

SERVOSTAR[®] S- and CD-Series Terminal Emulation Program

This document provides a sample program of how to create a terminal emulation program. This program also has examples of "higher-level" tasks. These tasks are executed by pressing the function keys on the computer keyboard. Additionally, these tasks are examples of how to use the subroutine calls provided in this program. It has been written using MS Professional BASIC.

For information on the SERVOSTAR[®]'s serial protocol, refer to the *SERVOSTAR[®] S- and CD-Series Serial Communications Protocol*.

Sample Program

```
DECLARE SUB GetRemote (remote%, NoComp%, Pmsg%)
DECLARE SUB DisplayBar (NoComp%, Pmsg%)
DECLARE SUB LineInput (Data$, Timeout%, NoComp%, Pmsg%)
DECLARE SUB GetMultipleData (Data$, Timeout%, NoComp%, Pmsg%)
DECLARE SUB PutGetMultipleCommand (Cmnd$, Data$, NoComp%, Pmsg%)
DECLARE SUB PutAckCommand (Cmnd$, NoComp%, Pmsg%)
DECLARE SUB GetAckmode (active%, NoComp%, Pmsg%)
DECLARE SUB CAscii (Textin$, Textout$)
DECLARE SUB GetActive (active%, NoComp%, Pmsg%)
DECLARE SUB GetReady (ready%, NoComp%, Pmsg%)
DECLARE SUB PutGetCommand (Cmnd$, Data$, NoComp%, Pmsg%)
DECLARE SUB PutCommand (Cmnd$, NoComp%, Pmsg%)
DECLARE SUB SetFlag (flag%, Value%, NoComp%, Pmsg%)
DECLARE SUB GetFlag (flag%, Value%, NoComp%, Pmsg%)
DECLARE SUB SetVariable (Variable%, Value%, NoComp%, Pmsg%)
DECLARE SUB GetVariable (Variable%, Value%, NoComp%, Pmsg%)
DECLARE SUB PutData (Data$, CharDelay%, NoComp%, Pmsg%)
DECLARE SUB GetData (Data$, Msg$, Timeout%, NoComp%, Pmsg%)
DECLARE SUB GetEcho (Cmnd$, Echo$, Msg$, Timeout%, NoComp%, Pmsg%)
DECLARE SUB GetPrompt (Msg$, NoComp%, Pmsg%)
DECLARE SUB PutStringFast (Data$)
DECLARE SUB PutStringWait (Data$, Msg$, CharDelay%, NoComp%, Pmsg%)
```

DEFINT A-Z

REM

REM SAMPLE SERIAL COMMUNICATIONS PROGRAM WITH SERIAL
REM PROTOCOL COMMUNICATION ROUTINES FOR THE SERVOSTAR[®]
REM VERSION 2.00 (SSDEMO8.BAS)

REM

```
BL$ = CHR$(7)
LF$ = CHR$(10)
Cr$ = CHR$(13)
ESC$ = CHR$(27)
qu$ = CHR$(34)
```

```
NFOR = 15
NBAK = 1
RFOR = 0
RBAK = 7
```

```

ON ERROR GOTO RECOVER

OPEN "COM1:9600,N,8,1,CD0,DS0,CS0" FOR RANDOM AS #1

COLOR NFOR, NBAK
CLS

LOCATE 24, 1
GetAckmode ackmode, NoComp, 0
GetActive active, NoComp, 0

LOCATE , , 1
COLOR RFOR, RBAK
LOCATE 25, 1
IF active AND ackmode = 0 THEN
    PRINT "F1 JOG|F2 STOP|F3 RDY|F4 STAT|F5 DUMP|F6 EN|F7 CKSM|F8 CKACK|F9 ACK0|F10 EXIT";
    ELSEIF active AND ackmode = 1 THEN
        PRINT "F1 JOG|F2 STOP |F3 RDY|F4 STAT|F5 DUMP|F6 EN|F7 CKSM|F8 CKACK|F9 ACK1|F10 EXIT";
    ELSEIF active AND ackmode = 2 THEN
        PRINT "F1 JOG|F2 STOP |F3 RDY|F4 STAT|F5 DUMP|F6 EN|F7 CKSM|F8 CKACK|F9 ACK2|F10 EXIT";
    ELSEIF active = 0 AND ackmode = 0 THEN
        PRINT "F1 JOG|F2 STOP|F3 RDY|F4 STAT|F5 DUMP|F6 DIS|F7 CKSM|F8 CKACK|F9 ACK0|F10 EXIT";
    ELSEIF active = 0 AND ackmode = 1 THEN
        PRINT "F1 JOG|F2 STOP|F3 RDY|F4 STAT|F5 DUMP|F6 DIS|F7 CKSM|F8 CKACK|F9 ACK1|F10 EXIT";
    ELSEIF active = 0 AND ackmode = 2 THEN
        PRINT "F1 JOG|F2 STOP|F3 RDY|F4 STAT|F5 DUMP|F6 DIS|F7 CKSM|F8 CKACK|F9 ACK2|F10 EXIT";
END IF

COLOR NFOR, NBAK
LOCATE 24, 1

PRINT "*** Communications Port Open ***"
IF ackmode = 2 THEN
    PRINT
    PRINT "<ack>-->";
ELSE
    PutStringFast Cr$
END IF

REM ** MAIN LOOP STARTS HERE **
MAIN:
    IF LOC(1) = 0 GOTO MLOOP      'IS COM BUFFER EMPTY ?
    A$ = INPUT$(1, #1)           ' If NO, THEN GO CHECK THE KEYBOARD
    A = ASC(A$)
    IF A > 31 THEN
        PRINT A$;
    ELSE
        SELECT CASE A
            CASE 13                '<cr> ?
                LOCATE , 1, 1
            CASE 10                '<lf> ?
                PRINT
            CASE 8                 '<bs> ?
                IF POS(1) > 1 THEN
                    LOCATE , POS(1) - 1
                    PRINT " ";
                    LOCATE , POS(1) - 1
                END IF
            CASE 6                 '<ack> ?
                PRINT "<ack>";
            CASE 7                 '<bel> ?
                PRINT BL$;
            CASE 21                '<nak> ?
                PRINT "<nak>";
        END SELECT
    END IF

MLOOP:
    O$ = INKEY$: IF O$ = "" THEN GOTO MAIN  'ARE ANY KEYS PRESSED?
    K = ASC(MID$(O$, 1, 1))                ' IF NO, THEN GO CHECK THE SERIAL PORT

```

```

IF K = 0 THEN
SELECT CASE ASC(MID$(O$, 2, 1))
CASE 45
CLOSE : END
CASE 59                                '<F1>
'EXECUTES THE JOG COMMAND AT THE USER'S DISCRETION.  DEFAULT IS J 100

GetAckmode ackmode, NoComp, 0
LINE INPUT "Please enter 'J <How fast?> <How long in ms>' (<cr>='J 100'): "; speed$
IF ackmode = 2 AND speed$ = "" THEN
speed$ = "J 100FB"                    'ADDS THE CORRECT CHECKSUM
ELSEIF ackmode = 1 AND speed$ = "" OR ackmode = 0 AND speed$ = "" THEN
speed$ = "J 100"
ELSEIF ackmode = 2 THEN
IF LEN(speed$) <> 0 THEN                'ALLOWS THE CORRECT CHECKSUM
FOR x = 1 TO LEN(speed$)              'TO BE ADDED TO THE USER ENTRY
D& = D& + ASC(MID$(speed$, x, 1))
NEXT
y$ = "0000h"
z$ = HEX$(D&)
MID$(y$, 5 - LEN(z$), LEN(z$)) = z$
YY$ = MID$(HEX$(D&), LEN(HEX$(D&)) - 1)
speed$ = speed$ + YY$
END IF
END IF
PutCommand speed$, NoComp, 1
IF ackmode <> 2 THEN
PutStringFast Cr$
ELSE
PRINT
PRINT "<ack>-->";
END IF

CASE 60                                '<F2>
'STOPS THE CURRENT EXECUTION OF A COMMAND, WHILE LEAVING THE DRIVE ENABLED

GetAckmode ackmode, NoComp, 0
IF ackmode = 2 THEN
PutCommand "STOP46", NoComp, 1
PRINT
PRINT "<ack>-->";
ELSE
PutCommand "STOP", NoComp, 1
PutStringFast Cr$
END IF

CASE 61                                '<F3>
'THIS FUNCTION KEY RETURNS THE VALUE OF THREE IMPORTANT VARIABLES INDICATING DRIVE STATUS

GetReady ready, NoComp, 1
NoComp1 = NoComp
GetActive active, NoComp, 0
IF NoComp1 = 0 AND NoComp = 0 THEN
PRINT "ACTIVE="; active; "    READY="; ready; "    REMOTE=";
GetRemote remote, NoComp, 0
PRINT remote
END IF
IF ackmode = 2 THEN
PRINT
PRINT "<ack>-->";
ELSE
PutStringFast Cr$
END IF

```

```

CASE 62                                '<F4>
'THIS FUNCTION KEY EXECUTES AND DISPLAYS THE STATUS VALUES

    GetAckmode ackmode, NoComp, 0
    IF ackmode = 2 THEN
        Cmnd$ = "STATUSE4"
    ELSE
        Cmnd$ = "STATUS"
    END IF
    PutGetCommand Cmnd$, Data$, NoComp, 1
    PRINT Data$
    IF ackmode = 2 THEN
        PRINT
        PRINT "<ack>-->";
    ELSE
        PutStringFast Cr$
    END IF

CASE 63                                '<F5>
'THIS FUNCTION KEY EXECUTES THE DUMP COMMAND SHOWING THE USER THE EFFECTS OF
'MULTIPLE LINE OUTPUT

    IF ackmode <> 2 THEN
        PutGetMultipleCommand "DUMP", Data$, NoComp, 1
        PRINT Data$
        PutStringFast Cr$
    ELSE
        PutGetMultipleCommand "DUMP36", Data$, NoComp, 1
        PRINT Data$
        PRINT
        PRINT "<ack>-->";
    END IF

CASE 64                                '<F6>
'THIS FUNCTION KEY TOGGLES THE DRIVE BETWEEN ENABLED AND DISABLED

    GetReady ready, NoComp, 0
    GetRemote remote, NoComp, 0
    IF NoComp = 0 THEN
        IF remote = 0 THEN 'REMOTE=0 SO SWITCH WILL NOT ALLOW DRIVE TO TOGGLE
            PRINT "The Remote Switch is set to Disable and will not allow the Drive to toggle."
            PRINT "Please switch it to Enable"
        ELSEIF ready = 1 AND remote = 1 THEN
            GetAckmode ackmode, NoComp, 0
            IF ackmode = 2 THEN
                PutCommand "DISE0", NoComp, 1    'Command w/ CheckSum
            ELSE PutCommand "DIS", NoComp, 1
            END IF
            PRINT "THE DRIVE IS NOW DISABLED"
            DisplayBar NoComp, 0
        ELSEIF ready = 0 AND remote = 1 THEN
            GetAckmode ackmode, NoComp, 0
            IF ackmode = 2 THEN
                PutCommand "EN93", NoComp, 0    'Command w/ CheckSum
            ELSE PutCommand "EN", NoComp, 0
            END IF
            PRINT "THE DRIVE IS NOW ENABLED"
            DisplayBar NoComp, 0
        END IF
    END IF
    IF ackmode = 2 THEN
        PRINT
        PRINT "<ack>-->";
    ELSE
        PutStringFast Cr$
    END IF

```

```

CASE 65                                '<F7>
'CHECKSUM: USED MAINLY WITH ACKMODE=2. THIS FUNCTION ALLOWS THE USER TO ENTER A STRING
'OF INPUT THAT THE FOLLOWING CODE AUTOMATICALLY EVALUATES AND COMPUTES THE CHECKSUM

D& = 0
PRINT
LINE INPUT "Enter the string to Checksum: "; x$
IF LEN(x$) <> 0 THEN
    FOR x = 1 TO LEN(x$)
        D& = D& + ASC(MID$(x$, x, 1))
    NEXT
    PRINT
    PRINT "The String is:          ";
    COLOR RFOR, RBAK
    PRINT x$;
    COLOR NFOR, NBAK
    PRINT
    y$ = "0000h"
    z$ = HEX$(D&)
    MID$(y$, 5 - LEN(z$), LEN(z$)) = z$
    PRINT "The Total Checksum is: "; y$
    PRINT "The " + PROD$ + " Checksum is: ";
    YY$ = MID$(HEX$(D&), LEN(HEX$(D&)) - 1)
    COLOR RFOR, RBAK
    PRINT YY$;
    COLOR NFOR, NBAK
    PRINT
    PRINT "Enter this string:      ";
    COLOR RFOR, RBAK
    PRINT x$ + YY$;
    COLOR NFOR, NBAK
    PRINT
    PRINT
    PRINT "Press <Enter> to send the command or press <Esc> to abort."
    PRINT
    PRINT "-->";
    REC = 1
    REC$ = x$ + YY$
    PutStringFast REC$
ELSE
    PutStringFast Cr$
END IF

CASE 66                                '<F8>
'THIS FUNCTION KEY DISPLAYS THE CURRENT ACKMODE STATE OF THE DRIVE

GetAckmode ackmode, NoComp, 0
PRINT "ACKMODE = "; ackmode
IF ackmode = 2 THEN
    PRINT
    PRINT "<ack>-->";
ELSE
    PutStringFast Cr$
END IF

```

```

CASE 67                                '<F9>
'THIS FUNCTION KEY TOGGLES BETWEEN THE ACKMODE STATES

GetAckmode ackmode, NoComp, 0
IF ackmode = 0 THEN
    PutCommand "Ackmode = 1", NoComp, 1
    DisplayBar NoComp, 0
    PutStringFast Cr$
ELSEIF ackmode = 1 THEN
    PutCommand "Ackmode = 2", NoComp, 1
    DisplayBar NoComp, 0
    PRINT
    PRINT "<ack>-->";
ELSE
    PutAckCommand "ackmode = 081", NoComp, 0
    DisplayBar NoComp, 0
    PutStringFast Cr$
END IF

CASE 68                                '<F10>
'THIS FUNCTION KEY EXITS BOTH THE COMMUNICATIONS PORT AND THE PROGRAM

CLOSE : END
END SELECT
IF K > 31 OR K = 8 OR K = 13 OR K = 22 OR K = 24 THEN
    PRINT #1, O$;
ELSEIF K = 27 THEN
    PRINT #1, O$;
    A$ = INPUT$(LOC(1), 1)
    A$ = ""
    D! = TIMER + .5
    WHILE D! > TIMER: WEND
    PRINT #1, Cr$;
END IF
GOTO MAIN
RECOVER:
RESUME

```

```

*****
'* This routine turns a Text string with unprintable or unviewable
'* characters into an output text string that is printable or viewable.
'*
'* CAscii (TEXTIN$, TEXTOUT$)
'* TEXTIN$ = Input text string to parse.
'* TEXTOUT$ = Output text string to parse.
'* - an empty sting returns <Empty>.
'* - a string that does not have any unprintable characters
'* returns the origional string unchanged.
*****
SUB CAscii (Textin$, Textout$)
  IF LEN(Textin$) = 0 THEN
    Textout$ = "<Empty>"
  ELSE
    Textout$ = ""
    FOR x = 1 TO LEN(Textin$)
      y$ = (MID$(Textin$, x, 1))
      y = ASC(y$)
      IF y < 32 THEN
        IF y = 7 THEN
          Textout$ = Textout$ + "<Bell>"
        ELSEIF y = 8 THEN
          Textout$ = Textout$ + "<BS>"
        ELSEIF y = 9 THEN
          Textout$ = Textout$ + "<HT>"
        ELSEIF y = 10 THEN
          Textout$ = Textout$ + "<LF>"
        ELSEIF y = 11 THEN
          Textout$ = Textout$ + "<VT>"
        ELSEIF y = 12 THEN
          Textout$ = Textout$ + "<FF>"
        ELSEIF y = 13 THEN
          Textout$ = Textout$ + "<CR>"
        ELSEIF y = 27 THEN
          Textout$ = Textout$ + "<Esc>"
        ELSE
          Textout$ = Textout$ + "<" + MID$(STR$(y), 2) + ">"
        END IF
      ELSE
        Textout$ = Textout$ + y$
      END IF
    NEXT x
  END IF
END SUB

```

```

*****
'* This routine displays the correct function bar on the terminal screen.
'*
'* DisplayBar (NoComp%, Pmsg%)
'*     NoComp%   = Not-Complete flag:
'*       - If 0 then Command was received, echoed, and prompted.
'*       - If 1 then The routine timed out.
'*       - If 2 then Not enough echo characters were received before timeout.
'*       - If 3 then Echo & Command mismatch.
'*       - If 4 then An Error Message was received.
'*     Pmsg%     = Print Message if equal to 1.
*****
SUB DisplayBar (NoComp%, Pmsg%)

NFOR = 15
NBAK = 1
RFOR = 0
RBAK = 7

GetActive act, NoComp, 1
GetAckmode ackmode, NoComp, 0

COLOR RFOR, RBAK
y = CSRLIN
x = POS(x)
LOCATE 25, 1

IF act = 1 AND ackmode = 0 THEN
    PRINT " F1 JOG|F2 STOP |F3 RDY|F4 STAT|F5 DUMP| F6 EN|F7 CKSM|F8 CKACK|F9 ACK0|F10 EXIT";
ELSEIF act = 1 AND ackmode = 1 THEN
    PRINT " F1 JOG|F2 STOP |F3 RDY|F4 STAT|F5 DUMP| F6 EN|F7 CKSM|F8 CKACK|F9 ACK1|F10 EXIT";
ELSEIF act = 1 AND ackmode = 2 THEN
    PRINT " F1 JOG|F2 STOP |F3 RDY|F4 STAT|F5 DUMP| F6 EN|F7 CKSM|F8 CKACK|F9 ACK2|F10 EXIT";
ELSEIF act = 0 AND ackmode = 0 THEN
    PRINT " F1 JOG|F2 STOP|F3 RDY|F4 STAT|F5 DUMP| F6 DIS|F7 CKSM|F8 CKACK|F9 ACK0|F10 EXIT";
ELSEIF act = 0 AND ackmode = 1 THEN
    PRINT " F1 JOG|F2 STOP|F3 RDY|F4 STAT|F5 DUMP| F6 DIS|F7 CKSM|F8 CKACK|F9 ACK1|F10 EXIT";
ELSEIF act = 0 AND ackmode = 2 THEN
    PRINT " F1 JOG|F2 STOP|F3 RDY|F4 STAT|F5 DUMP| F6 DIS|F7 CKSM|F8 CKACK|F9 ACK2|F10 EXIT";
END IF

COLOR NFOR, NBAK
LOCATE y, x

END SUB

```

```

*****
* This routine gets the SERVOSTAR drive ACKMODE status.
* The ACKMODE status indicates if the drive is in Ackmodes 0,1, or 2.
* It waits for the echo and for the system prompt.
* GetAckmode (Active%, NoComp%, Pmsg%)
* Active% = SERVOSTAR® drive ACKMODE status.
* NoComp% = Not-Complete flag:
* - If 0 then Command was received, echoed, and prompted.
* - If 1 then The routine timed out.
* - If 2 then Not enough echo characters were received before timeout.
* - If 3 then Echo & Command mismatch.
* - If 4 then An Error Message was received.
* Pmsg% = Print Message if equal to 1.
*****
SUB GetAckmode (active%, NoComp%, Pmsg%)
  BL$ = CHR$(7)
  LF$ = CHR$(10)
  Cr$ = CHR$(13)
  Cmdnd$ = "ACKMODE" + CHR$(13)
  Msg$ = "GetAckMode-Cmnd=ACKMODE"
  PutStringWait Cmdnd$, Msg$, 2, NoComp, Pmsg%
  IF NoComp = 0 THEN
    GetEcho Cmdnd$, Msg$, Echo$, 4, NoComp, Pmsg% 'ENSURES CORRECT ECHO WAS
    IF NoComp = 0 THEN 'RECEIVED
    'RECEIVES ACKMODE STATUS
    Timeout = 4
    Data$ = ""
    TOut! = TIMER + Timeout
    WHILE TOut! > TIMER
      IF LOC(1) THEN
        A$ = INPUT$(1, #1)
        IF A$ >= " " OR A$ = BL$ THEN Data$ = Data$ + A$ 'ADDS CHARACTERS TO STRING
        IF A$ = LF$ THEN 'CHECK FOR -->
          x = INSTR(1, Data$, BL$) 'CHARACTER POS OF BELL
          'TEST FOR BELL CHARACTER, INDICATING AN ERROR MESSAGE
          IF x AND INSTR(1, Data$, "ERR 22") THEN
            IF Pmsg THEN
              PRINT Msg$
              PRINT "*** GetData - Error #4 - Error Message Received ***"
              PRINT "DATA="; Data$
            END IF
            NoComp = 4 'Error message received.
            GOTO ContAck
          END IF
          NoComp = 0
          GOTO ContAck
        END IF
      END IF
    WEND
    NoComp = 1 'Time Out, No data at all.
    IF Pmsg THEN PRINT "*** GetData - Error #1 - Time Out, No Data Received ***"
  ContAck: IF NoComp = 0 OR NoComp = 4 THEN 'IF NOCOMP=4, THEN ERROR WAS
    IF INSTR(1, Data$, "ERR 22") THEN 'RECEIVED, A CHECKSUM ERROR
      active = 2 'INDICATING ACKMODE=2
    ELSE
      active = VAL(Data$)
    END IF
  END IF
END IF
END IF
END IF
IF NoComp = 0 THEN
  GetPrompt Msg$, NoComp1, Pmsg
  IF NoComp = 0 THEN NoComp = NoComp1
END IF
END IF
END IF
END SUB

```

```

*****
'* This routine gets the SERVOSTAR drive ACTIVE status flag.
'* The ACTIVE flag indicates if the drive is Enabled ("EN") and if
'* the drive REMOTE ENABLE is On.
'* It returns either a 0 or a 1.
'* It waits for the echo and for the system prompt.
'* GetActive (active%, NoComp%, Pmsg%)
'* Active% = SERVOSTAR drive ACTIVE Flag (0 or 1).
'* NoComp% = Not-Complete flag:
'* - If 0 then Command was received, echoed, and prompted.
'* - If 1 then The routine timed out.
'* - If 2 then Not enough echo characters were received before timeout.
'* - If 3 then Echo & Command mismatch.
'* - If 4 then An Error Message was received.
'* Pmsg% = Print Message if equal to 1.
*****
SUB GetActive (active%, NoComp%, Pmsg%)
    active = 0
    GetAckmode ackmode, NoComp, 0
    IF ackmode = 2 THEN
        Cmnid$ = "ACTIVEBC" + CHR$(13)
    ELSE
        Cmnid$ = "ACTIVE" + CHR$(13)
    END IF
    Msg$ = "GetActive-Cmnid=ACTIVE"
    PutStringWait Cmnid$, Msg$, 2, NoComp, 1
    IF NoComp = 0 THEN
        GetEcho Cmnid$, Msg$, Echo$, 4, NoComp, 1 'ENSURES CORRECT ECHO WAS
        IF NoComp = 0 THEN 'RECIEVED
            GetData Data$, Msg$, 2, NoComp, 1
            IF NoComp = 0 THEN
                active = VAL(Data$) 'ASSIGNS VALUE OF ACTIVE FLAG
            END IF
        END IF
    END IF
    IF NoComp = 0 OR NoComp > 2 THEN
        GetPrompt Msg$, NoComp1, 1
        IF NoComp = 0 THEN NoComp = NoComp1
    END IF
END IF
END SUB

```

```

*****
'* This routine gets Data from the SERVOSTAR up to the first <lf>.
'*
'* GetData (Data$, TimeOut%, NoComp%, Pmsg%)
'*   Data$      = Data from the Com Port in String format.
'*   TimeOut%   = Total time to wait for string in Seconds.
'*   NoComp%    = Not-Complete flag:
'*       - If 0 then Data was received.
'*       - If 1 then the routine timed out.
'*       - If 4 then an Error Message was received.
'*       Data$ contains the error message on return.
'*   Pmsg%      = Print Message if equal to 1.
*****
SUB GetData (Data$, Msg$, Timeout%, NoComp%, Pmsg%)
  BL$ = CHR$(7)
  LF$ = CHR$(10)
  Cr$ = CHR$(13)

  Data$ = ""
  TOut! = TIMER + Timeout
  WHILE TOut! > TIMER
    IF LOC(1) THEN
      A$ = INPUT$(1, #1)
      IF A$ >= " " OR A$ = BL$ THEN Data$ = Data$ + A$ 'ADDS CHARACTERS TO STRING
      IF A$ = LF$ THEN 'CHECK FOR -->
        x = INSTR(1, Data$, BL$) 'CHARACTER POS OF BELL
        'TEST FOR BELL CHARACTER, INDICATING AN ERROR MESSAGE
        IF x THEN
          TOut! = TIMER + Timeout
          WHILE (TOut! > TIMER) AND (INSTR(1, Data$, "->") = 0)
            IF LOC(1) THEN Data$ = Data$ + INPUT$(LOC(1), #1)
          WEND
          IF Pmsg THEN
            PRINT Msg$
            PRINT "*** Getdata - Error #4 - Error Message Received ***"
            PRINT "DATA="; Data$
          END IF
          NoComp = 4 'Error message received.
          EXIT SUB
        END IF
      NoComp = 0
      EXIT SUB
    END IF
  END IF
WEND
NoComp = 1 'Time Out, No data at all.
IF Pmsg THEN PRINT "*** GetData - Error #1 - Time Out, No Data Received ***"
END SUB

```

```

*****
'* This routine Gets Echo from the SERVOSTAR and compare it to the Command.
'*
'* GetEcho (Cmdn$, Echo$, Msg$, TimeOut%, NoComp, Pmsg%)
'*   Cmdn$      = Command String to check echo against.
'*   Echo$      = Echo String received.
'*   Msg$       = Message in case of Error; Helps keep track of where during
'*               execution
'*   TimeOut%   = Total time to wait for string in Seconds.
'*   NoComp     = Not-Complete flag:
'*       - If 0 then the Echo matched the Cmdn$ String.
'*       - If 1 then The routine timed out.
'*       - If 2 then Not enough echo characters were received before timeout.
'*       - If 3 then Echo & Command mismatch.
'*       - If 4 then An Error Message was received.
'*       Echo$ contains the error message on return.
'*   Pmsg%      = Print Message if equal to 1.
*****
SUB GetEcho (Cmdn$, Echo$, Msg$, Timeout%, NoComp%, Pmsg%)
  BL$ = CHR$(7)
  LF$ = CHR$(10)
  Cr$ = CHR$(13)
  Echo$ = ""
  TOut! = TIMER + Timeout
  NumChar = LEN(Cmdn$) + 1      '(Cmdn$+<Lf>)
  WHILE LOC(1) < NumChar AND TOut! > TIMER: WEND
  IF LOC(1) = 0 THEN
    NoComp = 1      'Time Out, No echo at all.
    IF Pmsg THEN
      PRINT Msg$
      PRINT "*** GetEcho - Error #1 - Time Out, No Echo ***"
    END IF
    EXIT SUB
  ELSEIF LOC(1) < NumChar THEN
    NoComp = 2      'Not enough echo characters.
    IF Pmsg THEN
      PRINT Msg$
      PRINT "*** GetEcho - Error #2 - Not Enough Echo Characters ***"
    END IF
    EXIT SUB
  ELSE
    NoComp = 0
  END IF
  Echo$ = INPUT$(NumChar, #1)
  IF Echo$ <> Cmdn$ + LF$ THEN
    NoComp = 3      'Echo & Command mismatch.
    IF Pmsg THEN
      PRINT Msg$
      PRINT "*** GetEcho - Error #3 - Echo & Command Did Not Match ***"
    END IF
    WHILE (TOut! > TIMER) AND (INSTR(1, Echo$, "->") = 0): WEND
    IF LOC(1) THEN Echo$ = Echo$ + INPUT$(LOC(1), #1)
    x = INSTR(1, Echo$, BL$)
    IF x THEN
      'test for Bell character
      WHILE (TOut! > TIMER) AND (INSTR(1, Echo$, "->") = 0): WEND
      IF LOC(1) THEN Echo$ = Echo$ + INPUT$(LOC(1), #1)
      NoComp = 4      'Error message.
      y = INSTR(x, Echo$, Cr$) - x
      IF y >= 0 THEN Echo$ = MID$(Echo$, x, y)
      IF Pmsg THEN
        PRINT Msg$
        PRINT "*** GetEcho - Error #4 - Error Message Received ***"
        PRINT Echo$
      END IF
    END IF
  END IF
END SUB

```

```

*****
' * This routine Gets the Value of a User Flag from the SERVOSTAR.
' *
' * GetFlag (Flag%, Value%, NoComp%, Pmsg%)
' *   Flag%      = Flag number to get the value of (0-50).
' *   Value%     = Value of User Flag (0 or 1).
' *   NoComp%    = Not-Complete flag:
' *     - If 0 then Data was received.
' *     - If 1 then the routine timed out.
' *     - If 2 then Not enough echo characters were received before timeout.
' *     - If 3 then Echo & Command mismatch.
' *     - If 4 then An Error Message was received.
' *     - If 10 then User Flag Number too Big.
' *     - If 11 then User Flag Number too Small.
' *     - If 12 then Flag Value out of bounds - too Big.
' *     - If 13 then Flag Value out of bounds - too Small.
' *   Pmsg%      = Print Message if equal to 1.
*****
SUB GetFlag (flag%, Value%, NoComp%, Pmsg%)
  Value = 0
  Msg$ = "GetFlag - Command="
  IF flag > 50 THEN
    NoComp = 10
    PRINT Msg$
    IF Pmsg THEN PRINT "*** GetFlag - Error #10 - User Flag Number too Big ***"
    EXIT SUB
  ELSEIF flag < 0 THEN
    NoComp = 11
    PRINT Msg$
    IF Pmsg THEN PRINT "*** GetFlag - Error #11 - User Flag Number too Small ***"
    EXIT SUB
  END IF
  IF Value > 1 THEN
    NoComp = 12
    PRINT Msg$
    IF Pmsg THEN PRINT "*** GetFlag - Error #12 - Flag Value out of bounds - too Big ***"
    EXIT SUB
  ELSEIF Value < 0 THEN
    NoComp = 13
    PRINT Msg$
    IF Pmsg THEN PRINT "*** GetFlag - Error #13 - Flag Value out of bounds - too Small ***"
    EXIT SUB
  END IF
  Cmdn$ = "XS" + MID$(STR$(flag), 2) + "[1]" + CHR$(13)
  PutStringWait Cmdn$, Msg$, 2, NoComp, Pmsg
  IF NoComp = 0 THEN
    GetEcho Cmdn$, Echo$, Msg$, 4, NoComp, Pmsg
    IF NoComp = 0 THEN
      GetData Data$, Msg$, 2, NoComp, Pmsg
      IF NoComp = 0 THEN
        Value = VAL(Data$)
      ELSE
        Value = 0
      END IF
    END IF
  END IF
  IF NoComp = 0 OR NoComp > 2 THEN
    GetPrompt Msg$, NoComp1, Pmsg
    IF NoComp = 0 THEN NoComp = NoComp1
  END IF
END SUB

```

```

*****
'* This routine receives all data until a SERVOSTAR Prompt or until a timeout
'* of 4 seconds of trying.
'*
'* GetPrompt (Msg$, NoComp%, Pmsg%)
'*   Msg$      = Message in case of Error; Helps keep track of where during
'*               execution
'*   NoComp%   = Not-Complete flag:
'*       - If 0 then the Prompt was received.
'*       - If 1 then the Routine timed out.
'*       - If 4 then an Error Message was received.
'*   Pmsg%     = Print Message if equal to 1.
*****
SUB GetPrompt (Msg$, NoComp%, Pmsg%)
  BL$ = CHR$(7)
  LF$ = CHR$(10)
  Cr$ = CHR$(13)
  Timeout = 4
  TOut! = TIMER + Timeout
  NoComp = 0
  x$ = ""

  WHILE (INSTR(1, x$, "->") = 0)
    IF LOC(1) THEN x$ = x$ + INPUT$(LOC(1), #1)
    IF TIMER > TOut! THEN
      NoComp = 1 'Time Out
      IF Pmsg THEN
        PRINT Msg$
        PRINT "*** GetPrompt - Error #1 - Time Out, No Echo ***"
      END IF
      EXIT SUB
    END IF
  WEND

  x = INSTR(1, x$, BL$)
  IF x THEN
    'test for Bell character
    IF LOC(1) THEN x$ = x$ + INPUT$(LOC(1), #1)
    NoComp = 4 'Error message.
    IF Pmsg THEN
      PRINT Msg$
      PRINT "*** GetPrompt - Error #4 - Error Message Received ***"
      y = INSTR(x, x$, Cr$) - x
      IF y >= 0 THEN PRINT MID$(x$, x, y)
    END IF
  END IF
END SUB

```

```

*****
'* This routine gets the SERVOSTAR drive READY status flag.
'* The READY flag indicates if the drive is Enabled ("EN").
'* It returns either a 0 or a 1.
'* It waits for the echo and for the system prompt.
'*
'* GetEnable (ready%, NoComp%, Pmsg%)
'* Ready% = SERVOSTAR® drive READY Flag (0 or 1).
'* NoComp% = Not-Complete flag:
'* - If 0 then Command was received, echoed, and prompted.
'* - If 1 then The routine timed out.
'* - If 2 then Not enough echo characters were received before timeout.
'* - If 3 then Echo & Command mismatch.
'* - If 4 then An Error Message was received.
'* Pmsg% = Print Message if equal to 1.
*****
SUB GetReady (ready%, NoComp%, Pmsg%)
    ready = 0
    Msg$ = "GetReady - Command=Ready"
    GetAckmode ackmode, NoComp, 0
    IF ackmode = 2 THEN
        Cmnd$ = "READY75" + CHR$(13)
    ELSE Cmnd$ = "READY" + CHR$(13)
    END IF
    PutStringWait Cmnd$, Msg$, 2, NoComp, Pmsg
    IF NoComp = 0 THEN
        GetEcho Cmnd$, Echo$, Msg$, 4, NoComp, Pmsg
        IF NoComp = 0 THEN
            Msg$ = "GetReady-Cmnd=READY"
            GetData Data$, Msg$, 2, NoComp, Pmsg
            IF NoComp = 0 THEN
                ready = VAL(Data$)
            END IF
        END IF
    END IF
    IF NoComp = 0 OR NoComp > 2 THEN
        GetPrompt Msg$, NoComp1, Pmsg
        IF NoComp = 0 THEN NoComp = NoComp1
    END IF
END SUB

```

```

*****
'* This routine gets the SERVOSTAR drive REMOTE status flag.
'* The REMOTE flag indicates if the drive is Enabled ("EN") by the Remote
'* switch.
'* It returns either a 0 or a 1.
'* It waits for the echo and for the system prompt.
'*
'* GetRemote (remote%, NoComp%, Pmsg%)
'* Remote% = SERVOSTAR® drive REMOTE Flag (0 or 1).
'* NoComp% = Not-Complete flag:
'* - If 0 then Command was received, echoed, and prompted.
'* - If 1 then The routine timed out.
'* - If 2 then Not enough echo characters were received before timeout.
'* - If 3 then Echo & Command mismatch.
'* - If 4 then An Error Message was received.
'* Pmsg% = Print Message if equal to 1.
*****
SUB GetRemote (remote%, NoComp%, Pmsg%)
    ready = 0
    Msg$ = "GetRemote-Cmnd=REMOTE"
    GetAckmode ackmode, NoComp, Pmsg
    IF ackmode = 2 THEN
        Cmnd$ = "REMOTTECC" + CHR$(13)
    ELSE Cmnd$ = "REMOTE" + CHR$(13)
    END IF
    PutStringWait Cmnd$, Msg$, 2, NoComp, Pmsg
    IF NoComp = 0 THEN
        GetEcho Cmnd$, Echo$, Msg$, 4, NoComp, Pmsg
        IF NoComp = 0 THEN
            GetData Data$, Msg$, 2, NoComp, Pmsg
            IF NoComp = 0 THEN
                remote = VAL(Data$)
            END IF
        END IF
    END IF
    IF NoComp = 0 OR NoComp > 2 THEN
        GetPrompt Msg$, NoComp1, Pmsg
        IF NoComp = 0 THEN NoComp = NoComp1
    END IF
END SUB

```

```

*****
'* This routine Inputs an entire string of data until a <lf> character
'* is received. Control characters are stripped from the string.
'*
'* LineInput (Data$, Timeout%, NoComp%, Pmsg%)
'*   Data$      = Data from the Com Port in String format.
'*   Timeout%   = Total time to wait for String in Seconds.
'*   NoComp%    = Not-Complete flag:
'*     - If 0 then Data was received.
'*     - If 1 then the routine timed out.
'*     - If 2 then the routine received a "->" system prompt.
'*   Pmsg%      = Print Message if equal to 1.
*****
SUB LineInput (Data$, Timeout%, NoComp%, Pmsg%)
  LF$ = CHR$(10)
  Cr$ = CHR$(13)
  Data$ = ""
  TOut! = TIMER + Timeout
  Msg$ = "LineInput - Command= "

  WHILE TOut! > TIMER
    IF LOC(1) THEN
      A$ = INPUT$(1, #1)
      IF A$ >= " " THEN Data$ = Data$ + A$ 'ADD DATA TP CHARACTER STRING
      IF A$ = LF$ THEN
        NoComp = 0
        EXIT SUB
      ELSEIF INSTR(1, Data$, "->") THEN
        NoComp = 2
        EXIT SUB
      END IF
    END IF
  WEND
  NoComp = 1 'Time Out, No echo at all.
  IF Pmsg THEN
    PRINT Msg$
    PRINT "*** Error #1 - Time Out, No Echo ***"
  END IF
END SUB

*****
'* This routine Puts a command to the SERVOSTAR. It waits for the echo and
'* the system prompt. It assumes you are in Ackmode = 2
'*
'* PutAckCommand (Cmnd$, NoComp%, Pmsg%)
'*   Cmnd$      = Any SERVOSTAR command in a String format.
'*   A Carrage Return is automatically appended to the Cmnd$ string.
'*   NoComp%    = Not-Complete flag:
'*     - If 0 then Command was received, echoed, and prompted.
'*     - If 1 then The routine timed out.
'*     - If 2 then Not enough echo characters were received before timeout.
'*     - If 3 then Echo & Command mismatch.
'*     - If 4 then An Error Message was received.
'*   Pmsg%      = Print Message if equal to 1.
*****
SUB PutAckCommand (Cmnd$, NoComp%, Pmsg%)
  Cmnd$ = Cmnd$ + CHR$(13)
  PutStringWait Cmnd$, Msg$, 2, NoComp, Pmsg
  IF NoComp = 0 THEN
    GetEcho Cmnd$, Echo$, Msg$, 4, NoComp, Pmsg
  END IF
  IF NoComp = 0 OR NoComp > 2 THEN
    GetPrompt Msg$, NoComp1, Pmsg
    IF NoComp = 0 THEN NoComp = NoComp1
  END IF
END SUB

```

```

*****
' * This routine Puts a command to the SERVOSTAR.
' * It waits for the echo and system prompt.
' *
' * PutCommand (Cmnd$, NoComp%, Pmsg%)
' *   Cmnd$      = AnySERVOSTAR command in a String format.
' *   A Carrage Return is automatically appended to the Cmnd$ string.
' *   NoComp%    = Not-Complete flag:
' *     - If 0 then Command was received, echoed, and prompted.
' *     - If 1 then The routine timed out.
' *     - If 2 then Not enough echo characters were received before timeout.
' *     - If 3 then Echo & Command mismatch.
' *     - If 4 then An Error Message was received.
' *   Pmsg%      = Print Message if equal to 1.
*****
SUB PutCommand (Cmnd$, NoComp%, Pmsg%)
  Cmnd$ = Cmnd$ + CHR$(13)
  Msg$ = "PutCommand - Command=" + Cmnd$
  PutStringWait Cmnd$, Msg$, 2, NoComp, Pmsg
  IF NoComp = 0 THEN
    GetEcho Cmnd$, Echo$, Msg$, 4, NoComp, Pmsg
  END IF
  IF NoComp = 0 OR NoComp > 2 THEN
    GetPrompt Msg$, NoComp1, Pmsg
    IF NoComp = 0 THEN NoComp = NoComp1
  END IF
END SUB

*****
' * This routine Puts a command to the SERVOSTAR
' * It returns data generated by the command.
' * It waits for the echo and system prompt.
' *
' * PutCommand (Cmnd$, Data$, NoComp%, Pmsg%)
' *   Cmnd$      = Any SERVOSTAR® command in a String format.
' *   Data$      = Data generated by Cmnd$ in string format.
' *   NoComp%    = Not-Complete flag:
' *     - If 0 then Command was received, echoed, and prompted.
' *     - If 1 then The routine timed out.
' *     - If 2 then Not enough echo characters were received before timeout.
' *     - If 3 then Echo & Command mismatch.
' *     - If 4 then An Error Message was received.
' *   Pmsg%      = Print Message if equal to 1.
*****
SUB PutGetCommand (Cmnd$, Data$, NoComp%, Pmsg%)
  Data$ = ""
  Cmnd$ = Cmnd$ + CHR$(13)
  Msg$ = "PutGetCommand-Cmnd=" + Cmnd$
  PutStringWait Cmnd$, Msg$, 2, NoComp, Pmsg
  IF NoComp = 0 THEN
    GetEcho Cmnd$, Echo$, Msg$, 4, NoComp, Pmsg
    IF NoComp = 0 THEN
      GetData Data$, Msg$, 5, NoComp, Pmsg
      IF NoComp THEN
        Data$ = ""
      END IF
    END IF
  END IF
  IF NoComp = 0 OR NoComp > 2 THEN
    GetPrompt Msg$, NoComp1, Pmsg
    IF NoComp = 0 THEN NoComp = NoComp1
  END IF
END SUB

```

```

*****
' * This routine Puts a command to the SERVOSTAR.
' * It returns the multiple line data generated by the command.
' * It waits for the echo and system prompt.
' *
' * PutGetMultipleCommand (Cmnd$, Data$, NoComp%, Pmsg%)
' *   Cmnd$      = Any SERVOSTAR command in a String format.
' *   Data$      = Data generated by Cmnd$ in string format.
' *   NoComp%    = Not-Complete flag:
' *     - If 0 then Command was received, echoed, and prompted.
' *     - If 1 then The routine timed out.
' *     - If 2 then Not enough echo characters were received before timeout.
' *     - If 3 then Echo & Command mismatch.
' *     - If 4 then An Error Message was received.
' *   Pmsg%      = Print Message if equal to 1.
*****
SUB PutGetMultipleCommand (Cmnd$, Data$, NoComp%, Pmsg%)
  Cr$ = CHR$(13)
  Msg$ = "PutGetMultipleCommand - Command= " + Cmnd$
  Data$ = ""
  Cmnd1$ = Cmnd$ + CHR$(13)
  PutStringWait Cmnd1$, Msg$, 2, NoComp, 0
  IF NoComp = 0 THEN
    GetEcho Cmnd1$, Echo$, Msg$, 2, NoComp, 0
    IF NoComp = 0 THEN
      Varrecl:
        LineInput Text$, 4, NoComp, 0  'RECIEVES A FULL LINE OF DATA
        IF NoComp = 0 THEN
          Data$ = Data$ + Text$ + Cr$  'ADD TO CHARACTER STRING
          GOTO Varrecl                'GET NEXT LINE OF DATA
        ELSEIF NoComp = 2 THEN
          EXIT SUB
        END IF
        IF Pmsg THEN
          PRINT Msg$
          PRINT "*** Error #1 - Time Out, No System Prompt ***"
        END IF
      END IF
    END IF
  END SUB

*****
' * This routine Puts a String to the SERVOSTAR Fast.
' * This does not wait for the SERVOSTAR to echo anything.  This is useful for
' *   when the drive is not expected to echo anything, such as while in
' *   MULTIDROP Mode or when ECHO=0.
' *
' * PutStringWaitFast (Data$)
' *   Data$      = Any String data to send to the SERVOSTAR.
*****
SUB PutStringFast (Data$)
  PRINT #1, Data$;
END SUB

```

```

*****
'* This routine Puts a String to the SERVOSTAR.
'* Each character is sent to the SERVOSTAR one at a time and after each
'* character is sent the routine waits for the echo back from the SERVOSTAR
'* before sending the next character to the SERVOSTAR.
'* All commands sent with this routine need to produce echo.
'* i.e. MultiDrop axis select commands don't produce echo.
'* The SERVOSTAR flag ECHO must be On.
'*
'* PutStringWait (Data$, Msg$, CharDelay%, NoComp%, Pmsg%)
'* Data$ = Any String data to send to the SERVOSTAR.
'* Msg$ = Message in case of Error; Helps keep track of where during
'* execution
'* CharDelay% = Time to delay between character echos in seconds.
'* NoComp% = Not-Complete flag:
'* - If 0 then Data was received.
'* - If 1 then the routine timed out.
'* - If 2 then Not enough echo characters were received before timeout.
'* - If 3 then Echo & Command mismatch.
'* - If 4 then An Error Message was received.
'* Pmsg% = Print Message if equal to 1.
*****
SUB PutStringWait (Data$, Msg$, CharDelay%, NoComp%, Pmsg%)
  FOR DataChar = 1 TO LEN(Data$)
    x = LOC(1)
    Delay! = TIMER + CharDelay
    PRINT #1, MID$(Data$, DataChar, 1);
    WHILE x = LOC(1) AND Delay! > TIMER: WEND
    IF LOC(1) = x THEN
      NoComp = 1
      IF Pmsg THEN
        PRINT Msg$
        PRINT "*** PutStringWait - Error #1 - Data Echo Time Out ***"
      END IF
      EXIT SUB
    END IF
  NEXT
  NoComp = 0
END SUB

```