TechNote

Using a Data Table to Send an ASCII Character String

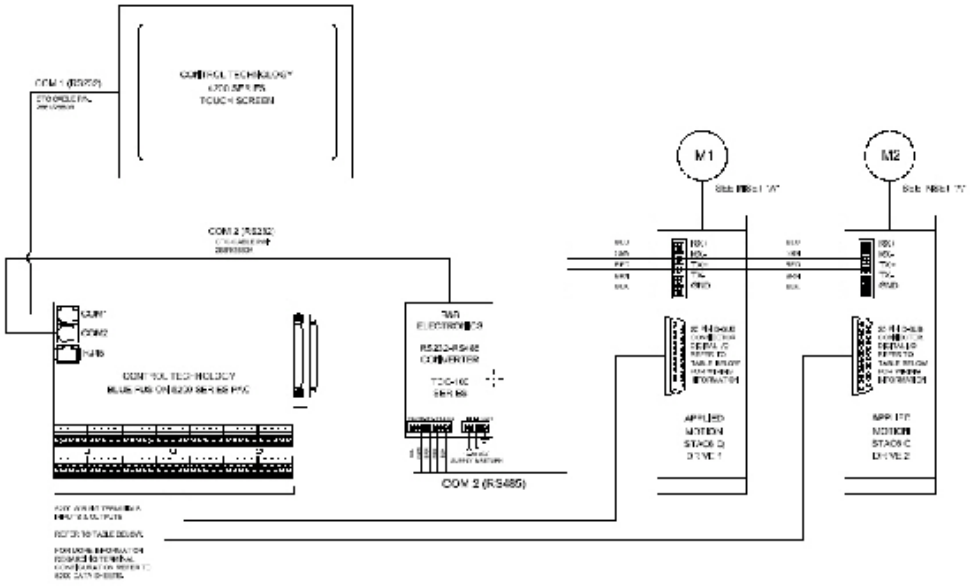
8

Control Technology Corporation, Hopkinton, MA • 800.282.5008 • www.ctc-control.com

This TechNote is an example of programming a Control Technology controller to send an ASCII data string to remote equipment via the RS232 communications port.

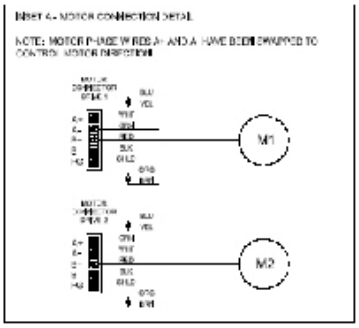
This example is a condensed version of a program, which demonstrates sending simple ASCII character commands to multiple Applied Motion servo drives via RS485 communication, allowing the operator to modify the motion profile parameters of their machine and thus changing process results on the fly. One may utilize a similar or slightly modified program such as this example to send a data string to virtually any ASCII-based serial communicating device such as a message display, or a serial printer etc.

The figure below depicts the components and wiring scheme used in this demonstration.



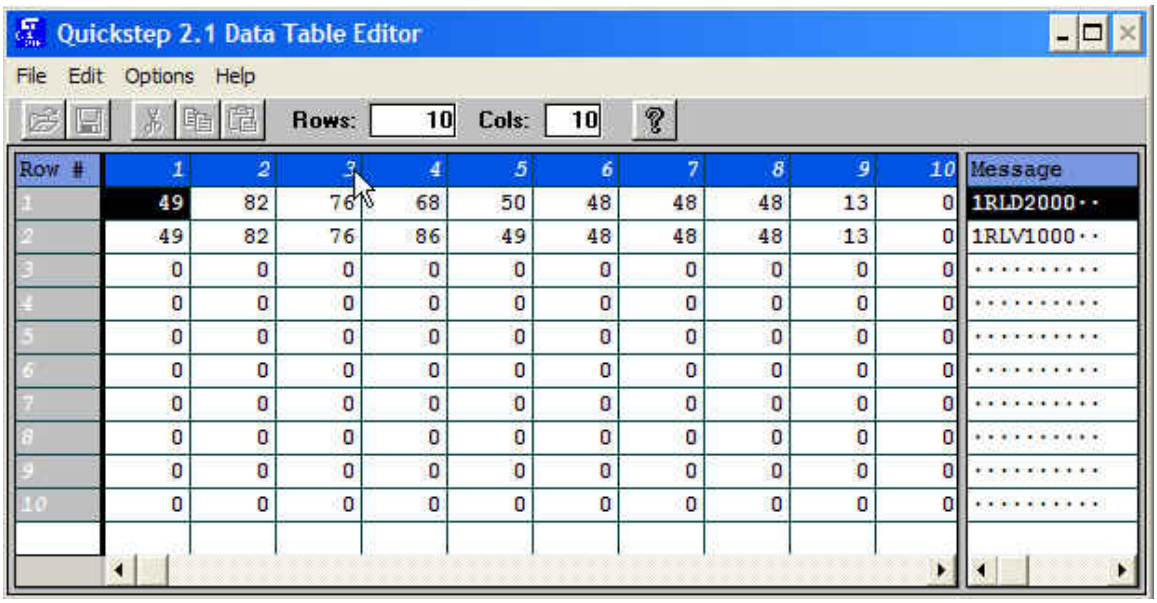
APPLIED MOTION STAGE OUPUT #	25-PIN CABLE PIN #	25-PIN CABLE COLOR	TAG NAME	CONTROL TECH. BLUE FUSION INPUT #	TERM INAL
Y1	14	BRN/WHT	INTEGRITY DET.	DIGITAL IN 1	TS1-2
Y2	15	RED/WHT	IDLE	DIGITAL IN 2	TS1-2
Y3	16	ORA/WHT	ALARM	DIGITAL IN 3	TS1-3
Y COMMON	17	GRN/WHT	N/A	INPUT COMMON	TS1-1

APPLIED MOTION STAGE INPUT #	25-PIN CABLE PIN #	25-PIN CABLE COLOR	TAG NAME	CONTROL TECH. BLUE FUSION OUTPUT #	TERM INAL
X1-	11	PNK	SPARE 1	DIGITAL OUT 1	TS1-2
X2-	9	GRY	SPARE 2	DIGITAL OUT 2	TS1-2
X3	7	BLU	REGISTRATION	DIGITAL OUT 3	TS1-3
X4	5	TAN	FEED	DIGITAL OUT 4	TS1-3
X5	3	YEL	LOG+	DIGITAL OUT 5	TS1-4
X6	4	ORG	ACK/SAFETY DET.	DIGITAL OUT 6	TS1-4
X COMMON	8	WHT	N/A	24 VDC	24 VDC SRC
X1+	12	[illegible]	N/A	5 VDC	5 VDC SRC
X2+	10	WHT	N/A	5 VDC	5 VDC SRC



For this particular example, we configure a Data Table in the Blue Fusion to store pre-determined servo drive commands. The Quick Step program simply “points” to a row in the Data Table to send to the servo drive. An “acknowledge” character and CR from the servo drive ensures proper data reception.

The first step of programming would be to configure the Data Table within the controller. For information on configuring a Data Table, refer to *Document No. MAN-1000-A: Quick Step User Guide*. For this example a simple table, 10 rows tall by 10 columns wide is constructed:



Row #	1	2	3	4	5	6	7	8	9	10	Message
1	49	82	76	68	50	48	48	48	13	0	1RLD2000··
2	49	82	76	86	49	48	48	48	13	0	1RLV1000··
3	0	0	0	0	0	0	0	0	0	0	··········
4	0	0	0	0	0	0	0	0	0	0	··········
5	0	0	0	0	0	0	0	0	0	0	··········
6	0	0	0	0	0	0	0	0	0	0	··········
7	0	0	0	0	0	0	0	0	0	0	··········
8	0	0	0	0	0	0	0	0	0	0	··········
9	0	0	0	0	0	0	0	0	0	0	··········
10	0	0	0	0	0	0	0	0	0	0	··········

Note: The original program as written is capable of sending eight drive parameters to two drives wired to an RS485 bus. For purposes of this Tech Tip, condensing the program and using only data rows 1 and 2 for sending two drive parameters to a single drive will concisely aid reader comprehension. By entering your string under the Message Column, the Quick Step programming software automatically generates the decimal equivalent of each ASCII character and assigns it to the appropriate cell. If our program points to row one of the Data Table, we will send the servo drive command, “1RLD2000”. This command sets the Distance parameter in the Applied Motion drive to 2000 pulses. Row 2 sets the Velocity parameter.

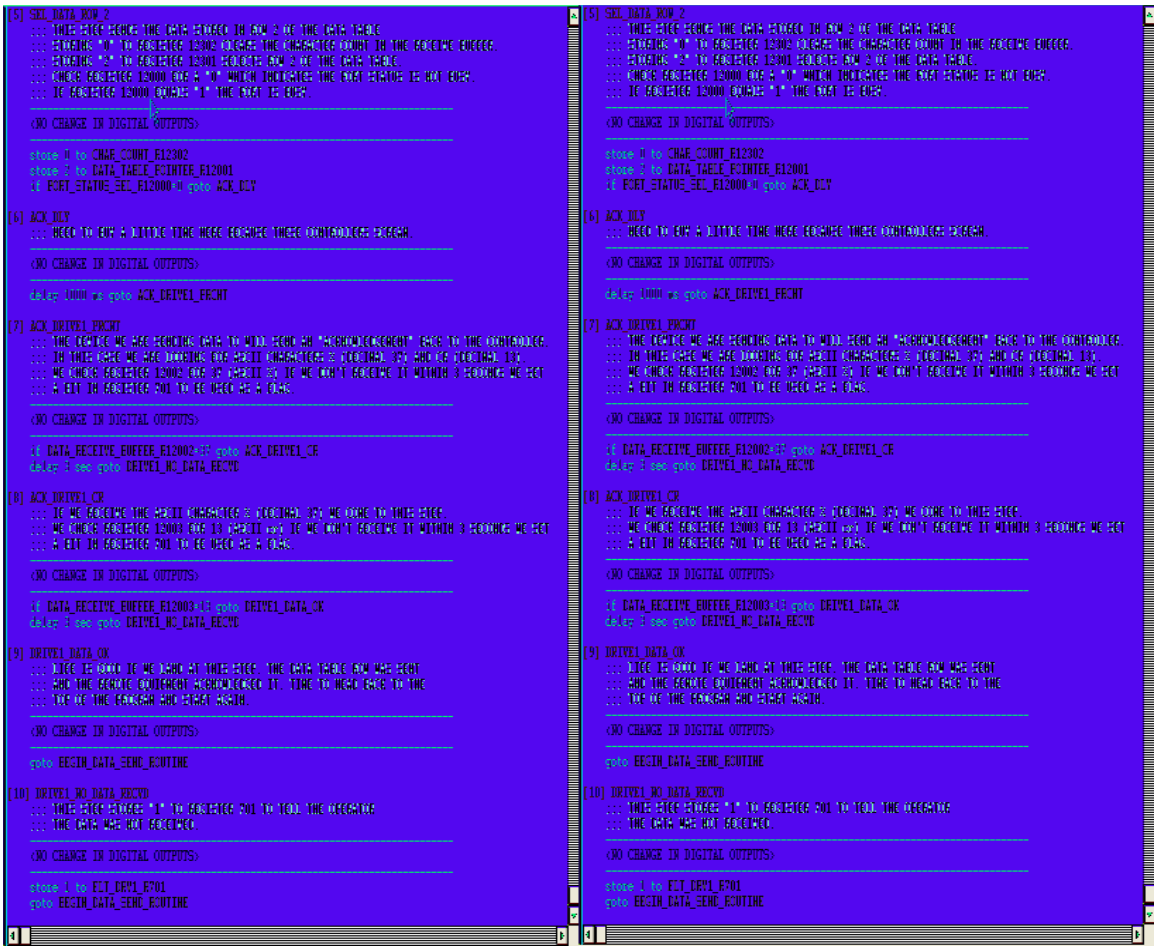
An example Quick Step program to accomplish this is as follows:

```
[1] INITIALIZE_PORT
    ;;; INITIALIZES THE COMMUNICATION PORT.
    ;;; STORING "2" TO REGISTER 12000 SELECTS COMMUNICATION PORT 2
    ;;; STORING "5" TO REGISTER 12301 SETS THE SPEED OF COMMUNICATION PORT 2 TO 9600 BAUD
    ;;; STORING "0" TO REGISTER 12303 DISABLES AUTOMATIC PARSING
    ------------------------------------------------------------------------------
    <TURN OFF ALL DIGITAL OUTPUTS>
    ------------------------------------------------------------------------------
    store 2 to PORT_STATUS_SEL_R12000
    store 5 to BAUD_RATE_R12301
    store 0 to AUTO_PARSING_R12303
    goto BEGIN_DATA_SEND_ROUTINE

[2] BEGIN_DATA_SEND_ROUTINE
    ;;; THIS STEP STORES 0's (RESETS) TO OUR SEND REGISTERS.
    ;;; i.e. USER REGISTERS THE USER DEFINED TO BE USED AS A "SWITCH" TO INITIATE
    ;;; SENDING A PARTICULAR ROW OF DATA FROM THE DATA TABLE.
    ;;; WHEN WE STORE A "1" TO ONE OF THESE REGISTERS WE WILL
    ;;; INITIATE A SEND IN STEP 3
    ------------------------------------------------------------------------------
    <NO CHANGE IN DIGITAL OUTPUTS>
    ------------------------------------------------------------------------------
    store 0 to SEND_DISTANCE_R60
    store 0 to SEND_VELOCITY_R61
    goto SEND_DATA_NOW

[3] SEND_DATA_NOW
    ------------------------------------------------------------------------------
    <NO CHANGE IN DIGITAL OUTPUTS>
    ------------------------------------------------------------------------------
    if SEND_DISTANCE_R60=1 goto SEL_DATA_ROW_1
    if SEND_VELOCITY_R61=1 goto SEL_DATA_ROW_2

[4] SEL_DATA_ROW_1
    ;;; STEPS 4 AND 5 POINT TO A ROW IN THE DATA TABLE TO SEND.
    ;;; THIS STEP SENDS THE DATA STORED IN ROW 1 OF THE DATA TABLE
    ;;; STORING "0" TO REGISTER 12302 CLEARS THE CHARACTER COUNT IN THE RECEIVE BUFFER.
    ;;; STORING "1" TO REGISTER 12301 SELECTS ROW 1 OF THE DATA TABLE.
    ;;; CHECK REGISTER 12000 FOR A "0" WHICH INDICATES THE PORT STATUS IS NOT BUSY.
    ;;; IF REGISTER 12000 EQUALS "1" THE PORT IS BUSY.
    ------------------------------------------------------------------------------
    <NO CHANGE IN DIGITAL OUTPUTS>
    ------------------------------------------------------------------------------
    store 0 to CHAR_COUNT_R12302
    store 1 to DATA_TABLE_POINTER_R12001
    if PORT_STATUS_SEL_R12000=0 goto ACK_DLY
```



For more information on this example or support for other programming solutions, please contact the Technical Support group at Control Technology Corporation.