.
67	StatusValue	Get		Ö	Ö	**C:** SINT Level, Index **R:** INT Value	All	Returns the current value for the specified screen. Note only valid if screen is type: Standard, else returns invalid data length error.

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
68	KeypadData	Get/ Set		Ö	Ö	**C:** UINT KeyValue **R:** Harmonized keypad control and text data	All	Changes the current key status. When a key is released a NO_KEY is sent from the keypad. KeyValue codes conform to the standard harmonized keypad protocol. Returns buffered display data. The standard harmonized keypad protocol is embedded in a BBP packet. Note: the Watchdog feature should be used with keypad control devices for safety.
69	ClearAll	Set		Ö	Ö	USINT	All	Reserved for factory use.
70	LogClear	Set		Ö	Ö	USINT	All	Reserved for factory use.
71	AnalogInput1	Get		Ö	Ö	UINT	All	Reads the raw value of the A/D converters on the control. Update rate and resolution vary per control. Unused MSBs will be padded with zero.
72	AnalogInput2	Get				UINT	All	
73	SetAnalogOut1	Set/ Get		Ö	Ö	INT	All	Commands the DACs on the control, and / or option card. Analog output parameter must be set to Serial to be valid. (Note 8 bit DACs will only use the upper byte.)
74	SetAnalogOut2	Set/ Get		Ö	Ö	INT	All	
75	SetDigitalOut	Set		Ö	Ö	BYTE	All	Commands the digital outputs on the control, and / or option card. Only lowest four bits are used. The LSB corresponds to opto out #1. The opto output parameter must be set to serial to be valid.
76	GetDebugVal	Get		Ö	Ö	**C:** Int MemLoc **R:** INT Value	All	Reserved for factory use.
77	ControlRating	Get			Ö	C:INT R:INT	All	Returns Cont Current, Current Scaling, Default Motor Volts & Power Base ID.
78	FeedbackType	Get			Ö	C:Usint R:Usint	All	Returns 0 if no feedback module is installed, 1 if resolver board is installed & 2 if encoder board is installed.
250	ExtTrans1	----				USINT ExtTrans, DATA FIELD	All	Reserved for future use.

T#	Name	Type	inverter	vector	servo	Data Field	Class	Description
251	ExtTrans2	----				USINT ExtTrans, DATA FIELD	All	Reserved for future use.
252	ExtTrans3	----				USINT ExtTrans, DATA FIELD	All	Reserved for future use.
253	ExtTrans4	----				USINT ExtTrans, DATA FIELD	All	Reserved for future use.
254	ExtTrans5	----				USINT ExtTrans, DATA FIELD	All	Reserved for future use.
255	ExtTrans6	----				USINT ExtTrans, DATA FIELD	All	Reserved for future use.

Note: items shown in ~~strikethrough~~ are proposed for future versions of the protocol. They are not implemented in this version.

3.2.1 5 - CommandMode

Command

5	USINT CommandMode

Response

5	ST	USINT ComandMode

Type: Set/Get

This transaction is used to change the control's command mode. The command mode is an 8-bit value that changes the control mode of operation. Loading the appropriate value into the command mode register activates the appropriate operating mode. Only one mode can be selected at a time. The following is a description of the possible command modes:

3.2.1.1 Command Mode Table

Value	Mode	Class	Description
0	None	All	No mode selected. Output stage of control remains off or disabled (voltage and current removed from the motor), regardless of RunCmd condition.
1	Torque CMD selected source	D,S,V	Closes the current loop with command input from the source selected in the COMMAND SELECT parameter.
2	Torque CMD network	D,S,V	Closes the current loop with command input from the TorqueRef register.

Value	Mode	Class	Description
3	Speed CMD selected source	All	Closes the velocity loop with command input from the source selected in the COMMAND SELECT parameter.
4	Speed CMD network	All	Closes the velocity loop with command input from the SpeedRef register.
5	Orient	D*,S,V	C or Index channel orient. The motor will be commanded in the Fwd direction at the predefined homing speed until the index pulse is detected. The motor will then be commanded to hold position at the predefined home offset
6	Position CMD ABS	D*,S,V	Closes the position loop with an absolute position command from the PositionRef register.
7	Position CMD INC	D*,S,V	Closes the position loop with an incremental position command from the PositionRef register
~~8~~	~~Position Tracking~~	~~D*,S,V~~	~~Closes the position loop in position vs time tracking mode with command input from the PositionRef register. Optional feedforward from PositionSpeed.~~ ***Future Implementation***
9	Position CMD external	D*,S,V	Closes the position loop with command input from external option source (such as pulse follower exb card.)
~~10~~	~~Home~~	~~D*,S,V~~	~~Executes a position mode home.~~ ***Future Implementation***
11	Process Torque	All	Closes the torque process control loop. Commands come from the appropriate command input parameters.
12	Process Velocity	All	Closes the velocity process control loop. Commands come from the appropriate command input parameters.
13	Auto Tune	D,E,S,V	Changes command mode to Auto Tune. Test conditions are controlled by AutoTuneMode.

Default Condition

The command word is defaulted to 00H on power-up. This indicates a disabled condition.

3.2.2 23 – InternalValue (Not intended for customer use)

Command

23	USINT VariableIndex

Response

23	ST	INT InternalValue

Type: Get

This transaction returns the selected internal variable value.

3.2.2.1 Variable Index Table (TBD)

#	~~Name~~	~~Scale~~	~~Precision~~	~~Description~~	~~Class~~
~~0~~	~~ANA INPUT #1~~				~~ALL~~
~~1~~	~~ANA INPUT #2~~				~~ALL~~
~~2~~	~~ANA OUTPUT #1~~				~~ALL~~
~~3~~	~~ANA OUTPUT #2~~				~~ALL~~
	~~ELEC ANGLE~~				~~V,S~~
	~~ABS COMMAND~~				~~ALL~~
	~~PWM VOLTAGE~~				~~V,S,SV,O~~
	~~DIRECT CURRENT~~				
	~~CMD DIRECT CUR~~				
	~~QUAD CURRENT~~				
	~~CMD QUAD CUR~~				
	~~FIELD WEAKEN~~				
	~~FOLLOWING ERR~~				
	~~QUAD CONTROL~~				
	~~DIRECT CONTROL~~				
	~~AC VOLTAGE~~				
	~~BUS VOLTAGE~~				
	~~VECTOR ANGLE~~				
	~~POWER~~				
	~~OVERLOAD ACCUM~~				
	~~PH1 CURRENT~~				
	~~PH2 CURRENT~~				
	~~PH3 CURRENT~~				
	~~DELTA COUNT~~				
	~~% RATED LOAD~~				
	~~USER OUTPUT~~				
	~~CONTROL TEMP~~				
	~~BASE ID CODE~~				
	~~CONT CUR~~				
	~~PEAK CUR~~				
	~~PWM FREQ~~				
	~~AMPS / VOLT SCALE~~				
	~~RATED VOLTAGE~~				
	~~OUTPUT ZONE~~				

This table is to be defined in a future version of the specification.

3.2.3 30 - TerminalStrip

Command

31

Response

31	ST	WORD TerminalStrip

Type: Get

This transaction returns a bit-wise word representing the status of the control digital inputs and outputs.

BIT	NAME	Typical Single-Axis Location	Typical Multi-Axis Location
15	Not used		
14	Not used		
13	Not used		
12	Enable	J1/J4 - 8	J1B - 1
11	Forward	J1/J4 - 9	J1B - 2
10	Reverse	J1/J4 - 10	J1B - 3
9	Input 1	J1/J4 - 11	J1B - 4
8	Input 2	J1/J4 - 12	J1B - 5
7	Input 3	J1/J4 - 13	J1B - 6
6	Input 4	J1/J4 - 14	J1B - 7
5	Input 5	J1/J4 - 15	J1B - 8
4	Input 6	J1/J4 - 16	J1B - 9
3	Output 1	J1/J4 - 19	J1B - 10
2	Output 2	J1/J4 - 20	J1B - 11
1	Output 3	J1/J4 - 21	J1B - 12
0	Output 4	J1/J4 - 22	J1B - 13

A bit value of 1 indicates closed or on, 0 = open or off.

3.2.4 36 - 37 – OptionId#

Command

36

Response

36	ST	USINT OptionId

Type: Get

This transaction returns the id number for the option installed in the specified location.

ID	Option No	Option Name	Group
1	EXB001A01	RS232 SERIAL	2
2	EXB002A01	RS422/485 SERIAL	2
3	EXB003A01	ISOLATED INPUT	1
4	EXB004A01	4 OUT RELAY / 3-15 PSI	2
5	EXB005A01	PULSE FOLLOWER	1
6	EXB006A01	DC TACH	1
7	EXB007A01	HIGH RES ANALOG	2
8	EXB010A01	ANALOG OUT / 3 OUT RELAY	2
~~9~~	~~EB00??A00~~	~~MACRO COMMUNICTIONS~~	~~2~~
10	EB0095A00	HIGH SPEED COMMUNICATIONS RS232 / 485	2
70	EXB013A01	Devicenet	2

Group 1 is a lower board, group 2 is upper.

3.2.5 41 - WatchdogTime

Command

41	UINT WatchdogTime

Response

41	ST	UINT WatchdogTime

Type: Set/Get

Scale / Units: 10mS

This transaction is used to change the value of the network watchdog timer. The value is entered in milliseconds (mS). The watchdog timer is used to detect a communications loss. When the time between network commands exceeds the value stored in this register, the control faults and disables the motor. Each time a network command is received the internal timer is reset to zero. The host must continuously send commands to keep the timer reset. If desired a NULL transaction can be used to reset the timer. Setting the timer to zero disables this function. The minimum time value (other than zero) is 20mS (2). The maximum value is 60S (6000). Resolution varies among product classes.

3.2.6 42 - CommandBuffer

Command

42	Buffered transaction number	Buffered data up to 63 bytes

Response

42	ST

Type: Set

This transaction is used to buffer a command for later execution. This is useful for synchronizing commands to multiple controls. The command buffers of each control are sequentially loaded with the desired command. An ExecuteBuffer command is then given globally to all controls.

NOTE: Request transactions should not be used in conjunction with a global execute as responses are not sent to global commands.

NOTE: A NULL transaction may be stored in the buffer to keep an individual control from responding to a global execute command.

3.2.7 43 - ExecuteBuffer

Command

43

Response

43	ST	Buffered transaction response

Type: Set

This transaction executes the data stored in the command buffer as if it was just received from the host. The response will be the execute command buffer transaction number followed by the buffered command normal response. This is useful for synchronizing commands to multiple controls (with a global address.) (Note: if a global address is used no response will be given by the buffered command.

3.2.8 44 - Execute On Trigger

Command

44	UINT TriggerStatus

Response

44	ST	UINT TriggerStatus

Type: Set/Get

This transaction arms or disarms a “Wait Until Trigger” function. When armed the control will execute the data stored in the command buffer when it receives and external terminal strip signal. This is useful for synchronizing commands with an external event.

This command can change the status between disarmed and armed. This can be useful for backing out of a trigger command before an external signal is given. Once the trigger signal is given and the buffer command is executed, the trigger status changes to ‘executed’ and the trigger is disarmed. The trigger must be rearmed before another trigger event can occur.

0 =	Trigger disarmed (initial condition).
1 =	Trigger armed, awaiting external signal.
2 =	Trigger executed. (This is a disarmed condition.)

3.2.9 ParameterDetails

Below is a description of the data that is return during a parameter detail response:

INT PMin	Parameter minimum allowed value.
INT PMax	Parameter maximum allowed value. (Number of list items in an enumerated type parameter.)
INT Pdflt	Parameter default value (factory) value.
USINT Pprec	Indicates the number of decimal places to use for the parameter value.
BYTE Ptype	Returns bit-wise parameter type. Bit 0 = Numeric parameter Bit 1 = Enumerated list parameter Bit 2 = Can be changed while enabled Bit 3 = Default from calculation Bit 4 = Not set during ‘restore to factory’ Bit 5 = Signed parameter
STRING Pname	Returns string representing parameter name. For example “Preset Speed #1”. Max number of characters is 16.
STRING Punits	Returns the parameter engineering units string. Max number of characters is 4.

4 SPECIFICATION REVISION TABLE

SPEC REVISION	DOCUMENT REVISION	PUBLISHED DATE	DESCRIPTION
BBP 1.X	XA	3/?/96	Initial proposal stage.
BBP 1.X	XB	5/14/96	Additional proposals
BBP 1.XB	XB	5/16/96	Re-structured Data-link layer. Added length fields to both layers. Removed DLE sequences. Added support for sub-master addressing (future implementation.) Added immediate Data Link acknowledgments. Changed terminology: Application Layer became presentation layer. Data link layer packets are 'telegrams', total presentation portion of telegram is the 'message'. Individual presentation units are 'transactions'.
BBP 1.XC	XC	5/22/96	Changed Data Link layer back to ANSI style (with DLE sequences), (no length fields in either layer.) Use ANSI style select and poll messages. (This means all reads are delayed.) A total of 3 Data Link telegrams will be used: polling, command, and response. Changed layer terminology back to Application Layer again (it is more standard.) Restructured application transactions to return status code along with transaction number.
BBP 1.XD	XD	6/4/96	Adding the ability to retransmit responses from the slave. Removed duplicate addresses and DLE NAKs. Using a timeout only scheme for lost or invalid messages.
BBP 1.XE	XE	6/7/96	Adjusting the time-out specs.
BBP 1.XF	XF	6/10/96	Adjusting the time-out specs / CRC etc.
BBP 1.XG	XG	7/21/96	Misc Clean-up.
BBP 1.XH	XH	8/19/96	Changed CRC to CCITT. Misc.
BBP 1.XI	XI	8/27/96	Changed address range to 1-31. Added support for bit-wise address selection of global or point-to-point. Added bit-wise address selection and support for BCC. Changed time-out delay to BOOL (off or 1.5 character times.) Added Data type section to application layer. Completely restructured the transaction specification section. Removed transaction quick list.
BBP 1.XJ	XJ	9/12/96	Misc. typographical corrections. Updated slave flowchart. Added PositionFeedFwd. Moved ClearAll. Added LogClear. Renumbered transactions.
BBP1.XK	XK	2/25/97, 7/24/97	Changed TorqueRef scaling to 15bit = current limit. Misc notes and text changes for clarity. Added HzSpeedRef for inverters. Restructured parameter access transactions – removed many. Renamed block send and receive to BatchSend and BatchRcv. Combined the 'backwards compatible' keypad transactions into a single one. Added 3 transactions for re-creating keypad status screens in a smart keypad. Reserved 250 - 255 for future 'extended' transactions.
BBP1.00	-	9/3/97	First official version.

SPEC REVISION	DOCUMENT REVISION	PUBLISHED DATE	DESCRIPTION
BBP1.01	-	11/17/1997	Split data link layer and command set into separate documents to ease maintenance. Updated Transaction list to add MME required transactions 71 - 75. Removed the following transactions from the currently supported list: PositioniFeedFwd, HomeOffset, CommandBuffer, ExecuteBuffer, ExecuteOnTrigger, AutoTuneMode, AutoTuneData, AutoTuneStatus, BatchSend
BBP 1.01	A	12/9/97	Added BatchSend back to the supported list. Added GetDebugVal to list.
BBP 1.01	B	1/4/98	Added document information and copyright notices.
BBP 1.02	C	1/14/99	Corrected page numbers, added transaction numbers 77 & 78. Changed transaction 73 & 74 to a Set/Get. Removed strikethrough and added three columns to show which transactions are supported by what drives.
BBP 1.03	D	2/15/99	Changed S15 to inverter, S18 to vector & mint to servo on column headers.