Vaja 5

Zunanje prekinitve

Naloga

Napišite program, ki bo nadziral števec dogodkov na dveh pinih (tipke). Pritisk na T1 naj števec poveča za 1, pritisk na T4 pa zmanjša za 1.

Na prikazovalniku LCD naj bo ves čas prikazano trenutno stanje števca

Uporabite zunanje prekinitve

Različni pristopi

1. s povpraševanjem (polling), izvedba z zanko

V zanki ves čas opazujemo stanje na pinih in v primeru pritiska na tipko povečamo oz. zmanjšamo števec.

Hitrost odziva je odvisna od trajanja drugih opravil v zanki (npr. uporaba LCD)

2. z zunanjimi prekinitvami

Ob določenem signalu na pinu se glavni program avtomatsko prekine (stanje registrov se shrani na sklad), da se lahko izvede prekinitvena rutina (ISR)

Hiter odziv (majhna latenca)

Zunanje prekinitve

Konfiguracija NVIC (glej thirdparty/CMSIS/include/core_cm3.h)

__STATIC_INLINE void NVIC_EnableIRQ(IRQn_Type IRQn)

Omogoči prekinitev

IRQn: oznaka prekinitve (glej sam3x8e.h)

__STATIC_INLINE void NVIC_DisableIRQ(IRQn_Type IRQn)

Onemogoči prekinitev

__STATIC_INLINE void NVIC_ClearPendingIRQ(IRQn_Type IRQn)

Pobriše zahtevo po prekinitvi, če je slučajno prisotna

__STATIC_INLINE void NVIC_SetPriority(IRQn_Type IRQn, uint32_t priority)

Nastavi prioriteto prekinitve (0-15, 0 → najvišja med nastavljivimi)

PIO prekinitve

Vzporedna I/O vrata (PIOA, PIOB,...) imajo vsaka svoj prekinitveni signal in svojo ISR, ki mora ugotoviti, kateri pin je dejansko sprožil prekinitev

Funkcije (glej modul sam/drivers/pio)

void pio_enable_interrupt(Pio *p_pio, const uint32_t ul_mask)

p_pio: kazalec na strukturo z registri (npr. PIOC, glej sam3x8e.h)

ul_mask: maska pinov, za katere želimo omogočiti prekinitve

void pio_disable_interrupt(Pio *p_pio, const uint32_t ul_mask)

void pio_configure_interrupt(Pio *p_pio, const uint32_t ul_mask, const uint32_t ul_attr)

ul_attr: načini proženja prekinitev (npr. visoko/nizko stanje, fronta, ..., glej pio.h)

Za našo vajo → PIO_IT_FALL_EDGE

uint32_t pio_get_interrupt_status(const Pio *p_pio)

Vrne stanje prekinitev (statusni register), hkrati pobriše zastavice prekinitev I/O vrat

Potrebujemo, da ugotovimo, kateri pin je sprožil prekinitev

uint32_t pio_get_interrupt_mask(const Pio *p_pio)

Vrne masko vklopljenih prekinitev

PIO prekinitve

Prekinitvena rutina (ISR – interrupt service routine)

Deklaracije (prototipi) so že definirane za vse prekinitve (glej "sam3x8e.h")

Definicijo rutine bomo zapisali sami v "main.c"

```
void PIOC_Handler(){
…
}
```

ISR mora sama ugotoviti kateri pini vrat PIOC so sprožili zahtevo po prekinitvi

V sami ISR je potrebno pobrisati zastavice prekinitve I/O vrat

- Da lahko enota sproži naslednjo
- Različne periferne enote imajo različen način
- Za PIO vrata je potrebno prebrati statusni register