NVIC CDD

for

LCD\_LED\_UART

Prepared by: Norhan Nassar

Mar 27, 2020

**Contents**

[**1.** **Global Variables** 1](#_Toc36227208)

[**2.** **Configurations** 1](#_Toc36227209)

[**3.** **NVIC APIs** 2](#_Toc36227210)

# **Global Variables**

**N/A**

# **Configurations**

**N/A**

1. **NVIC APIs**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_EnableInterrupt(u32 Copy\_u32InterruptNum); | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes * External Interrupt Name and each external interrupt is mapped to a certain number. * Or System Exception like: MEMMANAGE, USAGEFAULT, BUSFAULT | Output signal | N/A |
| **Description** | The functionality of this API is to enable the given external interrupt number or one  of system exceptions | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_DisableInterrupt(u32 Copy\_u32InterruptNum); | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes * External Interrupt Name and each external interrupt is mapped to a certain number. * Or System Exception like: MEMMANAGE, USAGEFAULT, BUSFAULT | Output signal | N/A |
| **Description** | The functionality of this API is to disable the given external interrupt number or one  of system exceptions. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_SetPending(u32 Copy\_u32InterruptNum); | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes * External Interrupt Name and each external interrupt is mapped to a certain number. * Or System Exception like: NMI, MEMMANAGE, USAGEFAULT, BUSFAULT, SVCALL | Output signal | N/A |
| **Description** | The functionality of this API is to set pending flag by software of given external interrupt number or one of system exceptions. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_ClearPending(u32 Copy\_u32InterruptNum); | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes * External Interrupt Name and each external interrupt is mapped to a certain number. * Or System Exception like: MEMMANAGE, USAGEFAULT, BUSFAULT, SVCALL | Output signal | N/A |
| **Description** | The functionality of this API is clear pending flag by software of given external interrupt number or one of system exceptions. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_GetPendingFlag(u32 Copy\_u32InterruptNum,  u8\* Copy\_PtrPendingStatus) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes * External Interrupt Name and each external interrupt is mapped to a certain number. * Or System Exception like: NMI, PENDSV, SYSTICK, MEMMANAGE, USAGEFAULT, BUSFAULT, SVCALL   Copy\_PtrPendingStatus:   * Type: pointer to u8 * Description: It takes pointer to u8 data to return the pending flag status in it. | Output signal | N/A |
| **Description** | The functionality of this API is get the pending flag status for a given external interrupt number or one of system exceptions. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_GetActiveFlag(u32 Copy\_u32InterruptNum,u8\* Copy\_PtrActiveStatus) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes * External Interrupt Name and each external interrupt is mapped to a certain number. * Or System Exception like: NMI, PENDSV, SYSTICK, MEMMANAGE, USAGEFAULT, BUSFAULT, SVCALL   Copy\_PtrActiveStatus:   * Type: pointer to u8 * Description: It takes pointer to u8 data to return the Active flag status in it. | Output signal | N/A |
| **Description** | The functionality of this API is get the active flag status for a given external interrupt number or one of system exceptions. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_SetPriorityGrouping(u8 Copy\_u8PriorityGroup) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u8PriorityGroup:   * Type: u32 * Description: desired number of Priority group bits. | Output signal | N/A |
| **Description** | The functionality of this API is choose number for priority group bits  🡪 number of sub priority groups. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_GetPriorityGroup(u8\* Copy\_PtrPriGroup) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_PtrPriorityGroup:   * Type: Pointer to u8 * Description: pointer to return number of priority group bits in it. | Output signal | N/A |
| **Description** | The functionality of this API is to return number of priority group bits. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_SetPriority(u32 Copy\_u32InterruptNum,u8 Copy\_u8SubPriority, u8 Copy\_u8Preemp) | | |
| **Return type** | It’s an enum of u8 Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: external interrupt name like:   WWDG, PVD... and so on from vector table  Copy\_u8SubPriority:   * Type: u8 * Description: sub priority for this interrupt according to the priority group   if priority group is smaller than number of bits for priority to this chip so this sub priority must equal to zero  Copy\_u8Preemp:   * Type: u8 * Description: Preemption for given external number it will return NOT\_OK if the priority exceed maximum priority level for each chip | Output signal | N/A |
| **Description** | The functionality of this API is to set the priority for given external interrupt.. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_DisableGlobalInterrupt() | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to disable all exceptions except NMI and hard fault. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_EnableGlobalInterrupt() | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to enable global interrupt. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_DisGloInterrAndHardFault() | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to disable all exceptions even hard fault exception  But cannot disable NMI exception only | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NNVIC\_EnaGloInterrAndHardFault() | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to enable global interrupt and hard fault exception. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_DisInterrsWithPriority(u8 Copy\_u8InterruptNum) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to disable interrupts only with priority lower than or equal a certain level | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR NVIC\_SoftwareInterrupt(u32 Copy\_u8InterruptNum); | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to make software interrupt for given Interrupt Number | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_GetPendingInterruptNum(u8\* Copy\_PtrInterruptNum) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_PtrInterruptNum:   * Type: Pointer to u8 * Description: pointer to u8 to return number of interrupt that its pending flag is high in it. | Output signal | N/A |
| **Description** | The functionality of this API is to return the exception number of the highest priority pending enabled exception.  It returns error if there is no pending interrupt at this time. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_GetActiveInterruptNum(u8\* Copy\_PtrInterruptNum) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_PtrInterruptNum:   * Type: Pointer to u8 * Description: pointer to u8 to return number of interrupt that its active flag is high in it. | Output signal | N/A |
| **Description** | The functionality of this API is to return the active exception number  It returns error if there is no Active interrupt at this time (Thread mode) | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_SetVectorTableBaseOffset(u32 Copy\_u32BaseOffset,u32 Copy\_u32MemoryRegion) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32BaseOffset :   * Type: u32 * Description: the offset of the table base from memory address 0x00000000   Copy\_u32MemoryRegion:   * Type: u32 * Description: Code or SRAM to choose whether the vector table is in the code or SRAM memory region | Output signal | N/A |
| **Description** | The functionality of this API is to change memory address of vector table. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_GetVectorTableBaseOffset(u8\* Copy\_PtrMemoryRegion); | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_PtrMemoryRegion :   * Type: Pointer to u8 * Description: if the vector table is set at Code so it returns 1 at this pointer   And if it is at SRAM it will return 0 at this pointer  1 -> SRAM  0 -> CODE  Copy\_u32MemoryRegion:   * Type: u32 * Description: Code or SRAM to choose whether the vector table is in the code or SRAM memory region | Output signal | N/A |
| **Description** | The functionality of this API is to detect the vector table is set at Code or SRAM | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_EnableTrapDivByZero() | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to enable faulting or halting when the processor executes an SDIV or UDIV instruction with a divisor of 0. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_DisableTrapDivByZero() | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | N/A | Output signal | N/A |
| **Description** | The functionality of this API is to disable faulting or halting when the processor executes an SDIV or UDIV instruction with a divisor of 0. | | |
| **Type (Public/Private)** | Public | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_GetActiveFlag(u32 Copy\_u32InterruptNum,  u8\* Copy\_PtrActiveStatus) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes system exception like: NMI, PENDSV, SYSTICK, MEMMANAGE, USAGEFAULT, BUSFAULT, SVCALL   Copy\_PtrActiveStatus:   * Type: pointer to u8   Description: It takes pointer to u8 data to return the Active flag status in it. | Output signal | N/A |
| **Description** | To get Active flag for system exceptions. | | |
| **Type (Public/Private)** | Private | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_ICSR\_GetPendingFlag(u32 Copy\_u32InterruptNum,u8\* Copy\_PtrPendingStatus) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes system exception like: NMI, PENDSV, SYSTICK   Copy\_PtrActiveStatus:   * Type: pointer to u8   Description: It takes pointer to u8 data to return the Pending flag status in it. | Output signal | N/A |
| **Description** | To get Pending flag for system exceptions at ICSR Register. | | |
| **Type (Public/Private)** | Private | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | NVIC | | |
| **API Name** | STD\_ERROR SCB\_SHCSR\_GetPendingFlag(u32 Copy\_u32InterruptNum,u8\* Copy\_PtrPendingStatus) | | |
| **Return type** | It’s an enum of Error\_Status, it returns OK or NOT\_OK   |  |  | | --- | --- | | OK | 0 | | NOT\_OK | 1 | | | |
| **Input signal** | Copy\_u32InterruptNum:   * Type: u32 * Description: It takes system exception like: USAGEFAULT, BUSFAULT, MEMMANAGE or SVCALL   Copy\_PtrActiveStatus:   * Type: pointer to u8   Description: It takes pointer to u8 data to return the Pending flag status in it. | Output signal | N/A |
| **Description** | To get Pending flag for system exceptions at SHCSR Register. | | |
| **Type (Public/Private)** | Private | | |