libK8061

1.1

Generated by Doxygen 1.5.6

Sat Jan 17 09:36:22 2009

Contents

Chapter 1

Main Page

Author:

Bino Maiheu, binomaiheu@gmail.com (c) 2008

This page describes some general information on the libK8061 library, for a detailed description of all the routines, please see the **k8061.h** (p. 9) header file in the file list.

If you have problems with this library or comments, please drop me a line on the above email address.

1.1 Introduction to libK8061

libK8061 is basically a linux - port of the DLL that is provided with the Velleman k8061 USB interface card.

http://www.velleman.be/ot/en/product/view/?id=364910 http://www.velleman.be/downloads/0/infosheet uk.pdf

The library maintains an array of 8 usb filehandles which corresponds to the 8 possible addresses for the k8061 device on the USB bus. It then provides a set of routines which simply read analog/digital input values or set analog/digital output or the PWM value. It also contains an array with the jumper settings for the analog in/outputs so the user can set or read proper voltage levels instead of ADC/DAC values.

The library is distributed with a little command line interface to the card which enables the user to run some test commands to see whether his/her board works properly. This little ui tool is described in **The cmd8061 shell** (p. 3).

1.2 Compilation & installation

The library is based upon the default GNU autotools, so you should not have much difficulty in getting it to compile. Just issue the standard

```
bash$ tar -xvzf k8061-1.0.tar.gz
bash$ cd k8061-1.0
bash$ ./configure --prefix=/where/ever/you/want
bash$ make
bash$ make install
```

The library itself depends on the presence of libusb on your system along with it's development package libusb-devel. This package is included on most sytems and on fedora I've installed it using

```
bash# yum install libusb libusb-devel
```

So check you're own system's package management system (dpkg for Debian and Ubuntu, and I have no idea what SuSE uses nowadays...) Note that the compilation of the cmd8061 tool requires the presence of the GNU readline and history library ! See **cmd8061 compilation** (p. 4).

If you can't find it, feel free to compile libusb from source, you can find it at :

`http://libusb.sourceforge.net/`

1.3 Setting user permissions with udev

Using the USB interface via libusb is normally only possible as root. So one has to set some permissions right to be able to use USB devices as user. There are different ways to do that, but the cleanest is via the system's hotplug sytem. On fedora (my system) this is done via udev.

See `http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev.html`

I will describe what worked for me on a fedora 8 machine. One simply needs to create a new set of udev rules. On my system I have created a new group of users called **daqusers** to group all the ones with permission to write to any USB DAQ device on my system and added myself to that list. On most systems there are loads of graphical tools to do that, but here are the simple commands :

```
bash# groupadd daqusers
bash# usermod -a -G daqusers <username>
```

Next, you should create a udev rules file in

```
bash# vi /etc/udev/rules.d/60-usbdaq.rules
```

with the following contents

```
# Sets up udev rules for USB DAQ devices
SUBSYSTEM!="usb", ACTION!="add", GOTO="usbdaq_end"
# The Velleman k8061 USB board
SYSFS{idVendor}=="10cf", SYSFS{idProduct}=="8061", GROUP="daqusers", MODE="0660"
LABEL="usbdaq_end"
```

I had to log in & out for the system to pick up my new user affiliation, but after that it worked just fine and I could communicate to the Velleman k8061 device as myself.

Chapter 2

The cmd8061 shell

In order to play around a bit with the USB board and easily run some tests to see whether my soldering was done okay, I have created a user interface which enables you to easily send some commands down to the card or read some ADC or DIN values back.

This little tool is distributed along with the library and is called **cmd8061**.

2.1 cmd8061 compilation

cmd8061 Depends on libK8061 and is compiled along with it when the user issues a make in the top level source directory. Next to the libusb dependance, the configure script also checks the presence of the GNU readline and history libraries which are needed for the command line interface. On fedora you can install these with

```
bash# yum install readline readline-devel
```

The homepage for readline is found here :

`http://tiswww.case.edu/php/chet/readline/rltop.html`

2.2 cmd8061 usage

The **cmd8061** tool is fairly straightforward to use. It defined a small number of commands which issue library calls that interface with the k8061 USB board. When you startup the program you get a prompt. The prompt is done with libreadline and libhistory so therefore it has tab-completion as well as a command history the standard GNU-way.

```
[niblap] ~ $ cmd8061
cmd8061 Copyright (C) 2008  Bino Maiheu
A linux based command-line interface for the
Velleman K8061 Extended USB Interface

This program comes with ABSOLUTELY NO WARRANTY;
This is free software, and you are welcome to redistribute it
under conditions set out in the GNU General Public Licence version 2
See: http://www.gnu.org/licenses

Initializing interface...
Initializing k8061 USB devices...
Succesfully found K8061 card nr 0
cmd8061::0>
```

Then you can issue various command to .e.g. read the analog inputs or set the PWM value. when you issue the **help** command you'll see a list of possible commands. Always one card is selected and you can change to a different active USB card with the following command

```
cmd8061::1> card 0
Selected card 0
[Card 0 Status]
connected, power on, version : IC3: VK8061USB Rev: V1.3, IC6: VK8061CPU Rev: V1.5
[Card 0 Jumper Settings]
AIN Jumper Config 1->8: 0 0 0 0 0 0 0 0
AOUT Jumper Config 1->8: 0 0 0 0 0 0 0 0
cmd8061::0>
```

The program will output some status information of the selected card and update the number after the "cmd8061::" prompt, indicating the active card number. All commands that are issued, are done for the active card. Except of course the once that act upon the entire usb bus, like **scan** or **list**. The brief explanation should be enough for most commands in the help function, we will discuss some important commands below. To quit the program, the command is simply **quit**.

This is an overview of all the commands :

```
scan                            Scan usb bus and update k8061 array..
list                            List of known k8061 devices on USB bus.
stat                            Show status of the active k8061 card..
card    <icard>                 Select an active k8061 card on USB bus [0-7]..
get     <grp> [ch]              Get value of channel of group.
set     <grp> <ch> <v> [V]      Set value of channel of group to v, [AOUT in Volt].
reset                           Resets the board, output values to 0.
clr                             Clear read errors.
err                             Show read errors.
jmps    <grp> <hex>             Set jumper values for analog in/out groups.
wait    <secs>                  Wait for a number of seconds..
exec    <fname>                 Execute the file with commands.
quit                            Quit this program..
help                            Display help text on the commands.
info                            Display some usefull info about the program.
```

2.2.1 The get and set commands.

The get and set commands are used to control the different interface groups. See them with info :

```
cmd8061::0> info
Groups on the board are :
- dout ..... : digital outputs
- aout ..... : analog outputs
- din ...... : digital inputs
- ain ...... : analog inputs
- pwm ...... : pulser
```

In order e.g. to get the value of analog input 3 on the active card, simply issue

```
cmd8061::0> get ain 3
```

and the program will display the ADC value as an integer (and for the analog in/outputs) as a voltage value as well.

To display all the analog inputs, use

```
cmd8061::0> get ain
```

Identical behaviour exists for the digital inputs.

In order to set digital or analog outputs, use

```
cmd8061::0> set dout 2 0
cmd8061::0> set dout 3 1
```

This sets the digital output on channel 2 to low on the active card and the digital output on channel 3 to high.

If you want to set all the digital outputs to 1 of the active card use

```
cmd8061::0> set dout all 1
```

similarly you can set all the analog outputs to a value or individual analog outputs.

```
cmd8061::0> set aout 3 123
cmd8061::0> set aout all 234
```

Attention:

When setting an anlog output, you have the option of providing a voltage value instead of an integer DAC value. You can do this by adding a "V" to the command, like this :

```
cmd8061::0> set aout 3 0.87 V
cmd8061::0> set aout all 3.50 V
```

By issuing a get commmand on the dout, aout or pwm, the set values are read back and displayed.

2.2.2 The jmps command.

The jmps command sets the analog input/output jumpers, simply provide the group name (**ain** or **aout**) followed by a bitpattern which corresponds to the jumper settings, so AIN1 is bit 0, AIN8 bit 7, 1 means the jumper is there, 0 means it is not. An example is given here

```
cmd8061::0> jmps ain 0x0f
```

This tells the program and the library that the first 4 jumpers on the analog ADC inputs are there (AIN1->4), and on the 4 other ones (AIN5->8) not.

2.2.3 Running a macro

The command **exec** enables the user to run a file that has a set of commands. One usefull command with this is the **wait** command that pauses for the number of seconds given. In the distribution tarballs test/ directory there is a file called *dout-runoff.k8061* which contains such a macro to try.

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

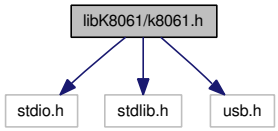
File Documentation

4.1 libK8061/k8061.h File Reference

4.1.1 Detailed Description

```
#include <stdio.h>
#include <stdlib.h>
#include <usb.h>
```

Include dependency graph for k8061.h:



Defines

- #define **k8061_idVendor** 0x10cf
  k8061 Vendor ID
- #define **k8061_idProduct** 0x8061
  k8061 Product ID
- #define **k8061_epRead** 0x81
  Value of k8061 read end-point.
- #define **k8061_epWrite** 0x01
  Value of k8061 write end-point.
- #define **k8061_ioTimeOut** 50
  timeout for usb_bulk_read/write calls
- #define **k8061_ioWait** 800

wait after usb_bulk_read/write calls

- #define **k8061_iConfig** 1
  usb config to use on card
- #define **k8061_iClaim** 0
  interface to claim
- #define **k8061_cmdReadAnalogChannel** 0x00
  k8061 Command: Read analog channel
- #define **k8061_cmdReadAllAnalog** 0x01
  k8061 Command: Read all analog channels
- #define **k8061_cmdSetAnalogChannel** 0x02
  k8061 Command: Set analog channel
- #define **k8061_cmdSetAllAnalog** 0x03
  k8061 Command: Set all analog channels
- #define **k8061_cmdOutputPWM** 0x04
  k8061 Command: Output PWM level
- #define **k8061_cmdReadDigitalByte** 0x05
  k8061 Command: Read the digital byte
- #define **k8061_cmdDigitalOut** 0x06
  k8061 Command: Set Digital output
- #define **k8061_cmdClearDigitalChannel** 0x07
  k8061 Command: Set a signal digital channel
- #define **k8061_cmdSetDigitalChannel** 0x08
  k8061 Command: Set a signal digital channel
- #define **k8061_cmdReadCounters** 0x09
  k8061 Command: Read the number of errors
- #define **k8061_cmdResetCounters** 0x0a
  k8061 Command: Reset the number of errors
- #define **k8061_cmdReadVersion** 0x0b
  k8061 Command: Read version
- #define **k8061_cmdJumpers** 0x0c
  k8061 Command: Get Jumper status
- #define **k8061_cmdPowerStatus** 0x0d
  k8061 Command: Power status

- #define **k8061_cmdReadDigitalOut** 0x0e
  k8061 Command: Read back digital out

- #define **k8061_cmdReadAnalogOut** 0x0f
  k8061 Command: Read back analog out

- #define **k8061_cmdReadPWMOut** 0x10
  k8061 Command: Read back PWM out

Functions

- void **k8061_Init** (void)
  Initialises the library by intialising the readline interface and the usb library.

- int **k8061_GetDeviceCount** (void)
  Scans for and returns the number of Velleman K8061 devices on the USB bus.

- int **k8061_OpenDevices** (void)
  See k8061_ScanBus.

- int **k8061_ScanBus** (int verb)
  Scans the complete usb bus hierarcy and looks for k8061 devices.

- void **k8061_CloseDevices** (void)
  Closes all k8061 devices that are currently open in the file handle list.

- int **k8061_ExecIO** (int iCard, int nSend, int nReceive)
  Performs a basic input / output operation to the k8061 device, this consists of sending a command defined my the _k8061_WriteBuffer of nSend bytes and reading back the result of the command in the _k8061_ReadBuffer of nReceive bytes.

- void **k8061_CheckOpen** (int iCard)
  Checks whether the card number really is open by performing a simple read & write to the device.

- void **k8061_ListDevices** (FILE *fp)
  Writes the status of all the devices that are currently open in the filehandle array.

- int **k8061_OutputAnalogChannel** (int iCard, int iChan, int value)
  Sets the analog output on channel iChan of card iCard.

- int **k8061_OutputAnalogChannelVolt** (int iCard, int iChan, double volt)
  Sets the analog output on channel iChan of card iCard to the voltage value.

- int **k8061_OutputAllAnalog** (int iCard, int value_arr[])
  Sets all analog output channels on card iCard to the values given by the array of DAC values.

- int **k8061_OutputAllAnalogVolt** (int iCard, double volt_arr[])
  Sets all analog output on channel on card iCard to the voltage values given by the array.

- int **k8061_ClearAnalogChannel** (int iCard, int iChan)
  Sets the analog output on channel iChan of card iCard to 0.
- int **k8061_ClearAllAnalog** (int iCard)
  Sets all the analog output on channels of card iCard to 0.
- int **k8061_SetAnalogChannel** (int iCard, int iChan)
  Sets the analog output on channel iChan of card iCard to its maximum (255).
- int **k8061_SetAllAnalog** (int iCard)
  Sets all the analog output on channels on card iCard to their maximum (255).
- int **k8061_ReadAnalogChannel** (int iCard, int iChan)
  Reads the value of the analog input channel.
- int **k8061_ReadAllAnalog** (int iCard, int value_arr[])
  Reads the value of all analog input channels on the card and returns them in the array.
- double **k8061_GetDACVolt** (int iCard, int chan, int val)
  Converts the DAC value (Analog Output) into volts using the stored DAC jumper configuration.
- double **k8061_GetADCVolt** (int iCard, int chan, int val)
  Converts the ADC value (Analog Input) into volts using the stored DAC jumper configuration.
- int **k8061_SetDACJumpers** (int iCard, unsigned char b)
  Sets the Analog Output jumper configuration on the card to the given pattern b encoded as a single byte.
- int **k8061_SetADCJumpers** (int iCard, unsigned char b)
  Sets the Analog Input jumper configuration on the card to the given pattern b encoded as a single byte.
- int **k8061_WriteJumpers** (int iCard, FILE *fp)
  Write the status of the ADC/DAC jumpers to the filepointer.
- int **k8061_SetAllDigital** (int iCard)
  Sets all digital outputs on the card to 1.
- int **k8061_ClearAllDigital** (int iCard)
  Sets all digital outputs on the card to 0.
- int **k8061_SetDigitalChannel** (int iCard, int iChan)
  Sets the digital output channel to 1.
- int **k8061_ClearDigitalChannel** (int iCard, int iChan)
  Sets the digital output channel to 0.
- int **k8061_OutputAllDigital** (int iCard, unsigned char bPattern)
  Set all digital output channels on the card according to some binary pattern, encoded as a single byte.

- int **k8061_ReadAllDigital** (int iCard, int din_arr[])
  Reads the all the digital inputs and stores them as 8 integer 0/1 values in the array.
- int **k8061_ReadAllDigitalByte** (int iCard, unsigned char *din_byte)
  Reads the all the digital inputs and returns them as a single byte (unsigned char).
- int **k8061_ReadDigitalChannel** (int iCard, int iChan)
  Reads a single digital channel and returns it's state as 0 or 1 unsigned char).
- int **k8061_OutputPWM** (int iCard, int value)
  Sets the PWM value, between 0 and 1024.
- int **k8061_WriteStatus** (int iCard, FILE *fp)
  Writes some status report of the card to the filepointer, this shows whether the card is connected, whether the 12 VDC power cord is connected and what the version of the IC firmware is.
- int **k8061_ReadVersion** (int iCard, char version_str[])
  Reads back the version of the IC firmware.
- int **k8061_IsConnected** (int iCard)
  Returns the _k8061_IsOpen value for the corresponding card.
- int **k8061_HasPower** (int iCard)
  Checks whether the card's 12 VDC power supply is ok.
- int **k8061_ReadBackDigitalOut** (int iCard, int dout_arr[])
  Reads back the digital outputs and stores them ad 8 integer ones or zeros in the dout array.
- int **k8061_ReadBackAnalogOut** (int iCard, int aout_arr[])
  Reads back the analog outputs and stores them ad 8 integer valuess in the aout array.
- int **k8061_ReadBackPWMOut** (int iCard)
  Reads back the set PWM value and returs it.
- int **k8061_ResetCounters** (int iCard)
  Resets the error counters.
- int **k8061_ReadCounters** (int iCard, int cntr_arr[])
  Reads the error counters.

Variables

- int **_k8061_IsOpen** [8]
  Array with flags for the open k8061 devices.
- usb_dev_handle * **_k8061_Handle** [8]
  Array with filehandles for the k8061 devices.

- char **_k8061_ReadBuffer** [50]
  The read buffer.
- char **_k8061_WriteBuffer** [50]
  The receive buffer.
- unsigned char **_k8061_jmp_adc** [8]
  Jumper settings for the AINs for each card.
- unsigned char **_k8061_jmp_dac** [8]
  Jumper settings for the AOUTs for each card.

4.1.2 Define Documentation

4.1.2.1 #define k8061_cmdClearDigitalChannel 0x07

k8061 Command: Set a signal digital channel

4.1.2.2 #define k8061_cmdDigitalOut 0x06

k8061 Command: Set Digital output

4.1.2.3 #define k8061_cmdJumpers 0x0c

k8061 Command: Get Jumper status

4.1.2.4 #define k8061_cmdOutputPWM 0x04

k8061 Command: Output PWM level

4.1.2.5 #define k8061_cmdPowerStatus 0x0d

k8061 Command: Power status

4.1.2.6 #define k8061_cmdReadAllAnalog 0x01

k8061 Command: Read all analog channels

4.1.2.7 #define k8061_cmdReadAnalogChannel 0x00

k8061 Command: Read analog channel

4.1.2.8 #define k8061_cmdReadAnalogOut 0x0f

k8061 Command: Read back analog out

4.1.2.9 #define k8061_cmdReadCounters 0x09

k8061 Command: Read the number of errors

4.1.2.10 #define k8061_cmdReadDigitalByte 0x05

k8061 Command: Read the digital byte

4.1.2.11 #define k8061_cmdReadDigitalOut 0x0e

k8061 Command: Read back digital out

4.1.2.12 #define k8061_cmdReadPWMOut 0x10

k8061 Command: Read back PWM out

4.1.2.13 #define k8061_cmdReadVersion 0x0b

k8061 Command: Read version

4.1.2.14 #define k8061_cmdResetCounters 0x0a

k8061 Command: Reset the number of errors

4.1.2.15 #define k8061_cmdSetAllAnalog 0x03

k8061 Command: Set all analog channels

4.1.2.16 #define k8061_cmdSetAnalogChannel 0x02

k8061 Command: Set analog channel

4.1.2.17 #define k8061_cmdSetDigitalChannel 0x08

k8061 Command: Set a signal digital channel

4.1.2.18 #define k8061_epRead 0x81

Value of k8061 read end-point.

4.1.2.19 #define k8061_epWrite 0x01

Value of k8061 write end-point.

4.1.2.20 #define k8061_iClaim 0

interface to claim

4.1.2.21 #define k8061_iConfig 1

usb config to use on card

4.1.2.22 #define k8061_idProduct 0x8061

k8061 Product ID

4.1.2.23 #define k8061_idVendor 0x10cf

k8061 Vendor ID

4.1.2.24 #define k8061_ioTimeOut 50

timeout for usb_bulk_read/write calls

4.1.2.25 #define k8061_ioWait 800

wait after usb_bulk_read/write calls

4.1.3 Function Documentation

4.1.3.1 void k8061_CheckOpen (int *iCard*)

Checks whether the card number really is open by performing a simple read & write to the device. If it failes, the filehandle array is updated.

Parameters:

iCard The number (address) of the k8061 card on the bus

4.1.3.2 int k8061_ClearAllAnalog (int *iCard*)

Sets all the analog output on channels of card iCard to 0.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

0 upon success, < 0 upon failure

4.1.3.3 int k8061_ClearAllDigital (int *iCard*)

Sets all digital outputs on the card to 0.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

0 upon success, < 0 upon failure

4.1.3.4 int k8061_ClearAnalogChannel (int *iCard*, int *iChan*)

Sets the analog output on channel iChan of card iCard to 0.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

iChan The channel number on the card [1-8]

Returns:

0 upon success, < 0 upon failure

4.1.3.5 int k8061_ClearDigitalChannel (int *iCard*, int *iChan*)

Sets the digital output channel to 0.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

iChan The channel number on the card [1-8]

Returns:

0 upon success, < 0 upon failure

4.1.3.6 void k8061_CloseDevices (void)

Closes all k8061 devices that are currently open in the file handle list.

4.1.3.7 int k8061_ExecIO (int *iCard*, int *nSend*, int *nReceive*)

Performs a basic input / output operation to the k8061 device, this consists of sending a command defined my the _k8061_WriteBuffer of nSend bytes and reading back the result of the command in the _k8061_ReadBuffer of nReceive bytes.

Parameters:

iCard The number (address) of the k8061 card on the bus

nSend Number of bytes to send to the k8061 device

nReceive Number of bytes to read back from the k8061 device

Returns:

0 upon success, < 0 upon failure

4.1.3.8 double k8061_GetADCVolt (int *iCard*, int *chan*, int *val*)

Converts the ADC value (Analog Input) into volts using the stored DAC jumper configuration.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]
chan The channel number on the card [1-8]
val The integer ADC value

Returns:

The ADC value in volts

4.1.3.9 double k8061_GetDACVolt (int *iCard*, int *chan*, int *val*)

Converts the DAC value (Analog Output) into volts using the stored DAC jumper configuration.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]
chan The channel number on the card [1-8]
val The integer DAC value

Returns:

The DAC value in volts

4.1.3.10 int k8061_GetDeviceCount (void)

Scans for and returns the number of Velleman K8061 devices on the USB bus.

Returns:

The number of K8061 devices found on the USB bus

4.1.3.11 int k8061_HasPower (int *iCard*)

Checks whether the card's 12 VDC power supply is ok.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

1 if it is, 0 if not

4.1.3.12 void k8061_Init (void)

Initialises the library by intialising the readline interface and the usb library.

4.1.3.13 int k8061_IsConnected (int *iCard*)

Returns the _k8061_IsOpen value for the corresponding card.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

Whether the card is connected in the filehandle array or not

4.1.3.14 void k8061_ListDevices (FILE * *fp*)

Writes the status of all the devices that are currently open in the filehandle array.

Parameters:

fp Output filepointer, use stdout for terminal

4.1.3.15 int k8061_OpenDevices (void)

See k8061_ScanBus.

4.1.3.16 int k8061_OutputAllAnalog (int *iCard*, int *value_arr*[])

Sets all analog output channels on card iCard to the values given by the array of DAC values.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

value_arr An array of 8 the integer DAC values

Returns:

0 upon success, < 0 upon failure

4.1.3.17 int k8061_OutputAllAnalogVolt (int *iCard*, double *volt_arr*[])

Sets all analog output on channel on card iCard to the voltage values given by the array.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

volt_arr An array of 8 voltage values (< 5 V or < 10 V, depending on the jumper)

Returns:

0 upon success, < 0 upon failure

4.1.3.18 int k8061_OutputAllDigital (int *iCard*, unsigned char *bPattern*)

Set all digital output channels on the card according to some binary pattern, encoded as a single byte.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]
bPattern The 8-bit pattern for the digital outputs

Returns:

0 upon success, < 0 upon failure

4.1.3.19 int k8061_OutputAnalogChannel (int *iCard*, int *iChan*, int *value*)

Sets the analog output on channel iChan of card iCard.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]
iChan The channel number on the card [1-8]
value The integer DAC value

Returns:

0 upon success, < 0 upon failure

4.1.3.20 int k8061_OutputAnalogChannelVolt (int *iCard*, int *iChan*, double *volt*)

Sets the analog output on channel iChan of card iCard to the voltage value.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]
iChan The channel number on the card [1-8]
volt A voltage value (< 5 V or < 10 V, depending on the jumper)

Returns:

0 upon success, < 0 upon failure

4.1.3.21 int k8061_OutputPWM (int *iCard*, int *value*)

Sets the PWM value, between 0 and 1024.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]
value The PWM value

Returns:

0 upon success, < 0 upon failure

4.1.3.22 int k8061_ReadAllAnalog (int *iCard*, int *value_arr*[ ])

Reads the value of all analog input channels on the card and returns them in the array.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

value_arr The array with the ADC values

Returns:

0 upon success, < 0 upon failure

4.1.3.23 int k8061_ReadAllDigital (int *iCard*, int *din_arr*[ ])

Reads the all the digital inputs and stores them as 8 integer 0/1 values in the array.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

din_arr Array with 0 or 1 corresponding to the digital inputs

Returns:

0 upon success, < 0 upon failure

4.1.3.24 int k8061_ReadAllDigitalByte (int *iCard*, unsigned char * *din_byte*)

Reads the all the digital inputs and returns them as a single byte (unsigned char).

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

din_byte Byte formed out of the 8 digital inputs

Returns:

0 upon success, < 0 upon failure

4.1.3.25 int k8061_ReadAnalogChannel (int *iCard*, int *iChan*)

Reads the value of the analog input channel.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

iChan The channel number on the card [1-8]

Returns:

The ADC value

4.1.3.26 int k8061_ReadBackAnalogOut (int *iCard*, int *aout_arr*[ ])

Reads back the analog outputs and stores them ad 8 integer valuess in the aout array.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

aout_arr Array to store the analog outputs

Returns:

0 upon success, < 0 upon failure

4.1.3.27 int k8061_ReadBackDigitalOut (int *iCard*, int *dout_arr*[ ])

Reads back the digital outputs and stores them ad 8 integer ones or zeros in the dout array.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

dout_arr Array to store the digital outputs

Returns:

0 upon success, < 0 upon failure

4.1.3.28 int k8061_ReadBackPWMOut (int *iCard*)

Reads back the set PWM value and returs it.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

the PWM value

4.1.3.29 int k8061_ReadCounters (int *iCard*, int *cntr_arr*[ ])

Reads the error counters.

.. don't really know what these do

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

cntr_arr Array of 2 integers

Returns:

0 upon success, < 0 upon failure

4.1.3.30 int k8061_ReadDigitalChannel (int *iCard*, int *iChan*)

Reads a single digital channel and returns it's state as 0 or 1 unsigned char).

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

iChan The channel number on the card [1-8]

Returns:

1 when on, 0 when off

4.1.3.31 int k8061_ReadVersion (int *iCard*, char *version_str*[])

Reads back the version of the IC firmware.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

version_str The string to hold the IC firmware

4.1.3.32 int k8061_ResetCounters (int *iCard*)

Resets the error counters.

.. don't really know what these do

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

0 upon success, < 0 upon failure

4.1.3.33 int k8061_ScanBus (int *verb*)

Scans the complete usb bus hierarcy and looks for k8061 devices.

If it finds a matching device, it tries to set the usb configuration and claim the interface. When that has succeeded, it reads the address of the k8061 device and opens the filehandle of the corresponding _k8061_Handle entry.

Parameters:

verb Some verbosity parameter

Returns:

0 upon success, < 0 upon failure

4.1.3.34 int k8061_SetADCJumpers (int *iCard*, unsigned char *b*)

Sets the Analog Input jumper configuration on the card to the given pattern b encoded as a single byte.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

b The bit pattern

Returns:

0 upon success, < 0 upon failure

4.1.3.35 int k8061_SetAllAnalog (int *iCard*)

Sets all the analog output on channels on card iCard to their maximum (255).

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

0 upon success, < 0 upon failure

4.1.3.36 int k8061_SetAllDigital (int *iCard*)

Sets all digital outputs on the card to 1.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

Returns:

0 upon success, < 0 upon failure

4.1.3.37 int k8061_SetAnalogChannel (int *iCard*, int *iChan*)

Sets the analog output on channel iChan of card iCard to its maximum (255).

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

iChan The channel number on the card [1-8]

Returns:

0 upon success, < 0 upon failure

4.1.3.38 int k8061_SetDACJumpers (int *iCard*, unsigned char *b*)

Sets the Analog Output jumper configuration on the card to the given pattern b encoded as a single byte.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

b The bit pattern

Returns:

0 upon success, < 0 upon failure

4.1.3.39 int k8061_SetDigitalChannel (int *iCard*, int *iChan*)

Sets the digital output channel to 1.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

iChan The channel number on the card [1-8]

Returns:

0 upon success, < 0 upon failure

4.1.3.40 int k8061_WriteJumpers (int *iCard*, FILE * *fp*)

Write the status of the ADC/DAC jumpers to the filepointer.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

fp The filepointer, use stdout to write to terminal

Returns:

0 upon success, < 0 upon failure

4.1.3.41 int k8061_WriteStatus (int *iCard*, FILE * *fp*)

Writes some status report of the card to the filepointer, this shows whether the card is connected, whether the 12 VDC power cord is connected and what the version of the IC firmware is.

Parameters:

iCard The number (address) of the k8061 card on the bus [0-7]

fp The filepointer, use stdout to write to terminal

4.1.4 Variable Documentation

4.1.4.1 usb_dev_handle* _k8061_Handle[8]

Array with filehandles for the k8061 devices.

4.1.4.2 int _k8061_IsOpen[8]

Array with flags for the open k8061 devices.

4.1.4.3 unsigned char _k8061_jmp_adc[8]

Jumper settings for the AINs for each card.

4.1.4.4 unsigned char _k8061_jmp_dac[8]

Jumper settings for the AOUTs for each card.

4.1.4.5 char _k8061_ReadBuffer[50]

The read buffer.

4.1.4.6 char _k8061_WriteBuffer[50]

The receive buffer.

Index