

Lecture 7

Design of Embedded Hardware and Firmware

Make the system work

*TSM_EmbHardw
Apr. 13, 2016*

Prof. A. Habegger
Bern University of Applied Sciences

The Goals



- ▶ Everyone knows the steps invoked to make the system work
- ▶ The SoPC works on every board
- ▶ You test the entire system including the firmware
- ▶ Among of different VHDL background, it is clear how we extend the embedded system
 - ▶ Memory Map
 - ▶ Avalon MM Slave and MM Master
 - ▶ Interfacing the cores from within FW
 - ▶ Using the user IP API

Get the Files from Moodle

Design of Embedded
Hardware and
Firmware

Prof. A. Habegger

The screenshot shows a Moodle course page with the following content:

- Course navigation bar: Access, Comparision betw, Repository - C, :: microLab :: Inst, huce:microlab:pro, Vous avez dit pro.
- User info: Sie sind ange
- File list:
 - Testbench for LCD DMA Interface
 - ctrl lcd avalonDMA
 - lab4-sol
 - Files for camera
- Text: You must generate a clock at 24 MHz from your PLL to be sent to the camera.
- Link: Your VHDL and C source for the LCD ctrl

Thema 7

Week 7 (Updated 13-04-2016)

This week we are going to spend the first half of the course discussing the entire system. During the second half you will either do the term exam or working on the embedded system to make it work.



[finalSobelFilter](#)



[Upload the sources of your project \(VHDL + C + qsys + TCL files\)](#)



Bern University
of Applied Sciences

Intro

Prerequisite

Quartus

QSys

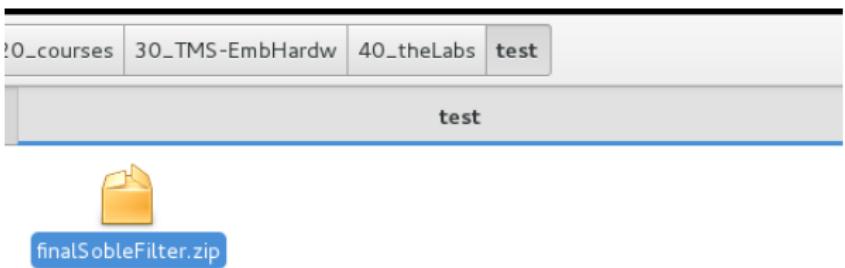
Program

Nios II SDK

System

- ▶ Login to Moodle and download the ZIP file. (*finalSobelFilter*)
- ▶ Save the ZIP somewhere on yours system disk

Unpack the archive



Bern University
of Applied Sciences

Intro

Prerequisite

Quartus

QSys

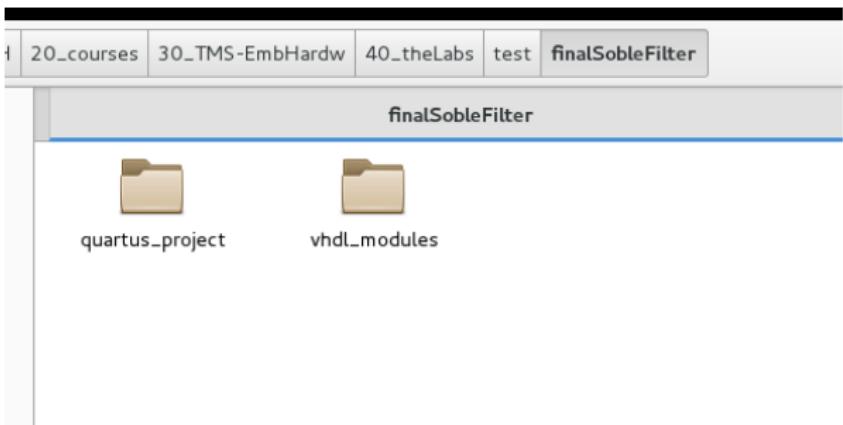
Program

Nios II SDK

System

- ▶ Navigate to the location where the files are stored
- ▶ Unpack the ZIP archive

Unpack the archive



- ▶ Navigate to the location where the files are stored
- ▶ Unpack the ZIP archive
- ▶ The unpacked archive will look as shown in figure above



Browse the Folders

- Open a terminal and switch to the location where the files are (`cd $LOCATION`).
- The following files are in the sub-folder `vhdl_modules` and `quartus_project`, respectively

```
└── vhdl_modules
    ├── camera_controller
    │   ├── cam_avalon_slave_behavior.vhdl
    │   ├── cam_avalon_slave_entity.vhdl
    │   ├── cam_dma_behaviour.vhdl
    │   ├── cam_dma_ctrl_behaviour.vhdl
    │   ├── cam_dma_ctrl_entity.vhdl
    │   ├── cam_dma_entity.vhdl
    │   ├── frame_interpreter_behavior.vhdl
    │   ├── frame_interpreter_entity.vhdl
    │   ├── pixel_interface_behavior.vhdl
    │   ├── pixel_interface_entity.vhdl
    │   ├── syncflop_behavior.vhdl
    │   └── syncflop_entity.vhdl
    ├── i2c_core
    │   ├── i2c_autodetect_behavior.vhdl
    │   ├── i2c_autodetect_entity.vhdl
    │   ├── i2c_avalon_slave_behavior.vhdl
    │   ├── i2c_avalon_slave_entity.vhdl
    │   ├── i2c_ctrl_behavior.vhdl
    │   ├── i2c_ctrl_entity.vhdl
    │   ├── i2c_data_behavior.vhdl
    │   ├── i2c_data_entity.vhdl
    │   ├── i2c_start_stop_behavior.vhdl
    │   └── i2c_start_stop_entity.vhdl
    ├── lcd_controller
    │   ├── dma_controller_behavior.vhdl
    │   ├── dma_controller_entity.vhdl
    │   ├── lcd_avalon_slave_behavior.vhdl
    │   ├── lcd_avalon_slave_entity.vhdl
    │   ├── lcd_dma_behavior.vhdl
    │   ├── lcd_dma_entity.vhdl
    │   ├── pixel_formatter_behavior.vhdl
    │   ├── pixel_formatter_entity.vhdl
    │   ├── send_receive_if_behavior.vhdl
    │   └── send_receive_if_entity.vhdl
    └── vga_controller
        ├── delay_line_behavior.vhdl
        └── delay_line_entity.vhdl
```

```
└── quartus_project
    ├── base_system.qsys
    ├── base_system.sopinfo
    ├── cam_dma_hw.tcl
    ├── i2c_master_hw.tcl
    ├── lcd_dma_hw.tcl
    ├── mse_demo.bdf
    ├── mse_demo.qpf
    ├── pins.tcl
    └── software
        └── sobel
            ├── create-this-app
            ├── Makefile
            └── readme.txt
                └── src
                    ├── camera.c
                    ├── camera.h
                    ├── dipswitch.c
                    ├── dipswitch.h
                    ├── grayscale.c
                    ├── grayscale.h
                    ├── i2c.c
                    ├── i2c.h
                    ├── lcd_simple.c
                    ├── lcd_simple.h
                    ├── main.c
                    ├── mt9d112.h
                    ├── sobel.c
                    ├── sobel.h
                    ├── vga.c
                    └── vga.h
            └── sobel_bsp
                ├── create-this-bsp
                ├── Makefile
                └── settings.bsp
                    └── uio_dsp_hw.tcl
```

Run Quartus

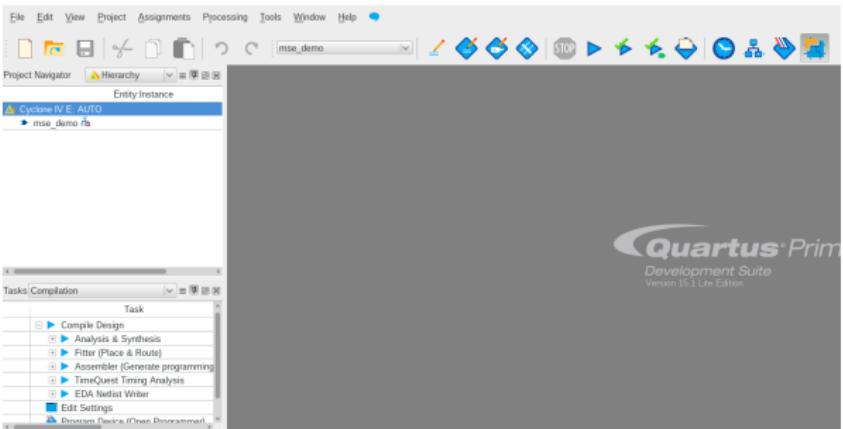


```
hpx3@qvel-hpx3: ~/BFH20_courses/30_THS-EmlHardware/40_theLab/test/finalSobelFilter/quartus_project
File Edit View Search Terminal Help
hpx3@qvel-hpx3: ~/BFH20_courses/30_THS-EmlHardware/40_theLab/test/finalSobelFilter/quartus_project$ Quartus $ "quartus mse_demo.qpf"
```

- Intro
- Prerequisite
- Quartus
- QSys
- Program
- Nios II SDK
- System

- ▶ Run quartus. To do so run the command *altera_shell* first
- ▶ If the command is not available run the file *nios2_command_shell.sh*, which is located in altera installation folder ↪ nios2eds
- ▶ Run quartus with the argument *\$PROJ_NAME.qps*, where *\$PRO_NAME* is the projects name.
- ▶ More details are shown in figure above.

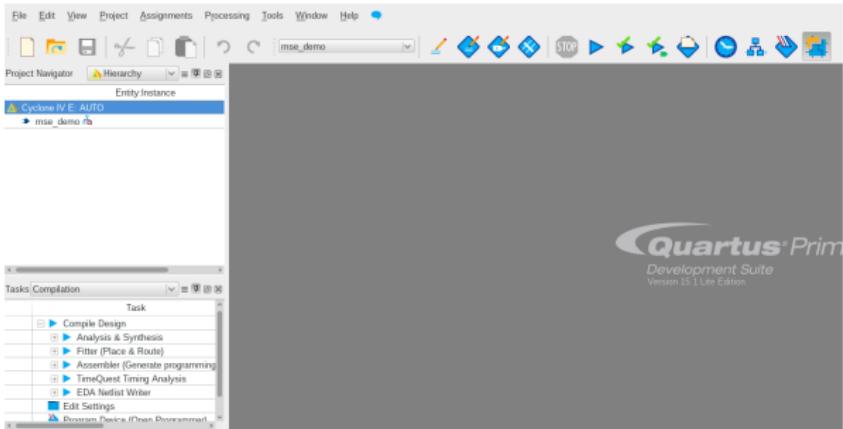
Run Quartus



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System

- Run quartus. To do so run the command *altera_shell* first
- If the command is not available run the file *nios2_command_shell.sh*, which is located in altera installation folder ↗ nios2eds
- Run quartus with the argument \$PROJ_NAME.qps, where \$PRO_NAME is the projects name.
- More details are shown in figure above.

Apply Correct Device Settings



Bern University
of Applied Sciences

Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System

- ▶ Apply correct device settings.
- ▶ Right click on the instance and tick settings

Apply Correct Device Settings

Device

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.

To determine the version of the Quartus Prime software in which your target device is supported, refer to the [Device Support List](#) webpage.

Device family

Family: Cyclone IV E Show in 'Available devices' list

Devices: All

Package: Any Core Speed grade: Any

Pin count: Any

Name filter:

Target device

Auto device selected by the Fitter

Specific device selected in 'Available devices' list

Other: n/a

Show advanced devices

[Device and Pin Options...](#)

Available devices:

Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Embedded multiplier 9-bit elements
EP4CE6E22A7	1.2V	6272	92	92	276480	30
EP4CE6E22C6	1.2V	6272	92	92	276480	30
EP4CE6E22C7	1.2V	6272	92	92	276480	30
EP4CE6E22C8	1.2V	6272	92	92	276480	30
EP4CE6E22C8L	1.0V	6272	92	92	276480	30
EP4CE6E22C9L	1.0V	6272	92	92	276480	30
EP4CE6E22I7	1.2V	6272	92	92	276480	30
EP4CE6E22I8L	1.0V	6272	92	92	276480	30
EP4CE6E22I8T	1.2V	6272	180	180	276480	30

Migration Devices... 0 migration devices selected

[Buy Software](#) OK Cancel Help

- ▶ Apply correct device settings.
- ▶ Right click on the instance and tick settings
- ▶ An overview of all available device configurations is listed

Apply Correct Device Settings



Intro

Prerequisite

Quartus

QSys

Program

Nios II SDK

System

Device

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.

To determine the version of the Quartus Prime software in which your target device is supported, refer to the [Device Support List](#) webpage.

Device family

Family: Cyclone IV E

Devices: All

Show in 'Available devices' list

Package: Any

Pin count: Any

Core Spd grade: Any

Name filter:

Target device

Auto device selected by the Filter

Specific device selected in 'Available devices' list

Other: n/a

Show advanced devices

[Device and Pin Options...](#)

Available devices:

Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Embedded multiplier 9-bit elements
EP4CE22U14I7	1.2V	22320	154	154	608256	132
EP4CE30F19A7	1.2V	28848	193	193	608256	132
EP4CE30F23A7	1.2V	28848	329	329	608256	132
EP4CE30F23C6	1.2V	28848	329	329	608256	132
EP4CE30F23C7	1.2V	28848	329	329	608256	132
EP4CE30F23C8	1.2V	28848	329	329	608256	132
EP4CE30F23C8L	1.0V	28848	329	329	608256	132
EP4CE30F23C9L	1.0V	28848	329	329	608256	132
EP4CE30F23I7	1.2V	28848	329	329	608256	132

Migration Devices... 0 migration devices selected

[Buy Software](#) OK Cancel Help

- Apply correct device settings.
- Right click on the instance and tick settings
- An overview of all available device configurations is listed
- Select the device mounted on the board (EP4CE30F237)
- Confirm the settings by click on OK button

Apply Correct Device Settings

Design of Embedded
Hardware and
Firmware

Prof. A. Habegger



Bern University
of Applied Sciences

Entity:Instance

Cyclone IV E: EP4CE30F23C7

▶ mse_demo

Intro

Prerequisite

Quartus

QSys

Program

Nios II SDK

System

- ▶ Check if correct device setting was applied
- ▶ Click on the project named “mse_demo” to open

Apply Correct Device Settings



- ▶ The top system block is shown in figure above
- ▶ Pins are not assigned yet
- ▶ Open QSys
- ▶ Select the SoPC configuration file (base_system.qsys)

Intro

Prerequisite

Quartus

QSys

Program

Nios II SDK

System

Apply Correct Device Settings



Intro

Prerequisite

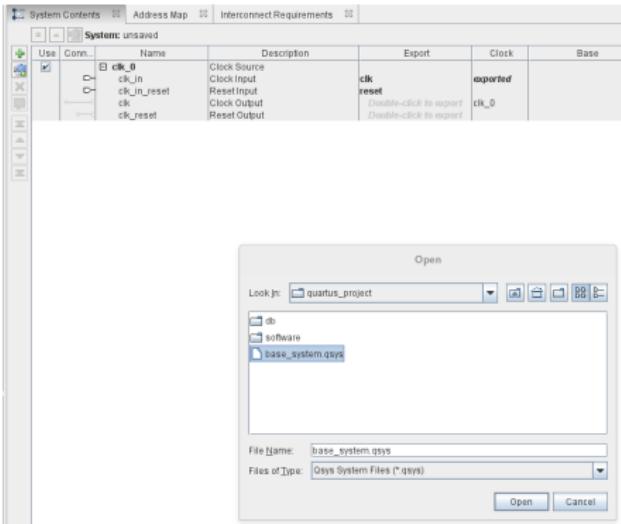
Quartus

QSys

Program

Nios II SDK

System



- ▶ The top system block is shown in figure above
- ▶ Pins are not assigned yet
- ▶ Open QSys
- ▶ Select the SoPC configuration file (base_system.qsys)

Build the SoPC

Design of Embedded
Hardware and
Firmware

Prof. A. Habegger



Bern University
of Applied Sciences

Intro

Prerequisite

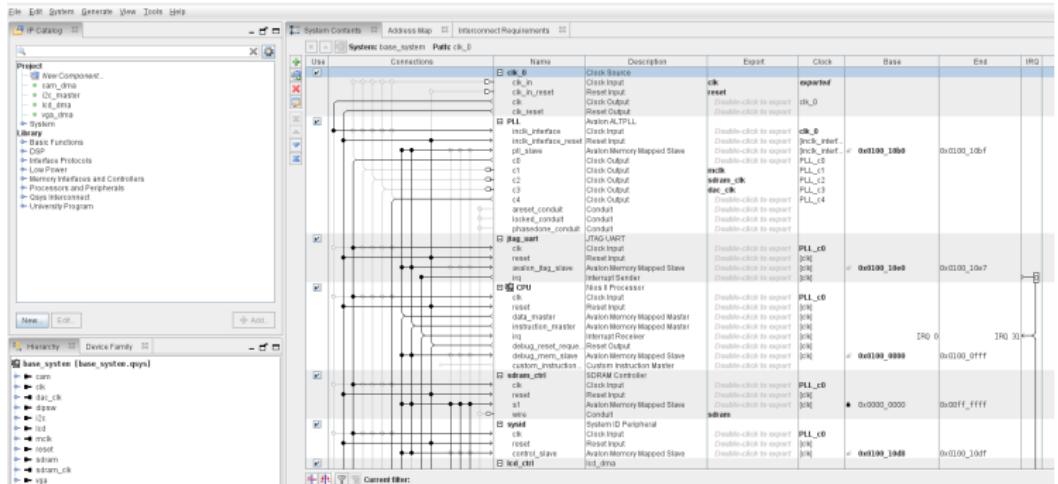
Quartus

QSys

Program

Nios II SDK

System



- ▶ Compare it loaded system looks like the one shown in figure
- ▶ Click on the button Generate HDL...
- ▶ Click Generate
- ▶ Wait...
- ▶ Switch back to Quartus

Build the SoPC



Intro

Prerequisite

Quartus

QSys

Program

Nios II SDK

System

Generation

Synthesis
Synthesis files are used to compile the system in a Quartus Prime project.
Create HDL design files for synthesis: **VHDL**

Create timing and resource estimates for third-party EDA synthesis tools.

Create block symbol file (.bsf)

Simulation
The simulation model contains generated HDL files for the simulator, and may include simulation-only features.
Simulation scripts for this component will be generated in a vendor-specific sub-directory in the specified output directory.
Follow the guidance in the generated simulation scripts about how to structure your design's simulation scripts and how to use the *ip-setup-simulation* and *ip-make-simscript* command-line utilities to compile all of the files needed for simulating all of the IP in your design.
Create simulation model:

Allow mixed-language simulation
Enable this if your simulator supports mixed-language simulation.

Output Directory
Path:

- ▶ Compare it loaded system looks like the one shown in figure
- ▶ Click on the button Generate HDL...
- ▶ Click Generate
- ▶ Wait...
- ▶ Switch back to Quartus



Intro

Prerequisite

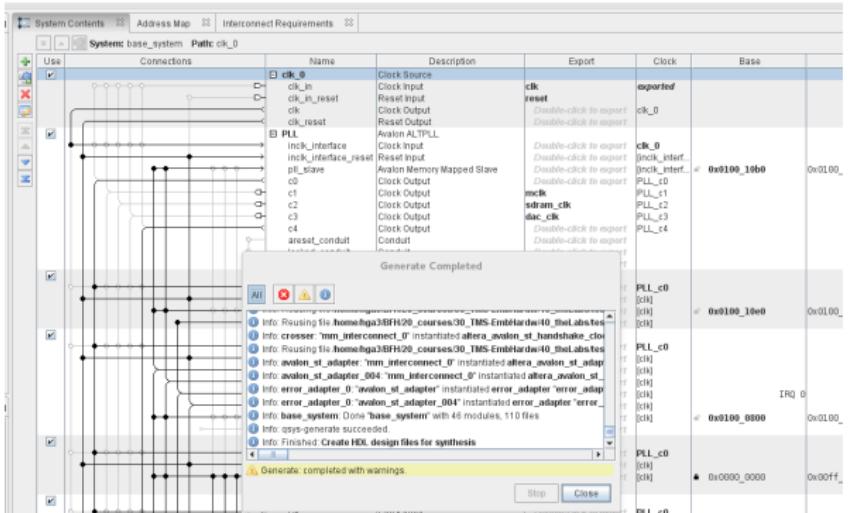
Quartus

QSys

Program

Nios II SDK

System

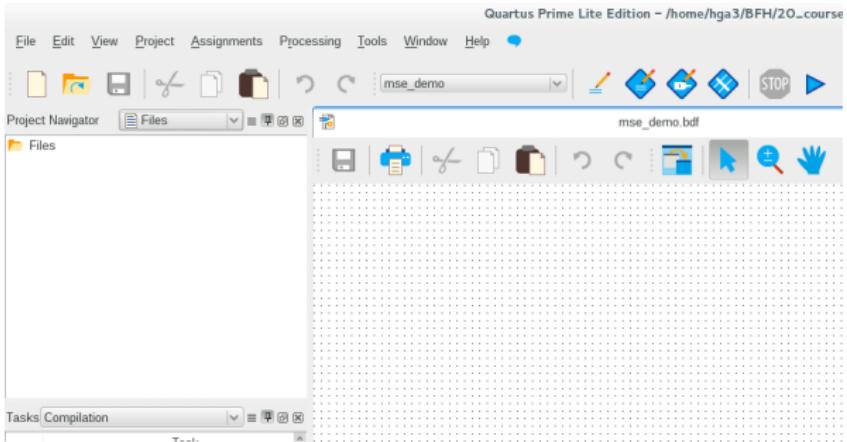


- ▶ Compare it loaded system looks like the one shown in figure
- ▶ Click on the button Generate HDL...
- ▶ Click Generate
- ▶ Wait...
- ▶ Switch back to Quartus

Add the Files



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System

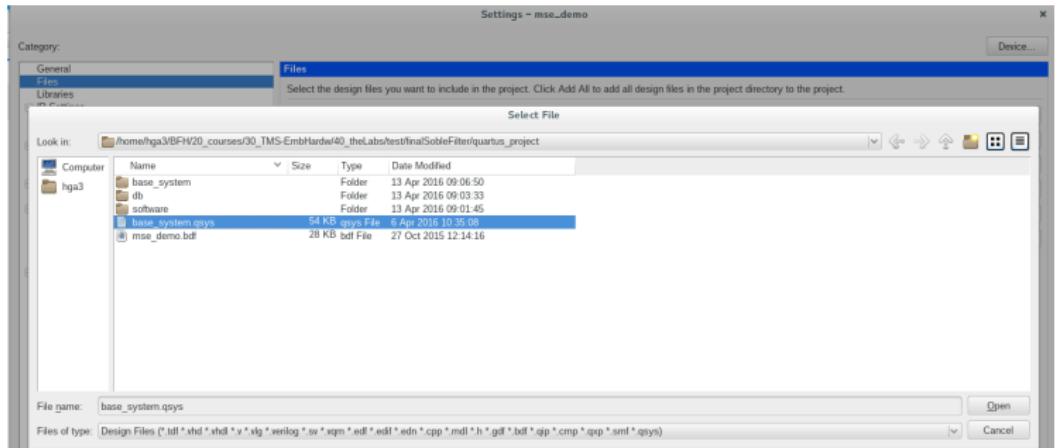


- ▶ Switch to **Files** in project navigator
- ▶ Add additional files (right click and select add...)
- ▶ Select the ***.qsys** and **mse_demo.bdf** files
- ▶ Add both files (one after the other...)
- ▶

Add the Files

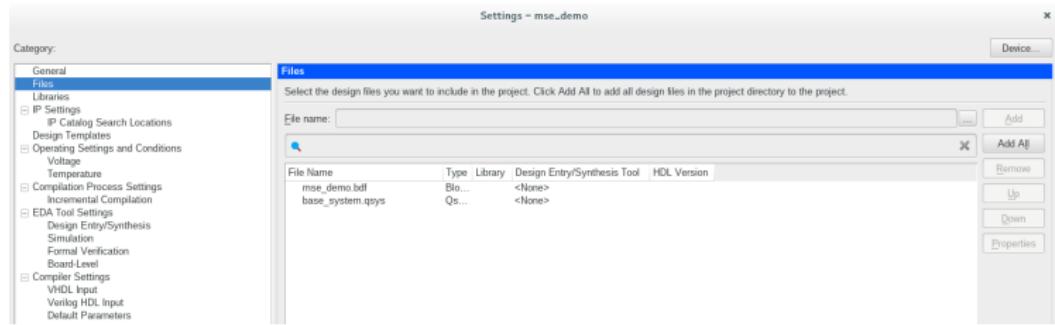


Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System



- ▶ Switch to **Files** in project navigator
- ▶ Add additional files (right click and select add...)
- ▶ Select the ***.qsys** and **mse_demo.bdf** files
- ▶ Add both files (one after the other...)
- ▶

Add the Files



- ▶ Switch to **Files** in project navigator
- ▶ Add additional files (right click and select add...)
- ▶ Select the ***.qsys** and ****mse_demo.bdf**** files
- ▶ Add both files (one after the other...)
- ▶

Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System



Intro

Prerequisite

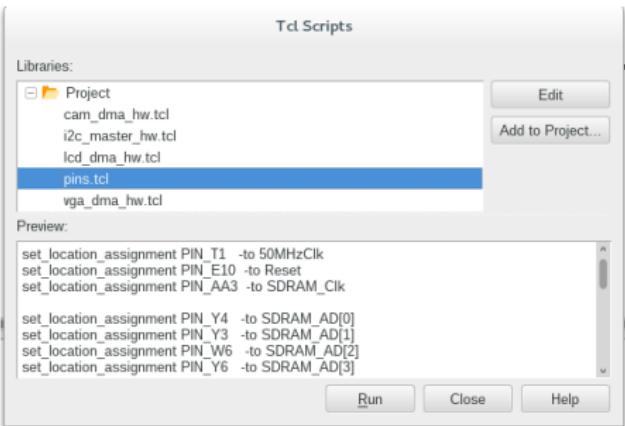
Quartus

QSys

Program

Nios II SDK

System

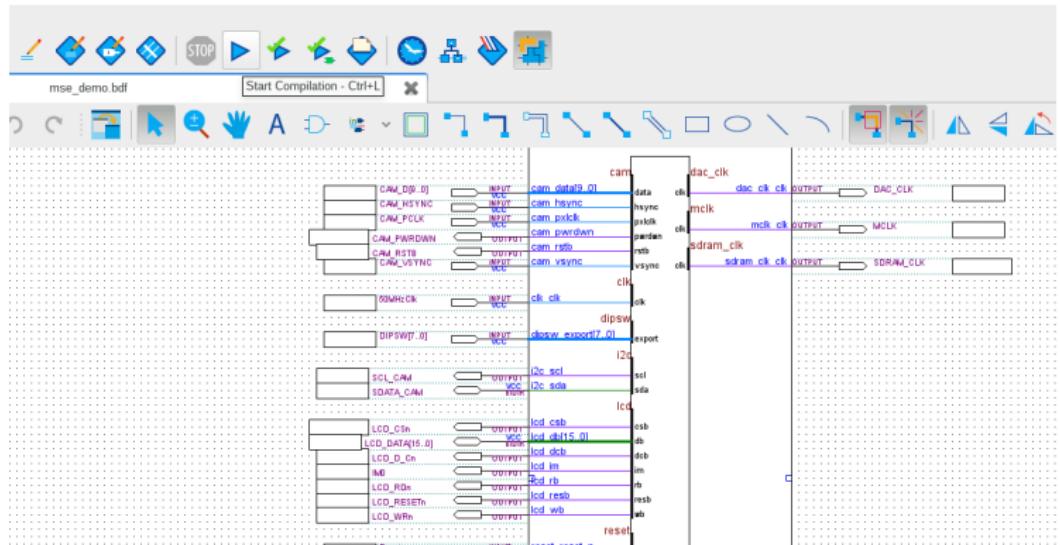


- ▶ Run the TCL script `pins.tcl` for pin assignment
- ▶ The file is located in the project folder
- ▶ Check if all pins have been assigned
- ▶ Your system should look like the one in figure above

TCL Scripting



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System



- ▶ Run the TCL script `pins.tcl` for pin assignment
- ▶ The file is located in the project folder
- ▶ Check if all pins have been assigned
- ▶ Your system should look like the one in figure above

Build the System



Intro

Prerequisite

Quartus

QSys

Program

Nios II SDK

System

The screenshot shows the Quartus Prime software interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, Help, and a toolbar with various icons. The Project Navigator on the left lists 'Files' with 'mse_demo.bdf' and 'base_system.qsys'. The main window displays the 'Flow Summary' for the project 'mse_demo.bdf'. The summary table provides detailed statistics about the design, such as flow status, revision information, and logic element counts. Below the summary is a 'Tasks Compilation' pane listing the build steps: 'Compile Design', 'Analysis & Synthesis', 'Fitter (Place & Route)', 'Assembler (Generate programming)', 'TimeQuest Timing Analysis', 'EDA Netlist Writer', and 'Edit Settings'. The bottom pane is the 'Quartus Prime Tcl Console' showing command-line interactions.

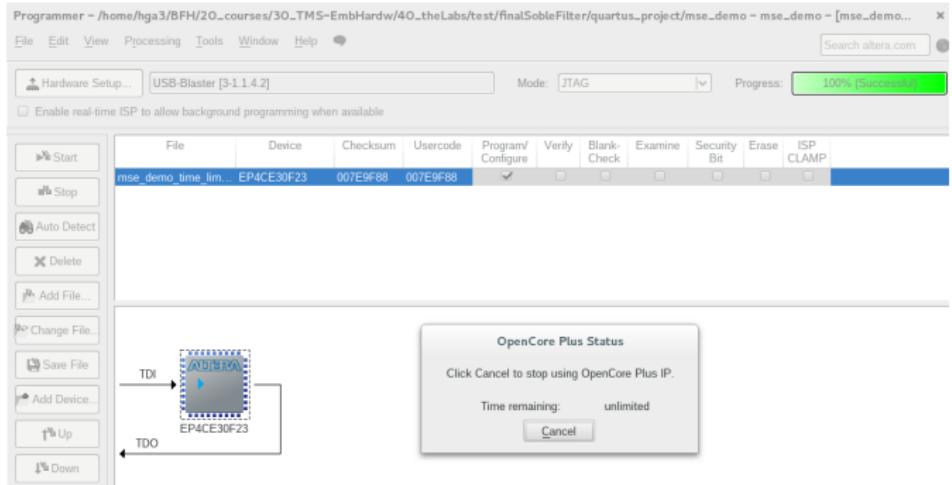
Category	Value
Flow Status	Successful - Wed Apr 13 09:12:17 2016
Quartus Prime Version	15.1.1 Build 189 12/02/2015 SJ Lite Edition
Revision Name	mse_demo
Top-level Entity Name	mse_demo
Family	Cyclone IV E
Device	EP4CE30F23C7
Timing Models	Final
Total logic elements	9,636 / 28,848 (33 %)
Total combinational functions	8,543 / 28,848 (30 %)
Dedicated logic registers	5,726 / 28,848 (20 %)
Total registers	5794
Total pins	121 / 329 (37 %)
Total virtual pins	0
Total memory bits	157,696 / 608,256 (26 %)
Embedded Multiplier 9-bit elements	6 / 132 (5 %)
Total PLLs	1 / 4 (25 %)

- ▶ Run the “Compile Design” process (click on the blue triangle)
- ▶ After successful compilation program the board
- ▶ Run the quartus programmer and select the target to program as well as the configuration file

Build the System

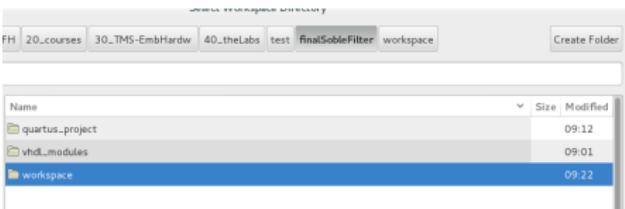


Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System



- ▶ Run the “Compile Design” process (click on the blue triangle)
- ▶ After successful compilation program the board
- ▶ Run the quartus programmer and select the target to program as well as the configuration file
- ▶ The configuration is time limited make sure you **not** click on cancel

Start SDK and select Workspace



- ▶ Start Nios II SDK
- ▶ Select a workspace (to do so create new folder ↪ create it in the finalSobelFilter directory)
- ▶ Choose the created folder as current workspace

Intro

Prerequisite

Quartus

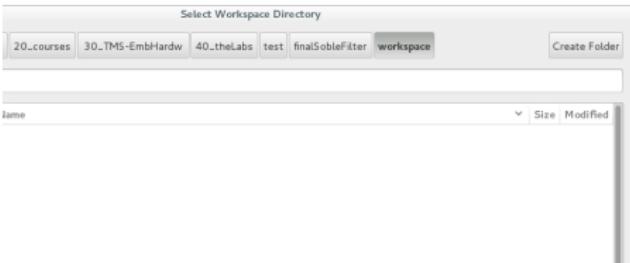
QSys

Program

Nios II SDK

System

Start SDK and select Workspace



- ▶ Start Nios II SDK
- ▶ Select a workspace (to do so create new folder ↪ create it in the finalSobelFilter directory)
- ▶ Choose the created folder as current workspace

Intro

Prerequisite

Quartus

QSys

Program

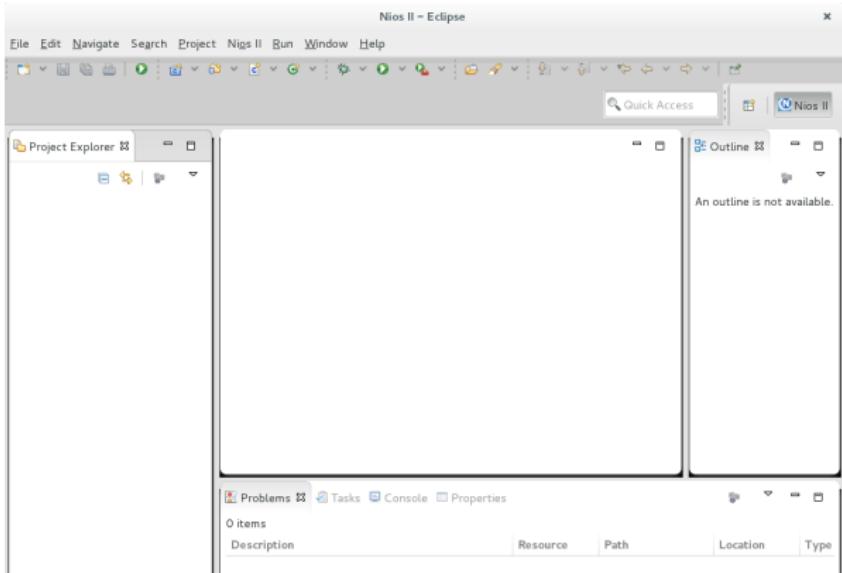
Nios II SDK

System

Import Firmware



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System

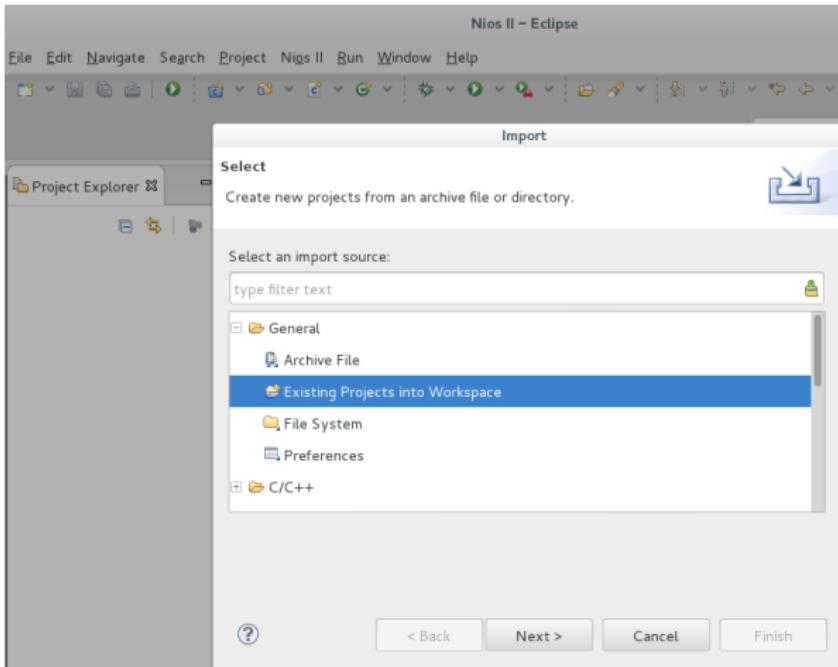


- ▶ Right click in project explorer
- ▶ Select Existing Projects
- ▶ Select the software folder located in quartus_project folder

Import Firmware

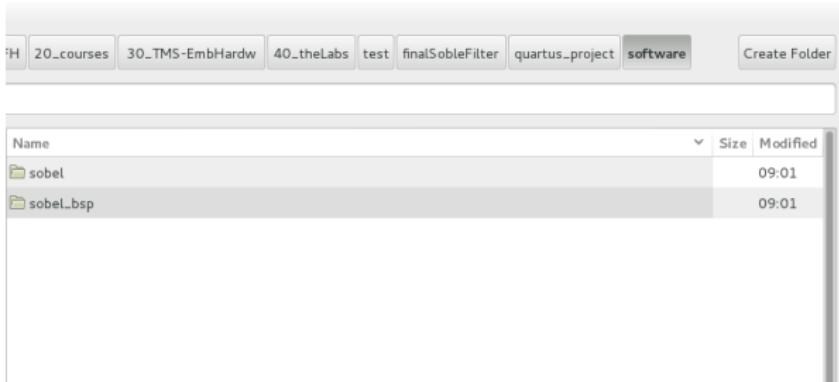


Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System



- ▶ Right click in project explorer
- ▶ Select Existing Projects
- ▶ Select the software folder located in quartus_project folder

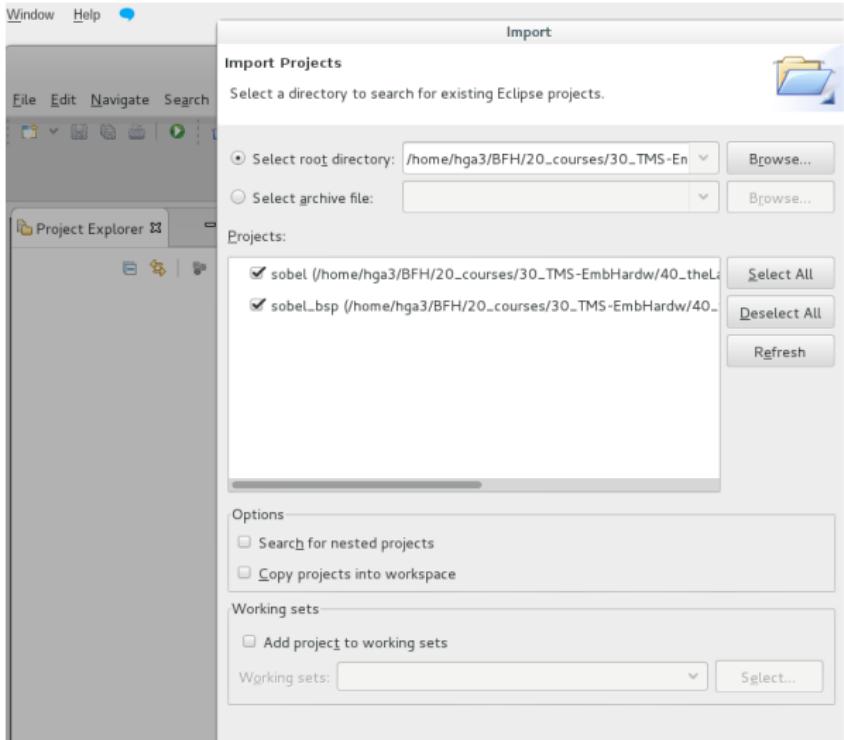
Import Firmware



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System

- ▶ Right click in project explorer
- ▶ Select Existing Projects
- ▶ Select the software folder located in quartus_project folder

Import Firmware



- ▶ Select both the sobel firmware and the corresponding BSP
- ▶ Rebuild the BSP first
- ▶ Rebuild BSP project and afterwards the sobel project



Import Firmware



Intro

Prerequisite

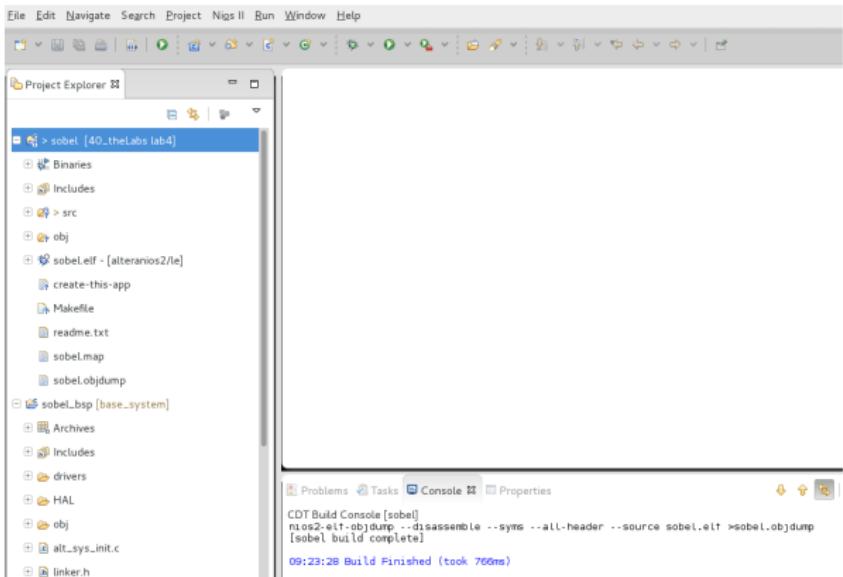
Quartus

QSys

Program

Nios II SDK

System

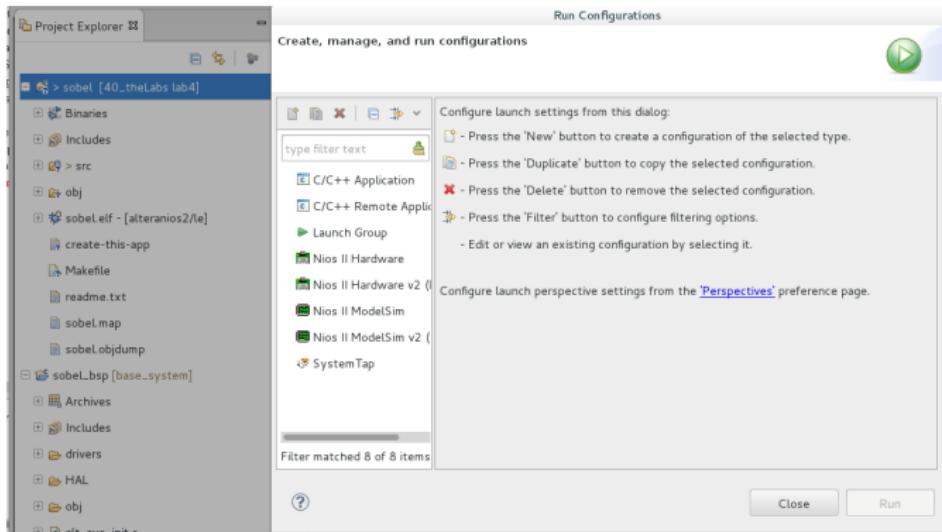


- ▶ Select both the sobel firmware and the corresponding BSP
- ▶ Rebuild the BSP first
- ▶ Rebuild BSP project and afterwards the sobel project

Download Firmware



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System

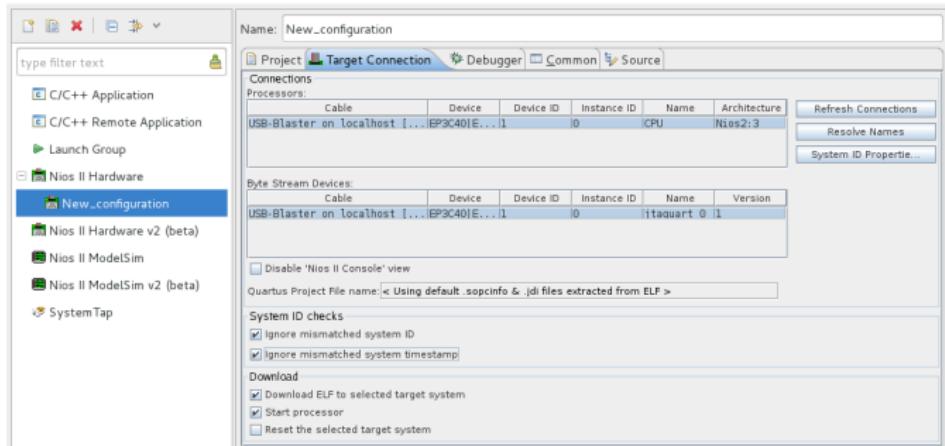


- ▶ Create new “run Configuration”
- ▶ Choose “Nios II Hardware”
- ▶ Tick both check-boxes for System ID checks (to deactivate the checks)
- ▶ Click on “Refresh connection”
- ▶ Click on “Apply” first and second on “Run”

Download Firmware



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System



- ▶ Create new “run Configuration”
- ▶ Choose “Nios II Hardware”
- ▶ Tick both check-boxes for System ID checks (to deactivate the checks)
- ▶ Click on “Refresh connection”
- ▶ Click on “Apply” first and second on “Run”

Download Firmware



Intro

Prerequisite

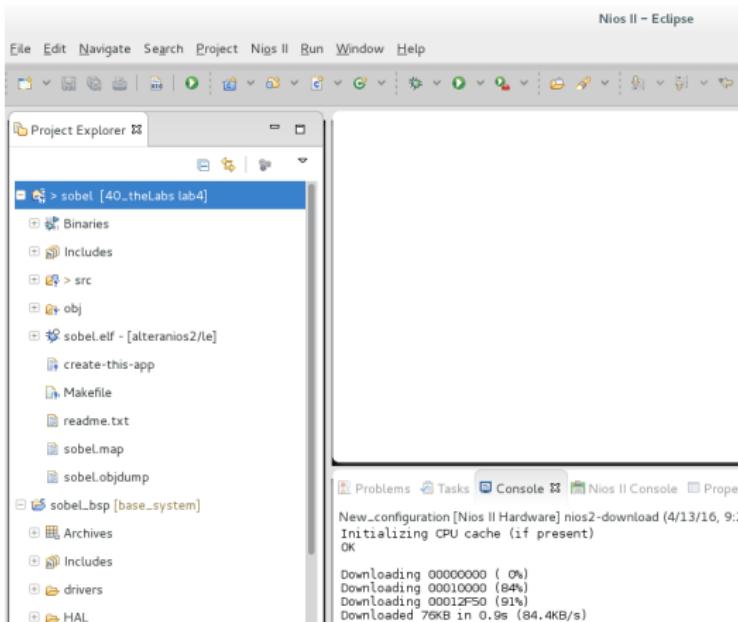
Quartus

QSys

Program

Nios II SDK

System



- ▶ Create new “run Configuration”
- ▶ Choose “Nios II Hardware”
- ▶ Tick both check-boxes for System ID checks (to deactivate the checks)
- ▶ Click on “Refresh connection”
- ▶ Click on “Apply” first and second on “Run”

Test on Device



Intro

Prerequisite

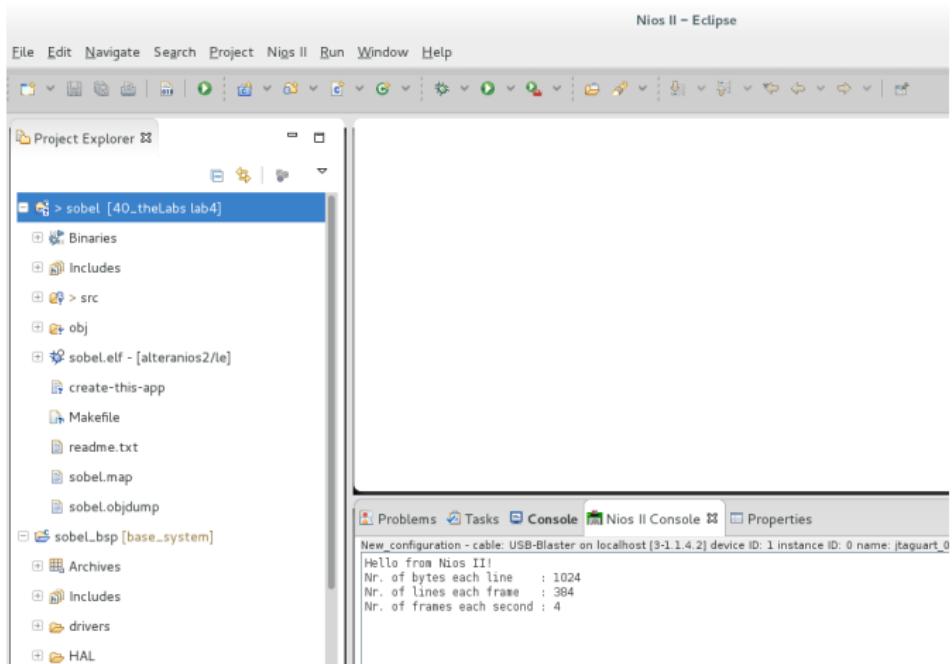
Quartus

QSys

Program

Nios II SDK

System



- The Firmware sends a hello message like the one shown in figure back
- Test the function as explained next

Test on Device



Design of Embedded
Hardware and
Firmware

Prof. A. Habegger



Bern University
of Applied Sciences

Intro

Prerequisite

Quartus

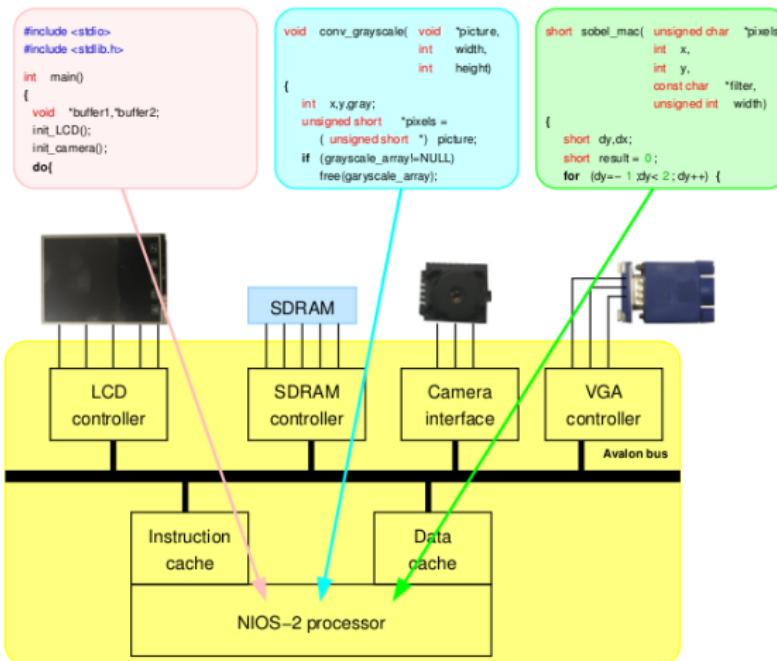
QSys

Program

Nios II SDK

System

Block Diagram

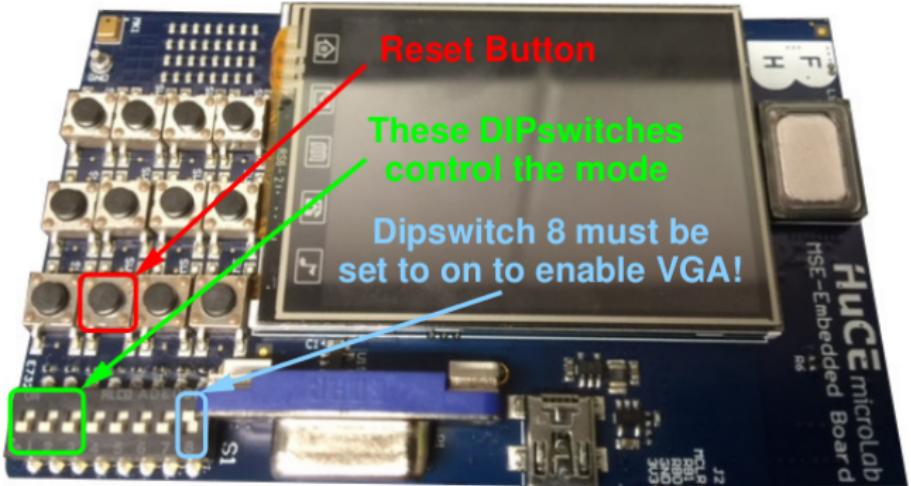


- ▶ Current system on the development board
- ▶ A VGA IP was added for demo on an external screen
- ▶ The SOBEL filtering is done on software
- ▶ Some parts of SOBEL are computation intense

Switch Modes



Intro
Prerequisite
Quartus
QSys
Program
Nios II SDK
System



SW1	SW2	SW3	Function
off	off	off	Direct camera → LCD
on	off	off	Grayscale
off	on	off	Sobel-X
on	on	off	Sobel-Y
X	X	on	Sobel complete