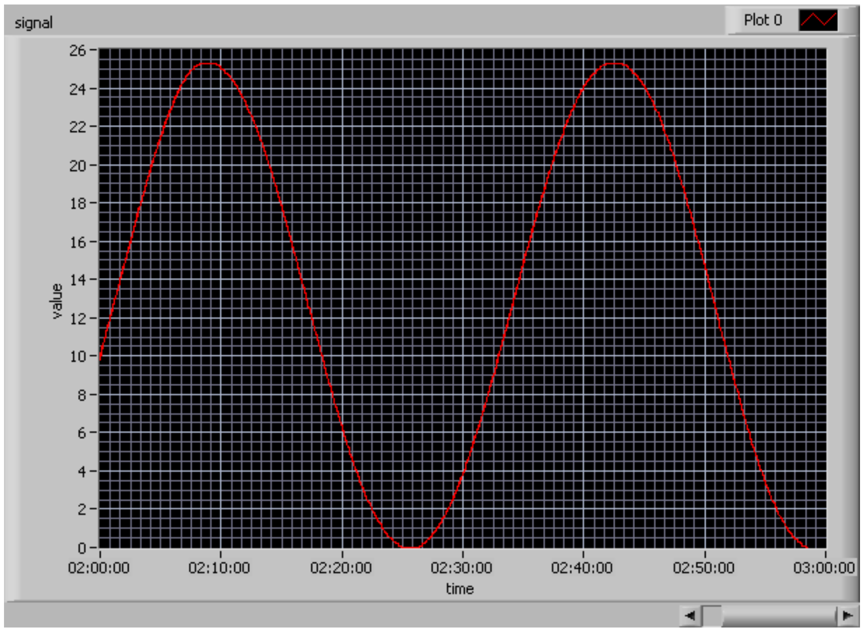
UDK - LabView Interface



Copyright information

⇒ Please check www.cesys.com to get the latest version of this document.

CESYS Gesellschaft für angewandte Mikroelektronik mbH

Zeppelinstrasse 6a

D – 91074 Herzogenaurach

Germany

UDK LabView Interface

Windows

Requirements

The following is required to build the UDK LabView Interface:

- CMake 2.6 or higher → www.cmake.org
- Microsoft Visual Studio 2005 or 2008; 2010 is experimental
- UDK C API (built version) → udkapic_vc[ver]_[arch].dll

Build the UDK LabView Interface dynamic link library

Prerequisites

There are some options that need to be fixed in msvc.cmake inside the UDK LabView Interface root directory:

- **USE_STATIC_RTL** If *0*, all projects are build against the dynamic runtime libraries. This requires the installation of the appropriate Visual Studio redistributable pack on every machine the UDK LabView Interface is used on.
- **UDK_ROOT** Path to the UDK root directory

Solution creation and build process

The preferred way is to open a command prompt inside the installation root of the UDK LabView Interface, let's assume c:\UDK\LabViewInterface.

```
c:
cd UDK\LabViewInterface
```

CMake allows the build directory separated to the source directory, so it's a good idea to do it inside an empty sub-directory:

```
mkdir build
cd build
```

The following code requires an installation of CMake and at least one supported Visual Studio version. If CMake isn't included into the PATH environment variable, the path must be specified as well:

```
cmake ..
```

This searches the preferred Visual Studio installation and creates projects for it. Visual Studio Express users may need to use the command prompt offered by their installation. If multiple Visual Studio versions are installed, CMake's command parameter '-G' can be used to specify a special one, see CMake's documentation in this case. This process creates the solution files inside *c:\UDK\LabViewInterface\build*. All subsequent tasks can be done in Visual Studio (with the created solution), another invocation of cmake isn't necessary under normal circumstances.

Important: The UDK LabView Interface must be build with the same toolchain and build flags like the UDK C API. Otherwise CMake will not be able to find the compiled UDK libraries.

Info: It is easy to create different builds with different Visual Studio versions by creating different build directories and invoke CMake with different '-G' options inside them:

```
c:
cd UDK\LabViewInterface
mkdir build2005
cd build2005
cmake -G"Visual Studio 8 2005" ..
cd ..
mkdir build2008
cd build2008
cmake -G"Visual Studio 9 2008" ..
```

Configuration example for EFM01.VI (Windows)

Introduction

Double click “efm01.vi” to open the vi. This will set the working directory to the directory containing “efm01.vi”.

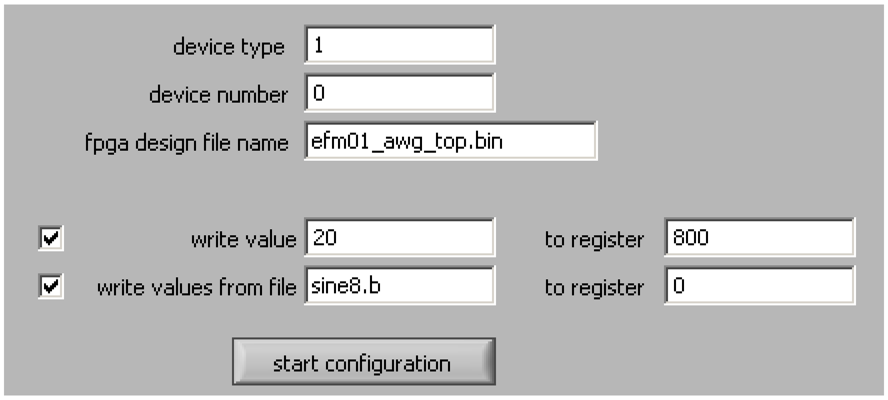
The working directory must contain

• UDK C API (built version)	→ udkapic_vc[ver]_[arch].dll
• LabView dynamic link library	→ udk_labview_if.dll

Device selection and FPGA configuration

The device type is set to 0x00020000 + x in the block diagram of the vi. In this configuration example x=**1** is used. This will specify all available EFM01 devices. For other devices “device type” must be changed to the corresponding value. The device number value will specify the device index. In the example the first found device is used (index **0**). Further the fpga design **efm01_awg_top.bin** is used. For this design it is necessary to write some values to certain addresses.

- a frequency value to register **0x800**:   value = (freq_out/48 MHz) * 2^32
  → **0x20** in this example
- a waveform to register **0x0**
  → **sine8.b** in this example

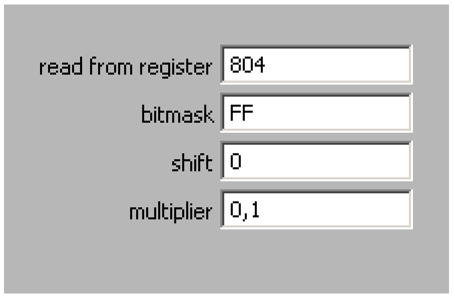


device type	1		
device number	0		
fpga design file name	efm01_awg_top.bin		
☑ write value	20	to register	800
☑ write values from file	sine8.b	to register	0

The **"start configuration"** button will start fpga configuration.

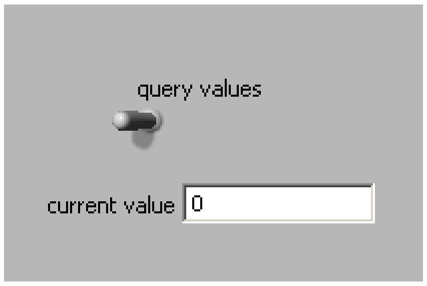
Reading data and value conversion

In this example with the efm01_awg_top.bin fpga design values can be received from register **0x804**. Some bytes can be extracted from the value with a bit mask, **0xFF** in this example. The result can be shifted (here it is not, shift = **0**) and multiplied with a scaling factor, **0,1** in this case.



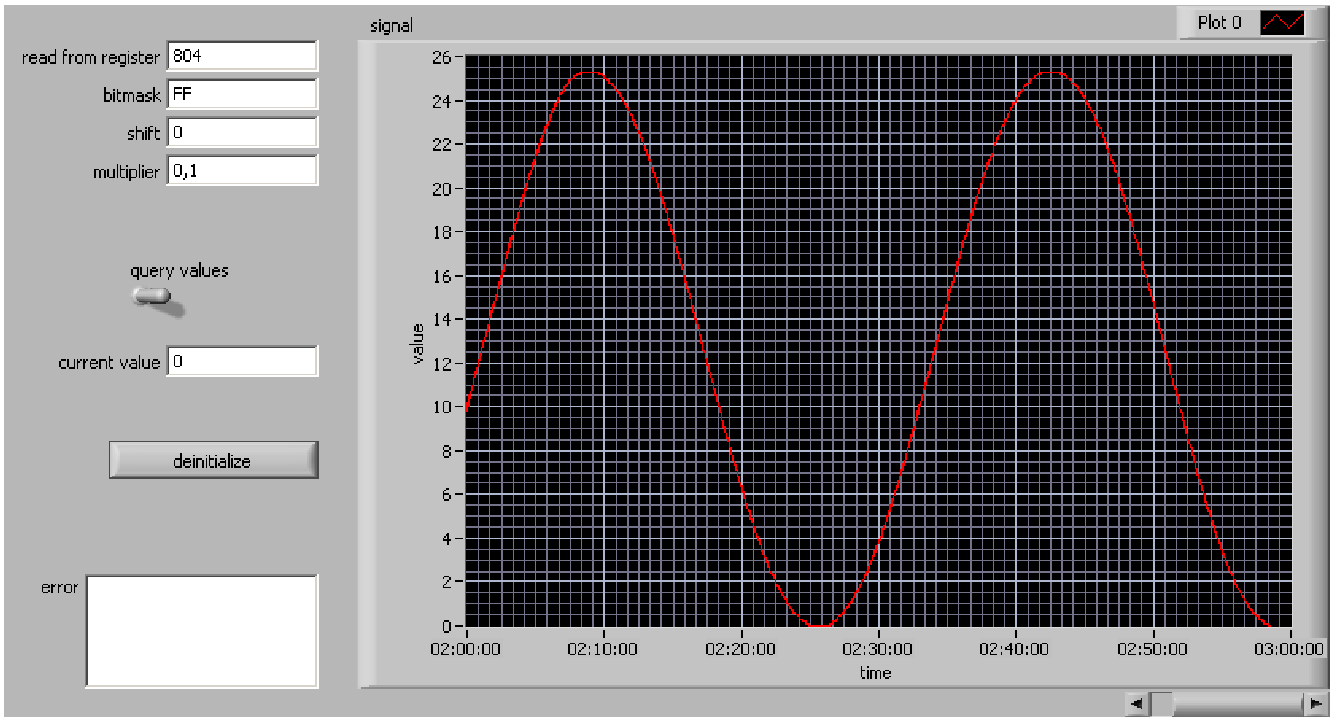
read from register	804
bitmask	FF
shift	0
multiplier	0,1

Switching on "query values" will start reading data.

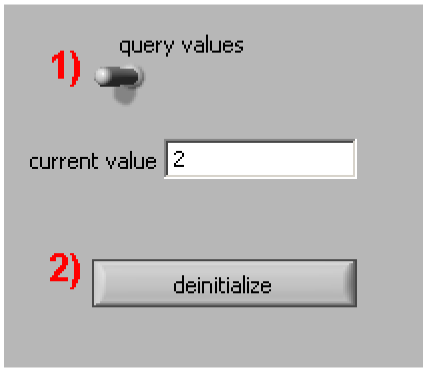


current value	0

The current value field shows the actual read (converted) value. A graph will be plotted in the signal area.

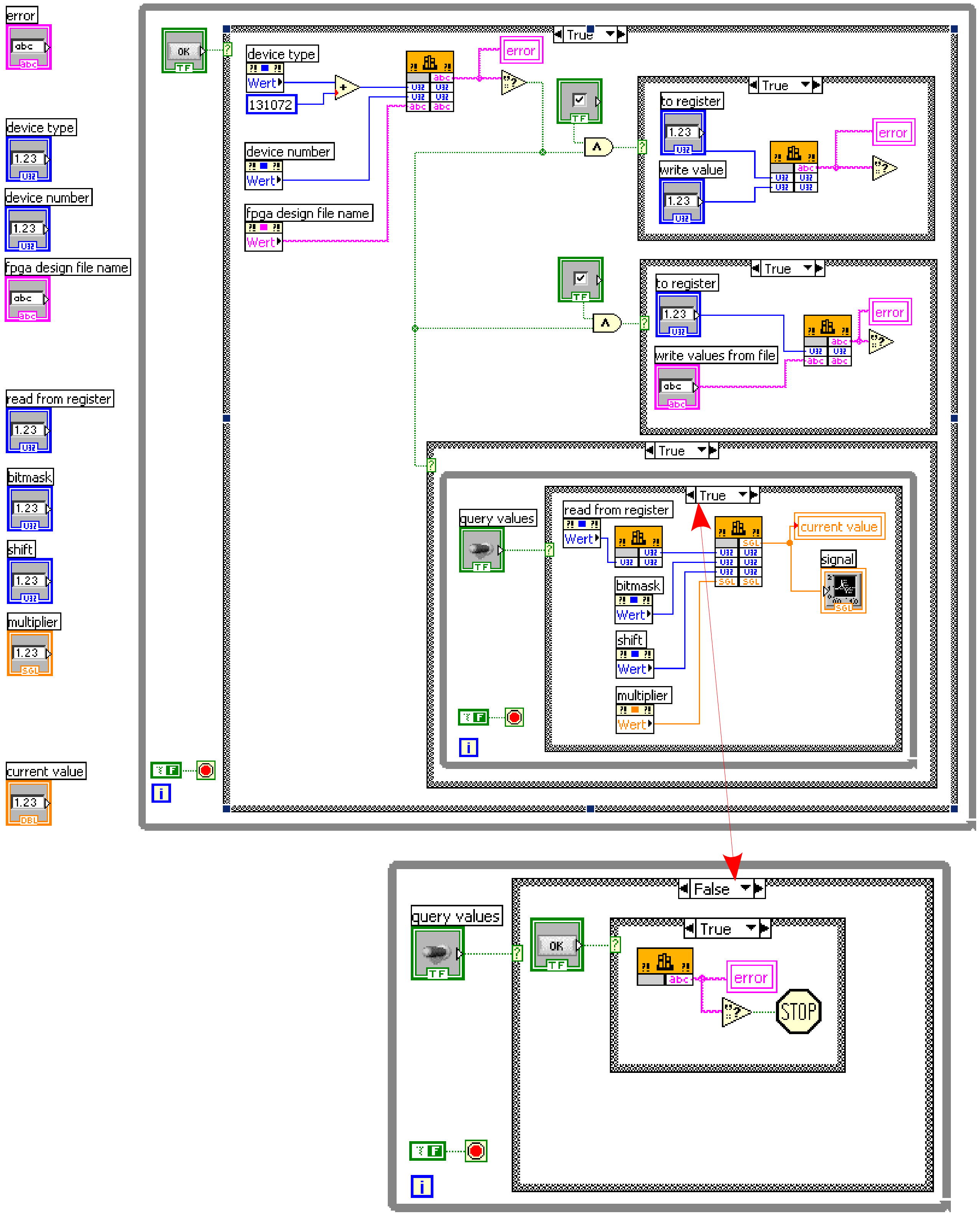

read from register 804
bitmask FF
shift 0
multiplier 0,1
query values
current value 0
deinitialize
error

Note: To safely close the device and deinitialize the API switch off **“query values”** and click the **“deinitialize”** button. The vi will stop automatically.



Important: If the “deinitialize” button is not clicked and the vi is stopped manually an error will occur if you run the vi again. Closing LabView and restarting the application should solve the problem and it will work probably. It might be necessary to replug the device.

Block diagram of the VI



Linux

Requirements

The following is required to build the UDK LabView Interface:

- GNU C++ compiler toolchain
- CMake 2.6 or higher → www.cmake.org
- UDK C API (built version) → libudkapic.so

Build the UDK LabView Interface dynamic link library

Prerequisites

There are some options that need to be fixed in msvc.cmake inside the UDK LabView Interface root directory:

- **CMAKE_BUILD_TYPE** Select build type, can be one of *Debug, Release, RelWithDebInfo, MinSizeRel*. If there should be at least 2 builds in parallel, remove this line and specify the type using command line option *-DCMAKE_BUILD_TYPE=…*.
- **UDK_ROOT** Path to the UDK root directory

Makefile creation and build process

The preferred way is to open a command prompt inside the installation root of the UDK LabView Interface, let's assume /UDK/LabViewInterface.

```
cd /UDK/LabViewInterface
mkdir build
cd build
cmake ..
```

If all external dependencies are met, this will finish creating a Makefile. To build the UDK LabView Interface, just invoke make:

```
make
```

Important: The UDK LabView Interface must be build with the same toolchain and build flags like the UDK C API. Otherwise CMake will not be able to find the compiled UDK libraries.

Methods/Functions

initialize

Configures device devNumber of type deviceType with design from file fpgaDesign.

C	const char *initialize(unsigned int deviceType, unsigned int devNumber, const char *fpgaDesign)

Returns error text if failed.

ReadReg

Queries value from register address.

C	unsigned int readReg(unsigned int address)

Returns unsigned int value from register address.

WriteReg

Writes value to register address.

C	const char * writeReg(unsigned int address, unsigned int value)

Returns error text if failed.

WriteValuesFromFileToDevice

Writes content of file filename to register address.

C	const char *writeValuesFromFileToDevice(unsigned int address, char *filename)

Returns error text if failed

deinitialize

Deinitializes configured devices.

C	const char *deinitialize()

Returns error text if failed.

convertValue

Converts unsigned int value to float newValue.

C	float convertValue(unsigned int value, unsigned int bitmask, unsigned int shift, float multiplier)

$$\mathtt{newValue} = ((\mathtt{value}\ \&= \mathtt{bitmask}) >> \mathtt{shift}) * \mathtt{multiplier}$$

Returns float newValue.

Table of Contents