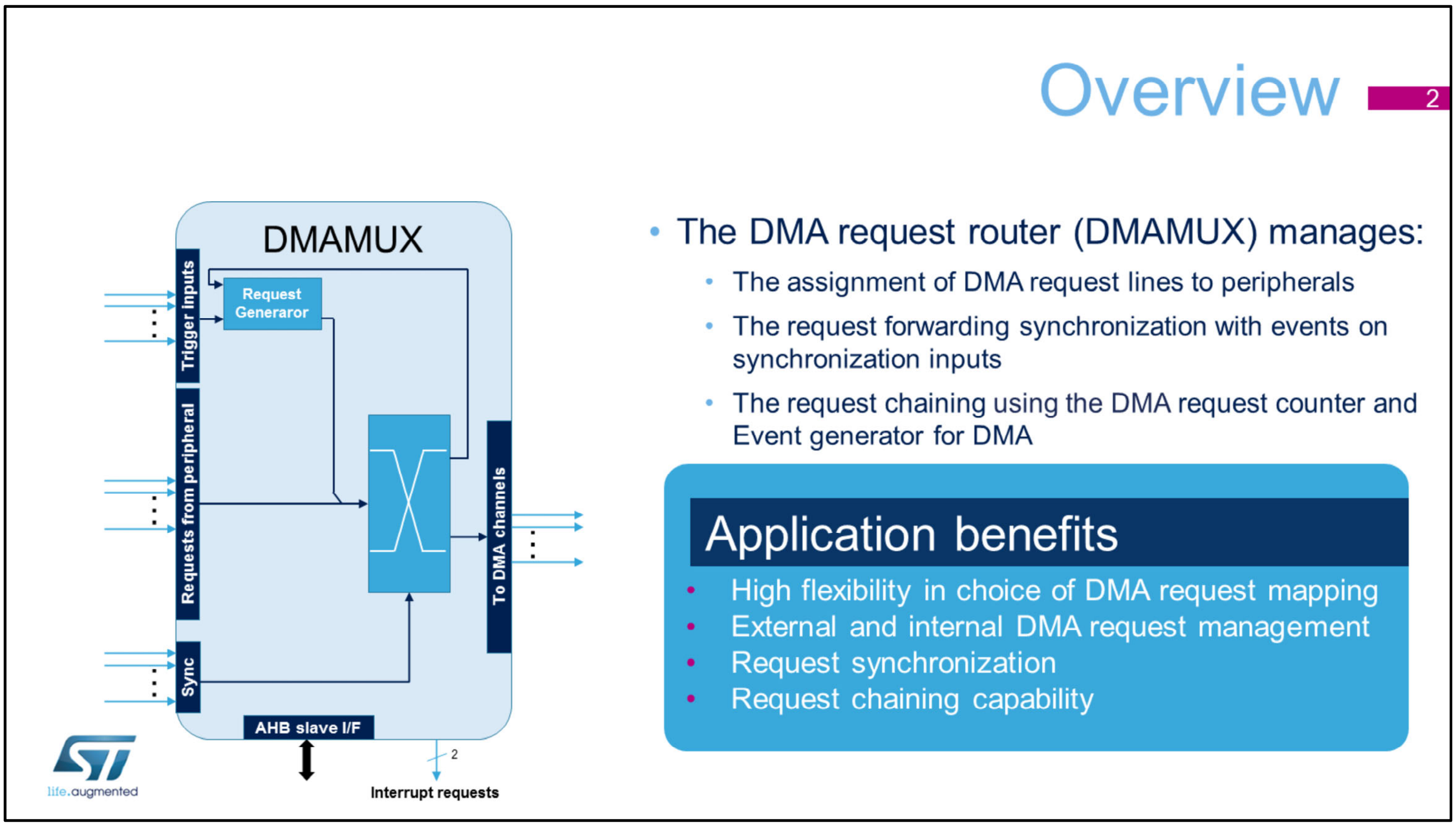
STM32G0 - DMAMUX

Direct Memory Access Multiplexer

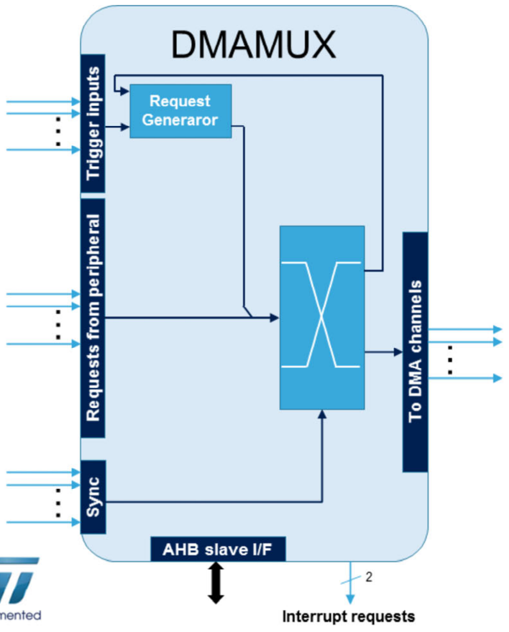
Revision 1.0

life.augmented

Welcome to the presentation of the STM32G0 DMA request multiplexer (DMAMUX). It covers the main features of this module.



Overview

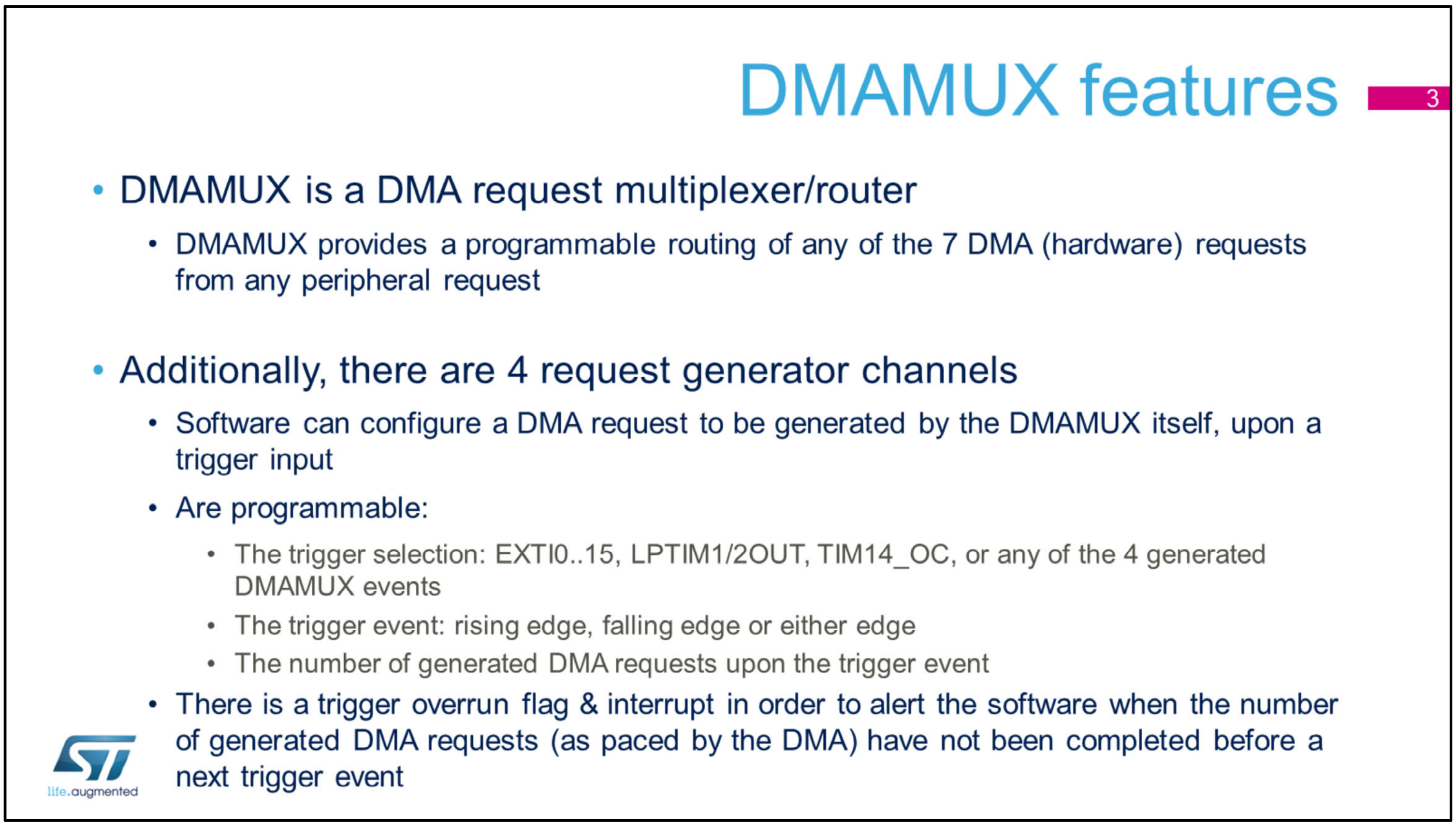


- The DMA request router (DMAMUX) manages:
  - The assignment of DMA request lines to peripherals
  - The request forwarding synchronization with events on synchronization inputs
  - The request chaining using the DMA request counter and Event generator for DMA

Application benefits

- High flexibility in choice of DMA request mapping
- External and internal DMA request management
- Request synchronization
- Request chaining capability

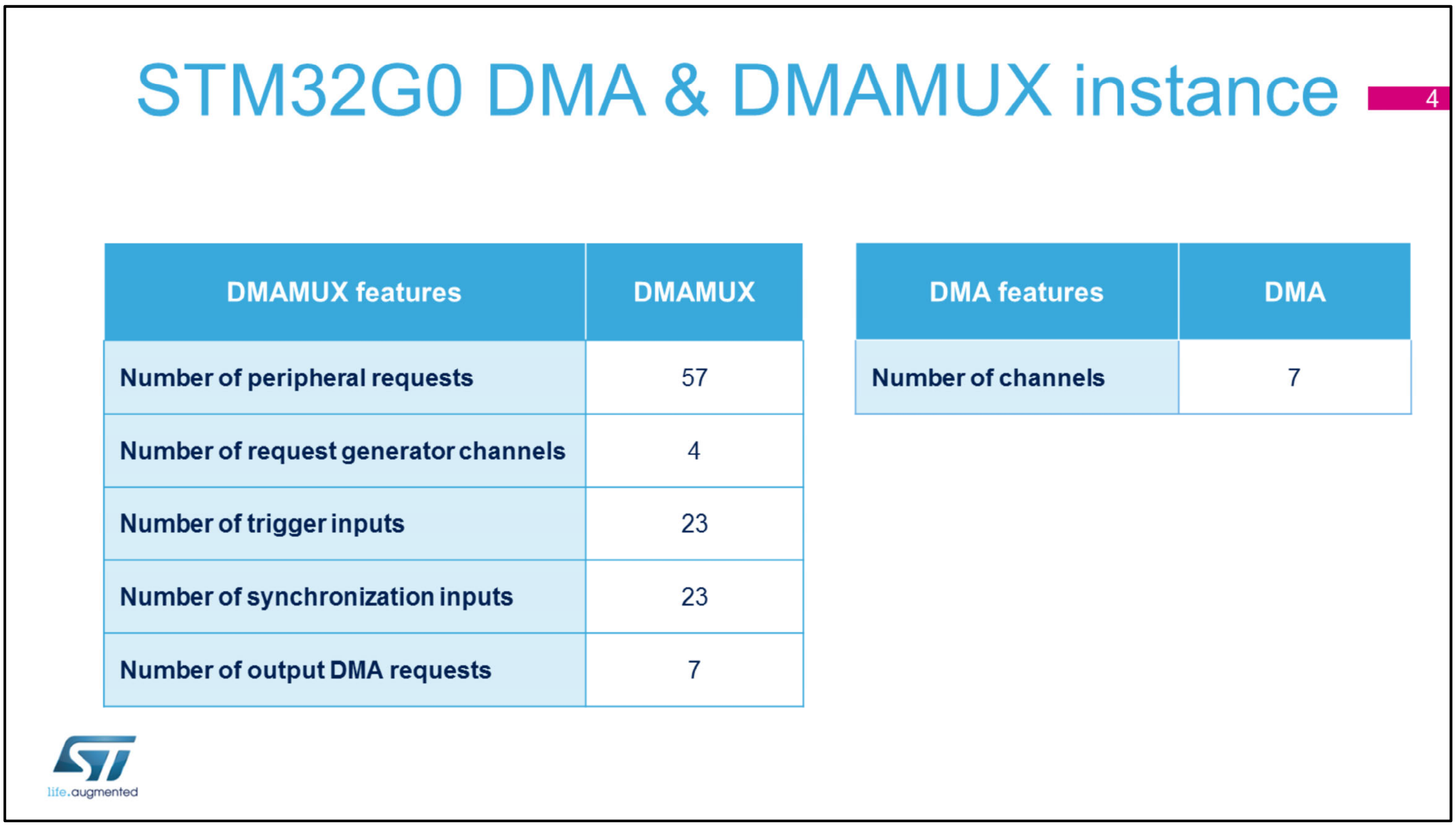
The DMAMUX request multiplexer allows routing a DMA request line between the STM32G0's peripherals and its DMA controllers. The routing function is ensured by a programmable multi-channel DMA request line multiplexer. Each channel selects a unique DMA request line, unconditionally or synchronously with events, from its DMAMUX synchronization inputs. The DMAMUX may also be used as a DMA request generator from programmable events on its input trigger signals.
Request chaining capability is based on an event generated on a particular output channel that is used as an input of the Request Generator to activate another channel.
The DMAMUX supports two interrupt request outputs.
DMAMUX registers are accessed through the AHB slave interface.


DMAMUX features

- DMAMUX is a DMA request multiplexer/router
  - DMAMUX provides a programmable routing of any of the 7 DMA (hardware) requests from any peripheral request
- Additionally, there are 4 request generator channels
  - Software can configure a DMA request to be generated by the DMAMUX itself, upon a trigger input
  - Are programmable:
    - The trigger selection: EXTI0..15, LPTIM1/2OUT, TIM14_OC, or any of the 4 generated DMAMUX events
    - The trigger event: rising edge, falling edge or either edge
    - The number of generated DMA requests upon the trigger event
  - There is a trigger overrun flag & interrupt in order to alert the software when the number of generated DMA requests (as paced by the DMA) have not been completed before a next trigger event

The DMAMUX is used to map the peripheral requests onto the 7 available DMA channels. This mapping is programmable.
Moreover the DMAMUX embeds a 4-channel request generator that convert triggers into DMA requests.
The following triggers are supported: the 16 external interrupts, low-power timers 1 and 2 timeouts, timer 14 output compare and 4 events generated by the DMAMUX itself.
DMAMUX events enable the user to chain DMA transfers without software intervention.
Each request generator has programmable registers to select:
- The trigger input,
- The trigger active edge,
- The number of the generated DMA request.

An overrun interrupt request is asserted when a new trigger is detected when the number of generated DMA requests caused by the previous trigger has not been completed.



STM32G0 DMA & DMAMUX instance

DMAMUX features	DMAMUX
Number of peripheral requests	57
Number of request generator channels	4
Number of trigger inputs	23
Number of synchronization inputs	23
Number of output DMA requests	7

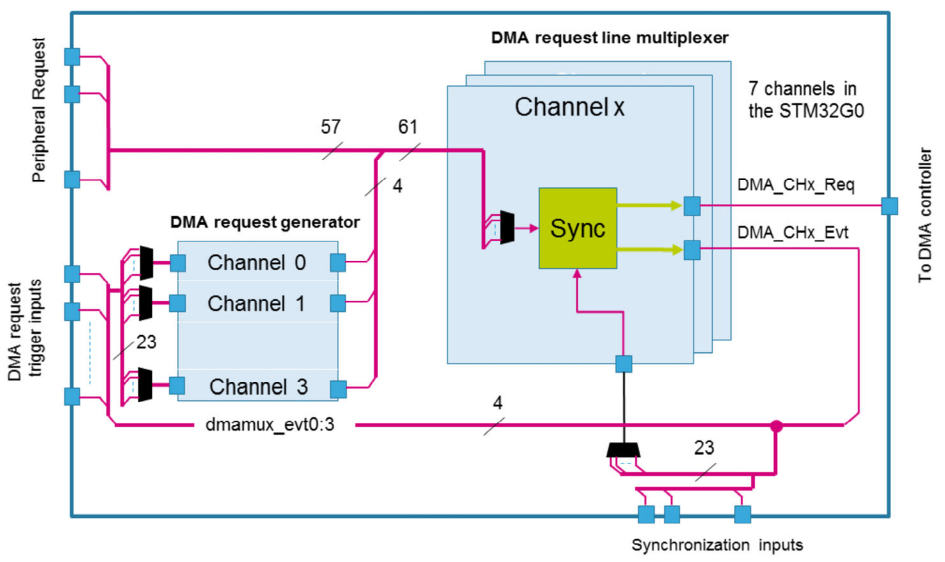
DMA features	DMA
Number of channels	7

life.augmented

The DMAMUX instantiated in the STM32G0 has the following features:

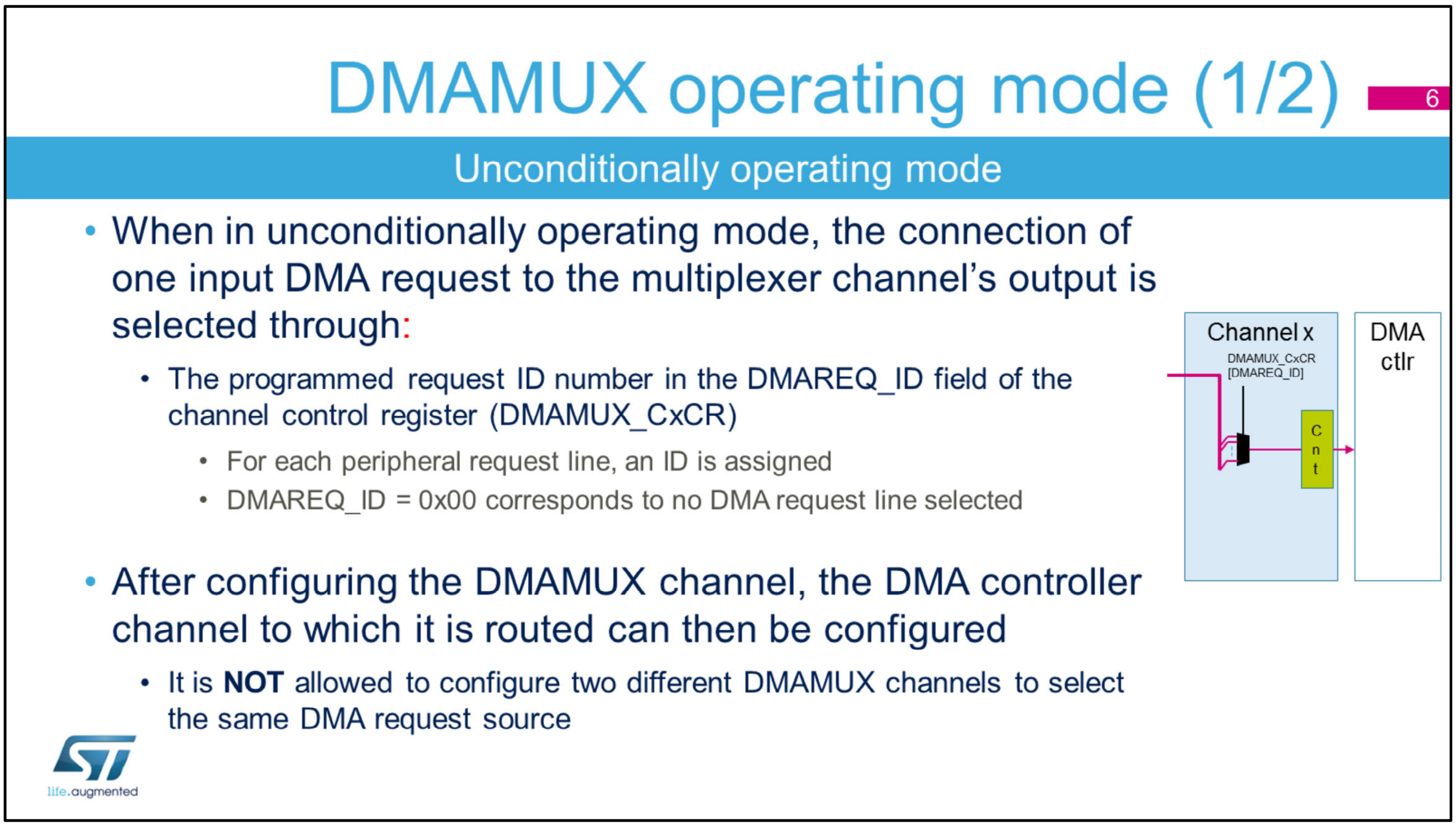
- 57 peripheral requests mapped to 7 DMA channels,
- 4 request generator channels,
- 23 trigger inputs,
- 23 synchronization inputs.

DMAMUX block diagram



The DMAMUX has two main sub-blocks: the request line multiplexer and the request line generator.
The DMAMUX request multiplexer enables routing a DMA request line between the STM32G0's peripherals and the DMA controller.
The routing function is ensured by a programmable multi-channel DMA request line multiplexer.
Each channel selects a unique DMA request line, unconditionally or synchronously with events, from its DMAMUX synchronization inputs.
The DMAMUX may also be used as a DMA request generator from programmable events on its input trigger signals.
The DMA request line multiplexor generates both a request to the DMA controller and also events that can be used as synchronization inputs as well as trigger inputs.

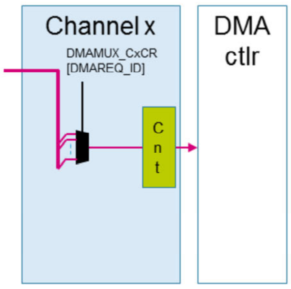
Do not confuse DMA request generator channels (0 to 3) with DMA request line multiplexer channels (1 to 7).



DMAMUX operating mode (1/2)

Unconditionally operating mode

- When in unconditionally operating mode, the connection of one input DMA request to the multiplexer channel's output is selected through:
  - The programmed request ID number in the DMAREQ_ID field of the channel control register (DMAMUX_CxCR)
    - For each peripheral request line, an ID is assigned
    - DMAREQ_ID = 0x00 corresponds to no DMA request line selected
- After configuring the DMAMUX channel, the DMA controller channel to which it is routed can then be configured
  - It is **NOT** allowed to configure two different DMAMUX channels to select the same DMA request source



The DMAMUX request multiplexer enables routing a DMA request line between a peripherals and a DMA channel in unconditionally operating mode.
When the multiplexer is set, it ensures the actual routing of DMA request/acknowledge control signals.

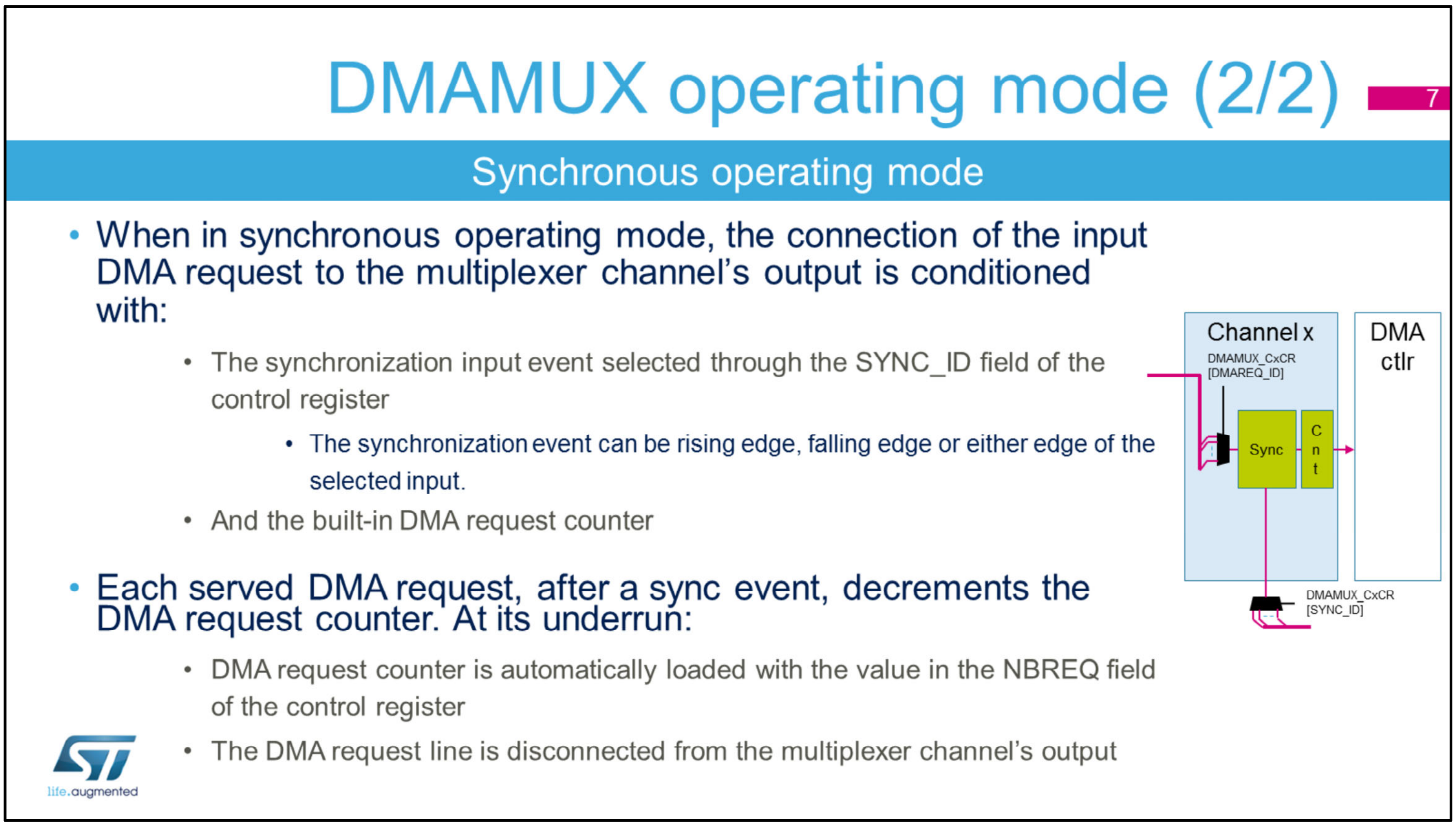
The connection of a peripheral request to the multiplexer channel's output is selected through the programmed request ID in the DMAREQ_ID field of the channel control register (DMAMUX_CxCR)

- For each peripheral request line, an ID is assigned.
- DMAREQ_ID = 0x00 corresponds to no DMA request line selected.

After configuring the DMAMUX channel, the DMA controller channel to which it is routed can then be configured.

It is not allowed to configure two different DMAMUX channels to select the same DMA request source.

The DMAMUX event output is generated when the DMA request counter reaches the value 0. Its operation will be explained in the next slides.

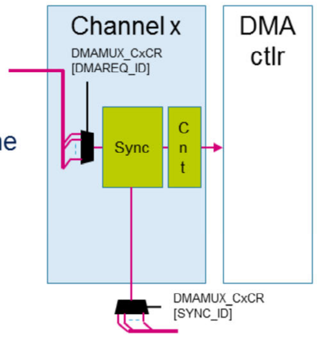


DMAMUX operating mode (2/2)

7

Synchronous operating mode

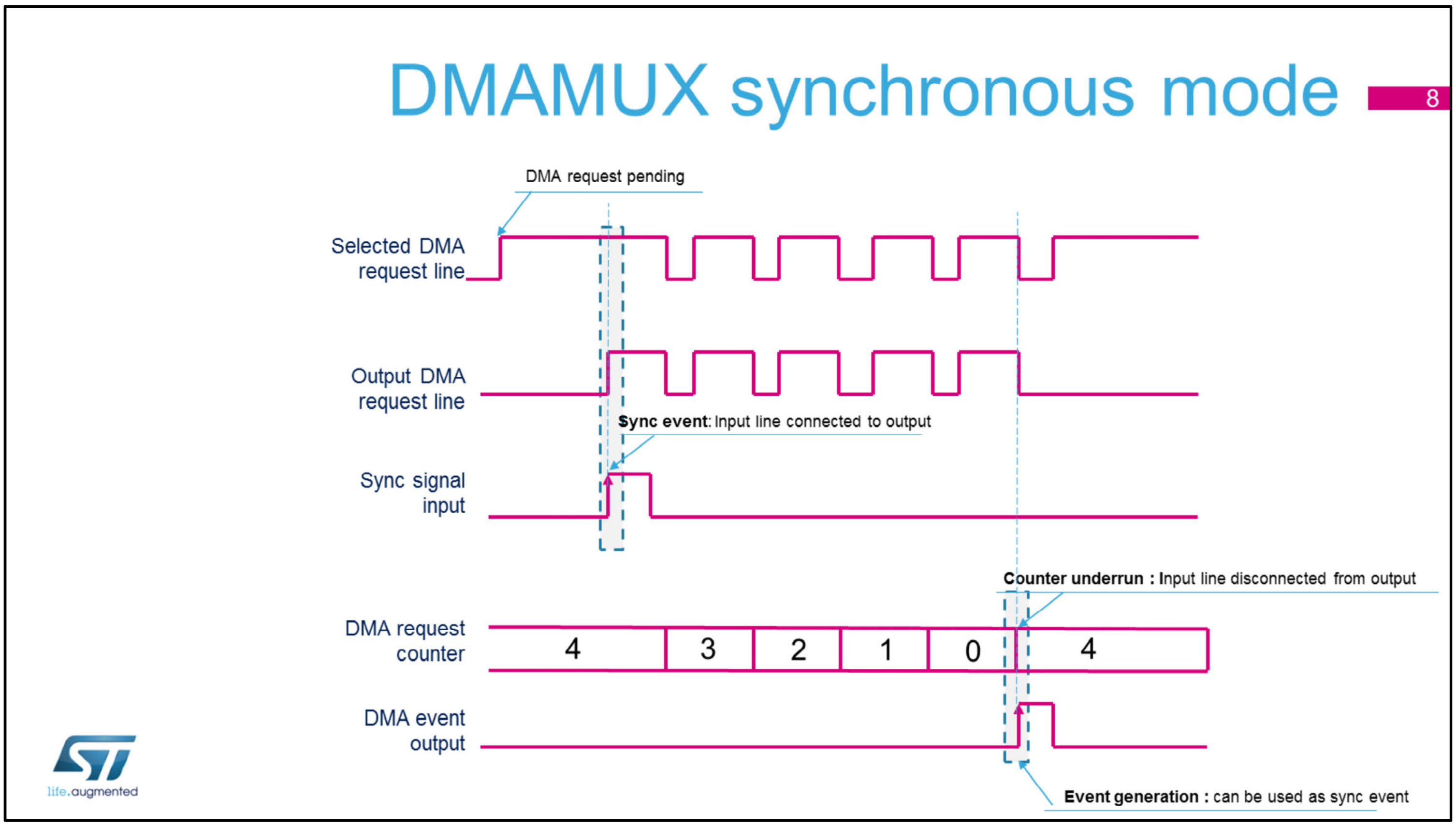
- When in synchronous operating mode, the connection of the input DMA request to the multiplexer channel's output is conditioned with:
  - The synchronization input event selected through the SYNC_ID field of the control register
    - The synchronization event can be rising edge, falling edge or either edge of the selected input.
  - And the built-in DMA request counter
- Each served DMA request, after a sync event, decrements the DMA request counter. At its underrun:
  - DMA request counter is automatically loaded with the value in the NBREQ field of the control register
  - The DMA request line is disconnected from the multiplexer channel's output



life.augmented

Each DMA request line multiplexer can individually be set to synchronous operating mode by setting the synchronization enable (SE) bit in its corresponding multiplexer channel control register (DMAMUX_CxCR).
The DMA request router has multiple synchronization inputs. The synchronization inputs are connected in parallel to all multiplexer channels.
When a multiplexer channel is in synchronous operating mode, the effective connection of the selected input DMA request line to the multiplexer channel's output is conditioned with events on the selected synchronization input and on a built-in DMA request counter.
Upon the synchronization event, the selected DMA request line is connected to the multiplexer channel's output. From this point on, each served DMA request (transition 1-to-0) on the selected DMA request line decrements the DMA request counter.

At its underrun, the DMA request counter is automatically loaded with the value in the NBREQ field of the control register and the DMA request line is disconnected from the multiplexer channel’s output. Thus, the number of DMA requests transferred to the multiplexer channel’s output following a synchronization event is the value in the NBREQ field plus one.

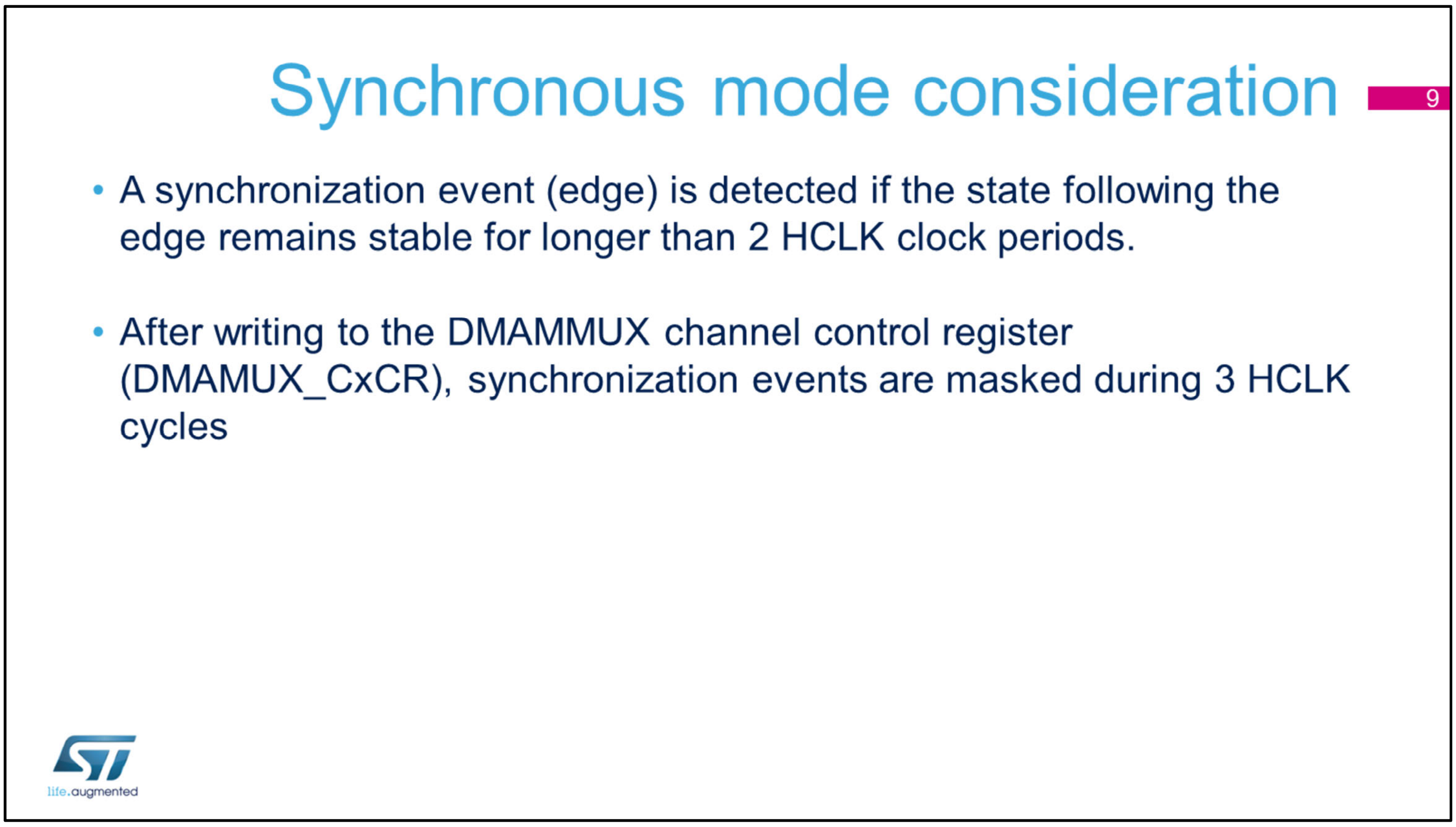


DMAMUX synchronous mode

DMA request pending
Selected DMA request line
Output DMA request line
Sync event: Input line connected to output
Sync signal input
Counter underrun : Input line disconnected from output
DMA request counter
4
3
2
1
0
4
DMA event output
Event generation : can be used as sync event
life.augmented

When the DMAMUX channel is configured in synchronous mode, its behavior is as follows.
The request multiplexer input (DMA request from the peripheral) can become active, but it will not be forwarded on the DMAMUX request multiplexer output until the synchronization signal is received.
When the sync event is received, the request multiplexer connects its input and output and all the peripheral requests will be forwarded.
Each DMA request forwarded will decrement the request multiplexer counter (user programmed value). When the counter reaches zero, the connection between the DMA controller and the peripheral is cut, waiting for a new synchronization event.
For each underrun of the counter, a request multiplexer line can generate an optional event to synchronize with a second DMAMUX line. The same event can be used in

some low-power scenarios to switch the system back to Stop mode without CPU intervention.
Synchronization mode can be used to automatically synchronize data transfers with a timer for example, or to trigger the transfers on a peripheral event.

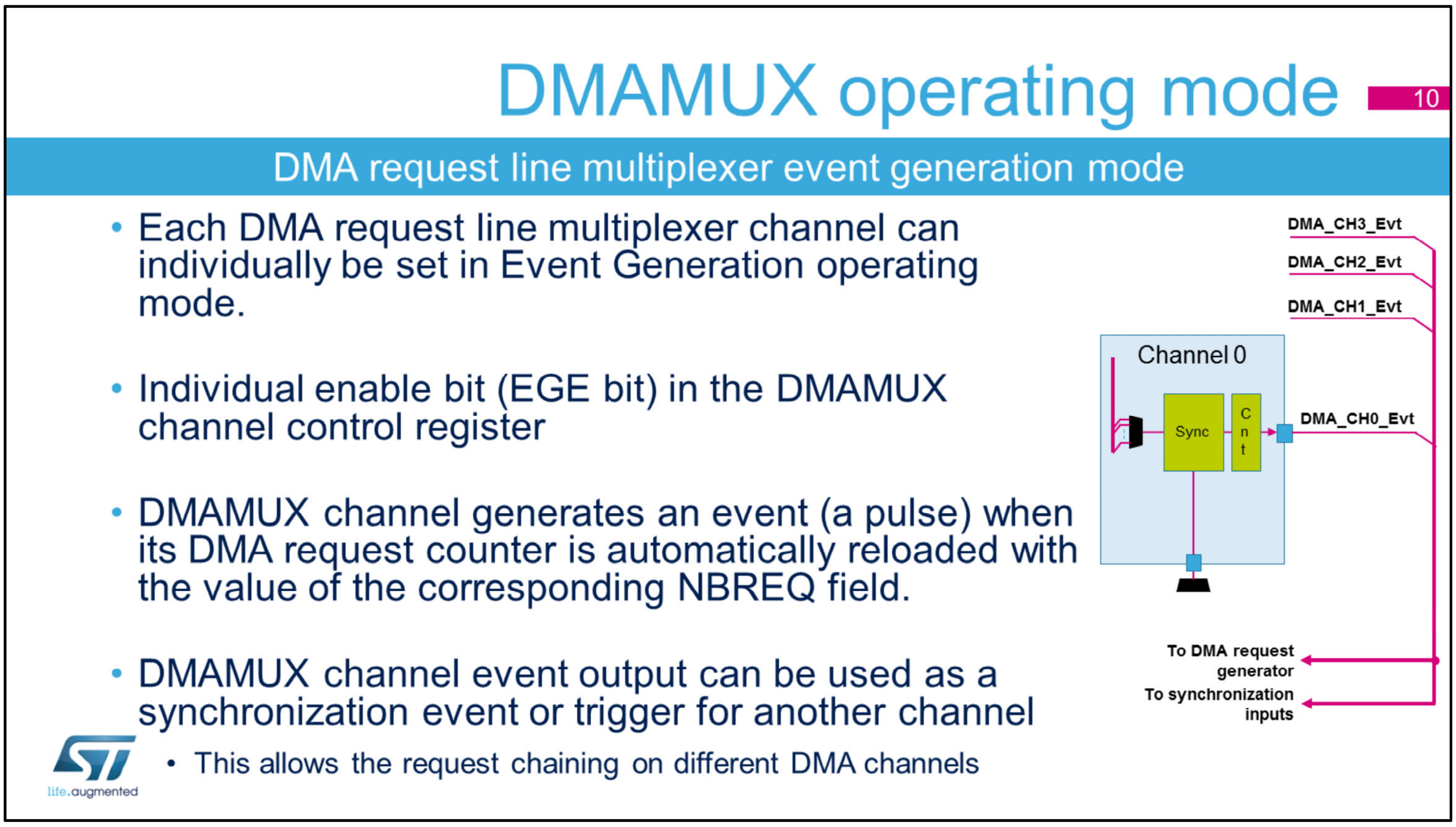


Synchronous mode consideration

- A synchronization event (edge) is detected if the state following the edge remains stable for longer than 2 HCLK clock periods.
- After writing to the DMAMMUX channel control register (DMAMUX_CxCR), synchronization events are masked during 3 HCLK cycles

life.augmented

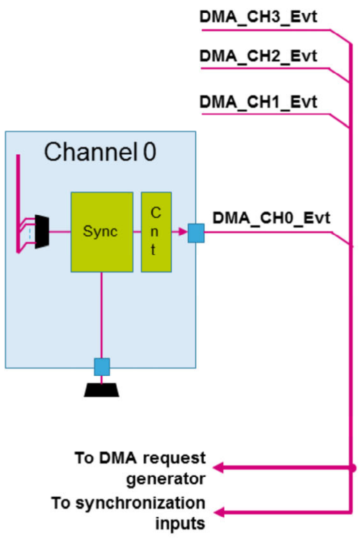
A synchronization event (edge) is detected if the state following the edge remains stable for longer than two AHB clock periods.
This delay ensures that glitches on the synchronization event are not taken into account.
After writing to the DMAMUX_CxCR control register, synchronization events are masked during 3 HCLK cycles.
This delay masks possible synchronization events that could occur while the control register is updated, causing metastability.
The synchronization event overrun condition occurs when a new synchronization event is received while the request multiplexer counter is different than zero.



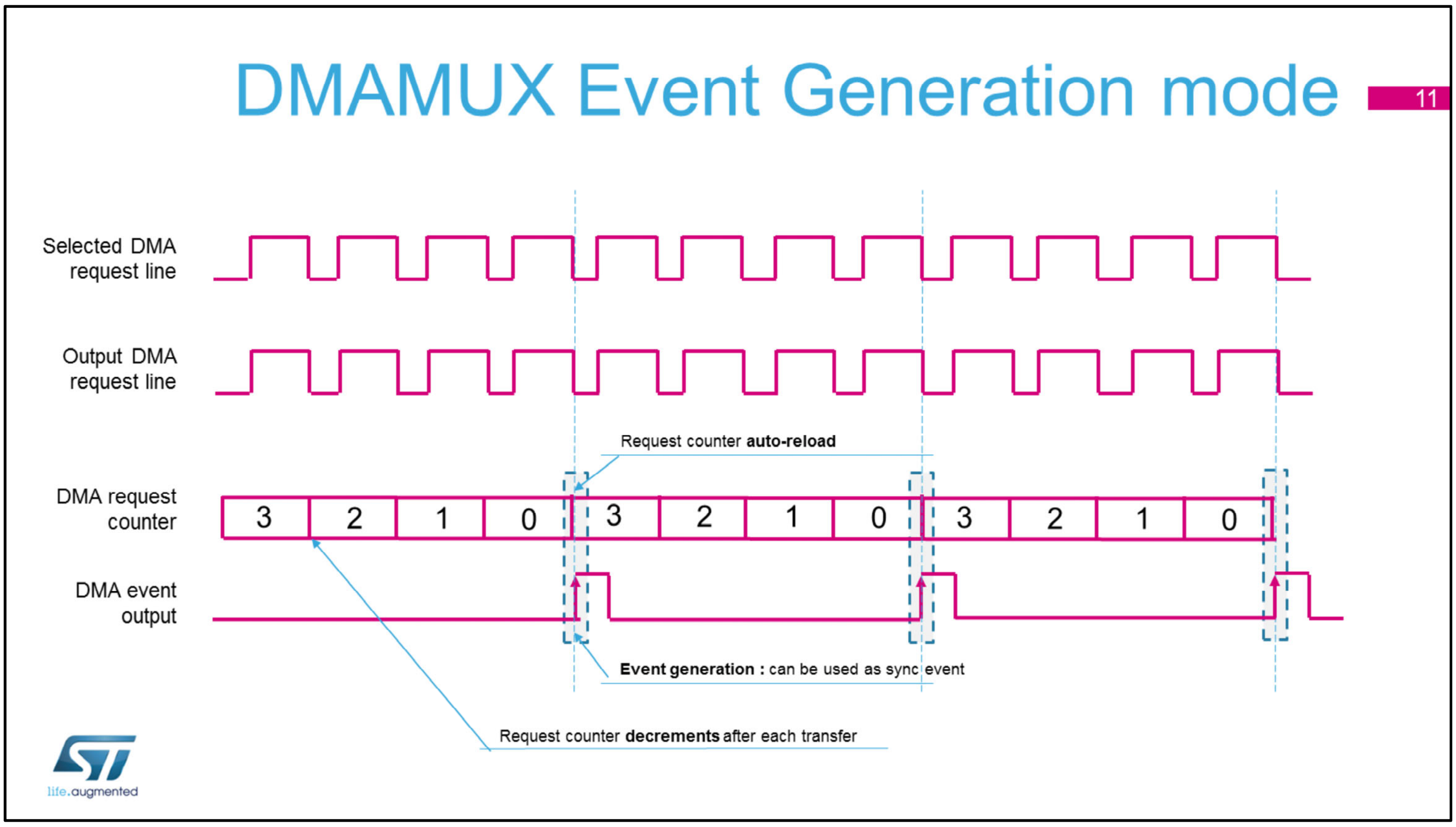
DMAMUX operating mode

DMA request line multiplexer event generation mode

- Each DMA request line multiplexer channel can individually be set in Event Generation operating mode.
- Individual enable bit (EGE bit) in the DMAMUX channel control register
- DMAMUX channel generates an event (a pulse) when its DMA request counter is automatically reloaded with the value of the corresponding NBREQ field.
- DMAMUX channel event output can be used as a synchronization event or trigger for another channel
  - This allows the request chaining on different DMA channels



When enabled, the multiplexer channel generates an event (a pulse) when its DMA request counter is automatically reloaded with the value of the corresponding NBREQ field.
The event generator is enabled by setting the EGE bit in the control register of the corresponding multiplexer channel.
Only four channels support the generation of events: channels 0 to 3.



DMAMUX Event Generation mode

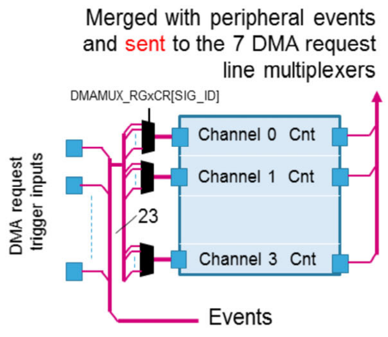
Selected DMA request line
Output DMA request line
DMA request counter
3
2
1
0
3
2
1
0
3
2
1
0
DMA event output
Request counter auto-reload
Event generation : can be used as sync event
Request counter decrements after each transfer

When the DMAMUX channel is in Event Generation mode, it generates an event (a pulse) when its DMA request counter is automatically reloaded. The request counter is decremented with the execution of a DMA request.
The DMAMUX channel event output can be used as a synchronization event or trigger for another channel.

DMAMUX operating mode

DMA request generator operating mode

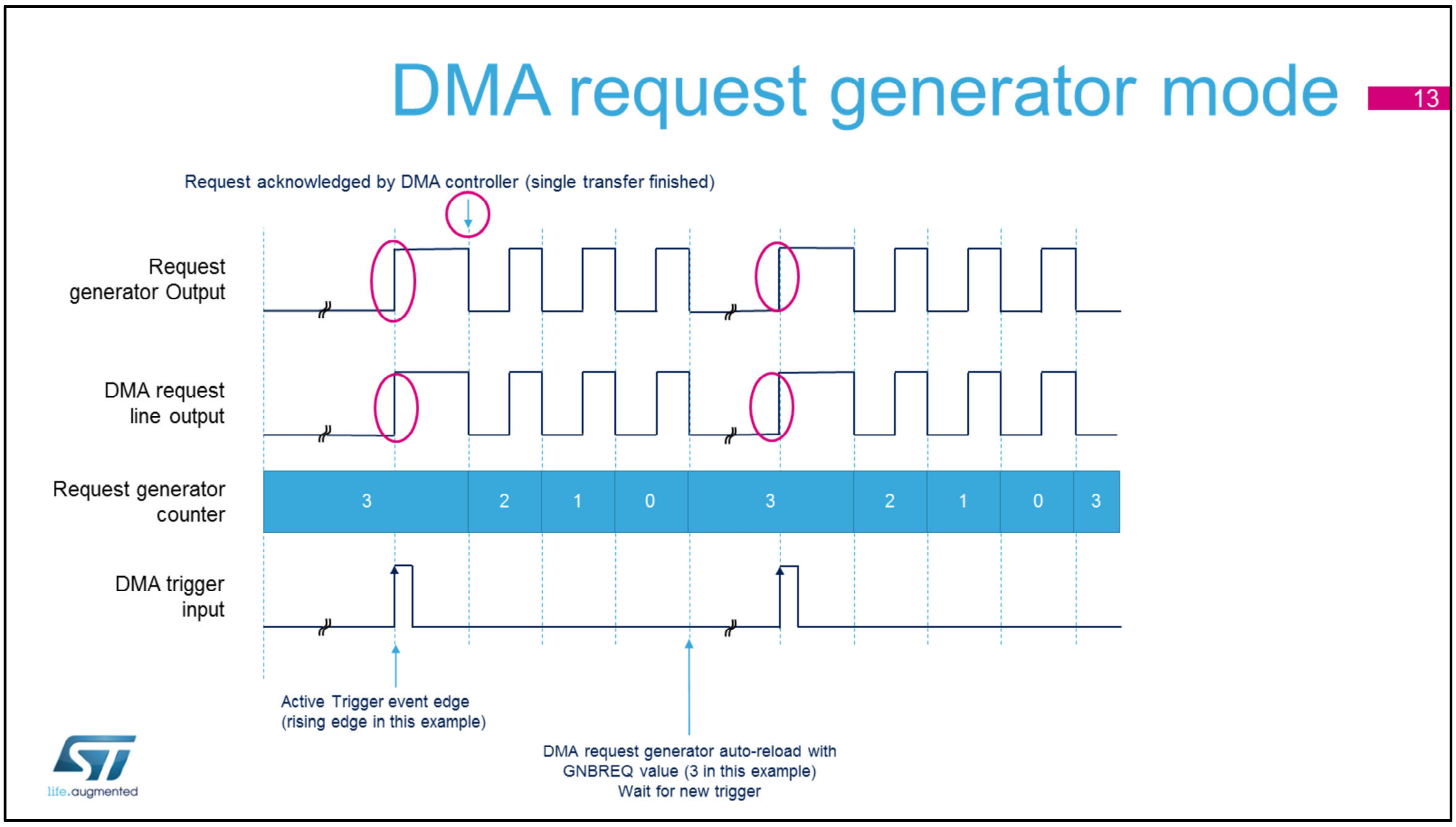
- When a request generator channel is enabled, it allows to produce DMA requests following trigger events
- The outputs of DMA generator channels go to inputs of the DMA request line multiplexer
- Each generator channel has its individual configuration register:
  - SIG_ID field corresponds to the request trigger input for generator.
  - GNBREQ field corresponds to the number of DMA requests **minus 1** to generate after a trigger event.
  - GPOL field corresponds to the active edge of the trigger input. Trigger events can be rising edge, falling edge or either edge on the trigger input



Merged with peripheral events and sent to the 7 DMA request line multiplexers
DMAMUX_RGxCR[SIG_ID]
DMA request trigger inputs
Channel 0 Cnt
Channel 1 Cnt
23
Channel 3 Cnt
Events

life.augmented

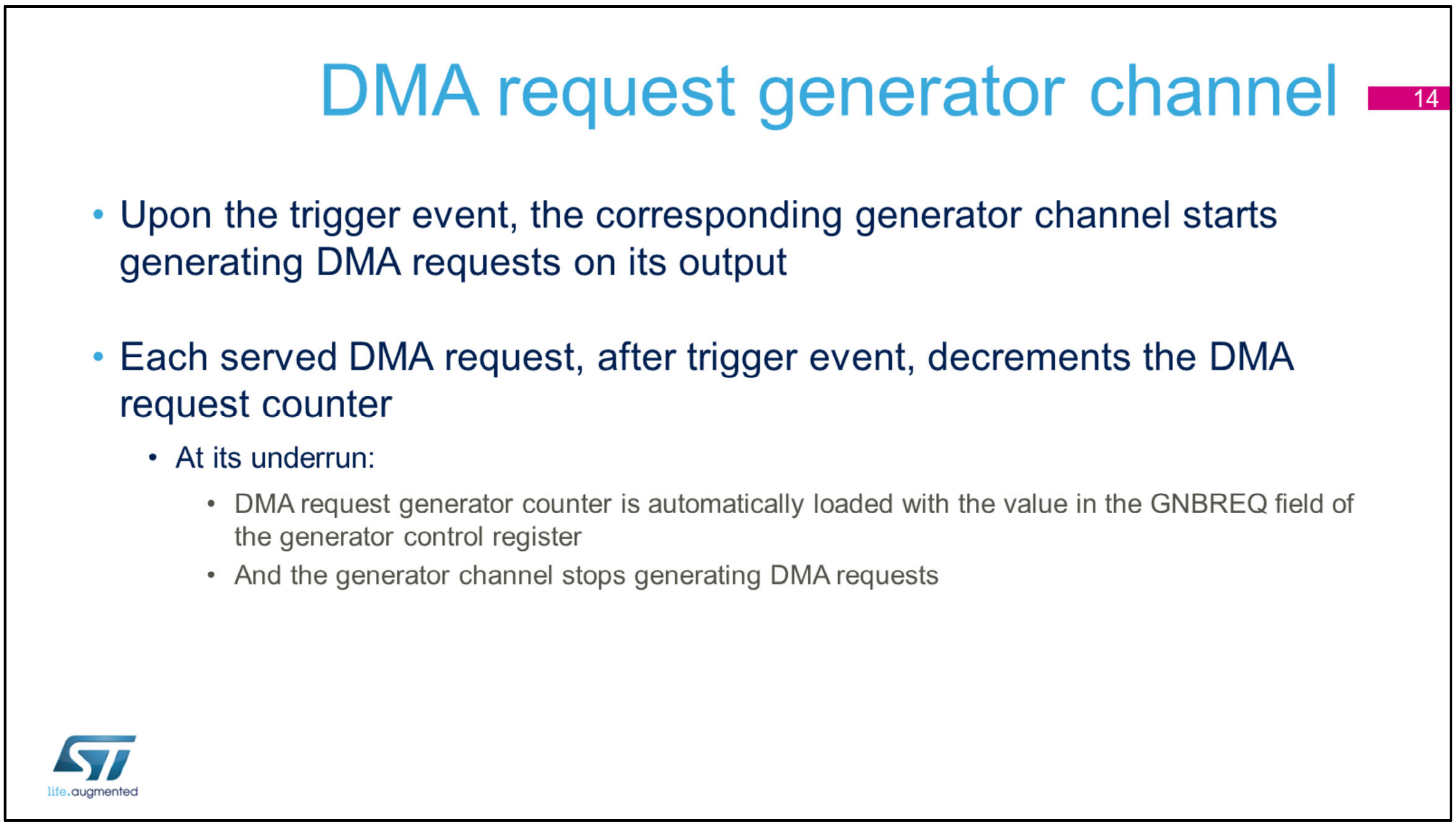
On its output, the DMA request generator produces DMA requests following trigger events on DMA request trigger inputs.
The DMA request generator has multiple 4 channels. DMA request trigger inputs are connected in parallel to the 4 channels.
The outputs of DMA generator channels go to inputs of the DMA request line multiplexer.
Each DMA request generator channel (named “generator channel” further in this section) has an enable bit.
The DMA request trigger input for generator channel x is selected through the SIG_ID field of the corresponding generator channel’s control register.
Trigger events on a DMA request trigger input can be rising edge, falling edge or either edge. The active edge is selected through the POL field of the corresponding generator channel’s control register.



DMA request generator mode

Request acknowledged by DMA controller (single transfer finished)
Request generator Output
DMA request line output
Request generator counter
3
2
1
0
3
2
1
0
3
DMA trigger input
Active Trigger event edge (rising edge in this example)
DMA request generator auto-reload with GNBREQ value (3 in this example) Wait for new trigger
life.augmented

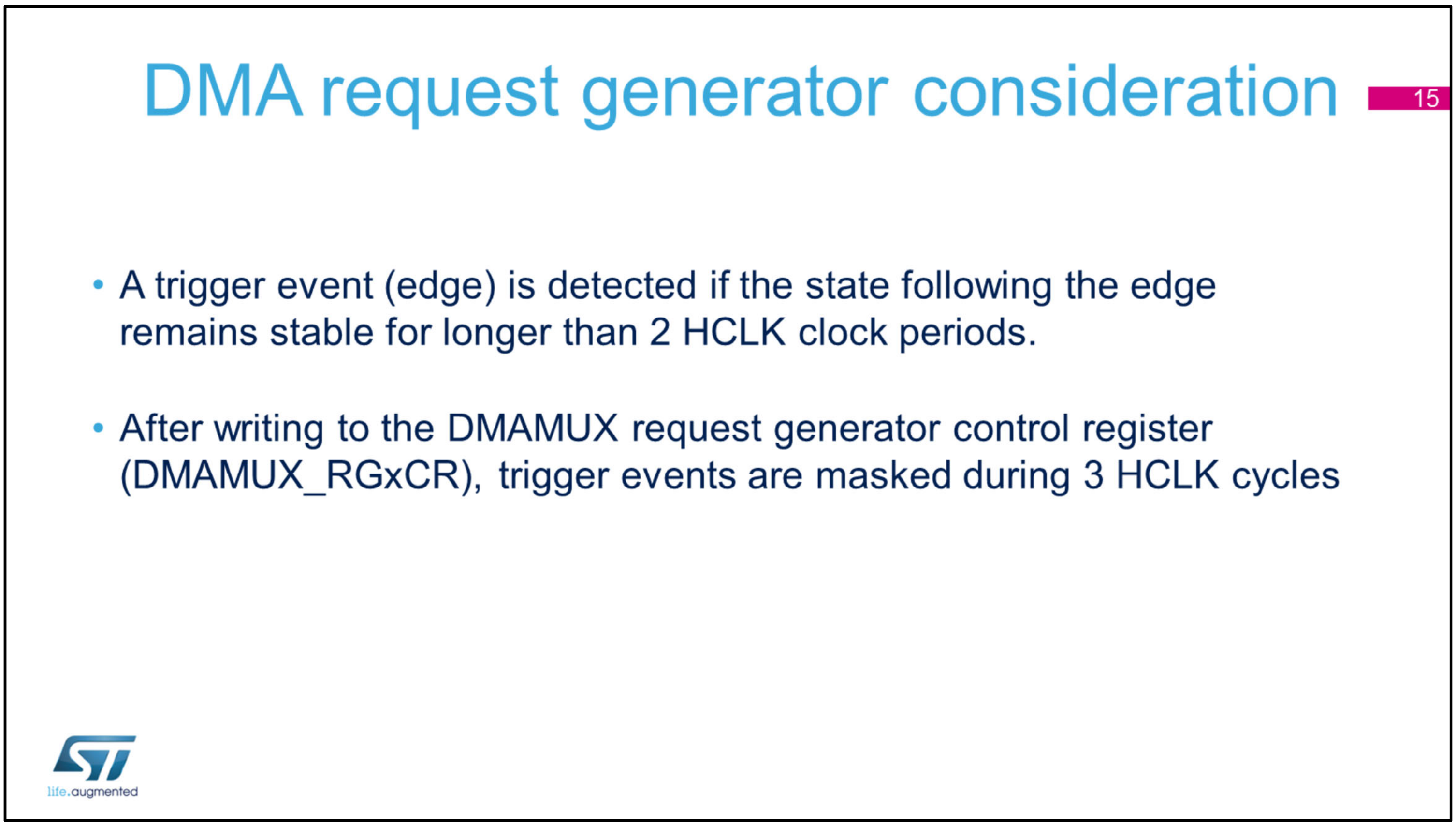
This slide shows how the DMA request generator can be used to generate a series of DMA requests from a single DMA trigger input edge detection.
Upon the trigger event, the corresponding generator channel starts generating DMA requests on its output.
Each served DMA request (i.e. when the request signal is de-asserted) decrements a built-in DMA request counter, internally to the DMAMUX request generator. At its underrun, the DMA request counter is automatically loaded with the value in GNBREQ field of the corresponding DMAMUX_RGxCR register and the request generator channel stops generating DMA requests.
Thus, the number of DMA requests generated after the trigger event is the value in the GNBREQ field plus one.



DMA request generator channel

- Upon the trigger event, the corresponding generator channel starts generating DMA requests on its output
- Each served DMA request, after trigger event, decrements the DMA request counter
  - At its underrun:
    - DMA request generator counter is automatically loaded with the value in the GNBREQ field of the generator control register
    - And the generator channel stops generating DMA requests

Upon the trigger event, the corresponding generator channel starts generating DMA requests on its output. Each served DMA request (transition 1-to-0) decrements a built-in DMA request counter. At its underrun, the DMA request counter is automatically loaded with the value in the GNBREQ field of the corresponding generator channel's control register and the generator channel stops generating DMA requests. Thus, the number of DMA requests generated after the trigger event is the value in the GNBREQ field plus one.

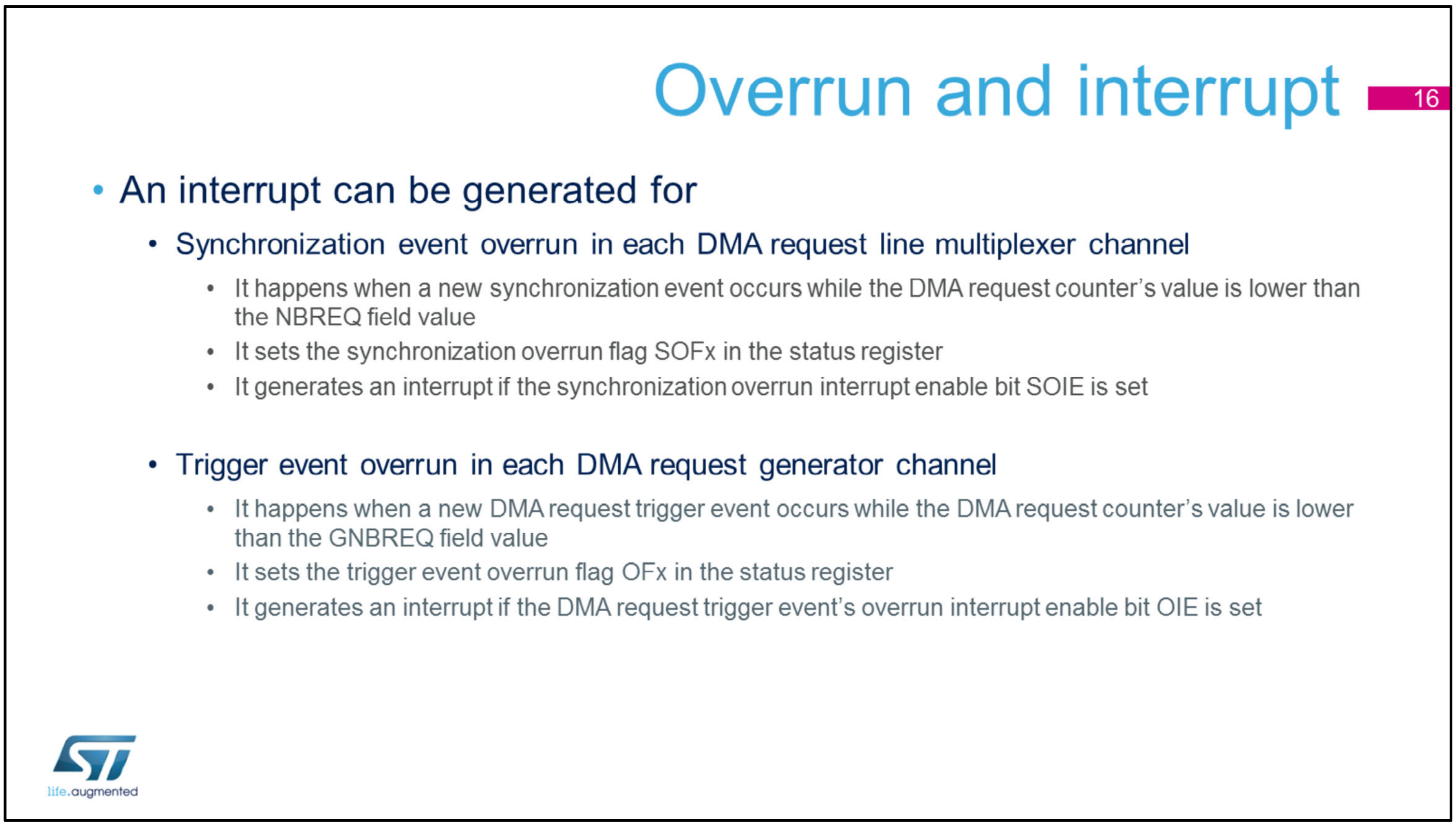


DMA request generator consideration

- A trigger event (edge) is detected if the state following the edge remains stable for longer than 2 HCLK clock periods.
- After writing to the DMAMUX request generator control register (DMAMUX_RGxCR), trigger events are masked during 3 HCLK cycles

life.augmented

A trigger event (edge) is detected if the state following the edge remains stable for longer than two AHB clock periods.
This delay ensures that glitches on the trigger input are not taken into account.
After writing to the DMAMUX_RGxCR control register, trigger events are masked during 3 HCLK cycles.
This delay masks possible trigger events that could occur while the control register is updated, causing metastability.



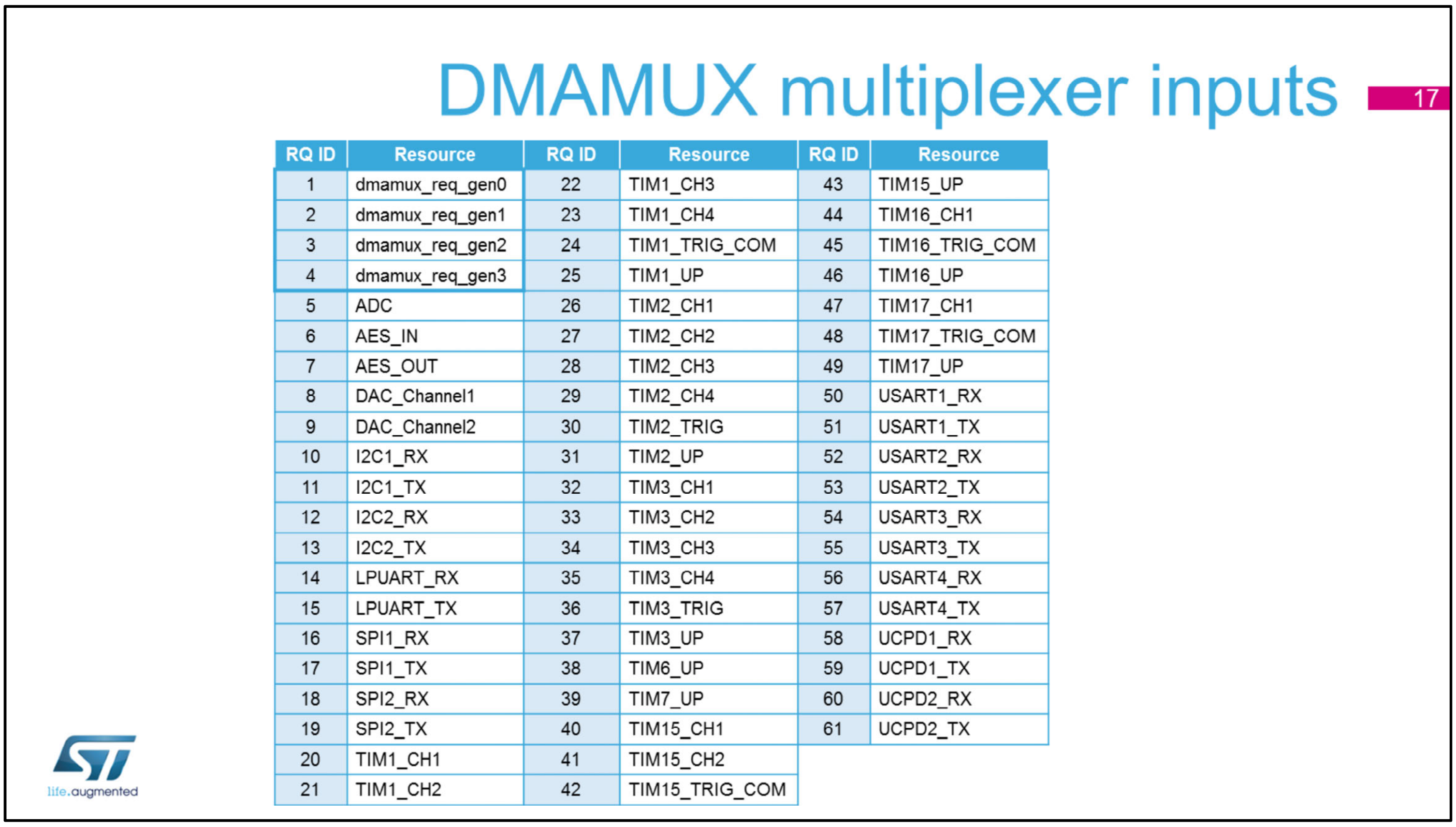
Overrun and interrupt

- An interrupt can be generated for
  - Synchronization event overrun in each DMA request line multiplexer channel
    - It happens when a new synchronization event occurs while the DMA request counter’s value is lower than the NBREQ field value
    - It sets the synchronization overrun flag SOFx in the status register
    - It generates an interrupt if the synchronization overrun interrupt enable bit SOIE is set
  - Trigger event overrun in each DMA request generator channel
    - It happens when a new DMA request trigger event occurs while the DMA request counter’s value is lower than the GNBREQ field value
    - It sets the trigger event overrun flag OFx in the status register
    - It generates an interrupt if the DMA request trigger event’s overrun interrupt enable bit OIE is set

If a new synchronization event occurs while the DMA request counter’s value is lower than the NBREQ field value, the synchronization event overrun flag SOFx is set in the status register DMAMUX_CSR.
This flag is reset by setting the associated clear bit CSOFx, in the DMAMUX_CFR register.
Setting the synchronization overrun flag generates an interrupt if the synchronization overrun interrupt enable bit SOIE is set in the configuration register of the corresponding multiplexer channel.

If a new DMA request trigger event occurs while the DMA request counter’s value is lower than the GNBREQ field value, the trigger event overrun flag OFx is set in the status register DMAMUX_RGSR.
The overrun flag OFx is reset by setting the associated clear bit COFx, in the DMAMUX_RGCFR register.

Setting the DMA request trigger overrun flag generates an interrupt if the DMA request trigger event’s overrun interrupt enable bit OIE is set in the control register of the corresponding generator channel.

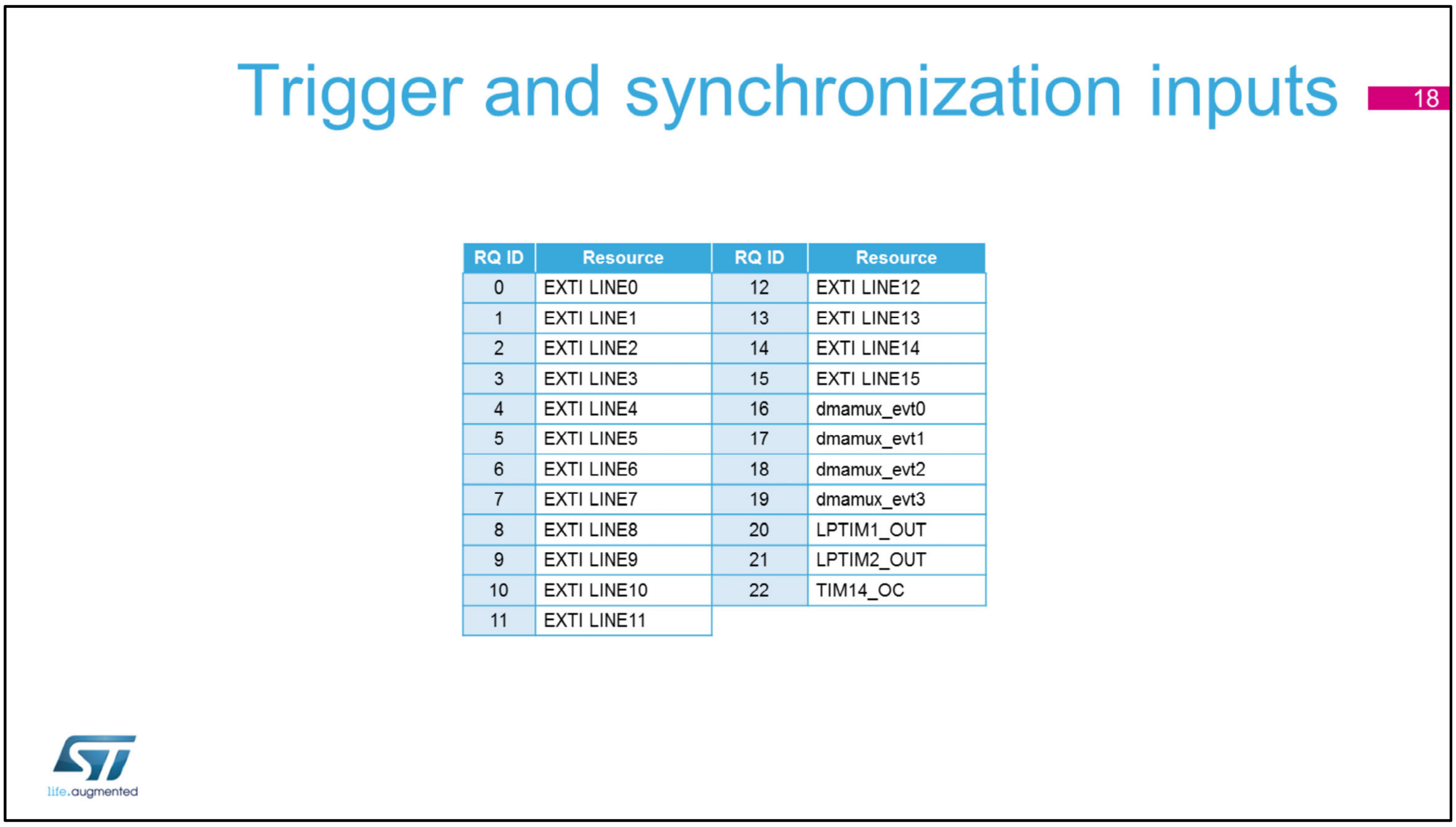


DMAMUX multiplexer inputs

RQ ID	Resource	RQ ID	Resource	RQ ID	Resource
1	dmamux_req_gen0	22	TIM1_CH3	43	TIM15_UP
2	dmamux_req_gen1	23	TIM1_CH4	44	TIM16_CH1
3	dmamux_req_gen2	24	TIM1_TRIG_COM	45	TIM16_TRIG_COM
4	dmamux_req_gen3	25	TIM1_UP	46	TIM16_UP
5	ADC	26	TIM2_CH1	47	TIM17_CH1
6	AES_IN	27	TIM2_CH2	48	TIM17_TRIG_COM
7	AES_OUT	28	TIM2_CH3	49	TIM17_UP
8	DAC_Channel1	29	TIM2_CH4	50	USART1_RX
9	DAC_Channel2	30	TIM2_TRIG	51	USART1_TX
10	I2C1_RX	31	TIM2_UP	52	USART2_RX
11	I2C1_TX	32	TIM3_CH1	53	USART2_TX
12	I2C2_RX	33	TIM3_CH2	54	USART3_RX
13	I2C2_TX	34	TIM3_CH3	55	USART3_TX
14	LPUART_RX	35	TIM3_CH4	56	USART4_RX
15	LPUART_TX	36	TIM3_TRIG	57	USART4_TX
16	SPI1_RX	37	TIM3_UP	58	UCPD1_RX
17	SPI1_TX	38	TIM6_UP	59	UCPD1_TX
18	SPI2_RX	39	TIM7_UP	60	UCPD2_RX
19	SPI2_TX	40	TIM15_CH1	61	UCPD2_TX
20	TIM1_CH1	41	TIM15_CH2		
21	TIM1_CH2	42	TIM15_TRIG_COM		

life.augmented

This table shows the list of the request inputs of the DMAMUX unit. Note that the actual number of request inputs is 57 + 4, since the requests numbered from 1 to 4 are the outputs of the 4 request generator channels.

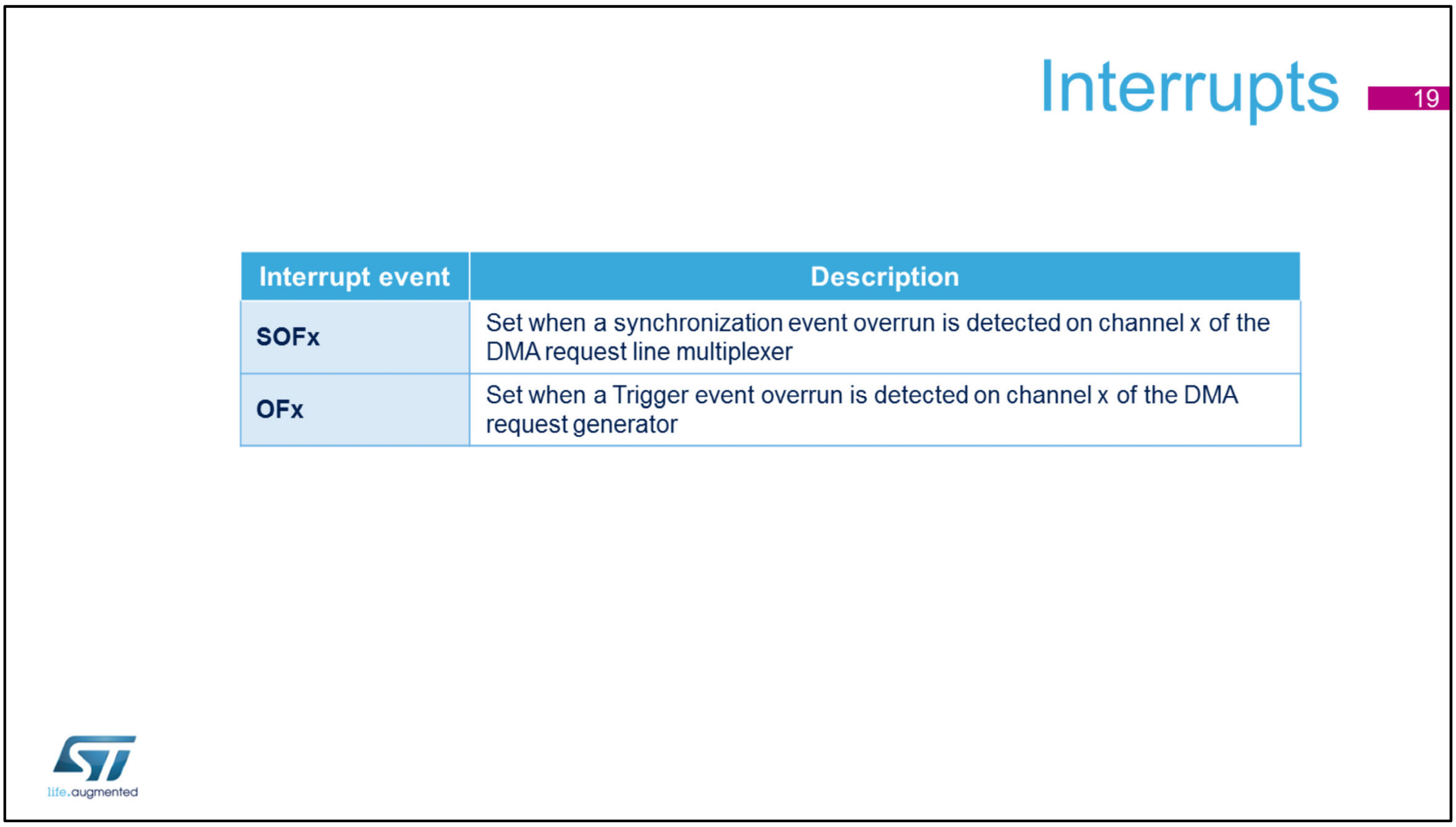


Trigger and synchronization inputs

RQ ID	Resource	RQ ID	Resource
0	EXTI LINE0	12	EXTI LINE12
1	EXTI LINE1	13	EXTI LINE13
2	EXTI LINE2	14	EXTI LINE14
3	EXTI LINE3	15	EXTI LINE15
4	EXTI LINE4	16	dmamux_evt0
5	EXTI LINE5	17	dmamux_evt1
6	EXTI LINE6	18	dmamux_evt2
7	EXTI LINE7	19	dmamux_evt3
8	EXTI LINE8	20	LPTIM1_OUT
9	EXTI LINE9	21	LPTIM2_OUT
10	EXTI LINE10	22	TIM14_OC
11	EXTI LINE11		

life.augmented

The trigger inputs and synchronization inputs are the same and have the same ID in the DMAMUX instantiated in the STM32G0.



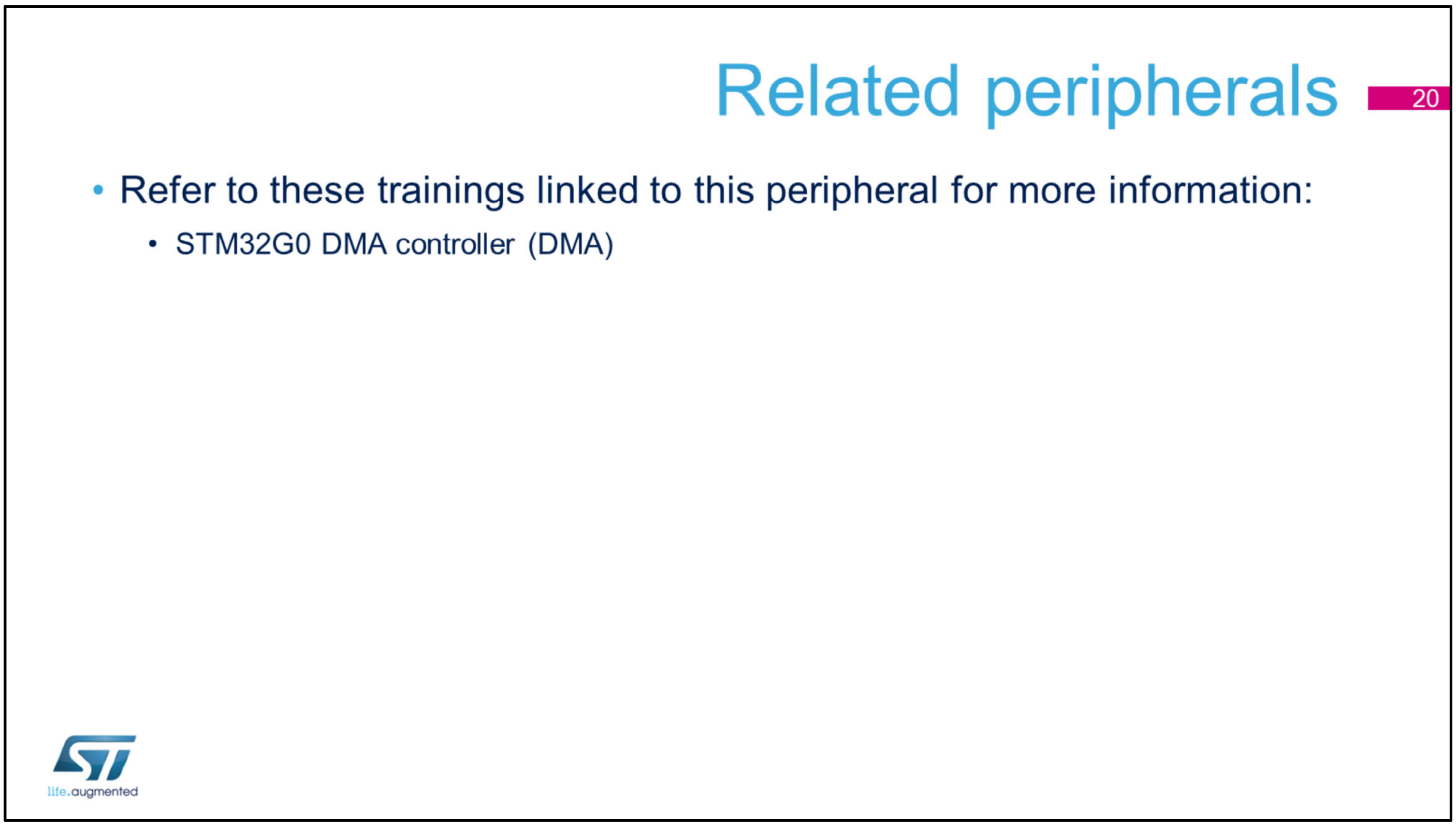
Interrupts

Interrupt event	Description
SOFx	Set when a synchronization event overrun is detected on channel x of the DMA request line multiplexer
OFx	Set when a Trigger event overrun is detected on channel x of the DMA request generator

An interrupt can be generated for:

- A synchronization event overrun in each DMA request line multiplexer channel
- A trigger event overrun in each DMA request generator channel

In both cases, per-channel individual interrupt enable bits are available.



Related peripherals

- Refer to these trainings linked to this peripheral for more information:
  - STM32G0 DMA controller (DMA)

Please refer to the training linked to this peripheral for more information:

- STM32G0 DMA controller (DMA)