API Reference for C FPGA Interface

Errors

const NiFpga_Status NiFpga_Status_Success = 0 `[static]`

No errors or warnings.

const NiFpga_Status NiFpga_Status_MemoryFull = -52000 `[static]`

A memory allocation failed.

const NiFpga_Status NiFpga_Status_FeatureUnsupported = -52002 `[static]`

The requested feature is not supported.

const NiFpga_Status NiFpga_Status_SoftwareFault = -52003 `[static]`

An unexpected software error occurred.

const NiFpga_Status NiFpga_Status_InvalidParameter = -52005 `[static]`

The parameter to a function was invalid.

const NiFpga_Status NiFpga_Status_ResourceNotFound = -52006 `[static]`

A needed resource was not found. This could be the NiFpga.dll/NiFpga.out library, or the .lvbitx bitfile.

const NiFpga_Status NiFpga_Status_ResourceNotInitialized = -52010 `[static]`

A needed resource was not properly initialized. This could occur if NiFpga_Initialize was not called.

const NiFpga_Status NiFpga_Status_FpgaBusy = -61141 `[static]`

The FPGA is busy.

const NiFpga_Status NiFpga_Status_FpgaInternalError = -61499 `[static]`

An unexpected internal error occurred.

Types

typedef int32_t NiFpga_Status

The result of a function. Negative values are errors and positive values are warnings.

typedef uint32_t NiFpga_Bool

A boolean value; either NiFpga_False or NiFpga_True.

typedef uint32_t NiFpga_Session

A handle to an FPGA session.

typedef void* NiFpga_IrqContext

See NiFpga_ReserveIrqContext for more information.

Constants

const NiFpga_Bool NiFpga_False = 0 [static]

Represents a false condition.

const NiFpga_Bool NiFpga_True = 1 [static]

Represents a true condition.

const uint32_t NiFpga_Infinite = 0xFFFFFFFF [static]

Represents an infinite timeout.

enum NiFpga_Irq

Enumeration of all possible IRQs.

Enumerator:

NiFpga_Irq_0
NiFpga_Irq_1
NiFpga_Irq_2
NiFpga_Irq_3
NiFpga_Irq_4
NiFpga_Irq_5
NiFpga_Irq_6
NiFpga_Irq_7
NiFpga_Irq_8
NiFpga_Irq_9
NiFpga_Irq_10
NiFpga_Irq_11
NiFpga_Irq_12
NiFpga_Irq_13
NiFpga_Irq_14
NiFpga_Irq_15
NiFpga_Irq_16
NiFpga_Irq_17
NiFpga_Irq_18
NiFpga_Irq_19
NiFpga_Irq_20
NiFpga_Irq_21
NiFpga_Irq_22
NiFpga_Irq_23
NiFpga_Irq_24
NiFpga_Irq_25
NiFpga_Irq_26
NiFpga_Irq_27
NiFpga_Irq_28
NiFpga_Irq_29
NiFpga_Irq_30
NiFpga_Irq_31

enum NiFpga_OpenAttribute

Attributes that NiFpga_Open accepts.

Enumerator:

NiFpga_OpenAttribute_NoRun

enum NiFpga_CloseAttribute

Attributes that NiFpga_Close accepts.

Enumerator:

NiFpga_CloseAttribute_NoResetIfLastSession

enum NiFpga_RunAttribute

Attributes that NiFpga_Run accepts.

Enumerator:

NiFpga_RunAttribute_WaitUntilDone

Required Functions

NiFpga_Status NiFpga_Initialize (void)

This must be called before all other function calls.

Warning:

This function is not thread safe.

Returns:

result of the call

NiFpga_Status NiFpga_Finalize (void)

This must be called after all other function calls.

Warning:

This function is not thread safe.

Returns:

result of the call

Status Functions

static NiFpga_Bool NiFpga_IsFatal (const NiFpga_Status *status*) `[inline, static]`

Tests whether a status is fatal. Errors are fatal but warnings are not.

Parameters:

status status to check for fatal

static NiFpga_Bool NiFpga_IsNotFatal (const NiFpga_Status *status*) `[inline, static]`

Tests whether a status is non-fatal. Success and warnings are not fatal.

Parameters:

status status to check for fatal

static NiFpga_Status NiFpga_SetStatus (NiFpga_Status *const *status*, const NiFpga_Status *newStatus*) `[inline, static]`

Conditionally sets the status to a new value. The previous status will be preserved unless the new status is "more fatal". This means that warnings or errors will overwrite successes, and only errors will overwrite warnings. Errors will never be overwritten with new errors, and warnings will never be overwritten with new warnings.

Parameters:

status status to conditionally set
newStatus new status value that may be set

#define NiFpga_IfIsNotFatal(status, expression)

Only evaluates the expression if the status is not fatal. The expression must evalutate to an NiFpga_Status, such as a call to any NiFpga_* function, as the status will be set to the returned status if the expression is evaluated.

This macro can be used to mimic status chaining in LabVIEW, where the status doesn't have to be explicitly checked after each call. Such code may look as follows:

NiFpga_Status status = NiFpga_Status_Success; NiFpga_IfIsNotFatal(status, NiFpga_WriteU32(...)); NiFpga_IfIsNotFatal(status, NiFpga_WriteU32(...)); NiFpga_IfIsNotFatal(status, NiFpga_WriteU32(...));

Parameters:

status status to check for fatal
expression expression to call if the incoming status is not fatal

---

Session Functions

NiFpga_Status NiFpga_Open (const char * *bitfile*, const char * *signature*, const char * *resource*, uint32_t *attribute*, NiFpga_Session * *session*)

Opens a session to the FPGA. This call will ensure that the contents of the bitfile are programmed to the FPGA. The FPGA will be run unless the NoRun attribute is used.

Note that the correct signature must be passed to ensure that your application was built with the correct generated header file for the bitfile found on disk.

On PharLap ETS systems, this path should be an absolute path starting with "C:\\".

Parameters:

bitfile path to the bitfile
signature signature of the bitfile
resource RIO resource string to open ("RIO0" or "rio://mysystem/RIO")
attribute bitwise OR of any NiFpga_OpenAttributes, or 0
session outputs the session handle, which must be closed when no longer needed

Returns:

result of the call

NiFpga_Status NiFpga_Close (NiFpga_Session *session*, uint32_t *attribute*)

Closes the session to the FPGA. The FPGA will be reset unless either another session is still open or the NoResetIfLastSession attribute is used.

Parameters:

session handle to a currently open session
attribute bitwise OR of any NiFpga_CloseAttributes, or 0

Returns:

result of the call

Method Functions

NiFpga_Status NiFpga_Run (NiFpga_Session *session*, uint32_t *attribute*)

Runs the FPGA VI on the target. If WaitUntilDone is used, the FPGA will run until it finishes executing (if ever).

Parameters:

session handle to a currently open session
attribute bitwise OR of any NiFpga_RunAttributes, or 0

Returns:

result of the call

NiFpga_Status NiFpga_Abort (NiFpga_Session *session*)

Aborts the FPGA VI.

Parameters:

session handle to a currently open session

Returns:

result of the call

NiFpga_Status NiFpga_Reset (NiFpga_Session *session*)

Resets the FPGA VI.

Parameters:

session handle to a currently open session

Returns:

result of the call

NiFpga_Status NiFpga_Download (NiFpga_Session *session*)

Re-downloads the FPGA bitstream to the target.

Parameters:

session handle to a currently open session

Returns:

result of the call

Read/Write Functions

NiFpga_Status NiFpga_ReadBool (NiFpga_Session *session*, uint32_t *indicator*, NiFpga_Bool * *value*)

Reads a boolean value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadI8 (NiFpga_Session *session*, uint32_t *indicator*, int8_t * *value*)

Reads a signed 8-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadU8 (NiFpga_Session *session*, uint32_t *indicator*, uint8_t * *value*)

Reads an unsigned 8-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadI16 (NiFpga_Session *session*, uint32_t *indicator*, int16_t * *value*)

Reads a signed 16-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadU16 (NiFpga_Session *session*, uint32_t *indicator*, uint16_t * *value*)

Reads an unsigned 16-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadI32 (NiFpga_Session *session*, uint32_t *indicator*, int32_t * *value*)

Reads a signed 32-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadU32 (NiFpga_Session *session*, uint32_t *indicator*, uint32_t * *value*)

Reads an unsigned 32-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadI64 (NiFpga_Session *session*, uint32_t *indicator*, int64_t * *value*)

Reads a signed 64-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_ReadU64 (NiFpga_Session *session*, uint32_t *indicator*, uint64_t * *value*)

Reads an unsigned 64-bit integer value from a given indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
value outputs the value that was read

Returns:

result of the call

NiFpga_Status NiFpga_WriteBool (NiFpga_Session *session*, uint32_t *control*, NiFpga_Bool *value*)

Writes a boolean value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteI8 (NiFpga_Session *session*, uint32_t *control*, int8_t *value*)

Writes a signed 8-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteU8 (NiFpga_Session *session*, uint32_t *control*, uint8_t *value*)

Writes an unsigned 8-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteI16 (NiFpga_Session *session*, uint32_t *control*, int16_t *value*)

Writes a signed 16-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteU16 (NiFpga_Session *session*, uint32_t *control*, uint16_t *value*)

Writes an unsigned 16-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteI32 (NiFpga_Session *session*, uint32_t *control*, int32_t *value*)

Writes a signed 32-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteU32 (NiFpga_Session *session*, uint32_t *control*, uint32_t *value*)

Writes an unsigned 32-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteI64 (NiFpga_Session *session*, uint32_t *control*, int64_t *value*)

Writes a signed 64-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_WriteU64 (NiFpga_Session *session*, uint32_t *control*, uint64_t *value*)

Writes an unsigned 64-bit integer value to a given control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
value value to write

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayBool (NiFpga_Session *session*, uint32_t *indicator*, NiFpga_Bool * *values*, uint32_t *size*)

Reads boolean values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayI8 (NiFpga_Session *session*, uint32_t *indicator*, int8_t * *values*, uint32_t *size*)

Reads signed 8-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayU8 (NiFpga_Session *session*, uint32_t *indicator*, uint8_t * *values*, uint32_t *size*)

Reads unsigned 8-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayI16 (NiFpga_Session *session*, uint32_t *indicator*, int16_t * *values*, uint32_t *size*)

Reads signed 16-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayU16 (NiFpga_Session *session*, uint32_t *indicator*, uint16_t * *values*, uint32_t *size*)

Reads unsigned 16-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayI32 (NiFpga_Session *session*, uint32_t *indicator*, int32_t * *values*, uint32_t *size*)

Reads signed 32-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayU32 (NiFpga_Session *session*, uint32_t *indicator*, uint32_t * *values*, uint32_t *size*)

Reads unsigned 32-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayI64 (NiFpga_Session *session*, uint32_t *indicator*, int64_t * *values*, uint32_t *size*)

Reads signed 64-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_ReadArrayU64 (NiFpga_Session *session*, uint32_t *indicator*, uint64_t * *values*, uint32_t *size*)

Reads unsigned 64-bit integer values from a given array indicator or control.

Parameters:

session handle to a currently open session
indicator indicator/control from which to read
values outputs the values that were read
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayBool (NiFpga_Session *session*, uint32_t *control*, const NiFpga_Bool * *values*, uint32_t *size*)

Writes boolean values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayI8 (NiFpga_Session *session*, uint32_t *control*, const int8_t * *values*, uint32_t *size*)

Writes signed 8-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayU8 (NiFpga_Session *session*, uint32_t *control*, const uint8_t * *values*, uint32_t *size*)

Writes unsigned 8-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayI16 (NiFpga_Session *session*, uint32_t *control*, const int16_t * *values*, uint32_t *size*)

Writes signed 16-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayU16 (NiFpga_Session *session*, uint32_t *control*, const uint16_t * *values*, uint32_t *size*)

Writes unsigned 16-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayI32 (NiFpga_Session *session*, uint32_t *control*, const int32_t * *values*, uint32_t *size*)

Writes signed 32-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write

size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayU32 (NiFpga_Session *session*, uint32_t *control*, const uint32_t * *values*, uint32_t *size*)

Writes unsigned 32-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayI64 (NiFpga_Session *session*, uint32_t *control*, const int64_t * *values*, uint32_t *size*)

Writes signed 64-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

NiFpga_Status NiFpga_WriteArrayU64 (NiFpga_Session *session*, uint32_t *control*, const uint64_t * *values*, uint32_t *size*)

Writes unsigned 64-bit integer values to a given array control or indicator.

Parameters:

session handle to a currently open session
control control/indicator to which to write
values values to write
size number of values in this array

Returns:

result of the call

Interrupt Functions

NiFpga_Status NiFpga_ReserveIrqContext (NiFpga_Session *session*, NiFpga_IrqContext * *context*)

IRQ contexts are single-threaded; Only one thread may be "waiting" on a particular context at any given time. Clients must reserve as many contexts as the application requires.

If a context is successfully reserved (returned status is not fatal), the context must be unreserved later. Otherwise, a memory leak will occur.

Parameters:

session handle to a currently open session
context outputs the IRQ context

Returns:

result of the call

NiFpga_Status NiFpga_UnreserveIrqContext (NiFpga_Session *session*, NiFpga_IrqContext *context*)

Unreserves an IRQ context obtained from NiFpga_ReserveIrqContext.

Parameters:

session handle to a currently open session
context IRQ context to unreserve

Returns:

result of the call

NiFpga_Status NiFpga_WaitOnIrqs (NiFpga_Session *session*, NiFpga_IrqContext *context*, uint32_t *irqs*, uint32_t *timeout*, uint32_t * *irqsAsserted*, NiFpga_Bool * *timedOut*)

Blocking call which stops the calling thread until the specified IRQs have been asserted on the FPGA, or the timeout period expires. An IRQ context must have been reserved earlier using NiFpga_ReserveIrqContext. The context must not be in use by other threads at the time of this call.

This call can be used to wait on a set of IRQs. If any IRQ in the set is asserted by the FPGA, this call will return. The irqsAsserted parameter can be used to determine which IRQs were asserted in this case.

Parameters:

session handle to a currently open session
context IRQ context with which to wait
irqs bitwise OR of NiFpga_Irqs
timeout timeout in milliseconds, or NiFpga_Infinite
irqsAsserted if non-NULL, outputs the set of IRQs that were asserted
timedOut if non-NULL, outputs whether the timeout expired

Returns:

result of the call

NiFpga_Status NiFpga_AcknowledgeIrqs (NiFpga_Session *session*, uint32_t *irqs*)

Acknowledges an IRQ or set of IRQs.

Parameters:

session handle to a currently open session
irqs bitwise OR of NiFpga_Irqs

Returns:

result of the call

FIFO Functions

NiFpga_Status NiFpga_ConfigureFifo (NiFpga_Session *session*, uint32_t *fifo*, uint32_t *depth*)

Configures the depth of a FIFO.

Parameters:

session handle to a currently open session
fifo the fifo to configure
depth the number of elements deep the fifo is

Returns:

result of the call

NiFpga_Status NiFpga_StartFifo (NiFpga_Session *session*, uint32_t *fifo*)

Starts a FIFO.

Parameters:

session handle to a currently open session
fifo the fifo to start

Returns:

result of the call

NiFpga_Status NiFpga_StopFifo (NiFpga_Session *session*, uint32_t *fifo*)

Stops a FIFO.

Parameters:

session handle to a currently open session
fifo the fifo to stop

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoBool (NiFpga_Session *session*, uint32_t *fifo*, NiFpga_Bool * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of booleans.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoI8 (NiFpga_Session *session*, uint32_t *fifo*, int8_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of signed 8-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoU8 (NiFpga_Session *session*, uint32_t *fifo*, uint8_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of unsigned 8-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoI16 (NiFpga_Session *session*, uint32_t *fifo*, int16_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of signed 16-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoU16 (NiFpga_Session *session*, uint32_t *fifo*, uint16_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of unsigned 16-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoI32 (NiFpga_Session *session*, uint32_t *fifo*, int32_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of signed 32-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoU32 (NiFpga_Session *session*, uint32_t *fifo*, uint32_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of unsigned 32-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoI64 (NiFpga_Session *session*, uint32_t *fifo*, int64_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of signed 64-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_ReadFifoU64 (NiFpga_Session *session*, uint32_t *fifo*, uint64_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Reads from a FIFO of unsigned 64-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo from which to read
data outputs the data that was read
count number of elements to read
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoBool (NiFpga_Session *session*, uint32_t *fifo*, const NiFpga_Bool * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of booleans.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write

count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoI8 (NiFpga_Session *session*, uint32_t *fifo*, const int8_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of signed 8-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoU8 (NiFpga_Session *session*, uint32_t *fifo*, const uint8_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of unsigned 8-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoI16 (NiFpga_Session *session*, uint32_t *fifo*, const int16_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of signed 16-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoU16 (NiFpga_Session *session*, uint32_t *fifo*, const uint16_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of unsigned 16-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoI32 (NiFpga_Session *session*, uint32_t *fifo*, const int32_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of signed 32-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoU32 (NiFpga_Session *session*, uint32_t *fifo*, const uint32_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of unsigned 32-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoI64 (NiFpga_Session *session*, uint32_t *fifo*, const int64_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of signed 64-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call

NiFpga_Status NiFpga_WriteFifoU64 (NiFpga_Session *session*, uint32_t *fifo*, const uint64_t * *data*, uint32_t *count*, uint32_t *timeout*, uint32_t * *remaining*)

Writes to a FIFO of unsigned 64-bit integers.

Parameters:

session handle to a currently open session
fifo the fifo to which to write
data data to write
count number of elements to write
timeout timeout in milliseconds, or NiFpga_Infinite
remaining if non-NULL, outputs the number of elements remaining

Returns:

result of the call