struck innovative systeme

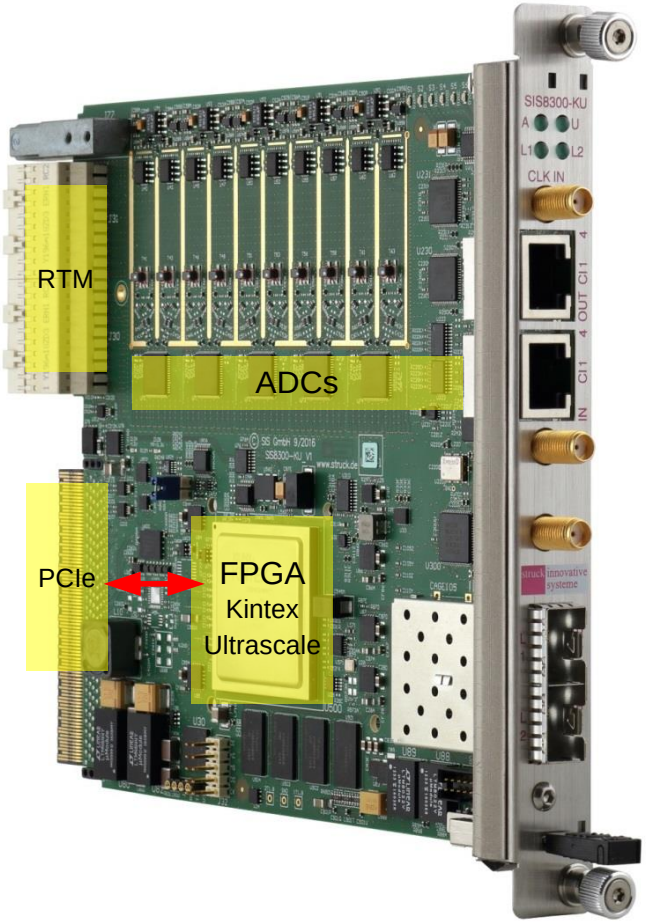
www.struck.de

Tino Häupke

How to talk to the hardware from the OS illustrated with Struck Digitizer AMC/RTM Combination

QUALITY SYSTEM CERTIFICATION DNV·GL ISO 9001

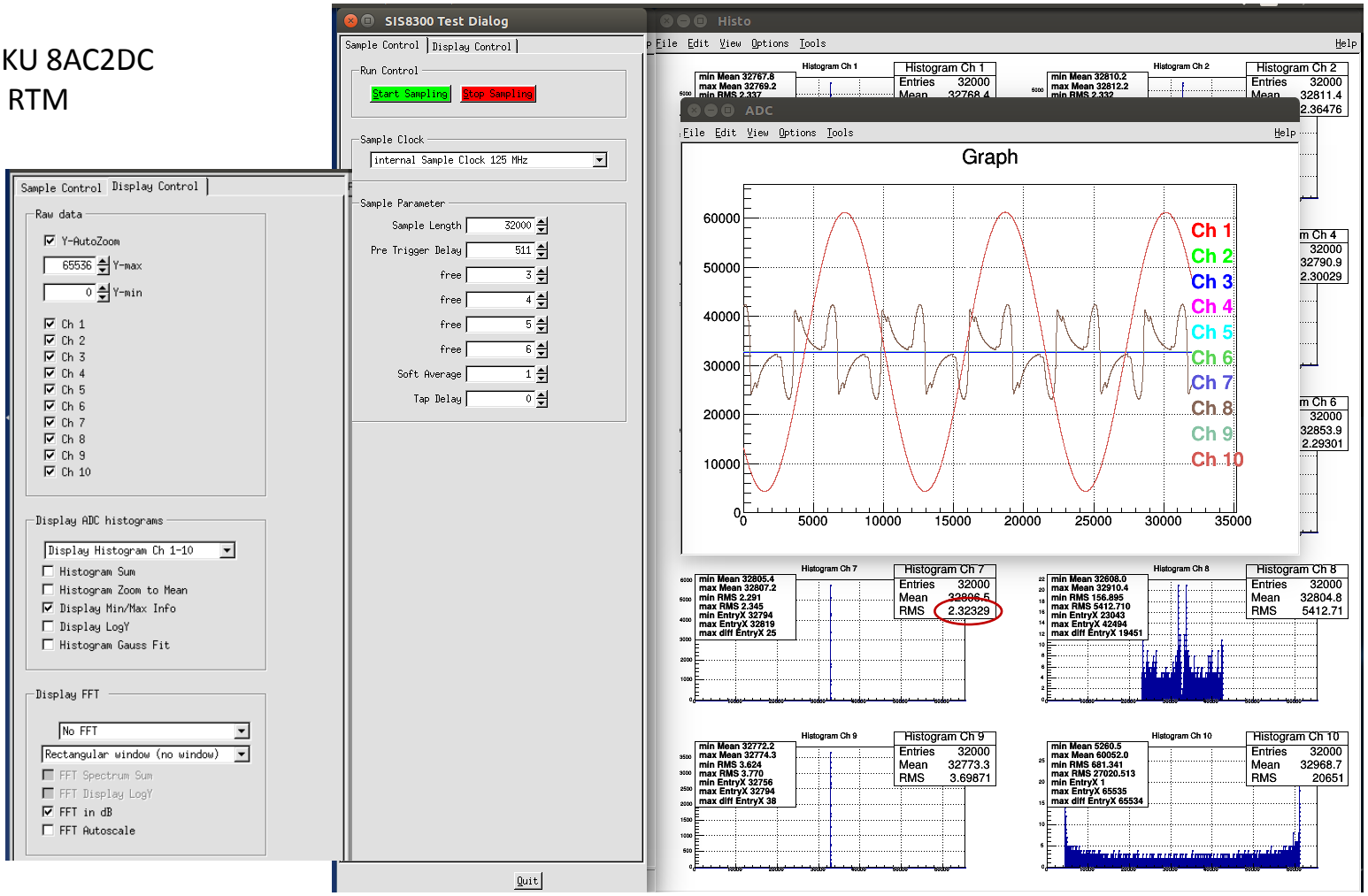
MTCA.4 Crate with a CPU, NAT-MCH and AMCs (SIS8300KU)



Our goal is to build an application to get and show ADC data

SIS8300-x ROOT GUI

Use SIS8300-KU 8AC2DC
with SIS8900 RTM



RMS for open AC/DC channel

PCI Express is used as Communication/Transfer bus

Applications:

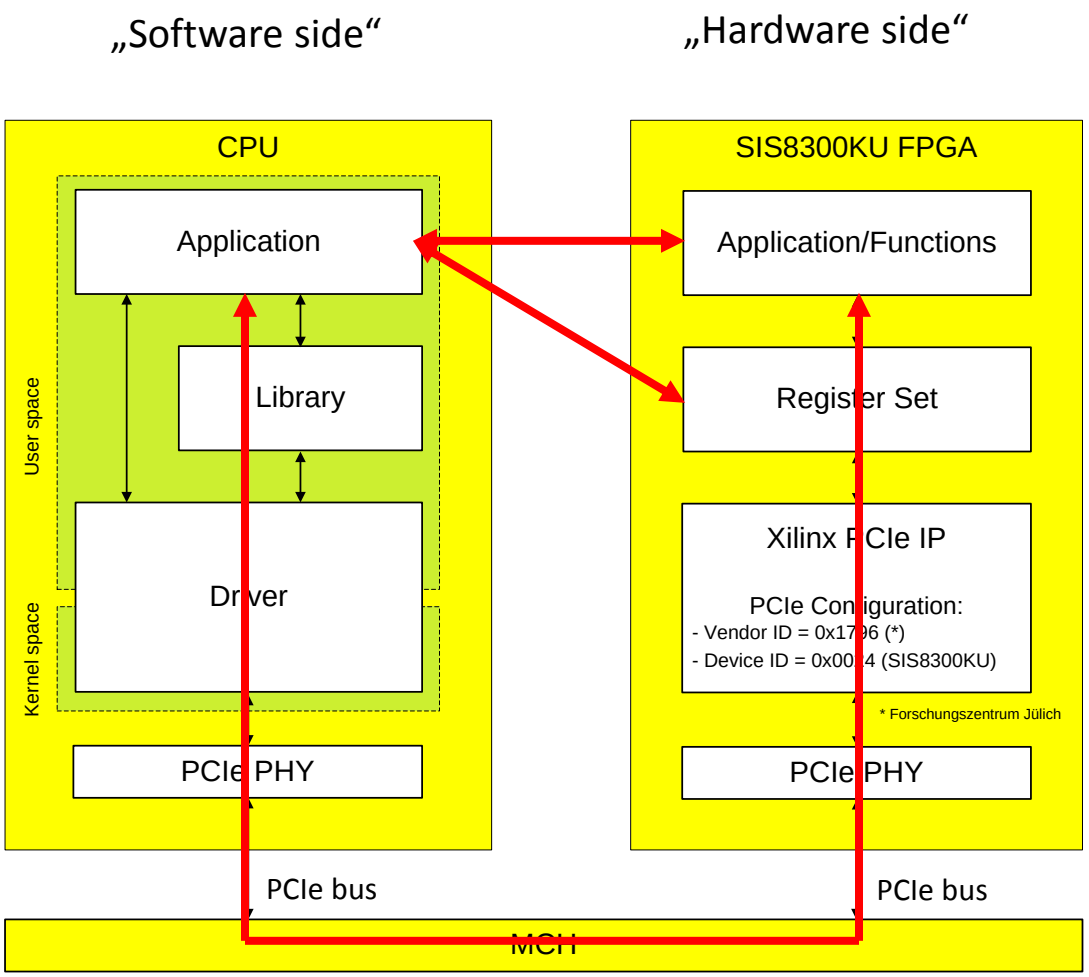
register_read

..

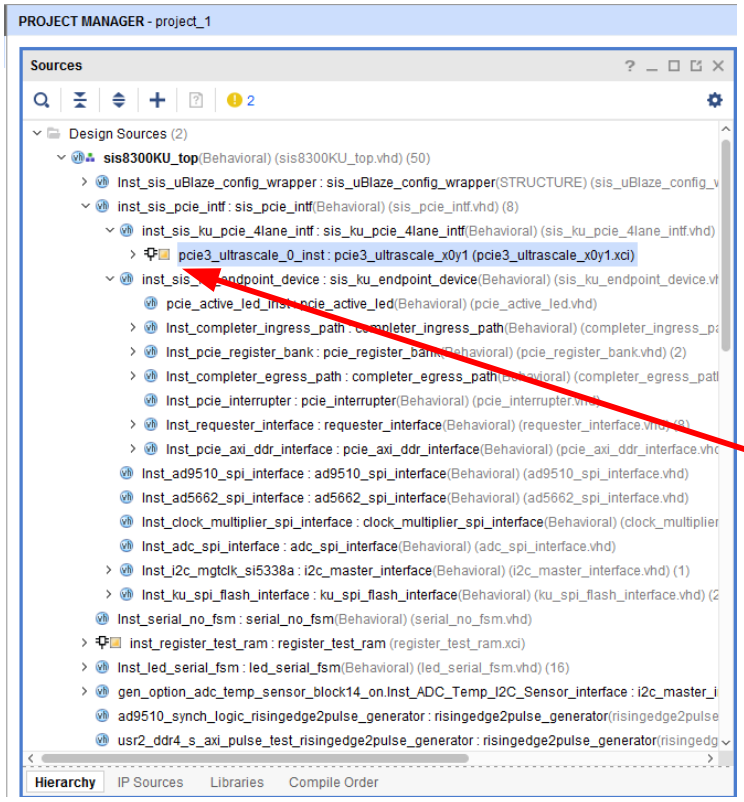
..

SIS8300-x ROOT GUI

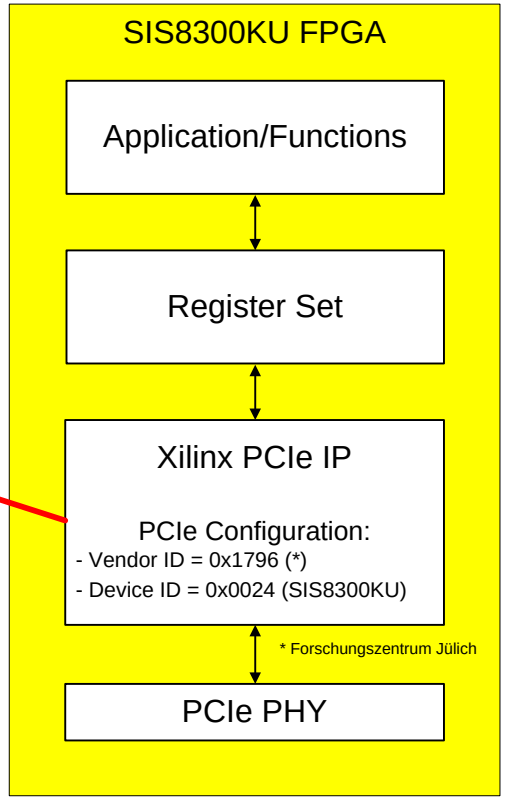
EPICS, ….



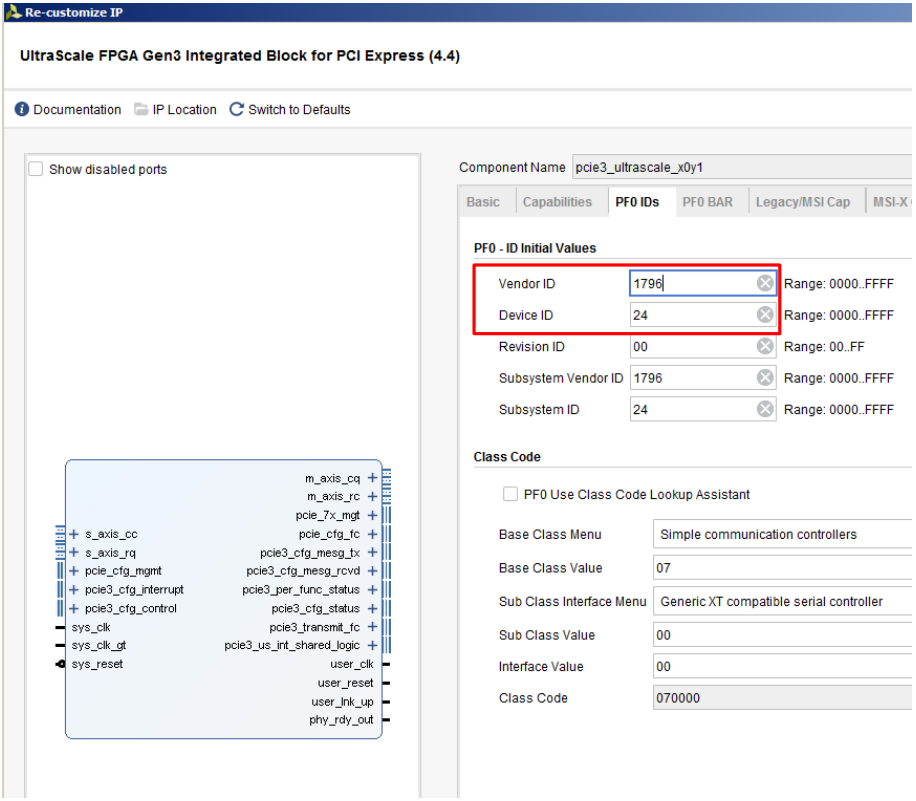
Struck SIS8300KU AXI-Based Xilinx Vivado FPGA Famework

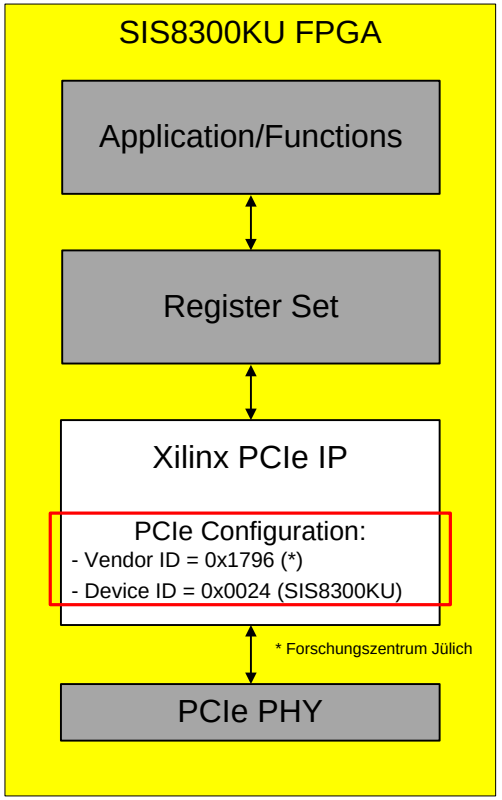


„Hardware side“

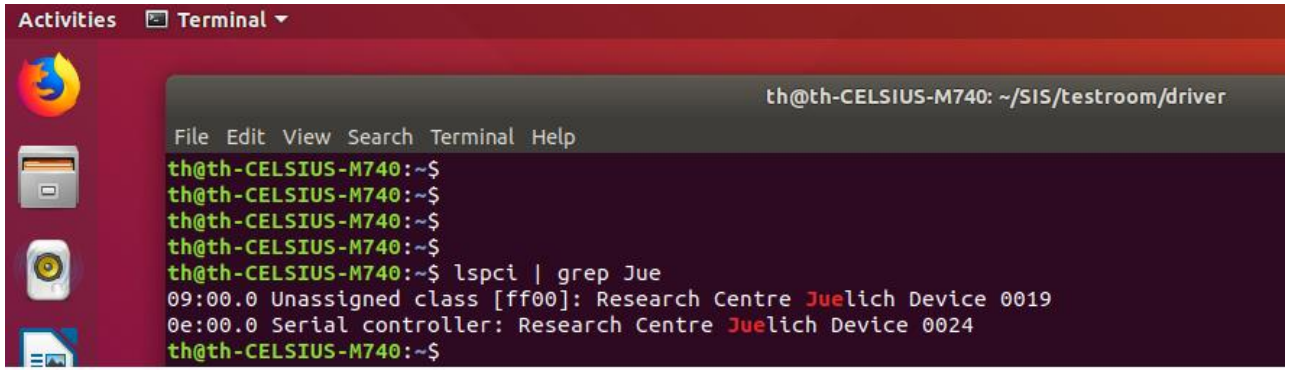


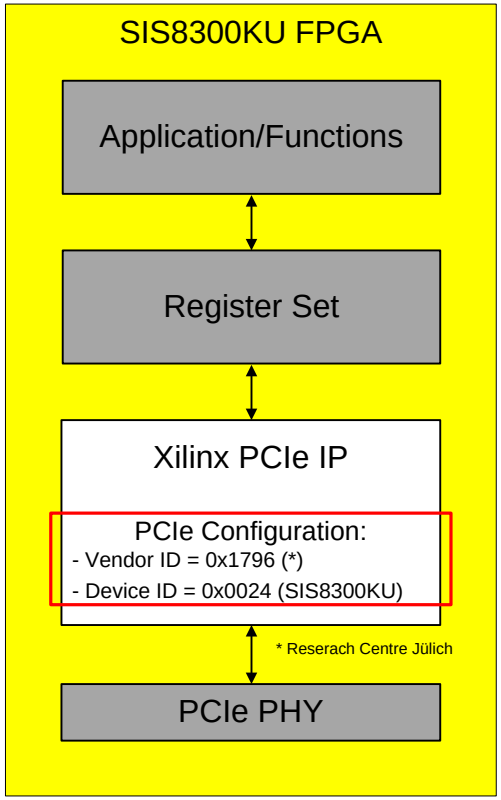
SIS8300KU FPGA PCI Express Configuration





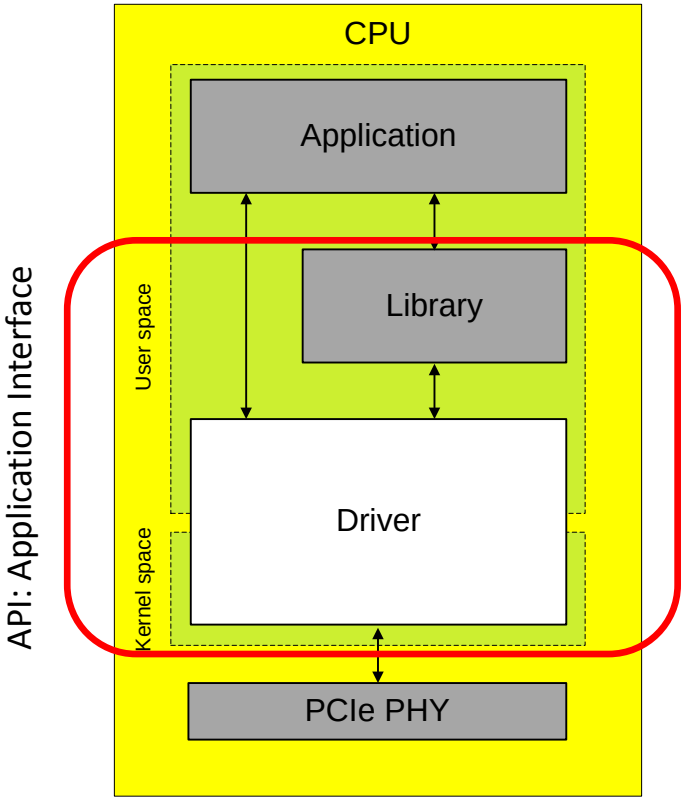
Check with „lspci“ if the OS has detected the SIS Devices

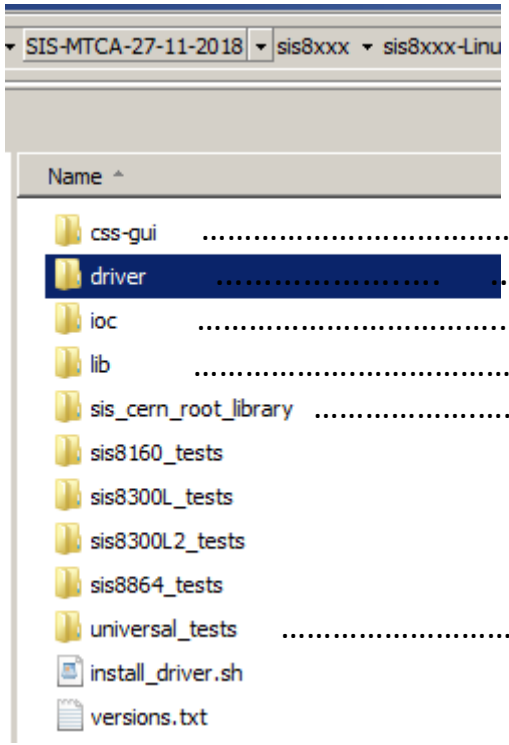




You will find the SIS8xxx driver on the DVD

„Software side“





EPICS example

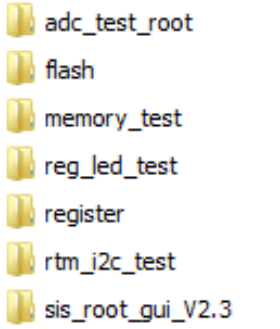
SIS driver

EPICS IOC

SIS library

SIS CERN root library

Applications/Tests for all SIS83xx



Why we need a device driver like the SIS830x driver

A device driver enables the Access from a Userspace Application to a Hardware Device (Kernel space)

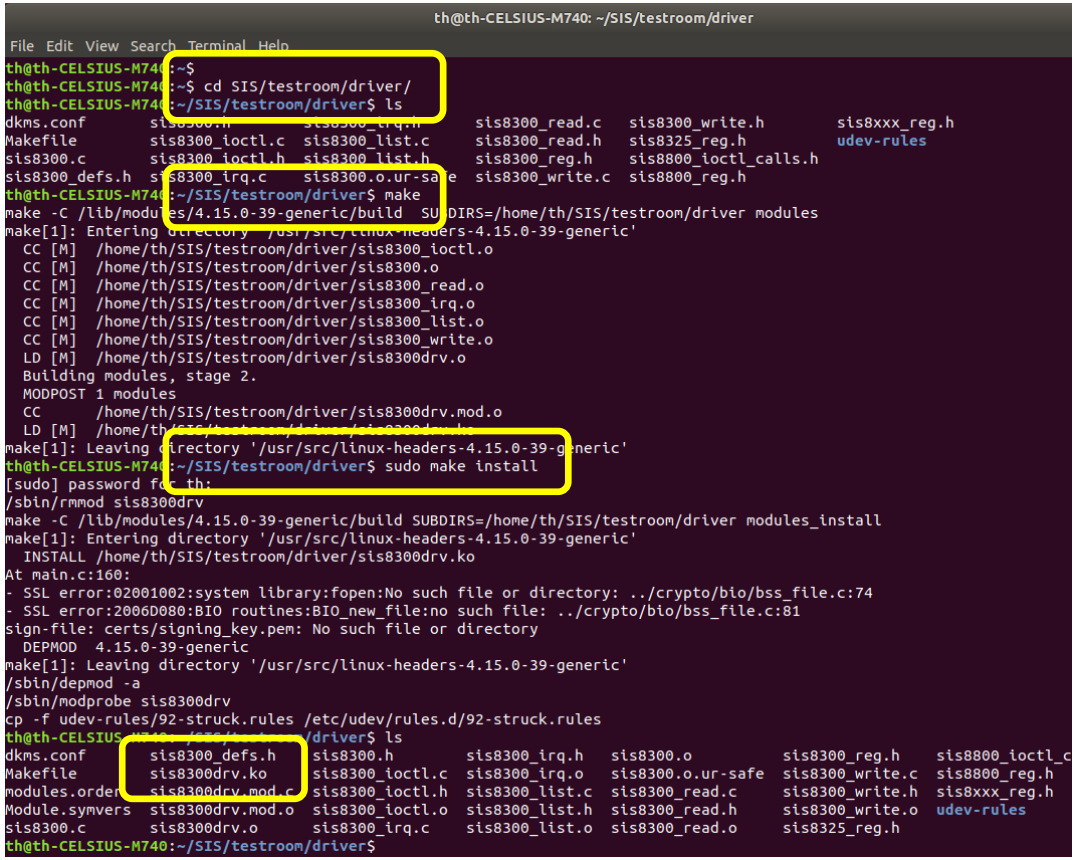
The driver provides:

Address-Mapped I/O (ioctl)
- the SIS8300KU „PCI bar 0 address space“ is mapped into the „processor address space“
- enables processor reads/writes from/to the SIS8300KU Register Set with the system call „ioctl“

Stream-Oriented I/O (DMA)
- are handled/triggered by writing to the „DMA“ and „IRQ“ register set
- enables data transfer from/to the User-Memory (PC) to/from the AMC (SIS8300KU) Memory

Interrupts
- are handled by writing to „IRQ“ register set

Install the SIS830x driver



1.
Copy the directory „sis8xxx-Linux“ from the DVD to a local directory on the PC

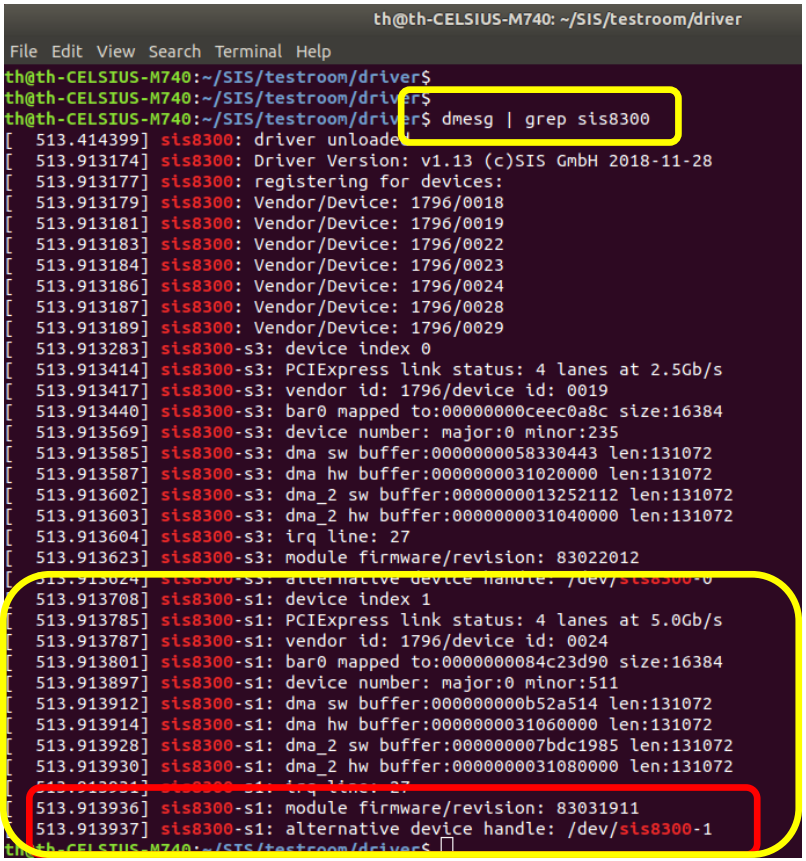
2.
Go to the directory „driver“

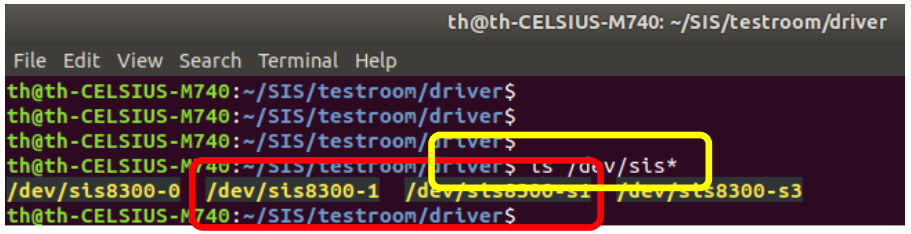
3.
Execute „make“

4.
Execute „sudo make install“ (as root)

5.
The kernel object „sis8300drv.ko“ is generated

You can check with „dmesg“ if the driver is loaded

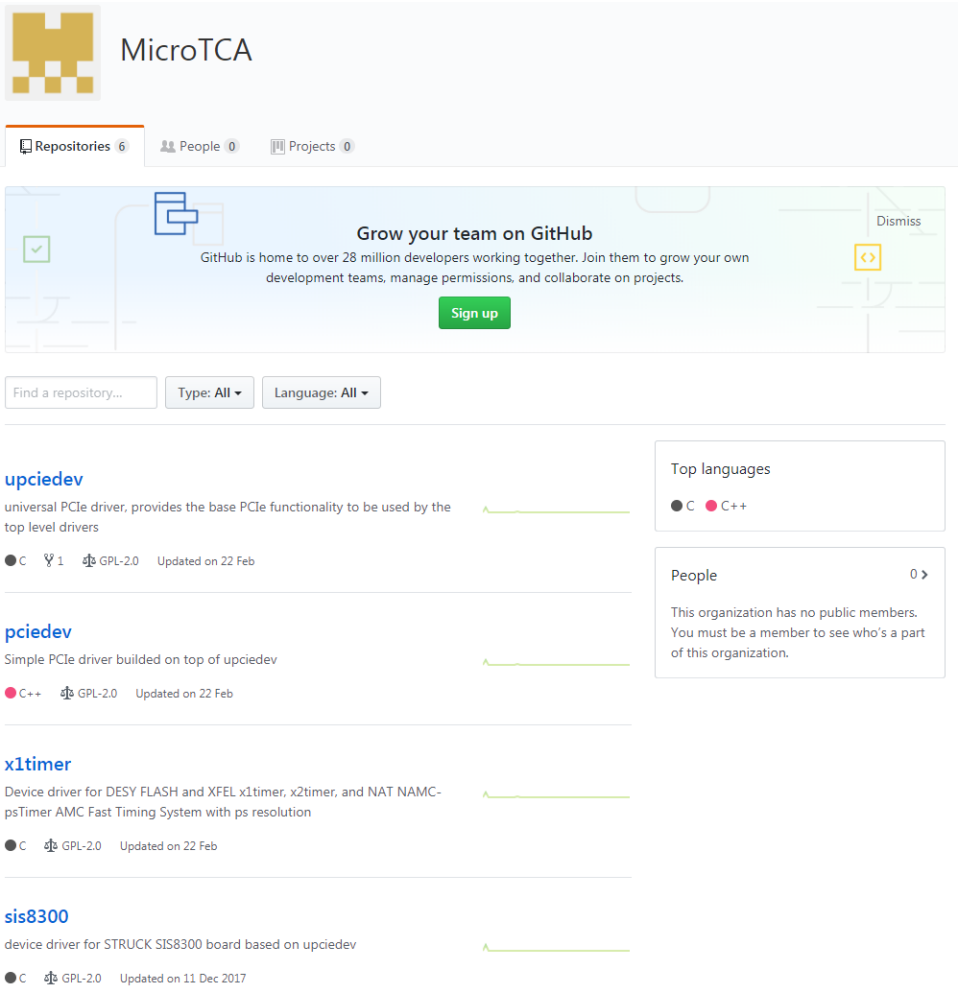




the driver generates two Device Handles/Names for each hardware device:

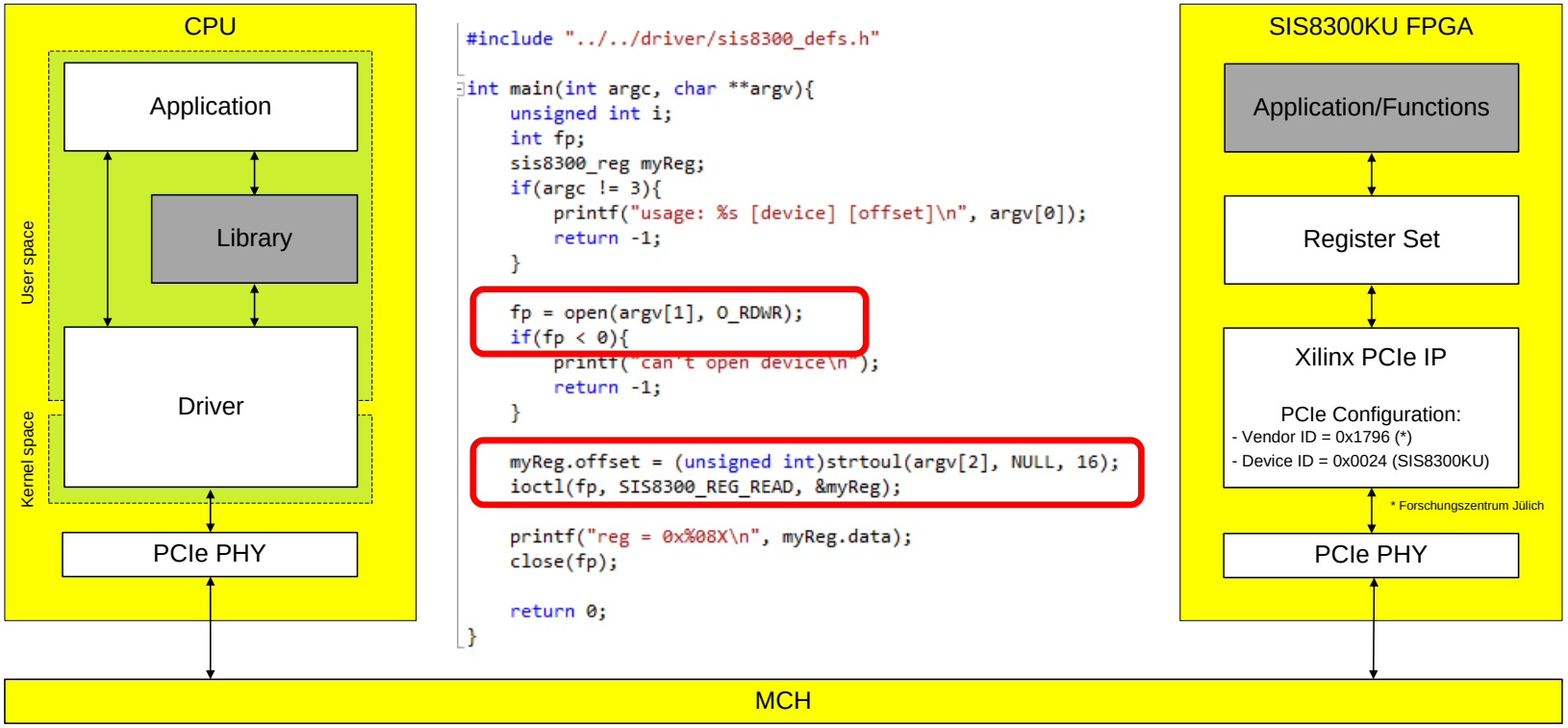
1. an enumerated Name: sis8300-1
2. a PCIe Slot dependent Name: sis8300-s1

A Desy SIS8300 driver is available, also (https://github.com/MicroTCA)



Register Read Example with the system call „ioctl“

Application: \universal_tests\register\regread_iocall.c




```c
#include "../../driver/sis8300_defs.h"

int main(int argc, char **argv){
    unsigned int i;
    int fp;
    sis8300_reg myReg;
    if(argc != 3){
        printf("usage: %s [device] [offset]\n", argv[0]);
        return -1;
    }

    fp = open(argv[1], O_RDWR);
    if(fp < 0){
        printf("can't open device\n");
        return -1;
    }

    myReg.offset = (unsigned int)strtoul(argv[2], NULL, 16);
    ioctl(fp, SIS8300_REG_READ, &myReg);

    printf("reg = 0x%08X\n", myReg.data);
    close(fp);

    return 0;
}
```

```
th@th-CELSIUS-M740:~/SIS/testroom/universal_tests/register$ ./regread_iocall /dev/sis8300-s1 0x1
reg = 0x00000081
th@th-CELSIUS-M740:~/SIS/testroom/universal_tests/register$ ./regread_iocall /dev/sis8300-s1 0x0
reg = 0x83031911
```

An Extract of Registers of the SIS830x FPGA Firmware Implementation (Register Set)

Register Address Map for Identification …

Offset	Access	Function
0x00	R	Module Identifier/Firmware Version register
0x01	R	Serial number register
0x02	R/W	reserved
0x03	R/W	reserved
0x04	R/W	User Control/Status register (JK)
0x05	R	Firmware Options register
0x06	R/W	ADC Temperature Sensor interface register
0x07	R	PCIe Status register

Register Address Map for the Application like „Sample Block Length“

0x120	R/W	ADC ch1 Memory Sample Start Block Address / Actual Block Address register
0x121	R/W	ADC ch2 Memory Sample Start Block Address / Actual Block Address register
. .		. .
0x129	R/W	ADC ch10 Memory Sample Start Block Address / Actual Block Address register
0x12A	R/W	ADC chx Sample Block Length register
0x12B	R/W	ADC chx Ringbuffer Delay register (0 to 2046)

An Extract of Registers of the SIS830x FPGA Firmware Implementation (Register Set)

Register Address Map for Stream-Oriented I/O (DMA)

0x200	R/W	DMA_READ_DST_ADR_LO32
0x201	R/W	DMA_READ_DST_ADR_HI32
0x202	R/W	DMA_READ_SRC_ADR_LO32
0x203	R/W	DMA_READ_LEN
0x204	R/W	DMA_READ_CTRL
0x205	R/W	DMA Readout Sample byte swap control
0x210	R/W	DMA_WRITE_SRC_ADR_LO32
0x211	R/W	DMA_WRITE_SRC_ADR_HI32
0x212	R/W	DMA_WRITE_DST_ADR_LO32
0x213	R/W	DMA_WRITE_LEN
0x214	R/W	DMA_WRITE_CTRL

Register Address Map for Interrupts

0x220	R/W	IRQ Enable
0x221	R	IRQ Status
0x222	W	IRQ Clear
0x223	KA	IRQ Refresh

Relating to the SHAPI – Standard Hardware API

MicroTCA™

Standard Hardware API Design Guide

Guideline for designing Hardware Access APIs for MTCA.4 Systems

MTCA_DG.1 R1.0

January 9, 2017

PICMG®

Open Modular Computing Specifications

µTCA®

SHAPI Register Address Map

Table 3-1: Standard Device Identification and Control Registers

31 16	15 0		
Magic Word	SHAPI Version	0x00	ro
First Module Address		0x04	ro
Hardware ID	Hardware Vendor ID	0x08	ro
Device Firmware ID	Device Vendor ID	0x0C	ro
Firmware Version		0x10	ro
Firmware Timestamp		0x14	ro
Firmware Name		0x18	ro
		0x1C	ro
		0x20	ro
Device Capabilities		0x24	ro
Device Status		0x28	ro
Device Control		0x2C	rw
Device Interrupt Mask		0x30	rw
Device Interrupt Flag		0x34	ro
Device Interrupt Active		0x38	ro
Scratch Register		0x3C	rw

The SIS8300KU IOCTL Register Access implementation looks like the SHAPI IOCTL Register Access implementation

The SIS8300KU DMA/IRQ implementation is similar to the SHAPI DMA/IRQ implementation

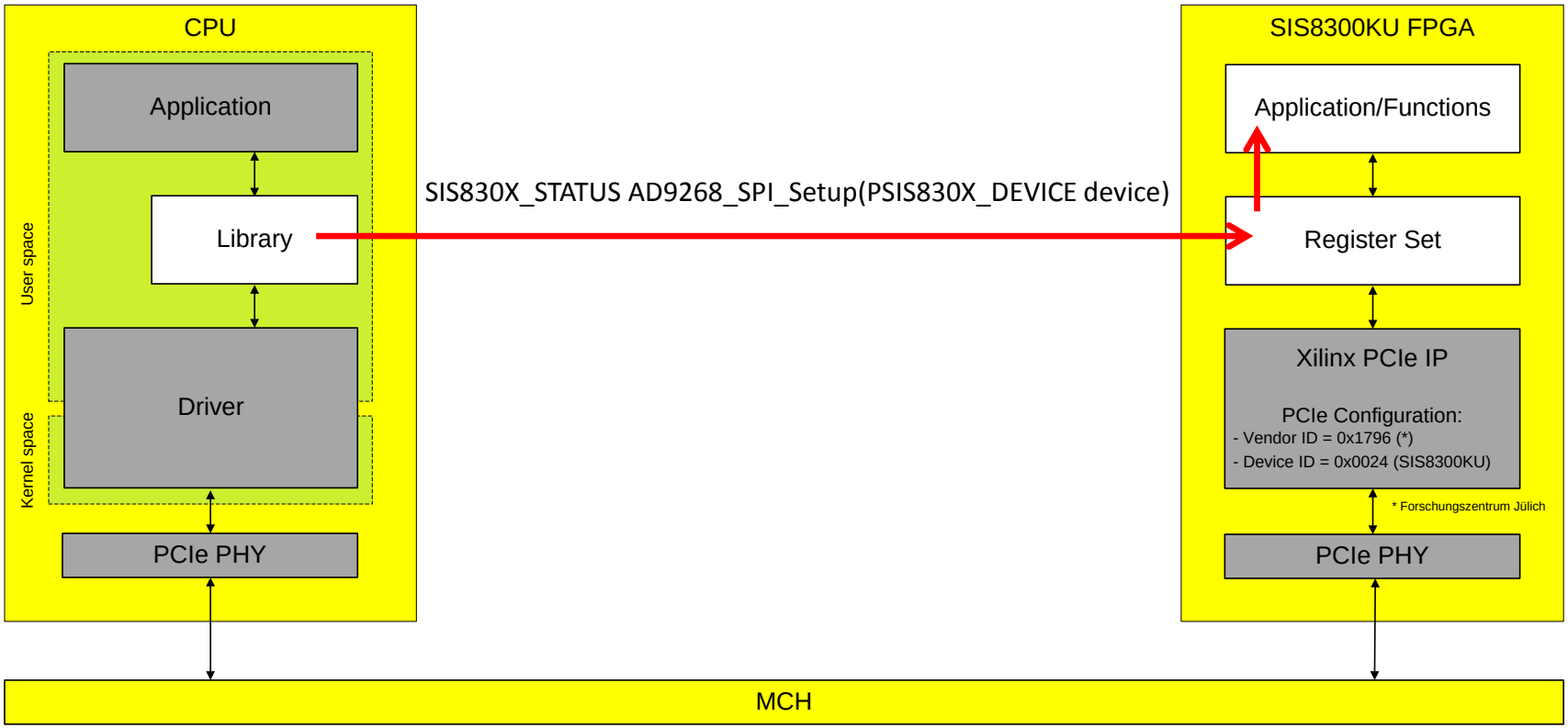
The SIS8300KU Register Address Map is NOT identical to the SHAPI Register Address Map

SIS830x library <--> SIS830x FPGA Firmware

The Library contains functions/calls which correspond to the SIS8300KU FPGA Firmware Functions

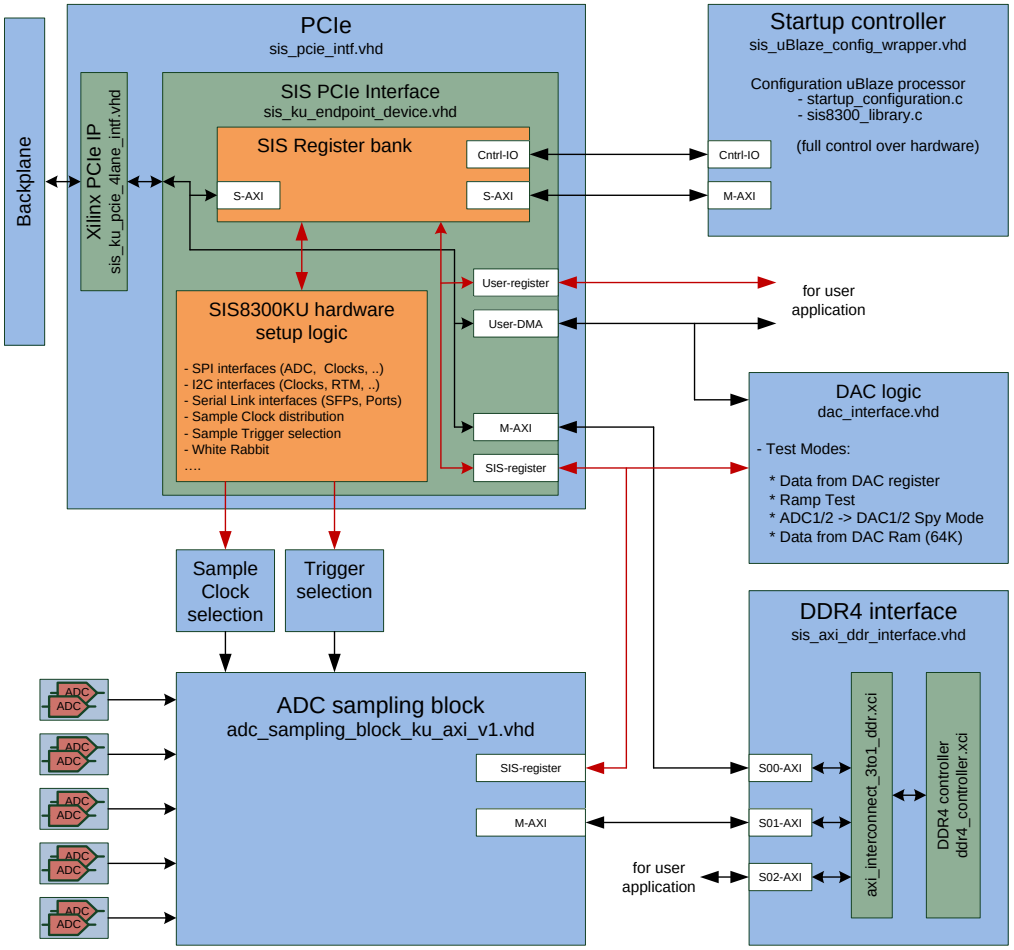
a Library call contains multiple cycles to execute a “function” on the SIS8300KU

for example: setup of the ADC chips via SPI



Block diagram: SIS8300KU AXI4-Based Xilinx Vivado FPGA Famework

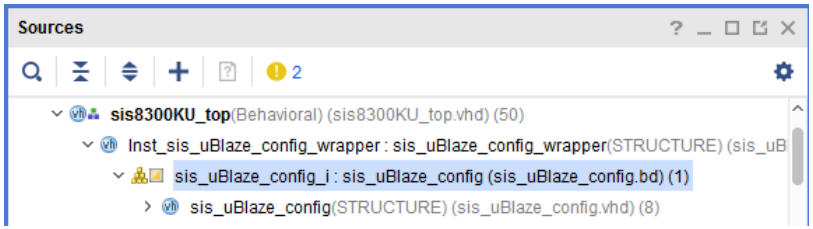
Struck SIS8300-KU AXI-Based Xilinx FPGA Framework



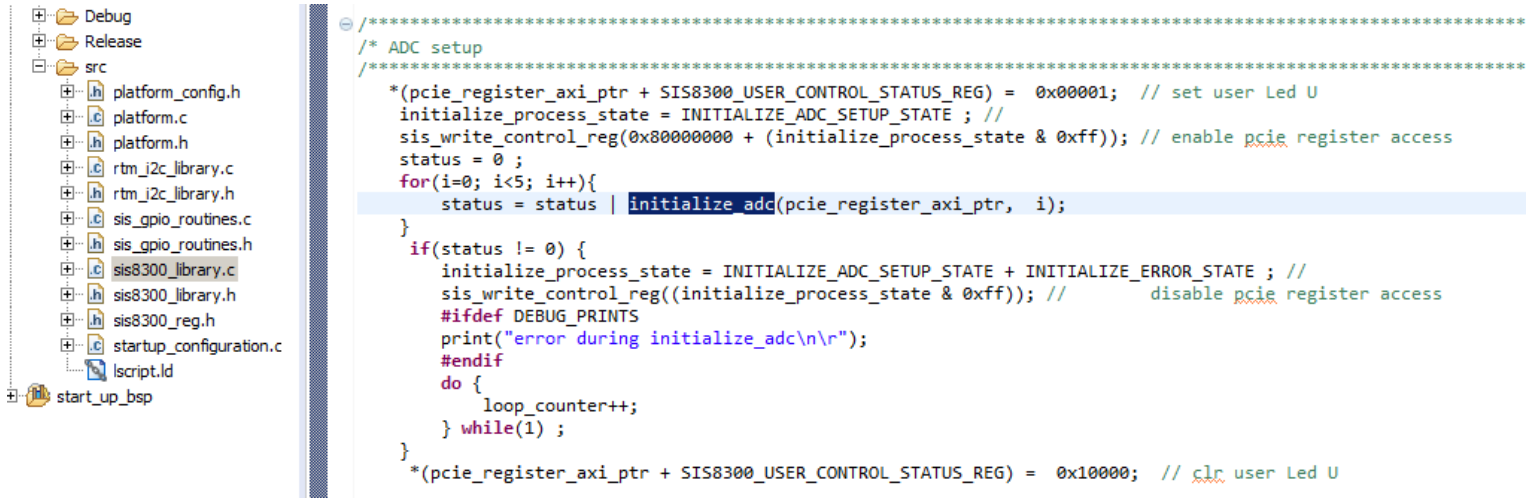
Startup Controller
(initially proposed by ESS)
Allows the complete Configuration
of a SIS8300-KU Digitizer
and an attached RTM
without MTCA CPU Interaction

AXI: Advanced eXtensible Interface protcol

Vivado FPGA Famework: Startup Controller



Xilinx Software Development Kit: SDK



Build the SIS830x library (libSIS830x.so)

```
th@th-CELSIUS-M740:~/SIS/testroom/lib$
th@th-CELSIUS-M740:~/SIS/testroom/lib$ ls libSIS830x/
ad9268.c       ads42lb69.c        libSIS830x.layout  si5326.c              sis830x.h      spiFlash.h
ad9268.h       ads42lb69.h        main.c             si5326.h              sis830xReg.h
ad9510.c       internal.c         Makefile           si5338aSynth.c        sis830xStat.h
ad9510.h       internal.h         obj                si5338aSynthDefault.h sis830xType.h
adc_ad9268.c   libSIS830x.cbp     rtmI2C.c           si5338aSynth.h        sis830xVer.h
adc_ad9268.h   libSIS830x.depend  rtmI2C.h           sis8300kuReg.h        spiFlash.c
th@th-CELSIUS-M740:~/SIS/testroom/lib$
th@th-CELSIUS-M740:~/SIS/testroom/lib$ make
for dir in libSIS830x/ libSIS8900/ libSIS8100/ libSIS8864/; do make -C $dir; done
make[1]: Entering directory '/home/th/SIS/testroom/lib/libSIS830x'
test -d obj/Release || mkdir -p obj/Release
gcc -Wall -fPIC -O2 -fPIC -I../../driver -c si5326.c -o obj/Release/si5326.o
gcc -Wall -fPIC -O2 -fPIC -I../../driver -c spiFlash.c -o obj/Release/spiFlash.o
```

1.
Go to the directory „lib“

2.
Execute „make“

```
th@th-CELSIUS-M740: ~/SIS/testroom/lib
File  Edit  View  Search  Terminal  Help
th@th-CELSIUS-M740:~/SIS/testroom/lib$
th@th-CELSIUS-M740:~/SIS/testroom/lib$
th@th-CELSIUS-M740:~/SIS/testroom/lib$ ls libSIS830x/
ad9268.c       ads42lb69.c        libSIS830x.layout  rtmI2C.h              sis8300kuReg.h  spiFlash.c
ad9268.h       ads42lb69.h        libSIS830x.so      si5326.c              sis830x.h       spiFlash.h
ad9510.c       internal.c         main.c             si5326.h              sis830xReg.h
ad9510.h       internal.h         Makefile           si5338aSynth.c        sis830xStat.h
adc_ad9268.c   libSIS830x.cbp     obj                si5338aSynthDefault.h sis830xType.h
adc_ad9268.h   libSIS830x.depend  rtmI2C.c           si5338aSynth.h        sis830xVer.h
th@th-CELSIUS-M740:~/SIS/testroom/lib$
```

3.
The library „libSIS830x.so“ is generated

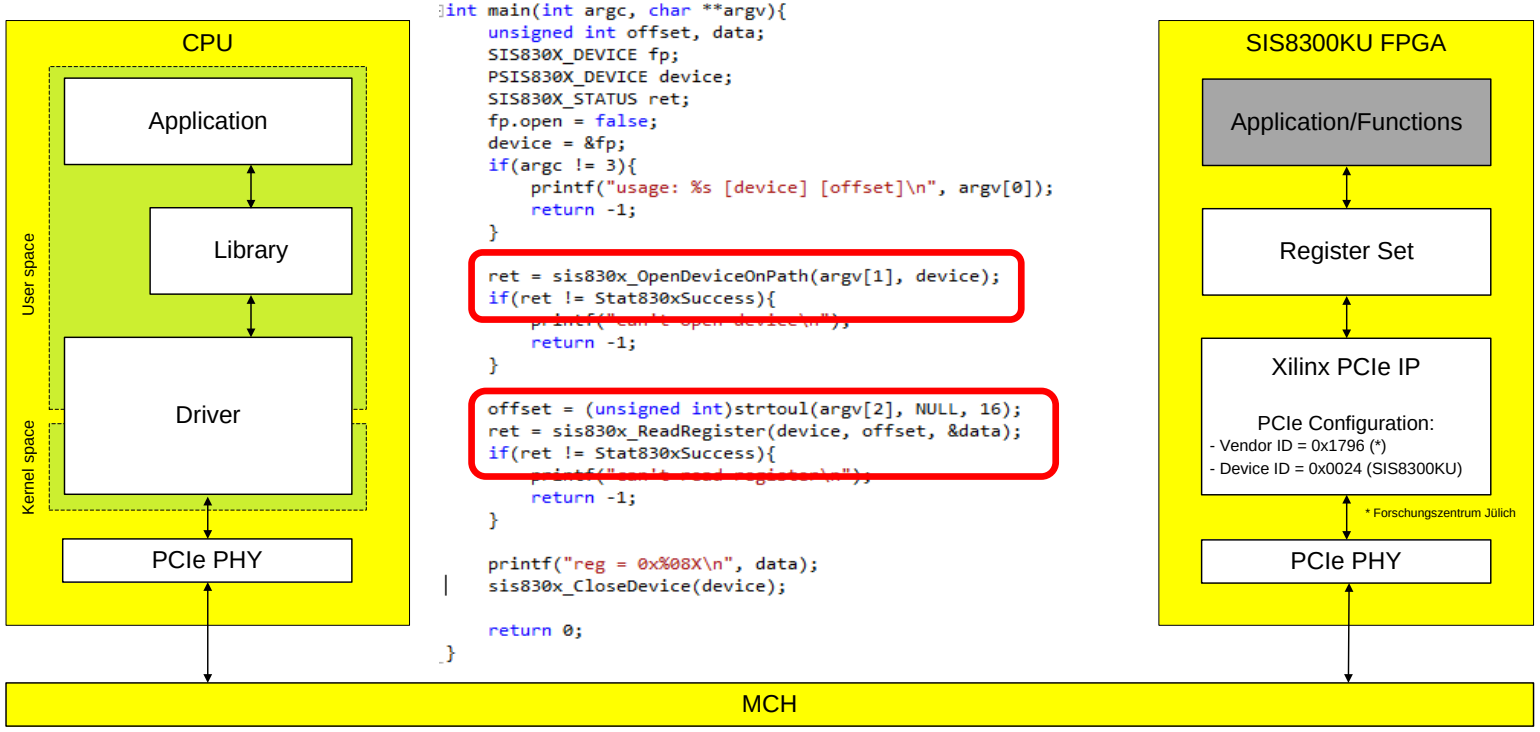
An Extract of calls of the SIS830x Library (…/lib/ libSIS830x/sis830x.h)

```
// device handling
SIS830X_STATUS sis830x_GetNumberOfDevices(int *num);
SIS830X_STATUS sis830x_OpenDeviceOnIdx(int idx, PSIS830X_DEVICE device);

SIS830X_STATUS sis8325_GetNumberOfDevices(int *num);
SIS830X_STATUS sis8325_OpenDeviceOnIdx(int idx, PSIS830X_DEVICE device);

SIS830X_STATUS sis830x_OpenDeviceOnPath(char *path, PSIS830X_DEVICE device);
SIS830X_STATUS sis830x_CloseDevice(PSIS830X_DEVICE device);



// general purpose register i/o
SIS830X_STATUS sis830x_ReadRegister(PSIS830X_DEVICE device, uint32_t addr, uint32_t *data);
SIS830X_STATUS sis830x_WriteRegister(PSIS830X_DEVICE device, uint32_t addr, uint32_t data);



// memory i/o
SIS830X_STATUS sis830x_ReadMemory(PSIS830X_DEVICE device, uint32_t addr, uint32_t *data, uint32_t len);
SIS830X_STATUS sis830x_WriteMemory(PSIS830X_DEVICE device, uint32_t addr, uint32_t *data, uint32_t len);



// ad9510 divider setup
SIS830X_STATUS sis830x_AD9510_SPI_Setup(PSIS830X_DEVICE device, unsigned int* ch_divider_configuration_array, unsigned int ad9510_synch_cmd);

// adc ad9268 setup (SIS8300/SIS8300L/SIS8300L2/SIS8300KU)
SIS830X_STATUS sis830x_ADC_AD9268_SPI_Setup(PSIS830X_DEVICE device);
SIS830X_STATUS sis830x_ADC_AD9268_SPI_Read(PSIS830X_DEVICE device, unsigned int adc_device_no, unsigned int spi_addr, unsigned int* read_data);
```

Register Read Example with a SIS83xx Library call „sis830x_ReadRegister(…)“

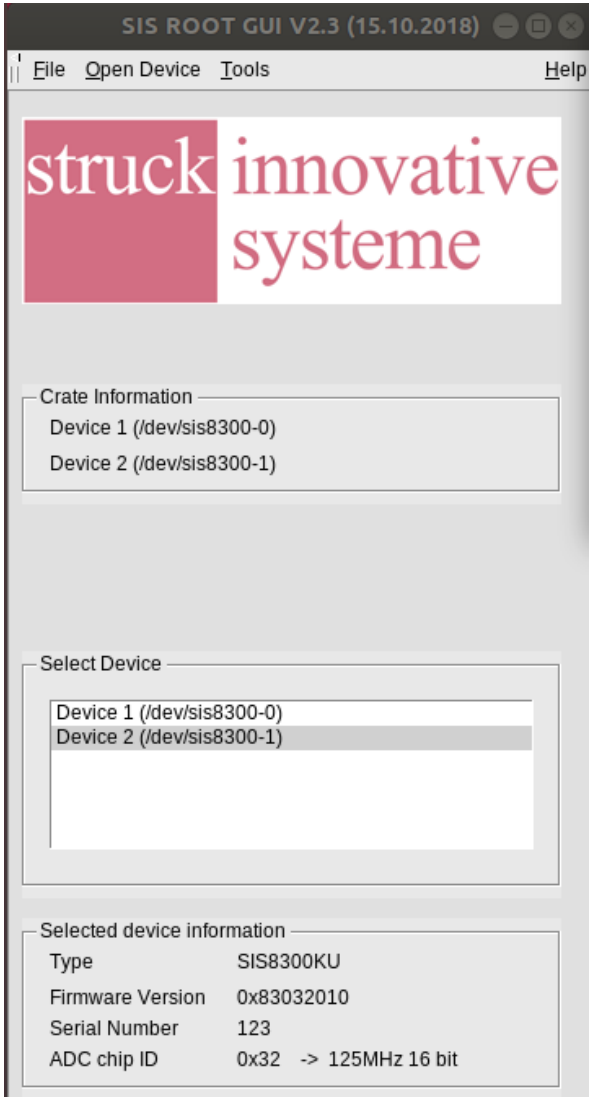
Application: \universal_tests\register\regread.c

```c
#include "../../lib/libSIS830x/sis830x.h"

int main(int argc, char **argv){
    unsigned int offset, data;
    SIS830X_DEVICE fp;
    PSIS830X_DEVICE device;
    SIS830X_STATUS ret;
    fp.open = false;
    device = &fp;
    if(argc != 3){
        printf("usage: %s [device] [offset]\n", argv[0]);
        return -1;
    }

    ret = sis830x_OpenDeviceOnPath(argv[1], device);
    if(ret != Stat830xSuccess){
        printf("can't open device\n");
        return -1;
    }

    offset = (unsigned int)strtoul(argv[2], NULL, 16);
    ret = sis830x_ReadRegister(device, offset, &data);
    if(ret != Stat830xSuccess){
        printf("can't read register\n");
        return -1;
    }

    printf("reg = 0x%08X\n", data);
    sis830x_CloseDevice(device);

    return 0;
}
```




```
th@th-CELSIUS-M740:~/SIS/testroom/universal_tests/register$ ./regread /dev/sis8300-s1 0x0
reg = 0x83031911
th@th-CELSIUS-M740:~/SIS/testroom/universal_tests/register$ ./regread /dev/sis8300-s1 0x1
reg = 0x00000081
```

SIS830x Root GUI Application

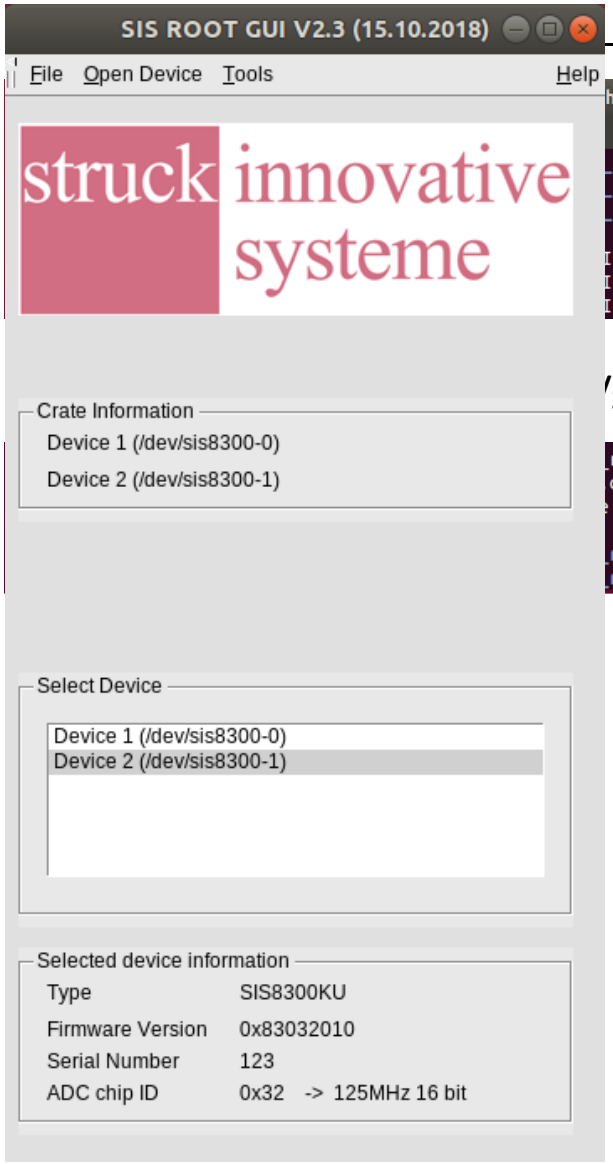


The „SIS830x Root GUI“ is a graphical application that provides access to serveral SIS ADC cards with different RTMs

The „SIS830x Root GUI“ application requires an installation of the CERN ROOT framework
→ https://root.cern.ch/downloading-root

Device	Supported
SIS8300	yes
SIS8300 v2	yes
SIS8300L	yes
SIS8300L v2	yes
SIS8300L2	yes
SIS8300KU	yes
SIS8300KU-250MHz	yes
SIS8325	yes
SIS8900	yes
DWC8300	yes
DWC8VM1	yes
DS8VM1	yes

Build the SIS830x Root GUI Application



_tests\sis_root_gui_Vx.y“ and excute „make“

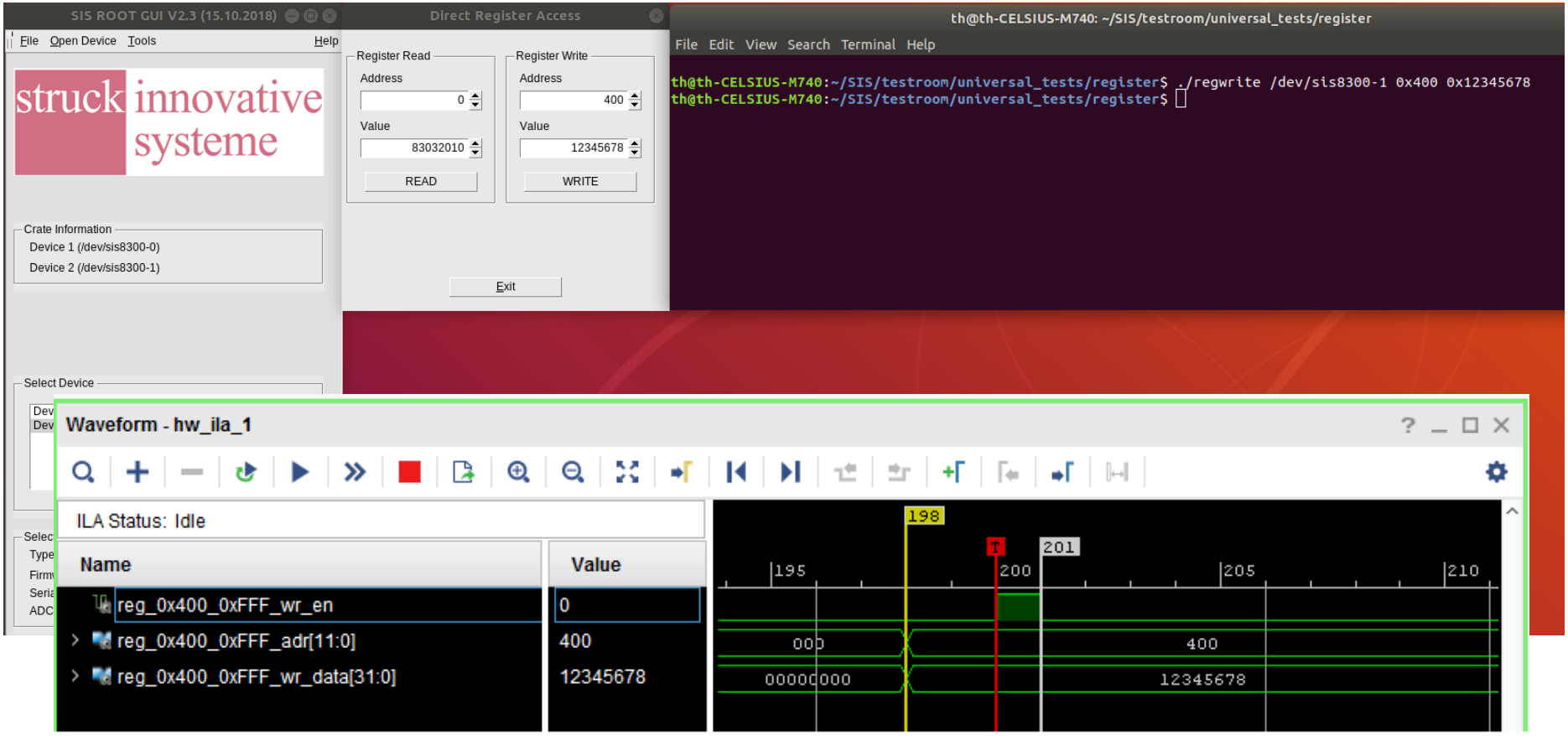
```
h@th-CELSIUS-M740: ~/SIS/testroom/universal_tests/sis_root_gui_V2.3
_root_gui_V2.3$
_root_gui_V2.3$
_root_gui_V2.3$ make

I../../lib/libSIS830x -I/home/th/root/include -c sis_rtm_device.cpp -o obj/Release/sis_rtm_device.o
I../../lib/libSIS830x -I/home/th/root/include -c sis_root_gui_tools.cpp -o obj/Release/sis_root_gui_tools.o
I../../lib/libSIS830x -I/home/th/root/include -c sis_root_gui_test1.cpp -o obj/Release/sis_root_gui_test1.o
```

/sis_root_gui“

```
_root_gui obj/Release/sis_rtm_device.o obj/Release/sis_root_gui_tools.o obj/Release/sis_root_gui_test1.o obj
o obj/Release/sis_dwc8300_class.o obj/Release/sis_dummy_rtm_class.o obj/Release/sis_ds8vm1_class.o  -lCore
 -lRint -lPostscript -lMatrix -lPhysics -lMathCore -lThread -lGui -pthread -lm -ldl -rdynamic -lfftw3 -s ..

_root_gui_V2.3$
_root_gui_V2.3$ ./sis_root_gui
```

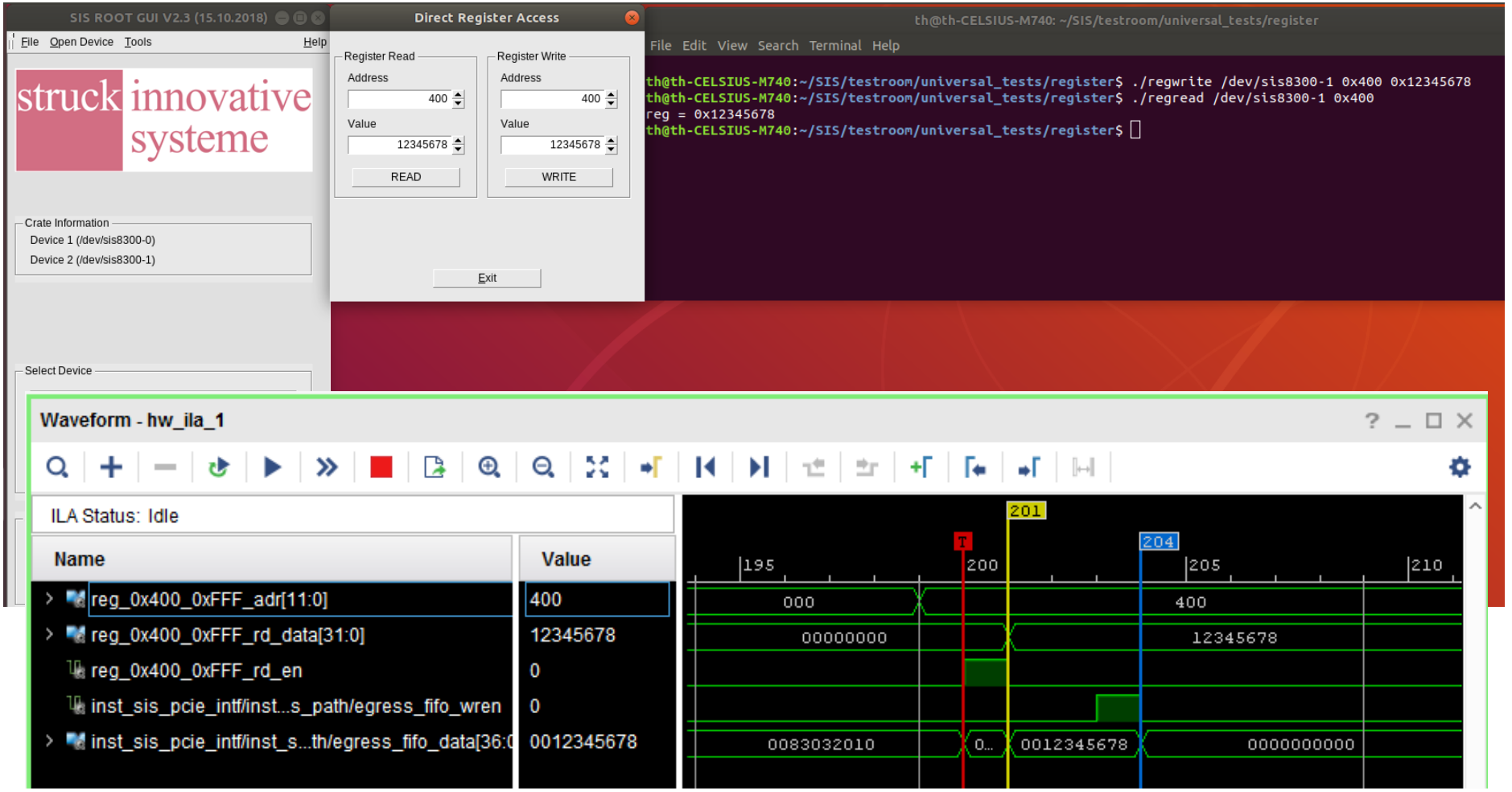
and the „main control panel“ will appear and displays all detected SIS devices

SIS830x Root GUI / Tools / Direct Register Access



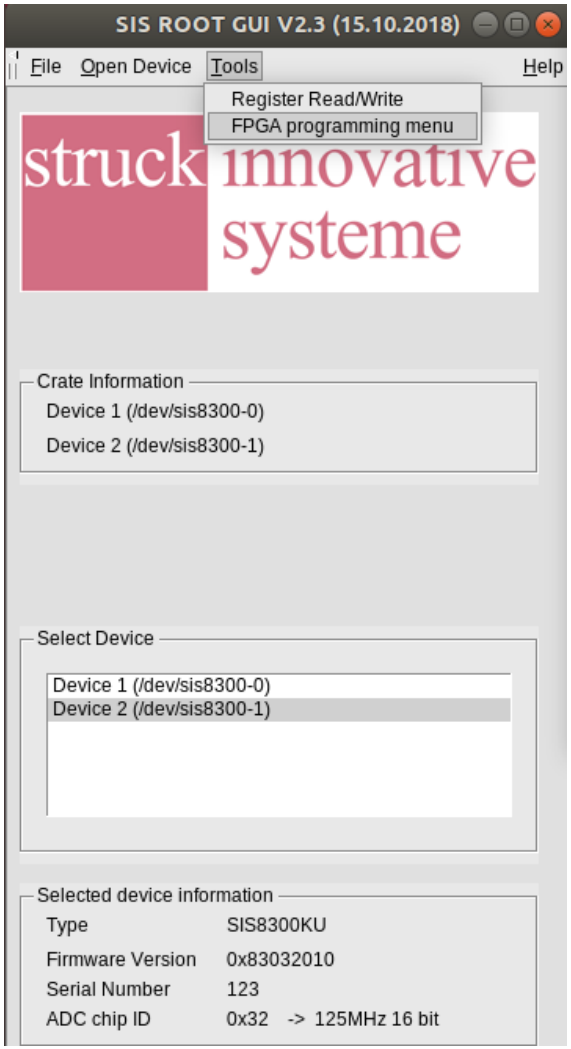
Use Vivado chipscope to observe what happens inside the FPGA with a „Register Write“

SIS830x Root GUI / Tools / Direct Register Access

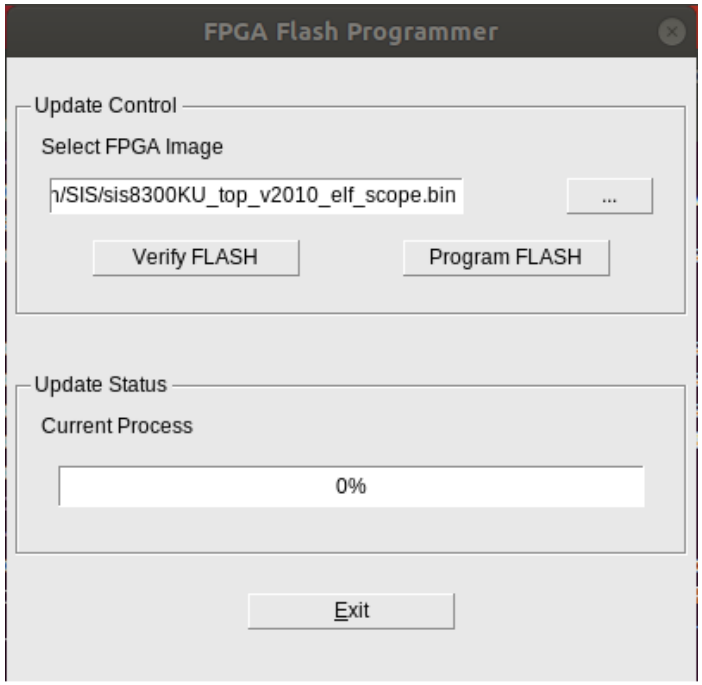


Use Vivado chipscope to observe what happens inside the FPGA with a „Register Read“

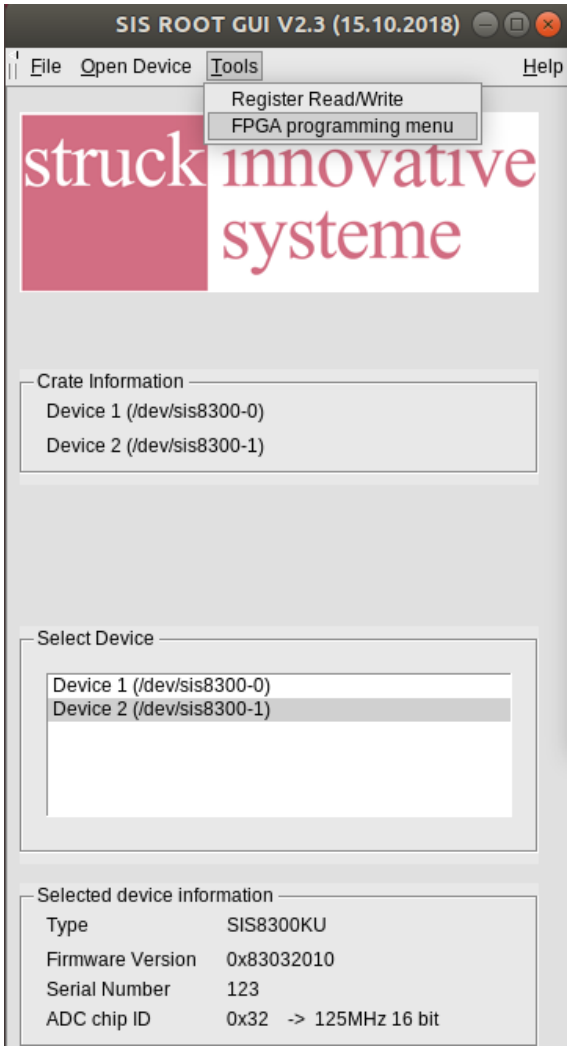
SIS830x Root GUI / Tools / FPGA programming menu



Select „Tools/FPGA programming menu“ and the „FPGA programming menu“ canvas will appear“

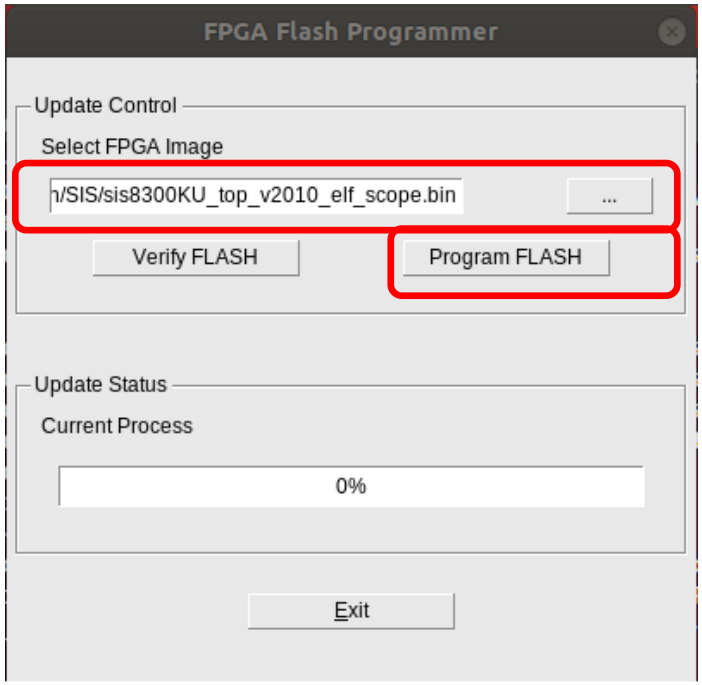


SIS830x Root GUI / Tools / FPGA programming menu

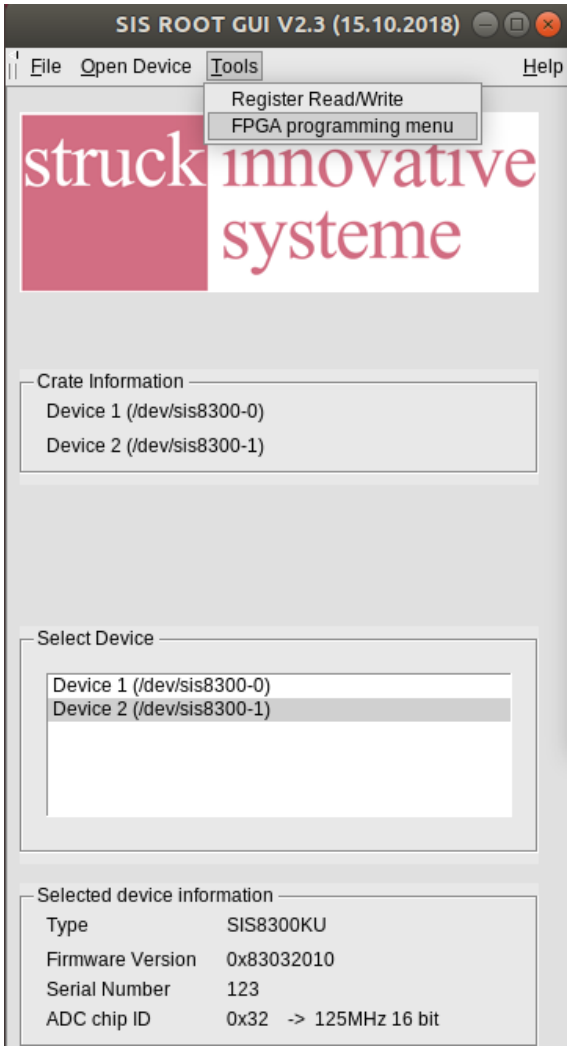


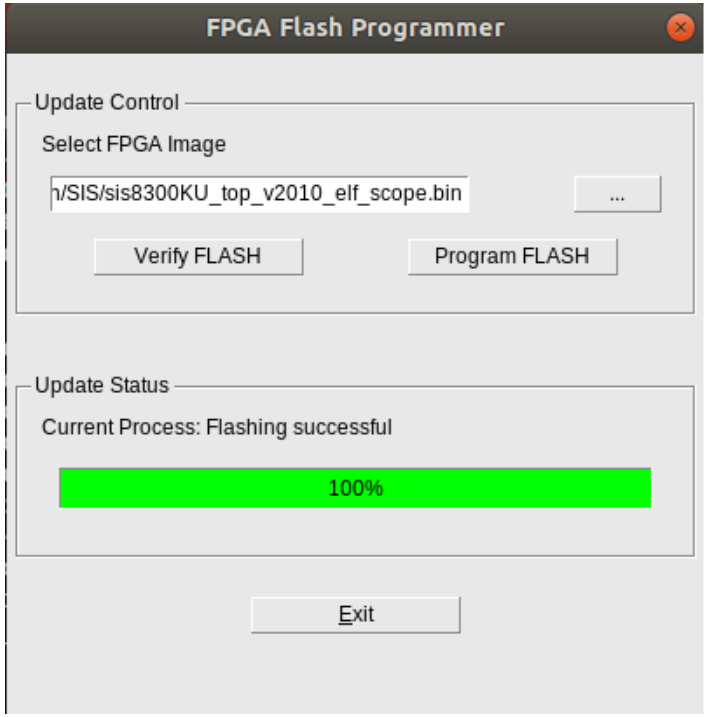
Select the „…“ button to get browser window to search for and select a FPGA image file (*.bin)

Start „Program FLASH“

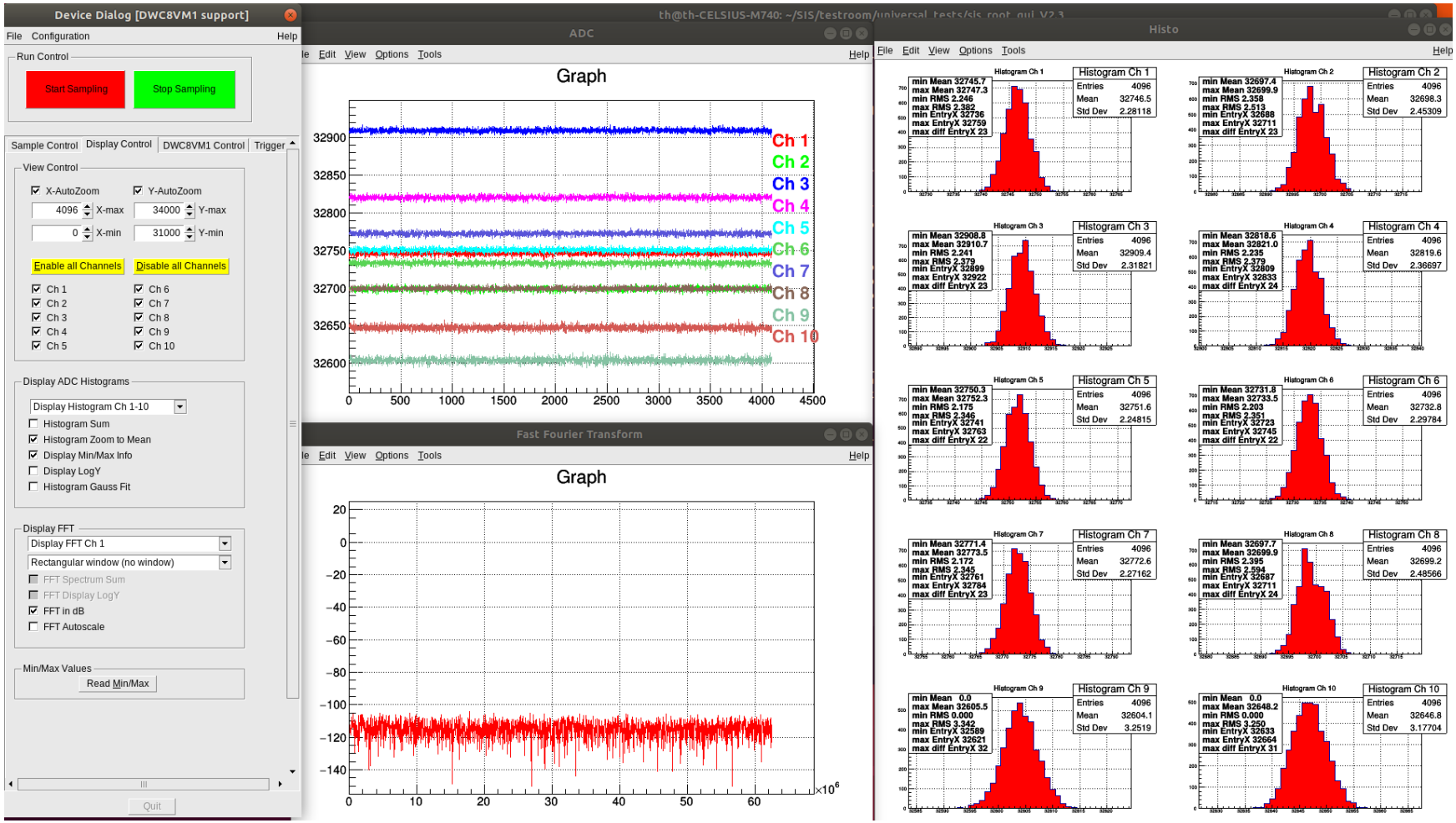


SIS830x Root GUI / Tools / FPGA programming menu

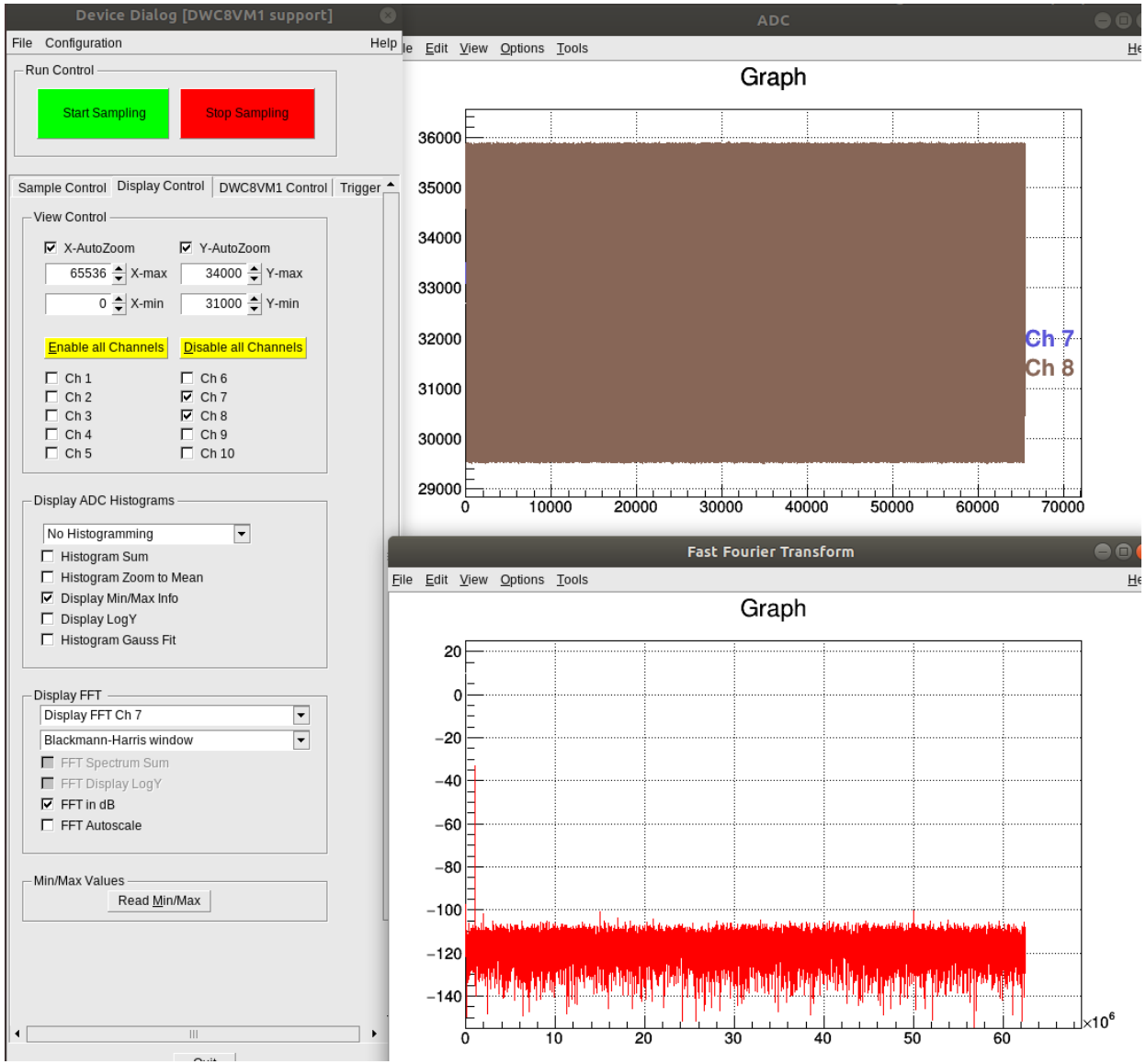




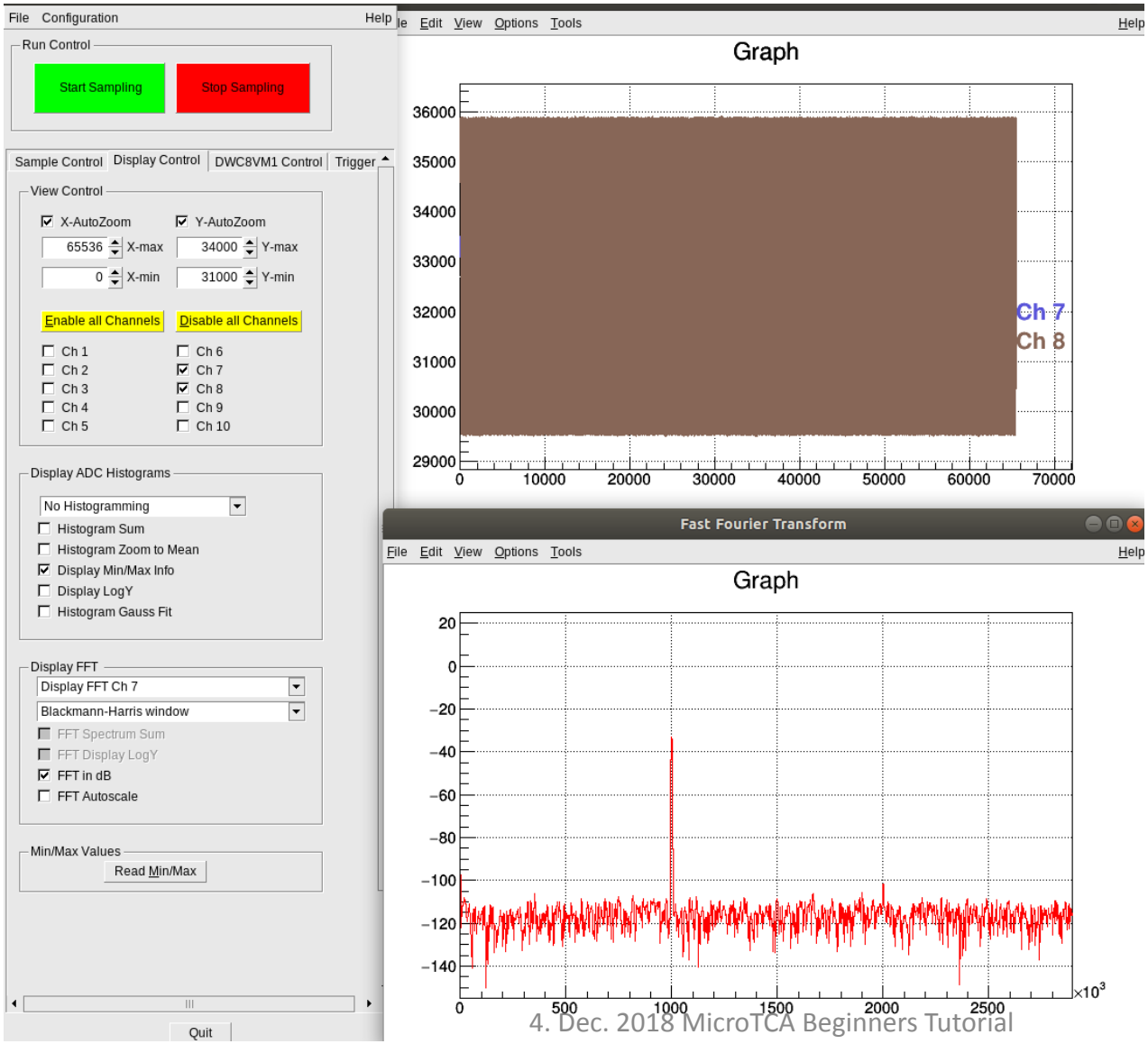
SIS830x Root GUI / Tools / Test / FFT



SIS830x Root GUI / Tools / Test / FFT



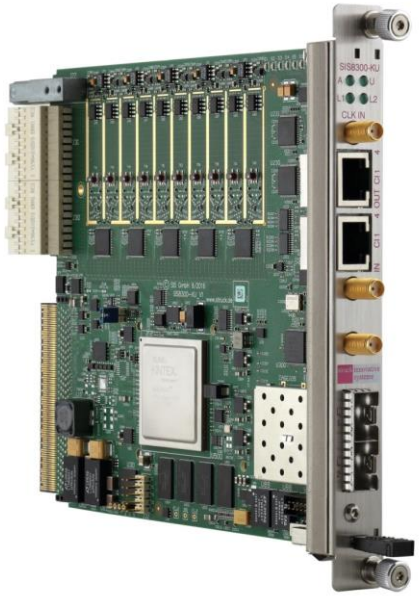
SIS830x Root GUI / Tools / Test / FFT



SIS830x Root GUI / Tools / Test with RTM DWC8VM1

DWC8VM1

SIS8300KU



LO = 700 MHz

LO: Local Oscillator input

REF: Reference input

REF = 699 MHz

VM = 699 MHz

VM: Vector modulator ouput

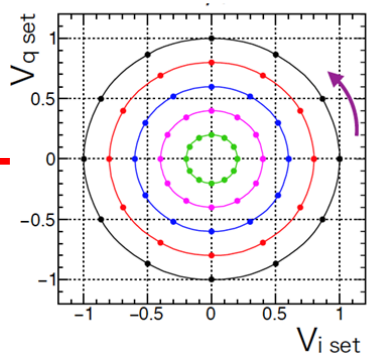
700MHz – 699 MHz = 1 MHz

Ch7: downconverted REF input

Ch8: downconverted VM output

DAC outputs I and Q

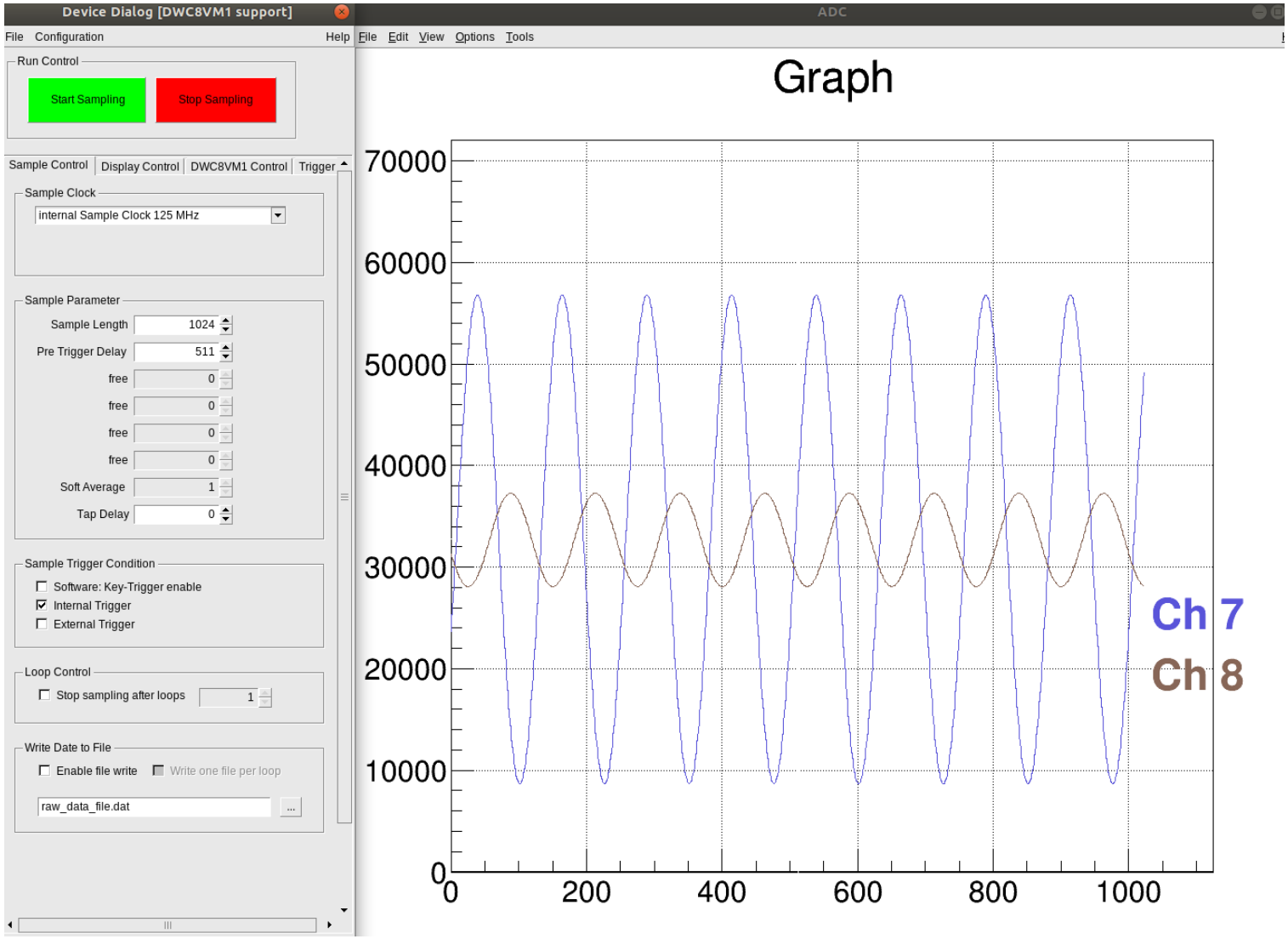
set values



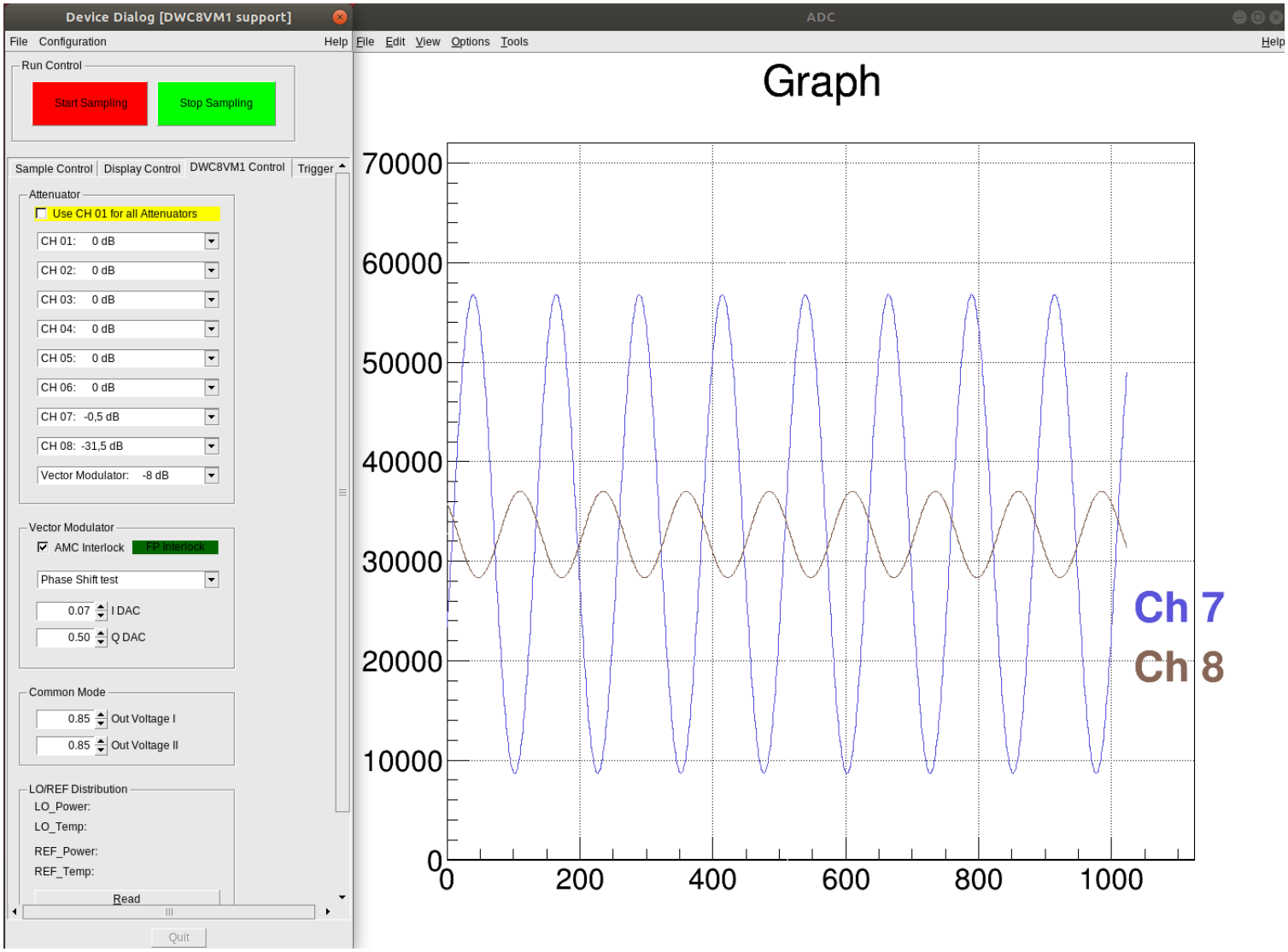
DAC outputs I and Q control the gain of VM output and the phase shifting to REF

courtesy: SPRING-8

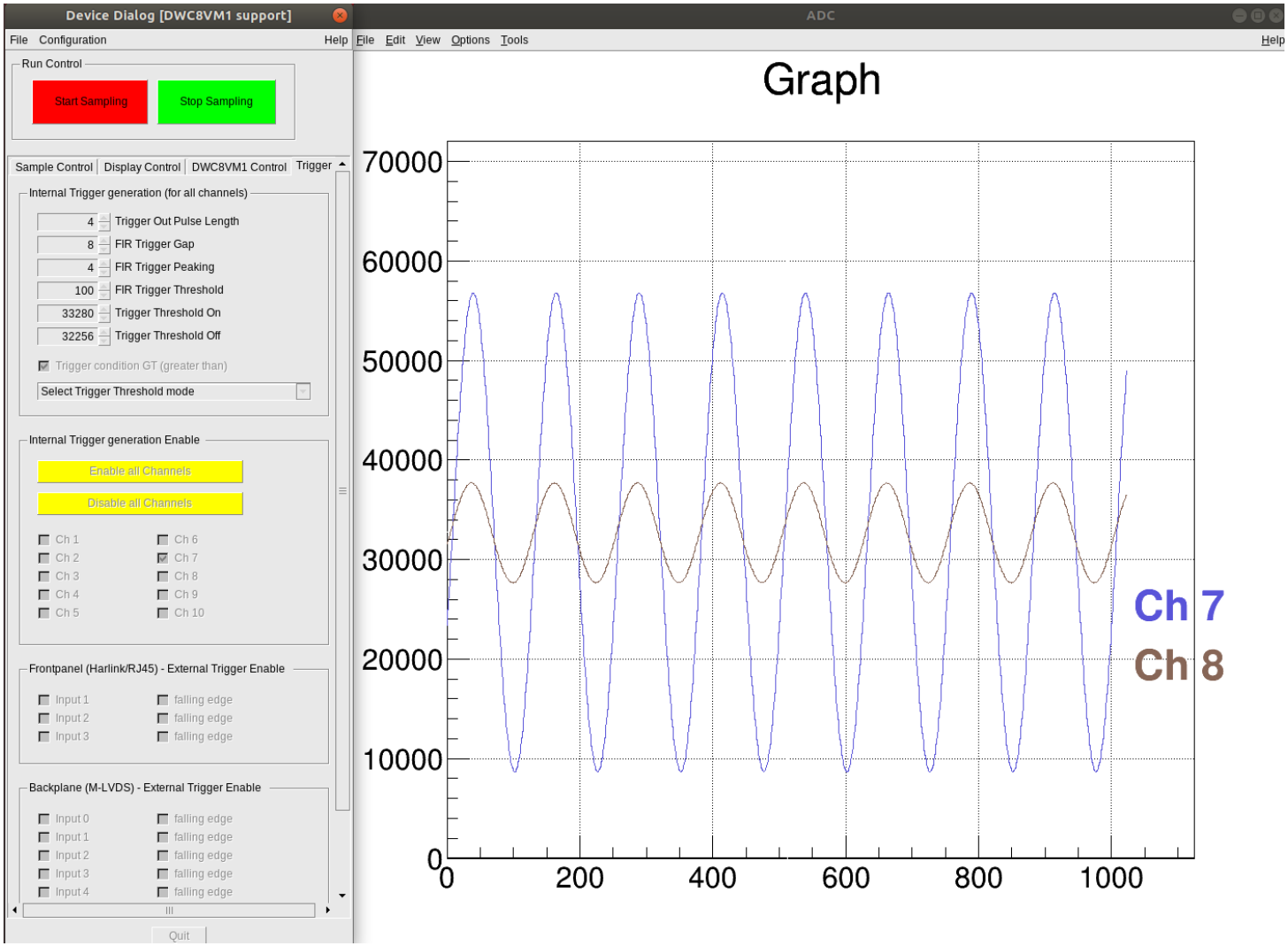
SIS830x Root GUI / Tools / Test with RTM DWC8VM1



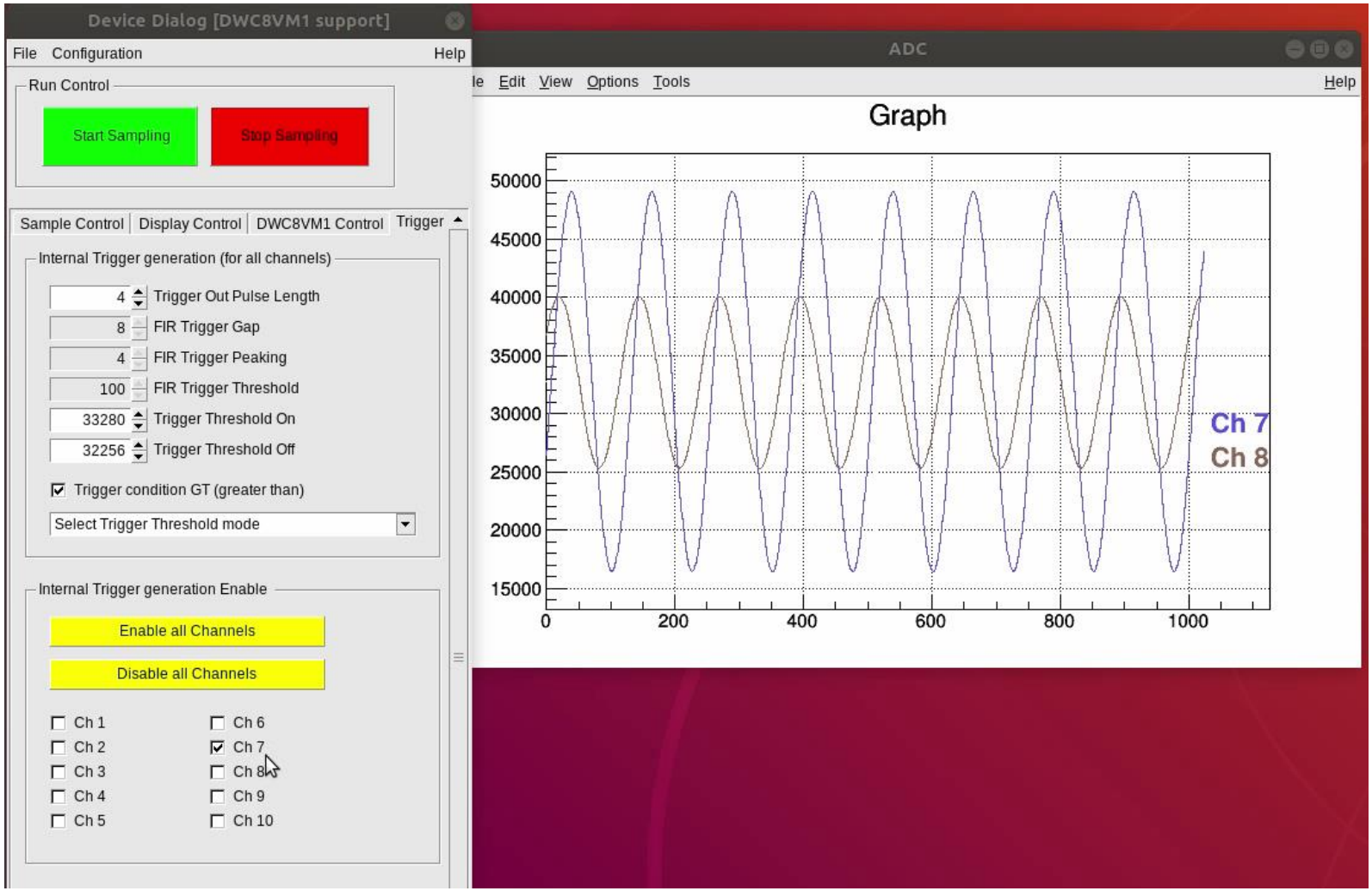
SIS830x Root GUI / Tools / Test with RTM DWC8VM1



SIS830x Root GUI / Tools / Test with RTM DWC8VM1



SIS830x Root GUI / Tools / Test with RTM DWC8VM1



Thank you!


7th MicroTCA Workshop
for Industry & Research
5 – 6 Dec 2018
CFEL, DESY, Hamburg
DESY