TRIO
MOTION TECHNOLOGY

PICKUP AND LEARN SERIES

TEXT FILES

Trio Motion Technology

Pick-up And Learn Series

First Edition • 2012

Trio Programming Guides are designed to aid learning of the TrioBASIC language through description and examples. Each one will cover a particular topic and discuss which commands and parameters in the TrioBASIC are required to complete the task.

A general understanding of TrioBASIC is required and it is recommended to attend an introduction to TrioBASIC training course. The programming guides are not a replacement for the TrioBASIC help files which can be found in *Motion* Perfect as well as the manual which cover each command and parameter in more detail and should be referenced when required.

Any examples given in the programming guide will work and have been tested on an isolated controller. If you choose to use these examples on a machine please take care that it will not cause damage or injury and that they are correctly included in the project changing parameters and values where required.

UK | USA | CHINA | INDIA
www.triomotion.com

SAFETY WARNING

During the installation or use of a control system, users of Trio products must ensure there is no possibility of injury to any person, or damage to machinery.
Control systems, especially during installation, can malfunction or behave unexpectedly. Bearing this in mind, users must ensure that even in the event of a malfunction or unexpected behaviour the safety of an operator or programmer is never compromised.

This doucment uses the following icons for your reference:

Information that relates to safety issues and critical software information

Information to highlight key features or methods.

Useful tips and techinques.

Example programs

Contents

Text files

INTRODUCTION

Text files are extremely flexible, they can be used to input data to a machine as well as reporting back statistics. A definition of a text file is a file that holds textual data, this is independent of the file extension. Two familiar file extensions are .txt and .bas which are for plain text documents and TrioBASIC programs respectively though many more exist. In simple terms if the file can be viewed in Notepad or similar application then it is a text file.

As text files do not have any automatic operation a program must be written on the controller to read from or write to them. There are various features built into the Trio Motion Technology products that simplify the loading, saving, editing and parsing of text files. These include commands in the TrioBASIC language as well as functions in PCMotion ActiveX.

This guide will provide an introduction to using text files on the Trio Motion Technology products including the following:

- How are text files stored on the controller
- How to load text files to the controller
- How the files can be accessed through the TrioBASIC

File storage

Every controller has two areas of memory available to the user, the built in program memory and the removable SD card memory. Both of these memory areas are capable of storing a variety of files, this guide will concentrate on how they can be used to store text files.

The SD card uses a similar file structure as a PC and so will store the text file directly, using the same file extension as is seen on a PC.

The controller memory is where the project is stored, it is possible to store text files as a file in the project and so it will be synchronised with the project on the PC when connected to Motion Perfect. If it is not required to synchronise the text file with the project then it can be saved as a text file in the program memory. A third option is to not save the file to the controller at all, but to place the text into a **FIFO** buffer. The **FIFO** buffers will also use the program memory.

As program memory is retained then any text file or **FIFO** buffer stored here is also retained on a power cycle and when using the '**EX**' TrioBASIC command.

It is important to select the most appropriate file location for the application. In general if the text file is going to be used multiple times then it should be a file. If it is large or if there are many files then they should be stored on the SD card. If the file only needs to be read once and the transfer time is significant to the cycle time then a **FIFO** can be used.

Streaming a file to a **FIFO** allows you to start to read/ process the file before it has completed transferring. This can reduce cycle time if the file transfer time is significant.

PROJECT TEXT FILE

The main use of the program memory is to hold the project. All files in the project are synchronised with the copy of a project on a PC, when Motion Perfect is connected in 'Sync' mode. This includes all TrioBASIC files, **MC_CONFIG**, IEC tasks and of course text files.

This means that it is then possible to use Motion Perfect to create, view and edit text files on the controller. The file can then be read by a TrioBASIC program during the operation of the machine.

To prevent project synchronisation problems it is recommended to only use *Motion* Perfect to create or edit project text files.

A 'project text file' has the type 'Text'. As with all text files they cannot be run like a TrioBASIC program and so have 'None' for the run type. If the program memory directory is read using the DIR command or through Motion Perfect it will display something like the following:

```
>>DIR
EPROM selected for power up
Memory available: 7077806
Selected program: NOTES
Directory is UNLOCKED
Program          Source     Code    Run Type       Code Type
-----------------------------------------------------------------
MC_CONFIG          38         34    Power Up       MC_CONFIG
READ_GCODE       6421       12680   Manual         Normal
TFL              1826         900   Manual         Normal
NOTES              82           0   None           Text
OK
>>
```

In the above example the **NOTES** file is type 'Text' and that run type is 'None' as text files cannot compile or run. No file extensions are used on files in program memory; this is replaced with the 'Code Type'.

As project files can be created, edited or deleted by *Motion* Perfect or by a TrioBASIC program a lock is enabled when the project text file is opened into the *Motion* Perfect editor or when opened by a TrioBASIC program. This prevents *Motion* Perfect or the TrioBASIC program from editing the file if the other has it open.

TEXT (TEMPORARY) FILE IN PROGRAM MEMORY

A text file in the program memory is very similar to the project text file apart from it is not part of the project. This means that it is in the same memory area but is not synchronised to the project and so not viewable or editable by Motion Perfect.

The text file in program memory is known as a temporary file in the controller and so has the type 'Temp'. Again like the project text file it cannot be run like a TrioBASIC program and so has the run type 'None'

Even though a text file in program memory is called a temporary file it is still in the program memory and so retained on power cycle and when using the '**EX**' TrioBASIC command.

If the program memory directory is read using the DIR command or through Motion Perfect it will display something like the following:

```
>>DIR
EPROM selected for power up
Memory available: 3810108
Selected program: GCODE
Directory is UNLOCKED
Program        Source         Code    Run Type       Code Type
-------------------------------------------------------------------
MC_CONFIG            20           17    Power Up       MC_CONFIG
READ_GCODE         2502          932    Manual         Normal
TFL                 577            0    Manual         Normal
GCODE           3792068            0    None           Temp
OK
>>
```

In the above example the **GCODE** file is type 'Text' and that run type is 'None' as text files cannot compile or run. No file extensions are used on files in program memory; this is replaced with the 'Code Type'.

If you need to delete the Temp file you can use the **DEL** command on the command line or in a program.

SD CARD

SD cards in the Trio Motion Technology products use the same FAT32 file structure as many PC's. This means that the files can be directly copied onto the SD card from the PC and they retain all the file extensions. This means that any file can be saved onto the SD card, however the TrioBASIC is only able to interact with text files.

The SD card has much larger storage than the controllers built in memory. This leads to two main advantages over any type of file in the program memory:

- The **SD** card can hold hundreds of files compared to the 32 in the controller memory
- The **SD** card can hold much larger files

The **SD** card has many other functions as well as just being a file store, see the **SD** card guide for more details on its capabilities.

If the SD card directory is read using the DIR D command or through Motion Perfect it will display something like the following:

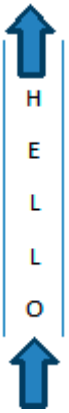
```
>>DIR D
 Volume is NO NAME
 Volume Serial Number is 0EC1-442B
 Directory of \
03/Jan/2012 03:26          24 GCODE.GCO     GCODE.GCO
17/May/2012 17:12           2 TEST.BAS      TEST.BAS
>>
```

In the above example 2 files are saved on the SD card, **GCODE**.GCO and **TEST**.BAS. Both are text files though the **TEST**.BAS is a TrioBASIC program. The file extensions of both files can clearly be seen.

Files on the **SD** card can be deleted using the **FILE** command.

FIFO

A First In First Out buffer is used to store text from a file. A **FIFO** buffer can be thought of as a pipe which holds text characters. Each character is loaded one at a time into the buffer and read out in the same order that they were loaded in. The example below shows the word '**HELLO**' being loaded through a **FIFO**.



Unlike a file which can vary in size a **FIFO** has a fixed length which can be declared when creating the **FIFO** using the **OPEN** command. If the **FIFO** is created automatically then it will have the length 256bytes. The length of the text that is loaded into the buffer may be longer or shorter. If the text is shorter then it is possible to load multiple files into the one **FIFO**. If the text is longer, the loading into the **FIFO** is automatically suspended until there is space available.

As the **FIFO** is a buffer it is possible to read from it and write to it at the same time. This means that a text file being loaded into the **FIFO** from a PC can be read out to the controller as soon as a character is available.

If the transfer time is significant compared to the machine cycle time then using a **FIFO** can help reduce overall cycle time compared to a fixed file.

Once a character has been read from the buffer it cannot be read a second time. If you want to read the file multiple times then it is better to save the file in the SD card or on the controller.

As FIFOs are stored in the controller memory they can be viewed using the DIR command. As with the text files they cannot be run like a TrioBASIC program and so have 'None' for the run type. If the program memory directory is read using the DIR command or through Motion Perfect it will display something like the following:

```
>>DIR
EPROM selected for power up
Memory available: 6805744
Selected program: TRANSFER_FILE
Directory is UNLOCKED
Program       Source      Code        Run Type    Code Type
--------------- ------- ------- -------- ---------
MC_CONFIG     38          34          Power Up    MC_CONFIG
TEST          452         552         Manual      Normal
NOTES         82          0           None        Text
READ_GCODE4   30242       14151       Manual      Normal
TRANSFER_FILE 0000        0           None        FIFO
OK>>
```

In the above example you can see a **FIFO** called **TRANSFER_FILE** has been initialised with size 10000. The code type is '**FIFO**' and Run Type is 'None' as you cannot compile or run this file.

If you need to delete the **FIFO** you can use the **DEL** command on the command line.

Loading files onto the controller

As text files are not always part of the project it is required to be able to load them to the controller at run time. This means loading a file while the project on the controller is running. There are multiple options available to load a file onto the controller, each method has advantages and disadvantages. The following section details these so that it is possible to pick the best method for an application.

DIRECTLY LOADING TO THE SD CARD

As the SD card is removable it is possible to remove it from the controller and plug it into a PC. The files can then be loaded to the SD card and it can be inserted back into the controller.

Do not remove the SD card from the PC or controller while files are being read or written to as this may corrupt the file or whole card.

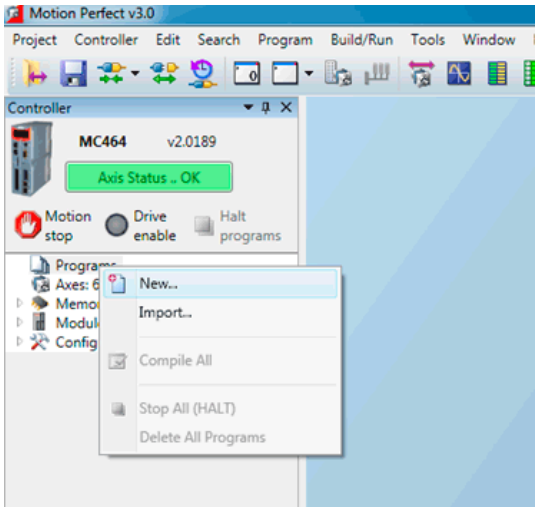
Although very simple this method is often not used as the controller is often inside a control cabinet which may be locked or inaccessible.

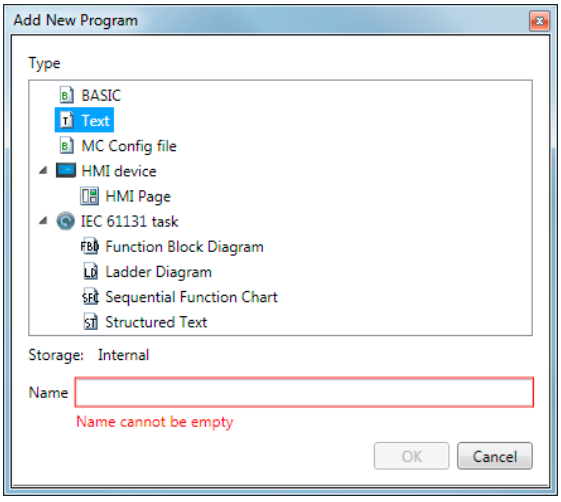
Please remember to format the SD cards as FAT32 so they can be used in the controller.

LOADING THROUGH MOTION PERFECT

The Motion Perfect editor has the capability of creating and editing text files. These are created as 'project text files' and are synchronised to the project.

The text file can be created using the Program -> New and selecting a text file. Once created they can be edited using the Motion Perfect editor.



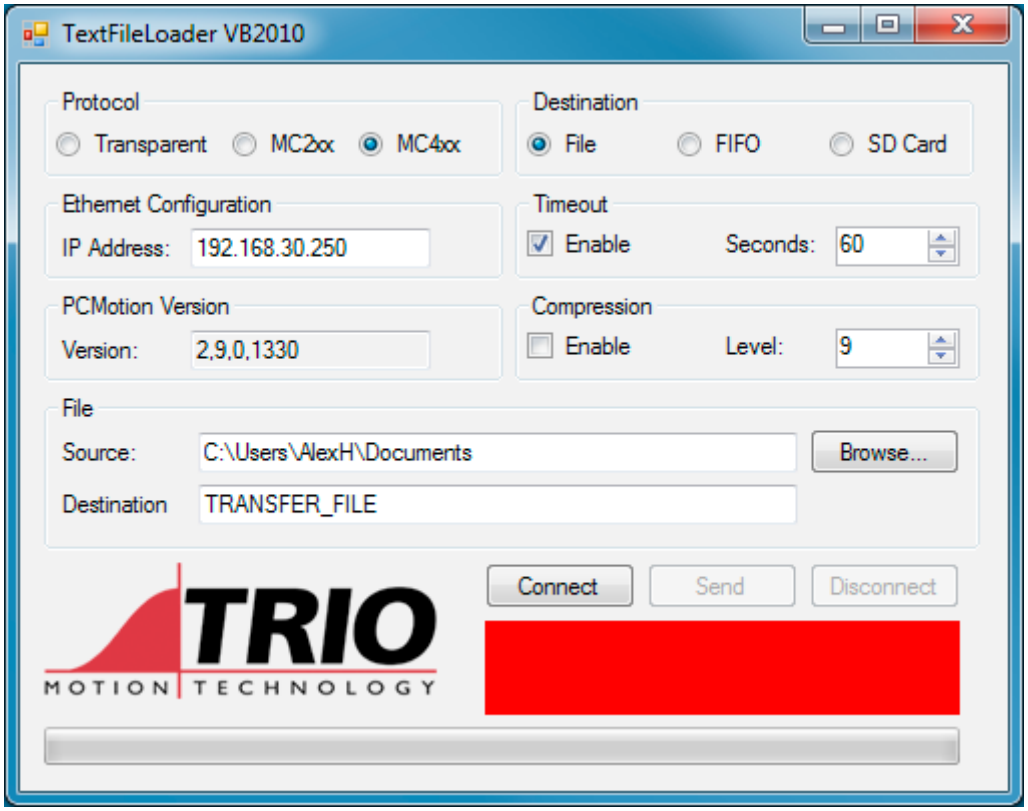


As project files can be created, edited or deleted by *Motion* Perfect or by a TrioBASIC program a lock is enabled when the project text file is opened into the *Motion* Perfect editor or when opened by a TrioBASIC program.

HIGH SPEED FILE TRANSFER

The high speed transfer requires an application using PCMotion ActiveX to send the file to the controller over an Ethernet connection. The PC application works in hand with a matching program on the controller to transfer the file as quickly as possible. Settings are available to send the file to a file in program memory, a **FIFO** or to the SD card. To further increase transfer speed the application can compress the file and extract it again as it is saved on the controller.

An example application is available on www.triomotion.com which can be downloaded as a standalone application or as source code so to provide a basis of a custom interface. The example is written in VB.net so easily re-written in other languages. Both require version 2.9.0.0 or newer of PCMotion to be installed. The example application allows manual setting of all of the parameters for the high speed file transfer so you can easily learn the functionality without having to write any code.



PCMOTION TEXTFILELOADER

The high speed file transfer is controlled by one command in PCMotion: TextFileLoader. This single command accepts parameters to configure the destination, protocol, timeout and compression. Before using the TextFileLoader command you must ensure that a PCMotion connection (Synchronous or Asynchronous) is open to the controller. When the TextFileLoader command is executed it will automatically start the built in program on the controller (**TEXT_FILE_LOADER_PROGRAM**). No configuration is required on the controller end.

The **TEXT_FILE_LOADER_PROGRAM** will run on a user process on the controller. There must be one available for it to run on, by default it will run on the highest available numbered process. If you wish to change the default process which the **TEXT_FILE_LOADER_PROGRAM** the parameter **TEXT_FILE_LOADER_PROC** can be set in the controller.

Once the **TEXT_FILE_LOADER_PROGRAM** has been started then it will be locked in the given protocol that caused it to auto-start. If the protocol must be changed then this program must be stopped by performing one of the following: **EX**, cycle power, or **STOP** "**TEXT_FILE_LOADER_PROGRAM**".

There is a **TEXT_FILE_LOADER** TrioBASIC command which can be used to read the transfer status as well as configure some default options for transparent mode.

The following section discusses the parameters in the PCMotion TextFileLoader command which are accessible through the VB application interface.

TEXTFILELOADER – PROTOCOL

Three protocols are available for the TextFileLoader.

The transparent protocol allows for a simple file transfer to the controller over Ethernet port 3241. By default it will transfer to a file in the controller memory called **TRANSFER_FILE**. Using the **TEXT_FILE_LOADER** command you can change the default destination for the transparent protocol. When using the transparent protocol any program can send text to the controller so it can be used without PCMotion.

By default the transparent protocol is disabled; set **IP_PROTOCOL_CONFIG** to the transparent protocol.

MC2xx protocol is only for connecting to the MC2 range of controllers. It is not recommended for new projects.

MC4xx protocol is for connecting to MC4 range of controllers over Ethernet port 10001. The protocol allows for any destination memory, allows the file to be named by the user and the file can be compressed so reducing transfer time.

TEXTFILELOADER – DESTINATION MEMORY

The text file loader is capable of loading a file to the SD card, file in internal memory or into a **FIFO**.

TEXTFILELOADER – TIMEOUT

The timeout parameter sets a timeout for the data transfer. With no timeout the application will wait for the transfer to complete.

TEXTFILELOADER – COMPRESSION

The compression parameter tells PCMotion to compress the file before sending it to the controller. Different levels of compression are available 0-9 with 0 no compression and 9 the most compressed. It takes longer to compress to a higher level, but the file size will be smaller. The smaller the file is the quicker it will transfer from the PC to the controller over Ethernet.

Higher compression levels are not always an advantage as the total transfer time consists of the compression time, data transfer and the uncompress time. This means if the file is smaller the compression times may be more significant than the data transfer so it may be faster overall to use a lower compression level. It is recommended to experiment with different compression levels to find a value that suits the application.

TEXTFILELOADER – DESTINATION

It is possible to select the destination file name. Remember that the file extension must be used if sending the file to the SD card.

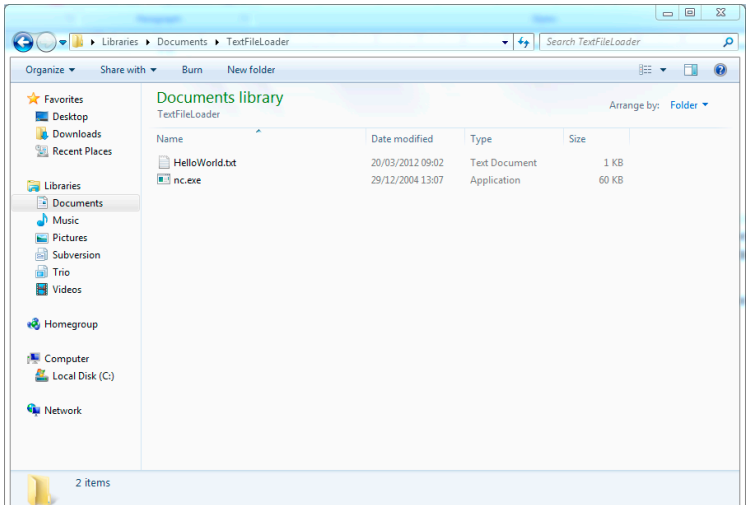
TRANSPARENT PROTOCOL WITH THIRD

It is possible to transfer the file to the controller using the transparent protocol. Various off the shelf third party applications are available. The example below uses Netcast to download a file over the transparent protocol to the **TRANSFER_FILE** on the controller.

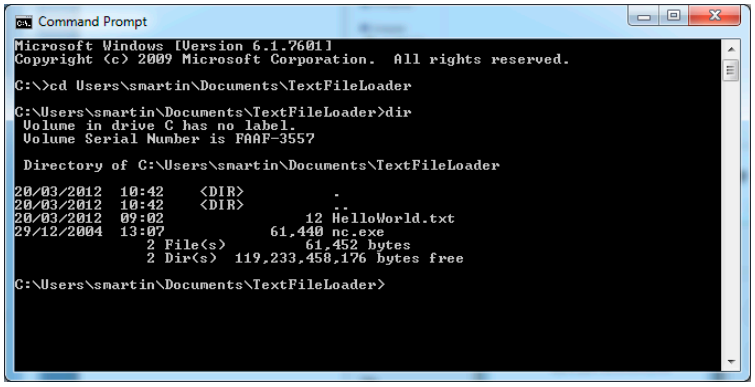
By default the transparent protocol is disabled; set **IP_PROTOCOL_CONFIG** to the transparent protocol.

Netcat is a simple networking utility which reads and writes data across network connections using the TCP/IP protocol. It allows you to read and write data over a network socket. At the time of writing this document a version for Windows can be downloaded from http://joncraton.org/blog/46/netcat-for-windows.

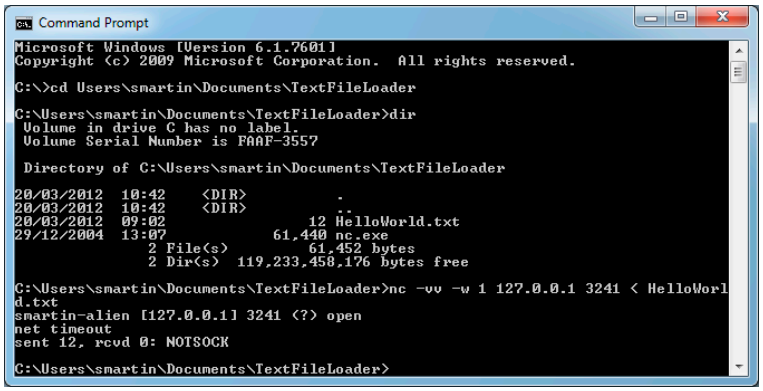
1. Download netcat for Windows and copy nc.exe to the same directory the HelloWorld.txt file



2. Open a command line window and navigate to the folder.



3. Send the file.



STREAMING USING A COMMUNICATIONS CHANNEL

Internally the text file loader streams the file to the new location. It is possible in TrioBASIC to do this manually using the **OPEN** command to open the file then **PRINT** to store the characters.It is simpler to use the TextFileLoader in PCMotion, but if that is not available then this is a manual alternative.

This method can be used when sending the file down over the serial port.

Example

The **OPEN** command is used to create a file, this can be in any of the locations discussed above. In this example it is a file on the SD card. Then while the file is being downloaded the controller reads in the characters and stores them in the file. Once the transfer is complete then the file is closed and ready for reading.

```
input_channel = 5
file_channel = 40
OPEN #file_channel AS "SD:File1" FOR OUTPUT
WHILE KEY#input_channel
  GET#input_channel,char
  PRINT#file_channel, CHR(char);
  PRINT#
WEND
CLOSE #fifo_channel
```

Reading a file in TrioBASIC

To read a file in TrioBASIC you must first open it as a stream. This is the same method for any storage location and one simple TrioBASIC command **OPEN**. The open command accesses the file or **FIFO** as a stream, this means it behaves like a communication port and you can use GET as normal to read out the character.

To aid the reading of strings TrioBASIC supports string variables you can use these to simplify the program that is reading the file. A good example of this is the G-Code program that reads in a file a line at a time then parses it looking for particular commands. This program is available from www.triomotion.com.

Example

The following program will configure a **FIFO** for use by the transparent protocol on the text file loader. Then when the transfer has started the program will read in the characters one at a time building up a line. When the line is complete it prints it to terminal 5.

```
  DIM line AS STRING(100)
  TEXT_FILE_LOADER(2,1,1)
  value = -1
  fifo_size = 10000
  fifo_channel=40
  OPEN #fifo_channel AS "TRANSFER_FILE" FOR FIFO_WRITE(fifo_
size)
  CLOSE #fifo_channel

  'Open FIFO to read
  OPEN #fifo_channel AS "TRANSFER_FILE" FOR FIFO_READ

    'Wait for a file transfer to start
    WAIT UNTIL TEXT_FILE_LOADER(1,0)

    'Process this file
    WHILE KEY#fifo_channel
      GET#fifo_channel, k
      IF k <> 13 THEN
        line = line + CHR(k)
      ELSE
        PRINT#5, line
      ENDIF
    WEND
  CLOSE #fifo_channel
```