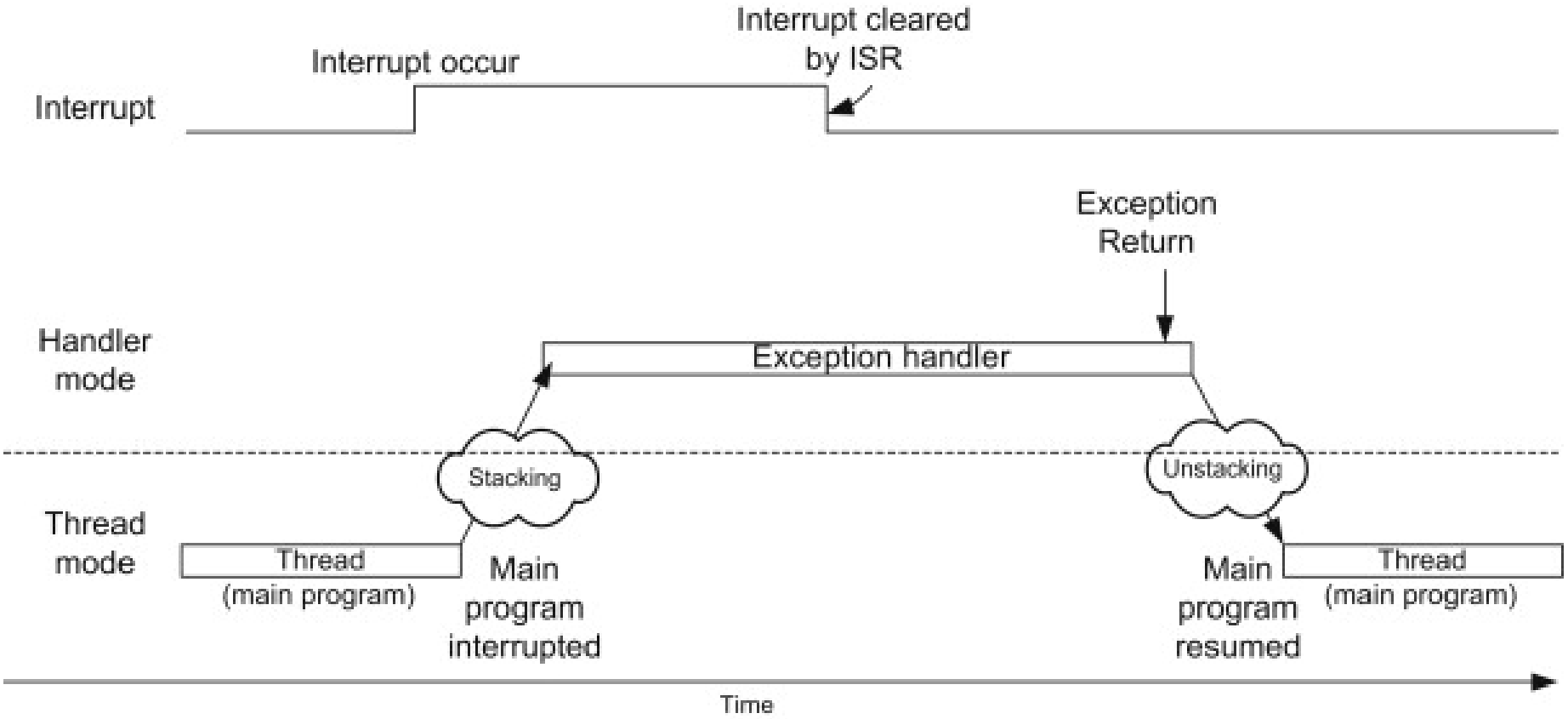
Lesson 4
Interrupt

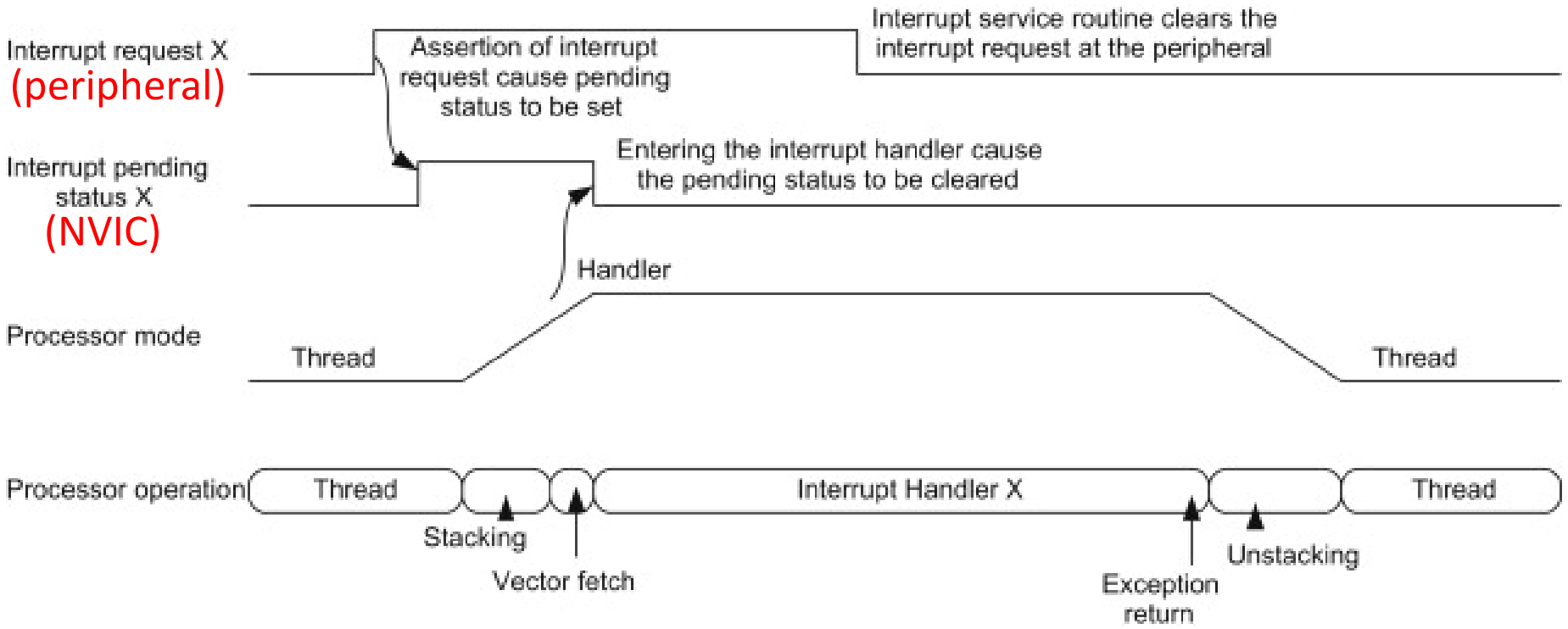
Lecturer: Harvard Tseng

Simply view at exception entry & exit



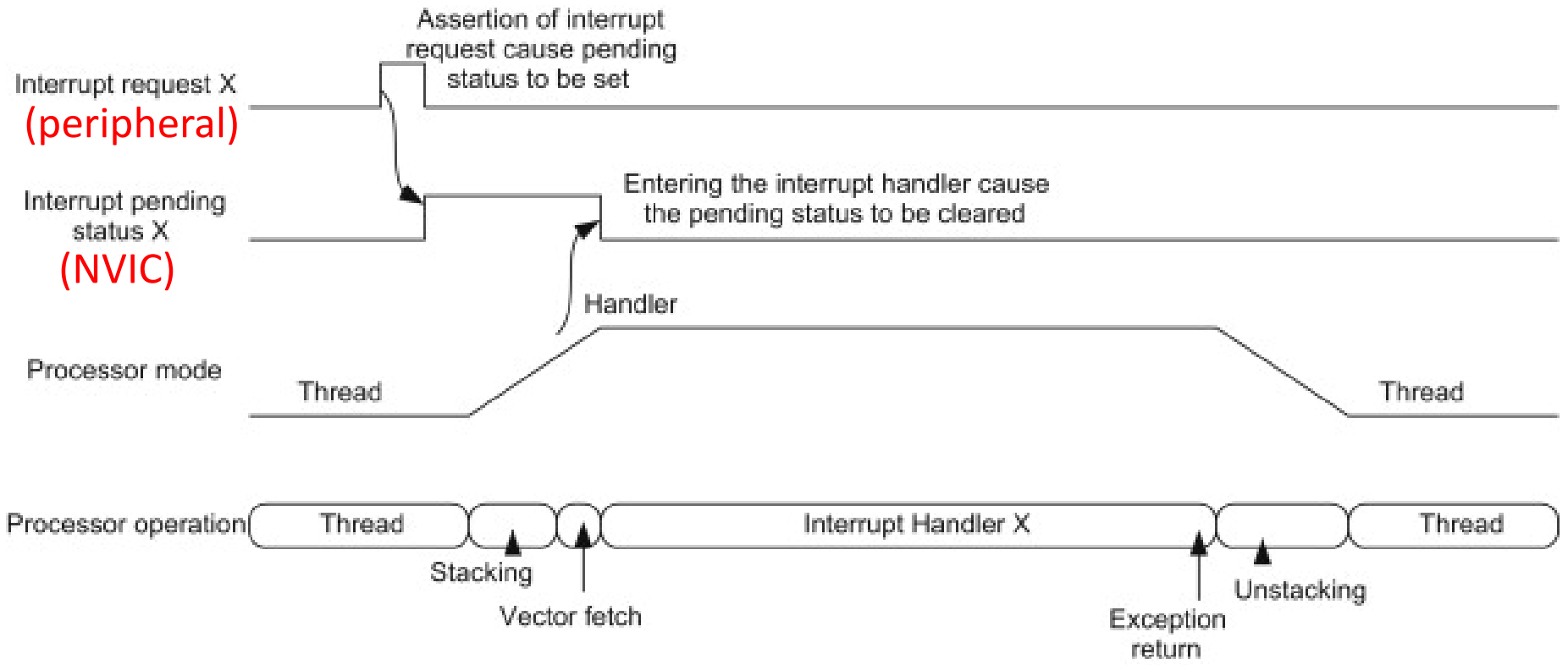
Level Trigger

- Peripheral interrupt request will store in status register(TIMx) or pending register(EXTI).



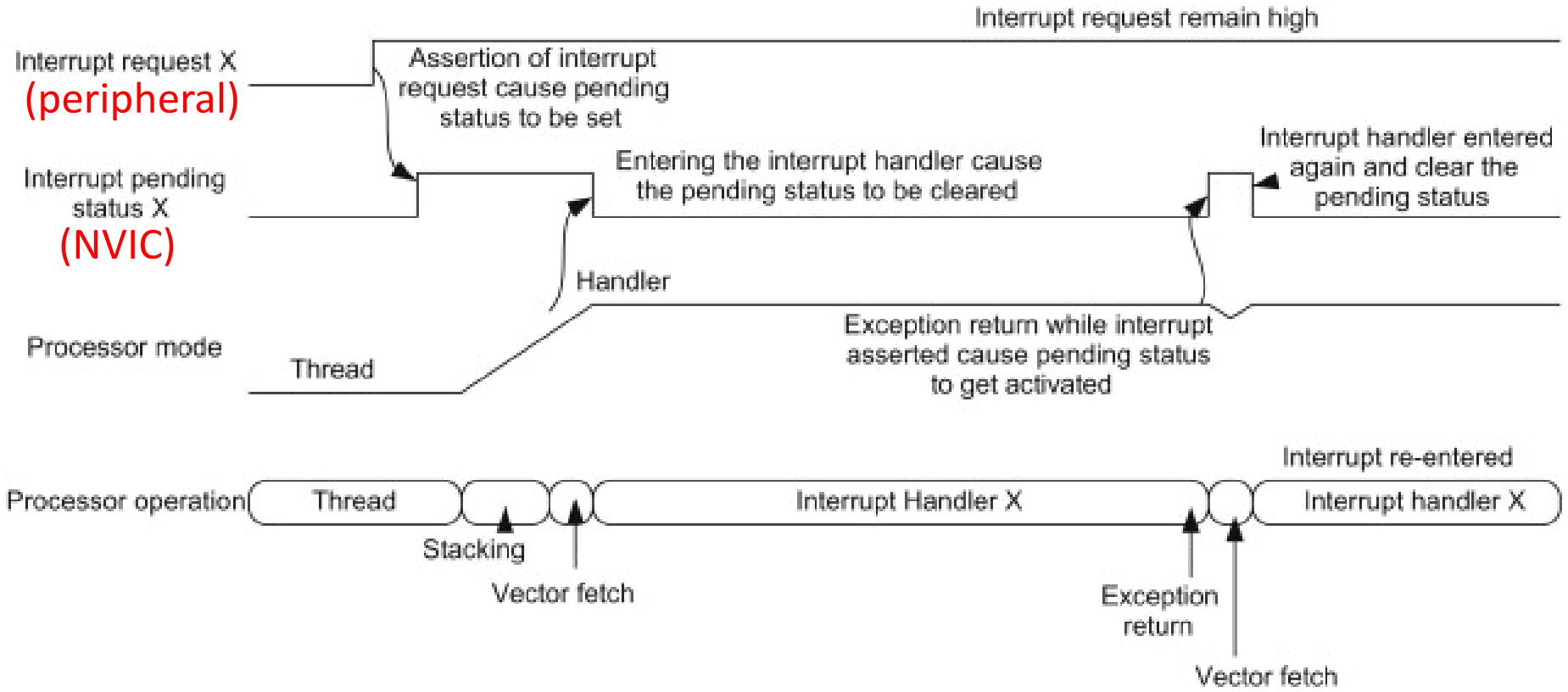
Pulse Trigger

- Pending status register will hold the request until the interrupt is being served.



If do not clear interrupt request

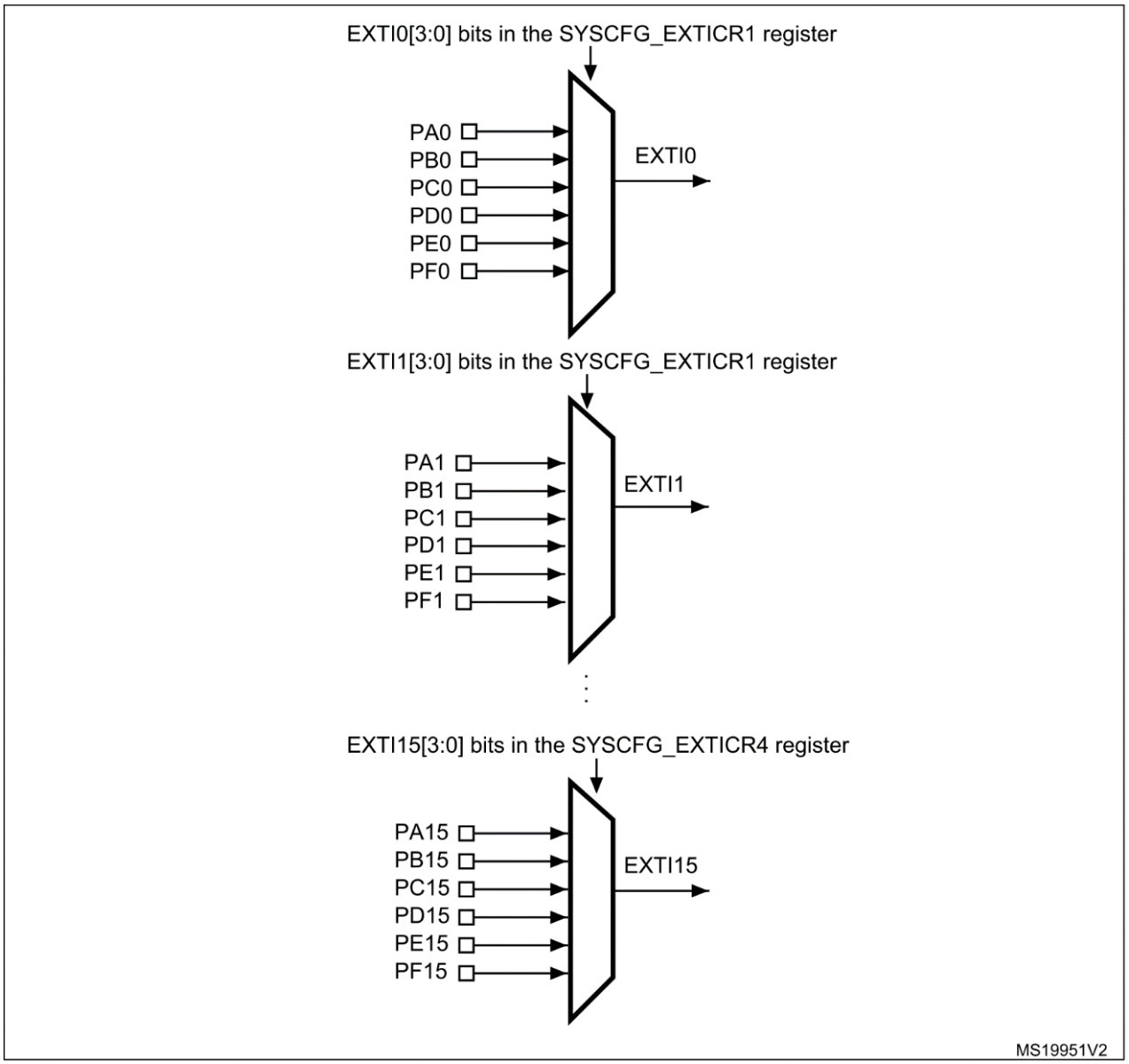
- Will re-enter interrupt handler again and again.



EXTI

Extended interrupts and events controller

External interrupt/event GPIO mapping



Register

- EXTI_IMR – Interrupt mask register – to mask interrupt request or not.
- EXTI_RTSR - Rising trigger selection register
- EXTI_FTSR - Falling trigger selection register
- EXTI_PR - Pending register - is set when the selected edge event arrives on the external interrupt line.

EXTI interrupt selection

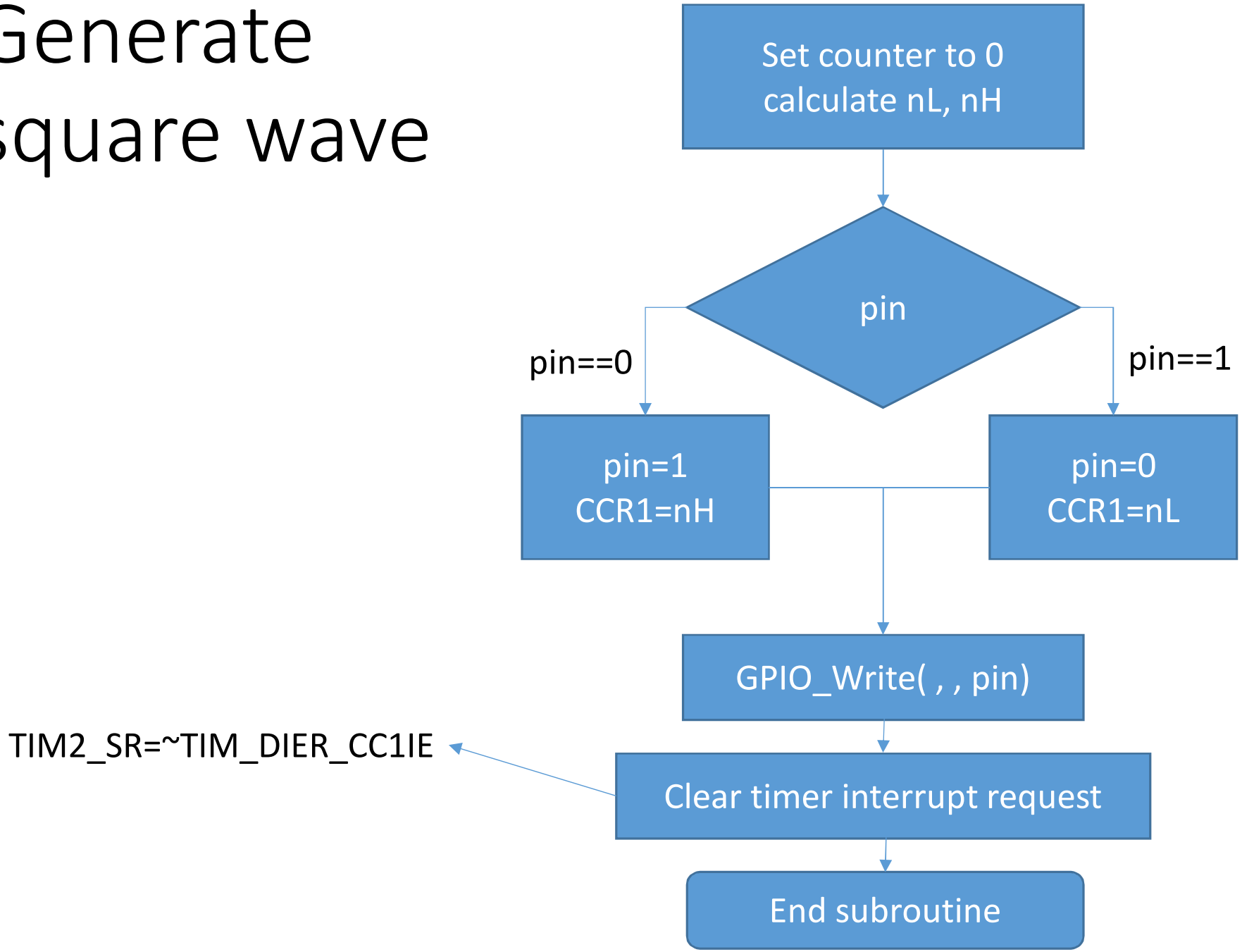
- Eg. PA0 as interrupt source.
- EXTI_IMR |= EXTI_IMR_IM0; // Set mask bit(EXTI_Line0)
- EXTI_RTSR |= EXTI_RTSR_RT0; // Active when rising edge occur
- NVIC_EnableIRQ(EXTI0_1_IRQn); // Enable NVIC IRQ channel

Timer

Register

- TIMx_CCRy - capture/compare register – if counter matches CCR value, will generate interrupt.
- TIMx_SR – status register – record CCR interrupt information.
- TIMx_DIER - DMA/Interrupt enable register – enable DMA/Interrupt request.

Generate square wave



Timer CCR interrupt configuration

- TIM2_DIER |= TIM_DIER_CC1IE; // Capture/Compare 1 interrupt enable
- TIM2_CCR1 = 100; // Give a match value to compare
- NVIC_EnableIRQ(TIM2_IRQn); // Enable NVIC IRQ channel

Peripheral Interrupt Handler

- Every interrupt handle's name is already well defined in startup_<device>.s .
- Implement handler function under main() or in independent source file.

```
int main(void){
    …
}
void TIM2_IRQHandler(void){
    …
}
```

Exercise

- If button pressed, increase frequency by 1. When frequency is bigger than 10, set to 1.
- Use ISR to handle actions when button pressed or timer match compare value.