Application Note

AN_412

FT600_FT601 USB Bridge chips Integration

Version 1.1

Issue Date: 2017-12-08

This application note is for designers who want to integrate the FT600 and FT601 USB Bridge chips into new designs.

Future Technology Devices International Limited (FTDI)

Unit 1, 2 Seaward Place, Glasgow G41 1HH, United Kingdom
Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758
Web Site: http://ftdichip.com

Table of Contents

1 Introduction

The FT600 and FT601, collectively referred as FT60x, are designed to bridge USB packets from USB2.0 and USB3.0 hosts to a USB function through a FIFO interface. The bridge implements the FIFO slave interface which may be configured as a FT245 (single channel) or FT600 (multi-channel) bus. In typical designs, an FPGA is used to implement the FIFO master interface to communicate with the bridge. The FPGA and other devices (not shown) implement the remaining features of the target USB function(s). The bridge handles all USB related configuration, control and data transfer and simplifies customer designs that require an USB2.0/USB3.0 device port.

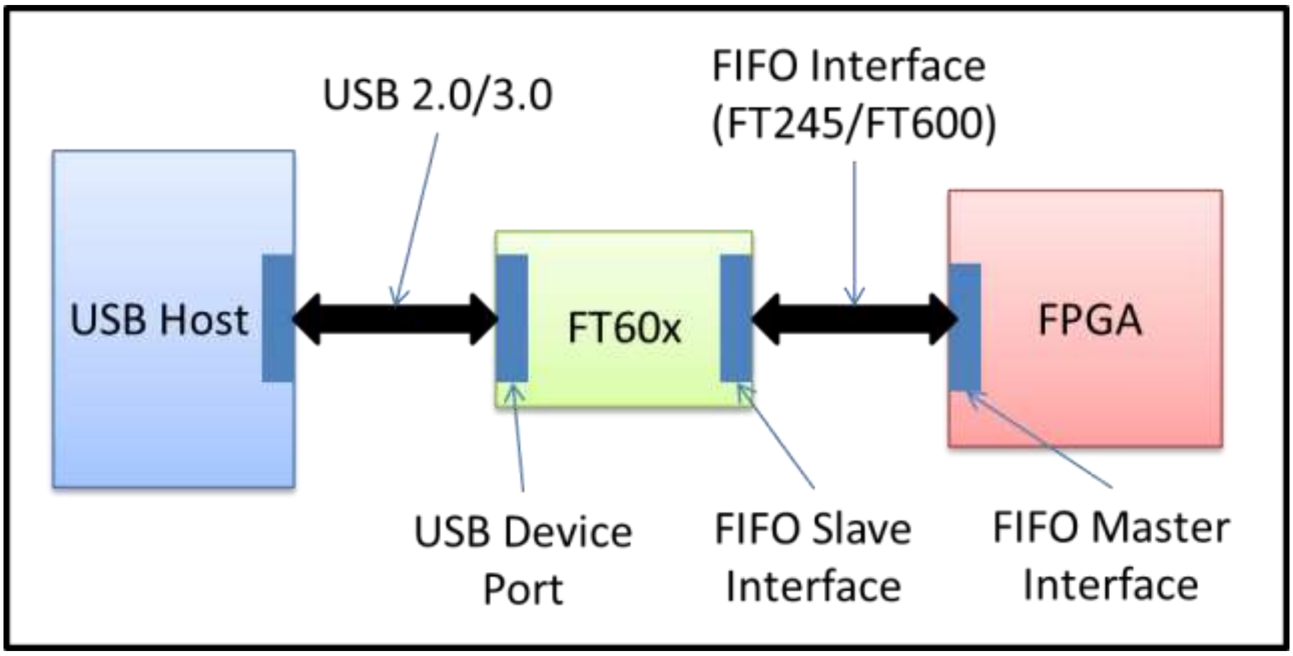

Figure 1 - FT60x System Block Diagram

2 Data Transfer Semantics

The FT600 and FT601 bridges may be viewed as USB-FIFO translators. i.e. any USB bulk packet from the USB host is placed into a FIFO for the FIFO master to read and any data from the FIFO master is in turn carried as USB bulk packets.

In order to achieve a high throughput on the bus, the FIFO master shall be designed to read or write in maximum sized packets except for the last packet in a transfer. This ensures high efficiency on the USB. The length of data to be exchanged between host and function is carried in a session message from the driver to the bridge.

In the IN direction, the FIFO master has to be designed to write in maximum sized packets except for the last packet which must be a short packet. This ensures that the bus bandwidth is used efficiently. When the FIFO master is designed to write in random sized bursts and when these bursts result in short packets, then the transfer requests will be ended prematurely. Figure 2 - Read Pipe Semantics – shows two behaviors controlled via the chip configuration.

The upper half of the figure shows the default behavior of the bridge in the read direction. The application has requested 3072 bytes; however, the FIFO interface master only wrote 1024 bytes and stopped. The bridge detects this as a session underrun and terminates the read session and returns the data collected to the host.

The lower half of the figure shows the behaviors of the bridge when it is configured to keep the session as long as the maximum sized packets are written into the FIFO. In this configuration, the session is not terminated when the master pauses between writing maximum sized packets. The session is ended when the full length of data is received by the bridge (shown) or if a short packet is written (not shown).

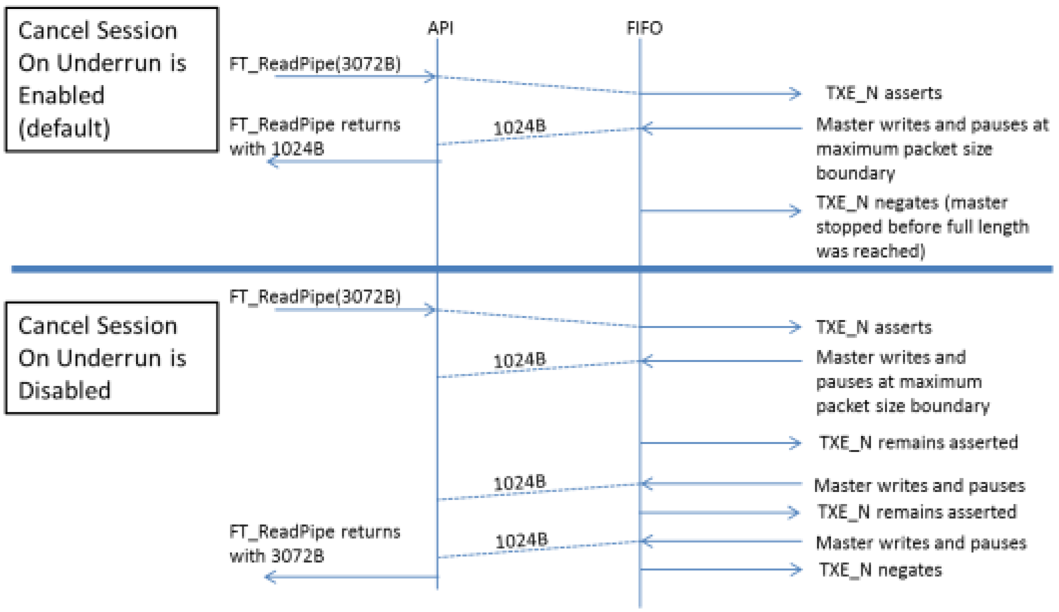
Cancel Session On Underrun is Enabled (default)
API
FIFO
FT_ReadPipe(3072B)
TXE_N asserts
FT_ReadPipe returns with 1024B
1024B
Master writes and pauses at maximum packet size boundary
TXE_N negates (master stopped before full length was reached)
Cancel Session On Underrun is Disabled
FT_ReadPipe(3072B)
TXE_N asserts
1024B
Master writes and pauses at maximum packet size boundary
TXE_N remains asserted
1024B
Master writes and pauses
TXE_N remains asserted
FT_ReadPipe returns with 3072B
1024B
Master writes and pauses
TXE_N negates

Figure 2 - Read Pipe Semantics

In the OUT direction, the FIFO master has to be designed to read out all the data from the host. If data is not read out promptly, then the OUT path may be bottlenecked and result in a write timeout. There is no mechanism for the FIFO interface master to "drop" the data in the write pipe.

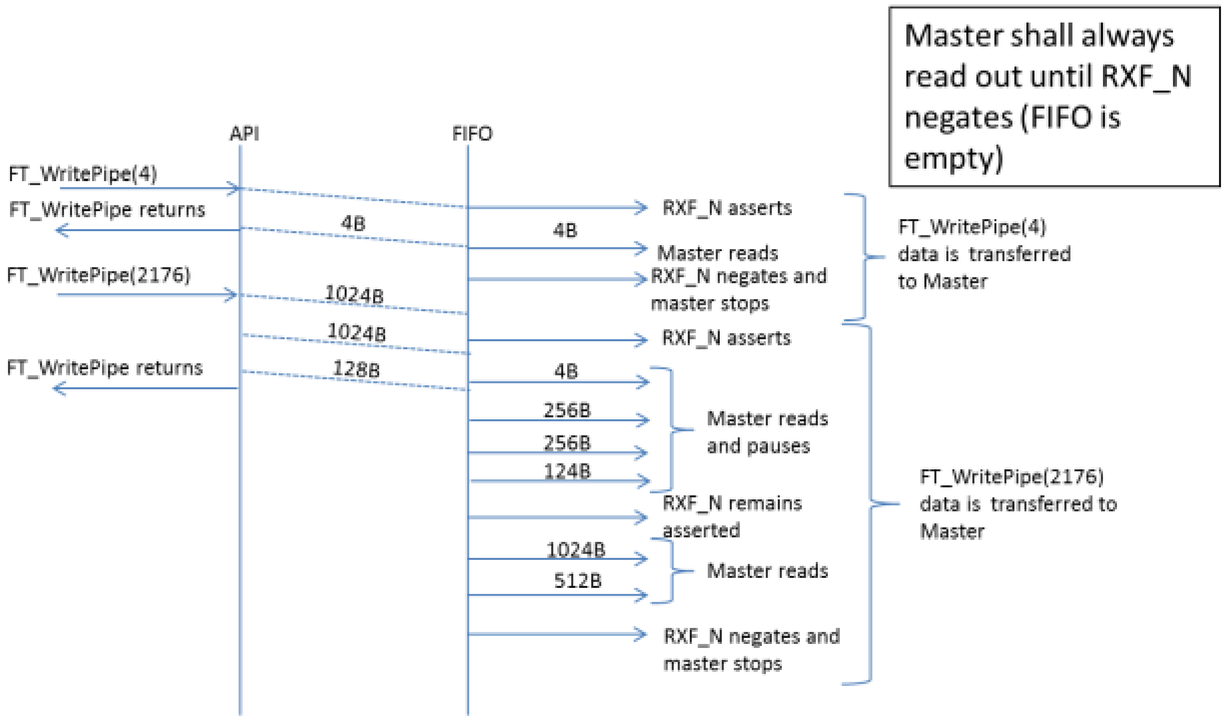
Master shall always read out until RXF_N negates (FIFO is empty)
API
FIFO
FT_WritePipe(4)
FT_WritePipe returns
FT_WritePipe(2176)
FT_WritePipe returns
4B
1024B
1024B
128B
4B
RXF_N asserts
Master reads
RXF_N negates and master stops
FT_WritePipe(4) data is transferred to Master
RXF_N asserts
4B
256B
256B
124B
Master reads and pauses
RXF_N remains asserted
1024B
512B
Master reads
RXF_N negates and master stops
FT_WritePipe(2176) data is transferred to Master

Figure 3 - Write Pipe Semantics

The designer is advised to check chip errata, TN_168 FT600_601 Errata Technical Note, so that the appropriate workarounds may be designed to overcome the errata.

3 Operating Modes

3.1 Chip Configuration

The chip is highly configurable and configurations are intended to provide flexibility in the manner in which USB packets are exchanged between the host and USB function (FIFO master). The important configuration options are explained below:

1. FIFO bus interface selection
2. FIFO channel and buffer selection
3. Session Underrun Handling
4. Notification Pipes

3.1.1 FIFO Bus Interface

The FIFO interface may operate in FT245 mode or in FT600 mode. The FT245 mode is a popular FTDI proprietary bus interface supported by most FTDI USB2.0 bridge devices. The FT600 mode is also a FTDI proprietary bus implemented in the FT600 and FT601 devices.

The main difference between FT245 and FT600 bus interfaces is the number of channels that they are able to support. The FT245 bus interface does not carry channel information and therefore supports only one channel while the FT600 bus interface is defined for up to 4 channels. Each channel is bi-directional.

3.1.2 FIFO Channel and Buffer

There are internal buffers in the device that are sized automatically depending on the number of channels that are configured. Each channel has 2 buffers (ping-pong) in each direction.

Bus Interface	Channel	Direction	Buffer size in each direction
FT245	1	IN only (uni-directional)	8KB
	1	OUT only (uni-directional)	8KB
	1	IN & OUT	4KB
FT600	1	IN only (uni-directional)	8KB
	1	OUT only (uni-directional)	8KB
	1	IN & OUT	4KB
	2	IN & OUT	2KB
	3	Not supported	-
	4	IN & OUT	1KB

Table 1 - FIFO Channel and Buffer

3.1.3 Session Underrun Handling

A session is initiated when a transfer request is initiated by the user application. A call to FT_ReadPipe or FT_WritePipe initiates a session. A session may trigger one or more transfer requests at the USB transfer level. The FIFO master reads or writes to the FIFO only when it is signaled that space or data is available for writing or reading.

A session underrun is a condition that occurs on IN endpoints. A session underrun is detected when the master stops writing (FT245 mode: WR_N is negated before TXE_N is negated) while there is more space in the FIFO to accept data. When this condition is detected, the data in the FIFO is transferred to the host.

The device may be configured to disable underrun detection. See the FT60X Configuration Programmer User Guide for further information.

When detection is disabled, the session remains open as long as the master pause the write at multiples of maximum sized packets of data and continues until the full length is reached.

3.1.4 Notification Pipes

Notifications may be enabled and only apply to IN endpoints of a channel. They are provided as a means to signal to the host how much data has been written into the FIFO by the master. The length of data that can be written is limited by the size of the buffer which is configured according to the number of channels. For example, in FT245 mode, the FIFO size is configured as 4KB and in each write, the master may only write 4KB of data.

In order to receive notifications, the user application has to register a callback with the API. When a notification is received, the callback to the user application indicates the length of data written into the FIFO and the user application must issue an FT_ReadPipe to read exactly the length of data reported in the callback notification. Reading less data than was signaled in the notification callback may result in FT_ReadPipe timeouts. Consequently, the user application shall only issue an FT_ReadPipe when notified and not independently of a notification.

When notifications are enabled on any endpoint, then the device will not be placed into selective suspend by the driver to ensure that the FIFO interface clock is not turned off.

3.1.5 GPIO Configuration

Two GPIO pins may be configured for general usage. These pins may be configured as input or output pins. The availability of the pins as GPIO is dependent on the chip configuration. At power-up or after a reset, the bridge checks if a valid chip configuration is present in the non-volatile memory. If a valid chip configuration is found, the pins are treated as user configurable GPIO pins otherwise the GPIO pins are treated as interface configuration pins and not available for GPIO operation.

Chip Configuration: Valid			
GPIO[1:0]	Direction	Data	Interpretation
User configurable	Input or Output	User defined	User defined
Chip Configuration: Invalid			
GPIO[1:0]	Direction	Data	Interpretation

Not available	Input	2'b00	FT245
		2'b01	FT600-1 Channel
		2'b10	FT600-2 Channel
		2'b11	FT600-4 Channel

Table 2 - GPIO Configuration

When the GPIO pins are available for general usage, they may be configured via the chip configuration and, additionally, via the GPIO API at run-time. The driver reads the chip configuration to determine the default state of the GPIO pins and keeps track of the changes to GPIO settings and directions. However, in cases where the driver is re-loaded without the bridge being reset in tandem (e.g. in self-powered devices), there is a chance that the driver's status and the actual pin status become unsynchronized. In this case, the application has to ensure that the GPIO direction and settings are synchronized.

When the chip configuration has been erased or is invalid, the user application may not use the GPIO pins and shall not use the GPIO APIs.

3.2 Abort Recovery

Sometimes a user application may wish to restart and reinitialize communication with its FIFO master. At other times, the FIFO master may become unresponsive or the bridge itself may enter into an unresponsive state. When the bridge or FIFO master is unresponsive, the application detects this via either an FT_ReadPipe or FT_WritePipe timeout. In order to recover from such an exception, there are two requirements to be fulfilled from the application level.

1. The application has to signal to the FIFO master that an abort is in progress. This signaling has to be carried out in an Out-Of-Band (OOB) manner.
2. The FIFO master needs to perform an unaligned write on the IN FIFO and a continuous read on the OUT FIFO in order to clear the FIFOs.

When FT_AbortPipe is called on a pipe, it causes all data that buffered in the pipe to be removed. The pipe is a logical construct that has one end at the D3XX API and the other end at the FIFO interface. The following sections illustrate read and write pipe abort recovery procedures.

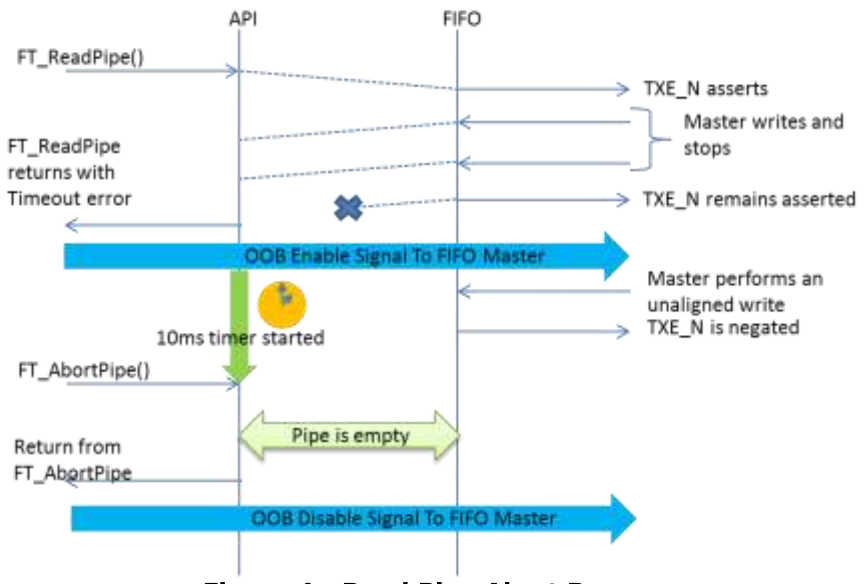

Figure 4 - Read Pipe Abort Recovery

The read pipe abort recovery process is as follows:

1. FT_ReadPipe is called on a pipe.
2. FT_ReadPipe returns with a timeout error.
3. Application initiates abort recovery
4. Application asserts a read side OOB signal to the FIFO master
5. FIFO master detects the read side OOB signal assertion
6. FIFO master performs an unaligned write if the corresponding FIFO data signal is asserted (e.g. in FT245 mode, the TXE_N signal is asserted low when the FIFO has space to receive data). An unaligned write ensures that the corresponding session at the bridge is closed and TXE_N will be negated
7. Application waits 10ms (or longer depending on FIFO master design)
8. Application calls FT_AbortPipe and returns
9. Application negates the read side OOB signal to the FIFO master
10. Abort procedure ends (pipe is empty)
11. Application may resume data transfer

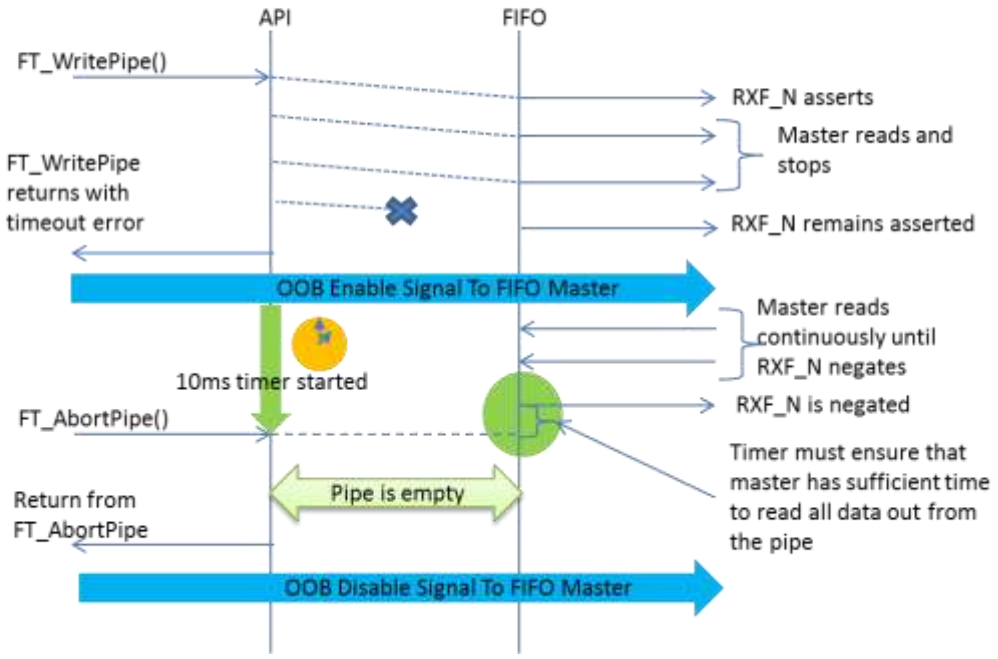
API
FIFO
FT_WritePipe()
RXF_N asserts
Master reads and stops
FT_WritePipe returns with timeout error
RXF_N remains asserted
OOB Enable Signal To FIFO Master
Master reads continuously until RXF_N negates
10ms timer started
FT_AbortPipe()
RXF_N is negated
Timer must ensure that master has sufficient time to read all data out from the pipe
Pipe is empty
Return from FT_AbortPipe
OOB Disable Signal To FIFO Master

Figure 5 - Write Pipe Abort Recovery

The write pipe abort recovery process is as follows:

1. FT_WritePipe is called on a pipe.
2. FT_WritePipe returns with a timeout error.
3. Application initiates abort recovery
4. Application asserts a write side OOB signal to the FIFO master
5. Application starts a 10ms (or longer) delay timer

6. FIFO master detects the write side OOB signal assertion
7. FIFO master reads data from the pipe until no more data is signaled (e.g. In FT245 mode, RXF_N signal is negated)
8. Application timer expires
9. Application calls FT_AbortPipe and returns from the call
10. Application negates the write side OOB signal to the FIFO master
11. Abort procedure ends (pipe is empty)

3.3 OOB Signals

3.3.1 GPIO as OOB Signals

The bridge provides 2 user programmable GPIO signals that may be used in a variety of ways.

1. The two GPIO signals may each be used as a read side OOB and a write side OOB signal. This allocation works well with the FT245 bus interface mode.
2. If a FIFO master reset is required, then one GPIO signal may be used as a FIFO master reset signal and the other as the OOB signal. In this situation, the OOB is treated as a global OOB for all channels.

3.3.2 Channel Status as OOB Signals

In the FT600 interface mode, up to 4 channels may be used. Then if there are any unused channels, the status bits of unused channels may be used to signal an OOB to the FIFO master. However, allocating an unused channel reduces the FIFO buffer allocated to the other channels. So, the designer has to balance FIFO buffer size and application bandwidth requirements against flexibility of using the status bits as OOB signals.

For example, if only two channels are used in FT600 mode, then the status bits of the other two channels may be used to signal OOB. A read or write on the unused pipe may be used as a signal to the FIFO master that an abort is required in the corresponding direction. The FIFO interface master has to remove the data in the OUT endpoint or write data into the IN endpoint to “acknowledge” the OOB request.

4 Contact Information

Head Office – Glasgow, UK

Future Technology Devices International Limited
Unit 1, 2 Seaward Place, Centurion Business Park
Glasgow G41 1HH
United Kingdom
Tel: +44 (0) 141 429 2777
Fax: +44 (0) 141 429 2758

E-mail (Sales)	sales1@ftdichip.com
E-mail (Support)	support1@ftdichip.com
E-mail (General Enquiries)	admin1@ftdichip.com

Branch Office – Tigard, Oregon, USA

Future Technology Devices International Limited (USA)
7130 SW Fir Loop
Tigard, OR 97223-8160
USA
Tel: +1 (503) 547 0988
Fax: +1 (503) 547 0987

E-mail (Sales)	us.sales@ftdichip.com
E-mail (Support)	us.support@ftdichip.com
E-mail (General Enquiries)	us.admin@ftdichip.com

Branch Office – Taipei, Taiwan

Future Technology Devices International Limited (Taiwan)
2F, No. 516, Sec. 1, NeiHu Road
Taipei 114
Taiwan , R.O.C.
Tel: +886 (0) 2 8797 1330
Fax: +886 (0) 2 8751 9737

E-mail (Sales)	tw.sales1@ftdichip.com
E-mail (Support)	tw.support1@ftdichip.com
E-mail (General Enquiries)	tw.admin1@ftdichip.com

Branch Office – Shanghai, China

Future Technology Devices International Limited (China)
Room 1103, No. 666 West Huaihai Road,
Shanghai, 200052
China
Tel: +86 21 62351596
Fax: +86 21 62351595

E-mail (Sales)	cn.sales@ftdichip.com
E-mail (Support)	cn.support@ftdichip.com
E-mail (General Enquiries)	cn.admin@ftdichip.com

Web Site

http://ftdichip.com

Distributor and Sales Representatives

Please visit the Sales Network page of the FTDI Web site for the contact details of our distributor(s) and sales representative(s) in your country.

Appendix A - References

Document References

- TN_168_FT600_601 Errata Technical Note
- AN_379 D3XX Programmers Guide
- DS_FT600Q-FT601Q IC Datasheet

Acronyms and Abbreviations

Terms	Description
Aligned read	An aligned read is a read from the FIFO interface when the slave asserts all byte enables. On a 16-bit interface, an aligned read is 16-bits wide and on a 32-bit interface, an aligned read is 32-bits wide
Aligned write	An aligned write is a write on the FIFO interface with all byte enables selected, i.e. on a 16-bit interface, an aligned write is 16-bits wide and on a 32-bit interface, an aligned write is 32-bits wide.
Cancel Session On Underrun	The device closes a session when it detects an underrun. A session underrun is detected when the FIFO interface reports that the master has stopped writing data before the FIFO buffer is full and the session requested length has not been reached.
Maximum sized packet	In USB2.0 (High Speed), a maximum sized packet has a length equal to 512 bytes and in USB 3.0 (Super Speed); a maximum sized packet has a length equal to 1024 bytes.
Pipe	A pipe is a logical construct that has one end at the D3XX API and the other end at the FIFO interface.
Session	A session consists of one or more USB transfers. A session is created when a read or write request is received by the driver layer. A session ends when the required length of data has been transferred or a short packet is received (session underrun).
Session Underrun	A session underrun is detected when a read request is active and the FIFO interface master wrote less than the requested number of bytes and stopped. e.g. If 1MB of data is requested in a read request, and the FIFO interface master wrote only 1025 bytes and stopped, the device detects this condition as a session underrun and ends the session.
Short packet	A short packet – a packet that is less than the maximum sized packet size. In USB2.0 (High Speed), a short packet has a non-zero length less than 512 bytes and in USB 3.0 (Super Speed), a short packet has a non-zero length less than 1024 bytes. A short packet signals an end of transfer on the endpoint.
Unaligned read	An unaligned read is a read from the FIFO interface that is not as wide as the interface, i.e. on a 16-bit interface, an unaligned read is a single byte read and on a 32-bit interface, an unaligned read is 1, 2, or 3 bytes. An

Terms	Description
	unaligned read from the slave always signals that it is a short packet
Unaligned write	An unaligned write is a write on the FIFO interface that is not as wide as the interface, i.e. on a 16-bit interface, an unaligned write is a single byte write and on a 32-bit interface, an unaligned write is 1, 2, or 3 bytes.
ZLP	A USB Zero-length packet

Appendix B – List of Tables & Figures

List of Tables

List of Figures

Appendix C – Revision History

Document Title	:	AN_412 FT600_FT601 USB Bridge chips Integration
Document Reference No.	:	FT_001332
Clearance No.	:	FTDI# 505
Product Page	:	http://www.ftdichip.com/FTProducts.htm
Document Feedback	:	Send Feedback

Revision	Changes	Date
1.0	Initial Release	2016-07-07
1.1	Updated Release	2017-12-08