MUNI FI

PV198 - Timers and Interrupts

One-chip Controllers

Dávid Danaj, Marek Vrbka

Faculty of Informatics, Masaryk University

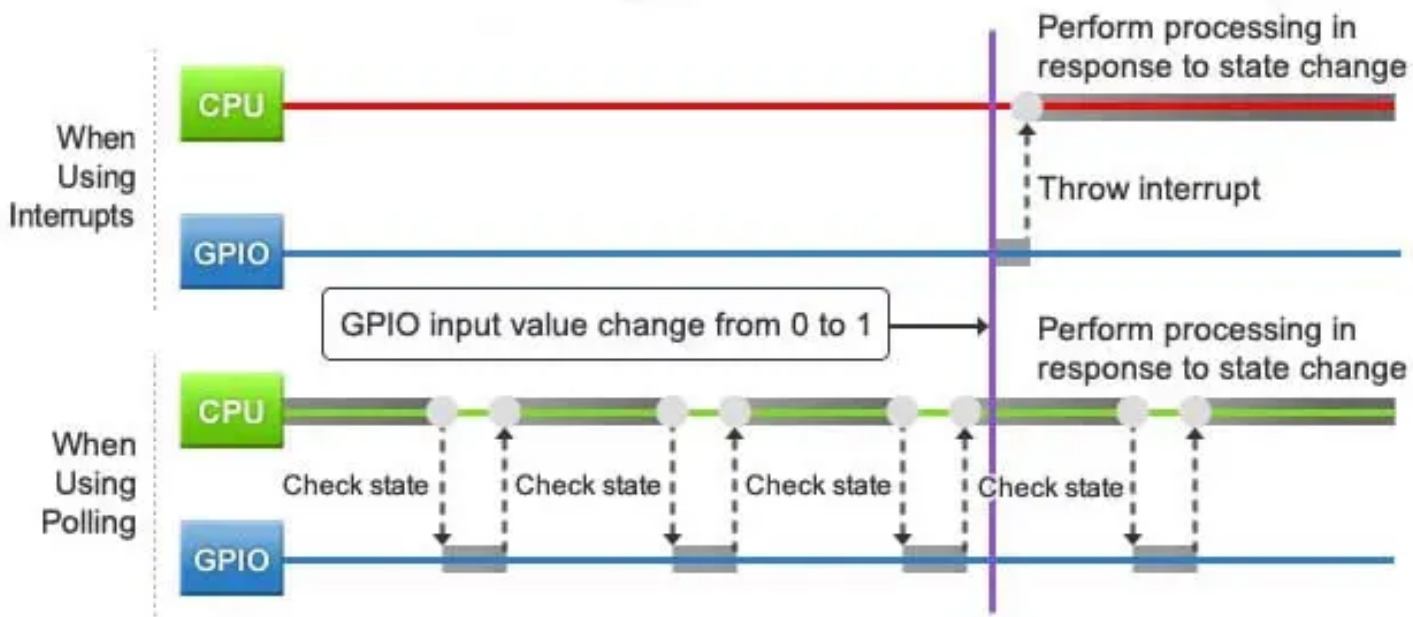
Content

Intro

- Do you have any questions related to HW2?
- Switch your branch to *Week*_03!

What are Interrupts

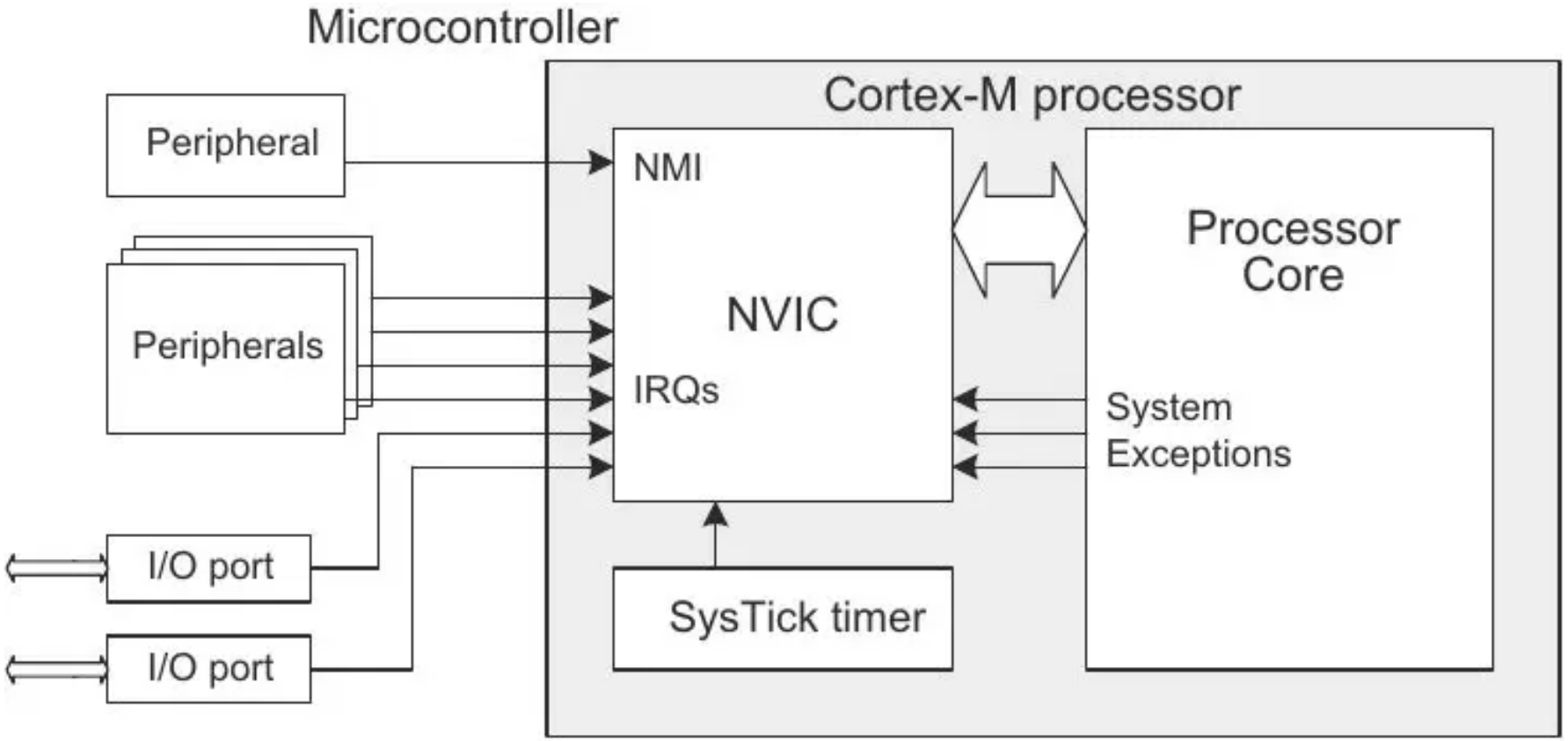
- Signal to the processor indicating an event



https://www.motioncontroltips.com/what-is-nested-vector-interrupt-control-nvic/

Interrupt Types

- Level vs. Edge trigger
- HW vs. SW interrupt



https://www.motioncontroltips.com/what-is-nested-vector-interrupt-control-nvic/

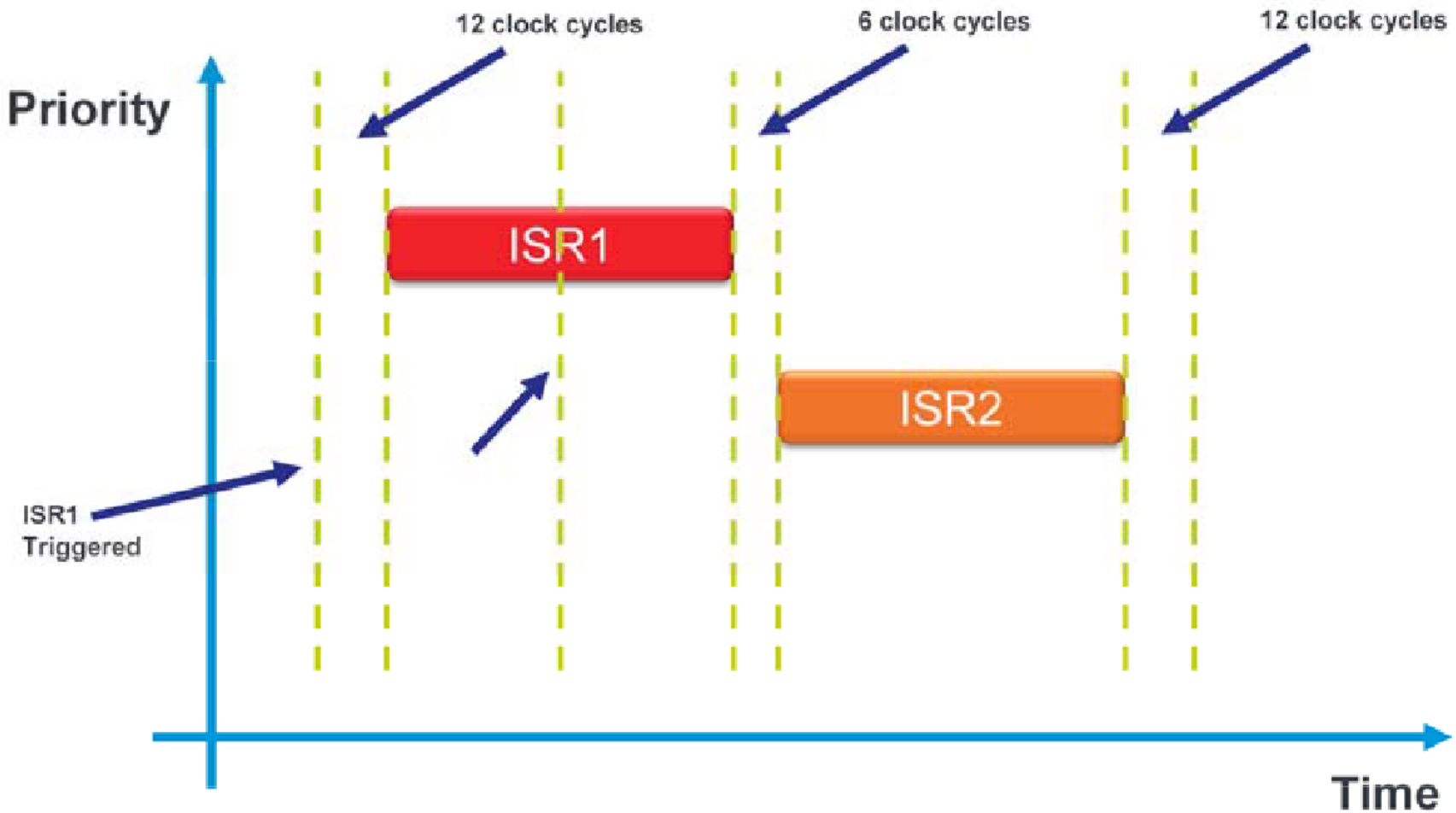
NVIC

- NVIC – Nested Vector Interrupt Controller
- Up to 120 interrupt sources
- Up to 16 priority levels
- Nested interrupts
- 12 clock cycles to enter/exit an ISR (Interrupt Service Routine)
- 6 clock cycles when switching from one ISR to another

Interrupt Operations

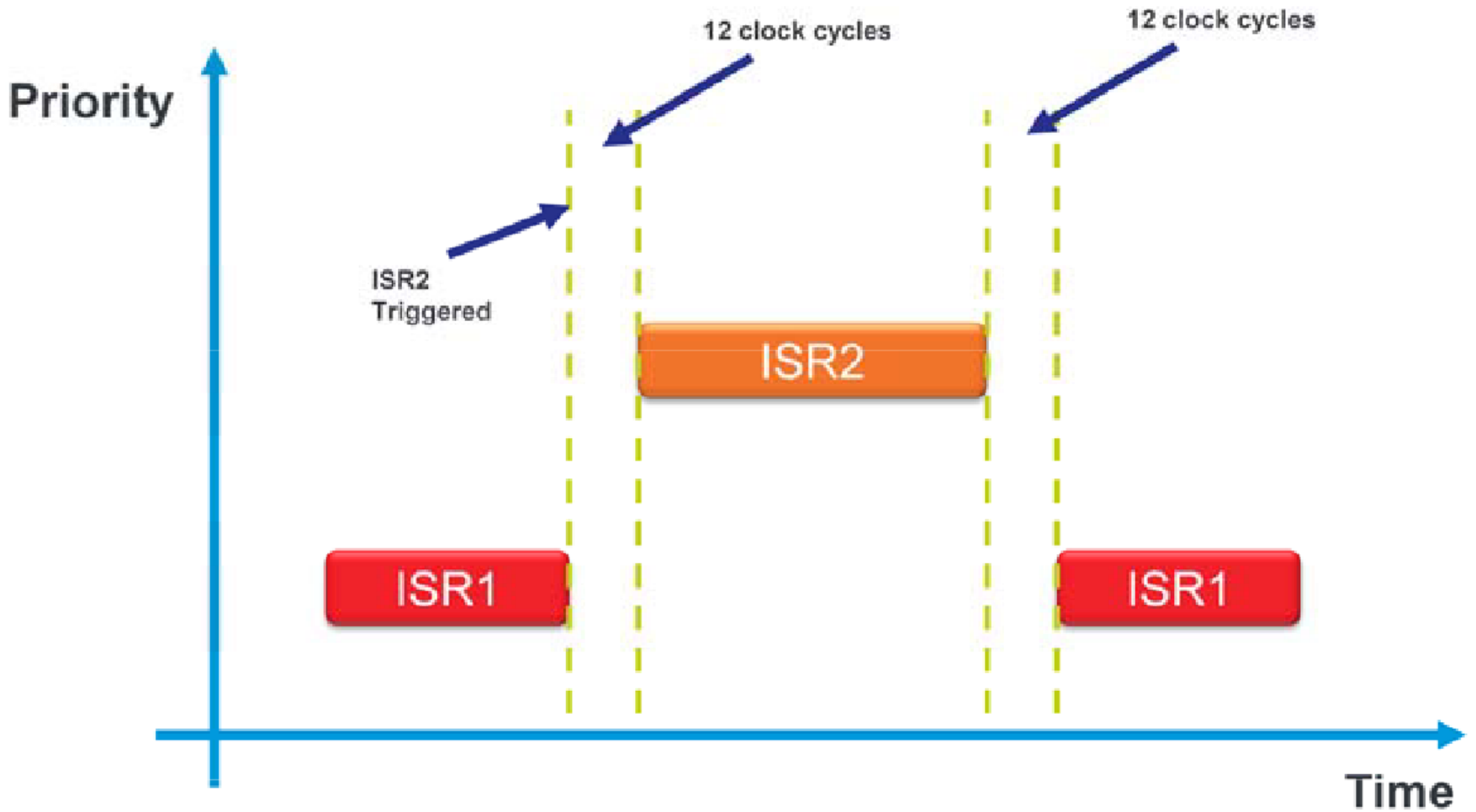
1. Interrupt is triggered
2. MCU finishes execution of the current instruction
3. Save current state (PC, flags, registers)
4. Get address of the ISR (Interrupt Service Routine) from the Interrupt Vector Table
5. Execute the ISR
6. Load previously saved state

Tail Chaining



https://www.nxp.com/docs/en/supporting-information/Nested-Vector-Interrupt-Controller-Training.pdf

Preemption



https://www.nxp.com/docs/en/supporting-information/Nested-Vector-Interrupt-Controller-Training.pdf

Preparation

Task

1. Prepare a new empty project
2. Setup functional group for buttons
3. Enable GPIO peripheral (note: device-specific driver)

Interrupt Code

1. Configure interrupt for a pin
   - Can be done in code or in the *Pins tool*
2. Enable interrupts for a port
   - GPIOA for SW3
   - Can be done in code or in the *Peripherals tool*
   - Define your handler name (*Peripherals tool*)
3. Write interrupt handler
   - Function with the same name as you defined the handler name
   - Clear the flag that triggers the interrupt (what will happen if you don't?)

```
void SW3_BUTTON_PRESSED_IRQ(void) {
    GPIO_PortClearInterruptFlags(BOARD_SW3_GPIO, 1U <<
    BOARD_SW3_GPIO_PIN);
    g_ButtonSw3Press = true;
}
```

Task

- Write an application that prints text to console when SW3 is pressed
- Use interrupts

Timers

- We could also call them counters
- They measure time intervals by counting
  - Clock cycles
  - Events
  - External clocks
- We can use them to measure time intervals
  - Periodic execution
  - Delay implementation

Periodic Interrupt Timer (PIT)

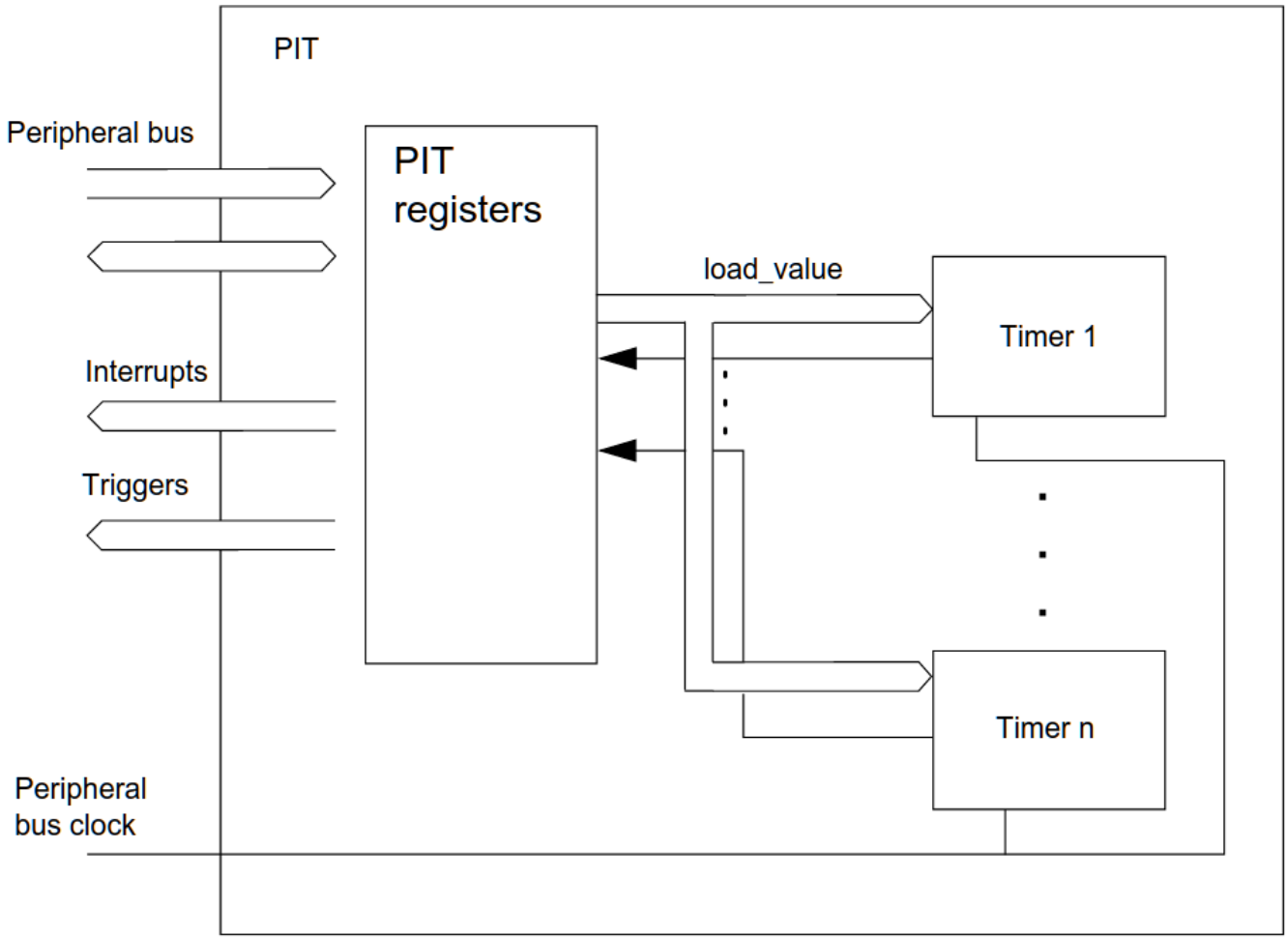


Figure 46-1. Block diagram of the PIT

https://www.mouser.com/datasheet/2/813/K66P144M180SF5RMV2-1074869.pdf, page 1268

FRDM-K66F Timers

- Programmable delay block (PDB)
- Flexible timer modules (FTM)
- Low Power TPM (TPM)
- Periodic interrupt timers (PIT)
- Low-power timer (LPTimer)
- Carrier modulator timer (CMT)
- Real-time clock (RTC)
- IEEE 1588 timers

PIT Features

- 4 timers
- Ability of timers to generate interrupts
- Independent timeout periods for each timer
- Chaining of timers

PIT configuration behavior

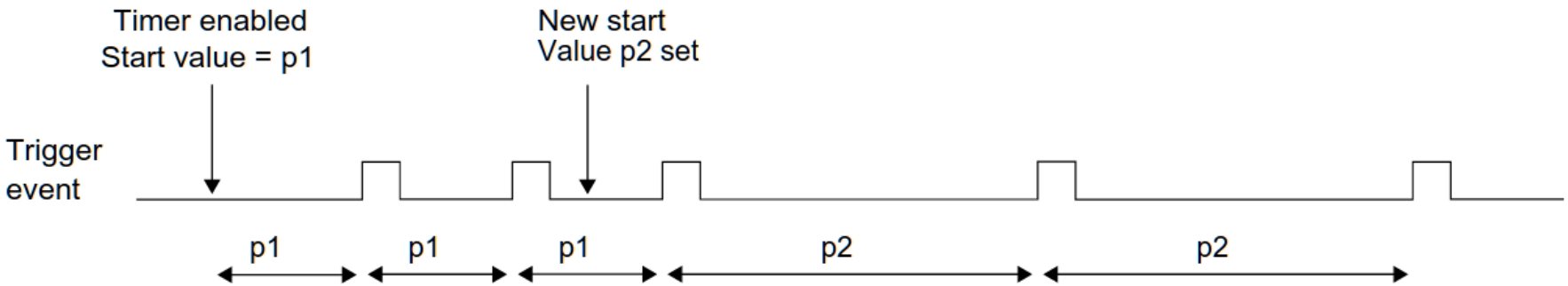


Figure 46-4. Dynamically setting a new load value

Simple use example

- Set value to a timer register
  - Timer Load Value Register (PIT_LDVALn)
- The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again

Which value to set?

- Interrupt every 0.5 seconds
- Bus clock: 60 MHz $\rightarrow$ clock period = 16.67 ns
- LDVAL trigger = $(period/clockperiod) - 1$
- $0.5\ s/16.67\ ns = 30000000 - 1 = 29999999\ cycles$

Task 2

- Write an application that prints text to console every 1 second
- Update it to toggle LED every 0.5 seconds
- Setup the system in a way that timer is enabled with SW3 button

Homework - Debounce a button using Timer Interrupts

- Use GPIO interrupt to detect first edge on the SW3 button
- When GPIO is triggered, start Timer and wait for constant time
- When Timer interrupt triggers, check GPIO value
- Print anything to console when press signal is accepted

MUNI
FACULTY
OF INFORMATICS