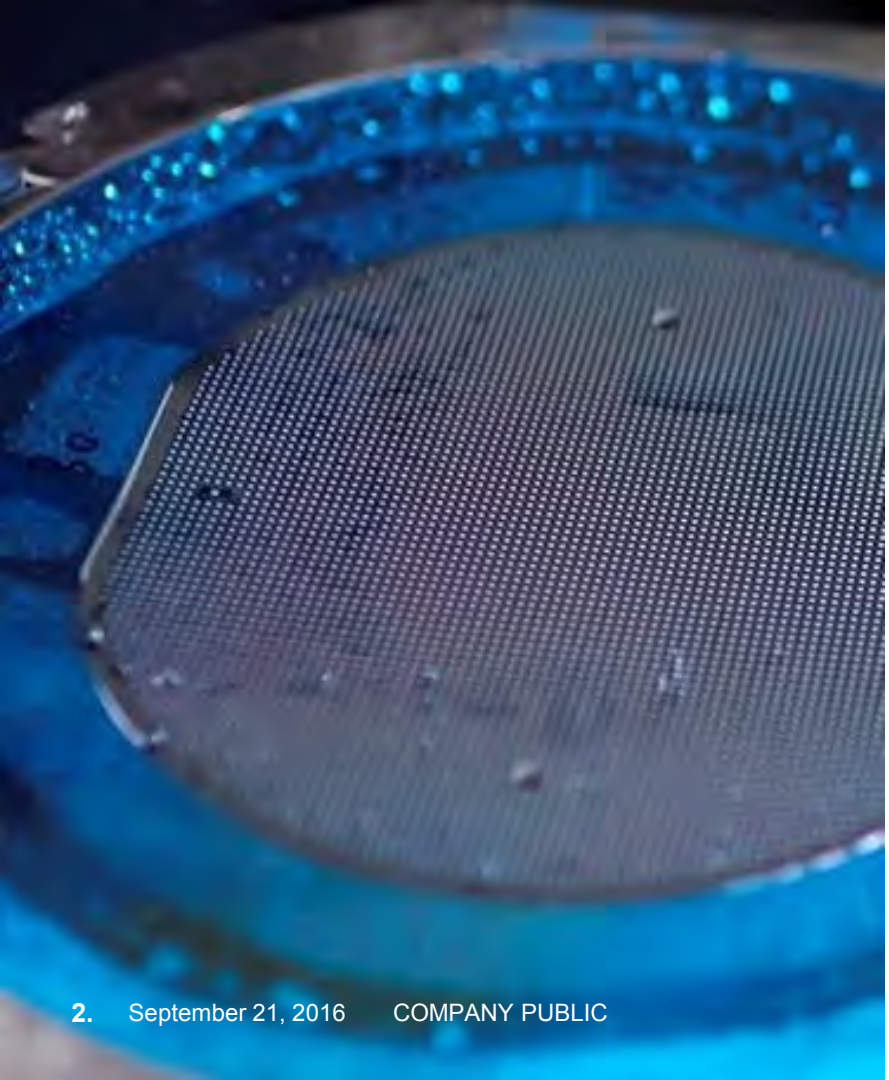
NHS3xxx Software overview

SECURE CONNECTIONS
FOR A SMARTER WORLD



Contents

- IC family
- Demo/Evaluation HW
- Development environment
- Architecture
- Documentation
- Release
- Quality

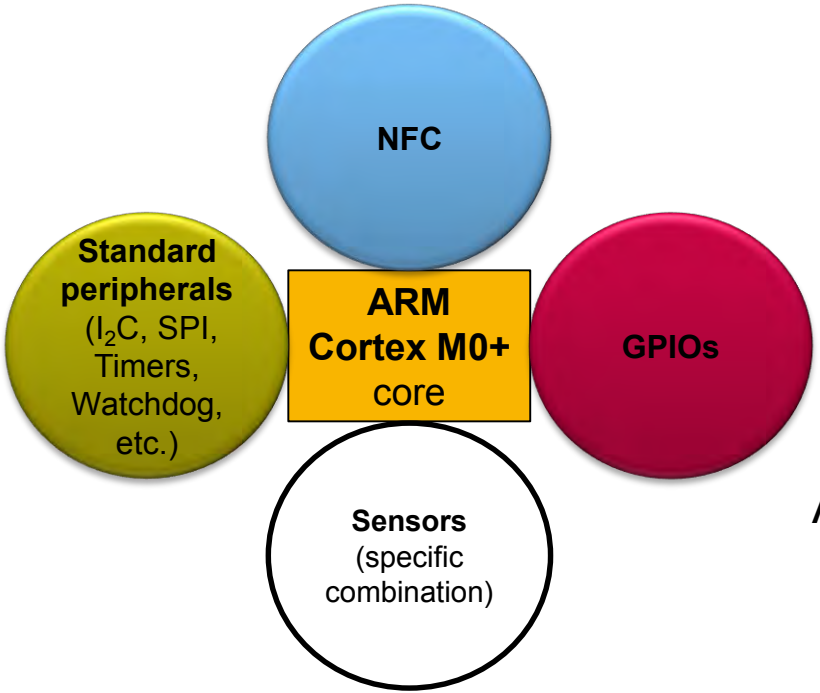
IC family

Smart Sensor

- Low cost
- Ultra-low power
- Programmable
- NFC enabled

Compute core

- 62.5 kHz – 8 MHz
- 32 k Flash
- 8 k Ram
- 4 k EEPROM



NHS3xxx

Available sensors

- ADC/DAC
- Capacitance
- Current
- Temperature

Demo/Evaluation HW

Typical setup

NHS3xxx board

SWD

USB Debugger (LPC-Link2)

USB

Development workstation

NFC host device

Development environment

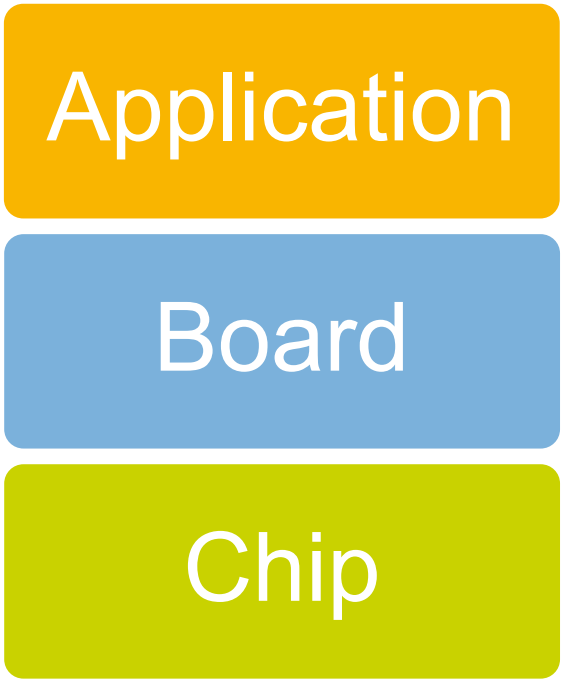
- Adapted for NHS3xxx (plugin)
- Eclipse based
- GNU C compiler, linker, libraries
- GDB debugger
- Integration with LPC-Link2
- Freely available

Demo/Evaluation HW

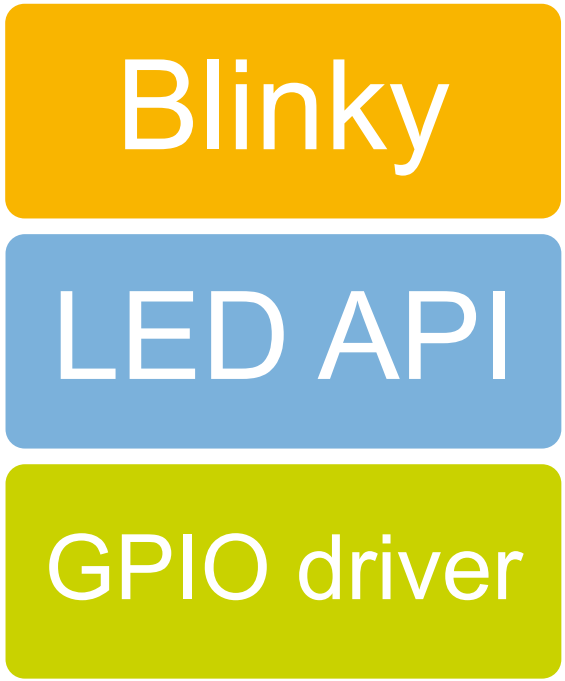
NHS3xxx board

- Features
  - Battery holder
  - LED
  - Button
  - Exposed pins
- Can be used
  - Standalone (Demo)
  - With debugger (Development)
- Use case specific boards:
  - Temperature Logger PCB
  - Therapy Adherence PCB

Architecture



An example

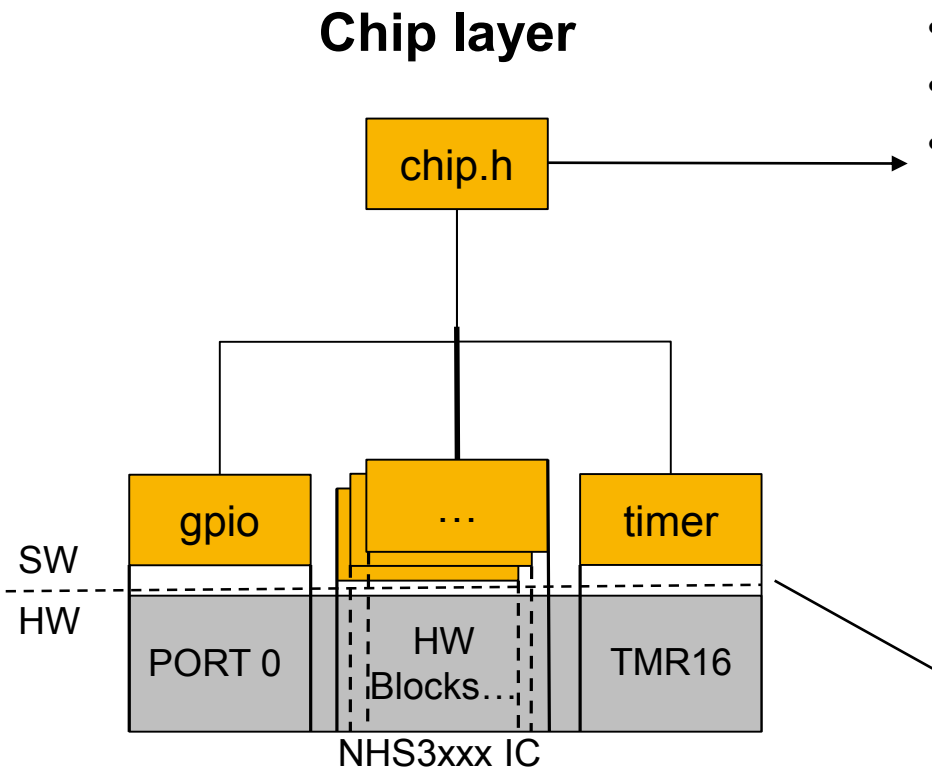


Program flow
(on/off period)

LED polarity
LED pin

Pin direction
Pin state

Architecture



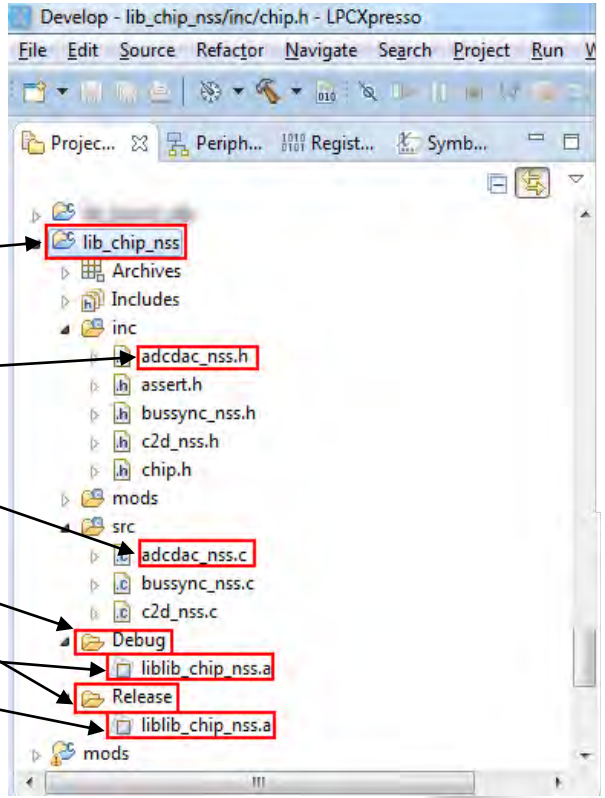
- Single entry point to the drivers
- Describes the specific IC model
- Publishes chip level info
  - PUBLIC oscillators
  - PUBLIC memories
  - Factory data addresses

1 driver per HW block
(direct mapping with HW)

Architecture

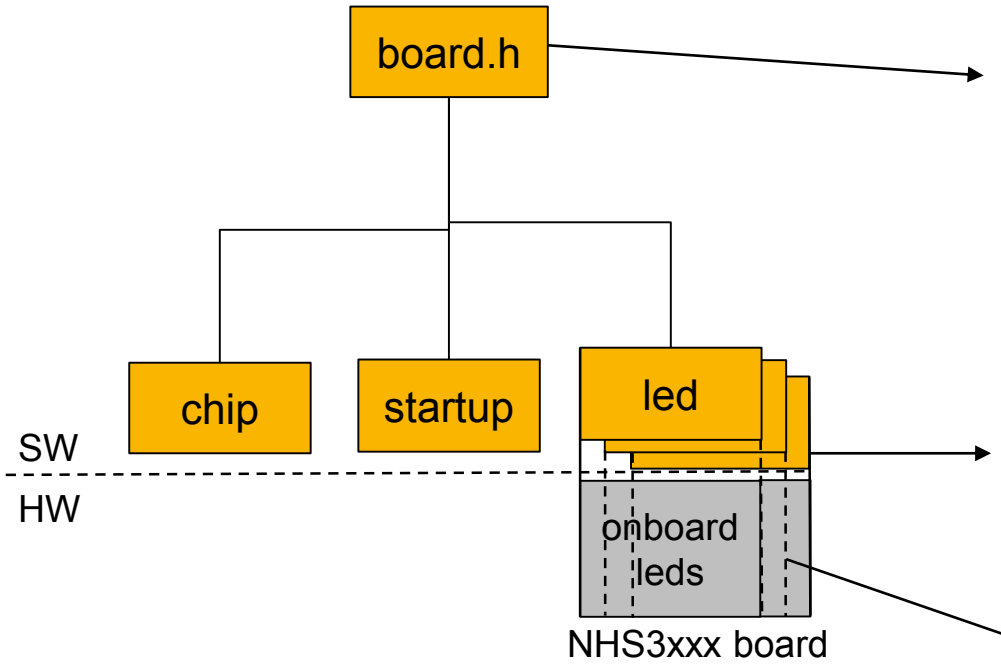
Chip layer – in LPCXpresso

- 1 LPCXpresso project
- 1 .c and 1 .h file per driver
- 2 flavors (Debug and Release)
- Builds into a library (.a)



Architecture

Board layer



- Single entry point to the HW
- Describes the specific board
- Abstracts application from HW

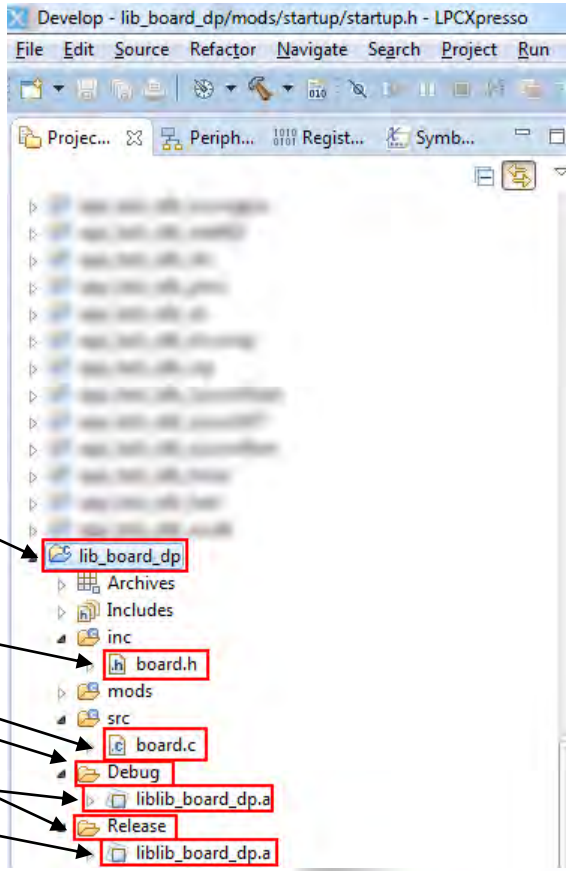
- A board always contains a chip
- Startup SW module
- API per board feature(e.g. LED)

other board specific functionalities

Architecture

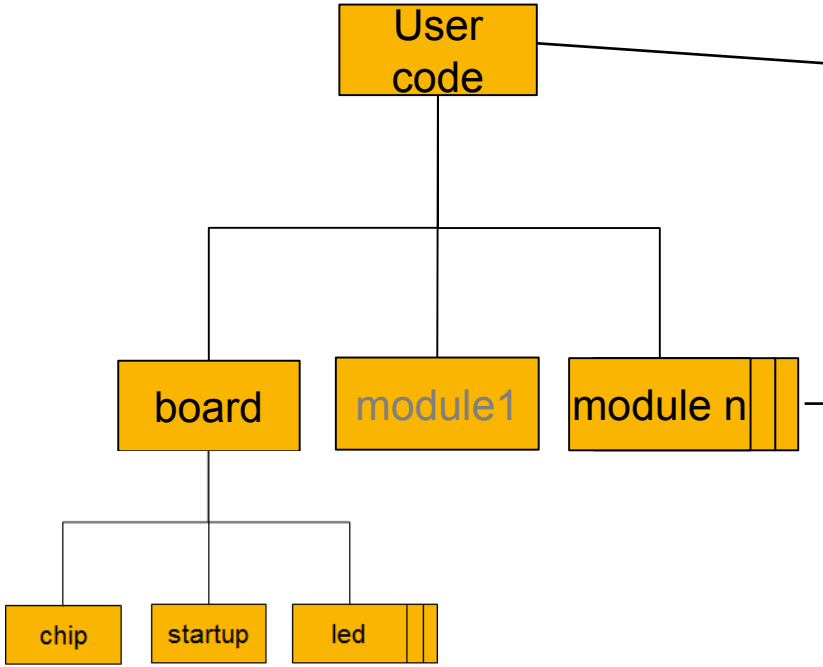
Board layer – in LPCXpresso

- 1 LPCXpresso project per board
- 1 .c and 1 .h file
- 2 flavors (Debug and Release)
- Builds into a library (.a)



Architecture

Application layer



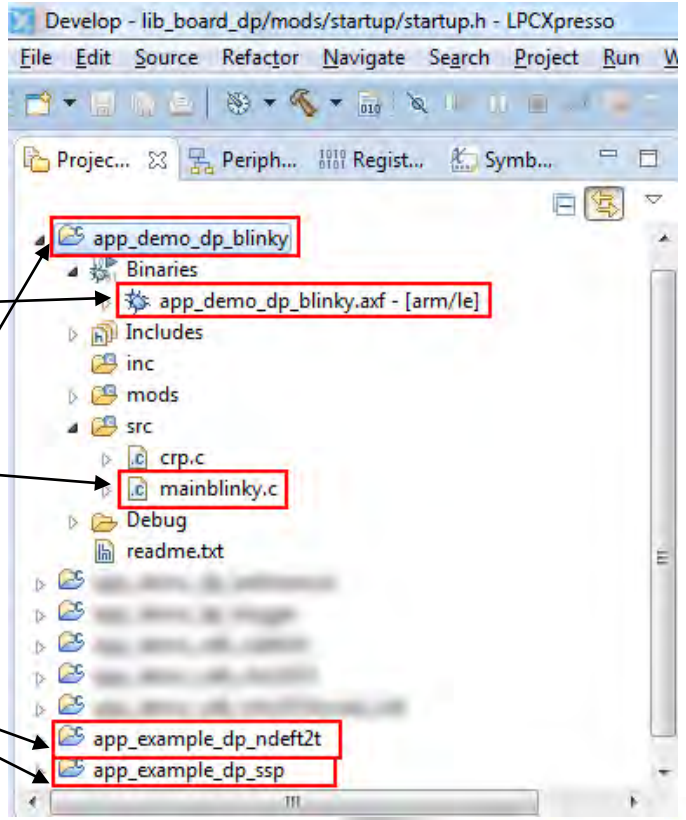
- Implements the application flow
- Accesses HW in an abstracted way

- An application runs on a board
- Modules can be re-used across applications (“mods” folder)

Architecture

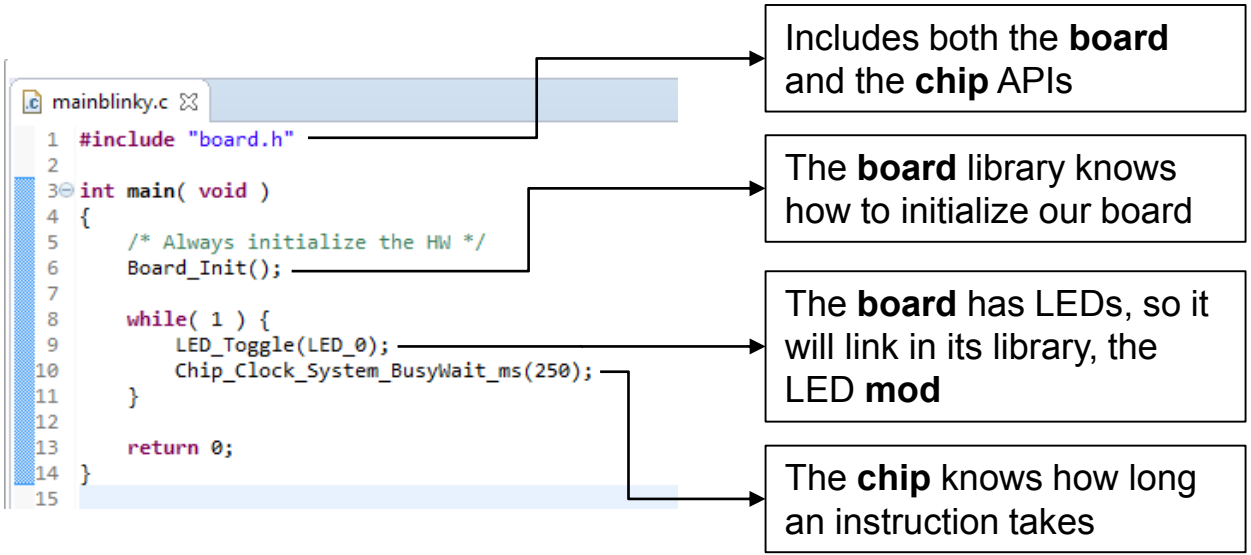
Application layer – in LPCXpresso

- Builds into an executable (.axf)
- At least 1 file (main.c)
- Links with chip and board libraries
- 1 LPCXpresso project per application



Architecture

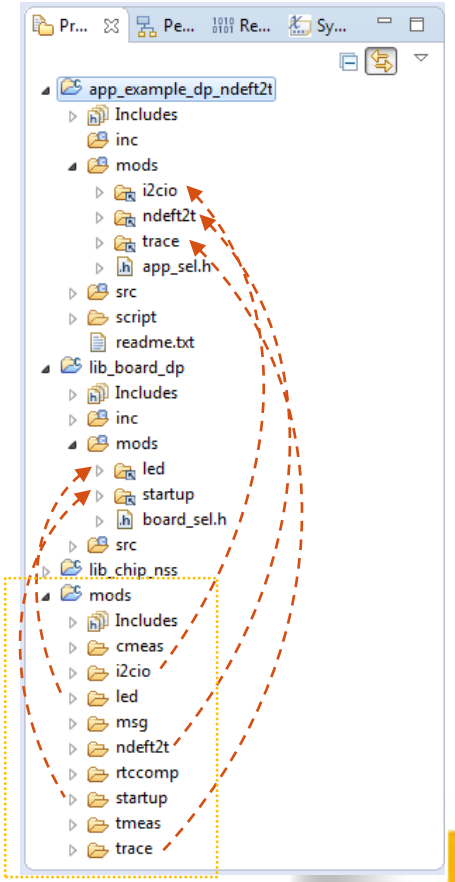
How it works in practice



Architecture

Code reusability

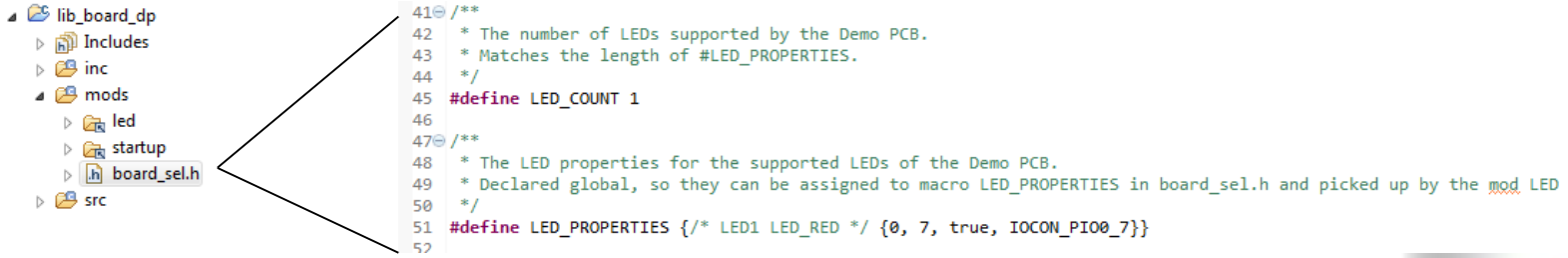
- The “mods” project is just a container of reusable modules (does not build)
- One folder in the “**mods**” project contains one module
- Modules can be reused in every **chip**, **board** or **application** project (a reference to the module is created in the “mods” folder of the respective project)
- The code of the module is compiled by the project they are referenced in



Architecture

Diversity

- Reusable modules support diversity
- Diversity settings for module “xxx” are described in “xxx_dft.h”
- The project that reuses the module is responsible for defining the required settings (in [chip|board|app]_sel.h)
- E.g.: for module ‘led’, the number of LEDs, the physical pins and the polarity differ per board



Documentation

NHS3xxx firmware

- Every API is documented
- Embedded in source code
- Doxygen style
- Outputs in html or pdf

```
146
147 /**
148  * @brief    Sets the System Clock frequency in Hz
149  * @param    frequency        : The System Clock frequency in Hz to set
150  * @note     This setting affects the core execution speed.
151  *           Only a set of frequencies is supported. If not valid,
152  *           the 'frequency' will be clipped to the closest supported value
153  *           higher than or equal to it.
154  *           The System Clock frequency range is (62.5 kHz - 8MHz).
155  *           Frequencies of 0 and higher than 8MHz are NOT allowed.
156  *           Use the #Chip_Clock_System_GetClockFreq to read to exact
157  *           frequency that was set.
158  */
159 void Chip_Clock_System_SetClockFreq(int frequency);
160
```

void Chip_Clock_System_SetClockFreq (int frequency)

Sets the System Clock frequency in Hz.

Parameters

frequency : The System Clock frequency in Hz to set

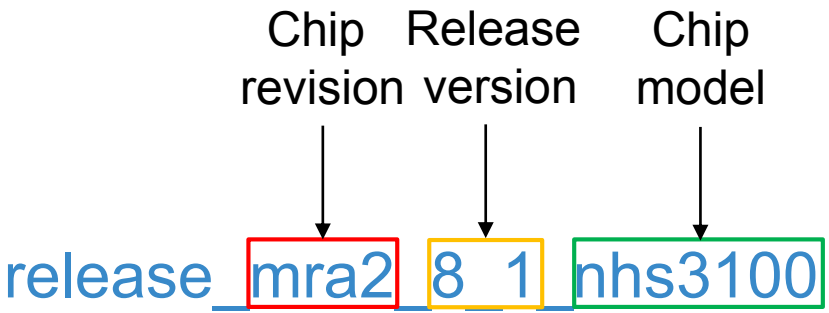
Note

This setting affects the core execution speed. Only a set of frequencies is supported. If not valid, the 'frequency' will be clipped to the closest supported value higher than or equal to it. The System Clock frequency range is (62.5 kHz - 8MHz). Frequencies of 0 and higher than 8MHz are NOT allowed. Use the **Chip_Clock_System_GetClockFreq** to read to exact frequency that was set.

Definition at line **91** of file **clock_nss.c**.

Release

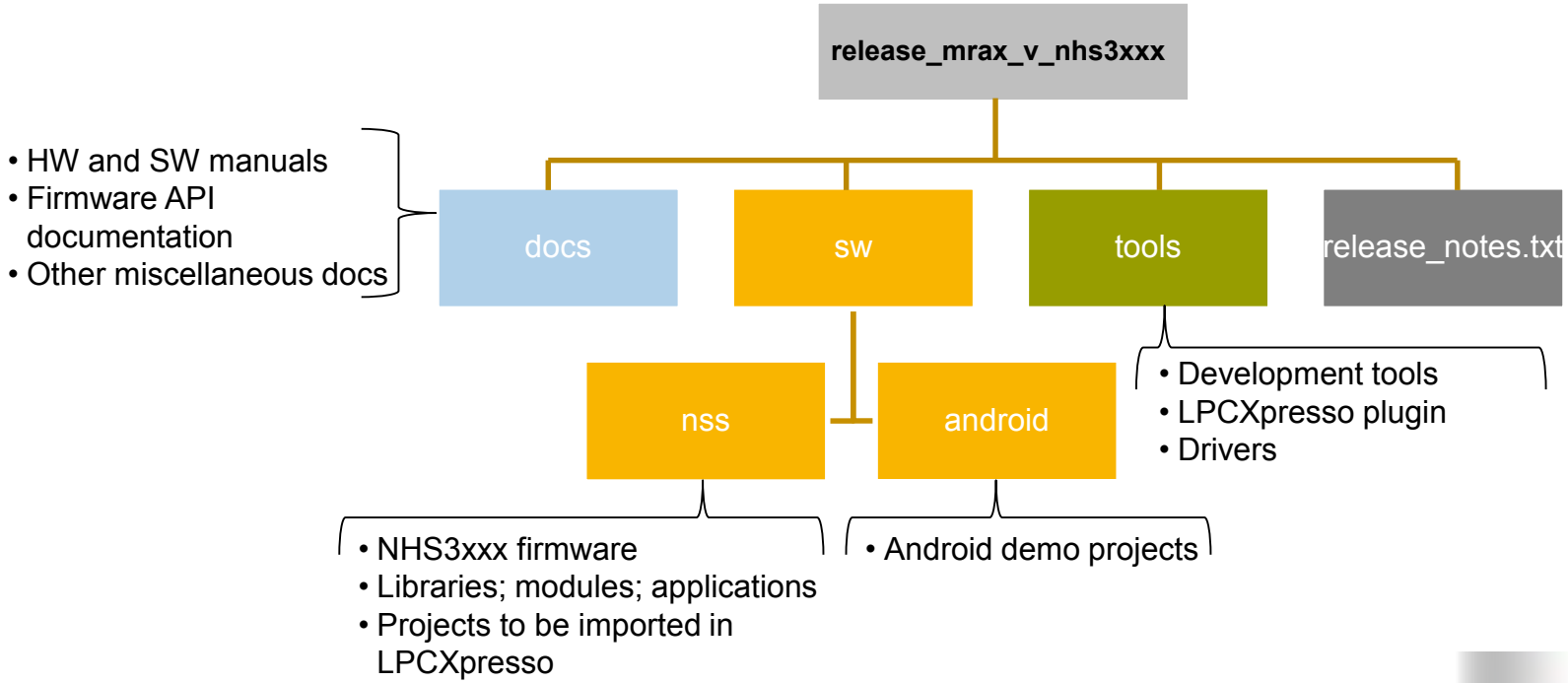
Naming



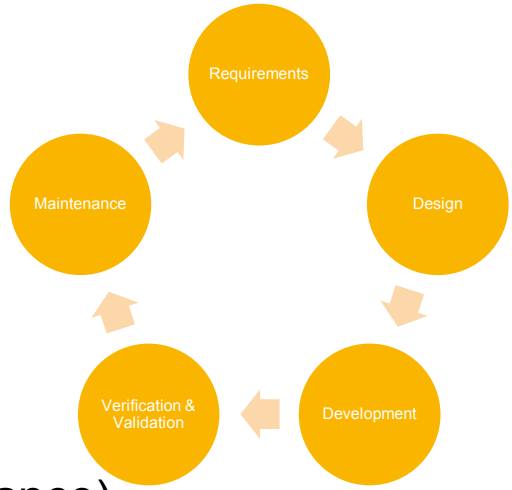
- A dedicated release per chip model (NHS3100, NHS3152, etc)
- Valid only for a single revision of the chip
- File tree structure is kept between versions to allow easy upgrade

Release

Structure



Quality



- Development process
  - From requirement to verification/validation
  - Coding style and implementation guidelines
  - Code reviews
  - Change request and problem report tracking (maintenance)

- Continuous Integration
  - Regression test suite at module level and integration level
  - Compiler (restrictive) warning free policy
  - Static code analysis and other quality metrics (work in progress)

NXP

SECURE CONNECTIONS
FOR A SMARTER WORLD