Hello, and welcome to this presentation of the STM32 nested vectored interrupt controller (NVIC). We will be presenting the features of this controller.
The interrupt controller belongs to the Cortex®-M0+ CPU enabling a close coupling with the processor core. The main features are:

- 32 maskable interrupt channels
- 4 programmable priority levels
- Low-latency exception and interrupt handling
- Power management control

Applications can benefit from dynamic prioritization of the interrupt levels, fast response to the requests thanks to low latency responses and tail chaining as well as from vector table relocation.
The NVIC provides a fast response to interrupt requests, allowing an application to quickly serve incoming events. The ARM V6-M limits to 32 the number of interrupt request inputs of the NVIC. However the STM32G0 implements more than 32 interrupt events thanks to the STM32G0's SYSCFG module.

- ARM Cortex®-M0+ limitation
- Since the number of interrupt events exceeds 32 in the STM32G0, the SYSCFG unit is in charge of combining several interrupts onto the same interrupt line
- User software can quickly determine the peripheral requesting the interrupt by reading ITLINEx registers in the SYSCFG module

- Dynamic reprioritization of interrupts
- Dynamic relocation of interrupt vector table

The priority assigned to each interrupt request is programmable and can be dynamically changed. 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. For instance, the vector table can be relocated in RAM.
Software is in charge of assigning a priority level to each interrupt as well as to all exception sources not including reset, NMI and hard fault. Whenever a peripheral interrupt is requested at the same time as a supervisor call instruction is executed, the relative priority of these hardware and software exceptions will dictate which one will be taken first.

Regarding the STM32G0, the NMI is caused by a SRAM parity error, a flash double ECC error or clock failure. The priority of any of the 32 peripheral interrupt requests is programmable in a dedicated priority field located in Cortex®-M0+ NVIC registers.

### Priority handling

- Regarding Cortex®-M CPUs exception management, the lower the value, the higher the priority.

<table>
<thead>
<tr>
<th>Exception source</th>
<th>Priority level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>-3</td>
</tr>
<tr>
<td>Non-Maskable Interrupt (NMI)</td>
<td>-2</td>
</tr>
<tr>
<td>Hard Fault</td>
<td>-1</td>
</tr>
<tr>
<td>Other exceptions including:</td>
<td></td>
</tr>
<tr>
<td>- Peripheral interrupts</td>
<td>Programmable level from 0 to 3</td>
</tr>
<tr>
<td>- Software exceptions</td>
<td></td>
</tr>
</tbody>
</table>

Fixed hardcoded priority

Programmable level from 0 to 3
The NVIC provides several features for efficient handling of exceptions. When an interrupt 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. A microcode present in the Cortex®-M0+ automatically pushes the context to the current stack and restores it upon interrupt return.

### Tail-chaining and nesting

- In order to explain the tail-chaining and nesting mechanism, let us consider the following peripheral interrupt sources:

<table>
<thead>
<tr>
<th>Interrupt source</th>
<th>Priority level</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ_A</td>
<td>0</td>
</tr>
<tr>
<td>IRQ_B</td>
<td>1</td>
</tr>
</tbody>
</table>

- Preemption and interrupt nesting
When an interrupt request with lower or equal priority is raised during execution of an interrupt handler, it becomes pending. Once the current interrupt handler is finished, the context saving and restoring process is skipped and control is transferred directly to the new exception handler when the previous handler is completed. So back-to-back interrupts with decreasing priorities (higher priority values) are chained with a very short latency of a few clock cycles.
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 the program context. Then tail-chaining will be used prior to executing the IRQ_B interrupt service routine.

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.
When accessing the NVIC registers, ensure that your code uses a correctly-aligned register access. Unaligned access is not supported for NVIC registers as well as all memory-mapped registers located in the Cortex®-M0+. An interrupt becomes pending when the source asks for service. Disabling the interrupt only prevents the processor from taking that interrupt. Make sure the related interrupt flag is cleared before enabling the interrupt vector.

Before relocating the vector table using the VTOR register, ensure that fault handlers, NMI and all enabled interrupts are correctly set up on the new location.
The NVIC is linked with the SYSCFG module and the Cortex-M0+ CPU. Please refer to the related presentations.
For detailed information, please refer to the programming manual for the Cortex®-M0+ core and the reference manual of the STM32G0.