Digilent Adept JTAG Interface (DJTG) Programmer's Reference Manual

Revision: August 16, 2010

1300 NE Henley Court, Suite 3
Pullman, WA 99163
(509) 334 6306 Voice | (509) 334 6300 Fax

Introduction

This document describes the programming interface to the Digilent Adept JTAG Interface (DJTG) subsystem for version 2 of the Digilent Adept software system. It describes the capabilities of the DJTG subsystem and the API functions used to access its features.

JTAG (Joint Test Access Group) is a synchronous serial interface primarily intended for in-system test, debug, and programming of integrated circuits. It is defined and described as I.E.E.E specification 1149.1. Familiarity with the I.E.E.E. 1149 specification is assumed in this document.

The JTAG specification defines four mandatory signals: TMS (Test Mode Select), TDI (Test Data In), TDO (Test Data Out), and TCK (Test Clock). JTAG compatible devices are connected in a serial arrangement, with the TDO of one device being connected to the TDI of the next device, forming a large shift register. This serial arrangement is called a JTAG scan chain.

The JTAG specification also defines a TAP (Test Access Port) controller. The TAP controller is a state machine that is used as the primary interface to the JTAG functionality of the device or devices on the JTAG scan chain. The TMS pin is used to move the TAP controller through its state diagram to various TAP states. The TMS pin is sampled on the rising edge of TCK and the next TAP state is determined by the current state and the signal level on TMS. The TAP controller contains an instruction register and a data register, and data can be shifted into and/or out of these registers when in the appropriate TAP controller states. Data shifted into the registers is presented on the TDI input and data shifted out is presented at the TDO output of the scan chain.

The DJTG interface provides a set of API functions that can be used to manipulate the TAP controller state as well as to shift data into and out of the JTAG scan chain.

All DJTG API calls return a Boolean value: TRUE if the call is successful, FALSE if not successful.

DJTG Port Properties

The port property bits are used to indicate which optional parts of the DJTG interface are supported by a given port. A DJTG port corresponds to a single set of the four JTAG signals and connects to a single JTAG scan chain.

The following port properties bits are defined for DJTG ports: These values are defined in the header file *djtg.h*.

dprpJtgSetSpeed	This bit indicates that the DjtgSetSpeed function is supported by the port.
dprpJtgSetPinState	This bit indicates that the DjtgSetTmsTdiTck function is supported by the port.

DJTG API Functions

The following API functions make up the DJTG interface.

DjtgGetVersion(char * szVersion)

Parameters
szVersion - pointer to buffer to receive version string

This function returns a version number string identifying the version number of the DJTG DLL. The symbol cchVersionMax declared in dpcdecl.h defines the longest string that can be returned in *szVersion*.

DjtgGetPortCount(HIF hif, INT32 * pcprt)

Parameters
hif - open interface handle on the device
pcprt - pointer to variable to receive count of ports

This function returns the number of DJTG ports supported by the device specified by interface handle *hif*.

DjtgGetPortProperties(HIF hif, INT32 prtReq, DWORD * pdprp)

Parameters
hif - open interface handle on the device
prtReq - port number to query
pdprp - pointer to variable to return port property bits

This function returns the port properties bits for the specified DJTG port. The port properties bits indicate the specific features of the DJTG specification implemented by the specified port.

DjtgEnable(HIF hif)

Parameters
hif - open interface handle on the device

This function is used to enable the default DJTG port (port 0) on the specified device. This function must be called before any functions that operate on the DJTG port may be called for the specified device.

DjtgEnableEx(HIF hif, INT32 prtReq)

Parameters
hif - open interface handle on the device
prtReq - DJTG port number

This function is used to enable a specific port on devices that support multiple DJTG ports. This function must be called before any functions that operate on the DJTG port may be called. The *prtReq* parameter specifies the port number of the DJTG port to enable.

DjtgDisable(HIF hif)

Parameters

hif	- open interface handle on the device

This function is used to disable and end access to the currently enabled DJTG port on the specified interface handle.

DjtgGetSpeed(HIF hif, DWORD * pfrqCur)

Parameters

hif	- open interface handle on the device
pfrqCur	- pointer to return current TCK frequency

This function is used to get the current speed of the DJTG port. The value returned in pfrqCur is in HZ.

DjtgSetSpeed(HIF hif, DWORD frqReq, DWORD * pfrqSet)

Parameters

hif	- open interface handle on the device
frqReq	- requested TCK clock frequency (in Hz)
pfrqSet	- pointer to return actual TCK frequency obtained

This function is used to set the clock frequency at which TCK will be clocked when shifting data into the JTAG scan chain. The TCK clock frequency will be set to the highest supported value that doesn't exceed frqReq, or the lowest frequency supported if frqReq is lower than the lowest supported value. The actual clock frequency obtained is returned in frqSet. This function is not supported unless the dprpJtgSetSpeed bit is set in the port properties for the port.

The clock frequency set will be the highest frequency that will appear on the TCK pin of the port. Depending on the device and the communications protocol used to communicate with the device, the average clock frequency (and therefore the average data rate in to or out of the device) may be lower.

DjtgSetTmsTdiTck(HIF hif, BOOL fTms, BOOL fTdi, BOOL fTck)

Parameters

hif	- open interface handle on the device
fTms	- sets the state of the TMS pin
fTdi	- sets the state of the TDI pin
fTck	- sets the state of the TCK pin

This function is used to set the JTAG interface pins to the specified states. A boolean TRUE value causes a pin to be set to the logic 1 state and a FALSE value causes the pin to be set to the logic 0 state. This function is not supported unless the dprpJtgSetPinState bit is set in the port properties for the port.

DjtgGetTmsTdiTdoTck(HIF hif, BOOL* pfTms, BOOL* pfTdi, BOOL* pfTdo, BOOL* pfTck)

Parameters

hif	- open interface handle on the device
pfTms	- pointer to return current state of TMS pin (true = 1, false = 0)
pfTdi	- pointer to return current state of TDI pin (true = 1, false = 0)
pfTdo	- pointer to return current state of TDO pin (true = 1, false = 0)
pfTck	- pointer to return current state of TCK pin (true = 1, false = 0)

This function is used to read the current state of the TMS, TDI, TDO, and TCK pins.

DjtgPutTdiBits(HIF hif, BOOL fTms, BYTE * rgbSnd, BYTE * rgbRcv, DWORD cbits, BOOL fOverlap)

Parameters

hif	- open interface handle on the device
fTms	- value theTMS pin will be held at during shifting (true = 1, false = 0)
rgbSnd	- buffer containing TDI bits. Bits are shifted in sequentially starting at the first element in the array, from LSB to MSB.
rgbRcv	-buffer to hold TDO bits. If not used, set to NULL.
cbits	- number of TDI bits to clock into the TAP controller
fOverlap	- TRUE if operation should be overlapped

This function shifts a specified number of bits into TDI and (optionally) returns bits shifted out TDO. The Buffer ggbSnd specifies bits to be shifted into TDI. Each bit is shifted in sequentially, starting at the first element in the array, from least significant bit to most significant bit. Below is an example of how the TDI bits are placed in each byte of rgbSnd.

rgbSnd[0]

TDI 8	TDI 7	TDI 6	TDI 5	TDI 4	TDI 3	TDI 2	TDI 1

rgbSnd[1]

TDI 16	TDI 15	TDI 14	TDI 13	TDI 12	TDI 11	TDI 10	TDI 9

If fReturnTdo is set to true, all bits shifted out of TDO are stored in rgbRcv. Each bit is shifted out sequentially starting at the first element in the rgbRcv, from lowest significant bit to most significant bit. TMS is held at the value specified by fTms while bits are being shifted into TDI.

DjtgPutTmsBits(HIF hif, BOOL fTdi, BYTE * rgbSnd, BYTE * rgbRcv, DWORD cbits, BOOL fOverlap)

Parameters

hif	- open interface handle on the device
fTdi	- value the TDI pin will be held at during shifting (true = 1, false = 0)
rgbSnd	- buffer containing TMS bits. Bits are shifted in sequentially starting at the first element in the array, from LSB to MSB.
rgbRcv	-buffer to hold TDO bits. If not used, set to NULL.
cbits	- number of TDI bits to clock into the TAP controller
fOverlap	- TRUE if operation should be overlapped

This function shifts a specified number of bits into TMS and (optionally) returns bits shifted out of TDO. The buffer rgbSnd specifies the bits to be shifted into TMS. Each bit is shifted in sequentially, starting at the first element in the array, from least significant bit to most significant bit. Below is an example of how the TMS bits are placed in each byte of rgbSnd.

rgbSnd[0]

TMS 8	TMS 7	TMS 6	TMS 5	TMS 4	TMS 3	TMS 2	TMS 1

rgbSnd[1]

TMS 16	TMS 15	TMS 14	TMS 13	TMS 12	TMS 11	TMS 10	TMS 9

If fReturnTdo is set to true, all bits shifted out of TDO are stored in rgbRcv. Each bit is shifted out sequentially starting at the first element in the rgbRcv, from lowest significant bit to most significant bit. TDI is held at the value specified by fTdi while bits are being shifted into TMS.

DjtgPutTmsTdiBits(HIF hif, BYTE * rgbSnd, BYTE * rgbRcv, DWORD cbitpairs, BOOL fOverlap)

Parameters

hif	- open interface handle on the device
rgbSnd	- buffer containing TMS bits. Bits are shifted in sequentially starting at the first element in the array, from LSB to MSB.
rgbRcv	-buffer to hold TDO bits. If not used, set to NULL.
cbitpairs	- number ofTMS and TDI bit pairs
fOverlap	- TRUE if operation should be overlapped

This function shifts a specified number of bit pairs into TMS and TDI and (optionally) returns bits shifted out of TDO. The buffer rgbSnd specifies the bit pairs to be shifted intoTMS and TDI. Each bit pair is shifted in sequentially; starting at the first element in the array, from least significant bit pair to most significant bit pair. In each pair, the TMS value is the MSB and the TDI value is the LSB. Below is an example of how the TMS/TDI bit pairs are placed in each byte of rgbSnd.

rgbSnd[0]

TMS 4	TDI 4	TMS 3	TDI 3	TMS 2	TDI 2	TMS 1	TDI 1

rgbSnd[1]

TMS 8	TDI 8	TMS 7	TDI 7	TMS 6	TDI 6	TMS 5	TDI 5

If fReturnTdo is set to true, all bits shifted out of TDO are stored in rgbRcv. Each bit is shifted out sequentially starting at the first element in the rgbRcv, from lowest significant bit to most significant bit.

DjtgGetTdoBits(HIF hif, BOOL fTdi, BOOL fTms, BYTE * rgbRcv, DWORD cbits, BOOL fOverlap)

Parameters

hif	- open interface handle on the device
fTdi	- value the TDI pin will be held at during shifting (true = 1, false = 0)
fTms	- value the TMS pin will be held at during shifting (true = 1, false = 0)
rgbRcv	- buffer to hold TDO bits
cbits	- number of TMS and TDI bit pairs
fOverlap	- TRUE if operation should be overlapped

This function shifts the specified number of bits out of TDO. The TDI pin will be held at the state specified by fTdi. If fTdi is TRUE the pin will be held at the logic '1' state. If fTdi is FALSE, the pin will be held at the logic '0' state. The bits shifted out of TDO are stored in rgbRcv. Each bit is shifted out sequentially starting at the first element in the rgbRcv, from lowest significant bit to most significant bit.

DjtgClockTck(HIF hif, BOOL fTms, BOOL fTdi, DWORD cclk, BOOL fOverlap)

Parameters

hif	- open interface handle on the device
fTdi	- value the TDI pin will be held at during shifting (true = 1, false = 0)
fTms	- value the TMS pin will be held at during shifting (true = 1, false = 0)
cclk	- number of cycles to clock TCK.
fOverlap	- TRUE if operation should be overlapped

This function clocks TCK a specified number of cycles. During this time, TDI and TMS are held at the specified values. TDO is not sampled.