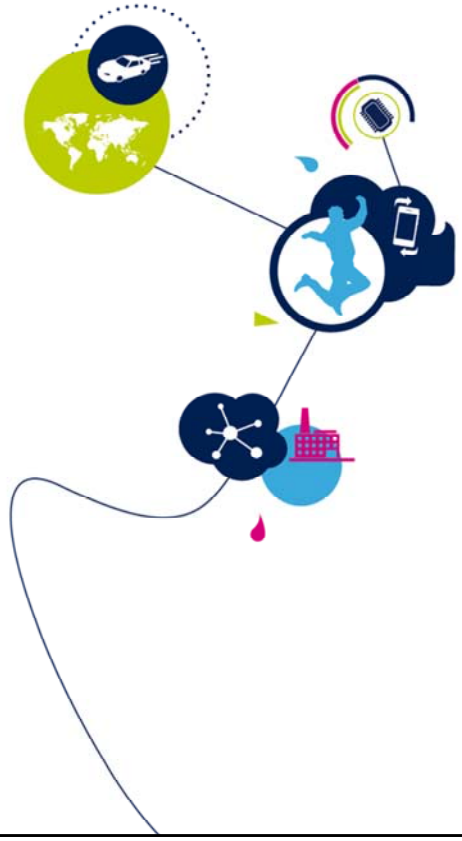
STM32G4- EXTI

Extended interrupts and events controller
Revision 1.0



life.augmented

Hello, and welcome to this presentation of the STM32G4 Extended Interrupts and Events Controller. We will be presenting the features of the EXTI controller.

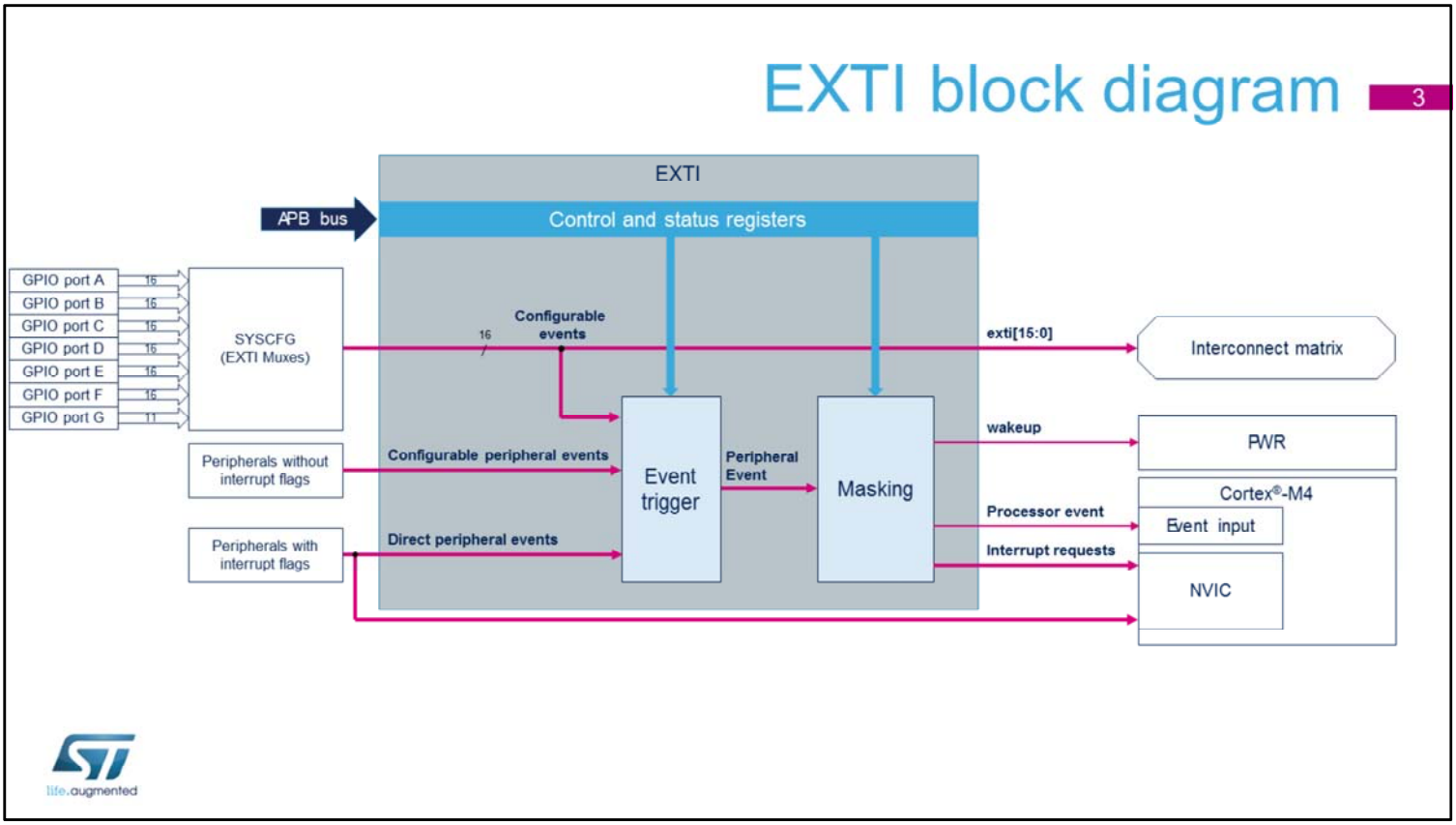
Overview

- 42 events / interrupt lines
  - 28 configurable events
  - 14 direct events
- Independent masks and configuration
- Software can emulate events or interrupts by writing to a dedicated register

Application benefits

- Manage external and internal wakeup events and interrupts
- Provide pending flag for Configurable events

The Extended interrupt and event controller (EXTI) provides up to 42 independent events, split into two categories – configurable events and direct events. Applications benefit through smarter use of low-power modes, taking advantage of the STM32G4's capability to wake up via external communication or requests.



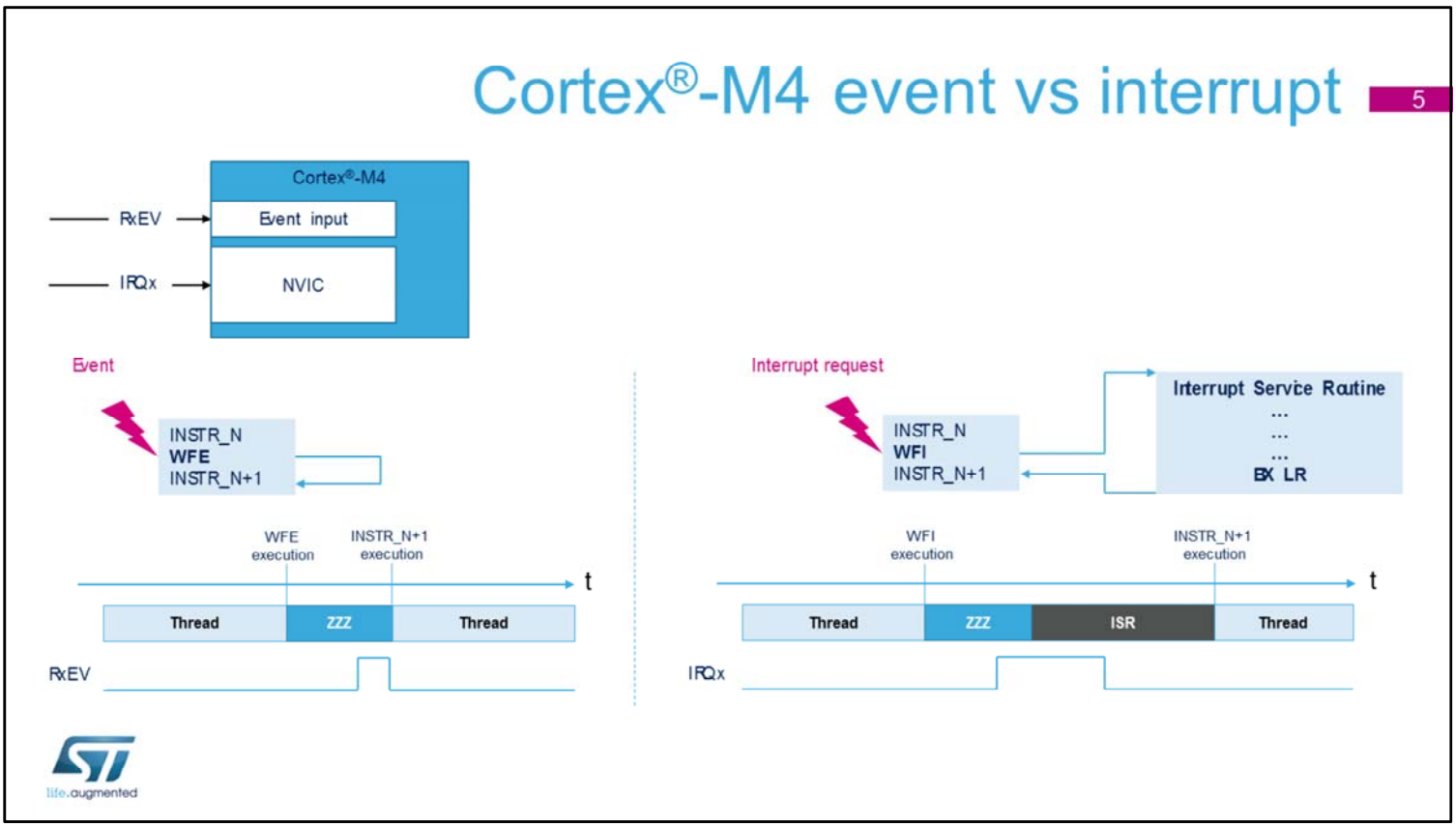
This is the block diagram of the extended interrupt and event controller.
Configurable peripheral events are generated by peripherals without interrupt capability, but which are able to issue a pulse. The EXTI controller provides interrupt detection, masking and software trigger.
Direct peripheral events are generated by peripherals supporting interrupt requests. In this case, the EXTI controller is used to generate events to the CPU and to request system wakeups.
Do not confuse peripheral events and processor event. Peripheral events are used by peripherals to indicate that they require processor attention.
The processor event is a pulse signal used by Arm CPUs to exit the Wait For Event low power state.

Key features

- Wake-up from Stop mode, interrupts and events generation
  - Independent interrupt and event masks
- Configurable events
  - Active edge selection
  - Dedicated pending flag
  - Trigger-able by software
  - Linked to:
    - GPIO, PVD, RTC, timestamp or CSS LSE, COMPx and PWMx
- Direct events
  - Status flag provided by related peripheral
  - Linked to:
    - USB, I2Cx, UARTx, USARTx, LPUART1, LPTIM1, and UCPDx

life.augmented

The Extended Interrupt and event controller can generate interrupt and event as well as wake up the processor from Stop modes.
Configurable events are linked with external interrupts from GPIOs, PVD, RTC, timestamp or CSS LSE, comparators and PWM modules.
Direct events are linked with USB, I2C, UART, USART, LPUART1, LPTIM1, and UCPD1.



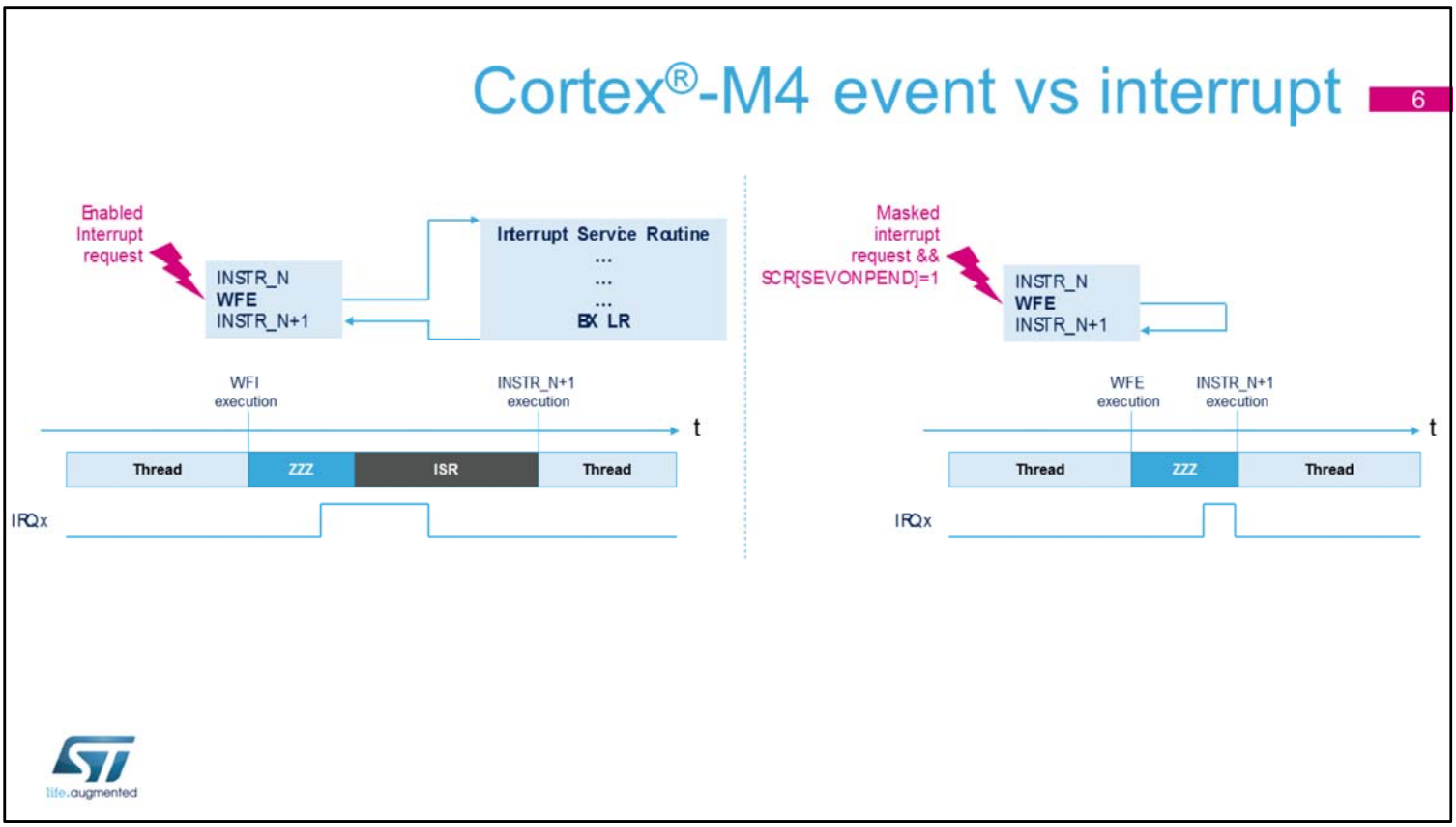
The Cortex®-M4 supports two ways to enter a low-power state:

1. Executing the Wait For Event (WFE) instruction
2. Executing the Wait For Interrupt (WFI) instruction.

With WFE, the first instruction executed after a wake-up event is the next sequential one, INSTR_N+1 in the sequence on the left.

By implementing WFI, the processor jumps to the Interrupt Service Routine when an enabled interrupt request is received.

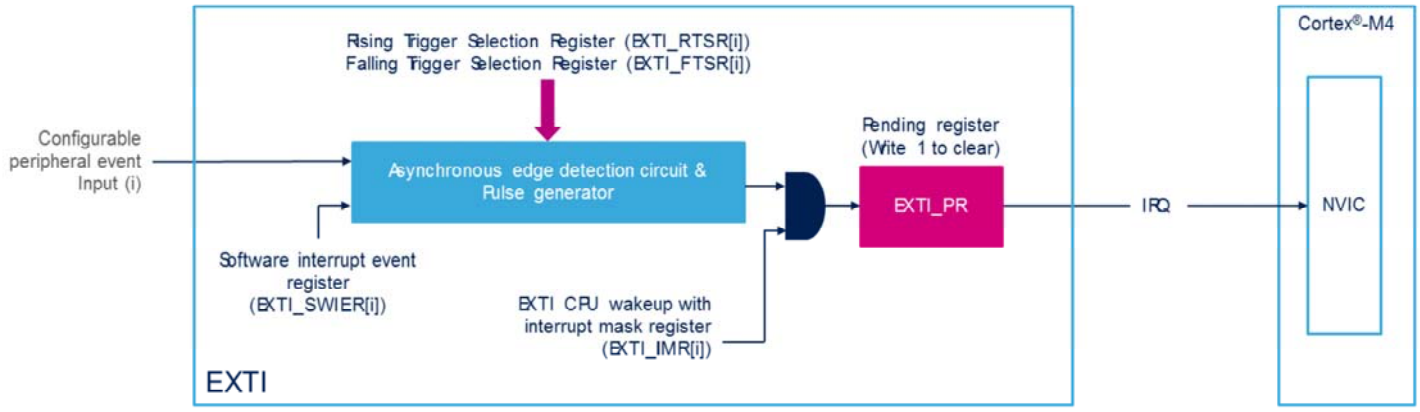
Note that an interrupt request is a WFE exit condition, but an event received on RXEV is not a WFI exit condition.



Receiving an enabled interrupt request while the processor is in WFE state causes a wakeup of the processor and the execution of the interrupt service routine.
When the Cortex-M4 control bit called SEVONPEND, which means Send Event on Pending, is set to one, receiving an interrupt request related to a masked interrupt causes a wakeup event. In this case the processor executes the next sequential instruction. Software may later decide to enable the next interrupt to be served.

Interrupt generation

- Using configurable events as interrupt requests:

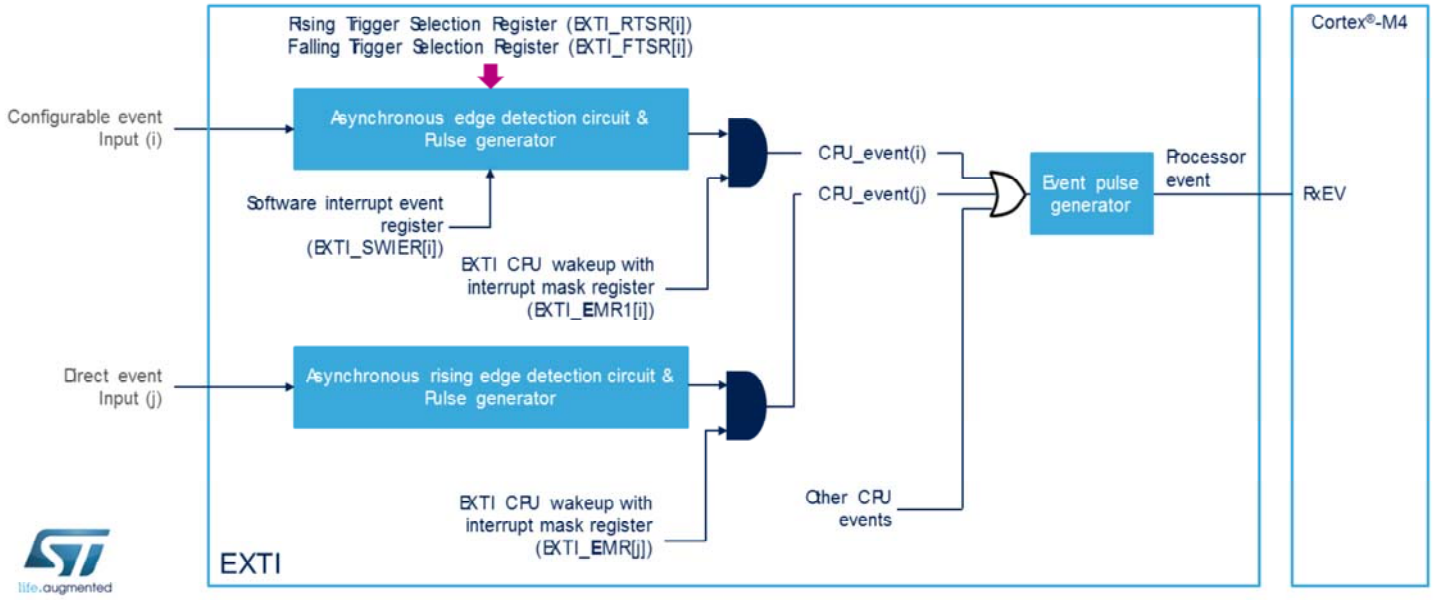


This figure explains the various stages enabling the conversion of a configurable peripheral event active edge into an interrupt request.
The first stage is the asynchronous edge detection circuit configured by two registers EXTI_RTSR and EXTI_FTSR.
Any edge, possibly both, can be chosen.
The software can emulate a configurable event by setting the corresponding bit in the EXTI_SWIER register.
The bit is auto-cleared by hardware.
An AND gate is used to mask or enable the generation of the interrupt to the NVIC.
Finally, a flag is set in the EXTI_PR register when the interrupt is generated to the NVIC. This flag enables the software to determine the cause of the interrupt.
This flag is expected to be cleared by the interrupt

service routine.

CPU event generation

- Using configurable and direct events as CPU event request:

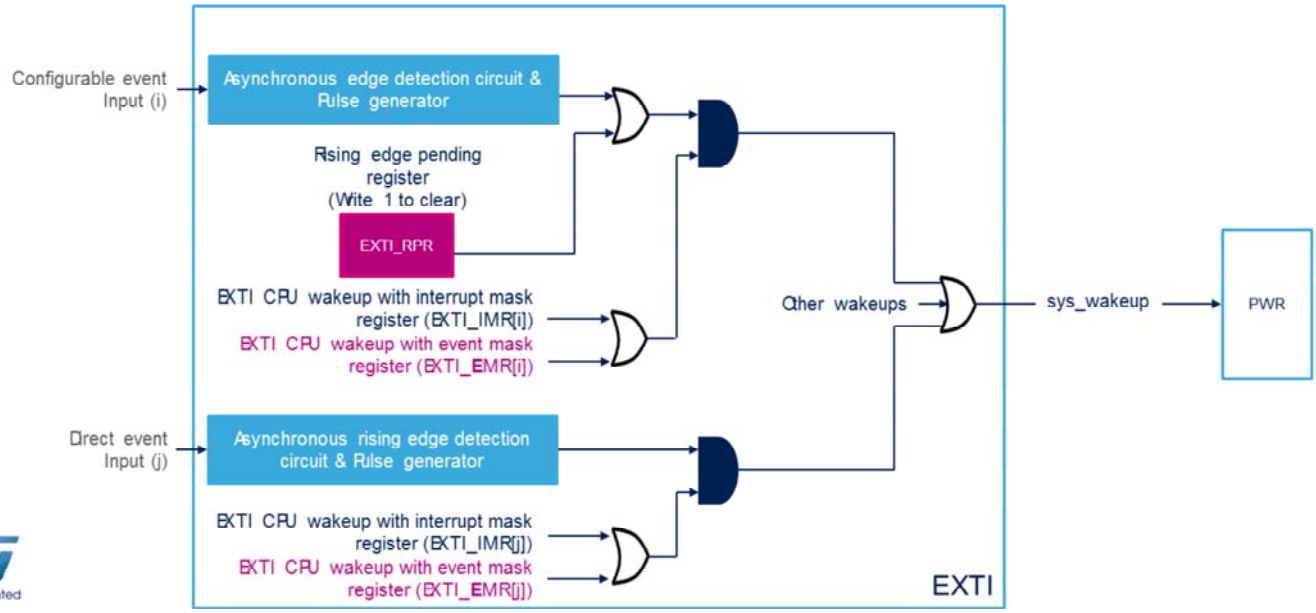


This figure explains the various stages enabling the conversion of a peripheral event active edge into a processor event.
Both Configurable and Direct peripheral events can be configured to issue events to the CPU, steered to its RxEV input.
Configurable event active edge is programmable in the EXTI_RTSR and EXTI_FTSR registers while direct events are always sensitive to a rising edge.
Software can emulate a configurable event by writing to the EXTI_SWIER register.
Unlike interrupt requests, the CPU has a unique event input, so all event requests are ORed together before entering the Event pulse generator.
The registers used to mask the generation of events are different from the ones used to mask the generation of interrupts: EXTI_EMR instead of EXTI_IMR.

Note that unlike interrupt requests, the pending bit corresponding to the peripheral event line is not set.

Wakeup event generation

- Using configurable and direct events as core and system wakeup requests:



The CPU wakeup signals generated by the EXTI block are connected to the PWR block, and are used to wake up the system and CPU sub-system bus clocks.
Both configurable and direct peripheral events are able to request a wakeup.
A wakeup occurs when an asynchronous edge detection circuit has detected an active edge or a flag is set to one in the EXTI_RPR register.
Consequently, software is expected to clear the flag in the EXTI_RPR register to disable the wakeup request when the source of the wakeup is a configurable event.
For direct events, the flag is located in the peripheral unit.
These flags enable the software to find the cause of the wakeup.
The wakeup indication is asserted when either the interrupt or the event generation is enabled, see the OR

gate combining EXTI_IMR and EXTI_EMR registers.

Direct event trigger logic CPU wakeup

- The direct events do not have an associated EXTI interrupt
  - The EXTI only wakes up the system and CPU sub-system clocks and may generate a CPU wakeup event
  - The peripheral interrupt, associated with the direct wakeup event wakes up the CPU
- The EXTI direct event is able to generate a processor event

life.augmented

A direct event is able through the EXTI controller to generate a CPU event and trigger a system wakeup.
The active edge of direct events is the rising edge.
Direct events do not rely on the EXTI controller to assert interrupt requests, because they have their dedicated lines to the NVIC.
Otherwise the same circuit as the one described in the previous slides is implemented. Direct events can be independently masked for event generation and interrupt generation. The interrupt mask is only used as a wakeup mask.

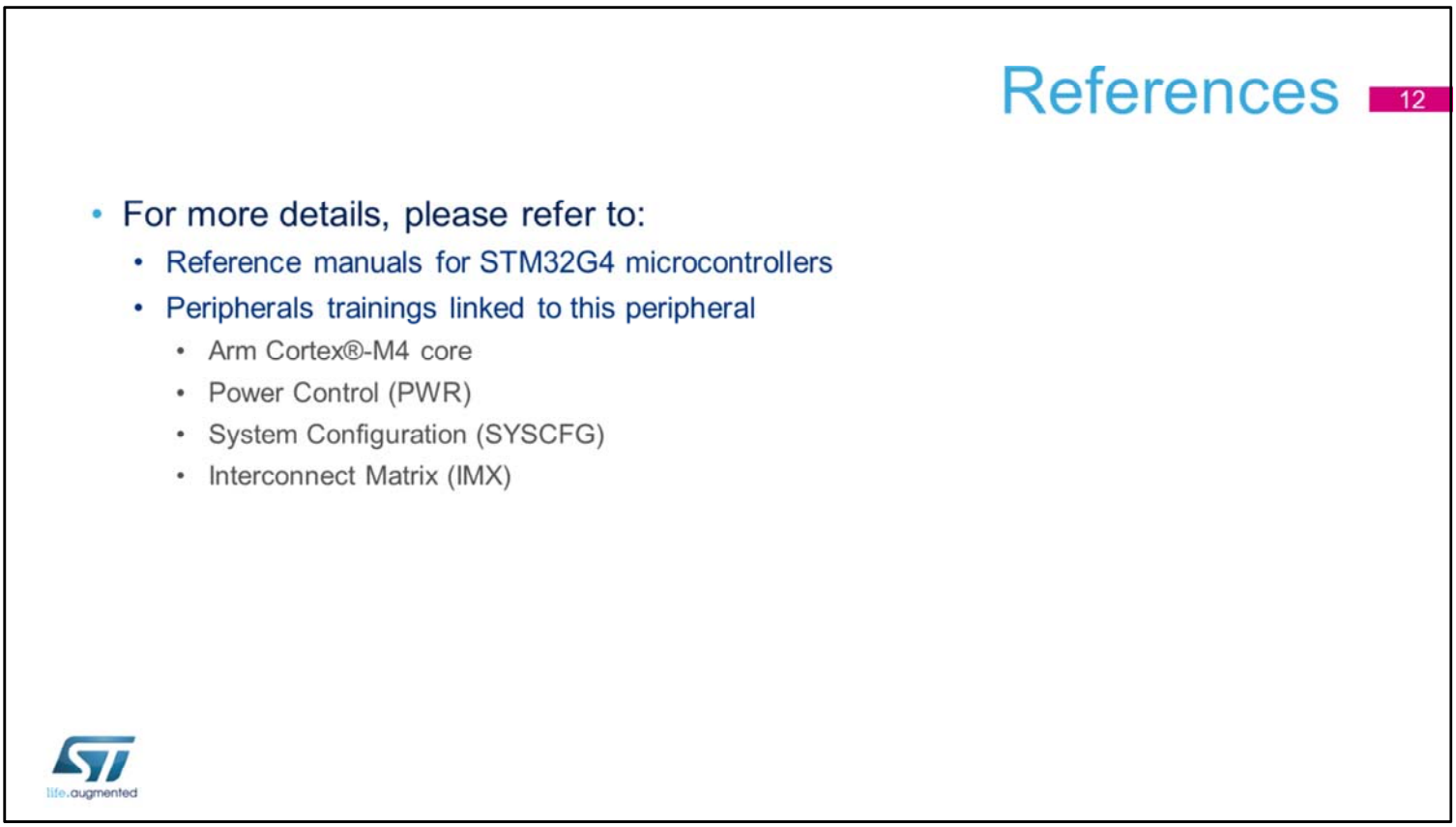
EXTI lines mapping

EXTI line	Line source	Line type
0-15	GPIO	Configurable
16	PVD output	Configurable
17	RTC alarm event	Configurable
18	USB device FS wakeup event	Direct
19	Timestamp or CSS_LSE	Configurable
20	RTC wakeup timer	Configurable
21	COMP1 output	Configurable
22	COMP2 output	Configurable
23	I2C1 wakeup	Direct
24	I2C2 wakeup	Direct
25	USART1 wakeup	Direct
26	USART2 wakeup	Direct
27	I2C3 wakeup	Direct
28	USART3 wakeup	Direct
29	COMP3 output	Configurable
30	COMP4 output	Configurable
31	COMP5 output	Configurable
32	COMP6 output	Configurable
33	COMP7 output	Configurable

EXTI line	Line source	Line type
34	UART4 wakeup	Direct
35	UART5 wakeup	Direct
36	LPUART1 wakeup	Direct
37	LPTIM1 wakeup	Direct
38	PWM1 wakeup	Configurable
39	PWM2 wakeup	Configurable
40	PWM3 wakeup	Configurable
41	PWM4 wakeup	Configurable
42	I2C4 wakeup	Direct
43	UCPD1 wakeup	Direct

life.augmented

This table provides all inputs of the EXTI block present in the STM32G4 microcontroller and indicates for each of them whether it is a configurable event input or a direct event input.



For more details about the System Configuration module, refer to the reference manual for STM32G4 microcontrollers.
Refer also to these trainings for more information if needed:

- Arm Cortex-M4 core
- Power control (PWR)
- System Configuration (SYSCFG)
- Interconnect Matrix (IMX).