STM32 embedded software

ST's Offer

life.augmented

STM32 embedded software offer - overview

Abstraction level

- **Virtual machines with partners** .Net, Java

  IS2T, Oracle, Mountaineer, ….

- mbed Enabled **mbed core**

- **"C" partners** Micrium, SEGGER, HCC, ….

- **STM32Cube HAL**

- **Standard peripheral libraries**

- CODE **STM32Snippets**

- **STM32Cube LL**

| STM32 device-specific *f.i.: STM32F072* | STM32 device-specific *f.i.: STM32F072* | STM32 series-specific *f.i.: STM32F1* | STM32 family-specific | Cortex-M -based MCU-specific | Any MCU | Beyond MCU world |

Portability level

ST's offering. **Free**

Partners' offer

life.augmented

Focus on ST's offer (Free)

STM32Snippets

- ## What is it ?
  - A collection of code examples directly based on STM32 peripheral registers, available in documentation and as software bundles

- ## Target audience
  - Low-level embedded system developers, typically coming from an 8-bit background, used to working in Assembly or C with little abstraction

- ## Features:
  - Highly optimized
  - Register level access
  - Small code expressions
  - Closely follows the reference manual
  - Debugging close to register level

- ## Limitations:
  - Specific to STM32 devices, not directly portable between series
  - Not matching complex peripherals such as USB
  - Lack of abstraction means developers must understand peripheral operation at register level
  - Available (today) on STM32 L0 and F0 series

Portability	Optimization (Memory & MIPS)	Ease of use	Readiness	Hardware coverage
	+++			+

Standard peripheral libraries (SPL)

- ## What is it ?
  - Collection of C libraries covering STM32 peripherals

- ## Target audience
  - Embedded systems developers with procedural C background. All existing STM32 customers prior to the STM32Cube launch, willing to keep same supporting technology for future projects, and same STM32 series.

- ## Features:
  - Average optimization, fitting lots of situations
  - No need for direct register manipulation
  - 100% coverage of all peripherals
  - Easier debugging of procedural code
  - Extensions for complex middleware such as USB/TCP-IP/Graphics/Touch Sense

- ## Limitations:
  - Specific to certain STM32 series.
  - No common HAL API prevents application portability between series
  - Middleware libraries may not be unified for each series
  - Doesn't support forward STM32 series starting with STM32 L0, L4 and F7

| Portability	Optimization (Memory & MIPS)	Ease of use	Readiness	Hardware coverage
++	++	+	++	+++

STM32Cube - Embedded software

Introduction

User code

STM32Cube
Middleware

LwIP TCP/IP | | | | RTOS

STM32Cube

| STM32Cube HAL | STM32Cube LL APIs |

- ## What is it ?
  - Full-feature packages with drivers, USB, TCP/IP, graphics, file system and RTOS
  - Set of common application programming interfaces, ensuring high portability inside whole STM32 family
  - Set of APIs directly based on STM32 peripheral registers
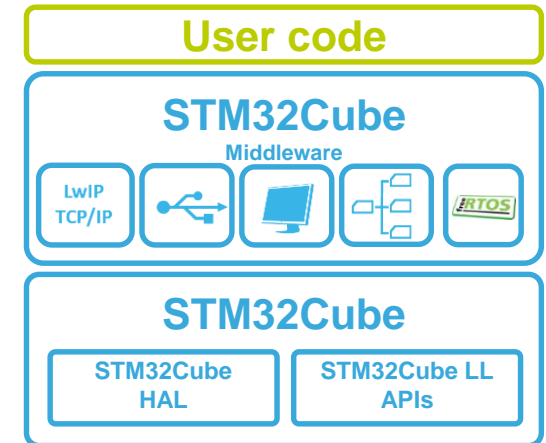
- ## Target audience
  - Hardware Abstraction Layer (HAL) APIs: Embedded system developers with a strong structured background. New customers looking for a fast way to evaluate STM32 and easy portability.
  - Low-Layer (LL) APIs: Low-level embedded system developers, typically coming from an 8-bit background, used to working with Assembly or C with little abstraction. Customers migrating from the SPL environment.
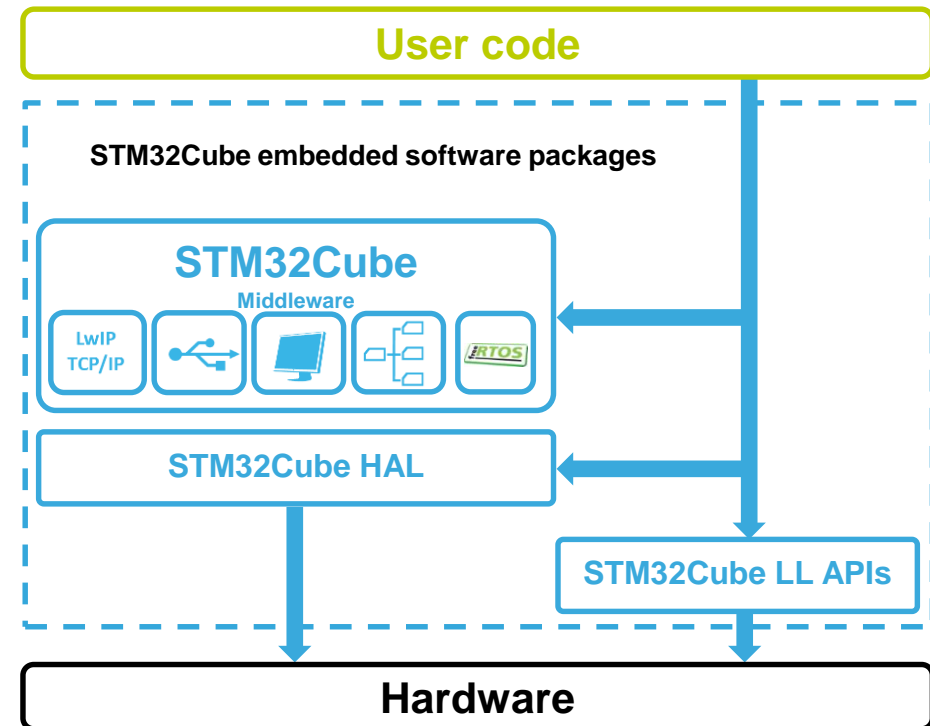
life.augmented

STM32Cube - Embedded software

Architecture overview

- Three entry points for the user application:
  - Middleware stacks
  - HAL APIs
  - LL APIs

- Possible concurrent usage of HAL and LL
  - Limitation: LL cannot be used with HAL for the same peripheral instance. Impossible to run concurrent processes on the same IP using both APIs, but sequential use is allowed.
  - Example of hybrid model:
    - Simpler static peripheral initialization with HAL
    - Optimized runtime peripheral handling with LL calls

User code

STM32Cube embedded software packages

STM32Cube
Middleware

LwIP TCP/IP

RTOS

STM32Cube HAL

STM32Cube LL APIs

Hardware

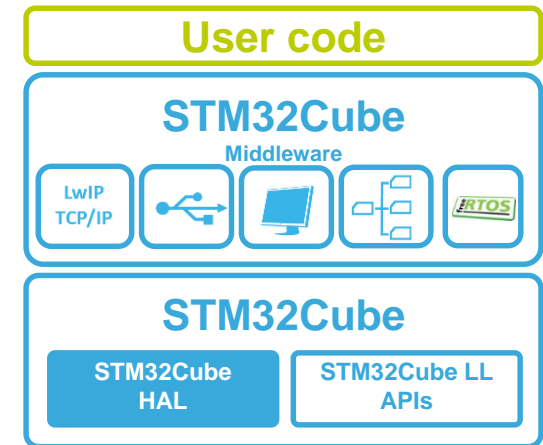life.augmented

STM32Cube - Embedded software

HAL APIs

- ## Features:
  - High level and functional abstraction
  - Easily portable from one series to another
  - 100% coverage of all peripherals
  - Integrates complex middleware such as USB/TCP-IP/graphics/Touch Sense/RTOS
  - Can work with STM32CubeMX tool to generate initialization code

User code

STM32Cube
Middleware

LwIP TCP/IP

STM32Cube

| STM32Cube HAL | STM32Cube LL APIs |

- ## Limitations:
  - May be challenging to low-level C programmers in the embedded space.
  - Higher portability creates bigger software footprints or more time spent executing adaptation code.

Portability	Optimization (Memory & MIPS)	Ease of use	Readiness	Hardware coverage
+++	+	++	+++	+++

life.augmented

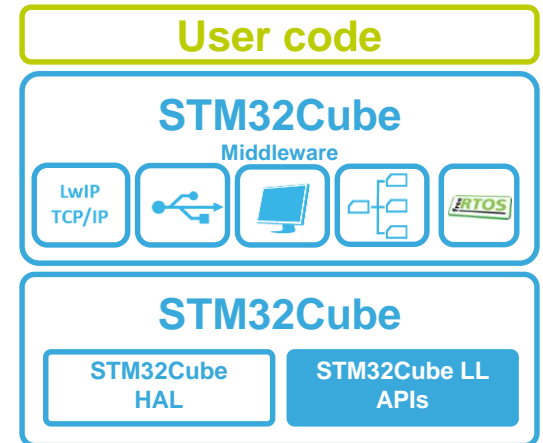# STM32Cube - Embedded software

Low-Layer APIs

- **Features:**
  - Highly optimized
  - Register level access
  - Small code expressions
  - Closely follows the reference manual
  - Debugging close to register level

User code

STM32Cube
Middleware

LwIP TCP/IP

STM32Cube

| STM32Cube HAL | STM32Cube LL APIs |

- **Limitations:**
  - Specific to STM32 devices, not directly portable between series
  - Not matching complex peripherals such as USB
  - Lack of abstraction means developers must understand peripheral operation at register level
  - Available (today) on STM32 L4 series

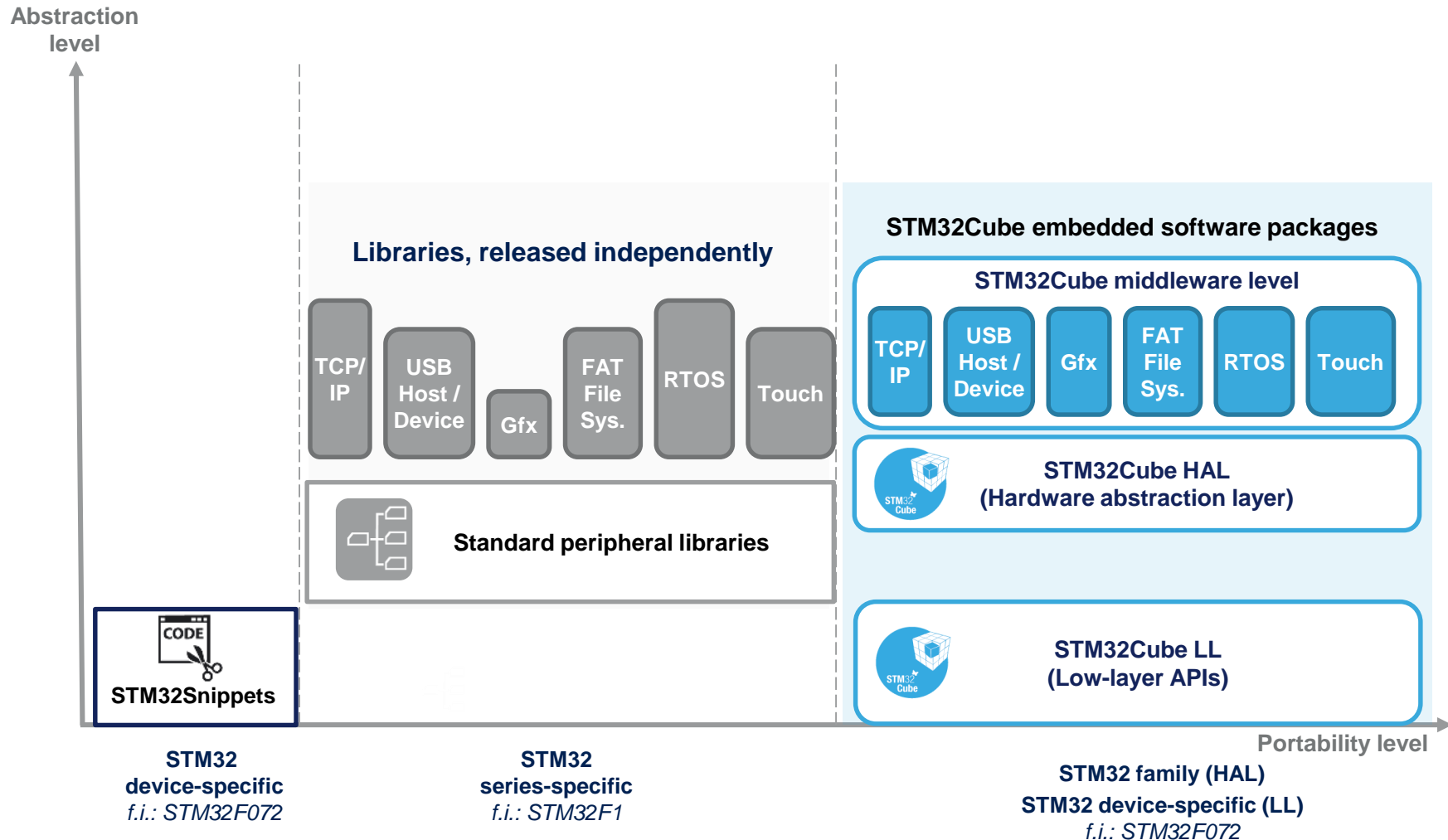| Portability	Optimization (Memory & MIPS)	Ease of use	Readiness	Hardware coverage
	+++			++

life.augmented

ST embedded software offer – comparison

Offer		Portability	Optimization (Memory & MIPS)	Ease of use	Readiness	Hardware coverage
STM32Snippets			+++			+
Standard peripheral library		++	++	+	++	+++
STM32 Cube	HAL APIs	+++	+	++	+++	+++
	LL APIs		+++			++

life.augmented

ST embedded software offer - positioning

Offer	Available for STM32								
	STM32 F0	STM32 F1	STM32 F3	STM32 F2	STM32 F4	STM32 F7	STM32 L0	STM32 L1	STM32 L4
STM32Snippets	Now	N.A.	N.A.	N.A.	N.A.	N.A.	Now	N.A.	N.A.
Standard Peripheral Library	Now	Now	Now	Now	Now	N.A.	N.A.	Now	N.A.
STM32Cube HAL	Now	Now	Now	Now	Now	Now	Now	Now	Now
STM32Cube LL	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	Now

life.augmented

What solution to choose? An FAQ

1. **I want to use a small footprint MCU; what should I use?**

   Abstraction has a cost. Therefore, if you need to take advantage of every single bit of memory, STM32Snippets or STM32Cube LL will be the best choice.

2. **I come from an 8-bit MCU world; what should I use?**

   If you prefer direct register manipulation, then the STM32Snippets or STM32Cube LL would be a good starting point. However, if you prefer structure 'C' level programming, then we recommend using the STM32Cube HAL or SPL.

3. **I today use SPL on STM32F103; should I switch to STM32Cube?**

   If you intend to use only MCUs that are part of the same series in the future (in this case STM32F1 series), then you should carry on with using SPL.

   If you plan to use different STM32 series in the future, then we recommend considering STM32Cube as this will make it much easier to move between series.

4. **I need a mix of portability and optimization; what can I do?**

   You can use the STM32Cube HALs and replace some of the calls with your specific optimized implementations, thus keeping maximum portability and isolating areas that are not portable, but optimized.

   HALs and LLs being partially usable concurrently (no possible concurrent runtime HAL and LL processes for the same peripheral), it is also possible to use a hybrid HAL and LL implementation to get the same advantages as mentioned above.

Migrating between offers

From	To		
	STM32Snippets	SPL	STM32Cube
STM32Snippets	Easy within same STM32 series Ex: between STM32F072 and STM32F030	No simple migration path. Application must be rewritten	No simple migration path to HALs. Application must be rewritten
	Almost not possible between different series Ex: between STM32F072 and STM32L053		Easy within same STM32 series when using LLs Ex: between STM32F072 and STM32F030
Standard peripheral library (SPL)	Some (but not all) SPL functions can be replaced with Snippets	Easy within same STM32 series Ex: Between STM32F401 and STM32F429	No simple migration path to HALs. Application must be rewritten
		Difficult between different STM32 series Ex: between STM32F100 and STM32F407	SPL functions can be replaced with LL calls to ease that migration
STM32Cube embedded software package — HAL APIs	Some (but not all) HAL functions can be replaced with Snippets	No simple migration path. Application must be rewritten	Yes, across all STM32 families
STM32Cube embedded software package — LL APIs	LL calls equivalent to snippets when addressing the exact same peripheral	Sets of LL calls can be replaced with SPL calls	Almost not possible between different series Ex: between STM32F407 and STM32L476

/STM32

@ST_World

st.com/e2e

www.st.com/stm32embeddedsoftware