Hello, and welcome to this presentation of the STM32G0 Extended Interrupts and Events Controller. We will be presenting the features of the EXTI controller.

Overview

- 34 events / interrupt lines
  - 19 configurable events
  - 15 direct events
- Independent masks and configuration

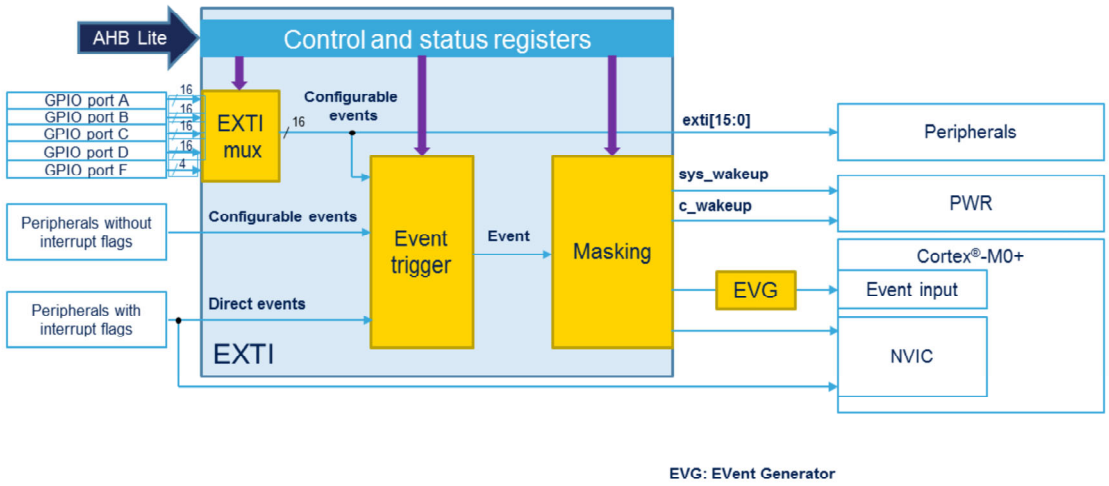
Application benefits

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

life.augmented

The Extended interrupt and event controller (EXTI) provides up to 34 independent events, split into two categories – configurable events and direct events.
Applications benefit through smarter use of low-power modes, taking advantage of the STM32G0's capability to wake up via external communication or requests.

EXTI block diagram



EVG: EVent Generator

This is the block diagram of the extended interrupt and event controller .

Configurable 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.

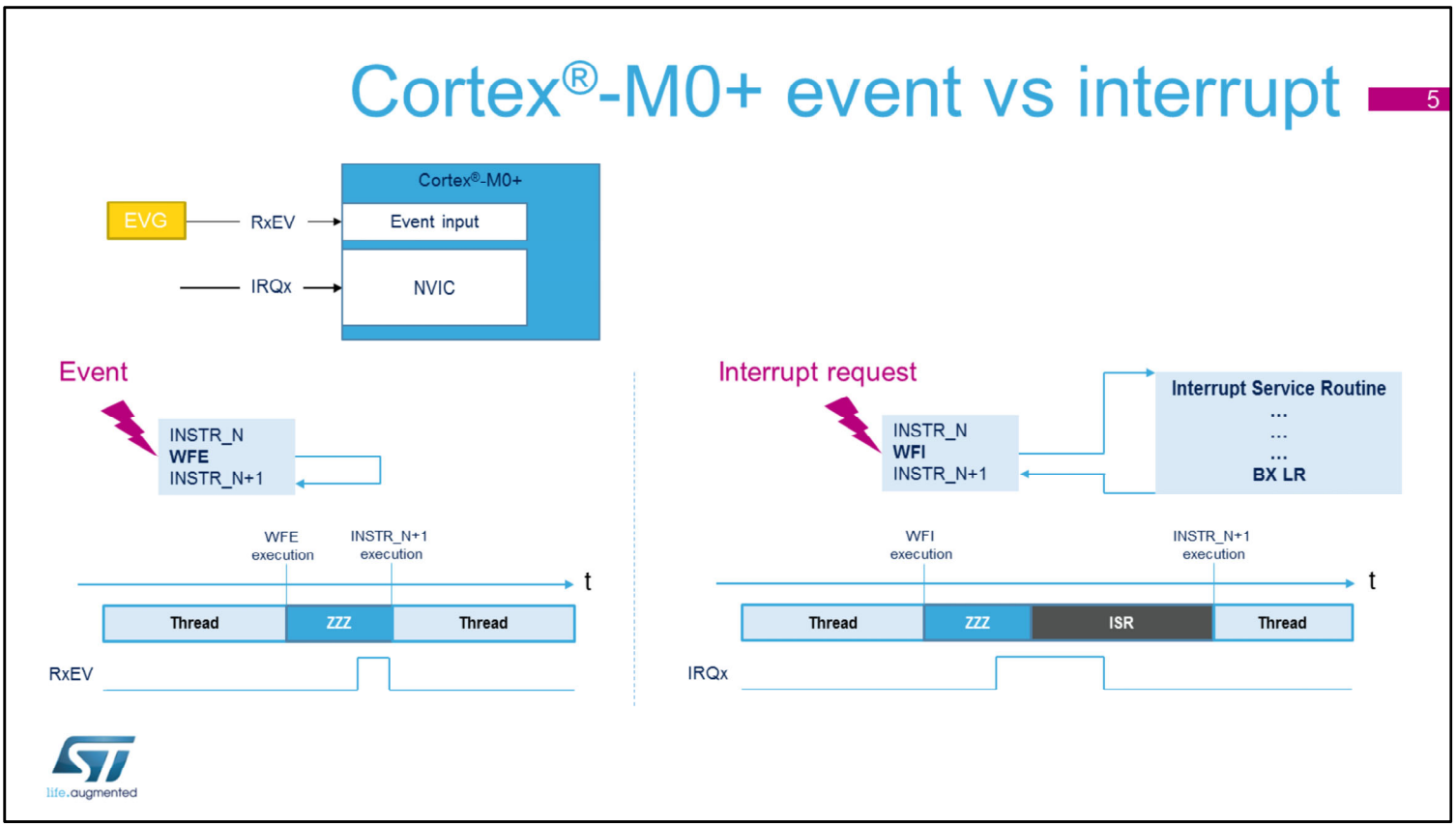
Direct events are generated by peripherals supporting interrupt requests. In this case, the EXTI controller is used to generate events to the CPU and to request system wakeups.

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, and COMPx
- Direct events
  - Status flag provided by related peripheral
  - Linked to:
    - RTC, TAMP, I2C1, USARTx, CEC, LPUART1, LPTIMx, LSE_CSS 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 and comparators COMP1 and COMP2.
Direct events are linked with RTC, Tamper, I2C1, USARTS 1 and 2, CEC, LPUART1, LPTIM 1 and 2, LSE, UCPD 1 and 2.



The Cortex®-M0+ supports two ways to enter a low-power state:

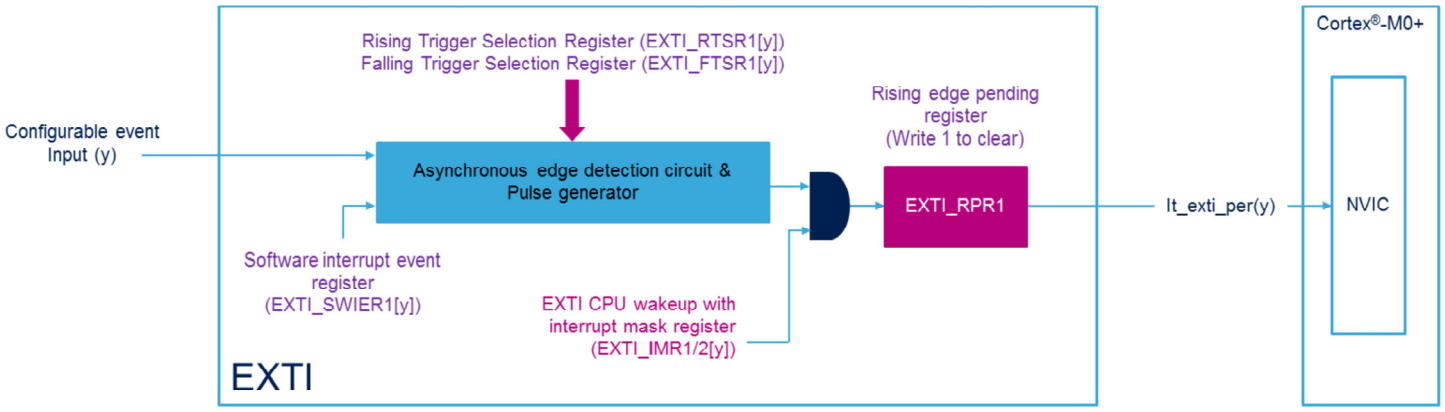
1. Executing the Wait For Event (WFE) instruction
2. Executing the Wait For Interrupt (WFI) instruction.

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.

Interrupt generation

- Using configurable events as interrupt requests:



This figure aims to explain the various stages enabling the conversion of a configurable event active edge into an interrupt request.

The first stage is the asynchronous edge detection circuit configured by two registers EXTI_RTSR1 and EXTI_FTSR1. 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.

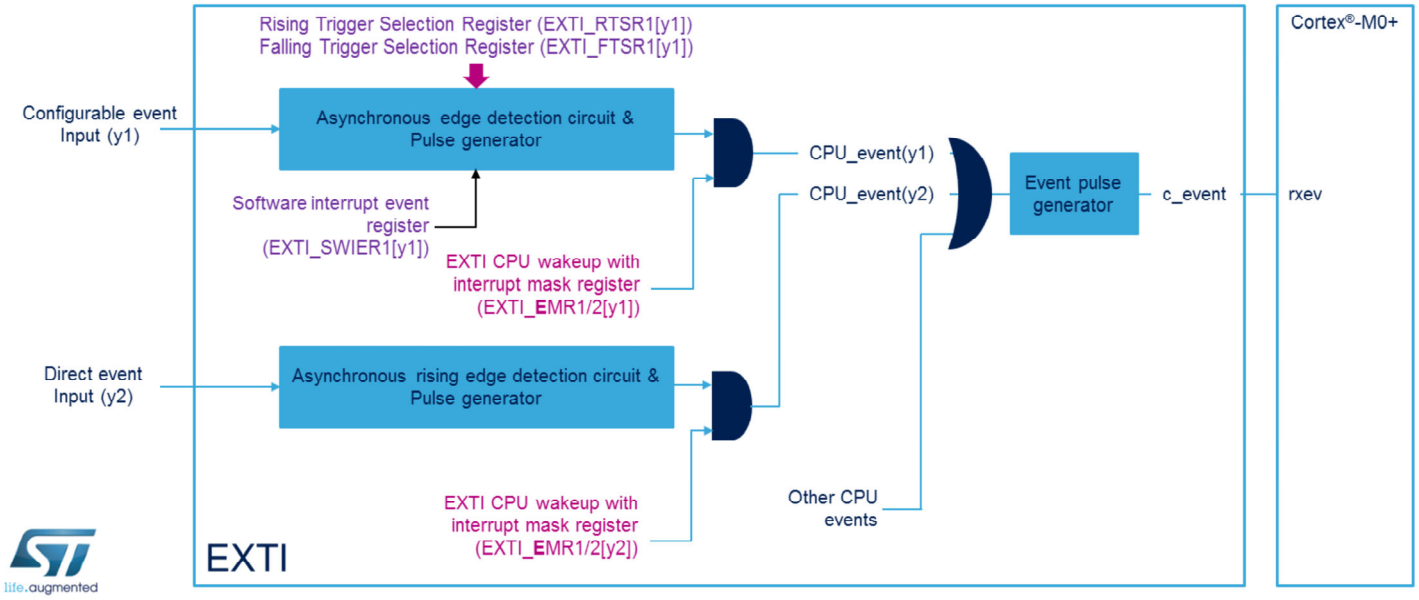
An AND gate is used to mask or enable the generation of the interrupt to the NVIC.

Finally, a flag is set in the EXTI_RPR1 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 aims to explain the various stages enabling the conversion of a configurable event active edge into a processor event.

Both Configurable and Direct events can be configured to issue events to the CPU, steered to its rxev input.

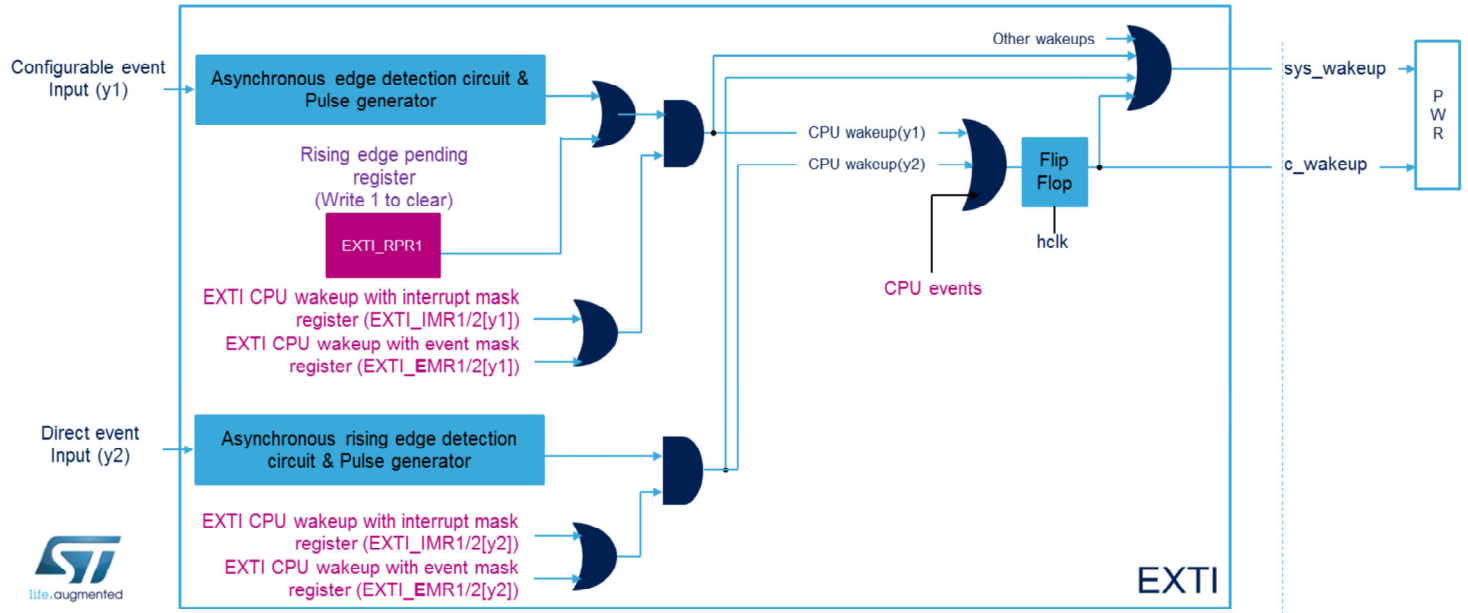
Unlike interrupt requests, the CPU has a unique event input, so all event requests are ORed together before entering the Event pulse generator.

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.

Wakeup event generation

8

- 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 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_RPR1 register.

Consequently, software is expected to clear the flag in the EXTI_RPR1 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.

All CPU wakeup signals are ORed together and then ORed with the event requests.
The sys_wakeup signal is asynchronous and wakes up the clocks. Once hclk is running, the synchronous c_wakeup is generated.

Otherwise the same circuit as the one described in the previous slides is implemented.

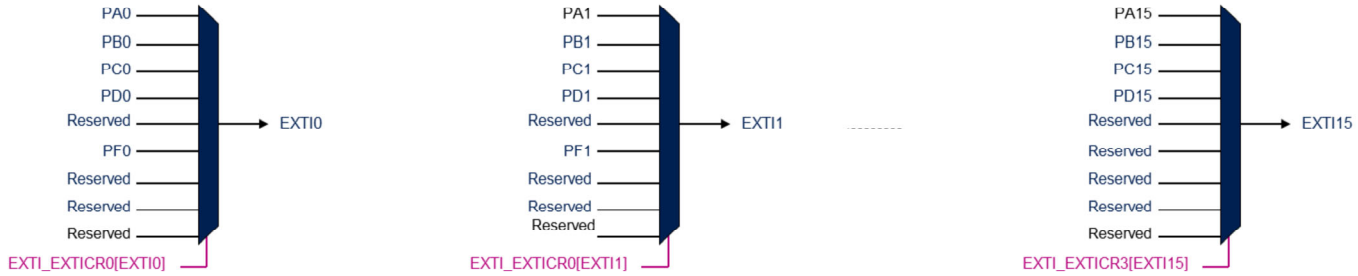
Direct event trigger logic CPU wakeup

- 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 synchronous interrupt, associated with the direct wakeup event wakes up the CPU
- The EXTI direct event is able to generate a CPU 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.

GPIO mux moved from SYSCFG to EXTI



- Two or more GPIO pads having the same number in different ports cannot be selected at the same time as EXTI configurable events.

The STM32G0 has 5 IO ports: Ports A to D are 16 pins wide, port F is 4 pins wide.
Each of the 16 EXTI configurable events related to GPIO ports has an independent multiplexor.
The EXTI multiplexer outputs are available as output signals from the EXTI block to trigger other IPs. The EXTI multiplexer outputs are available independently from any masks defined in the EXTI_IMR and EXTI_EMR registers.

EXTI lines mapping

EXTI line	Line source	Line type
0-15	**GPIO**	**Configurable**
16	**PVD output**	**Configurable**
17	**COMP1 output**	**Configurable**
18	**COMP2 output**	**Configurable**
19	**RTC**	**Direct**
20	Reserved	Direct
21	**TAMP**	**Direct**
22	Reserved	Direct
23	**I2C1 wakeup**	**Direct**
24	Reserved	Direct
25	**USART1 wakeup**	**Direct**
26	**USART2 wakeup**	**Direct**
27	**CEC wakeup**	**Direct**
28	**LPUART1 wakeup**	**Direct**
29	**LPTIM1**	**Direct**
30	**LPTIM2**	**Direct**
31	**LSE_CSS**	**Direct**
32	**UCPD1 wakeup**	**Direct**
33	**UCPD2 wakeup**	**Direct**

life.augmented

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