



RS485 SERIAL HARDWARE SPECIFICATION AND
COMMUNICATION PROTOCOL

for

Touch-Pad Interface Boards

EUU-7-102331xxx
EUU-7-102360xxx

Shipping Address

Dynapower Corporation
1020 Hinesburg Road
South Burlington, Vermont
USA 05403

Phone: 1-802-860-7200

Fax: 1-802-652-1371

Mailing Address

Dynapower Corporation
P.O. Box 9210
South Burlington, Vermont
USA 05407

www.dynapower.com

techsupport@dynapower.com

Table of Contents

- 1.0 Overview
- 2.0 RS485 Hardware Specification
 - 2.1 Touch-Pad Interface Implementation
 - 2.1.1 Format
 - 2.1.2 Baud Rate
 - 2.1.3 Addressing
- 3.0 Software Protocol
 - 3.1 Conventions
 - 3.2 Command Set

- Appendix A ASCII Character Set

1.0 Overview

This document describes the Serial Protocol and ASCII command language for the Touch Pad interface boards, EUU-7-102331xxx and EUU-7-102360xxx.

The command set is based on an intuitive set of ASCII based commands. This implementation provides an extremely flexible control platform compatible with most every personal computer and programmable logic controller. A simple inexpensive RS232 to RS485 converter may be used to talk up to 32 devices from a PC.

An Ethernet to serial converter will be available shortly from Dynapower Corporation.

To aid in system integration an OPC compliant driver is available for a fee to enable Dynapower power conversion equipment to be monitored and controlled from off the shelf SCADA(Supervisory Control and Data Acquisition) systems. A few examples are Allen Bradleys RSVIEW, and Intellution.

2.0 RS485 Hardware Specification

The serial hardware implementation is based upon the EIA/TIA RS485 standard. The RS485 standard permits a balanced transmission line to be shared in a party line or multidrop mode. As many as 32 driver/receiver pairs can share a multidrop network. Figure 1.0 shows a typical two-wire multidrop network. Note that the transmission line is terminated on both ends of the line but not at drop points in the middle of the line. Termination should only be used with high data rates (38.4 Kbaud) and long wiring runs (>2500 ft.). The signal ground line connection must be used in order to keep the common mode voltage that the receivers see within the specification range.

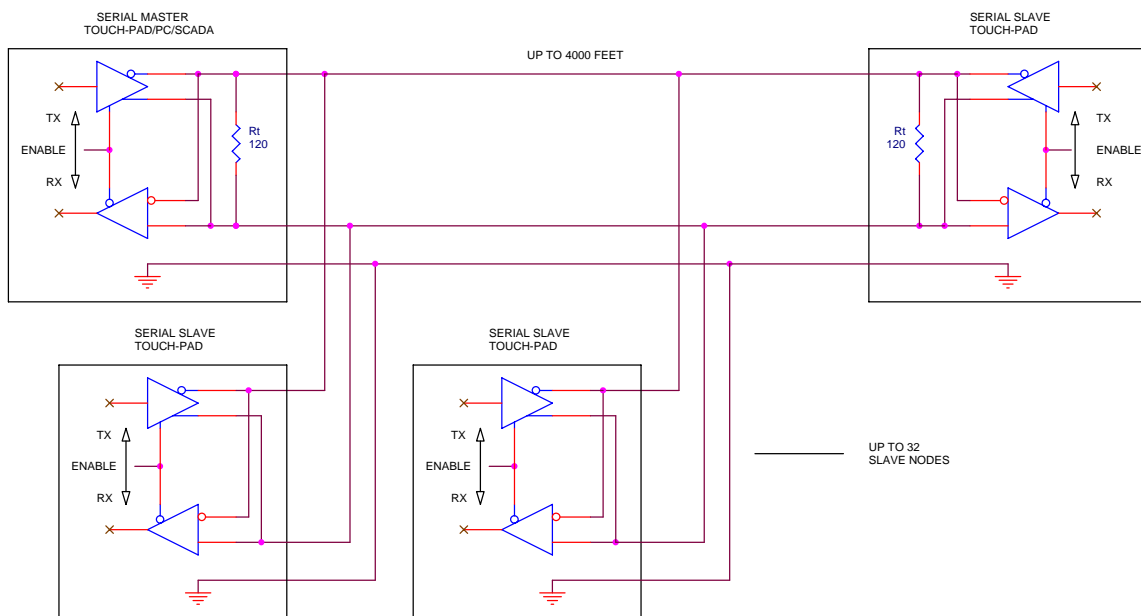


Figure 1.0

RS485 TWO WIRE MULTIDROP NETWORK

2.1 Touch-Pad Interface Implementation

The Touch Pad implementation is an extension of the standard RS485 specification. Two wire half duplex communication is supported up to 38.4Kbaud only. The Touch Pad system can support addressing up to 99 nodes. The standard 32 node network may be extended by use of repeaters. The half-duplex implementation requires that each node on the system be configured to be in a tri-stated (receive) mode unless queried by the system master control to send data. Each node automatically places itself in receive mode through software control.

The serial communication is defined by the baud rate and serial format. An address is also required to communicate to a specific device on a networked system. The information described below must be programmed at the touch pad level, defaults given, in the Serial Communications Menu. These parameters must match all devices on the network except for the node address, which must be unique for each device. Please reference the Touch-Pad operating manual for further information regarding the communications setup menu and selections.

2.1.1 Format

Serial formats supported are:

8 Bits, No Parity, One Stop Bit	8N1 (Default)
7 Bits, No Parity, One Stop Bit	7N1
7 Bits, Even Parity, One Stop Bit	7E1
7 Bits, Odd Parity, One Stop Bit	7O1

2.1.2 Baud Rate

Baud rates supported are:

2400 bps (bits per second)
4800
9600 (Default)
19,200
38,400

2.1.3 Address

Address range for the Dynapower touch pad system is from 01 (Default) to 99.

3.0 Software Protocol

The serial software protocol is based on a simple set of ASCII based instructions. The supported ASCII set is limited to the first 128 characters of the DOS character map.

The command format, conventions and command details will be explained in the following paragraphs.

3.1 Conventions

The command structure is as defined below:

[Address:]Command{!, =, ?}[data[,data]][#CSUM]{CR,LF}

[] Square brackets indicate optional items

{ } Curly braces indicate a selection of required, issued, or returned data

CR = Carriage Return

LF = Line Feed

The valid Address range is 01 to 99 followed by a colon. If the address is omitted the default address of 01 is assumed, which also indicates a global command. The default touch-pad address will be 01 and the computer master will be 00.

The Commands will be defined in detail in the next section. Commands are case insensitive, however the optional checksum feature requires the use of upper case characters in the checksum calculation.

A Command terminator {!, =, ?} is required for all issued commands. The exclamation point is used on commands that do not require any parameters. The equals sign is used to assign value(s) to a specific function. The question mark is used to indicate a command query which returns the state of the command item.

Data fields may or may not be required by the issued command. Any data field may be left empty for commands with multiple fields. If so the default or previously defined value is assumed.

An optional checksum may be used for issued commands to enhance the system error checking. This is primarily used for commands issued by automated means (executables, drivers, etc.). Exclusion of the checksum is useful in systems where commands are issued from a terminal program. If the checksum is included it must be preceded by the pound sign (#). The following set of rules are used to calculate a command checksum.

- a. Double quotes and characters within quotes including spaces are used. The ASCII hex values are added.
- b. Outside of quotes alpha characters converted to upper case. Spaces, tabs, address including colon, and the pound sign with checksum value are not included in the checksum calculation.
- c. The backspace character is also not included in the checksum, and may be used to remove previously typed characters. However the backspace cannot be used to go back past the checksum character (#) or the address terminator (:).
- d. The carriage return and or line feed characters are not included in the checksum.
- e. The checksum is transmitted as the lower two hex digits of the total sum.

Checksum Example

You type: :id="First Unit"#d6
 Touch-Pad converts to uppercase and sees: :ID="First Unit"#D6

The words "First Unit" are not converted to uppercase, because they are inside quotes.

The check summed characters are: ID="First Unit"

The ASCII hex codes for the characters are added together, and the low byte of the result is used as the checksum, printed as hex digits with a leading pound sign: #D6

I=h49 D=h44 ==h3D "=h22 F=h46 i=h69 r=h72 s=h73 t=h74 Space=h20
 U=h55 n=h6E i=h69 t=h74 "=h22

Total=4D6 The low byte is = D6

The unit address (if included) and leading colon are not included in the checksum, as well as the pound sign and checksum value.

A carriage return (CR) or linefeed (LF) is required to terminate any command sequence.

Each command sent will be acknowledged by the system immediately upon interpretation of the command, potentially before the command is executed. The return commands will be as follows:

FLT=A Fault prevents the command from being executed
 ACK = Acknowledged, valid command with valid parameters

NACK = Not Acknowledged, command not understood
Error = Command parameter invalid or out of range
CSUM = Check Sum error, Serial communication problem
RxError=Serial receive error
State = State error. Indicates a command may not execute due to the touch pads present operational status.

A command query will return the requested data for acknowledge or one of the above error messages. All returned acknowledgements or data will be preceded by a colon (:).

All numeric entry fields are limited to nine digits plus a decimal point and a sign digit if necessary.

3.2 Command Set

All commands are non-functional in Auto and Manual mode, Queries are always functional. The Kill! Command is always functional and is equivalent to an Emergency Stop.

For Clarity the command set will be presented without an address, checksum, and CR or LF specified. Additionally the voltage and current values will be presented without decimal point options. Nominally the voltage is a 4 digit number with the decimal valid in 2 positions, and the current is a 5 digit number with 3 valid decimal positions, as illustrated below.

Voltage: xx[DP]x[DP]x

Current: xx[DP]x[DP]x[DP]x

Each position indicated by [DP] is a valid decimal position as well as no decimal point at all. Only one decimal position is valid for either number at one time.

The following commands can use multiple forms.

Enable = E or EN
Disable = D or DIS

Forward = + or F or FWD
Reverse = - or R or REV

General Commands

ID="[Text String]" Board Identification. Allows assignment of serial number and board information. Text Strings are limited to 20 characters. An 'Error' is returned if more than 20 characters sent. Text strings must be enclosed in double quotation marks.

Issue: ID="236 A MOD0 SN12345"

ID? Returns serial number of board. All Returned strings unless otherwise indicated will be twenty characters in length, and padded with null characters at the end of the string.

Return: : "236 A MOD0 SN12345 "

VER? Returns software Version. This is the same as the second line of LCD text displayed when the system setup menu is accessed.

Return: : "128 Version 2.01 "

LCD? Returns two strings of 20 characters representing the first and second lines of the Liquid Crystal Display screen. This command may be additionally qualified to determine the running state of the advanced modes. These will be described as part of the advanced command definition.

Return: : "LCD Line 1 ", "LCD Line 2 "

SAVE! Save all data that has changed since previous save to EEPROM. The system will automatically save data based on an internal timing routine. The issuance of this command will immediately store any changed data.

SRQ? Status ReQuest. Returns three bytes of data representing the system status. The status is transmitted in hex code and will be as defined below.

First Byte:

	AC ON=1 AC OFF=0	DC ON=1 DC OFF=0	Warning=1	I mode=1 V mode=0	Auto=1	Manual=1	Reverse=1 Forward=0
0	0	0	0	0	0	0	0
MSB							LSB
HEX CODE (7 MAX)				HEX CODE (F MAX)			

Second Byte:

		End of AmpHour=1	End of Program=1	End of Cycle=1	Fault=1	Alarm=1	Reset Required=1
0	0	0	0	0	0	0	0
MSB							LSB
HEX CODE (3 MAX)				HEX CODE (F MAX)			

Third Byte:

			AmpHour Active=1	Program Active=1	Pulse Active=1	Cycle Active=1	Ramp Active=1
0	0	0	0	0	0	0	0
MSB							LSB
HEX CODE (1 MAX)			HEX CODE (F MAX)				

Return: :7F,3F,1F

Start and Stop Commands

AC={ON,OFF} Turn the AC power ON or OFF. Equivalent to the first Start key-press when set in Double Start or Remote Start mode. This function is only related to outputs assigned to the Step Start and/or Start outputs. If the unit is set to Single Start Mode the ON command will generate a DC ON after the appropriate system delay.

DC={ON,OFF} Turn ON or OFF the power supply DC output. Equivalent to the second Start key-press when set in Double Start or Remote Start mode. Will generate a 'State' error if the system is set to Single Start mode.

AC? AC power state query. Valid responses include ON for AC power on, OFF for AC power off, and FLT for a system fault. Includes validating M1C input if used in the system input definitions.

Return: : {ON,OFF,FLT}

DC? DC output state query. Valid responses include ON for dc output on, OFF for dc output off, and FLT for system fault. If transitioning from Forward to Reverse or vice versa the DC output is OFF during the transition time.

Return: : {ON,OFF,FLT}

FLT? System fault query. Returns a text string(s), up to 20 characters, indicating the system fault(s). If a fault is not present returned value is OK.

Return: : {OK,"Low Coolant Flow"}

KILL! Equivalent to an emergency stop. The AC and DC states are immediately placed in the OFF condition.

Alarm/Reset Command

RST={FLT,CYC,AH,PRG,ALM}

Resets a specific condition. The conditions that may require a reset include a Fault (FLT), End of Cycle (CYC), End of Amp-Hour program (AH), End of Program (PRG), or an audible alarm (ALM). An audible alarm is typically used in conjunction with one of the preceding conditions requiring a reset. The alarm may be silenced without resetting the condition causing the alarm. The internal counters associated with the above functions may also be reset by issuing this command. For example, the Cycle mode timed out but the Amp-Hour mode was active and had not reached its endpoint. The system is disabled requiring a Cycle reset to re-enable. The Amp-Hour counter may also be reset to the start point, or this count may be maintained and the Amp-Hour cycle completed upon re-enabling the system.

RST! Reset all conditions. Resets any of the states and counters listed above.

RST? Reset query. Returns all conditions requiring a reset if any. Returns OK for none, or one or more of the codes listed above.

Return: : {OK,FLT,CYC,AH,PRG,ALM}

CTRL Select Command

CTRL? Control mode query only. Returns A(Auto), M(Manual), S(Serial).
The control mode may only be set at the touch-pad terminal.

Return: : {A,M,S}

Polarity Command

POL! Reverse operating Polarity regardless of present polarity

POL={+,-} Set polarity to Forward (+) or Reverse (-).

POL? Query polarity. Returns + (Forward) or - (Reverse).

Return: : {+,-}

Voltage & Current Common Commands

VI={+,-},[xxxx],[xxxxx] The VI command serves to change the operating voltage, current and operating polarity while the system is in the DC ON state.

VI? Returns Voltage (V) and Current (I) meter/output operating values of the power conversion system being monitored. The command returns the voltage value with sign and decimal point followed by the current value with sign and decimal point.

Return: : +12.00,+10000

Set Point = Sign, Voltage, Current

The set point command serves to change the operating voltage and current for the specified polarity. These values take effect when the system is in a DC OFF state or in the polarity presently operating with DC ON.

SP={+,-},[xxxx],[xxxxx]

SP? Set Point query. Returns voltage and current set point with sign.

Return: : +12.00,+100.00

Advanced System Commands

Ramp Command

Ramp=[Polarity, Control, Mode, Base Value, Time, Time-Base]

RMP={+,-},{E,D},{V,I},{xxxx,xxxxx},{xxx,xx.x},{ms,{Sec,Min,Hrs}}

Issue: RMP=+,E,V,12.00,300,ms

E=Enable, D=Disable

I=Current, V=Voltage

ms=milliseconds, Sec=Seconds, Min=Minutes, Hrs=Hours

RMP?{+,-} Query Forward (Reverse) Ramp parameters. Return string as formatted above.

Return: :+,E,V,2.00,30.0,Sec

LCD?RMP Query Ramp Status. Returns string of current ramp status as would be shown on the LCD screen. The status includes Ramp parameter (V or I), Ramp set time, Ramp run time, and time units.

Return: :”Ramp: Set: 300ms ”,”Current Run: 123ms ”

Cycle Command

Cycle=[Control, Mode, {Time, Voltage}, {Time-Base}]

CYC={E,D},{T,V},{xxx,xx.x,xxxx},{ms,{Sec,Min,Hrs}}

Issue: CYC=E,T,12.0,Sec

Issue: CYC=E,V,12.00

T=Time, V=Voltage

CYC? Query Cycle parameters. Return string as above.

Return: :E,T,500,ms

LCD?CYC Query the Cycle status. Return string indicating Cycle set time, Cycle run time and Cycle time units as would be shown on the LCD display.

Return: :”Cycle: Set: 500ms ”,”Time Limit 333ms ”

Pulse Command

Pulse=[Polarity, Control, Mode, Base Value, On Time, Off Time, Time-Base]

PUL={+,-
,{E,D},{V,I},{xxxx,xxxxx}},{xxx,xx.x},{xxx,xx.x},{mS,{Sec,Min,Hrs}}

PUL?{+,-} Query Forward (Reverse) Pulse parameters. Return string as above.

Return: :+,E,I,1000,200,200,ms

LCD?PUL Query the Pulse status. Return string indicating Pulse on and off times as would be shown on the LCD display.

Return: :"Pulse: On : 100ms ","Current Off: 100ms "

Program Command

Program=[Control, Program #]

PRG={E,D},{1-12}

Program Edit=Step#, Mode, [Ramp], [Pulse], Voltage Set-Point, Current Set-Point, End Mode, End Time(Count), Repeat Count, Jump to Step#

PE=1,{F,R,S},{E,D}{V,I},{xxxx,xxxxx}},{xxx,xx.x},{ms,{Sec,Min,Hrs}},{E,D},{V,I},{xxxx,xxxxx}},{xxx,xx.x},{xxx,xx.x},{ms,{Sec,Min,Hrs}},{xxxx},{xxxxx},{T,V,AH},{xxx,xx.x},xxxx,xxxx},{ms,{Sec,Min,Hrs}},{A-S,A-M,AH,kAH},{xxx},{1-8}

Issue: PE=1,F,D,D,12.00,10000,T,60.0,S,1

Step# : 1-8

Mode: F=Forward, R=Reverse, S=Stop

End Mode: T=Time, AH=Amp-Hour, V=Voltage

End Time: ms=MilliSeconds, Sec=Seconds, Min=Minutes, Hrs=Hours

Repeat: 1-255 or infinity

Jump: 1-8, any previous step including current step.

PRG? Query to determine enabled program or disabled state.

Return: :E,2

PE?{1-8} Query step parameters for active program.

Issue: PE?2 Query step 2 parameters of current program.

LCD?PRG Query Program status. Returns string indicating Program number, Program run step, Repeat count, Program set parameter (AH, T, or V), and Program set/running parameter value.

Return: :”Prog 1 Step 1 # 1”,”Time: 0 / 1 ms “

Amp-Hour Command

AmpHour=Control, Mode, Amp-Hour Count, Units

AH={E,D},[{F,R,F+R,F-R}],[xxxx],[{A-s,A-m,AH,kAH}]

Issue: AH=E,F,2300,AH

F=Forward, R=Reverse

F+R=Total (Forward+Reverse), F-R=Difference(Forward-Reverse)

A-s=Amp-Seconds, A-m=Amp-Minutes

AH=Amp-Hours, kAH=kilo-Amp-Hours

AH? Query Amp-Hour parameters. Return string as above.

Return: :E,F,2300,AH

LCD?AH Amp-Hour Status query. Returns string of Amp-hour type (F, F+R, etc.), Amp-hour set time, Amp-hour run time, and Amp-hour units, as would be displayed on the LCD screen.

Return: :”AH Limit: 7 A-s”,”F+R Run: 0 A-s“

AHT?{F,R,F+R} Query Forward (Reverse or Total) Amp-Hour Totalizer count.

Return: :12345678

Appendix A ASCII Character Set

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

The hex equivalent is read from the table by row first then column. For example the hex value for a lower case g is 67.