

## Using STX and ETX command strings

Following these procedures to effectively write and read STX and ETX command strings to your drive. This sample code will show you how to add the STX and ETX codes to your data string for transmission. It will also show you how to strip these codes from incoming transmissions so that you can use the incoming data.

Below is how to setup the program in the builder pseudocode, using the user code insert function to add your own basic lines. This first section sets up your RS422 communication on COM port 1, and sets up the data string that is being sent (msg\$).

```
GOTO SCREEN begin
*>SCREEN begin
- TITLE:::::::::::: "Communicating Using STX and ETX" ::::::::::::::
  RS422 #1,ON
  stx$=chr$(2):etx$=chr$(3)
  msg$="003010300"
  gosub putdat
  gosub readloop
```

This next section puts the data out by adding your STX character to the front of your MSG string, and the ETX character to the end of the string, and then prints it to COM port 1.

```
- LABEL putdat
  msg1$=stx$+msg$+etx$
  print #1,msg1$;
  return
```

This section reads the COM port, discarding anything it finds until it runs across the STX character. When that happens it jumps falls through to the GETDAT loop.

```
- LABEL readloop
  c$=inkey$(#1)
  if c$="" then goto readloop
  if c$=<=>stx$ then goto readloop
```

This loop reads each character sequentially, and adds them to the RESP\$ string it collects each loop around. When it runs into the ETX character it stops reading and building the string, and returns. The RESP\$ string then contains the transmitted data that can be used in your program.

```
- LABEL getdat
  c$=inkey$(#1)
  if c$="" then goto getdat
  if c$=etx$ then return
  resp$=resp$+c$
  goto getdat
- END OF PSEUDOCODE
```