

# Automated Implementation of the Digital Configuration Interface for Application Specific Integrated Circuits

Zihong Lei

Final Master Thesis Presentation

Supervisor: Johannes Bastl, M.Sc

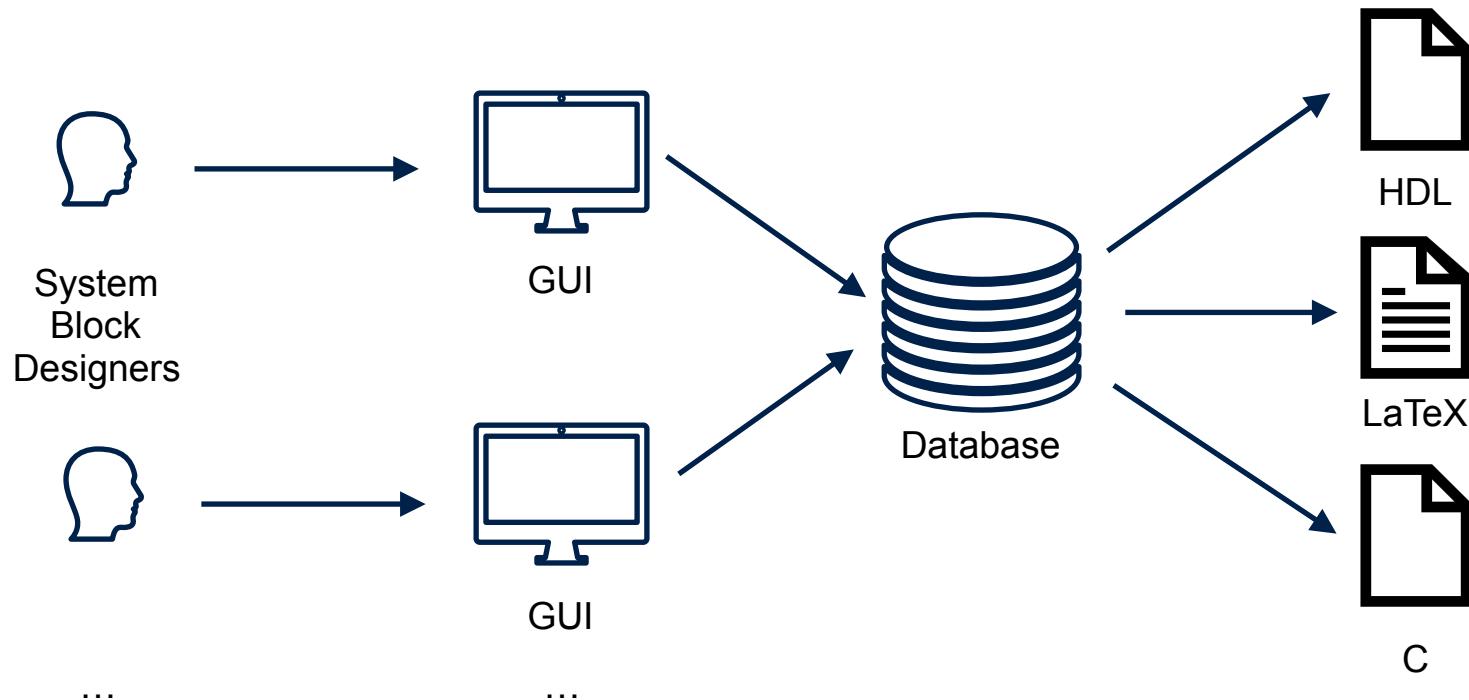
**Univ.-Prof. Dr.-Ing. Stefan Heinen**  
**Integrated Analog Circuits and RF Systems Laboratory**



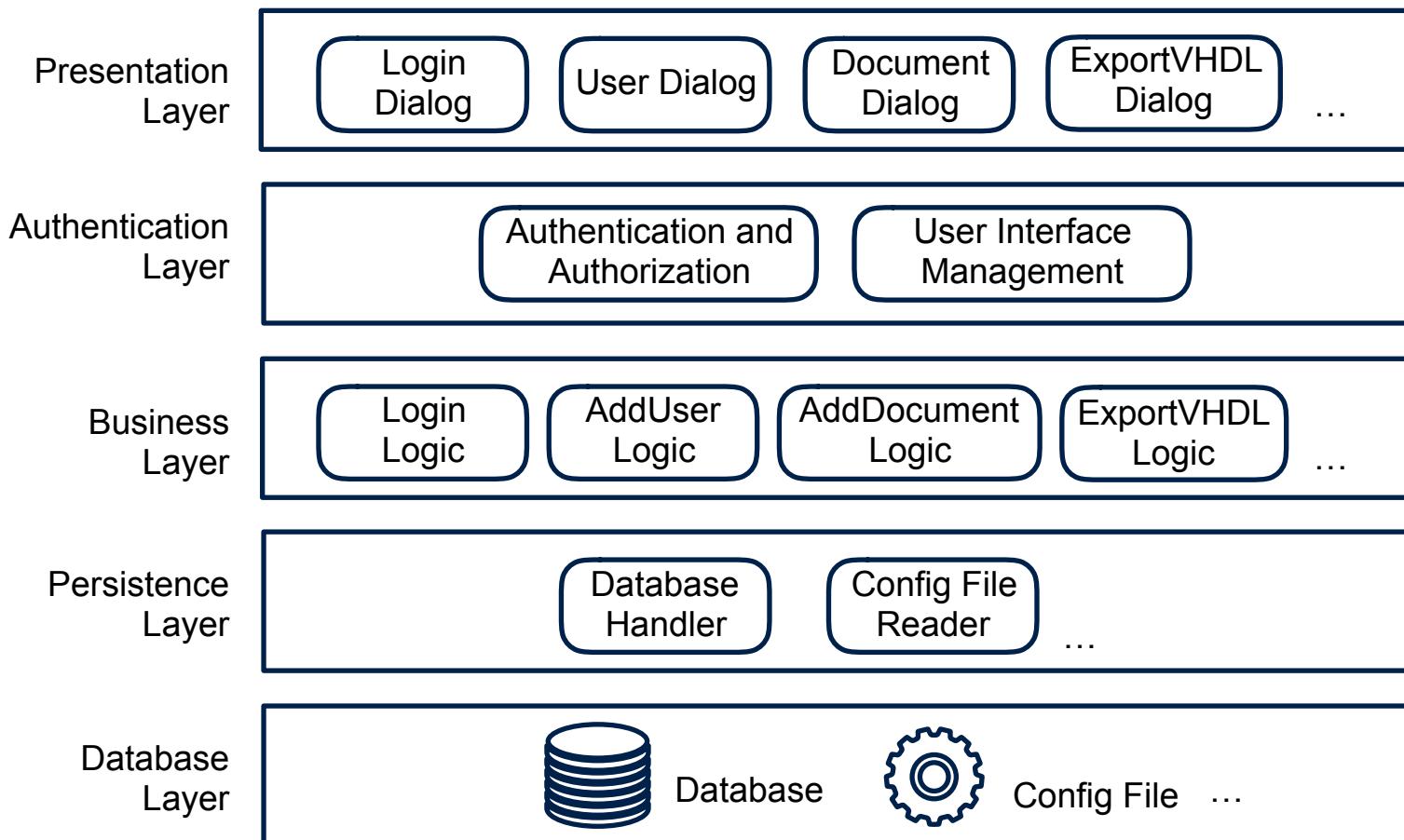
# Recap

# Proposal

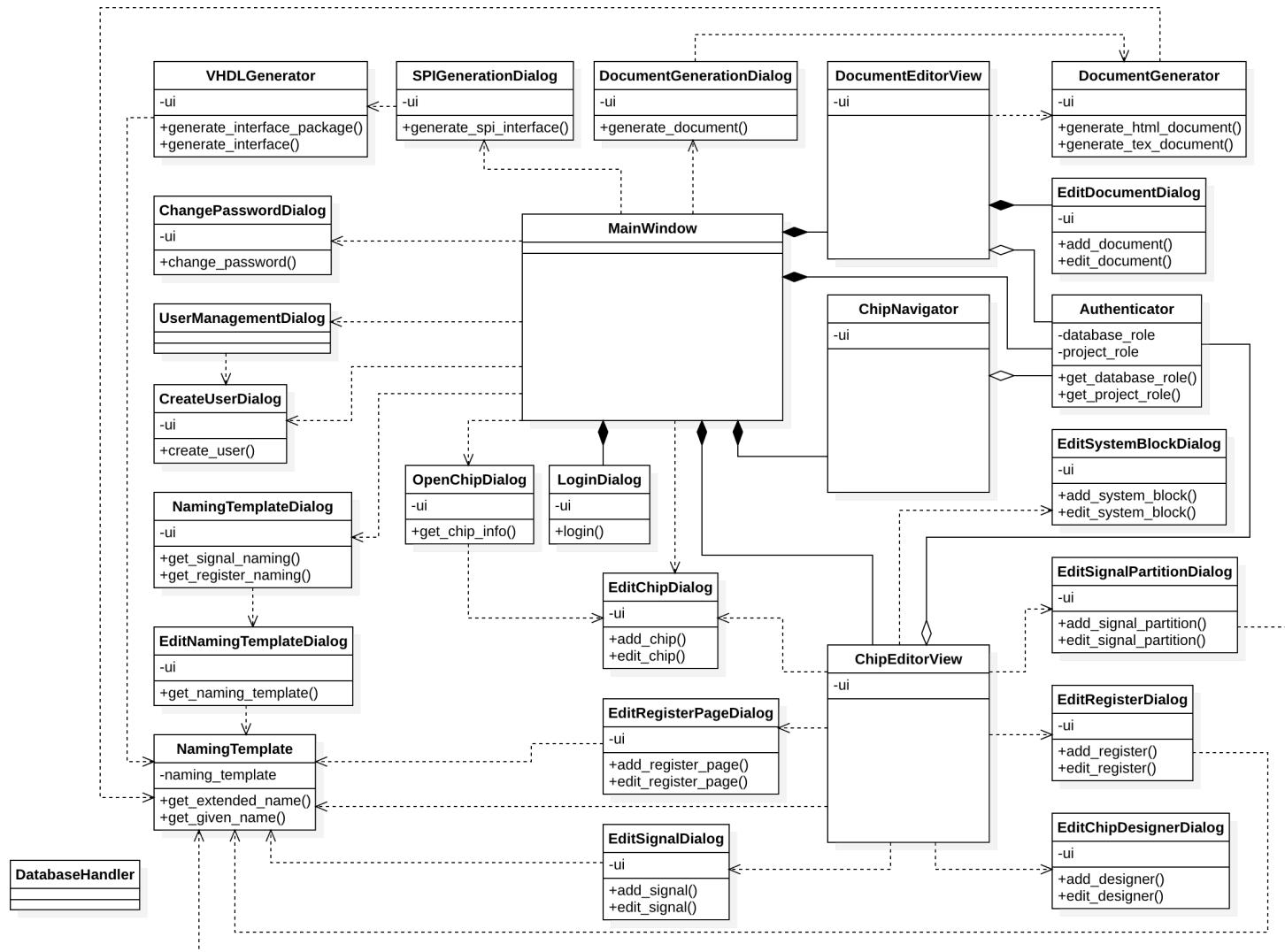
- A database centered software that aids in implementation of digital configuration interface for ASIC chips



# Layered Architecture Design



# Object-Oriented Software Design



# Module Design and Implementation

# Main Window

**Navigator**

- ▼ SampleChip
  - Global
  - Low-Dropout
  - State Machine
  - Baseband PLL

**Chip Editor**

**Basics**

Name	Owner	Register Width	Address Width	MSB First	
SampleChip	lei	8	12	true	

**Edit**

**System Block**

	Name	Abbreviation	Responsible	Start Address	Register Count	
1	Global	GLOBAL	lei	0x000	4	
2	Low-Dropout	LDO	lei	0x020	3	
3	State Machine	SM	lei	0x140	3	
4	Baseband PLL	BBPLL	lei	0x200	5	

**Add** **Remove**

**Designer**

	Name	Project Role
1	lei	admin

**Add** **Remove**

**Register Page**

Name	Control Signal	Page Count

**Add** **Remove**

**Document Editor**

Type	Content

**Add** **Remove**

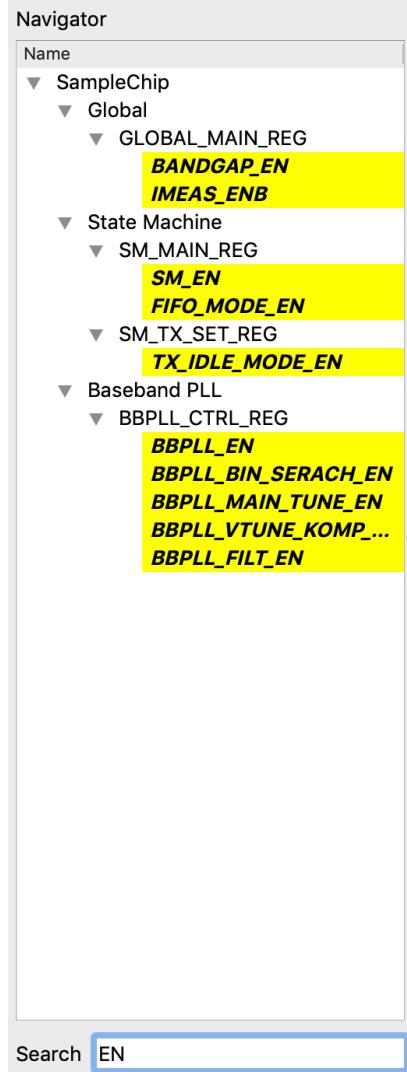
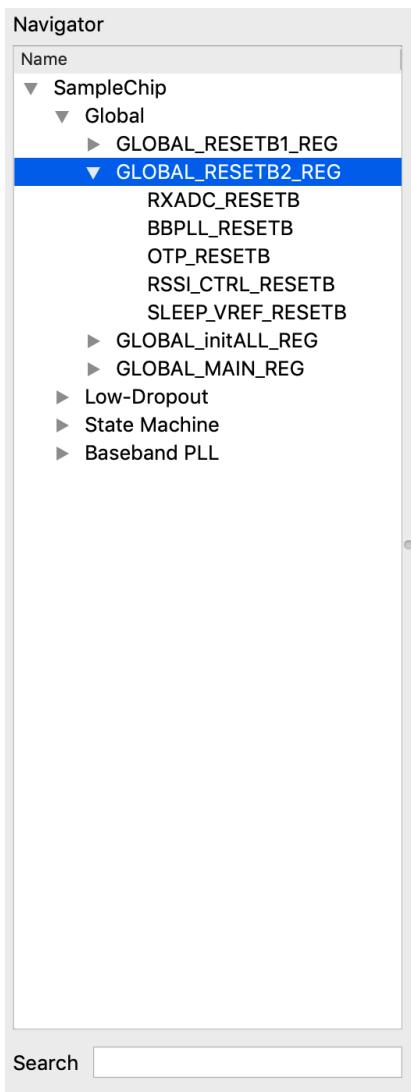
**Search**

The screenshot displays the main interface of a software tool. On the left, a vertical **Navigator** pane lists a project structure under **SampleChip**, including Global, Low-Dropout, State Machine, and Baseband PLL components. The central area features a **Chip Editor** with tabs for **Basics** (containing configuration for SampleChip) and **System Block** (listing four system blocks: Global, Low-Dropout, State Machine, and Baseband PLL with their respective details). Below these are **Designer** and **Register Page** sections, each with add and remove buttons. The right side contains a **Document Editor** panel with a table structure for managing document types and content, also with add and remove buttons. A search bar is located at the bottom left, and a status bar at the bottom right shows the date "Sep 2019".

# Chip Navigator

4-level tree widget  
chip, block, register, signal

Search area



# Chip Editor

## ■ Chip-level view

Chip Editor

Basics

Name	Owner	Register Width	Address Width	MSB First
SampleChip	lei	8	12	true

Edit

System Block

	Name	Abbreviation	Responsible	Start Address	Register Count
1	Global	GLOBAL	lei	0x000	4
2	Low-Dropout	LDO	lei	0x020	3
3	State Machine	SM	lei	0x140	3
4	Baseband PLL	BBPLL	lei	0x200	5

Add Remove

Designer

	Name	Project Role
1	lei	admin

Add Remove

Register Page

Name	Control Signal	Page Count

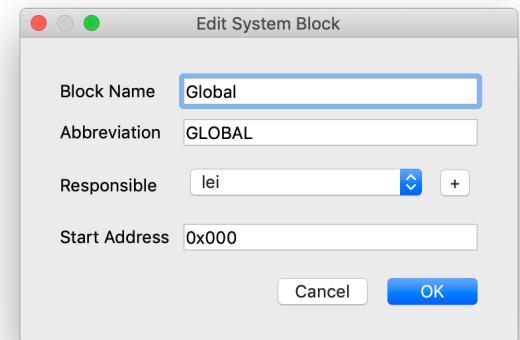
Add Remove

Designers

List of register pages

Basic Information about the chip

List of system blocks



EditSystemBlockDialog

# Chip Editor

## ■ Block-level view - Signal

Chip Editor

Signal Register

	Name	Width	Signal Type	Register Type	Value	Port
1	BANDGAP_EN	1	Control Signal	R/W	0x1	false
2	BANDGAP_PD	1	Control Signal	R/W	0x1	false
3	BBPLL_RESETB	1	Control Signal	R/W	0x0	false
4	BIAS_EXT	1	Control Signal	R/W	0x0	false
5	DEM_RESETB	1	Control Signal	R/W	0x0	false
6	GLOBAL_INIT	8	Control Signal	R/W	0x00	true
7	IMEAS_ENB	1	Control Signal	R/W	0x0	false
8	IMPROVE_SPI_COMMSS	1	Control Signal	R/W	0x0	false
9	OSCI_CTRL_RESETB	1	Control Signal	R/W	0x0	false
10	OTP_RESETB	1	Control Signal	R/W	0x0	false
11	PLL_RESETB	1	Control Signal	R/W	0x0	false
12	RSSI_CTRL_RESETB	1	Control Signal	R/W	0x0	false
13	RXADC_RESETB	1	Control Signal	R/W	0x0	false
14	SI_FFP_VRFF_RESETB	1	Control Signal	R/W	0x0	false

Add Remove

Signal Partition	Register Partition
<0:0>	GLOBAL_MAIN_REG<2:2>

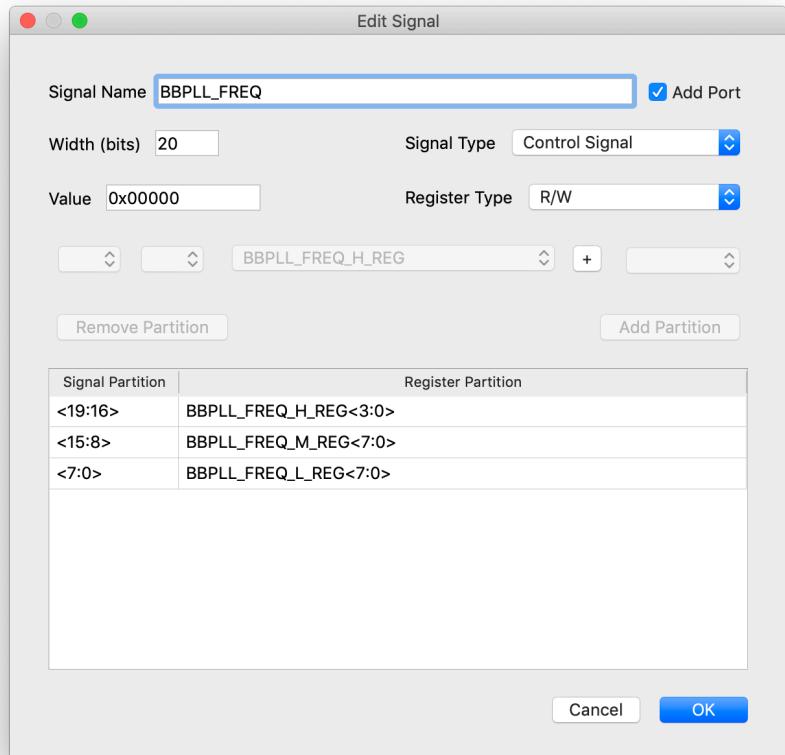
Add Remove

All signals in the current block

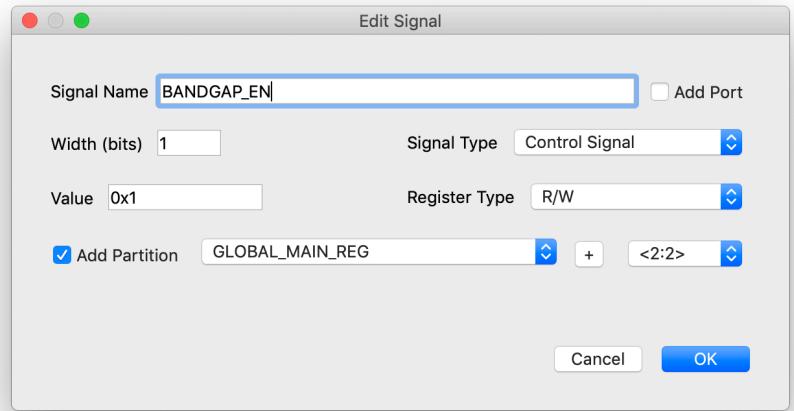
Signal-register mappings

# Chip Editor

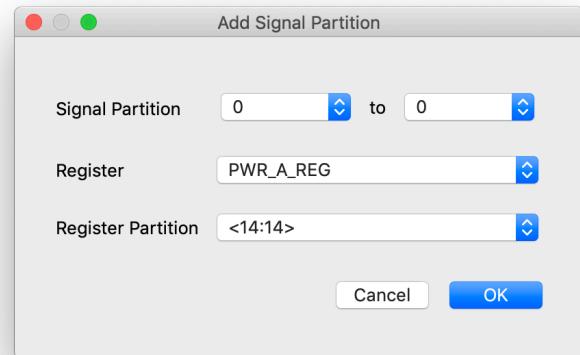
## ■ Block-level view - Signal



EditSignalDialog - Multi-bit



EditSignalDialog - Single-bit



EditSignalPartitionDialog

# Chip Editor

## ■ Block-level view - Register

Chip Editor

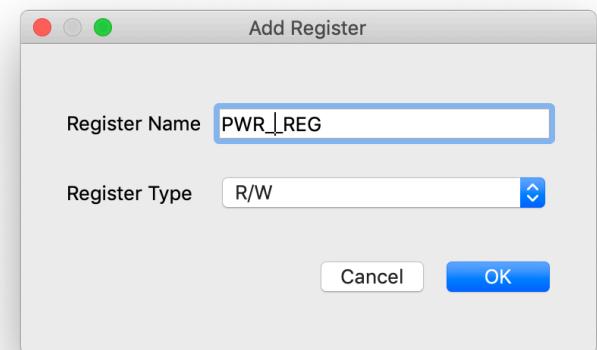
	Name	Register Type	Presumed Address
1	GLOBAL_RESETB1_REG	R/W	0x000
2	GLOBAL_RESETB2_REG	R/W	0x001
3	GLOBAL_INITALL_REG	R/W	0x002
4	GLOBAL_MAIN_REG	R/W	0x003

Add Remove

Register Partition	Signal Partition	Signal Init
<7:7>	NaN	NaN
<6:6>	WUR_RESETB	0x0
<5:5>	OSCI_CTRL_RESETB	0x0
<4:4>	ULP_RESET	0x0
<3:3>	SM_RESETB	0x0
<2:2>	TX_RESETB	0x0
<1:1>	DEM_RESETB	0x0
<0:0>	PLL_RESETB	0x0

Register-Signal mappings

All registers in the current block



EditRegisterDialog

# Document Editor

Document Editor

Type	Content
1 Text	Change the output voltage of the DVDD LDO.
2 Text	$\$V_{DVDD}=V_{ref, 500mV}\cdot\frac{174k\Omega}{65k\Omega}\cdot\frac{V_{LDO\_D\_VOUT<4:0>}}{2k\Omega}$ . The init value corresponds to 1.2V. 1 LSB corresponds to about 20mV around the init value. Digital tuning range is 1.0V...2.0V but the output is about 200mV lower than VDD for maximum output current (80 mA).

Type

Content

$\$V_{DVDD}=V_{ref, 500mV}\cdot\frac{174k\Omega}{65k\Omega}\cdot\frac{V_{LDO\_D\_VOUT<4:0>}}{2k\Omega}$ . The init value corresponds to 1.2V. 1 LSB corresponds to about 20mV around the init value. Digital tuning range is 1.0V...2.0V but the output is about 200mV lower than VDD for maximum output current (80 mA).

Preview

$V_{DVDD} = V_{ref, 500mV} \cdot \frac{174k\Omega - 2k\Omega \cdot LDO\_D\_VOUT<4:0>}{65k\Omega - 2k\Omega \cdot LDO\_D\_VOUT<4:0>}$ . The init value corresponds to 1.2V. 1 LSB corresponds to about 20mV around the init value. Digital tuning range is 1.0V...2.0V but the output is about 200mV lower than VDD for maximum output current (80 mA).

Show Preview

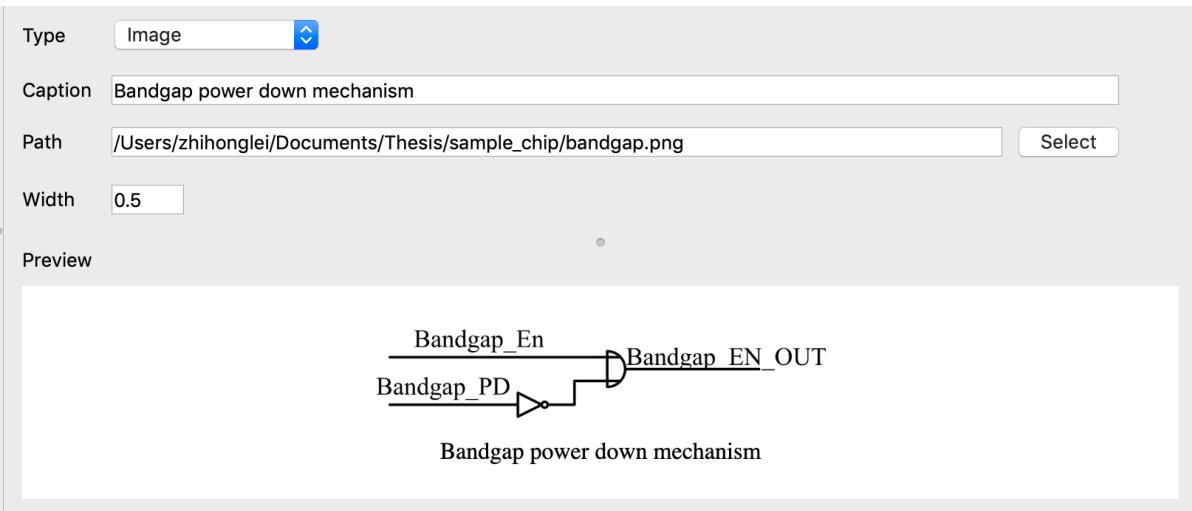
Add

List of documents of currently selected block/register/signal

Document editing area

Preview: HTML web viewer supporting LaTeX

# Document Editor



Image

Type

Row  Column

Caption

Table

SM_COMMAND<3:0>	Meaning
000	CMD_NONE
002	CMD_SLEEP
003	CMD_DEEPSLEEP
004	CMD_TX
005	CMD_RXIDLE
006	CMD_RX
007	CMD_RXHOLD

Preview

SM_COMMAND	
SM_COMMAND<3:0>	Meaning
000	CMD_NONE
002	CMD_SLEEP
003	CMD_DEEPSLEEP
004	CMD_TX
005	CMD_RXIDLE
006	CMD_RX
007	CMD_RXHOLD

Table

# Document Editor

## ■ Auto-completion

Type

Content

Change the output voltage of the DVDD LDO.  
ldo

- LDO
- LDO\_D\_VOUT\_REG
- LDO\_CLK\_VOUT\_REG
- LDO\_BOOST\_REG
- LDO\_D\_VOUT
- LDO\_CLK\_VOUT
- LDO\_D\_BOOST
- LDO\_CLK\_BOOST

Text content

Type

Row 8 Column 2

Caption sm\_

Table

SM_RESETB	Meaning
SM_MAIN_REG	CMD_NONE
SM_COMMAND_REG	CMD_SLEEP
SM_TX_SET_REG	CMD_DEEPSLEEP
SM_EN	CMD_TX
SM_RESETB	CMD_TWIDLE
SM_COMMAND	

Table & image caption

Type

Row 8 Column 2

Caption SM\_COMMAND

Table

sm_	Meaning
00 SM_RESETB	CMD_NONE
00 SM_MAIN_REG	CMD_SLEEP
00 SM_COMMAND_REG	CMD_DEEPSLEEP
00 SM_TX_SET_REG	CMD_TX
00 SM_EN	CMD_TWIDLE
00 SM_RESETB	
00 SM_COMMAND	

Table cell

# Document Editor

- Auto-completion - candidate words
  - Builtin
    - Chip name
    - Block names and abbreviations
    - Register names
    - Signal names
  - User defined
    - \${root}/completion/\*.txt



A screenshot of a Mac OS X-style text editor window titled "latex\_keywords.txt". The window contains a list of LaTeX commands, each preceded by a red dot indicating they are selected or highlighted. The list includes: \cdot, \sum, \begin, \end, \sqrt, \alpha, \Alpha, \omega, and \Omega.

```
\cdot
\sum
\begin
\end
\sqrt
\alpha
\Alpha
\omega
\Omega
```

Example word list

# Accounts and Passwords

- Database account: shared to all software users

```
# ${root}/global_settings.ini
[database]
database=ias
encrypted_password="TvCYyAGe6hcIyj4ftIg31w=="
host=localhost
key=vFCI399bBoVpu7o89rGbQ/TYyJj0G8ft
port=3306
username=root
```

- Software account



LoginDialog

# Accounts and Passwords

## ■ Chip designer account

Chip Editor

Basics

Name	Owner	Register Width	Address Width	MSB First
SampleChip	lei	8	12	true

System Block

	Name	Abbreviation	Responsible	Start Address	Register Count
1	Global	GLOBAL	lei	0x000	4
2	Low-Dropout	LDO	lei	0x020	3
3	State Machine	SM	lei	0x140	3
4	Baseband PLL	BBPLL	lei	0x200	5

Designer

	Name	Project Role
1	lei	admin

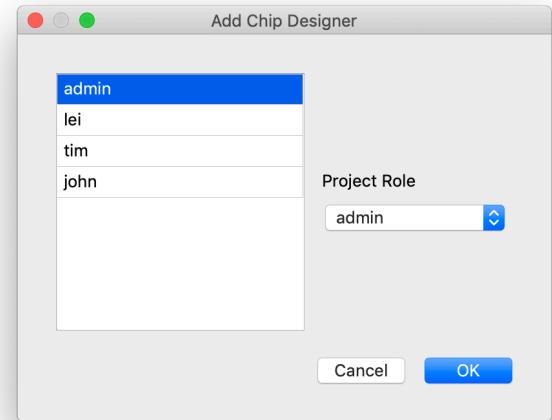
Register Page

	Name	Control Signal	Page Count

Add Remove

Add Remove

Add Remove



EditChipDesignerDialog

List of Chip Designers

# Permissions and Authentication

- Software/database permissions: software users
  - Add or remove a user account
  - Add or remove a chip project
- Project permissions: chip designers
  - Add system blocks
  - Remove system blocks the user is responsible for
  - Read all system blocks
  - Add or remove chip designers
  - Edit system blocks the user is responsible for (implicitly true for all chip designers)
- Authenticator class

# Signal and Register Naming

- Signals and registers may be named after a certain pattern

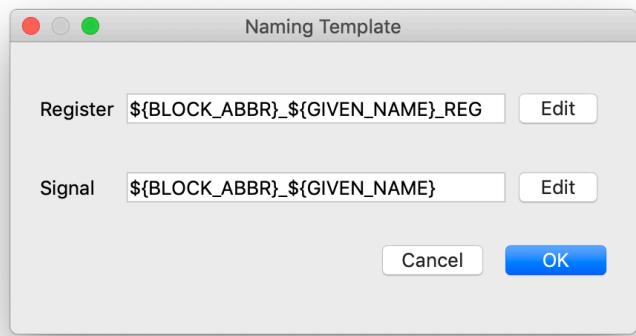


Block Abbr.   Given Name   Constant

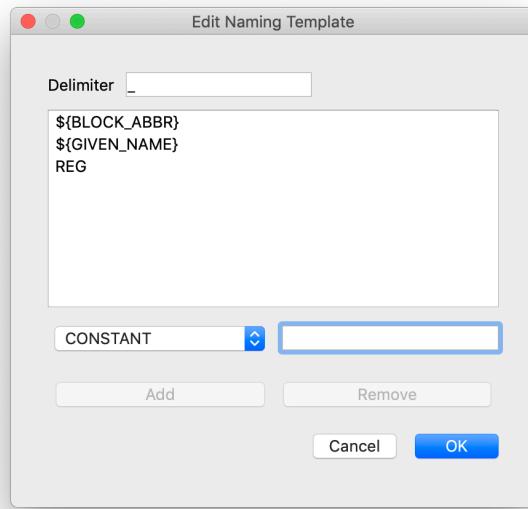
- Question: what if block abbreviations change?
  - Solution
    - Only store the given names in the database
    - Define naming templates for signals and registers
- Example: \${BLOCK\_ABBR}\_\${GIVEN\_NAME}\_REG
- Recover the extended name

# Signal and Register Naming

- NamingTemplate class
  - naming\_template: string
  - get\_extended\_name(given\_name: string) -> string
  - get\_given\_name(extended\_name: string) -> string
- Naming templates are easily adjustable

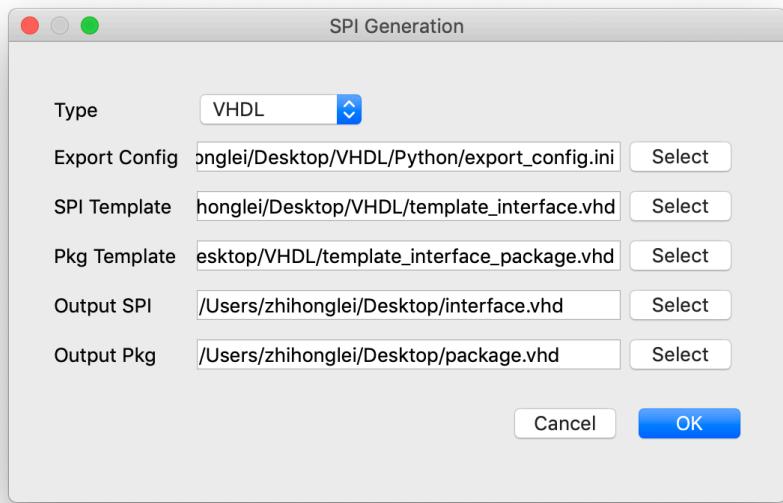


NamingTemplateDialog

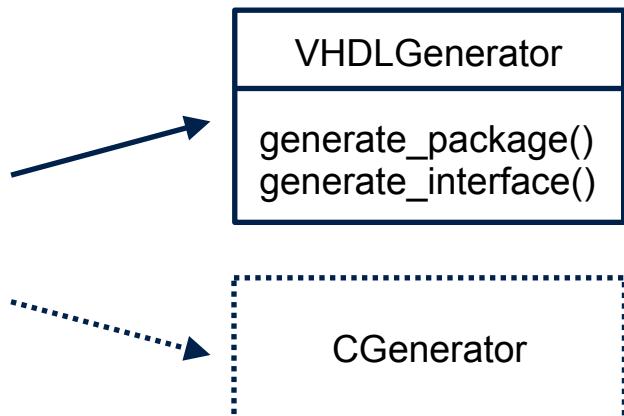


EditNamingTemplateDialog

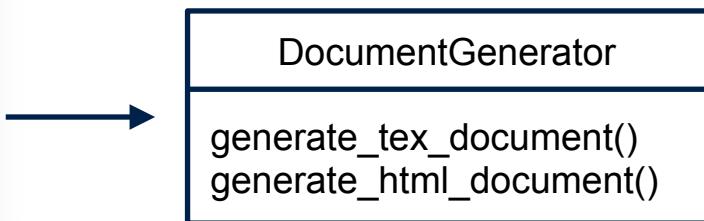
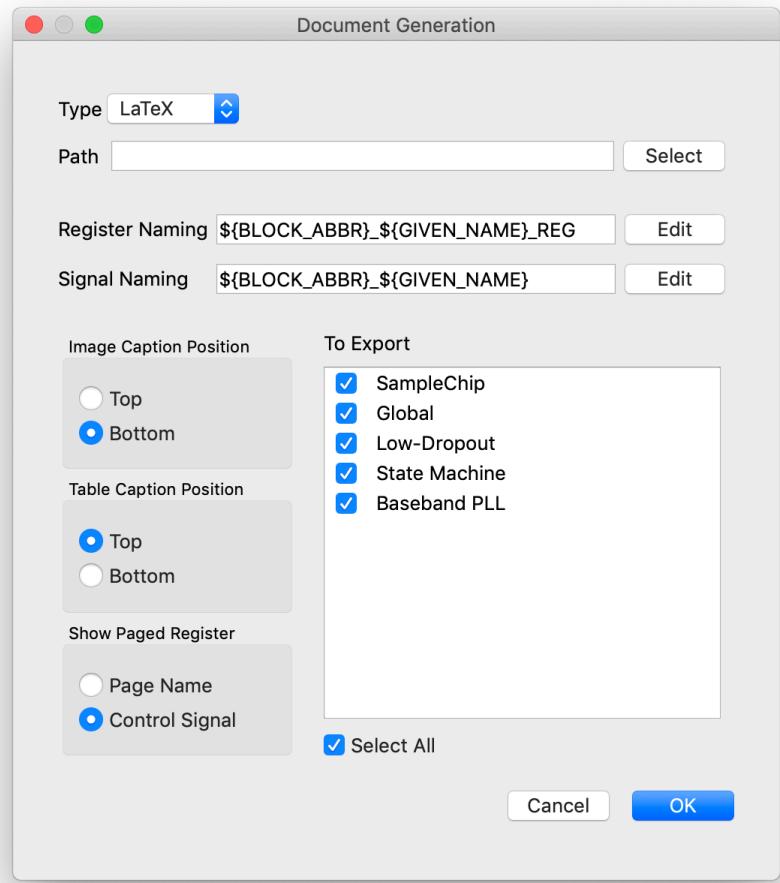
# SPI Interface Generation and Export



SPIGenerationDialog



# Document Generation and Export



DocumentGenerationDialog

# Documentation Preview

**Navigator**

- ▼ SampleChip
  - Global
  - Low-Dropout
  - State Machine
  - Baseband PLL

**Document Preview**

## SampleChip

1. [Global](#)
  - [GLOBAL\\_RESETB1\\_REG](#)
  - [GLOBAL\\_RESETB2\\_REG](#)
  - [GLOBAL\\_INITALL\\_REG](#)
  - [GLOBAL\\_MAIN\\_REG](#)
2. [Low-Dropout](#)
  - [LDO\\_D\\_VOUT\\_REG](#)
  - [LDO\\_CLK\\_VOUT\\_REG](#)
  - [LDO\\_BOOST\\_REG](#)
3. [State Machine](#)
  - [SM\\_MAIN\\_REG](#)
  - [SM\\_COMMAND\\_REG](#)
  - [SM\\_TX\\_SET\\_REG](#)
4. [Baseband PLL](#)
  - [BBPLL\\_CTRL\\_REG](#)
  - [BBPLL\\_FREQ\\_H\\_REG](#)
  - [BBPLL\\_FREQ\\_M\\_REG](#)
  - [BBPLL\\_FREQ\\_L\\_REG](#)
  - [BBPLL\\_POSTDIV\\_OSCICTRL\\_REG](#)

### Global

• [GLOBAL\\_RESETB1\\_REG - 0x000](#)

7	6	5	4	3	2	1	0
...	WUR_RESETB	OSCI_CTRL_RESETB	ULP_RESET	SM_RESETB	TX_RESETB	DEM_RESETB	PLL_RESETB
.	0	0	0	0	0	0	0

- WUR\_RESETB is written to WAKEUP<3>
- OSCI\_CTRL\_RESETB is written to OSCI\_CTRL<0>
- ULP\_RESET is written to ULP\_MAIN<1> and inverted to ULP\_MAIN<4>
- SM\_RESETB is written to SM\_MAIN<1>
- TX\_RESETB is written to TX\_MAIN<5>
- DEM\_RESETB is written to DEM\_MAIN<7>
- PLL\_RESETB is written to PLL\_MAIN<3>

GLOBAL\_RESETB2\_REG - 0x001

7	6	5	4	3	2	1	0
...			RXADC_RESETB	BBPLL_RESETB	OTP_RESETB	RSSI_CTRL_RESETB	SLEEP_VREF_RESETB
.			0	0	0	0	0

- RXADC\_RESETB is written to RXADC\_SET\_REG<7>
- BBPLL\_RESETB is written to BBPLL\_CTRL\_REG<7>
- OTP\_RESETB is written to OTP\_CONFIG\_REG<0>
- RSSI\_CTRL\_RESETB is written to RSSI\_CTRL\_REG<3>
- SLEEP\_VREF\_RESETB is written to SLEEP\_VREF\_REG<2>

Search

# Integration, Testing and Deployment

# Integration and Testing

- Module testing: right after implementation
- Integrated testing
  - Beta version delivered to users
  - The software was used on real-world projects
  - Errors and bugs were found and removed
  - Some functionalities were improved

# Deployment

- The software depends on external softwares in form of shared libraries
  - Qt modules
  - MySQL C++ Connector
- Problem: users have to manually install all dependencies and link them to the executable
- Deployment
  - Package the executable and all dependencies
  - Relink the executable to these dependencies
  - Deliver the software package to users
- We deployed the software using `linuxdeployqt` tool

# Thank you