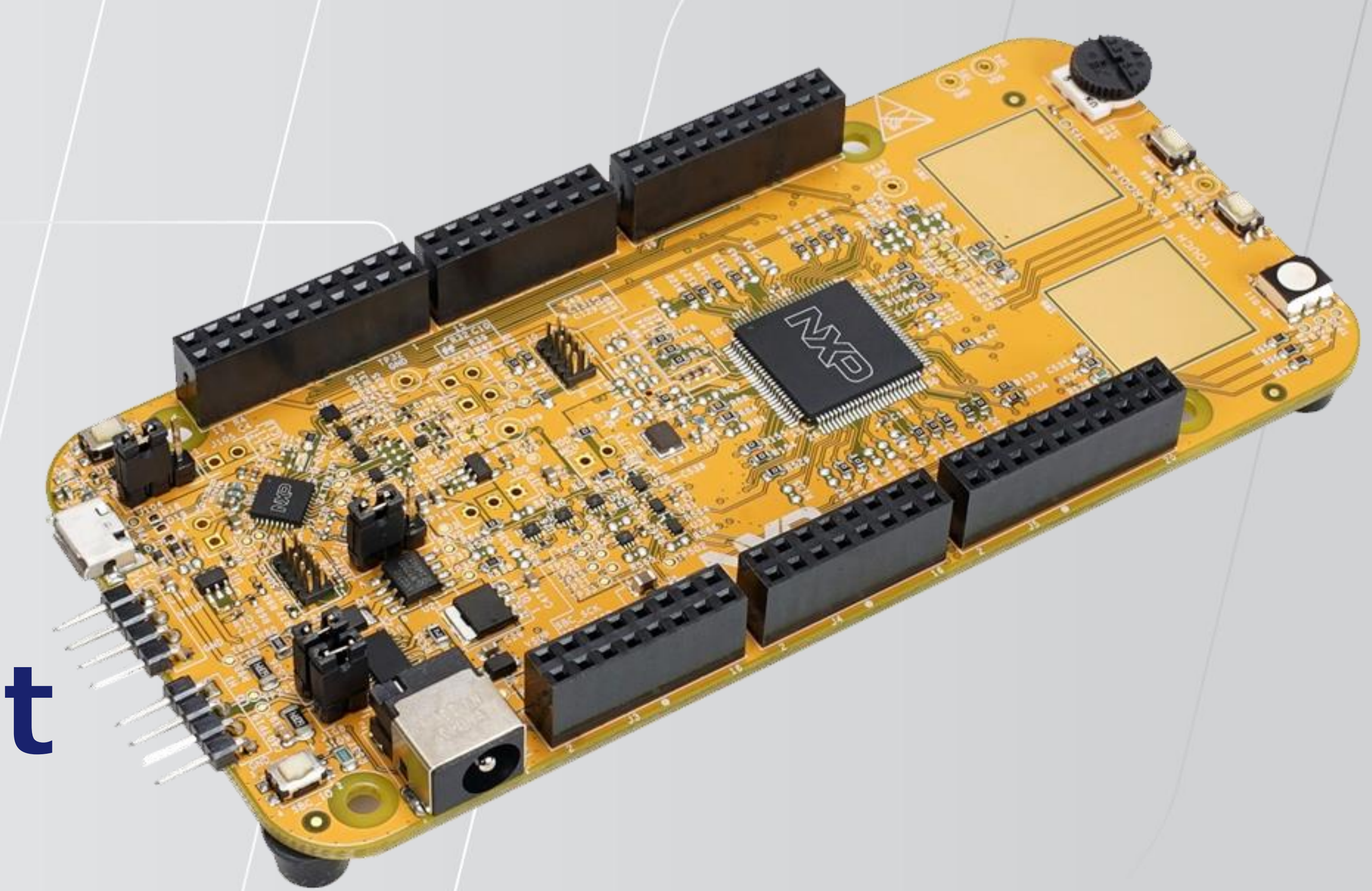
GPIO – General Purpose Input/Output



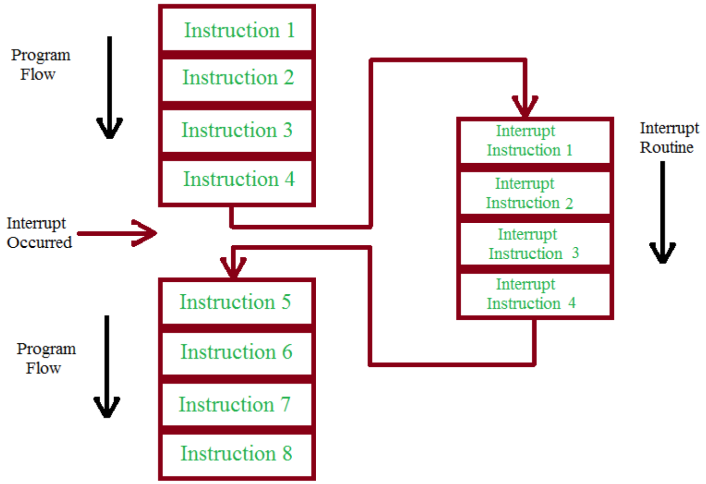
8 Dec 2024

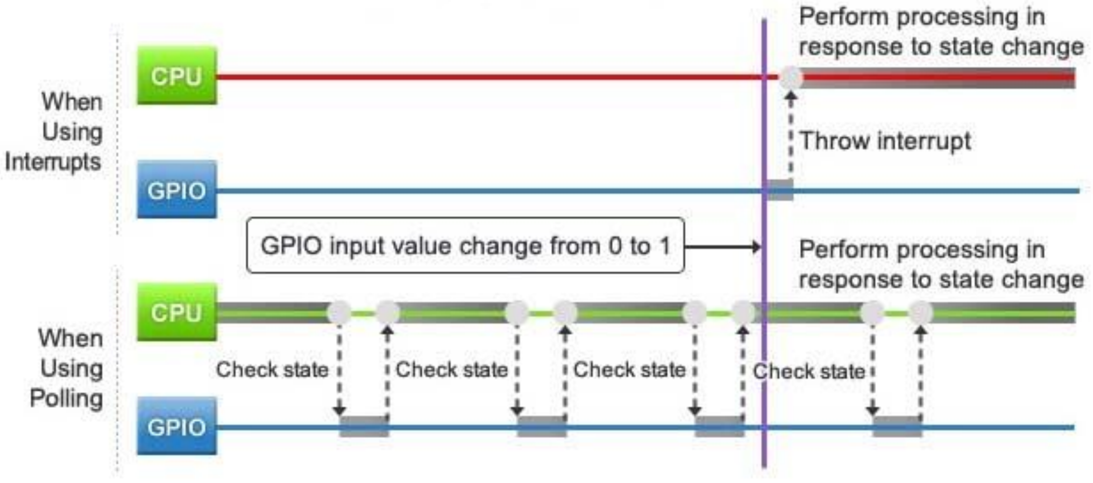
OUTLINE

1. Polling and interrupt
2. Modules overview
3. Port module – Interrupt Configuration
4. Configuration

1. Polling and interrupt

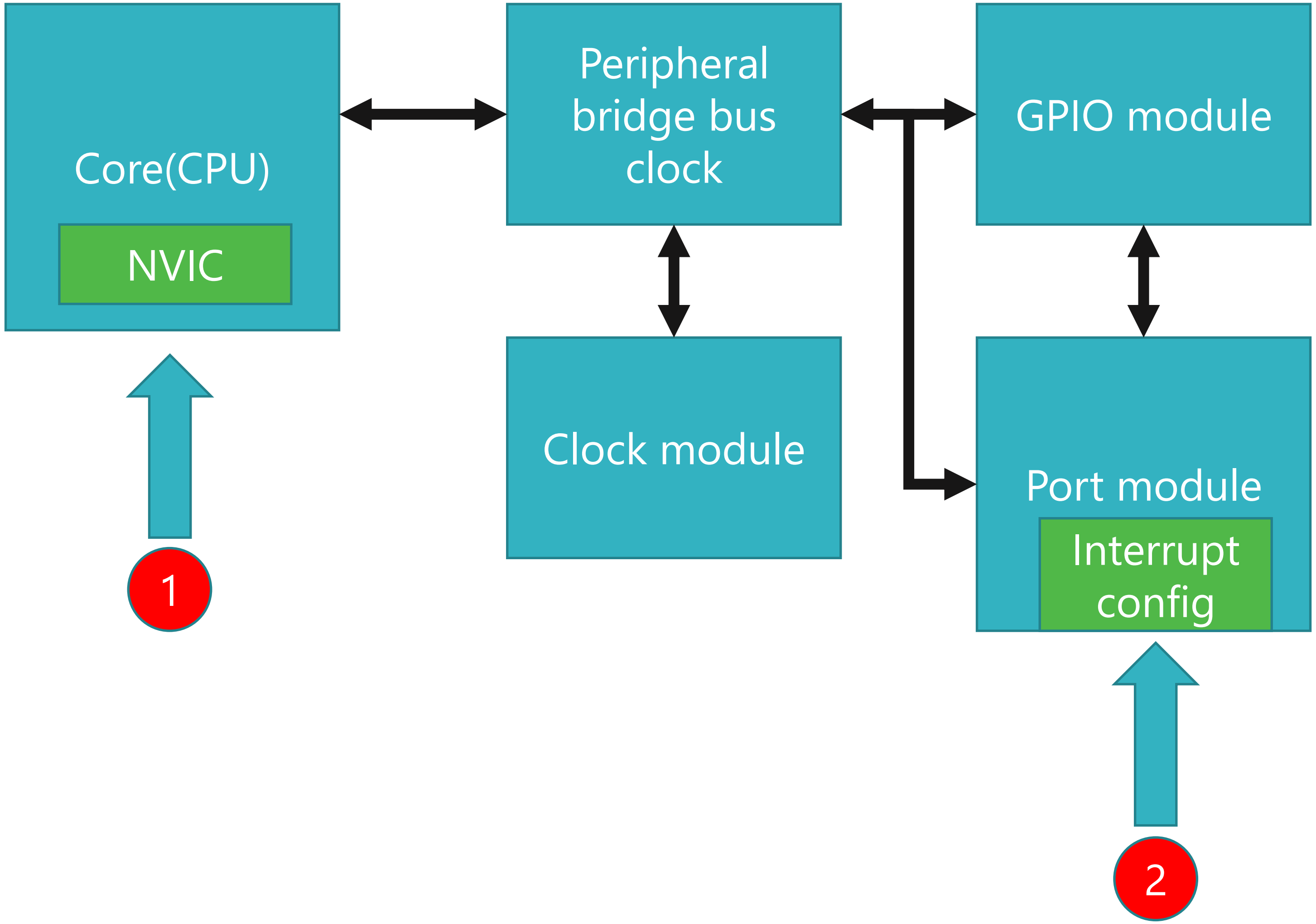
Criteria	Polling	Interrupt
Concept	Continuously checking device status	Notifying when an event occurs
Operation	The microcontroller runs a loop to check status	The microcontroller pauses the current program and executes the ISR
Advantages	- Simple and easy to implement - Easy to understand and control	- More efficient CPU usage - Frees up CPU resources for other tasks
Disadvantages	- Time-consuming and resource - Can reduce system performance	- More complex management - Potential for interrupt conflicts
Use Cases	When requirements are simple or real-time is not critical	When fast response and high performance are needed





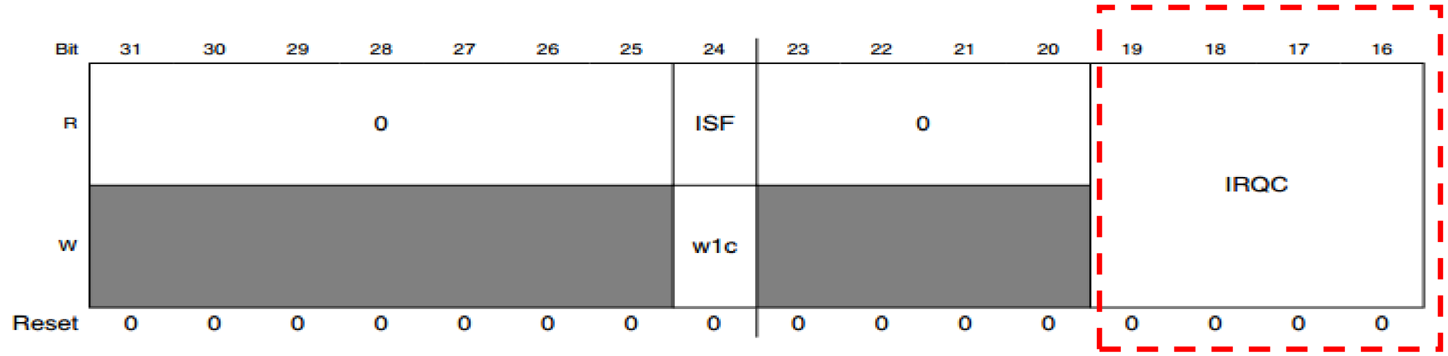
When Using Interrupts
CPU
GPIO
Perform processing in response to state change
Throw interrupt
GPIO input value change from 0 to 1
When Using Polling
CPU
GPIO
Perform processing in response to state change
Check state
Check state
Check state
Check state

2. Modules overview

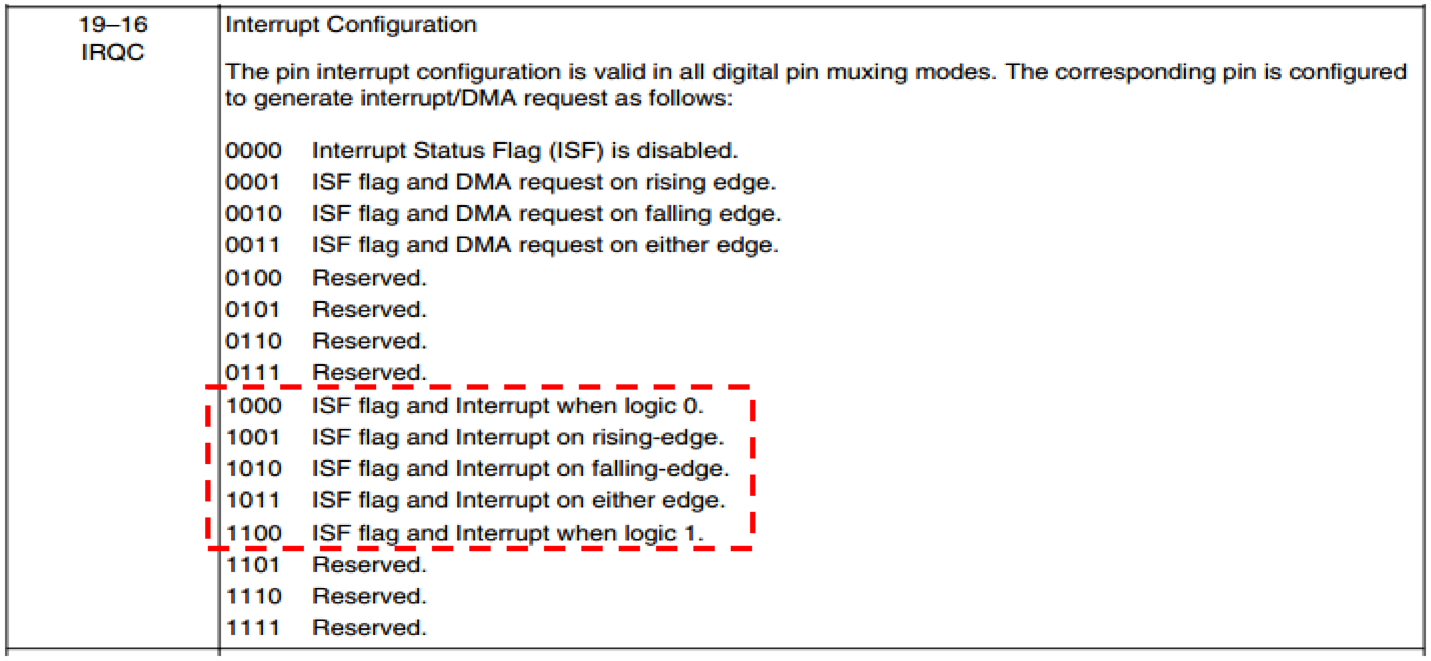


3. Port module – Interrupt Configuration

Pin Control Register n (PORT_PCR*n*)



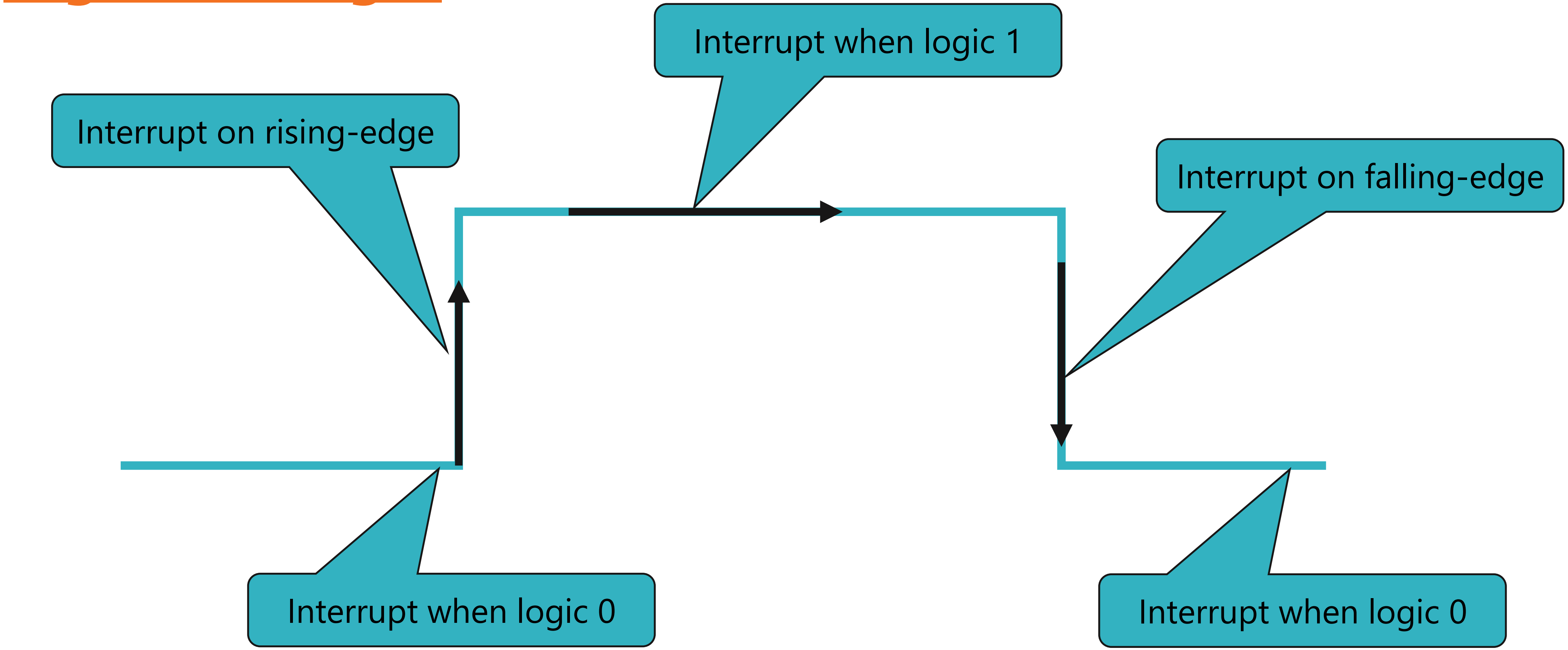
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							ISF	0				IRQC			
W	▨							w1c	▨							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



19–16 IRQC	Interrupt Configuration The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt/DMA request as follows: 0000 Interrupt Status Flag (ISF) is disabled. 0001 ISF flag and DMA request on rising edge. 0010 ISF flag and DMA request on falling edge. 0011 ISF flag and DMA request on either edge. 0100 Reserved. 0101 Reserved. 0110 Reserved. 0111 Reserved. 1000 ISF flag and Interrupt when logic 0. 1001 ISF flag and Interrupt on rising-edge. 1010 ISF flag and Interrupt on falling-edge. 1011 ISF flag and Interrupt on either edge. 1100 ISF flag and Interrupt when logic 1. 1101 Reserved. 1110 Reserved. 1111 Reserved.

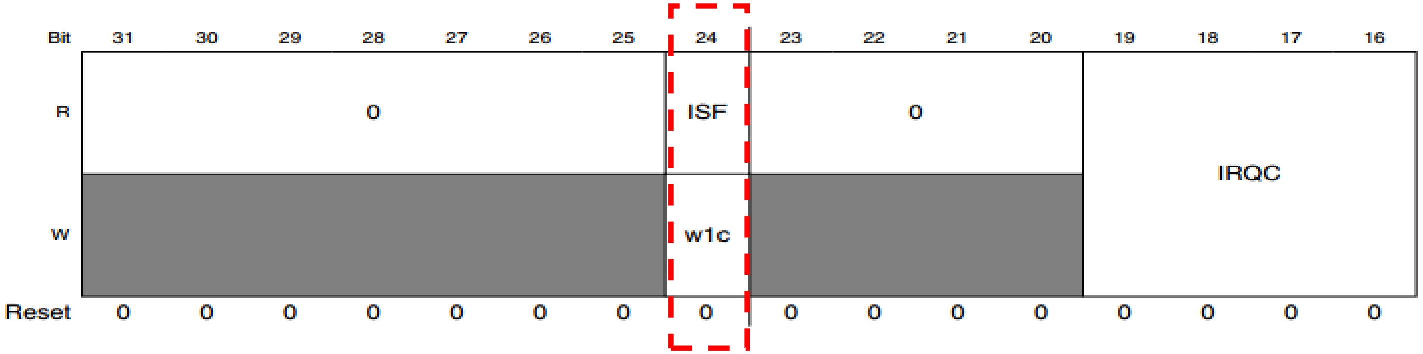
3. Port module – Interrupt Configuration

Logic and edge:

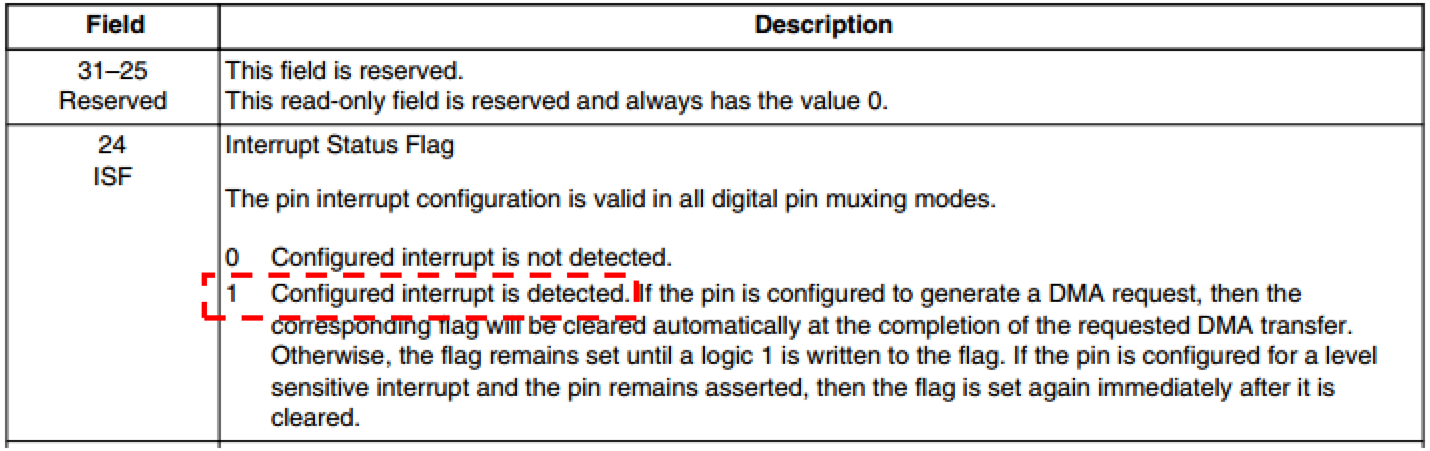


3. Port module – Interrupt Configuration

Pin Control Register n (PORT_PCR*n*)

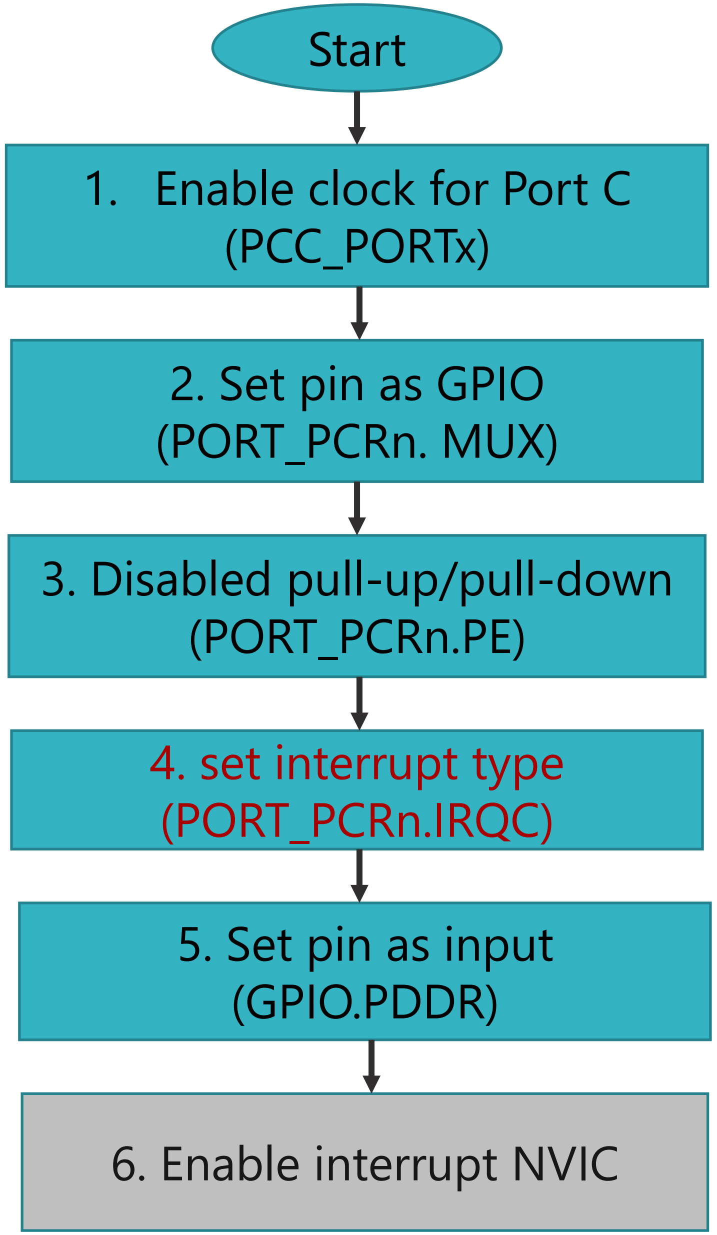


Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							ISF	0				IRQC			
W								w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 ISF	Interrupt Status Flag The pin interrupt configuration is valid in all digital pin muxing modes. 0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

4. Configuration


Start
1. Enable clock for Port C (PCC_PORTx)
2. Set pin as GPIO (PORT_PCRn. MUX)
3. Disabled pull-up/pull-down (PORT_PCRn.PE)
4. set interrupt type (PORT_PCRn.IRQC)
5. Set pin as input (GPIO.PDDR)
6. Enable interrupt NVIC

4.1. ISR (Interrupt Service Routine)

- ISR (Interrupt Service Routine) is a function or code segment that is executed when an interrupt occurs.
- When an interrupt occurs, the microcontroller temporarily pauses the current program execution and switches to executing the ISR to handle the related event.

The main characteristics of an ISR:

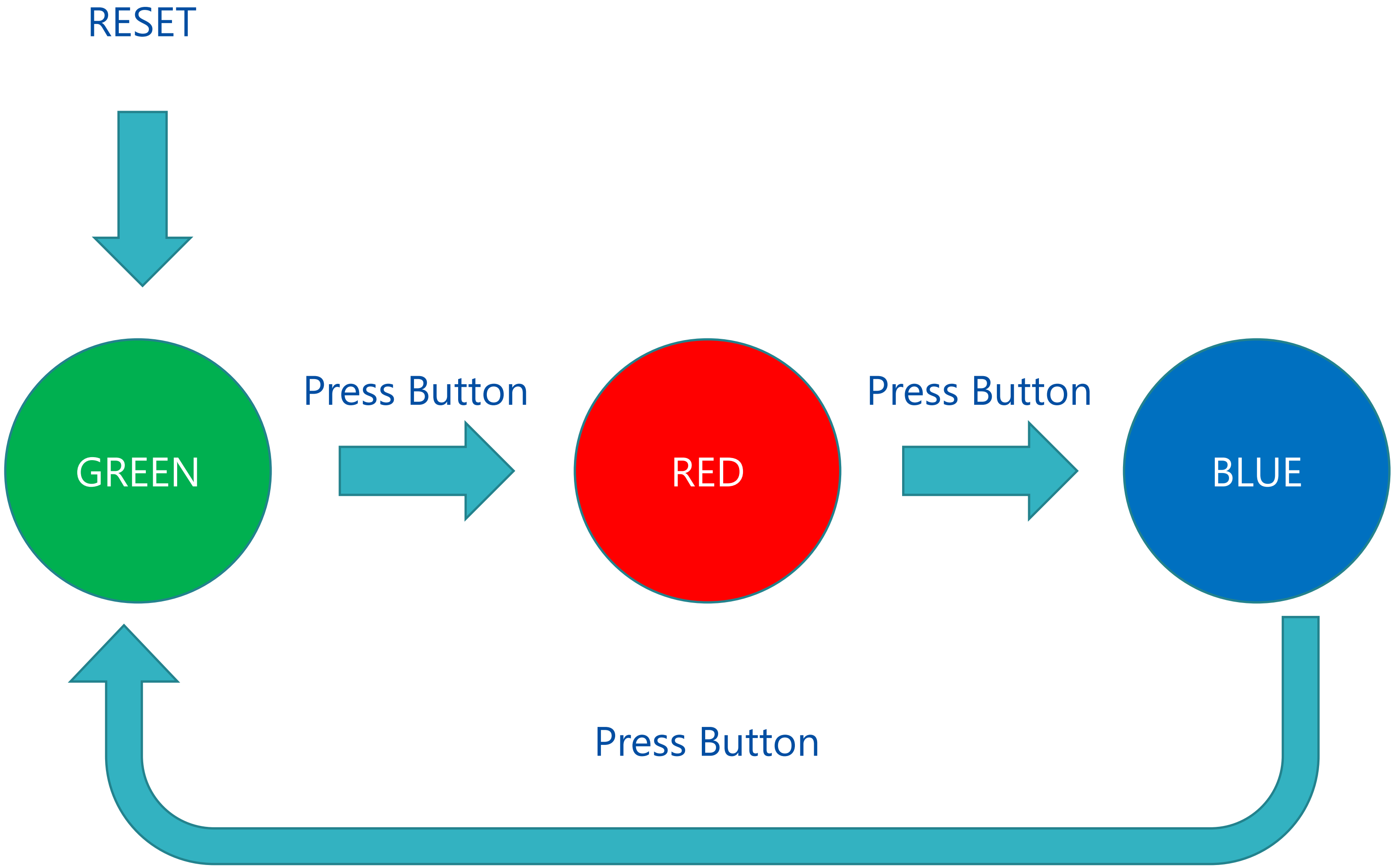
- ❖ Quick execution: The ISR should execute quickly to avoid long interruptions of the main process.
- ❖ Interrupts disabled: Interrupts are typically disabled during ISR execution to prevent conflicts.
- ❖ **Interrupt status flag**: After processing, the **interrupt status flag** is usually **cleared** to ensure the interrupt is not handled again immediately.

4.1. ISR (Interrupt Service Routine)

- The ISR (Interrupt Service Routine) for external interrupts is declared in the file startup_S32K144.S

No.	EXTIx	Name ISR
1	PORTA_x	PORTA_IRQHandler
2	PORTB_x	PORTB_IRQHandler
3	PORTC_x	PORTC_IRQHandler
4	PORTD_x	PORTD_IRQHandler
5	PORTE_x	PORTE_IRQHandler

4.2. Assignment



Thank you