www.triomotion.com

T E C H N I C A L N O T E

Doc No.:	**TN20-99**
Date:	**10 July 2007**
Version:	**1.1**
Subject:	**SD Flash card operation in Motion Coordinator**

Overview

Trio's SDCARD support provides a hardware interface to the widely available SDCARD flash devices, and software support for the SDCARD flash memory using the FAT32 file system. This system gives the Trio programmer large amounts storage for programs and projects while allowing fast and easy file copying from the PC platform to the Motion Coordinator platform.

Hardware and Software Platform

The SDCARD is supported in Trio's MC206X, MC224 and some custom designed Motion Coordinators. The SD-compatible controller must have software version v1.6629+ installed.

Format the SDCARD (FAT32)

The Trio controller and SDCARD support FAT32 file format system. Before using the SDCARD in a controller it must first be formatted in Windows using the FAT32 file system. This can be done by selecting "My Computer" from the Windows Desktop, then right click the SD flashdrive. Select "Format" from the menu and choose FAT32.

Controller Boot

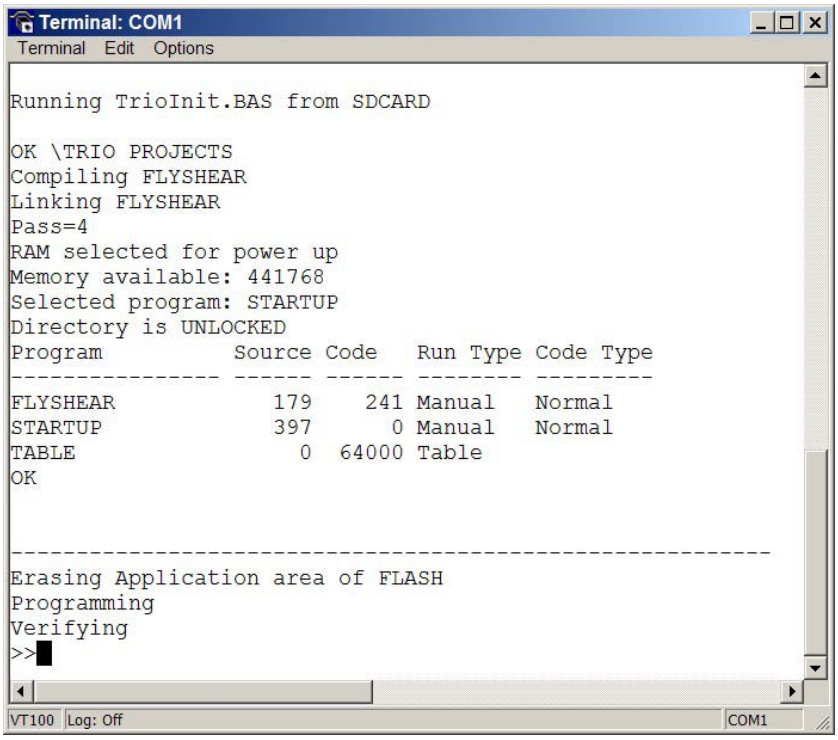
To allow the Motion Coordinator to automatically boot-up from the SD card on power up, a special file, named TRIOINIT.BAS can be included in the root directory of the SD card.

`TRIOINIT.BAS file`

At power up, the controller checks for the presence of an SDCARD with a valid format. If one is found then the controller searches for the script file TRIOINIT.BAS on the SDCARD. It must be in the root directory. TRIOINIT.BAS is a text file that can contain any valid TrioBASIC command which can be executed in a single line on the terminal. These commands will execute on the Command Line ">>". If TRIOINIT.BAS is found it is opened and the command list sequence is executed. If no TRIOINIT.BAS file is found, no action is taken and the controller boots as normal. File execution is stopped when the end of file is found, or an error occurs in a command.

Example 1 - TRIOINIT.BAS file to load a specific project at boot up -

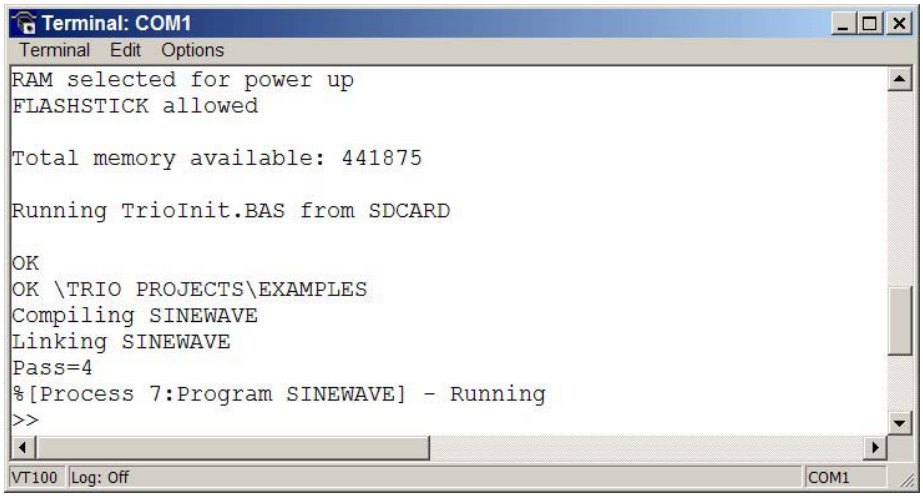
```
'=========================================================================
' Application:  SDCARD startup file
' Module:       TRIOINIT.BAS
' Platform:     Any with SD card (Note, this file resides on the SD card)
'-------------------------------------------------------------------------
PRINT ""
FILE "cd" "trio projects" 'change directory
LOAD_PROJECT "flying shear" 'load desired project
DIR 'shows the programs now loaded on the controller
PRINT ""
PRINT "-----------------------------------------------------------"
EPROM 'Save newly loaded project to controller's flash memory
```



Example 2 - TRIOINIT.BAS file to load and run a selected program at boot up based on a VR variable setting -

```
'=========================================================================
' Application:  SDCARD startup file
' Module:       TRIOINIT.BAS
' Platform:     Any with SD card (Note, this file resides on the SD card)
'
' This TRIOINIT file will select a specific program and load it based on a
' previously set VR(0) variable. VR(0) is global and non-volatile.
'-------------------------------------------------------------------------
PRINT ""
NEW ALL 'Clear controller RAM
FILE "CD" "Trio Projects\\Examples"
IF VR(0)<> 1 AND VR(0)<>2 AND VR(0)<>3 THEN FILE "load_program" "cam"
IF VR(0)=1 THEN FILE "load_program" "clock"
IF VR(0)=2 THEN FILE "load_program" "sinewave"
IF VR(0)=3 THEN FILE "load_program" "looping"
```

```
IF VR(0)<> 1 AND VR(0)<>2 AND VR(0)<>3 THEN RUN "cam"
IF VR(0)=1 THEN RUN "clock"
IF VR(0)=2 THEN RUN "sinewave"
IF VR(0)=3 THEN RUN "looping"
```



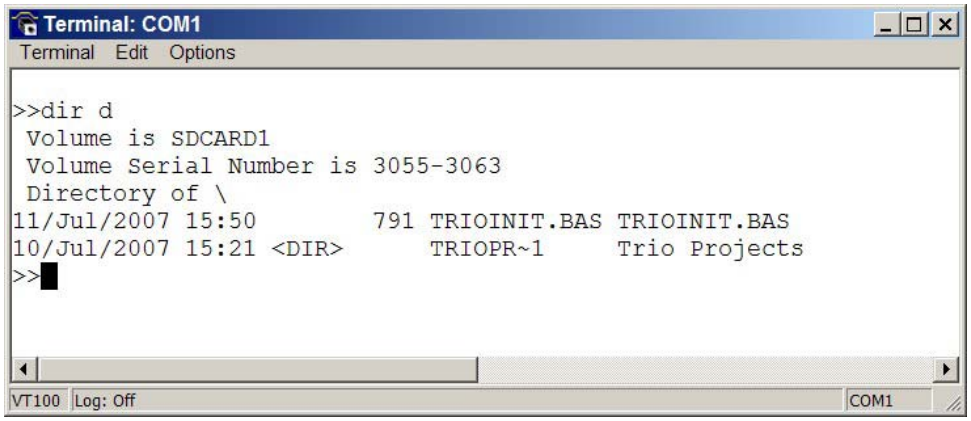
DIR D

Purpose – Display the directory contents of the SDCARD.

Works in: Command Line - ☑, Program - ☒

The **DIR D** command displays the contents of the current directory of the FAT32 file system stored on the SDCARD. Without the D parameter, the **DIR** command displays the contents of the current directory on the controller.

Example of **DIR D** usage at the Command Line ">>"



The directory listing has 5 columns:

- Last modification date.
- Last modification time.
- File size. If the directory item is a directory then this will have the value <DIR>.
- Short name.
- Long name.

STICK_WRITE(flash_file#, table_start[, length[, format]])

Purpose – Write controller TABLE data to the SDCARD.

Works in: Command Line - ☑, Program - ☑

The STICK_WRITE command allows writing a range of controller TABLE data to a file on the SDCARD. The file name given is SD000000.CSV or SD000000.BIN, depending on the data format defined by the **format** setting.

The **flash_file#** parameter specifies the SD file number to be written.

The **table_start** parameter specifies where to begin getting the TABLE data.

The **length** parameter specifies the number of TABLE values to be written to the file.

The **format** parameter specified the type of data file. If **format**=1 then the data is stored in ASCII format and has the extension "CSV", one value per line. If no format is specified, or **format**=0 then the data is stored in IEEE floating point binary format and has the extension "BIN", little-endian, i.e. the least significant byte first.

Note: If an old "NexFlash" flash stick is detected then the former STICK_WRITE command is performed. This means that the **length** and **format** parameters are invalid and will cause an error.

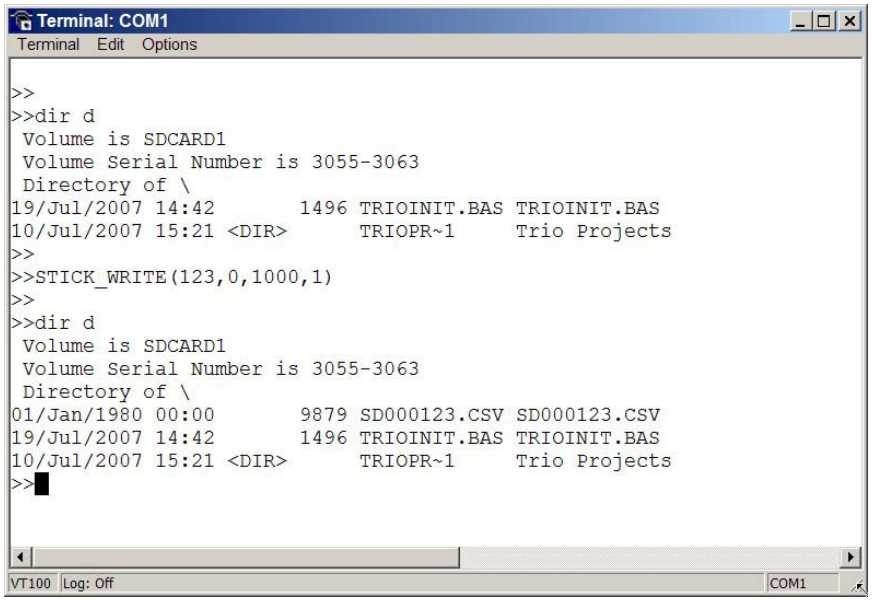
STICK_WRITE is a function and will return TRUE (-1) if the command was performed correctly and FALSE (0) otherwise. It will not cause a TrioBASIC error. If an identical file already exists it is overwritten.

Note that if a lot of files are to be stored to the SD card using **STICK_WRITE**, the time taken to create the new file will increase as more files are saved. The best strategy is to create new subdirectories and store only 10 or 20 data files in each one.

Example –

Write an ASCII file (#123) to the SDCARD storing 1000 TABLE locations starting at 0 (i.e. 0-999)

```
STICK_WRITE(123,0,1000,1)
```



STICK_READ(flash_file#, table_start[, format])

Works in: Command Line - ☑, Program - ☑

Purpose – Read TABLE data stored in a file from the SDCARD to the controller.

All the binary data in the file is read into the TABLE memory area of the Motion Coordinator.

The **flash_file#** parameter specifies the SD file to be opened.

The **table_start** parameter specifies where to begin storing the TABLE data in the controller.

The **format** parameter specifies the type of data. If **format**=1 then the data is read in ASCII format from a file with the extension "CSV", one value per line. If **format**=0 or no format is specified, then the data is read in IEEE floating point binary format, little-endian; i.e. the least significant byte first, from a file with the extension "BIN".

Note: If an old "NexFlash" flash stick is detected then the former STICK_READ command is performed. This means that the **format** parameter is invalid and will cause an error.

STICK_READ is a function and will return TRUE (-1) if the command was performed correctly and FALSE (0) otherwise. It will not cause a TrioBASIC error.

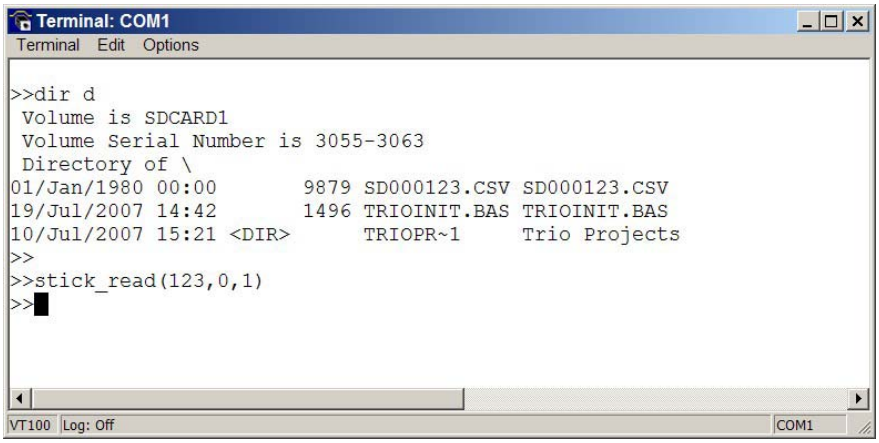
Examples –

Read in data from file #123 in ASCII format, and store on controller starting at TABLE location 0.

```
STICK_READ(123,0,1)
```

Read in data from file #123 in IEEE format, and storing on controller starting at TABLE location 1000.

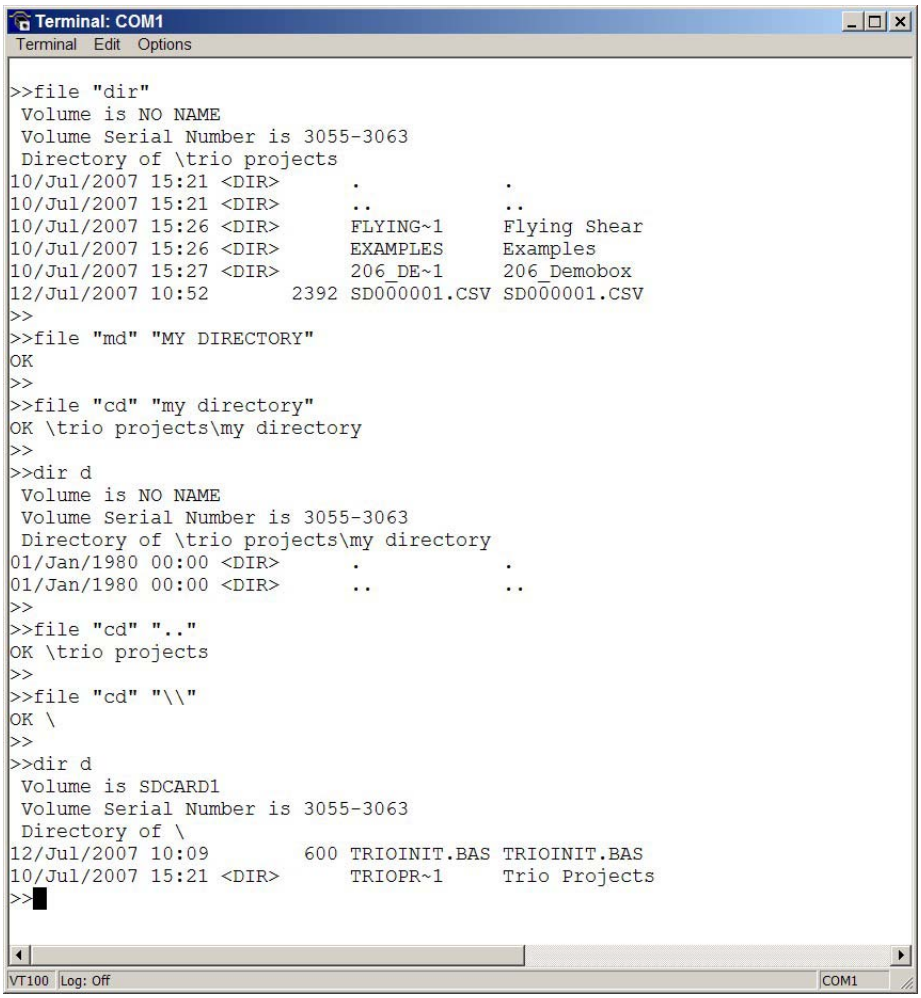
```
STICK_READ(123,1000,0)
```



FILE

All the **FILE** command parameters must be explicitly surrounded by quotes.

Example of **FILE** usage at the Command Line ">>"



FILE "DIR"

Purpose - Reads and displays the contents of the current directory of the FAT file system stored on the SDCARD. A synonym for **DIR D**.

Works in: Command Line - ☑, Program - ☑

FILE "MD" "directoryname"

Purpose - Make a directory within the current directory.

Works in: Command Line - ☑, Program - ☑

If the directory already exists an error is created.

FILE "CD" "directoryname"

Purpose - Change the current directory to a specific directory.

Works in: Command Line - ☑, Program - ☑

Examples -

To move down from the root directory –
```
FILE "CD" "projects"
```

To move down several levels to a specified directory –
```
FILE "CD" "project1\\project2\\project3"
```

To move up one level to the parent directory –
```
FILE "CD" ".."
```

To move up to the root directory –
```
FILE "CD" "\\"
```

FILE "RD" "directoryname"

Purpose - Delete a directory.

Works in: Command Line - ☑, Program - ☑

The directory must be empty of files before you can delete it. If the directory is not empty or does not exist an error is generated.

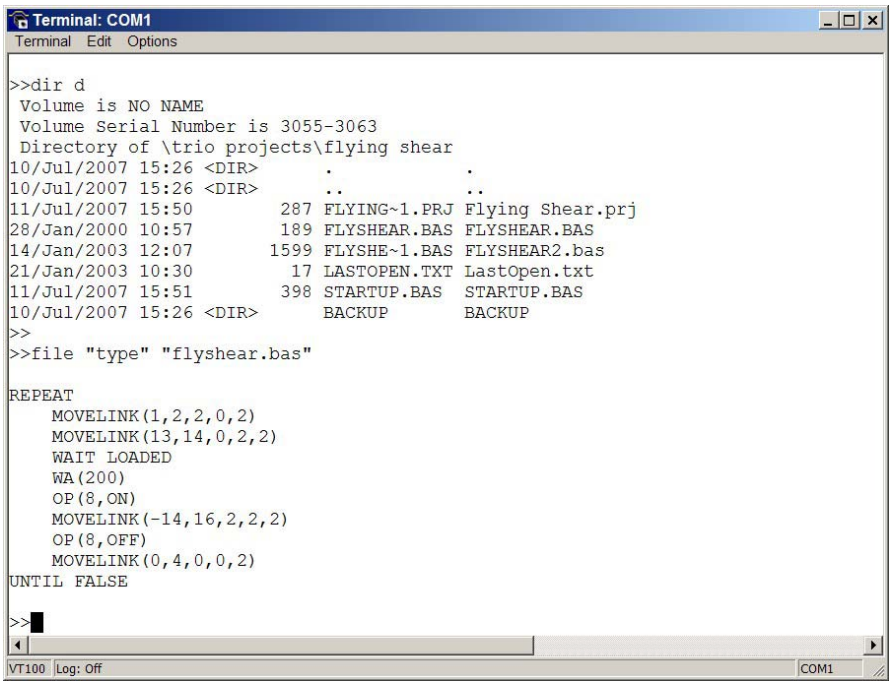
FILE "TYPE" "filename.bas"

Purpose - Show the contents of a file on the SDCARD.

Works in: Command Line - ☑, Program - ☑

The file is on the SD card the file extension (e.g .BAS) must be defined. The file is printed to the Command Line (Port 0) as a text file.

Example of **FILE TYPE** usage at the Command Line ">>"



FILE "DEL" "filename.ext"

Purpose - Delete the file <filename.ext> from the current directory.

Works in: Command Line - ☑, Program - ☑

```
>>dir d
 Volume is NO NAME
 Volume Serial Number is 9C15-9511
 Directory of \
01/Jan/1980 00:00       8192 SD000000.BIN SD000000.BIN
19/Mar/2007 17:11 <DIR>      MC2TES~1     mc2test_224
15/May/2007 11:54           30 TRIOINIT.BAS TRIOINIT.BAS
>>file "del" "SD000000.BIN"
OK
>>dir d
 Volume is NO NAME
 Volume Serial Number is 9C15-9511
 Directory of \
19/Mar/2007 17:11 <DIR>      MC2TES~1     mc2test_224
15/May/2007 11:54           30 TRIOINIT.BAS TRIOINIT.BAS
>>
```

FILE "LOAD_PROGRAM" "programname"

Purpose - Load the program <programname> from the current directory of the SDCARD to the controller.

Works in: Command Line - ☑, Program - ☒

An error at the Command Line will occur if the specified program is not found in the current directory.

FILE "LOAD_PROJECT" "projectname"

Purpose - Load a MotionPerfect project from the SDCARD to the controller.

Works in: Command Line - ☑, Program - ☒

An error at the Command Line will occur if the specified project is not found in the current directory.

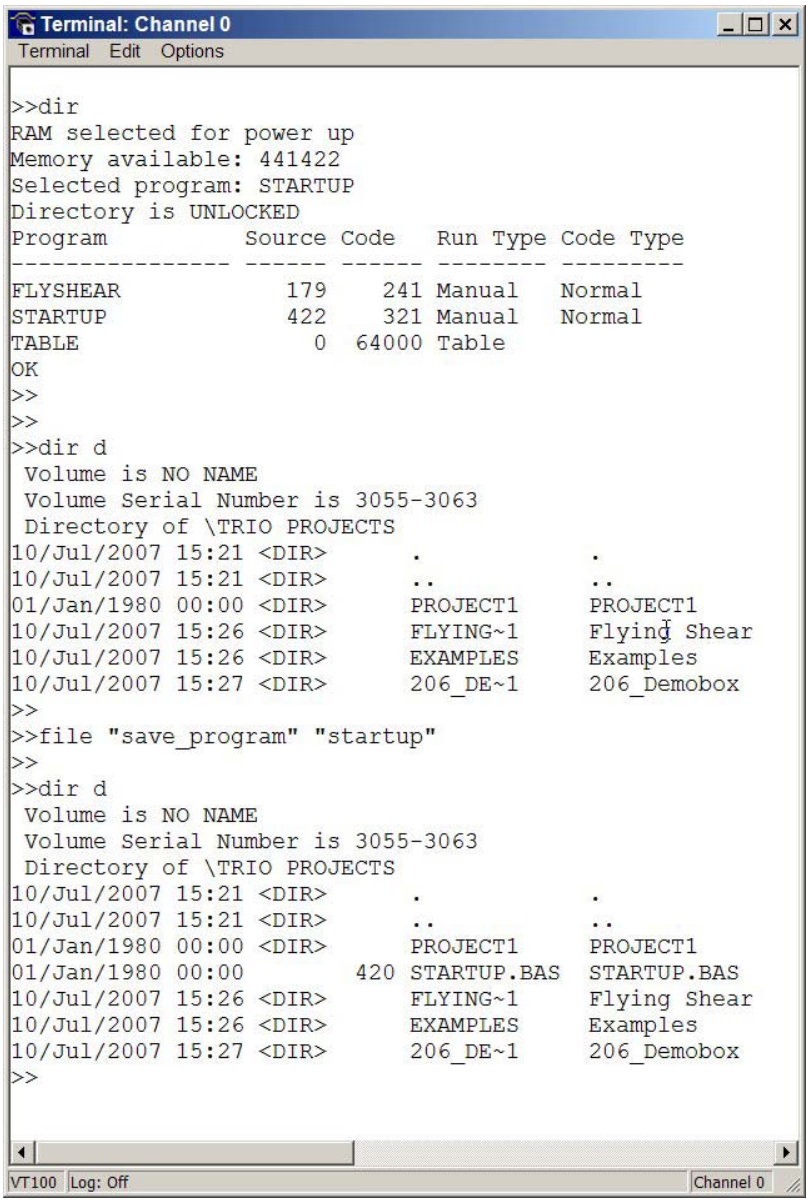
FILE "SAVE_PROGRAM" "programname"

Purpose - Store the program <programname> to the current directory on the SDCARD.

Works in: Command Line - ☑, Program - ☒

The program must exist in the controller's memory. The specified program will be copied to the current directory on the SDCARD. The command automatically adds a .BAS suffix so this must NOT be put in the command.

Example of **FILE SAVE_PROGRAM** at the Command Line ">>"



FILE "SAVE_PROJECT" "projectname"

Purpose – Saves the current project on the controller to the SDCARD.

Works in: Command Line - ☑, Program - ☒

Create the directory **projectname** inside the current directory, make this new directory the current directory, write all the program files on the controller to the directory and generate the **projectname.prj** file so that MotionPerfect can recognize and handle this project correctly.

Current restrictions

1. The controller only supports FAT32 file systems. This restriction will be lifted as soon as possible. If your SD card is not formatted FAT32 it must be formatted FAT32 using Windows before use.
2. Only one Controller process (program) can access the SDCARD at any one time.
3. Encrypted projects are not currently supported. This restriction will be lifted as soon as possible.

Functionality still to be implemented

EPROM(1)

Same as **FILE SAVE_PROJECT**; to provide compatibility with the old NexFlash memory stick.

EPROM(2)

The same as **EPROM(1)**, but creates/overwrites the **TRIOINIT.BAS** file to automatically load the project from SDCARD into RAM, and then is copied to the controller's internal flash memory using **EPROM(0)**.