EDR Enhanced API

Software Development Kit Programmers Manual for

EDR Enhanced DLL, COM and ActiveX Application Programming Interface (API).

Eagle Technology – Cape Town, South Africa

www.eagle.co.za

Software Development Kit

Data Acquisition and Process Control

© Eagle Technology
31-35 Hout Street • Cape Town • South Africa
Phone +27 21 423 4943 • Fax +27 21 424 4637
Email eagle@eagle.co.za

Product Warranty
Eagle Technology, South Africa, warrants its products/softmare from defect in material and workmanship from confirmed date of purchase for a period of one year if the conditions listed below are met. The product warranty will call the Eagle Technology Data Acquisition Device short as **ETDAQD**.

- The warranty does not apply to an **ETDAQD** that has been previously repaired, altered, extended by any other company or individual outside the premises of Eagle Technology.
- That a qualified person configure and install the **ETDAQD**, and damages caused to a device during installation shall make the warranty void and null.
- The warranty will not apply to conditions where the **ETDAQD** has been operated in a manner exceeding its specifications.

Eagle Technology, South Africa, does not take responsibility or liability of consequential damages, project delays, damaging of equipment or capital loss as a result of its products.

Eagle Technology, South Africa, holds the option and final decision to repair or replace any **ETDAQD.** Proof of purchase must be supplied when requesting a repair.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1 Introduction

EDR Enhanced (EDRE) is a powerful application program interface (API) between your data acquisition and control application and Eagle Technologies line of plug in boards for PCs. It is a software development kit (SDK) designed to simplify the programming of the Eagle data acquisition cards but not sacrificing any power in its functionality. An extension to EDRE is EDR Enhanced ActiveX (EDREX). EDREX is making use of COM technology and supports all 32-bit Windows programming models. EDREX has only got a few COM methods and properties to learn.

1.1 Architecture Overview

The EDRE API consists of three layers of software, Namely the device driver, DLL and ActiveX API. Each layer has got unique futures and is designed to perform a specific task.

At the lowest level you will find a device driver that is specific for each operating system. The device drivers are easy to install and support Plug and Play. A control panel applet supplies information of the current hardware that is installed. General information is also available like serial number, manufacturing date, etc.

The middle layer is implemented in a Windows dynamic link library and contains a lot of the intelligence to isolate the difficulty of communicating to the driver from the COM control or application. The DLL is platform independent and contains a database of all the drivers that is currently supported.

The third layer of software, which also serves as the API, is implemented in various ActiveX controls. Each type of control is connected to a different sub-system, A/D, D/A, Counter-Timer, Digital I/O and a utility control.

EDREX is supplied with extensive examples for all major Windows programming languages.

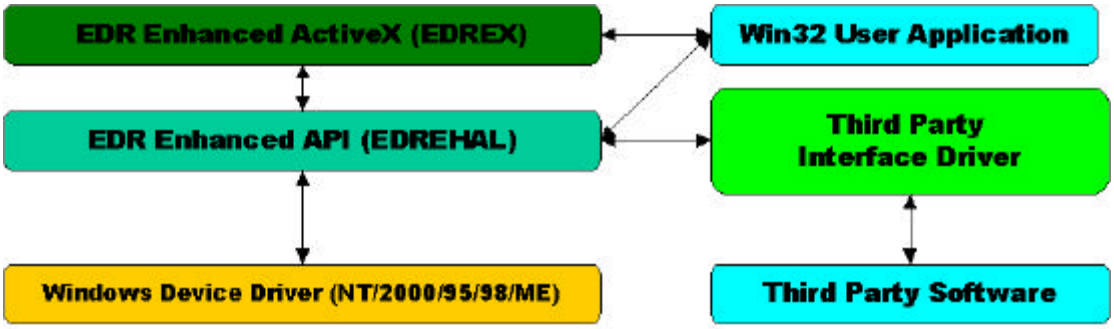


Figure 1-1 EDR Enhanced Architecture

1.2 Supported Operating Systems

The EDR Enhanced SDK supports the following operating systems.

- Windows 95.
- Windows 98.
- Windows 98 SE
- Windows 2000 Professional Edition
- Windows 2000 Server Edition
- Windows ME.
- Windows XP Home Edition
- Windows XP Professional Edition

1.3 Features

- Common API for all Windows platforms.
- Analog input single read or channel list scanning.
- Analog output single write or waveform generation.
- Digital I/O functions.
- Counter timer functions.
- Streaming to disk.
- Build in temperature conversion functions.

1.4 Installation – EDREAPI.EXE

The installation program for EDR Enhanced can be found on the Eagle Technology CD-Rom or can be downloaded off the Eagle Technology website. The installation file be found in the <CDROM>:\EDRE\API directory.

1.5 Contact Details

Below are the contact details of Eagle Technology.

Eagle Technology
PO Box 4376
Cape Town
8000
South Africa
Telephone +27 (021) 423 4943
Fax +27 (021) 424 4637
E-Mail eagle@eagle.co.za
Website http://www.eagle.co.za

2 Windows DLL Exported Functions

The chapter **Windows DLL Exported Functions** discusses the various functions that are exported by the EDR Enhanced Windows 32-bit DLL, **EDRAPI.DLL**. The chapter is broken up in sections dealing with each hardware sub-system.

2.1 The Utility Functions

2.1.1 EDRE_Query

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
QueryCode	32-bit unsigned integer	Query code. See appendix for a list of query code.
Param	32-bit unsigned integer	Extra query parameter.
Return	32-bit integer	Indicates success or failure

EDRE_Query is used to for many functions; to the status of a board, driver version, DLL version and many more functions.

2.2 The Digital Input/Output Functions

2.2.1 EDRE_DioWrite

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Port	32-bit unsigned integer	Port number
Value	32-bit unsigned integer	Value to write to port
Return	32-bit integer	Indicates success or failure

EDRE_DioWrite is used to write digital value to a specific output port. If the port is configurable, the driver will automatically configure the port for output. The returned error code will indicate if the operation was successful.

2.2.2 EDRE_DioRead

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Port	32-bit unsigned integer	Port number
*Value	Pointer to a 32-bit unsigned integer	Value to read from port
Return	32-bit integer	Indicates success or failure

EDRE_DioRead is used to read a digital value from a specific digital input port. If the port is configurable, the driver will configure it for input. The returned error code will indicate if the operation was successful.

2.2.3 EDRE_MioConfig

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Port	32-bit unsigned integer	Port number
Value	32-bit unsigned integer	Value to write to port
Return	32-bit integer	Indicates success or failure

EDRE_DioConfig is used to configure multiplexed digital I/O ports. The returned error code will indicate if the operation was successful.

2.3 The Counter/Timer Functions

2.3.1 EDRE_CTWrite

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Ct	32-bit unsigned integer	Counter number
Value	32-bit unsigned integer	Value to write to counter
Return	32-bit integer	Indicates success or failure

EDRE_CtWrite is used to write a value to a counter timer. The returned error code will indicate if the operation was successful.

2.3.2 EDRE_CTRead

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Ct	32-bit unsigned integer	Counter number
*Value	Pointer to a 32-bit unsigned integer	Value to read from counter
Return	32-bit integer	Indicates success or failure

EDRE_CtRead is used to read a value from a counter. The returned error code will indicate if the operation was successful.

2.3.3 EDRE_CTConfig

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Ct	32-bit unsigned integer	Counter number
Mode	32-bit unsigned integer	Counter mode
Type	32-bit unsigned integer	Counter type
ClkSrc	32-bit unsigned integer	Clock source
GateSrc	32-bit unsigned integer	Gate source
Return	32-bit integer	Indicates success or failure

EDRE_CtConfig is used to configure a counter timer. The returned error code will indicate if the operation was successful.

2.3.4 EDRE_CTSoftGate

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Ct	32-bit unsigned integer	Counter number
Gate	32-bit unsigned integer	Gate value
Return	32-bit integer	Indicates success or failure

EDRE_CtConfig is used to control the software gate, if configured for software gating. The returned error code will indicate if the operation was successful.

2.4 The Interrupt Functions

2.4.1 EDRE_IntEnable

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Return	32-bit integer	Indicates success or failure

EDRE_IntEnable is used to connect the device interrupt system to the bus. A returned error code will indicate if the operation was successful.

2.4.2 EDRE_IntDisable

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Return	32-bit integer	Indicates success or failure

EDRE_IntDisable is used to disconnect the device interrupt system from the bus. A returned error code will indicate if the operation was successful.

2.4.3 EDRE_IntConfigure

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Src	32-bit unsigned integer	Interrupt source
Mode	32-bit unsigned integer	Interrupt mode
Type	32-bit unsigned integer	Interrupt type
Return	32-bit integer	Indicates success or failure

EDRE_IntConfigure is used to configure the interrupt system. This function is only used where interrupts needs to be transferred to user space. A returned error code will indicate if the operation was successful.

2.4.4 EDRE_WaitOnInterrupt

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Return	32-bit integer	Indicates success or failure

EDRE_WaitOnInterrupt is used to wait for an interrupt to occur. The function makes use of overlapped I/O, so it will hang until an interrupt occur. If it needs to be release early use the interrupt release function. The returned code will indicate which interrupt was triggered. Please consult the hardware manual for the specific equipment.

2.4.5 EDRE_ReleaseInterrupt

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Return	32-bit integer	Indicates success or failure

EDRE_ReleaseInterrupt is used to release a call on ***EDRE_WaitOnInterrupt***. A returned error code will indicate if the operation was successful.

2.5 The Analog Output Functions

2.5.1 EDRE_DAWrite

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Channel	32-bit unsigned integer	DAC channel
uVoltage	32-bit integer	Microvoltage
Return	32-bit integer	Indicates success or failure

EDRE_DAWrite is used to output a voltage on a DAC channel. A returned error code will indicate if the operation was successful.

2.5.2 EDRE_DAConfig

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Channel	32-bit unsigned integer	DAC channel
Frequency	32-bit unsigned integer	Update frequency
ClkSrc	32-bit unsigned integer	Clock source
GateSrc	32-bit unsigned integer	Gate source
Continuous	32-bit unsigned integer	DAC mode
Length	32-bit unsigned integer	Size in samples of buffer
*uVoltage	Pointer to a 32-bit integer buffer	Microvoltage buffer
Return	32-bit integer	Indicates success or failure

EDRE_DAConfig is used to configure the ADC system. If the hardware supports independent, block or combined channels, the hardware will be configured in that way. Meaning the implementation of the channel parameter might change according to the hardware. Please the hardware manual for a particular piece of equipment. The entire configuration is done in one function call. The initial data must also be supplied. The returned error code will indicate if any errors occurred.

2.5.3 EDRE_DAControl

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Channel	32-bit unsigned integer	DAC channel
Command	32-bit unsigned integer	Command code See appendix for a list of command codes
Return	32-bit integer	Indicates success or failure

EDRE_DAControl is used to control a specific, block or combined set of channels depending on the hardware. A returned error code will indicate if the operation was successful.

2.5.4 EDRE_DAUpdateData

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Channel	32-bit unsigned integer	DAC channel
Length	32-bit unsigned integer	Size in samples of buffer
*uVoltage	Pointer to a 32-bit integer buffer	Microvoltage buffer
Return	32-bit integer	Indicates success or failure

EDRE_DAUpdateDatal is used to update the driver buffer for a specific or group of channels. A query can be made to check the space available in the driver buffer. Also query for any errors that might have occurred A returned error code will indicate if the operation was successful.

2.6 The Analog Input Functions

This section deals with functions relating to analog input functions.

2.6.1 EDRE_ADSingle

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Channel	32-bit unsigned integer	ADC channel.
Gain	32-bit unsigned integer	ADC gain.
Range	32-bit unsigned integer	ADC range.

*uVoltage	Variable to hold microvolt value	Converted voltage
Return	32-bit integer	Indicates success or failure

EDRE_ADSingle is used to read a single ADC channel with specific settings specified by *Gain* and *Range.* The method used to read the voltage is programmed I/O. An error code is return to indicate the successfulness of the called function.

2.6.2 EDRE_ADConfig

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
*Frequency	Pointer to a 32-bit unsigned integer	Sampling frequency
ClkSrc	32-bit unsigned integer	ADC clock source.
Burst	32-bit unsigned integer	0: Disable burst mode 1: Enable burst mode
Range	32-bit unsigned integer	ADC range
*ChanList	Pointer to an array of 32-bit unsigned integers	ADC channel list
*GainList	Pointer to an array of 32-bit unsigned integers	ADC gain list
ListSize	32-bit unsigned integer	Size of list.
Return	32-bit integer	Indicates success or failure

EDRE_ADConfig is used to configure the ADC sub-system for channel list mode. When called the hardware gets configured and ready to start scanning through the channel list. The use of the parameters is different for each series of boards. Make sure that you consult the specific hardware manual for the card that is being programmed. The returned error code will indicate success or failure. Also don't try and configure the board while it is busy. This will also result in a failed configuration call.

2.6.3 EDRE_ADStart

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Return	32-bit integer	Indicates success or failure

EDRE_ADStart is used to start the ADC channel list process. The board will be enabled to start running through the channel list. Before the operation can be started, the board must first be configured. If not the operation will fail.

2.6.4 EDRE_ADStop

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Return	32-bit integer	Indicates success or failure

EDRE_ADStop is used to stop the ADC process.

2.6.5 EDRE_ADGetData

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
*Buf	Pointer to an array of 32-bit signed integers	Buffer to hold microvolt values returned.
*BufSize	Pointer to an array of 32-bit unsigned integers	Variable to hold the value of requested samples and the actual number of samples returned.
Return	32-bit integer	Indicates success or failure

EDRE_ADGetData is used to retrieve data from the driver buffer when sampling data. Use ***EDRE_Query*** to check if any data is available. The query code must *ADUNREAD*. The buffer size parameter must contain the number of samples requested. However the driver can change this number. So even less samples can be returned, but not more. Make sure to

check this number when called. A returned error code will indicate if the function call was successful. Please note that the buffer contains data in microvolts. The buffer is organized in the same order as the channel list.

2.6.6 EDRE_ADGetDataRaw

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
*Buf	Pointer to an array of 16-bit unsigned words	Buffer to hold binary values returned.
*BufSize	Pointer to an array of 32-bit unsigned integers	Variable to hold the value of requested samples and the actual number of samples returned.
Return	32-bit integer	Indicates success or failure

EDRE_ADGetDataRaw is used to retrieve data from the driver buffer when sampling data. Use ***EDRE_Query*** to check if any data is available. The query code must *ADUNREAD*. The buffer size parameter must contain the number of samples requested. However the driver can change this number. So even less samples can be returned, but not more. Make sure to check this number when called. A returned error code will indicate if the function call was successful. . Please note that the buffer contains data in raw binary format. The buffer is organized in the same order as the channel list.

2.6.7 EDRE_ADOpenStreamFile

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Mode	32-bit unsigned integer	File Mode 0: Overwrite 1: Append
*Char	Pointer to an array of characters or string.	Specify file name and full path of streaming file.
Return	32-bit integer	Indicates success or failure

EDRE_ADOpenStreamFile is used to stream data to file. If a streaming file is opened and ***EDRE_ADGetData(Raw)*** get called, all data supplied by the driver gets written to the specified file.

2.6.8 EDRE_ADCloseStreamFile

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.

EDRE_ADCloseStreamFile is used to close and save a file that was opened by ***EDRE_ADOpenStreamFile.***

2.6.9 EDRE_ADGetDataFromFile

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Start	32-bit unsigned integer	Start position in file.
*Buf	Pointer to an array of 32-bit signed integers	Buffer to hold microvolt values returned.
*BufSize	Pointer to an array of 32-bit unsigned integers	Variable to hold the value of requested samples and the actual number of samples returned.
Return	32-bit integer	Indicates success or failure

EDRE_ADGetDataFromFile is used to retrieve data from a file. The starting position and number of samples can be specified. The *BufSize* parameter will specify the number of actual samples copied to the buffer.

2.6.10 EDRE_ADGetDataRawFromFile

Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
Start	32-bit unsigned integer	Start position in file.
*Buf	Pointer to an array of 16-bit unsigned words	Buffer to hold binary values returned.
*BufSize	Pointer to an array of 32-bit unsigned integers	Variable to hold the value of requested samples and the actual number of samples returned.
Return	32-bit integer	Indicates success or failure

EDRE_ADGetDataRawFromFile is used to retrieve binary data from a file. The starting position and number of samples can be specified. The *BufSize* parameter will specify the number of actual samples copied to the buffer.

2.7 The Temperature Functions

2.7.1 EDRE_CalcCJCmC

Parameter	Type	Description
cjcuv	32-bit integer	Cold junction compensation channel microvoltage.
Return	32-bit integer	Returns cold junction temp in milli degrees.

EDRE_CalcCJCmC is used to calculate the temperature of the cold junction channel.

2.7.2 EDRE_CalcRTDmC

Parameter	Type	Description
rtduv	32-bit integer	RTD microvolt
Return	32-bit integer	Returns RTD in milli degrees.

EDRE_CalcCJCmC is used to calculate the temperature of a RTD channel.

2.7.3 EDRE_CalcTCmC

Parameter	Type	Description
tctype	32-bit integer	Thermocouple type
tcuv	32-bit integer	Thermocouple temp
ambientmc	32-bit integer	Ambient temp
Return	32-bit integer	Returns thermo temp in milli degrees

EDRE_CalcTCmC is used to calculated the temperature of a thermocouple channel.

2.8 The String Functions

2.8.1 EDRE_StrBoardName

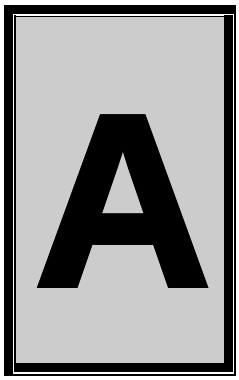
Parameter	Type	Description
Sn	32-bit unsigned integer	Board's serial number.
*StrBoardName	Pointer to a character array or string	Variable to hold a string
Return	32-bit integer	Indicates success or failure

EDRE_StrBoardName is used to retrieve a board's name or text description. The returned error code will indicate if the operation was successful.

2.8.2 EDRE_StrError

Parameter	Type	Description
Error	32-bit unsigned integer	Error number
*StrBoardName	Pointer to a character array or string	Variable to hold a string
Return	32-bit integer	Indicates success or failure

EDRE_StrErrore is used to retrieve a text description of a specific error code. The returned error code will indicate if the operation was successful.



A. Constants

1. Query Codes

Name	Value	Description
APIMAJOR	1	Query EDRE API major version number.
APIMINOR	2	Query EDRE API minor version number.
APIBUILD	3	Query EDRE API build version number.
APIOS	4	Query EDRE API OS type.
APINUMDEV	5	Query number of devices installed.
BRDTYPE	10	Query a board's type.
BRDREV	11	Query a board's revision.
BRDYEAR	12	Query a board's manufactured year.
BRDMONTH	13	Query a board's manufactured month.
BRDDAY	14	Query a board's manufactured day.
BRDSERIALNO	15	Query a board's serial number.
DRVMAJOR	20	Query a driver's major version number.
DRVMINOR	21	Query a driver's minor version number.
DRVBUILD	22	Query a driver's build version number.
ADNUMCHAN	100	Query number of ADC channel.
ADNUMSH	101	Query number of samples-and-hold channels.
ADMAXFREQ	102	Query maximum sampling frequency.
ADBUSY	103	Check if ADC system is busy.
ADFIFOSIZE	104	Get ADC hardware FIFO size.
ADFIFOOVER	105	Check for FIFO overrun condition.
ADBUFFSIZE	106	Check software buffer size.
ADBUFFOVER	107	Check for circular buffer overrun.
ADBUFFALLOC	108	Check if software buffer is allocated.
ADUNREAD	109	Get number of samples available.
ADEXTCLK	110	Get status of external clock line – PCI30FG.
ADEXTTRIG	111	Get status of external trigger line – PCI30FG.
ADBURST	112	Check if burst mode is enabled.
ADRANGE	113	Get ADC range.
DANUMCHAN	200	Query number of DAC channels.
DAMAXFREQ	201	Query maximum DAC output frequency.
DABUSY	202	Check if DAC system is busy.
DAFIFOSZ	203	Get DAC FIFO size.
CTNUM	300	Query number of counter-timer channels.
CTBUSY	301	Check if counter-timer system is busy.
DIONUMPORT	400	Query number of digital I/O ports.
DIOQRYPORT	401	Query a specific port for capabilities.
DIOPORTWIDTH	402	Get a specific port's width.
INTNUMSRC	500	Query number of interrupts sources.
INTSTATUS	501	Queries interrupt system's status.
INTBUSCONNECT	502	Connect interrupt system to bus.
INTISAVAILABLE	503	Check if an interrupt is available.
INTNUMTRIG	504	Check number times interrupted

Table A-1 Query Codes

2. Error Codes

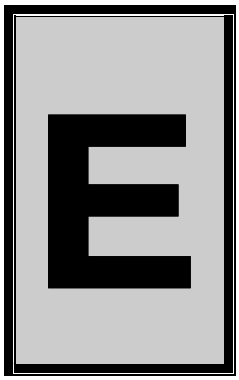
Name	Value	Description
EDRE_OK	0	Function successfully.
EDRE_FAIL	-1	Function call failed.
EDRE_BAD_FN	-2	Invalid function call.
EDRE_BAD_SN	-3	Invalid serial number.
EDRE_BAD_DEVICE	-4	Invalid device.
EDRE_BAD_OS	-5	Function not supported by operating system.
EDRE_EVENT_FAILED	-6	Wait on event failed.
EDRE_EVENT_TIMEOUT	-7	Event timed out.
EDRE_INT_SET	-8	Interrupt in use.
EDRE_DA_BAD_RANGE	-9	DAC value out of range.
EDRE_AD_BAD_CHANLIST	-10	Channel list size out of range.
EDRE_BAD_FREQUECY	-11	Frequency out of range.
EDRE_BAD_BUFFER_SIZE	-12	Data passed by buffer incorrectly sized
EDRE_BAD_PORT	-13	Port value out of range.
EDRE_BAD_PARAMETER	-14	Invalid parameter value specified.
EDRE_BUSY	-15	System busy.
EDRE_IO_FAIL	-16	IO call failed.
EDRE_BAD_ADGAIN	-17	ADC-gain out of range.
EDRE_BAD_QUERY	-18	Query value not supported.
EDRE_BAD_CHAN	-19	Channel number out of range.
EDRE_BAD_VALUE	-20	Configuration value specified out of range.
EDRE_BAD_CT	-21	Counter-timer channel out of range.
EDRE_BAD_CHANLIST	-22	Channel list invalid.
EDRE_BAD_CONFIG	-23	Configuration invalid.
EDRE_BAD_MODE	-24	Mode not valid.
EDRE_HW_ERROR	-25	Hardware error occurred.
EDRE_HW_BUSY	-26	Hardware busy.
EDRE_BAD_BUFFER	-27	Buffer invalid.
EDRE_REG_ERROR	-28	Registry error occurred.
EDRE_OUT_RES	-29	Out of resources.
EDRE_IO_PENDING	-30	Waiting on I/O completion
EDRE_NET_ERROR	-31	Network error
EDRE_FILE_ERROR	-32	File error
EDRE_FILE_OPEN	-33	File already open

Table A-2 EDR Enhanced Error Codes

3. Digital I/O Codes

Name	Value	Description
DIOOUT	0	Port is an output.
DIOIN	1	Port is an input.
DIOINOROUT	2	Port can be configured as in or out.
DIOINANDOUT	3	Port is an input and an output.

Table A-3 Digital I/O Codes



B. Additional Information

For ordering information please contact Eagle Technology directly or visit our website www.eagle.co.za. They can also be emailed at eagle@eagle.co.za.