LINUX driver for PCI and Ethernet

C/C++ API Function Calls

Error codes and API function descriptions

a) A List of all the Error Code Returned by each C Function:

DMCNOERROR	0
DMCWARNING_MONITOR	1
DMCERROR_TIMEOUT	-1
DMCERROR_COMMAND	-2
DMCERROR_CONTROLLER	-3
DMCERROR_FILE	-4
DMCERROR_DRIVER	-5
DMCERROR_HANDLE	-6
DMCERROR_HMODULE	-7
DMCERROR_MEMORY	-8
DMCERROR_BUFFERFULL	-9
DMCERROR_RESPONSEDATA	-10
DMCERROR_DMA	-11
DMCERROR_ARGUMENT	-12
DMCERROR_DATARECORD	-13
DMCERROR_DOWNLOAD	-14
DMCERROR_FIRMWARE	-15
DMCERROR_CONVERSION	-16
DMCERROR_RESOURCE	-17
DMCERROR_REGISTRY	-18
DMCERROR_BUSY	-19
DMCERROR_DEVICE_DISCONNECTED	-20

b) A List of all the API Functions and There Prototypes:

extern LONG FAR GALILCALL DMCInitLibrary(void);
Initialize the library. This function MUST be called before using the library.

extern LONG FAR GALILCALL DMCOpen(PCONTROLLERINFO pcontrollerinfo, PHANDLEDMC phdmc);
Open communications with the Galil controller. The handle to the Galil controller is returned in the argument phdmc.

pcontrollerinfo	Galil controller information. Users should declare a variable of type CONTROLLERINFO, fill-in the necessary

values, and pass the address of the variable to the function.

phdmc — Buffer to receive the handle to the Galil controller to be used for all subsequent API calls. Users should declare a variable of type HANDLEDMC and pass the address of the variable to the function. Output only.

extern LONG FAR GALILCALL DMCClose(HANDLEDMC hdmc);
Close communications with the Galil controller.

hdmc — Handle to the Galil controller.

extern LONG FAR GALILCALL DMCEnableInterrupts(HANDLEDMC hdmc, pid_t pid);
Install an interrupt handler for a bus controller. When an interrupt occurs, the process identifed by pid will be notified via a QNX message

hdmc — Handle to the Galil controller.
pid — Process ID.

extern LONG FAR GALILCALL DMCDisableInterrupts(HANDLEDMC hdmc);
Remove an interrupt handler for a bus controller.

hdmc — Handle to the Galil controller.

extern LONG FAR GALILCALL DMCCommand(HANDLEDMC hdmc, PSZ pszCommand, PCHAR pchResponse, ULONG cbResponse);
Send a DMC command in ascii format to the Galil controller.

hdmc — Handle to the Galil controller.
pszCommand — The command to send to the Galil controller.
pchResponse — Buffer to receive the response data. If the buffer is too small to recieve all the response data from the controller, the error code DMCERROR_BUFFERFULL will be returned. The user may get additional response data by calling the function DMCGetAdditionalResponse. The length of the additonal response data may ascertained by call the function DMCGetAdditionalResponseLen. If the response data from the controller is too large for the internal additional response buffer, the error code DMCERROR_RESPONSEDATA will be returned. Output only.
cbResponse — Length of the buffer.

extern LONG FAR GALILCALL DMCBinaryCommand(HANDLEDMC hdmc, PBYTE pbCommand, ULONG ulCommandLength, PCHAR pchResponse, ULONG cbResponse);
Send a DMC command in binary format to the Galil controller.

hdmc — Handle to the Galil controller.
pbCommand — The command to send to the Galil controller in binary.
UlCommandLength — The length of the command (binary commands are not null-terminated).
pchResponse — Buffer to receive the response data. If the buffer is too small to recieve all the response data from the controller, the error code DMCERROR_BUFFERFULL will be returned. The user may get additional response data by calling the function DMCGetAdditionalResponse. The length of the additonal response data may ascertained by

	call the function DMCGetAdditionalResponseLen. If the response data from the controller is too large for the internal additional response buffer, the error code DMCERROR_RESPONSEDATA will be returned. Output only.
cbResponse	Length of the buffer.

extern LONG FAR GALILCALL DMCGetUnsolicitedResponse(HANDLEDMC hdmc, PCHAR pchResponse, ULONG cbResponse);
Query the Galil controller for unsolicited responses. These are messages output from programs running in the background in the Galil controller. The data placed in the user buffer (pchResponse) is NULL terminated.
NOTE: This function requires that the CW property be set to 1. The DMC command CW1 makes bit 7 high for characters in all messages which originate from the controller. This is how the library is able to distinguish between responses from commands (foreground) and messages from the controller(background). The DMC command CW2 disables this action.

hdmc	Handle to the Galil controller.
pchResponse	Buffer to receive the response data.
cbResponse	Length of the buffer.

extern LONG FAR GALILCALL DMCWriteData(HANDLEDMC hdmc, PCHAR pchBuffer, ULONG cbBuffer, PULONG pulBytesWritten);
Low-level I/O routine to write data to the Galil controller. Data is written to the Galil controller only if it is "ready" to receive it. The function will attempt to write exactly cbBuffer characters to the controller. NOTE: For Win32 and WinRT driver the maximum number of bytes which can be written each time is 64. There are no restrictions with the Galil driver.

hdmc	Handle to the Galil controller.
pchBuffer	Buffer to write the data from.
cbBuffer	Length of the buffer.
pulBytesWritten	Number of bytes written.

extern LONG FAR GALILCALL DMCReadData(HANDLEDMC hdmc, PCHAR pchBuffer, ULONG cbBuffer, PULONG pulBytesRead);
Low-level I/O routine to read data from the Galil controller. The routine will read what ever is currently in the FIFO (bus controller) or communications port input queue (serial controller). The function will read up to cbBuffer characters from the controller. The data placed in the user buffer (pchBuffer) is NOT NULL terminated. The data returned is not guaranteed to be a complete response - you may have to call this function repeatedly to get a complete response.
NOTE: For Win32 and WinRT driver the maximum number of bytes which can be read each time is 64. There are no restrictions with the Galil driver.

hdmc	Handle to the Galil controller.
pchBuffer	Buffer to read the data into.
cbBuffer	Length of the buffer.
pulBytesRead	Number of bytes read.

extern LONG FAR GALILCALL DMCGetAdditionalResponseLen(HANDLEDMC hdmc, PULONG pulResponseLen);
Query the Galil controller for the length of additional response data. There will be more response data available if the DMCCommand function returned DMCERROR_BUFFERFULL.

hdmc Handle to the Galil controller.
pulResponseLen Buffer to receive the additional response data length. Output only.

extern LONG FAR GALILCALL DMCGetAdditionalResponse(HANDLEDMC hdmc, PCHAR pchResponse, ULONG cbResponse);
Query the Galil controller for more response data. There will be more response data available if the DMCCommand function returned DMCERROR_BUFFERFULL. Once this function is called, the internal additonal response buffer is cleared.

hdmc Handle to the Galil controller.
pchResponse Buffer to receive the response data. Output only.
cbResponse Length of the buffer.

extern LONG FAR GALILCALL DMCGetInterruptStatus(HANDLEDMC hdmc, PUSHORT pusStatus);
Get the interrupt status from the Galil controller.

hdmc Handle to the Galil controller.
PUSHORT Buffer to recieve the interrupt status. Output only.

extern LONG FAR GALILCALL DMCError(HANDLEDMC hdmc, LONG lError, PCHAR pchMessage, ULONG cbMessage);
Retrieve the error message text from a DMCERROR_COMMAND error.

hdmc Handle to the Galil controller.
pchMessage Buffer to receive the error message text. Output only.
cbMessage Length of the buffer.

extern LONG FAR GALILCALL DMCClear(HANDLEDMC hdmc);
Clear the Galil controller FIFO.

hdmc Handle to the Galil controller.

extern LONG FAR GALILCALL DMCReset(HANDLEDMC hdmc);
Reset the Galil controller.

hdmc Handle to the Galil controller.

extern LONG FAR GALILCALL DMCMasterReset(HANDLEDMC hdmc);
Master reset the Galil controller.

hdmc Handle to the Galil controller.

extern LONG FAR GALILCALL DMCVersion(HANDLEDMC hdmc, PCHAR pchVersion, ULONG cbVersion);
Get the version of the Galil controller.

hdmc Handle to the Galil controller.
pchVersion Buffer to receive the version information. Output only.
cbVersion Length of the buffer.

extern LONG FAR GALILCALL DMCDownloadFile(HANDLEDMC hdmc, PSZ pszFileName, PSZ pszLabel);
Download a file to the Galil controller.

hdmc	Handle to the Galil controller.
pszFileName	File name to download to the Galil controller.
pszLabel	Program label to download to. This argument is ignored if NULL.

extern LONG FAR GALILCALL DMCDownloadFromBuffer(HANDLEDMC hdmc, PSZ pszBuffer, PSZ pszLabel);
Download from a buffer to the Galil controller.

hdmc	Handle to the Galil controller.
pszBuffer	Buffer of DMC commands to download to the Galil controller.
pszLabel	Program label to download to. This argument is ignored if NULL.

extern LONG FAR GALILCALL DMCUploadFile(HANDLEDMC hdmc, PSZ pszFileName);
Upload a file from the Galil controller.

hdmc	Handle to the Galil controller.
pszFileName	File name to upload from the Galil controller.

extern LONG FAR GALILCALL DMCUploadToBuffer(HANDLEDMC hdmc, PCHAR pchBuffer, ULONG cbBuffer);
Upload to a buffer from the Galil controller.

hdmc	Handle to the Galil controller.
pchBuffer	Buffer of DMC commands to upload from the Galil controller. Output only.
cbBuffer	Length of the buffer.

extern LONG FAR GALILCALL DMCSendFile(HANDLEDMC hdmc, PSZ pszFileName);
Send a file consisting of DMC commands in ascii format to the Galil controller.

hdmc	Handle to the Galil controller.
pszFileName	File name to send to the Galil controller.

The following lists the Binary communications functions for the DMC-1200, DMC-1600, DMC-1700, DMC-1800, and DMC-2000:

extern LONG FAR GALILCALL DMCSendBinaryFile(HANDLEDMC hdmc, PSZ pszFileName);
Send a file consisting of DMC commands in binary format to the Galil controller.

hdmc	Handle to the Galil controller.
pszFileName	File name to send to the Galil controller.

extern LONG FAR GALILCALL DMCArrayDownload(HANDLEDMC hdmc, PSZ pszArrayName, USHORT usFirstElement, USHORT usLastElement, PCHAR pchData, ULONG cbData, PULONG cbBytesWritten);
Download an array to the Galil controller. The array must exist. Array data can be delimited by a comma or CR (0x0D) or CR/LF (0x0D0A).
NOTE: The firmware on the controller must be recent enough to support the QD command.

hdmc	Handle to the Galil controller.
pszArrayName	Array name to download to the Galil controller.
usFirstElement	First array element.
usLastElement	Last array element.
pchData	Buffer to write the array data from. Data does not need to be NULL terminated.
cbData	Length of the array data in the buffer.
cbBytesWritten	Number of bytes written.

extern LONG FAR GALILCALL DMCArrayUpload(HANDLEDMC hdmc, PSZ pszArrayName, USHORT usFirstElement, USHORT usLastElement, PCHAR pchData, ULONG cbData, PULONG pulBytesRead, SHORT fComma);
Upload an array from the Galil controller. The array must exist. Array data will be delimited by a comma or CR (0x0D) depending of the value of fComma.
NOTE: The firmware on the controller must be recent enough to support the QU command.

hdmc	Handle to the Galil controller.
pszArrayName	Array name to upload from the Galil controller.
usFirstElement	First array element.
usLastElement	Last array element.
pchData	Buffer to read the array data into. Array data will not be NULL terminated.
cbData	Length of the buffer.
pulBytesRead	Number of bytes read.
fComma	0 = delimit by "\r", 1 = delimit by ",".

extern LONG FAR GALILCALL DMCCommand_AsciiToBinary(HANDLEDMC hdmc, PSZ pszAsciiCommand, ULONG ulAsciiCommandLength, PBYTE pbBinaryResult, ULONG cbBinaryResult, ULONG FAR *pulBinaryResultLength);
Convert an ascii DMC command to a binary DMC command.

hdmc	Handle to the Galil controller.
pszAsciiCommand	Ascii DMC command(s) to be converted.
ulAsciiCommandLength	Length of DMC command(s).
pbBinaryResult	Buffer to receive the translated DMC command.
cbBinaryResult	Length of the buffer.
pulBinaryResultLength	Length of the translated DMC command.

extern LONG FAR GALILCALL DMCCommand_BinaryToAscii(HANDLEDMC hdmc, PBYTE pbBinaryCommand, ULONG ulBinaryCommandLength, PSZ pszAsciiResult, ULONG cbAsciiResult, ULONG FAR *pulAsciiResultLength);
Convert a binary DMC command to an ascii DMC command.

hdmc	Handle to the Galil controller.
pbBinaryCommand	Binary DMC command(s) to be converted.
ulBinaryCommandLength	Length of DMC command(s).
pszAsciiResult	Buffer to receive the translated DMC command.
cbAsciiResult	Length of the buffer.
pulAsciiResultLength	Length of the translated DMC command.

extern LONG FAR GALILCALL DMCFile_AsciiToBinary(HANDLEDMC hdmc, PSZ pszInputFileName, PSZ pszOutputFileName);
Convert a file consisting of ascii commands to a file consisting of binary commands.

hdmc Handle to the Galil controller.
pszInputFileName File name for the input ascii file.
pszOutputFileName File name for the output binary file.

extern LONG FAR GALILCALL DMCFile_BinaryToAscii(HANDLEDMC hdmc, PSZ pszInputFileName, PSZ pszOutputFileName);
Convert a file consisting of binary commands to a file consisting of ascii commands.

hdmc Handle to the Galil controller.
pszInputFileName File name for the input binary file.
pszOutputFileName File name for the output ascii file.

extern LONG FAR GALILCALL DMCReadSpecialConversionFile(HANDLEDMC hdmc, PSZ pszFileName);
Read into memory a special BinaryToAscii/AsciiToBinary conversion table.

hdmc Handle to the Galil controller.
pszFileName File name for the special conversion file.

Data record access (DMA/FIFO) functions for the DMC-1600, DMC-1700, and DMC-1800:

extern LONG FAR GALILCALL DMCRefreshDataRecord(HANDLEDMC hdmc, ULONG ulLength);
Refresh the data record used for fast polling.

hdmc Handle to the Galil controller.
ulLength Refresh size in bytes. Set to 0 unless you do not want a full-buffer refresh.

extern LONG FAR GALILCALL DMCGetDataRecord(HANDLEDMC hdmc, USHORT usGeneralOffset, USHORT usAxisInfoOffset, PUSHORT pusDataType, PLONG plData);
Get a data item from the data record used for fast polling. Gets one item from the data record by using offsets (see data record constants defined in DMCDRC.H). To retrieve data record items by Id instead of offset, use the function DMCGetDataRecordByItemId.

hdmc Handle to the Galil controller.
usGeneralOffset Data record offset for general data item.
usAxisInfoOffset Additional data record offset for axis data item.
pusDataType Data type of the data item. If you are using the standard, pre-defined offsets, set this argument to zero before calling this function. The actual data type of the data item is returned on output.
plData Buffer to receive the data record data. Output only.

extern LONG FAR GALILCALL DMCGetDataRecordByItemId(HANDLEDMC hdmc, USHORT usItemId, USHORT usAxisId, PUSHORT pusDataType, PLONG plData);
Get a data item from the data record used for fast polling. Gets one item from the data record by using Id (see data record Ids defined in DMCDRC.H). To retrieve data record items by offset instead of Id, use the function DMCGetDataRecord.

hdmc	Handle to the Galil controller.
usItemId	Data record item Id.
usAxisId	Axis Id used for axis data items.
pusDataType	Data type of the data item. The data type of the data item is returned on output. Output Only.
plData	Buffer to receive the data record data. Output only.

extern LONG FAR GALILCALL DMCGetDataRecordSize(HANDLEDMC hdmc, PUSHORT pusRecordSize);
Get the size of the data record used for fast polling.

hdmc	Handle to the Galil controller.
pusRecordSize	The size of the data record is returned on output. Output Only.

extern LONG FAR GALILCALL DMCCopyDataRecord(HANDLEDMC hdmc, PVOID pDataRecord);
Get a copy of the data record used for fast polling. The data record is only as recent as the last call made to DMCRefreshDataRecord.

hdmc	Handle to the Galil controller.
pDataRecord	A copy of the data record is returned on output. Output Only.

extern LONG FAR GALILCALL DMCGetDataRecordRevision(HANDLEDMC hdmc, PUSHORT pusRevision);
Get the revision of the data record structure used for fast polling.

hdmc	Handle to the Galil controller.
pusRevision	The revision of the data record structure is returned on output. Output Only.

extern LONG FAR GALILCALL DMCGetDataRecordQR(HANDLEDMC hdmc, PDMCDATARECORDQR pdmcdatarecordqr, USHORT usRecordLength);
Get a copy of the data record used for fast polling via the Galil QR command. For use with controllers which do not support data record access through either DMA or secondary FIFO.

hdmc	Handle to the Galil controller.
pdmcdatarecordqr	Buffer to receive the data record data; pointer to a DMCDATARECORDQR structure. Output only.
usRecordLength	Length or size of the DMCDATARECORDQR structure (pdmcdatarecordqr). The size of this structure can be customized based on the number of axes. See the definition of DMCDATARECORDQR in DMCDRC.H.

Diagnostics functions for all controllers:

extern LONG FAR GALILCALL DMCDiagnosticsOn(HANDLEDMC hdmc, PSZ pszFileName, SHORT fAppend);
Turn on diagnostics.

hdmc	Handle to the Galil controller.
pszFileName	File name for the diagnostic file.
fAppend	TRUE if the file will open for append, otherwise FALSE.

extern LONG FAR GALILCALL DMCDiagnosticsOff(HANDLEDMC hdmc);

Turn off diagnostics.

hdmc Handle to the Galil controller.

Configuration functions for all controllers:

extern LONG FAR GALILCALL DMCGetTimeout(HANDLEDMC hdmc, LONG FAR* pTimeout);
Get current time-out value. The default is 1000.

hdmc Handle to the Galil controller.
pTimeout Buffer to receive the current time-out value in milliseconds. Output only.

extern LONG FAR GALILCALL DMCSetTimeout(HANDLEDMC hdmc, LONG lTimeout);
Set time-out value. The default is 1000. If the time-out value is set to zero, the DLLs will ignore time-out errors. This is useful for sending Galil commands which do not return a response, such as providing records to the DL or QD commands.

hdmc Handle to the Galil controller.
lTimeout Time-out value in milliseconds.

Utility functions:

extern LONG FAR GALILCALL DMCSendFirmwareFile(HANDLEDMC hdmc, PSZ pszFileName);
Update the controller's firmware (flash EEPROM).

hdmc Handle to the Galil controller.
pszFileName File name for the special conversion file.

extern LONG FAR GALILCALL DMCWaitForMotionComplete(HANDLEDMC hdmc, PSZ pszAxes);
Wait for motion complete by querying the controller. The function returns when motion is complete.

hdmc Handle to the Galil controller.
pszAxes Which axes to wait for: X, Y, Z, W, E, F, G, H, or S for coordinated motion. To wait for more than one axis (other than coordinated motion), simply concatenate the axis letters in the string.

extern LONG FAR GALILCALL DMCEnumPCIBusControllers(PCONTROLLERINFO pcontrollerinfo, PUSHORT pusCount);
Enumerate or list all the Galil PCI bus controllers installed in the system. The user needs to make two calls to this function. The first call should have a NULL for the argument pcontrollerinfo. The number of CONTROLLERINFO structs (number of Galil PCI bus controllers found) will be returned in the argument pusCount. The second call should have sufficient memory allocated for all the CONTROLLERINFO structs to be returned and pass the pointer to that memory as the argument pcontrollerinfo. It is the users responsibility to allocate and free memory to hold the CONTROLLERINFO structs.

pusCount Pointer to the number of CONTROLLERINFO structs returned. Users should declare a variable of type USHORT and pass the

address of the variable to the function.

pcontrollerinfo Pointer to one or more CONTROLLERINFO structs. The user must allocate and free memory for this function.

extern LONG FAR GALILCALL DMCGetPCIBusController(PCONTROLLERINFO pcontrollerinfo);

Get the controller information for a specific PCI bus controller. Use the ulSerialNumber field to denote which controller you want the information for. If you set the ulSerialNumber field to 0, the function will retrieve the information for the first controller it finds on the PCI bus. That will usually be the one in the lowest bus number/slot number configuration.

pcontrollerinfo Galil controller information. Users should declare a variable of type CONTROLLERINFO, fill-in the ulSerialNumber, and pass the address of the variable to the function.