# TOE1G-IP FTP Server Demo Reference Design Manual

Rev1.1 2-Sep-16

## 1 Introduction

File Transfer Protocol (FTP) is the protocol designed for file sharing over internet. By using TCP/IP for lower layer, FTP can transfer data reliably and efficiently.

Reference documents

1. File Transfer Protocol: http://tools.ietf.org/html/rfc959
2. File Transfer Protocol: http://www.tcpipguide.com/free/t_FileTransferProtocolFTP.htm
3. FTP Sequence: www.eventhelix.com/realtimemantra/networking/FTP.pdf
4. List of FTP commands: http://en.wikipedia.org/wiki/List_of_FTP_commands
5. List of FTP server return codes:http://en.wikipedia.org/wiki/List_of_FTP_server_return_codes

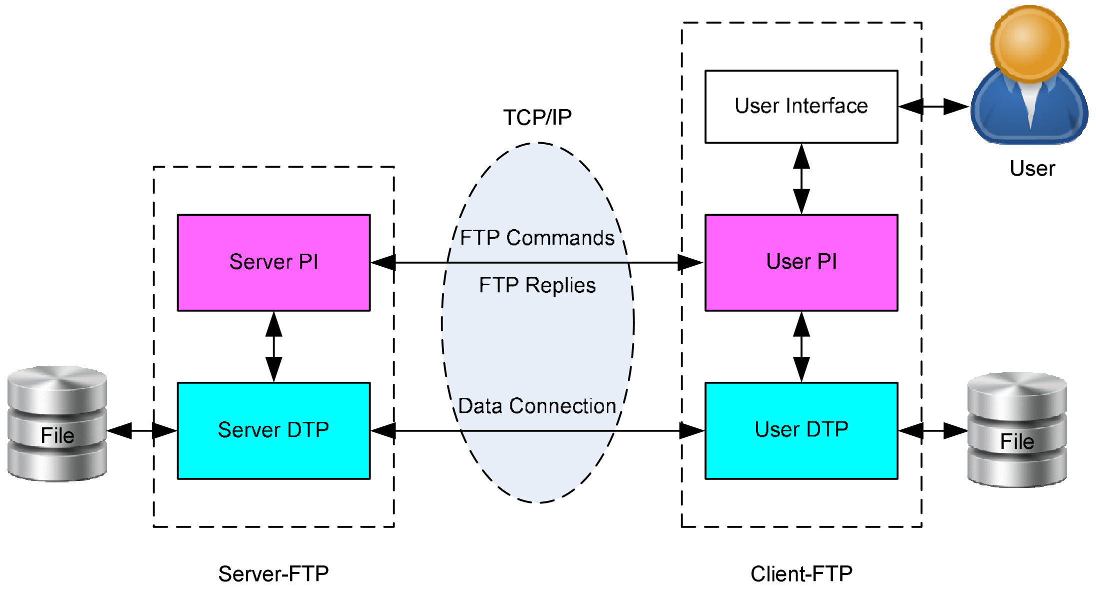


Figure 1 FTP Model

As shown in Figure 1, there are two hosts with its own storage for storing data in file format. One is server to store shared data in the network and another is client for general user access. To transfer file by using FTP protocol, two-port connections are required, i.e. control port to transfer FTP commands from client to server and FTP replies from server to client, and data port to transfer data between the hosts.

Server Protocol Interpreter (Server-PI) and User Protocol Interpreter (User-PI) are responsible for managing the control connection on the server and client respectively. Server-PI listens the well-known TCP port for FTP (Port 21) to check connection requests from User-PI. After connection is established, Server-PI can receive FTP commands from User-PI, send back FTP replies, and manage the server data transfer process. For client side, User-PI is responsible to process commands from the user interface, forward command to Server-PI, receive back replies, and manage the user data transfer process.

Server Data Transfer Process (Server-DTP) and User Data Transfer Process (User-DTP) are used to send or receive data. The data connection can be established from Server-DTP (active mode) or from the User-DTP (passive mode). Generally, passive mode FTP is implemented to resolve the firewall problem on FTP client. The data connection is established to transfer the file between the server and the client, and terminated when the file transfer is complete. Both Server-DTP and User-DTP interacts with the local file system to read and write files.

User Interface provides the human user to issue commands and see responses from the FTP software.

## 1.1 FTP Connection Establishment and User Authentication

The first process after user initiates the connection is user authentication to allow only authorized user to access FTP server. The details of connection establishment and login are described as follows.

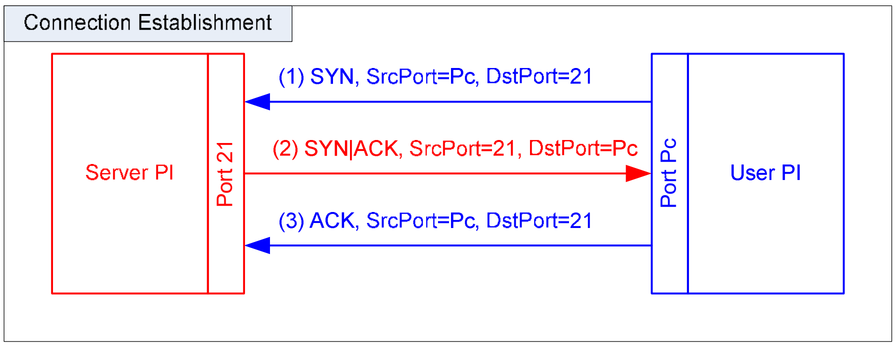

Connection Establishment
Server PI
Port 21
(1) SYN, SrcPort=Pc, DstPort=21
(2) SYN|ACK, SrcPort=21, DstPort=Pc
(3) ACK, SrcPort=Pc, DstPort=21
Port Pc
User PI

Figure 2 Connection Establishment on Control Port

Based on TCP protocol, connection establishment is three-way handshake, as shown in Figure 2. FTP client sends TCP packet with setting SYN flag to Port 21 at Server. New request is detected by FTP Server, and TCP packet with SYN and ACK flag will be returned from Server to accept the new connection. Client sends acknowledgement by ACK flag to complete this process. After that, Server and Client can communicate together by using FTP command and replies.

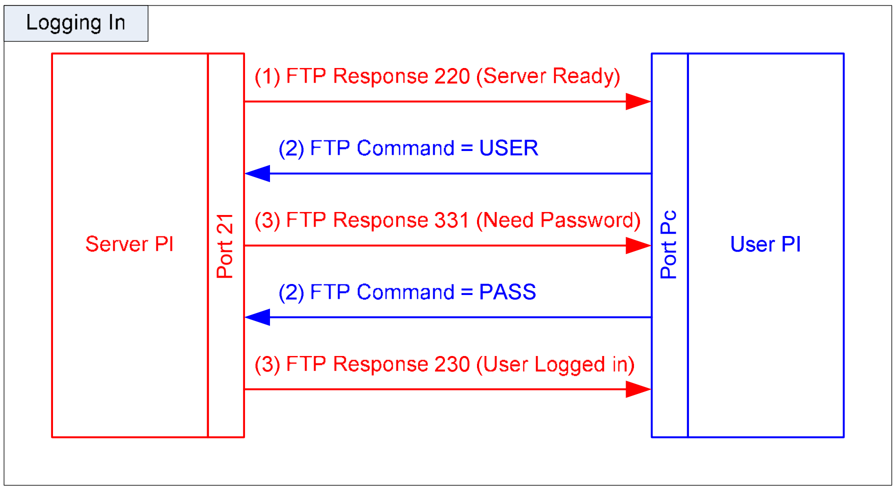

Logging In
Server PI
Port 21
(1) FTP Response 220 (Server Ready)
(2) FTP Command = USER
(3) FTP Response 331 (Need Password)
(2) FTP Command = PASS
(3) FTP Response 230 (User Logged in)
Port Pc
User PI

Figure 3 User Authentication

As shown in Figure 3, after control connection is established, two FTP commands are sent for input username (USER command) and password (PASS command) in a sequence. Finally, Server returns response 230 to allow the new session opened.

## 1.2 FTP Data Connection Management by Passive mode

The reference design implements data connection establishment by Client (Passive mode) which is general usage. So, this topic describes only the sequence of Passive mode.

Some FTP commands such as LIST (List subdirectories or files), STOR (Store files), and RETR (Retrieve files) require data transfer through data connection. Before transferring data, PASV command is used to setup data connection firstly to specify port number for Server-DTP.

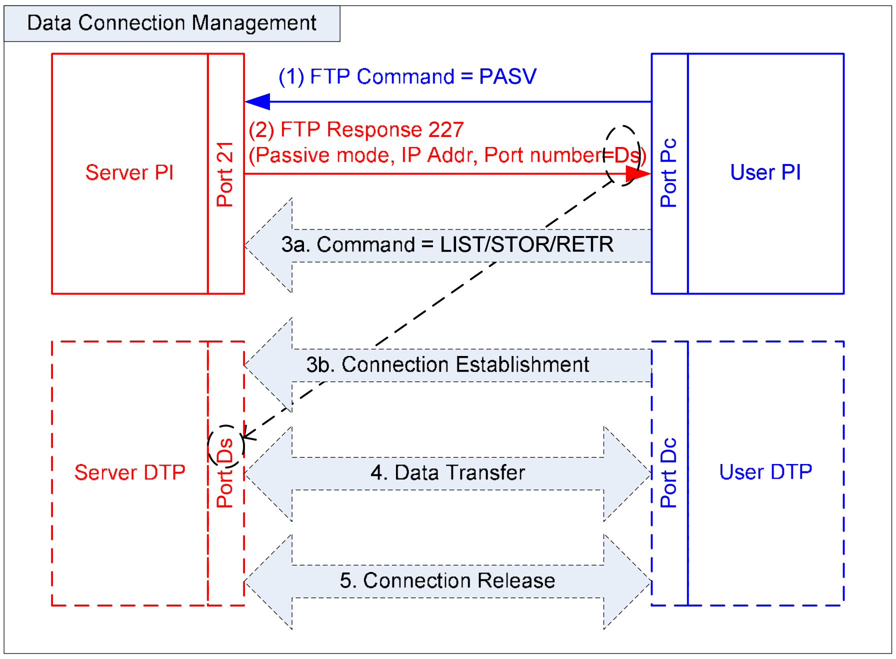


Figure 4 Data Connection Management in Passive Mode

After receiving PASV command, Server returns response 227 to inform IP address and data port number at Server side to Client. In this example, the data port at Server side is assumed to be Ds. Then, Server-PI would instruct Server-DTP to listen on Port Ds. User-PI sends FTP command which requires data transfer to Server-PI while User-DTP establishes data connection to transfer the data. The sequence of FTP command at step 3a) and data connection establishment at step 3b) can be swapped, depending on FTP client behavior. Data connection will be released after complete data transferring.

## 1.3 LIST Command

LIST command is used to transfer a list of files in the specified directory through data connection, so PASV command must be sent firstly to initialize data connection. User-PI sends LIST command to Server while User-DTP establishes data connection. The sequence of Step2a) and step2b) is possibly swapped.

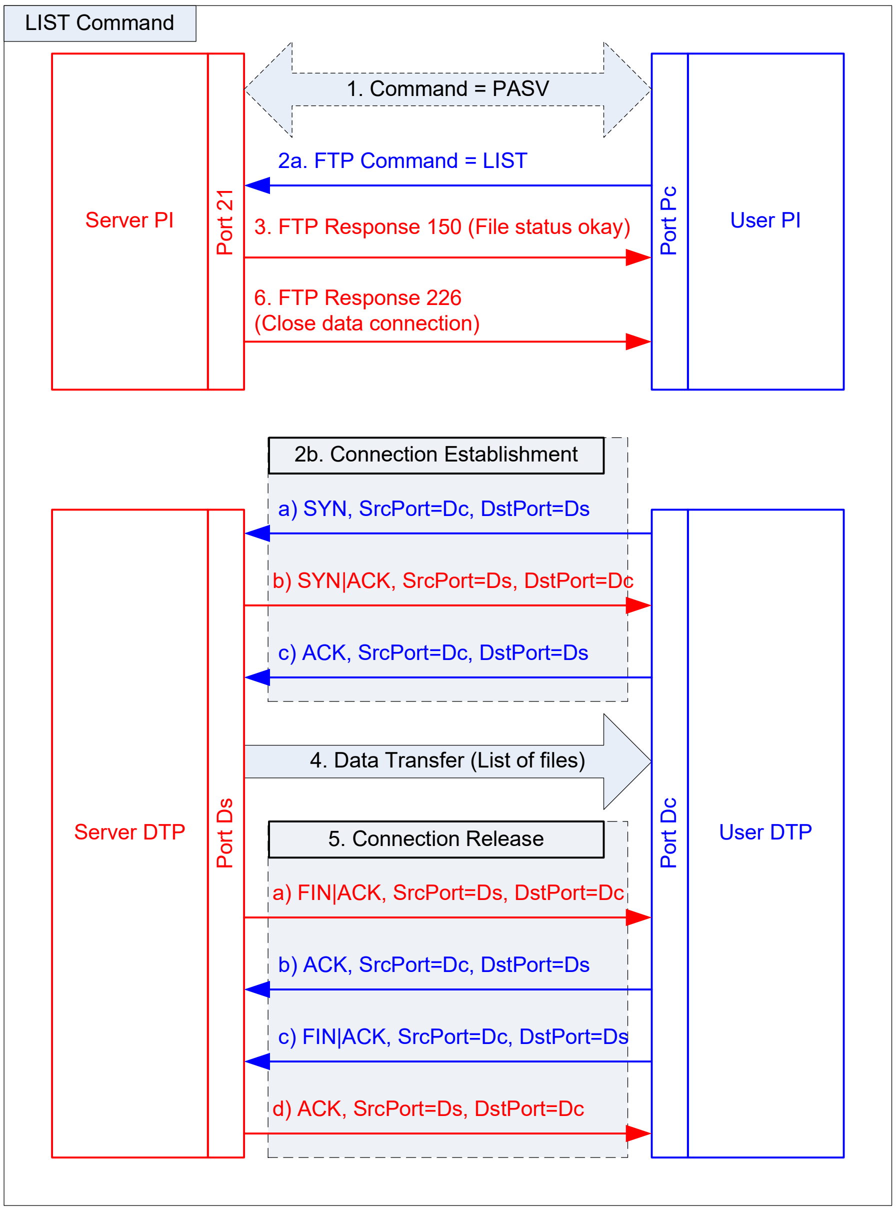

LIST Command
Server PI
Port 21
1. Command = PASV
2a. FTP Command = LIST
3. FTP Response 150 (File status okay)
6. FTP Response 226
(Close data connection)
Port Pc
User PI
2b. Connection Establishment
a) SYN, SrcPort=Dc, DstPort=Ds
b) SYN|ACK, SrcPort=Ds, DstPort=Dc
c) ACK, SrcPort=Dc, DstPort=Ds
4. Data Transfer (List of files)
5. Connection Release
a) FIN|ACK, SrcPort=Ds, DstPort=Dc
b) ACK, SrcPort=Dc, DstPort=Ds
c) FIN|ACK, SrcPort=Dc, DstPort=Ds
d) ACK, SrcPort=Ds, DstPort=Dc
Server DTP
Port Ds
Port Dc
User DTP

Figure 5 LIST Command Sequence

For Server side, Server-PI returns response 150 for LIST command, and Server-DTP returns the list of files to Client. When complete data transfer, data connection will be released by Server-DTP. Response 226 is sent out by Server-PI to complete LIST command operation.

## 1.4 STOR Command

STOR command is used to store file from Client to Server. As shown in Figure 6, the sequence of STOR command for Server-PI is similar to LIST command, but data transfer direction in data connection is swapped. Server gets file name by decoding from STOR command, and gets file size by calculating the total number of data transfer size during data connection available. Since file data is transferred from Client to Server, the data connection will be released by User-DTP at the end of transfer.

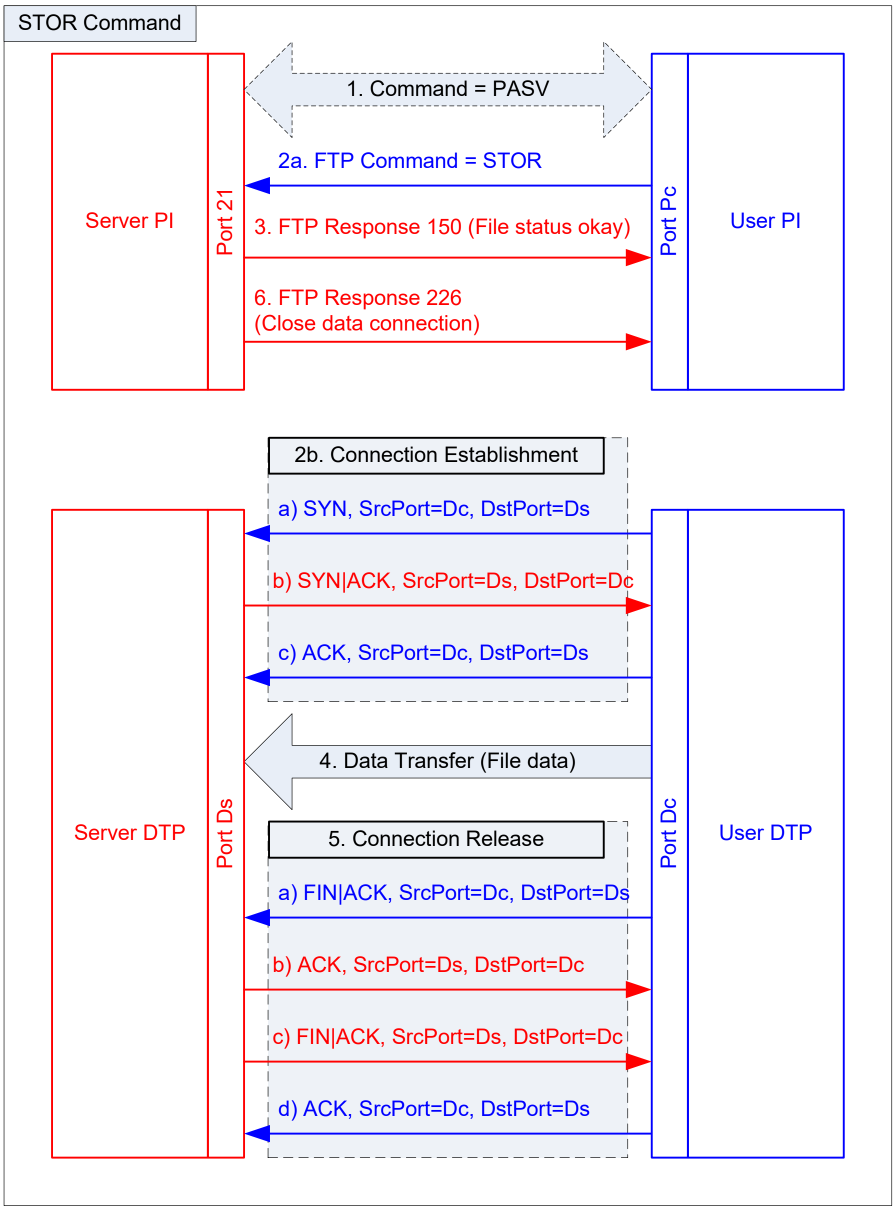

STOR Command
Server PI
Port 21
1. Command = PASV
2a. FTP Command = STOR
3. FTP Response 150 (File status okay)
6. FTP Response 226 (Close data connection)
Port Pc
User PI
2b. Connection Establishment
a) SYN, SrcPort=Dc, DstPort=Ds
b) SYN|ACK, SrcPort=Ds, DstPort=Dc
c) ACK, SrcPort=Dc, DstPort=Ds
Server DTP
Port Ds
4. Data Transfer (File data)
5. Connection Release
a) FIN|ACK, SrcPort=Dc, DstPort=Ds
b) ACK, SrcPort=Ds, DstPort=Dc
c) FIN|ACK, SrcPort=Ds, DstPort=Dc
d) ACK, SrcPort=Dc, DstPort=Ds
Port Dc
User DTP

Figure 6 STOR Command Sequence

## 1.5 RETR Command

RETR command is used to retrieve file from Server. The sequence is similar to LIST command, but returned data from Server is file data, not list of files. Similar to STOR command, file name can be decoded from RETR command parameter. The data connection will be released by Server-DTP after data complete transfer.

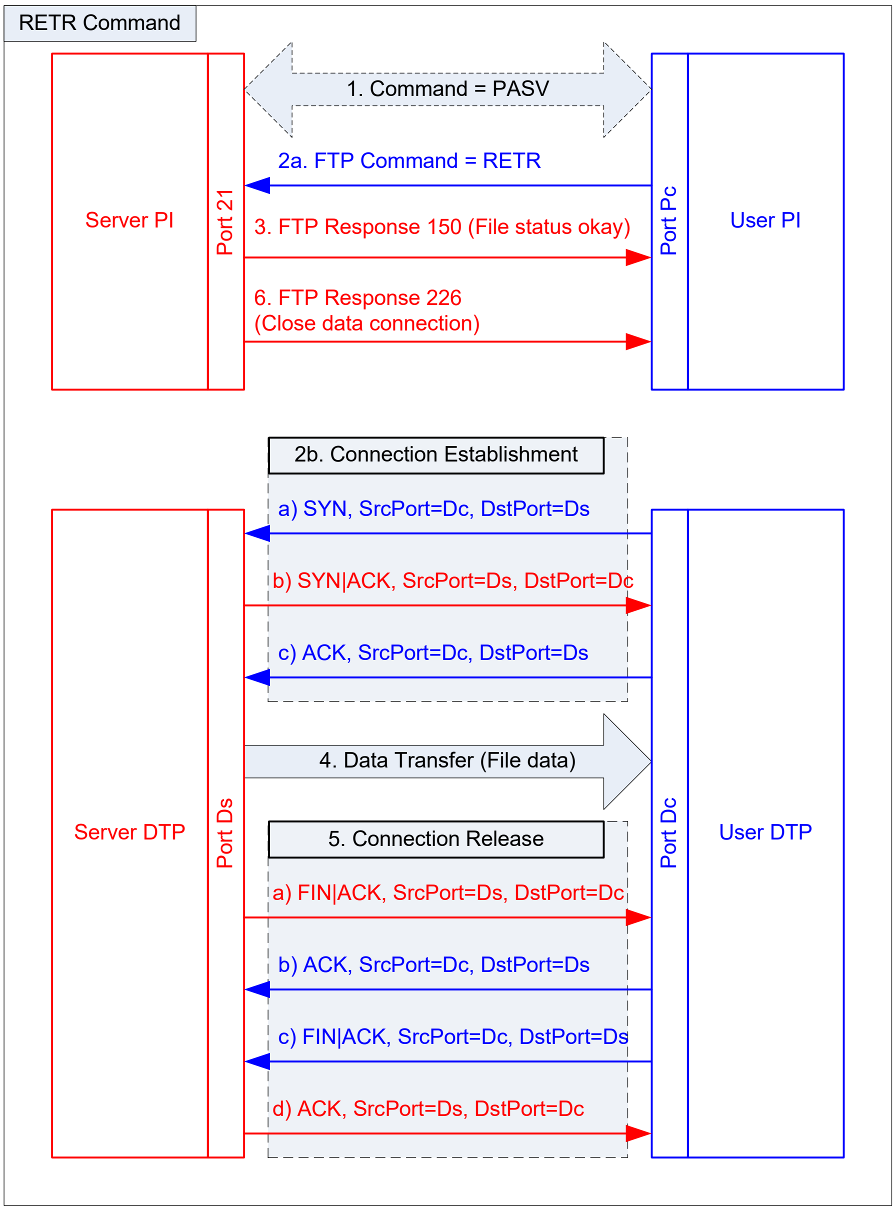

RETR Command
Server PI
Port 21
1. Command = PASV
2a. FTP Command = RETR
3. FTP Response 150 (File status okay)
6. FTP Response 226
(Close data connection)
Port Pc
User PI
2b. Connection Establishment
a) SYN, SrcPort=Dc, DstPort=Ds
b) SYN|ACK, SrcPort=Ds, DstPort=Dc
c) ACK, SrcPort=Dc, DstPort=Ds
4. Data Transfer (File data)
5. Connection Release
a) FIN|ACK, SrcPort=Ds, DstPort=Dc
b) ACK, SrcPort=Dc, DstPort=Ds
c) FIN|ACK, SrcPort=Dc, DstPort=Ds
d) ACK, SrcPort=Ds, DstPort=Dc
Server DTP
Port Ds
Port Dc
User DTP

Figure 7 RETR Command Sequence

## 2 Hardware Structure

As shown in Figure 8, the hardware of FTP Server demo is similar to the hardware of two-port demo design. So, it is recommended for the user to read two-port reference manual firstly ("dg_toe1gip_2port_refdesign_xilinx_en"). Only CPU firmware is developed for FTP server demo. This document describes only additional information for FTP Server demo.

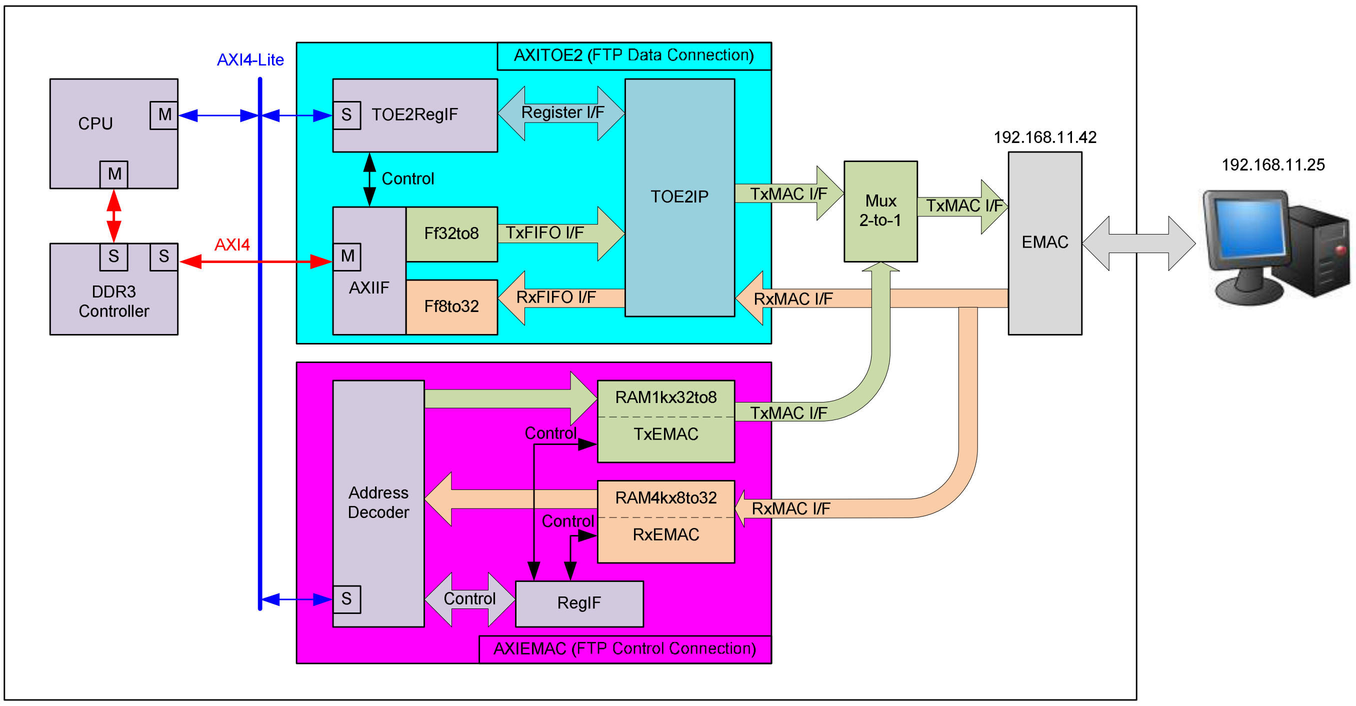


Figure 8 FTP Server Demo Hardware Block Diagram

Comparing to two-port design, Fast connection is used to be FTP Data connection while slow connection is used to be FTP Control connection. FTP Control connection is used to receive FTP command from the client and send FTP replies back to the client. The control port is fixed to port#21. So, CPU firmware must set different value for header filtering function inside RxEMAC module to support FTP control connection instead of ICMP protocol.

To support FTP server demo, the filter header of RxEMAC is programmed to be following value.

- Destination MAC address = broadcast ID or set value from CPU
- Destination IP address of = set value from CPU
- Protocol = TCP (RXPATT23_REG=0x06, RXPATTEN_REG[0]='1')
- Destination port number = 21 (RXPATT36_REG=0x0015, RXPATTEN_REG[2:1]="11")

From above setting, only FTP command packet will be stored to RxRAM.

# 3 Software

The details of FTP server sequence for both control and data connection has been described in the introduction part. In the demo, the main controller is implemented by CPU firmware. CPU can control each connection through register access. Ethernet data of control port are constructed and decoded through TxRAM and RxRAM inside AXIEMAC. While TCP data of data port are transferred through DDR3. For FTP server application, next topic describes about DDR3 memory map, implemented FTP commands and replies, and software design sequence.

## 3.1 DDR3 Memory Map

The data in FTP server requires to store in file system format. To support simple file system, DDR3 memory is used to store not only the data file of FTP, but also store file information as shown in Figure 9. Simple file system does not support sub-directory and support at most 16 files.

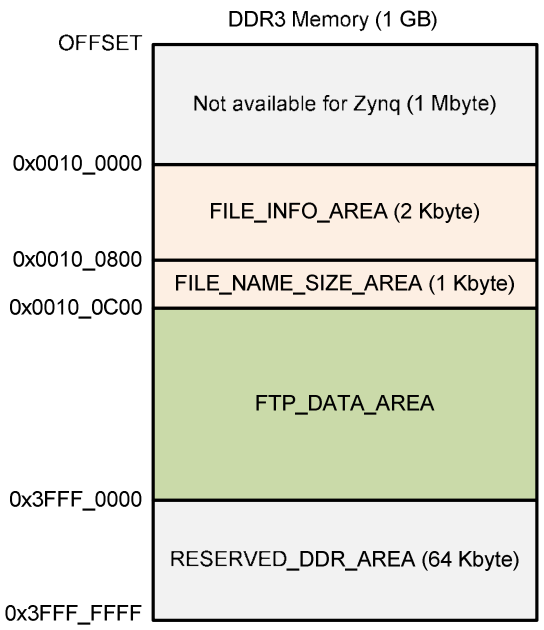


Figure 9 DDR3 Memory Map

DDR3 can be split into four areas, i.e.

1) FILE_INFO_AREA: Data in this area is prepared to return "LIST" command. The file information is file name, file size, permission, and modified date. 2 Kbyte is reserved for this area to support 16 files, but actual usage area is about 1.2 Kbyte.
2) FILE_NAME_SIZE_AREA: Data in this area is prepared to return "MLSD" command. The returned values for MLSD command are file size and file name. Since returned data format of MLSD and LIST command are different, the information must be prepared in different DDR3 area. 1 Kbyte is reserved for this area.
3) FTP_DATA_AREA: Emulate to be the storage of FTP server. So, total file size is about 1 GB – 1 Mbyte - 67 Kbyte.
4) RESERVED_DDR_AREA: This area is used to flush the data in the buffer within hardware (TOE1G-IP and FIFO) when data from user to store in FTP_DATA_AREA is overflow. Maximum buffer size of TOE1G-IP is 64 Kbyte, so 64 Kbyte is reserved for this area.

## 3.2 FTP Command

There are many FTP commands in the standard, and some commands are related to file system operation. To support minimum FTP command, the demo implements based on two FTP client softwares, i.e. FileZilla, and built-in FTP client of Windows OS. Lists of implemented command are follows.

| FTP Command | Description | Implemented FTP Response | Data Conn |
|---|---|---|---|
| USER | Authentication username | 331-Username OK, need password | No |
| PASS | Authentication password | 230-User logged in | |
| FEAT | Get the feature list implemented by the server | 211-System status (SIZE, MLST, UTF8) | |
| PASV | Enter passive mode | 227-Enter Pass Mode | |
| TYPE | Sets the transfer mode | 200-Requested action completed | |
| NOOP | No operation | | |
| PWD | Print working directory | 257-“PATHNAME” created | |
| XPWD | Print current working directory | | |
| SIZE | Return the size of a file | 213-File status | |
| QUIT | Disconnect | 221-Service closing control connection | |
| LIST | Return information of a file or directory if specified | 150-File status OK 226-Close data connection | Yes |
| MLSD | List the contents of a directory | | |
| RETR | Retrieve a copy of the file | | |
| STOR | Accept data and store data as a file at the server site | | |

Table 1 Implemented FTP Command Description

As shown in Table 1, the 3rd column shows FTP response number which is implemented in the demo. The 4th column shows data connection requirement for each command. There are four commands, i.e. LIST, MLSD, RETR, and STOR which requires data transferring through data connection, so PASV command will be sent from the client firstly. While other FTP commands do not require data transferring and only FTP response is returned from the server to client.

For not implemented FTP command, two types of response are returned, i.e. 550 (permission denied) for DELE (delete file) command, and 202 (not implemented) for other FTP commands such as CWD, SYST, PORT, OPTS, and SITE.

Since DELE command is not implemented, user cannot delete the file from Server. File on server is stored in DDR3, so the file will not available after power-down system.

## 3.3 FTP Server

The sequence of FTP server firmware is follows.

1) Initialize TOE1G-IP and AXIEMAC parameters, i.e.
   - FTP Server MAC Address = 0x000102030405
   - FTP Server IP Address = 192.168.11.25
   - FTP Client IP Address = 192.168.11.42
   - FTP Server Port for Data connection = 50000
   - TOE1G-IP Timeout = 2 seconds
2) Release RST flag of TOE1G-IP to start ARP request/reply by TOE1G-IP between Server and Client. CPU waits until BSY flag of TOE1G-IP is de-asserted to '0'.
3) Initialize parameters in the firmware.
4) Polling RXADDR_REG to monitor that received data available in RXRAM.
5) Dump received packet to received temp buffer on firmware. Only 256-byte size is reserved by defining "TMPBUF_SIZE" parameter value.
6) Calculate IP and TCP checksum to check that packet is valid. Return error when checksum is not correct.
7) Compare Client port number in the packet to the port number in the active session.
   a) If the value is matched, load sequence number and acknowledge number of match active session to continue the next TCP packet transfer.
   b) If the value is not matched (new session), store client port number and initial value of sequence number and acknowledge number for TCP packet transfer.

   Note: The demo firmware implements to support up to 50 new sessions. The session cannot be reused after closing it.
8) There are four types of received packet for FTP control port, i.e.
   a) SYN flag. This is packet for open the new session of FTP control. To send acknowledgement, CPU creates TCP packet to TxRAM and transfers to EMAC. After session is established, CPU sends welcome message to Client.
   b) FIN flag. This is packet for close the current session of FTP control. Similar to SYN flag, CPU creates TCP packet to return acknowledgement. After that, connection will be released.
   c) ACK with data. For control port, the data from Client is only FTP command. As shown in the 4th column of Table 1, FTP command can be split into two types.
      - For FTP command without data connection, only FTP response will be created by CPU through TxRAM. The details of FTP response for each FTP command are shown in the 3rd column of Table 1.
      - For FTP command with data connection, firstly CPU waits until data connection is established by monitoring CONNON_REG. After that, response 150 is created by CPU to confirm that now data connection at server is ready. After complete data transfer and data connection is closed, CPU returns response 226 to confirm that now data connection at server is released. More details of the sequence for FTP command with data connection will be described in next topic.
   d) ACK without data. This is acknowledgement packet returned from the client.
9) Go back to step 4) to wait new FTP command from the client.

## 3.4 FTP Command with data connection sequence

### 3.4.1 LIST

1) Check data connection has already established by monitoring CONNON_REG.
2) Write FTP response 150 to TxRAM and send out to EMAC.
3) Return file information which is stored in FILE_INFO_AREA through data connection by setting parameters such as DDR3 address, transfer size, and start transfer flag to TOE1G-IP and AXITOE1G register.
4) Wait until TOE1G-IP completes transfer by monitoring bit 0 of TOE1G_CMD_REG bit.
5) Wait until AXITOE1G transfers data from DDR to EMAC completely by monitoring bit 0 of AXITOE1G_CTRL.
6) Set TOE1G-IP register to send active close for data connection.
7) Wait until TOE1G-IP completes connection release by monitoring bit 0 of TOE1G_CMD_REG.
8) Write FTP response 226 to TxRAM and send out to EMAC.

### 3.4.2 MLSD

This command is implemented by using same flow as LIST command. The different point is only the returned information from DDR3 which is assigned in the different area. Data returned from MLSD is stored in FILE_NAME_SIZE_AREA.

### 3.4.3 STOR

1) Decode filename from FTP command.
2) Write FTP response 150 to TxRAM and send out to EMAC.
3) Wait until data is available in received FIFO of TOE1G-IP, and then set control signal of AXITOE1G to dump data from TOE1G-IP to DDR3. DDR3 start address is the start address of available area in FTP_DATA_AREA.
4) After end of each data transfer, next DDR3 start address and total file size will be re-calculated for next transfer.
5) Step 3) – 4) run in loop until data connection is closed by the client. The status can be monitored by CONNON_REG.
6) Wait until TOE1G-IP and AXITOE1G complete data transfer.
7) Update file information, i.e. name, size, permission, and modified date to FILE_INFO_AREA and FILE_NAME_SIZE_AREA. File size is equal to total data transfer which is calculated in step 4). Also, the next start address of available area is equal to the end address of current transfer but round up to align sector unit for easily management.
8) Write FTP response 226 to TxRAM and send out to EMAC.

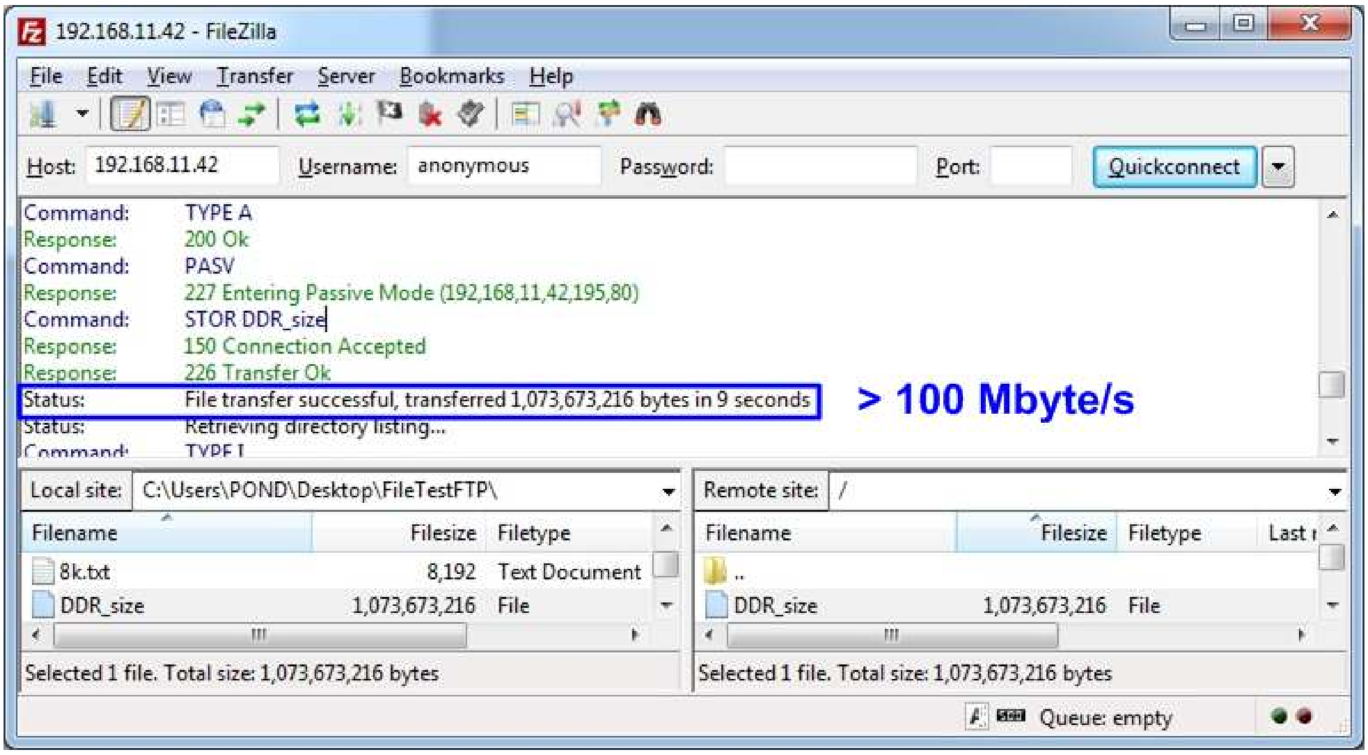
### 3.4.4 RETR

1) Similar to step1) – 2) of LIST command.
2) Decode filename from FTP command, and use to search the match name.
3) Load start DDR3 address of the match file and file size value. Then, set start address and transfer size to the register for data burst transfer.
4) Similar to step3) – 8) of LIST command, but start address of DDR3 and total transfer size of RETR command are set from step2) and 3).

## 4 Necessary consideration

FTP Server source in reference design does not include error recovery from illegal/unexpected behavior. To simplify software, the source code supports the minimum FTP command so that user can easily understand fundamental FTP Server software operation. Since minimum FTP command is different depending on FTP client software, the code is designed to support the two popular FTP client softwares, i.e. FileZilla and built-in FTP client of Windows OS. To use small resource, the design can support up to 16 files, and maximum filename is 30 characters. Also, delete file feature is not implemented to simplify file system implementation.

For control connection, the firmware does not include the process to recovery the data when lost packet is found for both transmit and received side. Also, the firmware does not include the function to decode TCP header length, so it can support only the packet which has TCP header length equal to 20 byte.


192.168.11.42 - FileZilla
File Edit View Transfer Server Bookmarks Help
Host: 192.168.11.42 Username: anonymous Password: Port: Quickconnect

```
Command:    TYPE A
Response:   200 Ok
Command:    PASV
Response:   227 Entering Passive Mode (192,168,11,42,195,80)
Command:    STOR DDR_size
Response:   150 Connection Accepted
Response:   226 Transfer Ok
Status:     File transfer successful, transferred 1,073,673,216 bytes in 9 seconds
Status:     Retrieving directory listing...
Command:    TYPE I
```

> 100 Mbyte/s
Local site: C:\Users\POND\Desktop\FileTestFTP\
Remote site: /

| Filename | Filesize | Filetype |
|---|---|---|
| 8k.txt | 8,192 | Text Document |
| DDR_size | 1,073,673,216 | File |

Selected 1 file. Total size: 1,073,673,216 bytes

| Filename | Filesize | Filetype |
|---|---|---|
| .. | | |
| DDR_size | 1,073,673,216 | File |

Selected 1 file. Total size: 1,073,673,216 bytes

Figure 10 Example Test Result by FileZilla

## 5 Revision History

| Revision | Date | Description |
|---|---|---|
| 1.0 | 9-Jan-15 | Initial version release |
| 1.1 | 2-Sep-16 | IP core product renamed from TOE2-IP to TOE1G-IP |