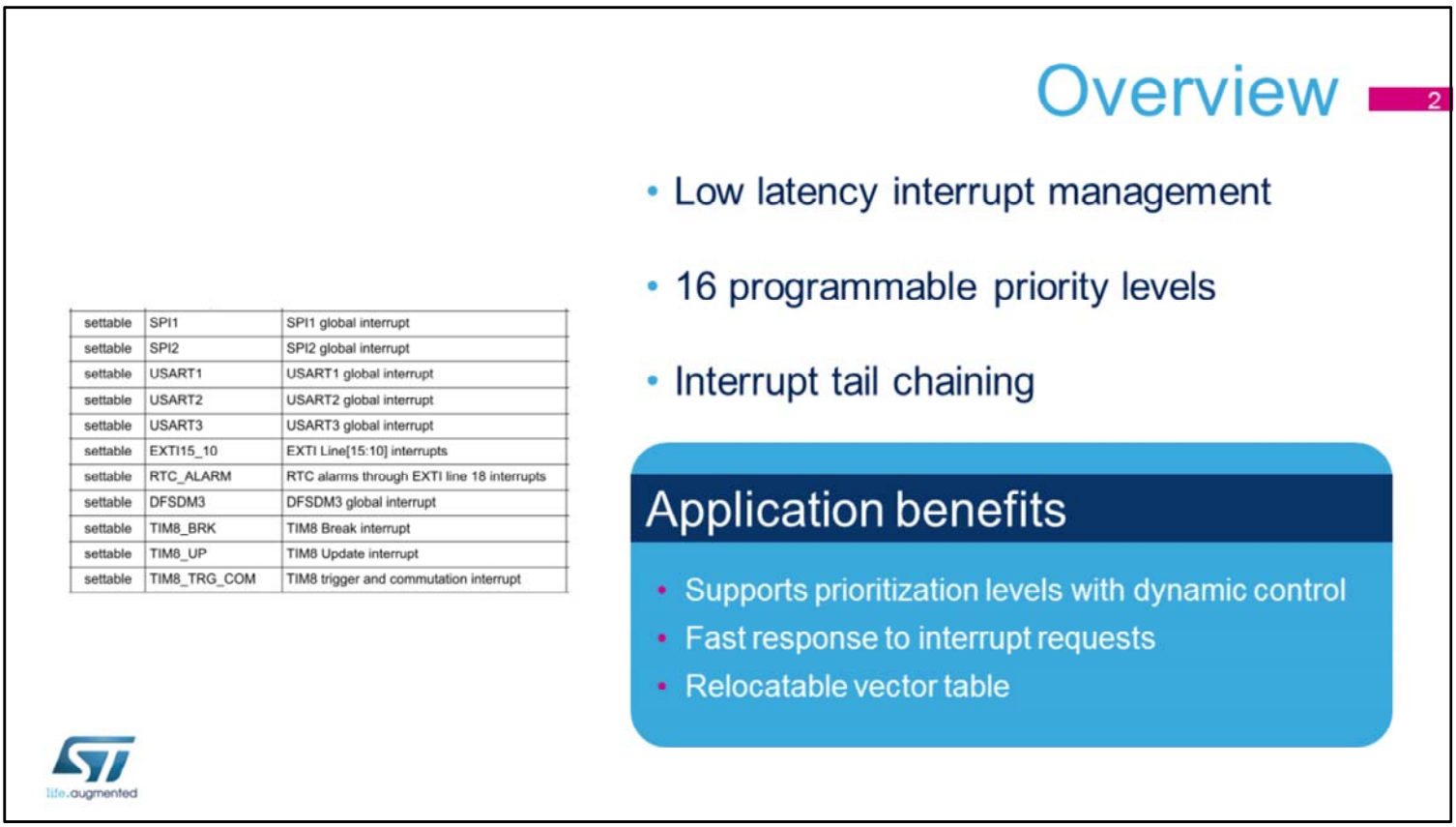
Hello, and welcome to this presentation of the STM32 nested vectored interrupt controller. We will be presenting the features of this controller.



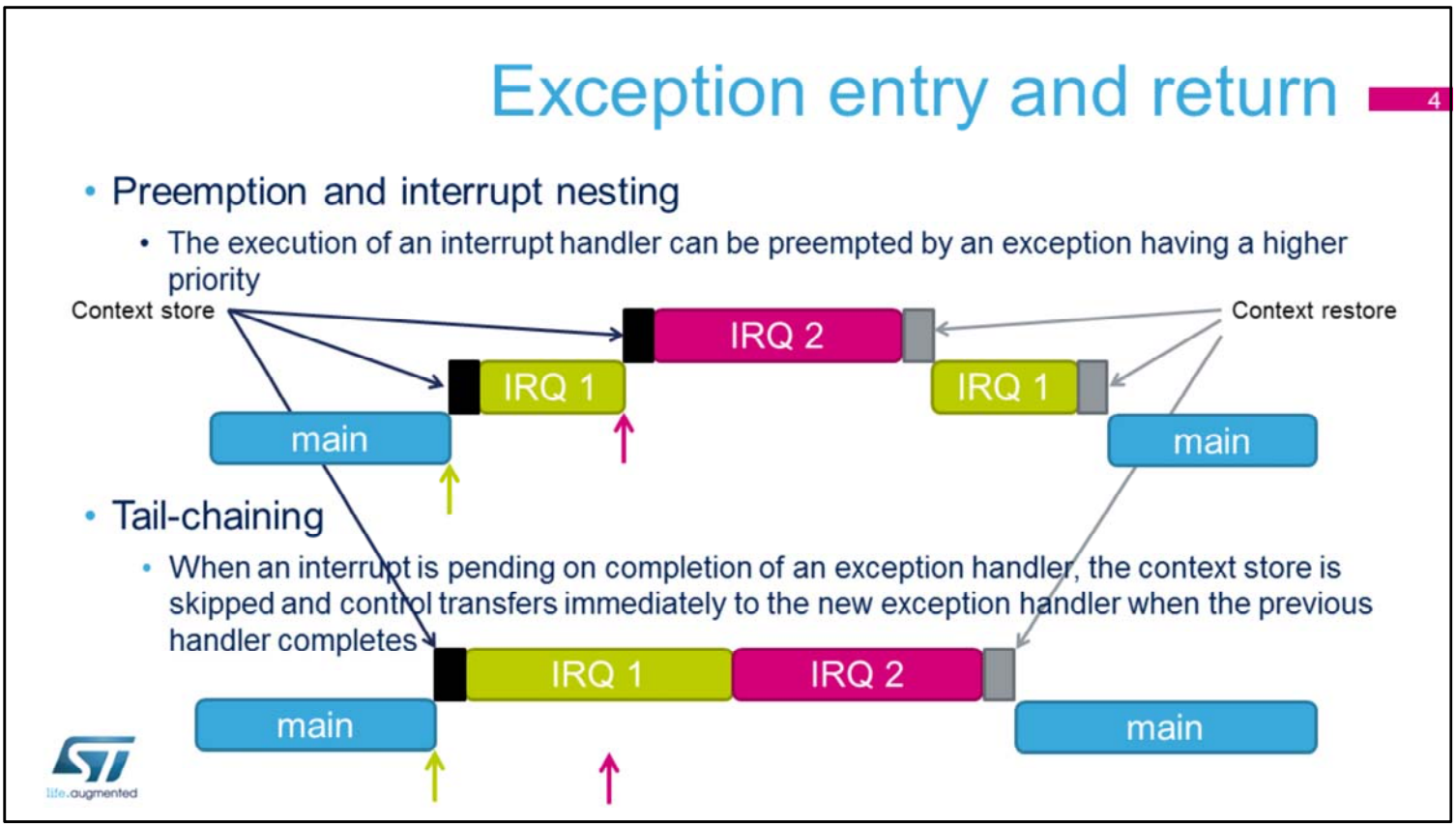
settable	SPI1	SPI1 global interrupt
settable	SPI2	SPI2 global interrupt
settable	USART1	USART1 global interrupt
settable	USART2	USART2 global interrupt
settable	USART3	USART3 global interrupt
settable	EXTI15_10	EXTI Line[15:10] interrupts
settable	RTC_ALARM	RTC alarms through EXTI line 18 interrupts
settable	DFSDM3	DFSDM3 global interrupt
settable	TIM8_BRK	TIM8 Break interrupt
settable	TIM8_UP	TIM8 Update interrupt
settable	TIM8_TRG_COM	TIM8 trigger and commutation interrupt

The Nested Vectored Interrupt Controller embedded inside of the STM32L4 microcontroller provides up to 91 interrupt channels (on STM32L49x/4A6 devices), served with low latency. One of 16 priorities could be assigned to each interrupt source.
Application could benefit from dynamic prioritization of the interrupt levels, fast response to the requests thanks to low latency response and tail chaining and also from vector table relocation.

Key features

- Fast response to the interrupt requests
  - Most peripherals have a dedicated interrupt
- Dynamic reprioritization of interrupts
- Dynamic relocation of interrupt vector table

The NVIC provides a fast response to interrupt requests, allowing an application to quickly serve incoming events. Most of the peripherals have a unique interrupt vector, making development of the application easier (with less need to programmatically determine the source of an interrupt during processing). The interrupt vector table can also be relocated, which allows the system designer to adapt the placement of interrupt service routines to the application's memory layout.



The NVIC provides several features for efficient handling of exceptions.
When an interrupt handler is served and a new request with higher priority arrives, the new exception can preempt the current one. This is called nested exception handling. The previous exception handler resumes execution after the higher priority exception is handled.
When there is an interrupt request with low priority raised during execution of an interrupt handler, it becomes pending. Once current interrupt handler is finished, the context saving and restoring process is skipped and control is transferred directly to the new exception handler to decrease interrupt latency.

Exception entry and return

- Late-arriving
  - When higher-priority exception occurs during state saving for a previous exception, the processor switches to handle the higher priority exception immediately.

- Return
  - When the exception handler is finished and no other exception is pending, the processor pops the stack and restores the program state before the interrupt occurred.

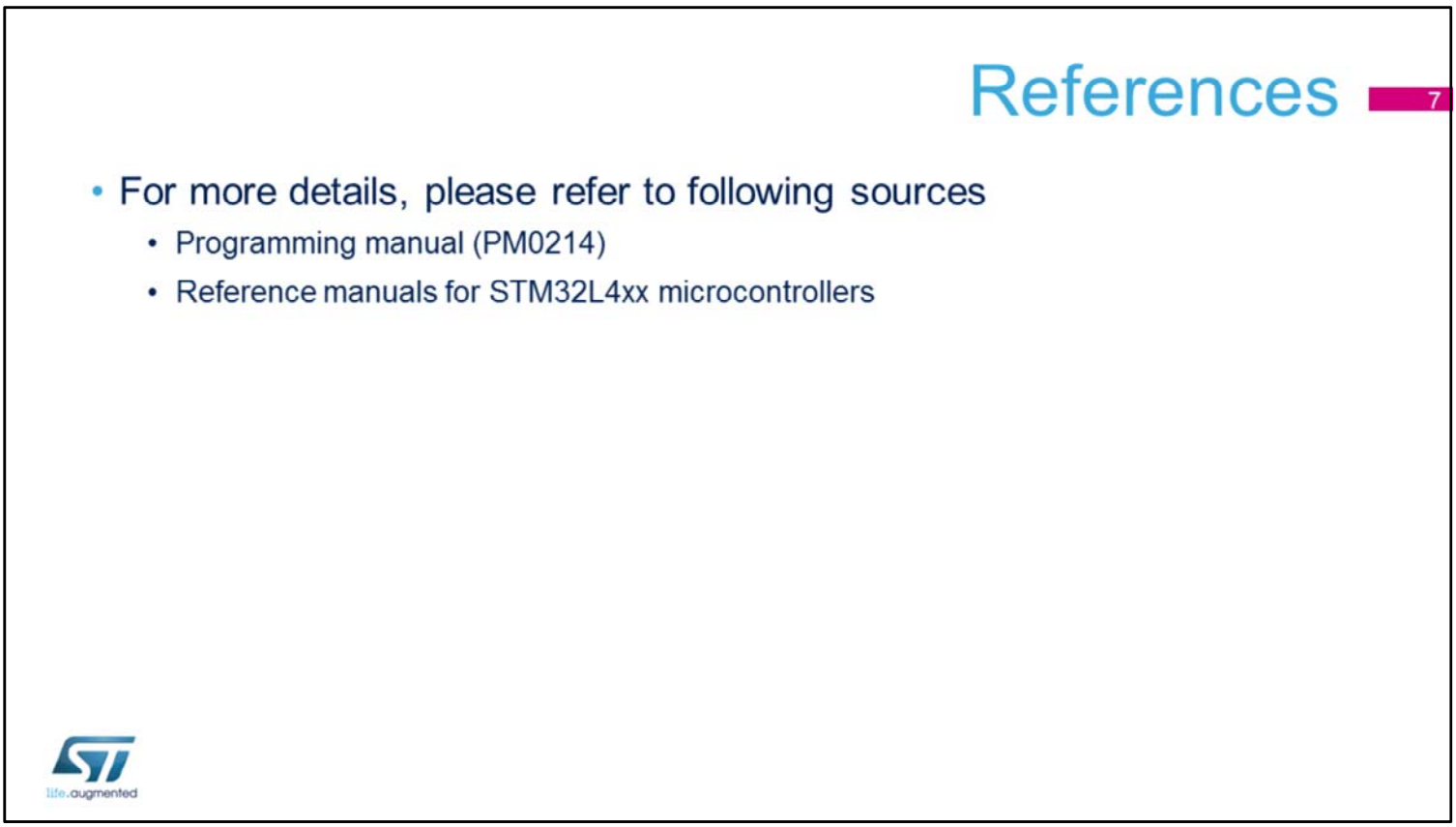
When an interrupt arrives, the processor first saves the program context before executing the interrupt handler. If the processor is performing this context-saving operation when an interrupt of higher priority arrives, the processor switches directly to handling the higher-priority interrupt when it is finished saving program context .
When all of the exception handlers have been run and no other exception is pending, the processor restores the previous context from the stack and returns to normal application execution.

NVIC hints and tips

- Ensure software uses correctly-aligned register accesses
- An interrupt can become pending even if disabled
  - Disabling an interrupt only prevents the processor from taking that interrupt
- Before relocating the vector table, ensure new entries are setup for all enabled interrupts
  - This includes fault handlers and NMI
  - Do this before programming VTOR to relocate vector table

life.augmented

When accessing the NVIC registers, ensure that your code uses correctly-aligned register access. Unaligned access is not supported for NVIC registers.
Interrupt becomes pending when the source ask for service. Disabling of the interrupt only prevent processor from taking that interrupt. Make sure related interrupt flag is cleared before enabling the interrupt vector.
Before relocating the vector table using VTOR register, ensure fault handlers, NMI and all enabled interrupts are correctly set up on new location.



For detailed information, please refer mainly to programming manual PM0214.