



Vhodno-izhodne naprave (VIN)

Predavanja

10. Povezovanje V/I naprav

Robert Rozman

rozman@fri.uni-lj.si

Tekoča obvestila

Programski jezik „C“ – viri

Dodani primeri projektov za STM32 :

- ▶ **CANBUS Sniffer, CANBUS IEX Modul (Cybrotech Avtomatizacija)**
 - ▶ STM32+Shield „Mikroelektronika“ ali
 - ▶ STM32+ CANBUS PHY
- ▶ **MODBUS Protokol Demo**
 - ▶ STM32 (VCOMPort) <—> PC (ModRSsim2 – Modbus Slave(server) simulator)
- ▶ Vabljen predavanja



Vsebina

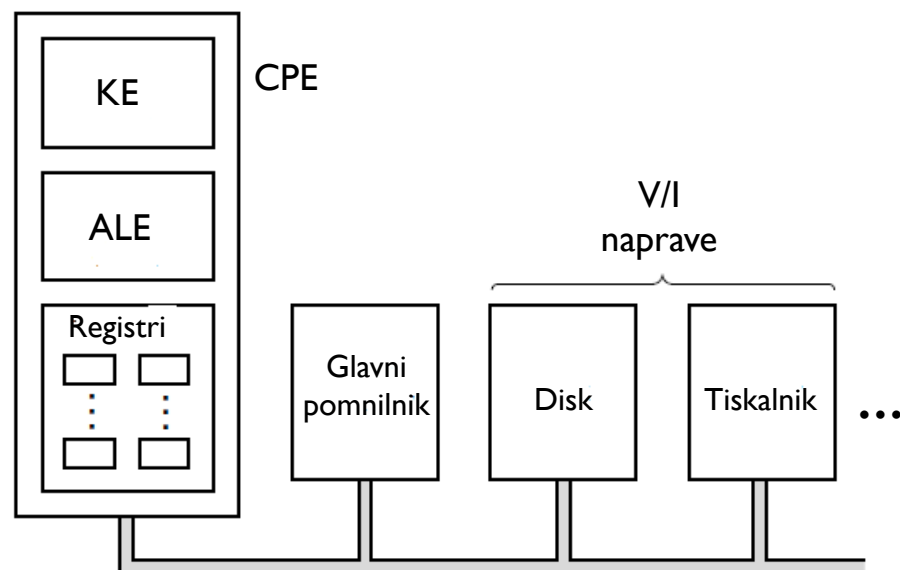
Uvod

1. V/I naprave in vodila
 1. Programski prenosi V/I (ang. programmed I/O-PIO)
 2. Neposreden dostop do pomnilnika (ang. Direct Memory Access – DMA)
 3. Krmilnik V/I naprave
2. Prekinitve, izjeme, pasti
3. Komunikacija računalnika z V/I napravami

Uvod

□ Vhodno/izhodni sistem (V/I sistem) sestavljajo

- V/I naprave
 - Uporaba: vhod – enkratno branje, izhod – samo pisanje, zunanji pomnilniki – večkratno branje in pisanje
 - Povezava: V/I naprava je povezana s človekom, V/I naprava je povezana z računalnikom, procesorjem
 - Hitrost: največja hitrost (b/s) s katero se prenašajo podatki med V/I napravo in procesorjem ali pomnilnikom
- Prenosne poti: vodilo, točka-v-točko
- Izvajanje V/I operacij
 - Programski vhod/izhod
 - Neposreden dostop do pomnilnika
- Krmilniki V/I naprav

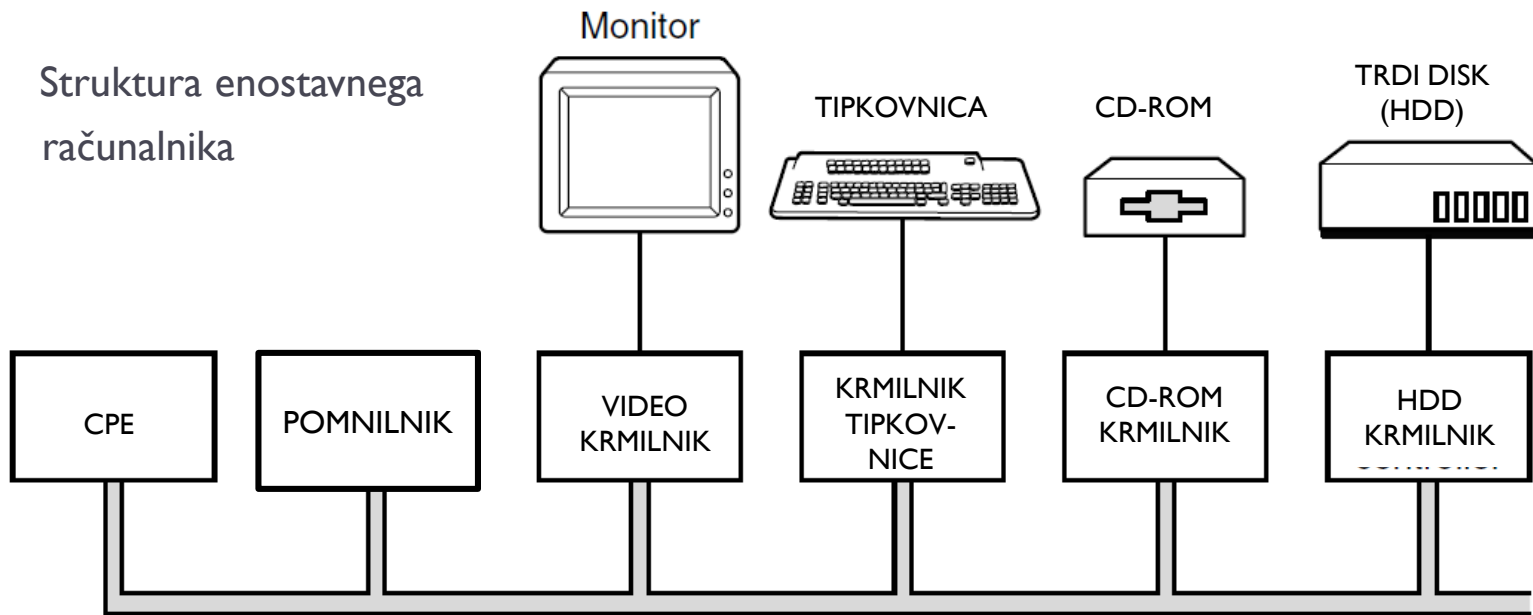


1. V/I naprave in vodila

□ Von Neumannov računalnik - organizacija s CPE in V/I napravami, kjer upoštevamo:

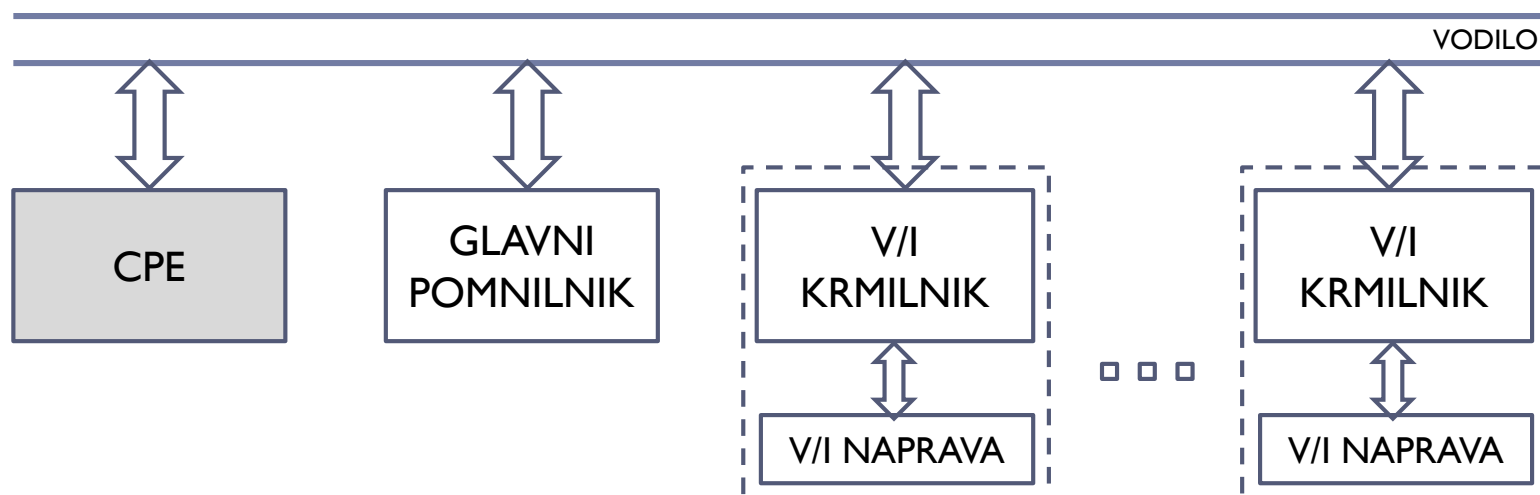
- V/I naprave za **pretvarjanje informacij** iz ene oblike v drugo (tipkovnica, miška, zaslon, tiskalnik ...).
- V/I naprave za **shranjevanje informacij** – pomožni pomnilniki (trdi disk, SSD, magnetni trak, DVD ...).
- **Način delovanja V/I sistema** - prenos podatkov med V/I sistemom, CPE in/ali glavnim pomnilnikom.

- Struktura enostavnega računalnika



1.1. Programski prenosi V/I (ang. programmed I/O-PIO)

- Prenos med CPE in v/I napravo je realiziran z ustreznim zaporedjem ukazov. Za prenos vsakega podatka je potrebno izvršiti več ukazov. **CPE izvaja program**, ki:
 - prične V/I operacijo,
 - nadzoruje njeno izvajanje,
 - prenaša podatke,
 - zaključi V/I operacijo.
- Operaciji, ki se uporabljata: branje iz V/I naprave in pisanje na V/I napravo



BRANJE iz V/I naprave preko V/I krmilnika

- Podatek se iz V/I naprave preko CPE zapiše v glavni pomnilnik

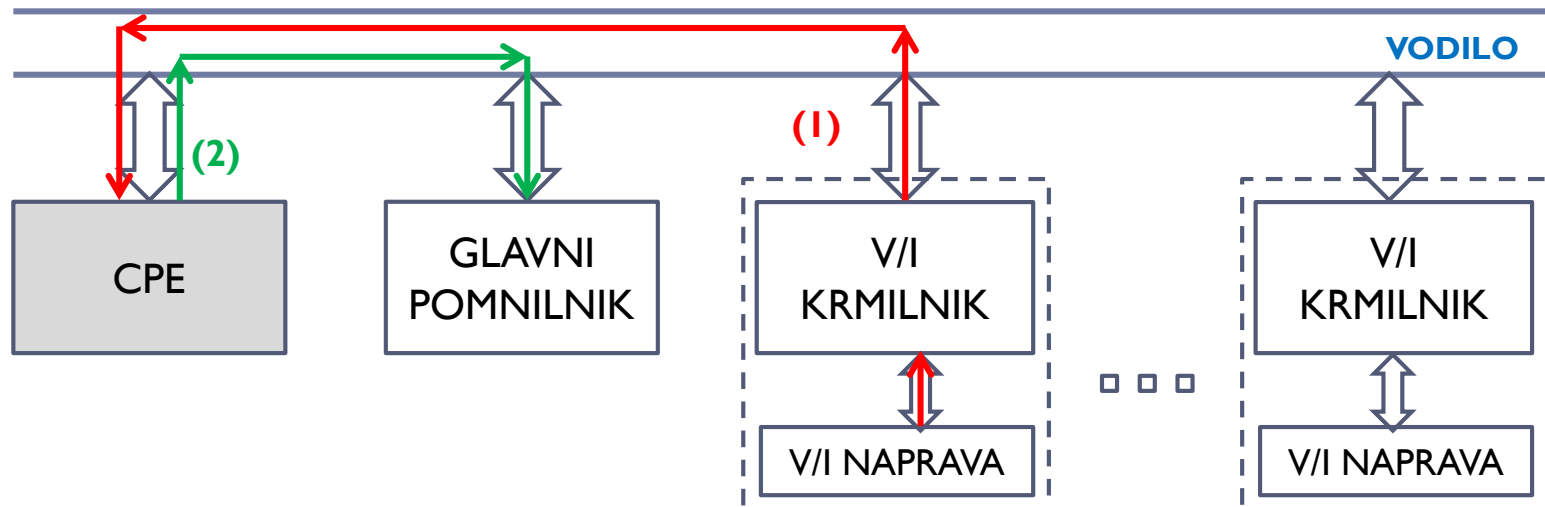
(V/I naprava → CPE → glavni pomnilnik)

(1) V/I enota ima shranjen podatek, ali ga pretvori iz ene oblike v drugo.

Preko V/I krmilnika se podatek pošlje na vodilo.

CPE sprejme podatek.

(2) CPE shrani podatek v glavni pomnilnik.



PISANJE ▾ V/I napravo preko V/I krmilnika

- Podatek se iz glavnega pomnilnika preko CPE pošlje V/I napravi.

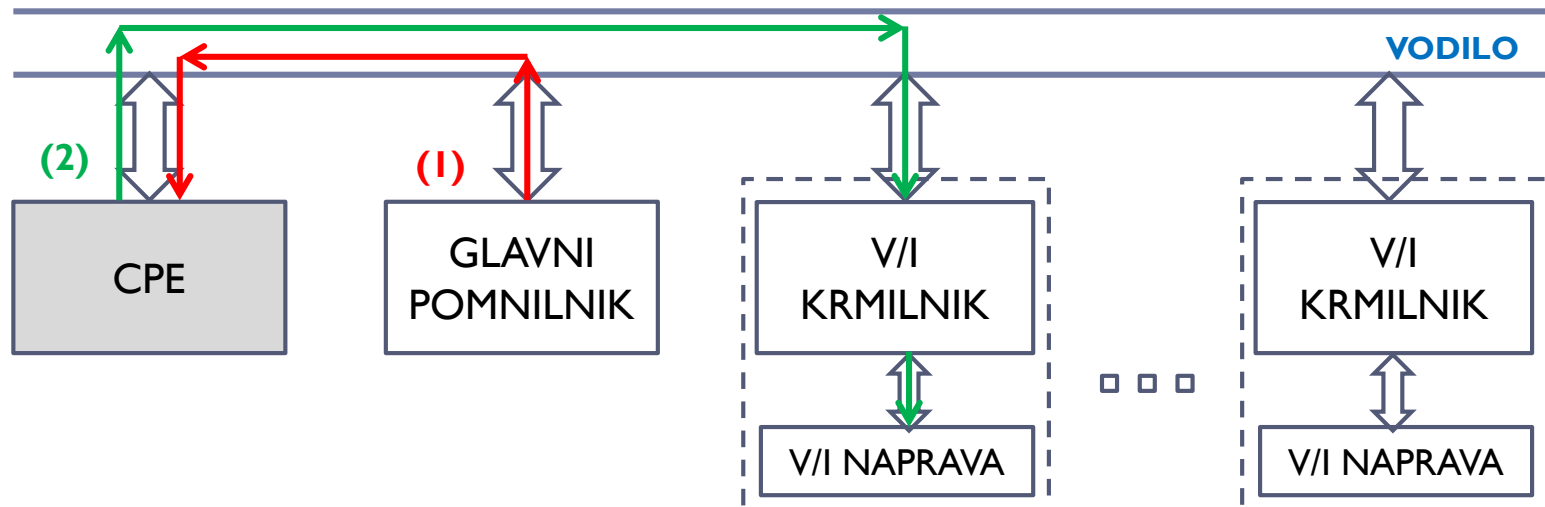
(glavni pomnilnik → CPE → V/I naprava)

(1) CPE prebere podatek iz glavnega pomnilnika.

(2) CPE pošlje podatek na vodilo.

V/I krmilnik sprejme podatek.

V/I naprava shrani podatek.



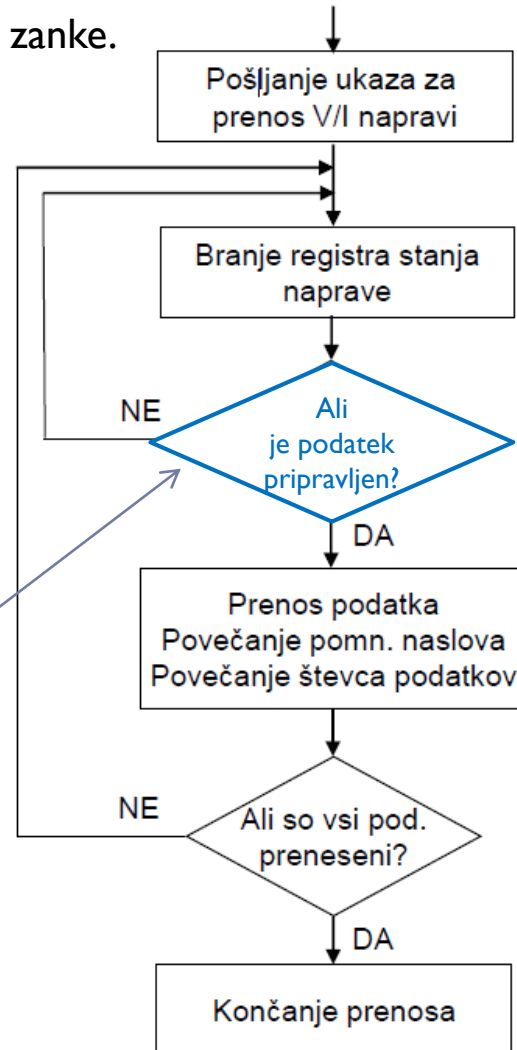
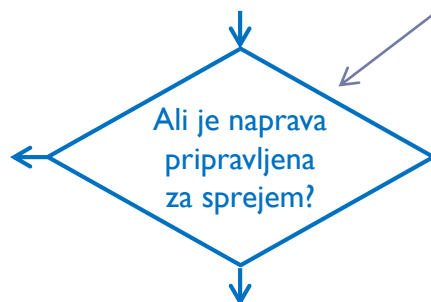
Ponazoritev programskega V/I prenosa (branje podatka iz V/I naprave)

❑ Prenos vsakega podatka zahteva obhod ene zanke.

❑ Programsko izpraševanje (ang. pooling) - preverjanje pripravljenosti naprave

❑ Postopek branja preverja:
Ali je podatek pripravljen?

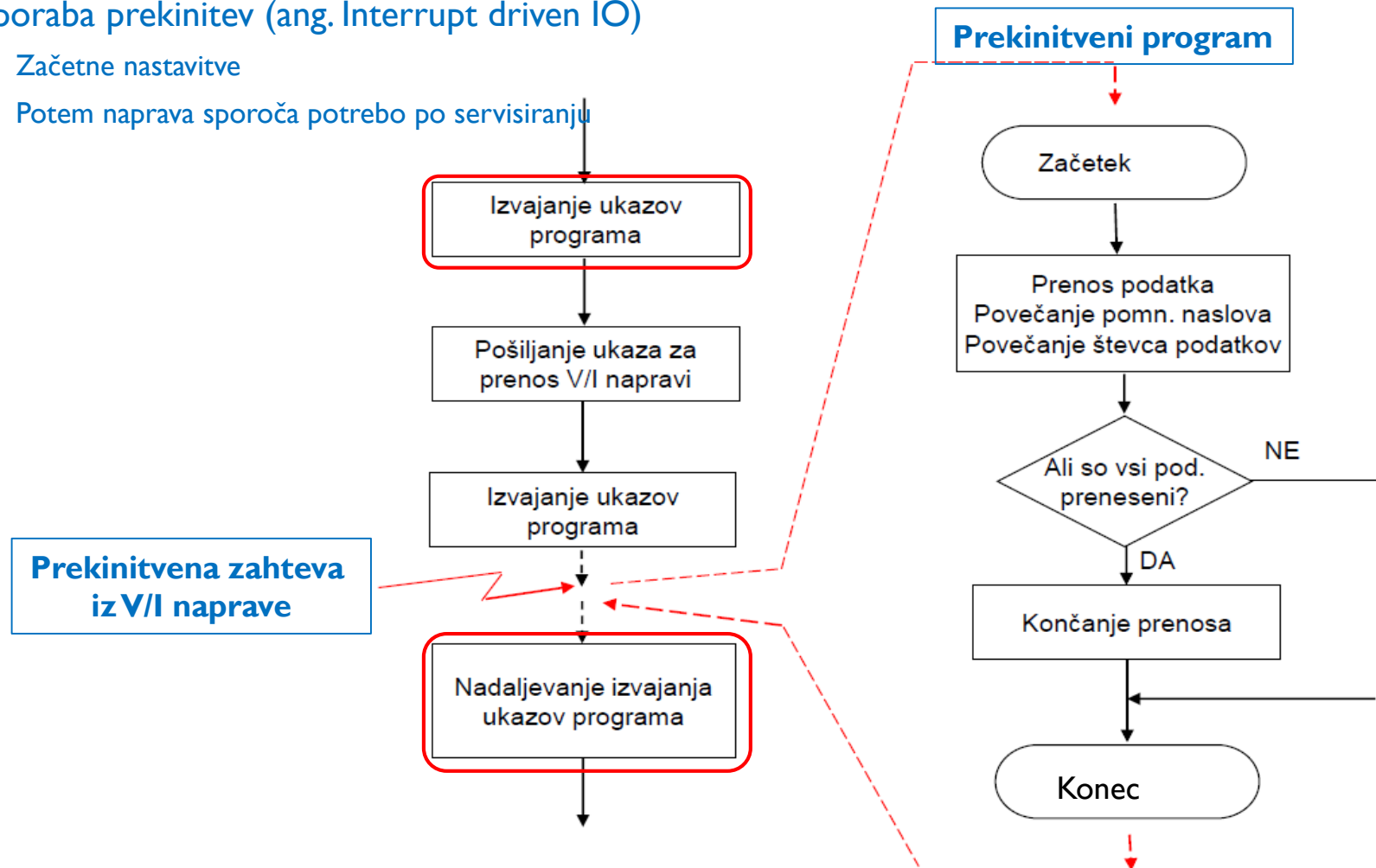
❑ Postopek pisanja se od branja razlikuje samo v preverjanju:



Ponazoritev programskega V/I prenosa (branje podatka iz V/I naprave)

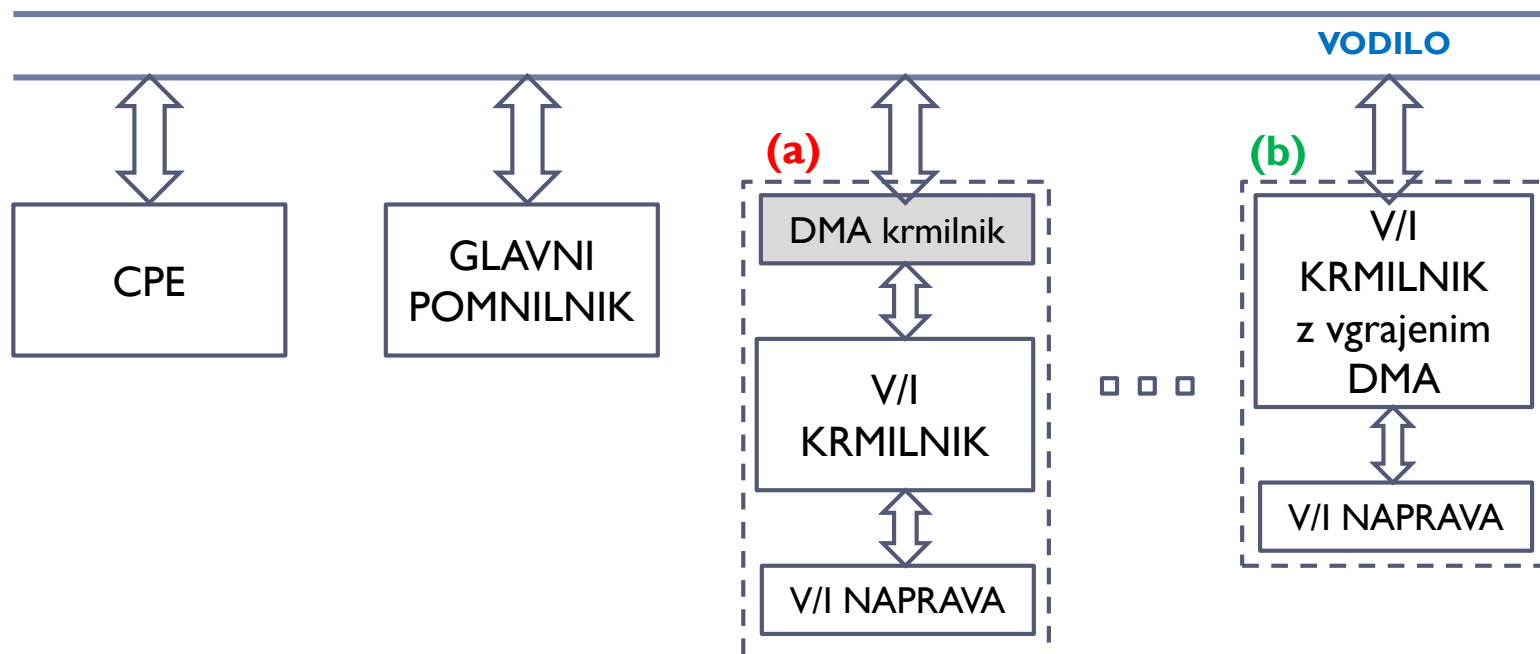
❑ Uporaba prekinitev (ang. Interrupt driven IO)

- ❑ Začetne nastavitve
- ❑ Potem naprava sporoča potrebo po servisiranju



1.2. DMA prenos (ang. Direct Memory Access)

- ❑ DMA predstavlja aparaturno rešitev za **neposreden dostop do pomnilnika**.
- ❑ Izvedba prenosa:
 - (a) V/I naprava ima vključena dva krmilnika (V/I krmilnik in DMA krmilnik)
 - (b) V/I naprava ima vključen V/I krmilnik, ki ima vgrajen DMA krmilnik

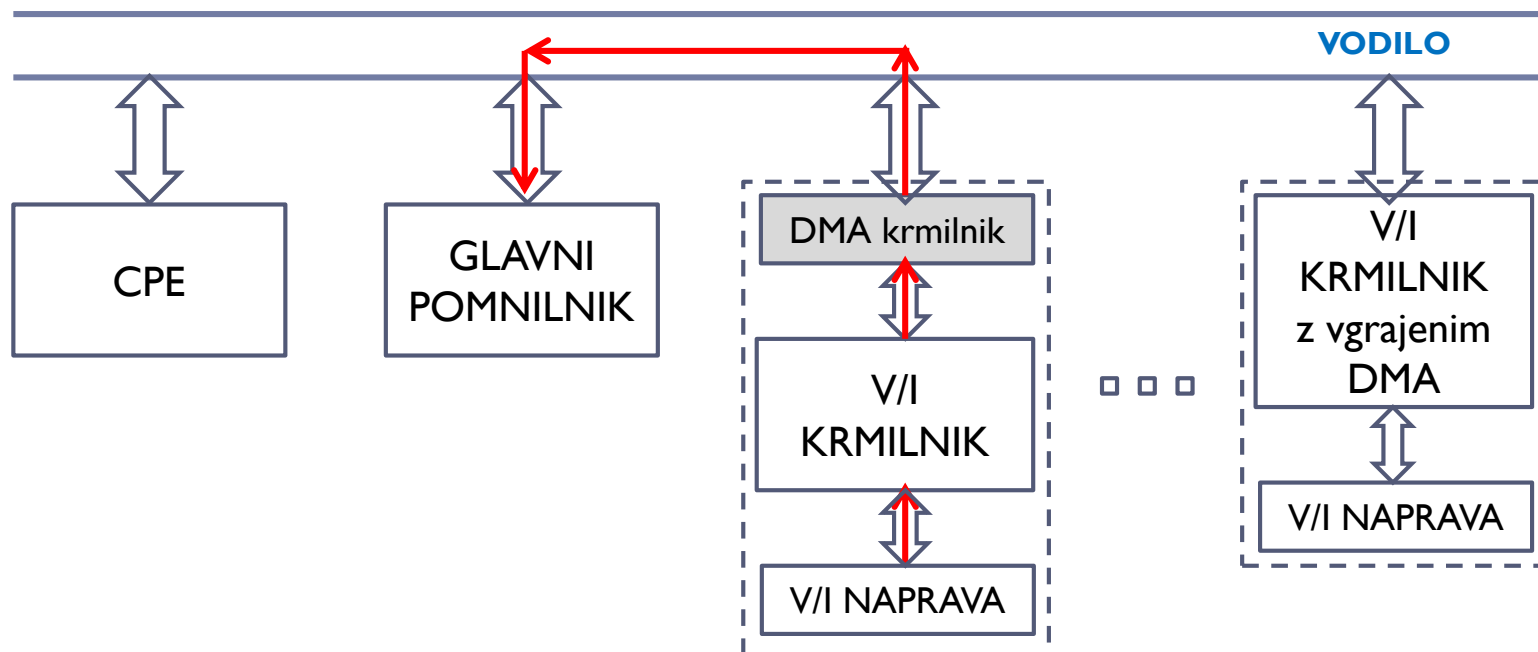


BRANJE iz V/I naprave z DMA krmilnikom

- Podatek se iz V/I naprave zapiše v glavni pomnilnik.

(V/I naprava → glavni pomnilnik)

- V/I enota ima shranjen podatek, ali ga pretvori iz ene oblike v drugo.
- Preko V/I krmilnika in DMA krmilnika se podatek pošlje na vodilo.
- Podatek se shrani v glavni pomnilnik.

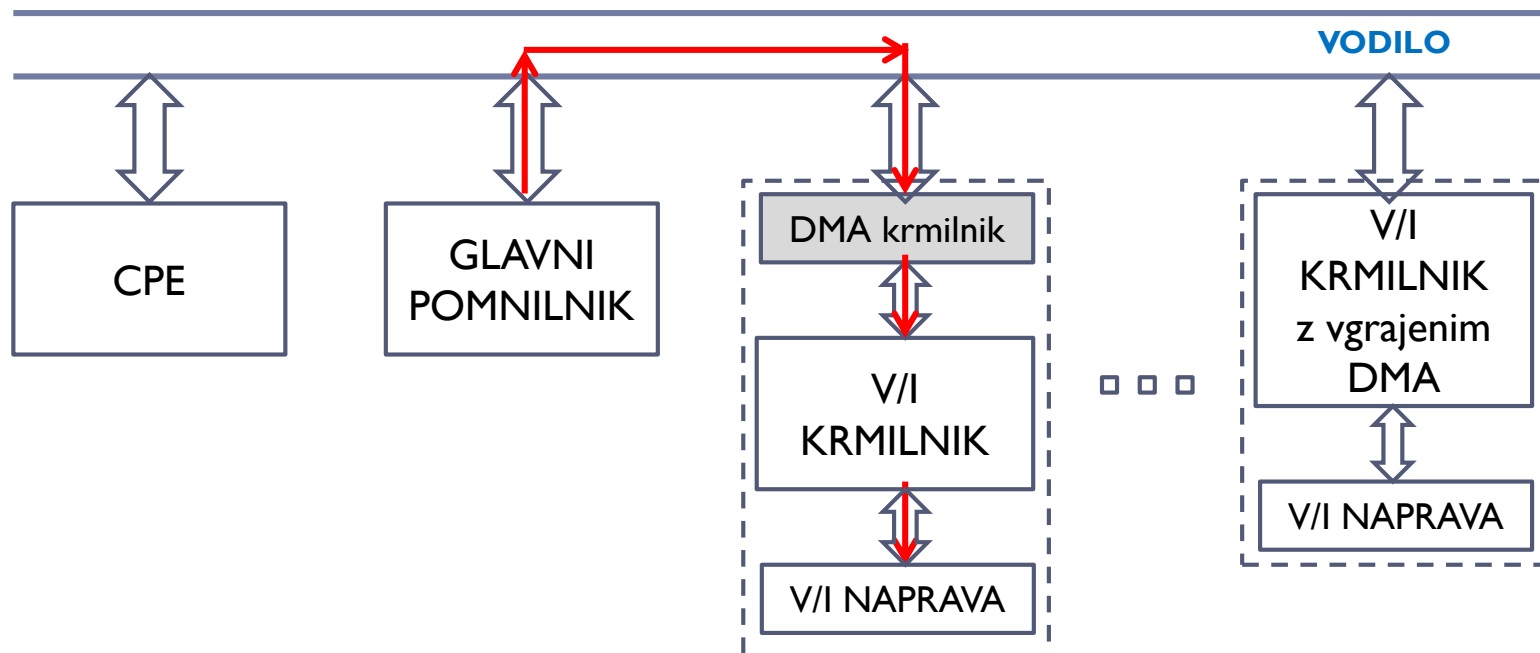


PISANJE v V/I napravo z DMA krmilnikom

- Podatek se iz glavnega pomnilnika pošlje V/I napravi.

(glavni pomnilnik → V/I naprava)

- Iz glavnega pomnilnika se pošlje podatek na vodilo.
- DMA krmilnik sprejme in posreduje podatek V/I krmilniku.
- V/I naprava shrani podatek.

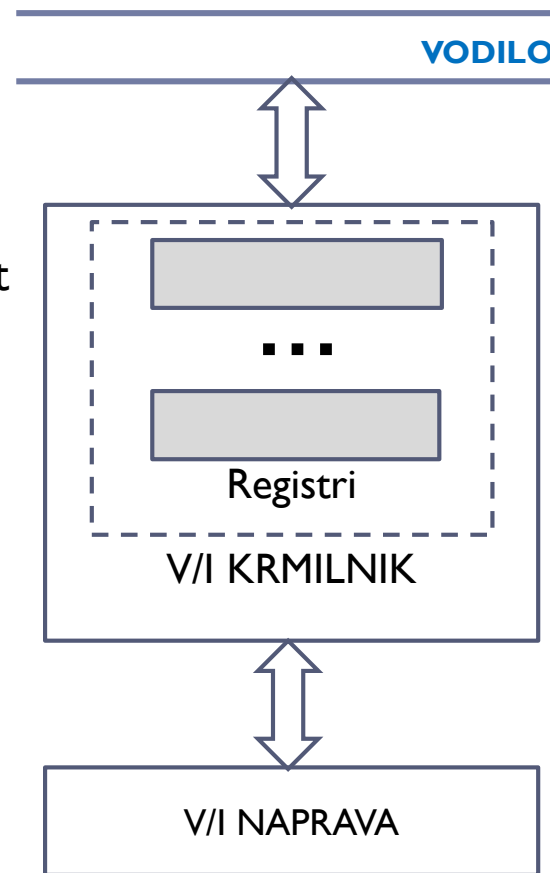


Povzetek DMA (Direct Memory Access)

- ❑ DMA omogoča **hiter prenos** velike količine podatkov.
- ❑ Podatki se med V/I napravo in pomnilnikom prenašajo **direktno, brez CPE**, če
 - obstaja povezava med krmilnikom V/I naprave in glavnim pomnilnikom in
 - obstaja krmilnik V/I naprav, ki ga imenujemo DMA krmilnik.
- ❑ DMA krmilnik:
 - tvori pomnilne naslove in kontrolne signale za dostop do pomnilnika,
 - šteje prenesene besede,
 - izvrši prenos, ko je podatek pripravljen,
 - obvesti CPE o zaključku V/I operacije (običajno s prekinitvijo).
- ❑ CPE izvede naslednje operacije:
 - v DMA krmilnik vpiše začetni naslov polja v glavnem pomnilniku.
 - v števeni register DMA krmilnika vpiše število besed, ki se prenesejo.
 - v DMA krmilnik vpiše vrednosti za način delovanja in operacije, ki se izvedejo.
 - v DMA krmilnik pošlje ukaz za začetek izvajanja prenosa.
 - preveri ali je bila operacija uspešno izvedena, ko DMA krmilnik pošlje potrditev.

1.3. Krmilnik V/I naprave

- ❑ Vsaka V/I naprava je priključena preko V/I krmilnika naprave.
- ❑ Gledano iz strani CPE je enostaven V/I krmilnik videti kot **določeno število namenskih registrov**.
- ❑ **Pisanje** v določen register (ukazni register) lahko sproži operacijo v V/I napravi (**ukaz napravi**).
- ❑ **Branje** iz določenega registra (statusni register) odraža stanje naprave po V/I operaciji (**status naprave**).
- ❑ **Prenos informacij** poteka z branjem iz določenega V/I registra ali s pisanjem v register V/I krmilnika (**podatkovni register**)
 - ❑ Krmilnik uravnava delovanje naprave glede na stanje bitov v registrih



- Registri v enostavnejšem V/I krmilniku so običajno dosegljivi na dva načina:

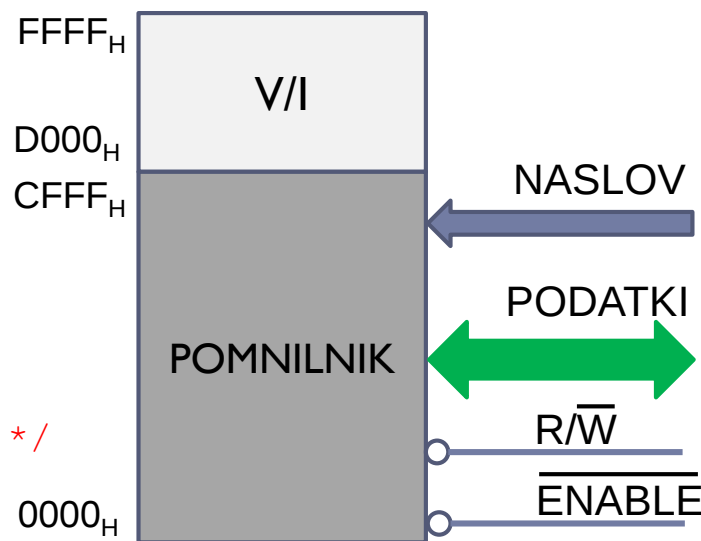
I. Pomnilniško preslikani vhod/izhod

- Registri krmilnika V/I so pomnilniške lokacije glavnega pomnilnika.
- Del pomnilniškega naslovnega prostora je uporabljen za V/I naprave.
- Uporabljajo se ukazi, ki imajo enega od operandov v pomnilniku.
- Naslovi V/I registrov - fizični naslovni prostor, ki se ne sme preslikati v predpomnilnik.
- Primer: Procesor ARM

- Naslovni prostor:

- Glavni pomnilnik: $0000_H - CFFF_H$
- Pomnilnik V/I: $D000_H - FFFF_H$

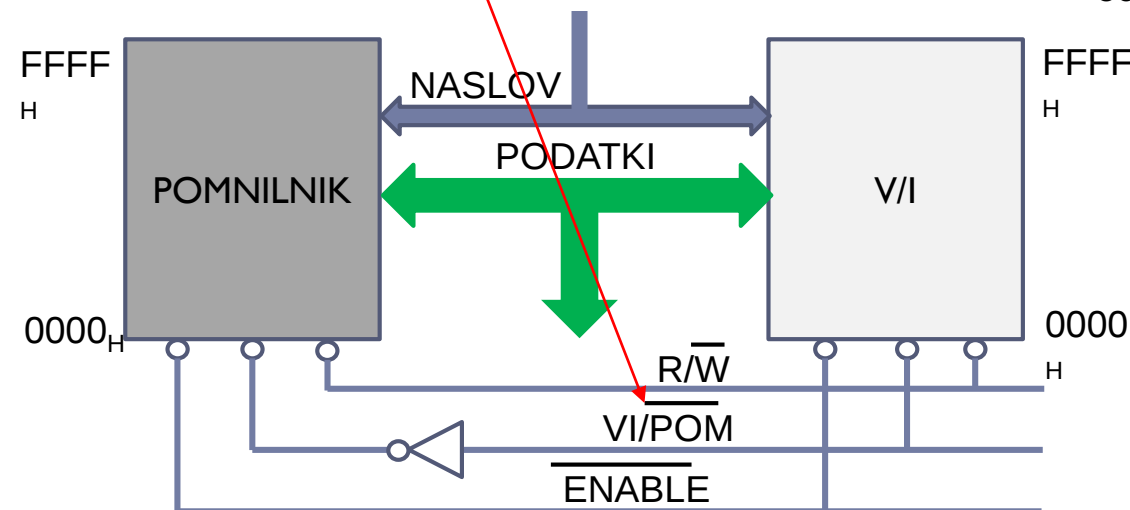
```
.equ AIC_BASE,      0xFFFFF000 /*naslov AIC */  
.equ AIC_SMR17,    0x044 /*Odmiki registrov v AIC */  
.equ AIC_SVR17,    0x0C4  
.equ AIC_IECR,     0x120  
.equ AIC_EOICR,    0x130
```



II. Ločen vhodno/izhodni naslovni prostor

- Registri krmilnika V/I so pomnilniške lokacije v ločenem naslovnem prostoru.
- Uporabljajo se posebni V/I ukazi.
- CPE aktivira posebne signale za dostop do V/I naprav (Read/Write, izbira, omogočanje).
- Primer: Procesorji INTEL

IN Read from a port
OUT Write to a port
INS/INSB Input string from port/Input byte string from port
INS/INSW Input string from port/Input word string from port
INS/INSD Input string from port/Input doubleword string from port
OUTS/OUTSB Output string to port/Output byte string to port
OUTS/OUTSW Output string to port/Output word string to port
OUTS/OUTSD Output string to port/Output doubleword string to port

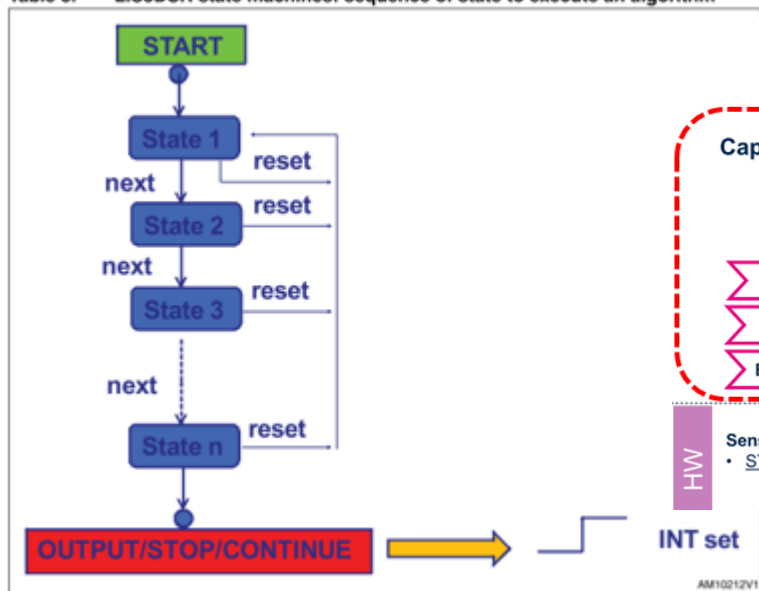


- Pogosto pa lahko dostopamo do registrov le posredno:

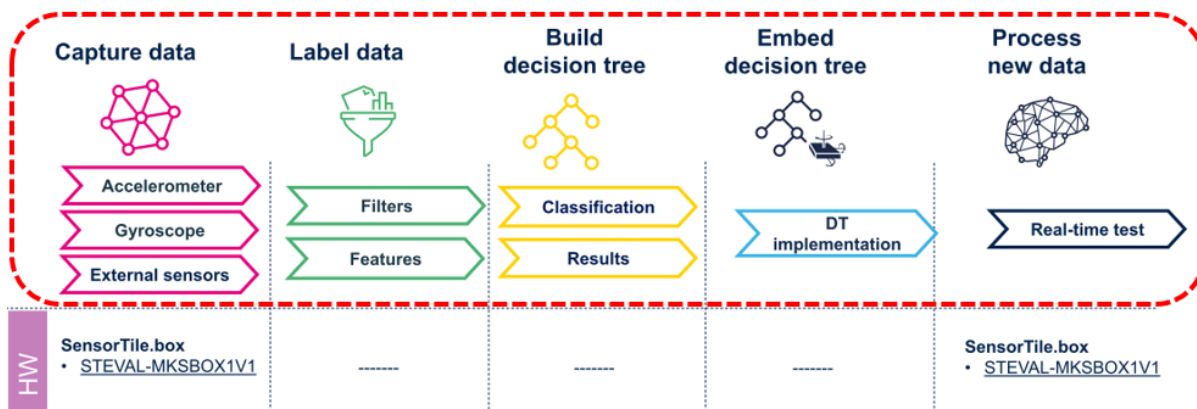
III. Posredno upravljanje (naslavljanje) preko V/I procesorjev, mikrokrmilnikov

- Naslovi registrov krmilnikov so lahko v posebnem naslovnem prostoru, do katerega imajo dostop samo V/I procesorji, mikrokrmilniki
- Pogosto v novejših V/I napravah procesorji/krmilniki omogočajo tudi naprednejše, višjenivojske funkcije in upravljamo napravo na tem nivoju.
 - Npr. končni avtomati, procesiranje podatkov, uporaba že naučenih modeli (plitke, globoke nevronske mreže in ostali koncepti s področja UI)

Table 8. LIS3DSH state machines: sequence of state to execute an algorithm



LSM6DSOX – SensorTile.box



- Ali pa so V/I naprave zunanje in dosegljive preko komunikacijskih poti (I2C, SPI, UART, ...):

IV. Posredno upravljanje (naslavljanje) preko komunikacijskih poti

- Registri so še vedno lahko dosegljivi v posebnem naslovnem prostoru
- Tipičen potek komunikacije za dostop do registrov:
 - **Izbira** (naslavljanje naprave)
 - **Naslavljanje registra** in izvedba operacije (branje, pisanje)
 - ,do katerega imajo dostop samo V/I procesorji, mikrokrmilniki

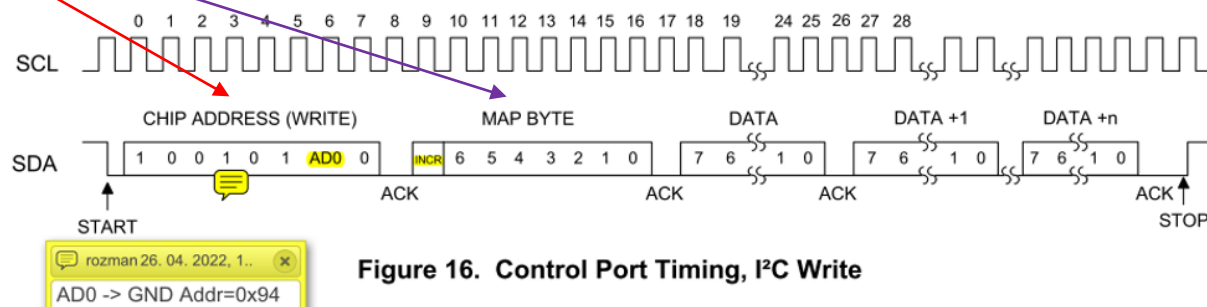
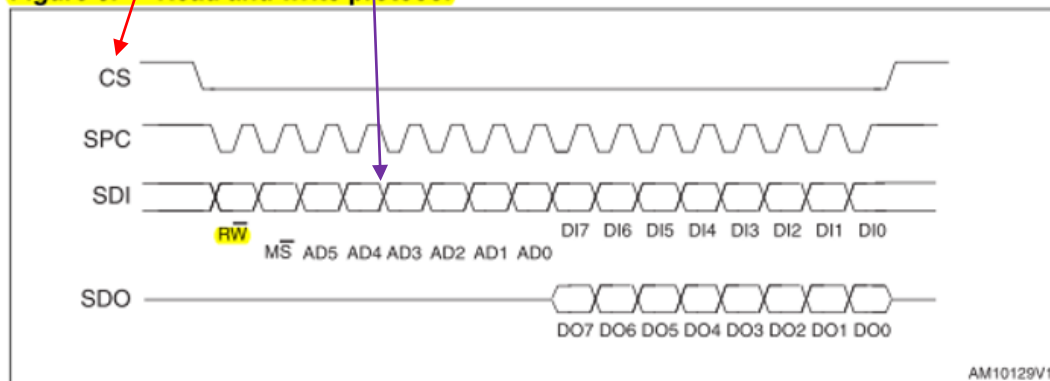


Figure 16. Control Port Timing, I²C Write

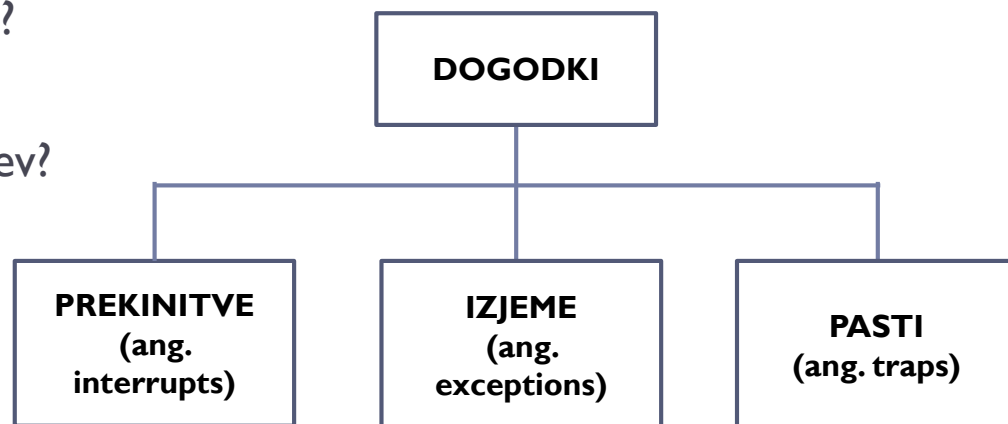
Figure 6. Read and write protocol



2. Prekinitve, pasti in izjeme

□ **Prekinitve** (ang. interrupt)

- Kaj je prekinitvev (ang. interrupt)?
- Kdo sproži prekinitvev?
- Kdaj se CPE odzove na prekinitvev?
- Kako se izvede prekinitvev?
- Kaj je prekinitveni program?
- Kaj je prioriteta prekinitvev?
- Kakšne rešitve obstajajo zanjo?



- **Past** (ang. trap) je prekinitvev, ki jo zahteva CPE ob posebnem dogodku ali na zahtevo programerja (ukazni dekodler zazna nepoznani ukaz, določena stran s podatki ni v fizičnem pomnilniku in CPE mora dostaviti podatke iz virtualnega pomnilnika, ...).
- **Izjema** (ang. exception) – prekinitvev, ki se avtomatsko generira v procesorju (aritmetična operacija – prekoračitev velikosti podatka ('overflow'), deljenje z 0, ...).

□ Prekinitev

- označuje **dogodek**, ki povzroči prenehanje izvajanja tekočega programa. Prične se izvajati drug program – **prekinitveni servisni program**.
- je **signal**, ki se pošlje procesorju, da se **prekine trenutni proces**. Lahko ga generira strojna naprava ali programska oprema.
- Potencialni izvori prekinitev in izvedba (shema desno)

Strojna oprema

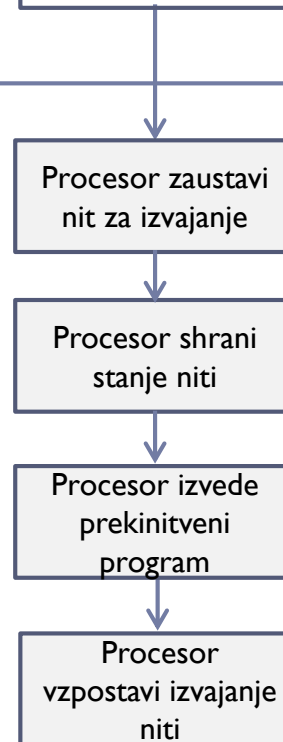
Zahteva za prekinitev je poslana procesorju iz V/I naprave

Procesor

Izjema / past je poslana od procesorja k procesorju

Programska oprema

Programski **prekinitveni ukaz** naloži procesor



❑ **Prioriteta** – pomemben je vrstni red obravnave zahtev

- Procesor ima več prekinitvenih vhodov in je nanj povezanih več V/I naprav.
- Prioriteta določa **vrstni red zahtev**, če je istočasno prisotnih več zahtev.
- Vgnezdene prekinitve – zahteva z višjo prioriteto lahko prekine izvedbo prekinitvenega programa naprave z nižjo prioriteto.
- Hitrejšje V/I naprave (trdi disk) imajo višjo prioriteto kot počasnejše (tipkovnica).

- **Prekinitveni krmilnik** ima modul za določanje prioritete (**prioritetni kodirnik**) pri izvedbi prekinitev, če ima CPE en sam bit za omogočanje prekinitev..

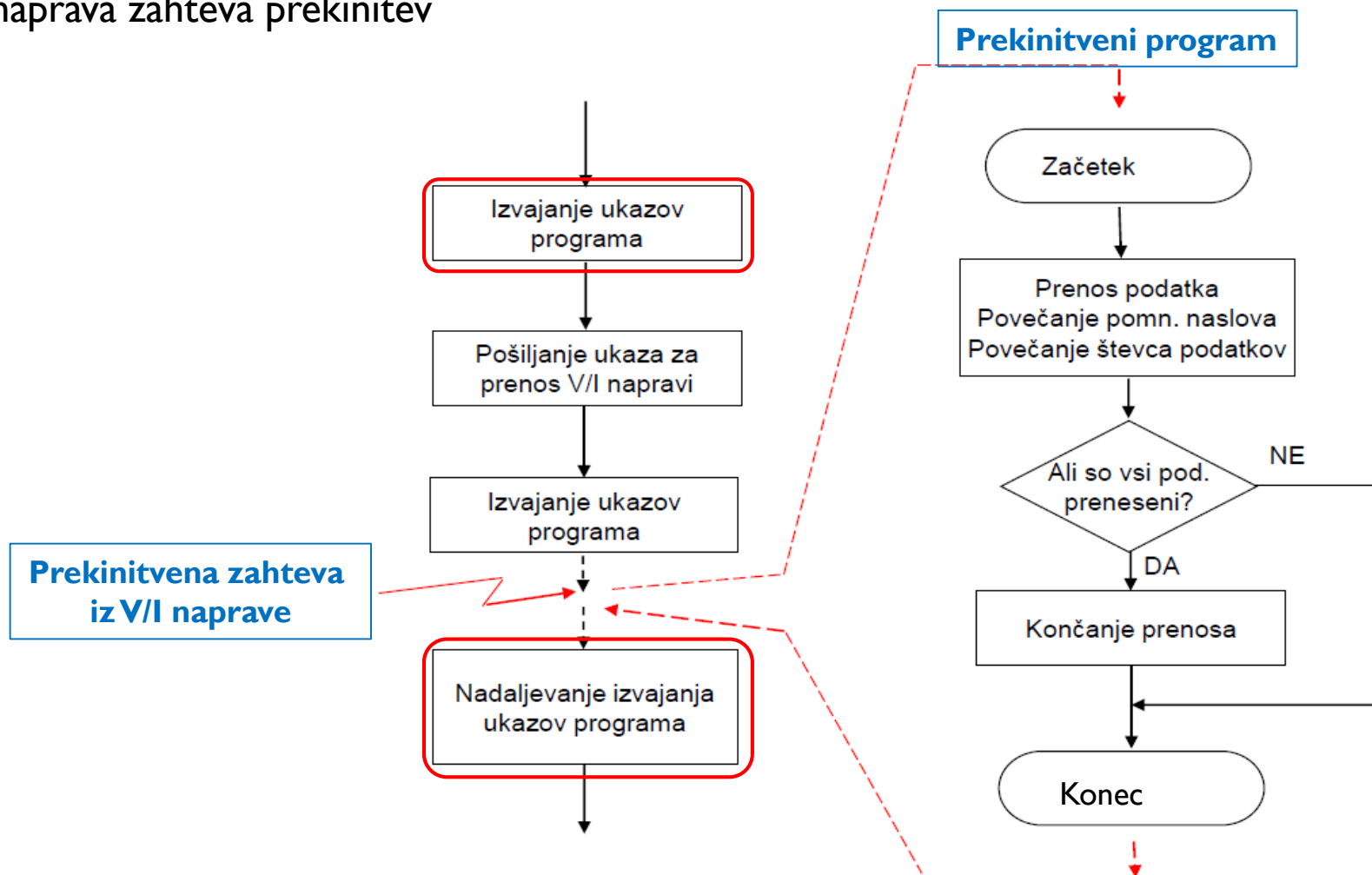
- Primer: 4-bitni prioritetni kodirnik
 - D_0 - najnižja prioriteta
 - V – veljaven indikator, ki je enak 1, če je en ali več vhodov enakih 1.

VHODI				IZHODI		
D_0	D_1	D_2	D_3	Y_1	Y_0	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

Izvedba prekinitve

- ❑ S prekinitvijo je CPE obveščena o spremembah, ko V/I naprava:
 - zahteva povezavo s CPE za izvedbo operacije.
 - konča določeno operacijo.
- ❑ Prekinitev je **asinhronska zahteva**, ki
 - ni povezana z nobenim ukazom.
 - ne prepreči izvedbe ukaza, ki se izvaja.
- ❑ CPE mora imeti **vgrajen prekinitveni mehanizem** za:
 - identifikacijo V/I naprave, ki je sprožila prekinitev.
 - izvedbo prioritete V/I naprav.
- ❑ Branje iz V/I naprave poteka v naslednjih korakih:
 - CPE pošlje napravi ukaz za prenos.
 - CPE nadaljuje z izvajanjem programa.
 - Ko je naprava pripravljena, pošlje v CPE zahtevo za prekinitev.
 - Vključi se prekinitveni program za branje podatkov iz V/I naprave.
 - CPE nadaljuje z izvajanjem programa.

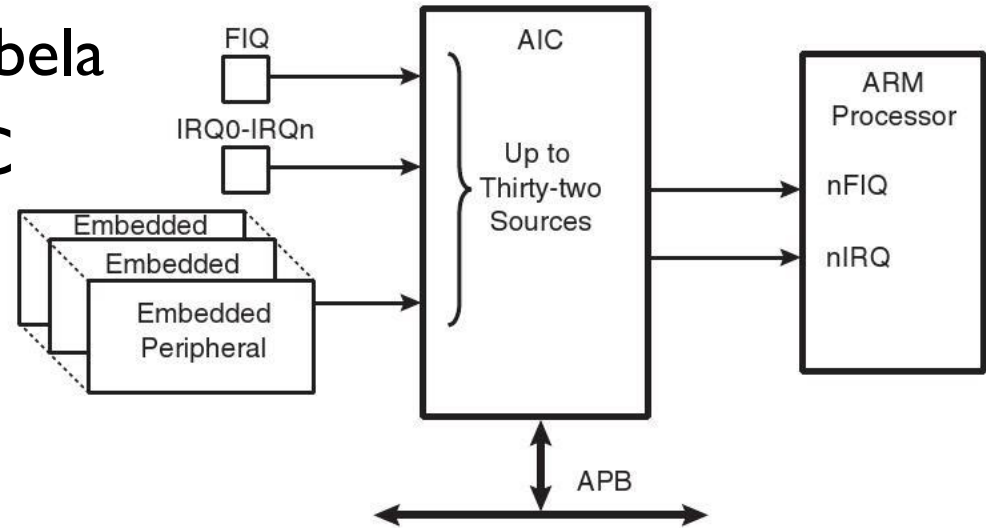
□ V/I naprava zahteva prekinitev



FRI-SMS: Prekinitveni krmilnik AIC (Advanced Interrupt Controller)

- ▶ Enostavna vektorska tabela
- ▶ Specifični vektorji v AIC

Prekinitve v / Izjema	Način delovanja	Vektor
Reset	svc	0x00000000
Undefined Inst.	und	0x00000004
SWI	svc	0x00000008
Prefetch Abort	abt	0x0000000C
Data Abort	abt	0x00000010
Reserved		0x00000014
IRQ	irq	0x00000018
FIQ	fiq	0x0000001C



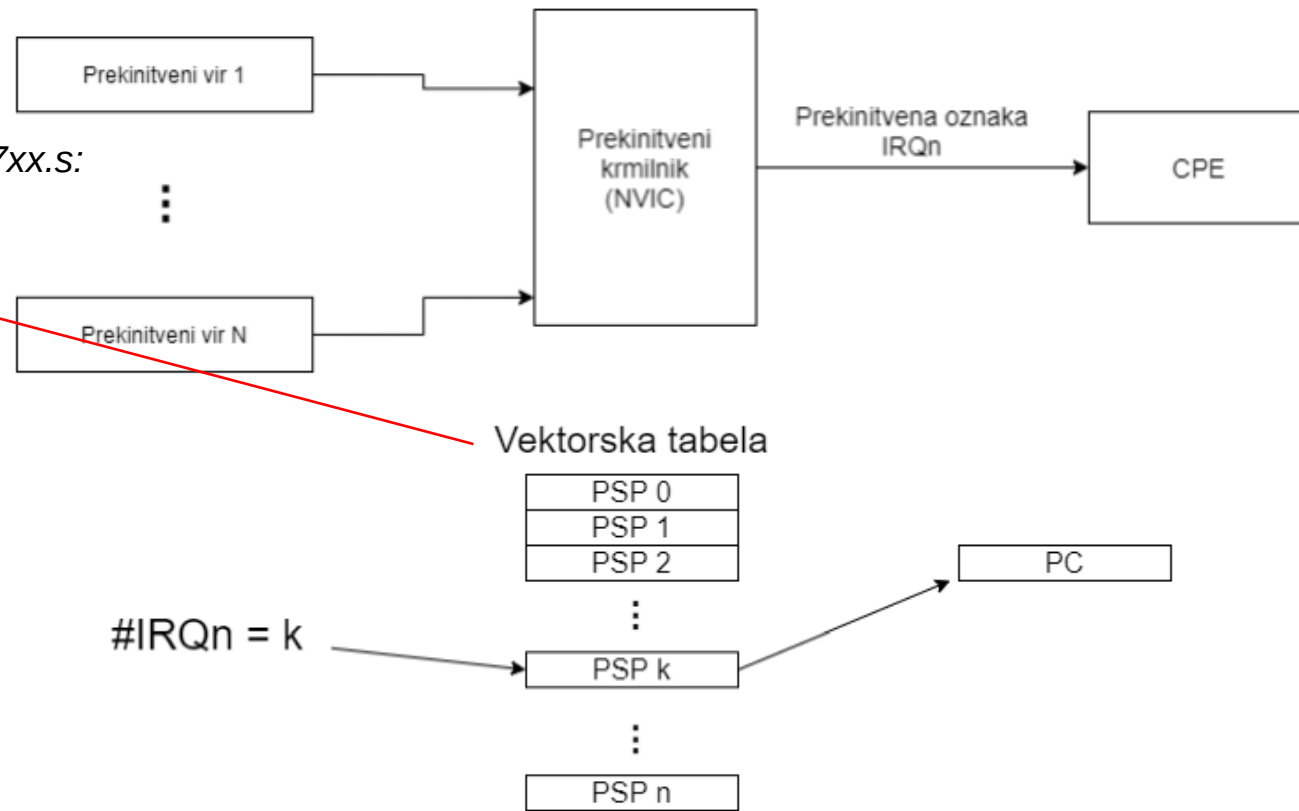
ARM: Prekinitveni krmilnik NVIC (Nested Vectored Interrupt Controller)

- ▶ Razširjena vektorska tabela
- ▶ Enostavnejša uporaba

`HAL_NVIC_SetPriority(EXTIO_IRQn, 1, 2);`
`HAL_NVIC_EnableIRQ(EXTIO_IRQn);`

Core/Startup/startup_stm32f407xx.s:

```
g_pfnVectors:
.word _estack
.word Reset_Handler
.word NMI_Handler
.word HardFault_Handler
.word MemManage_Handler
.word BusFault_Handler
.word UsageFault_Handler
...
.word FLASH_IRQHandler
.word RCC_IRQHandler
.word EXTI0_IRQHandler
.word EXTI1_IRQHandler
.word EXTI2_IRQHandler
...
.word TIM3_IRQHandler
```

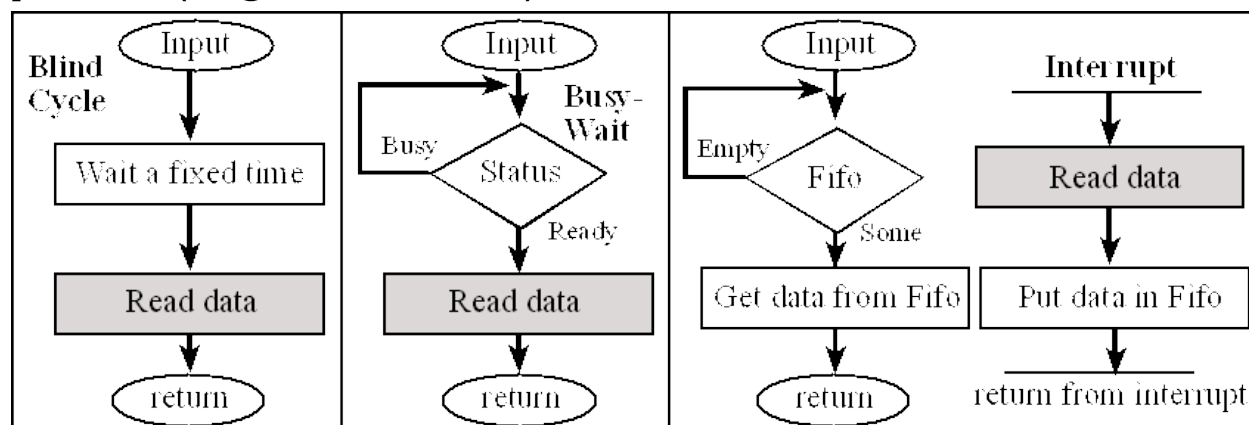


Vir: ORS - Bulić

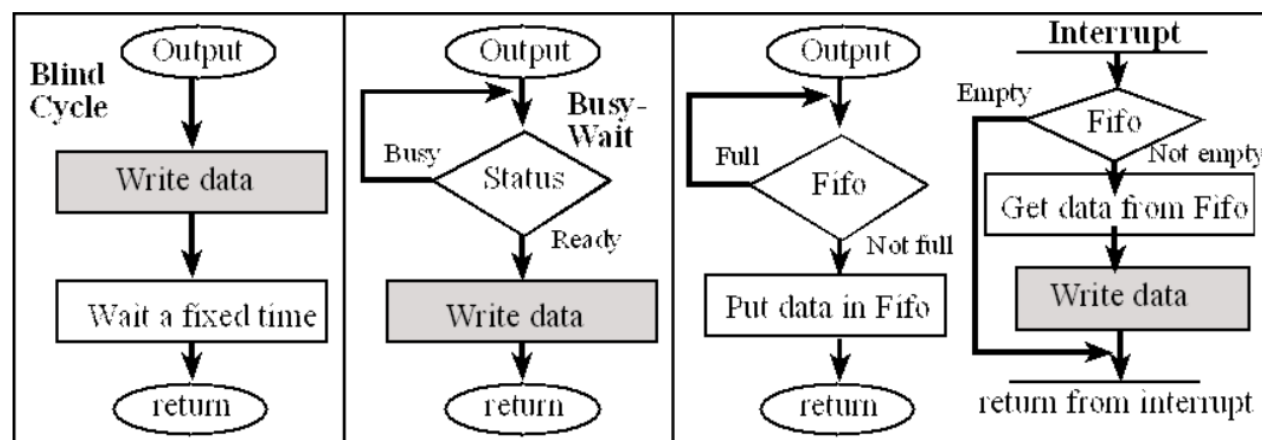
3. Komunikacija računalnika z V/I napravami

- ❑ Komunikacija med CPE in V/I napravami je običajno serijska
- ❑ Zmogljivost sistema (ang. performance) – mera za njegovo primernost, uporabnost
 - **Zakasnitev** (ang. latency) – čas med oddajo zahtevka in odzivom sistema.
 - **Pasovna širina** (ang. bandwidth) ali propustnost (ang. throughput) – št. prenesenih bitov/s.
 - **Prioriteta** (ang. priority) – vrstni red izvedbe, če je zaznanih več V/I zahtev.
 - **Sinhronizacijski mehanizmi** – strojna in programska oprema
- ❑ Potek programske komunikacije (sinhronizacije) s strojno opremo:
 - Slep cikl (ang. blind cycle) – npr, vklop LCD zaslona
 - Zasedeno-čakaj (ang. busy-wait) – npr. „is device busy“ ?
 - Prekinitev (ang. interrupt)
 - Programsko izpraševanje (ang. polling) – npr. „čakam na zastavico“
 - Neposreden dostop do pomnilnika (ang. DMA – Direct Memory Access)
- ❑ Vir: http://users.ece.utexas.edu/~valvano/Volume I/E-Book/C I I _SerialInterface.htm

❑ Branje iz V/I naprave (ang. Read data)



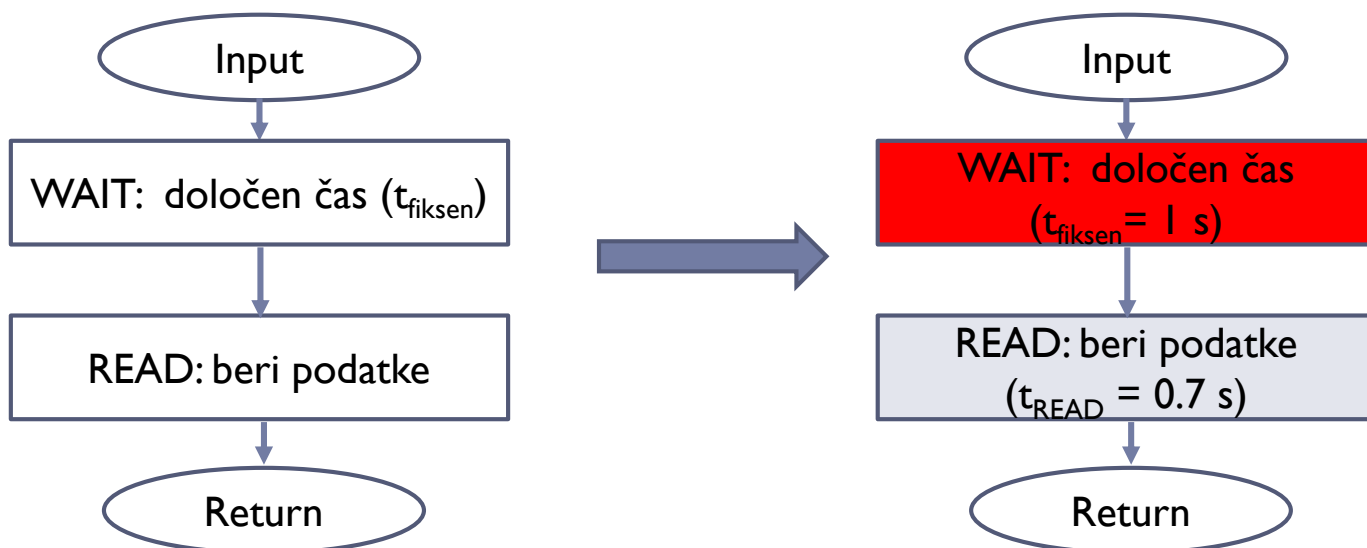
❑ Pisanje na V/I napravo (ang. Write data):



Branje iz V/I naprave

- ❑ **Slepi cikel** (ang. 'Blind Cycle'): Programska oprema za vhodno napravo:
 - sproži (zažene) zunanjo vhodno strojno opremo,
 - počaka določen čas in bere podatke iz naprave.
- ❑ Programska oprema počaka določen čas in predvideva, da se bo vhodni/izhodni postopek končal, preden poteče ta določen čas.
- ❑ Uporaba: za izvedbo operacije branja je potreben kratek in predvidljiv čas izvedbe.
- ❑ Program:

Izvedba: **Branje podatka iz V/I naprave v vnaprej določenem času**

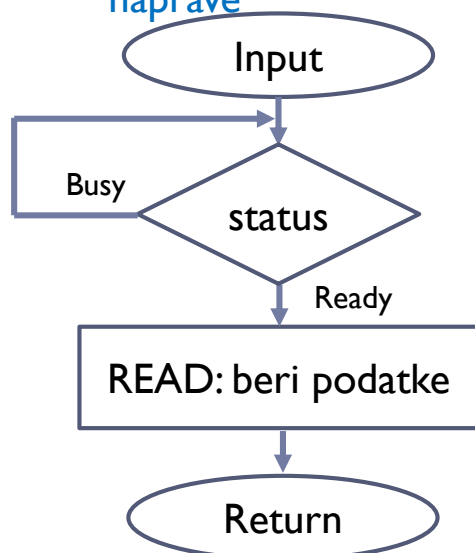


❑ Zasedeno-čakaj (ang. 'Busy-Wait')

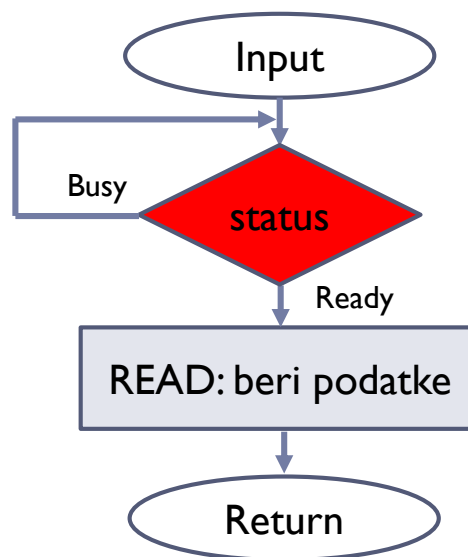
- programska zanka, ki preverja stanje V/I naprave, čaka na zaključeno stanje.
- Programska oprema za vhodno napravo počaka, da ta dobi nove podatke in jih nato prebere z vhodne naprave.

❑ Uporaba: Sinhronizacija je primerna v situacijah, ko je programski sistem razmeroma preprost in odziv v realnem času ni pomemben.

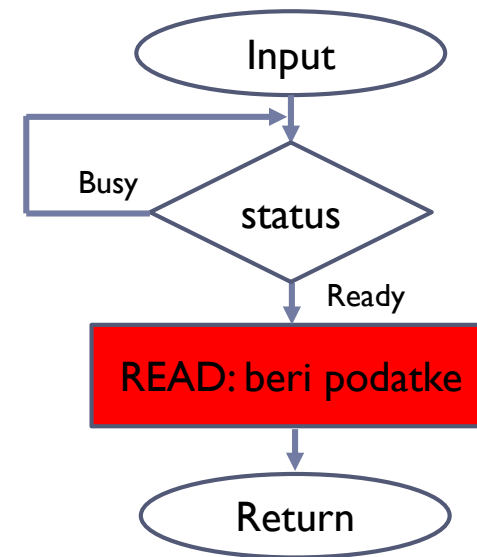
▪ Program:
naprave



Izvedba: Preverjanje zasedenosti



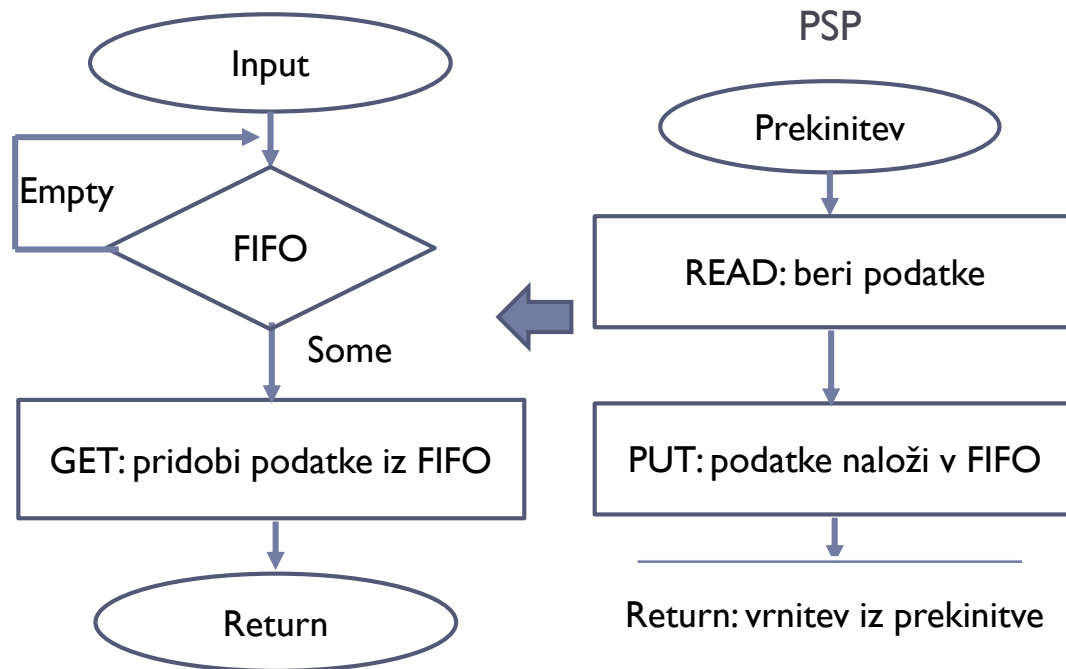
Branje podatka iz V/I



❑ Prekinitev (ang. Interrupt):

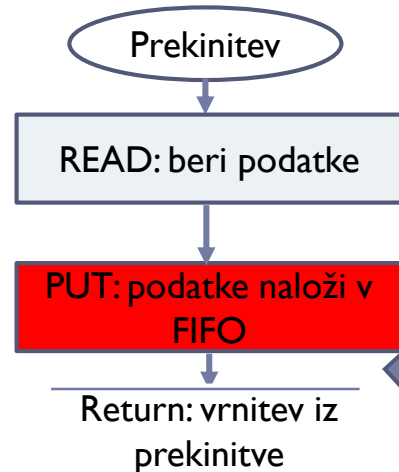
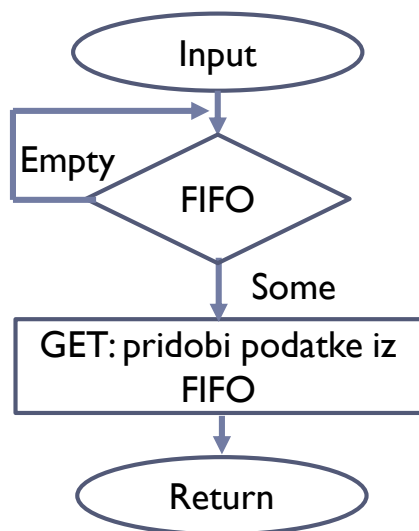
- Strojna oprema povzroči izvajanje posebne programske opreme (prekinitveni program).
- Strojna oprema vhodne naprave zahteva prekinitev, ko ima vhodna naprava nove podatke v FIFO.
- Prekinitev programske opreme bo prebrana z vhodne naprave in shranjena v globalnem RAM-u.

▪ Program

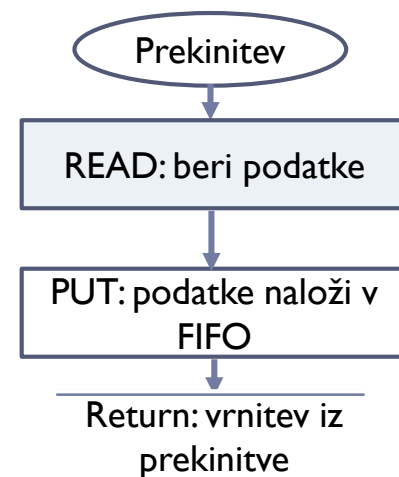
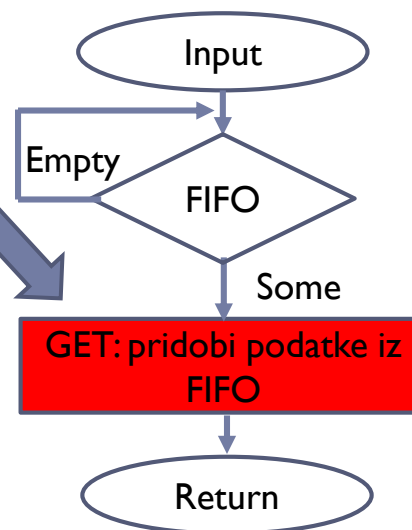


- Izvedba

V/I naprava:
nalaga podatke v FIFO in
sproži prekinitev



Program v računalniku pridobiva
podatke iz FIFO



Primer: STM32 – ADC na 3 načine

❑ Programsko spraševanje (angl. Software Polling)

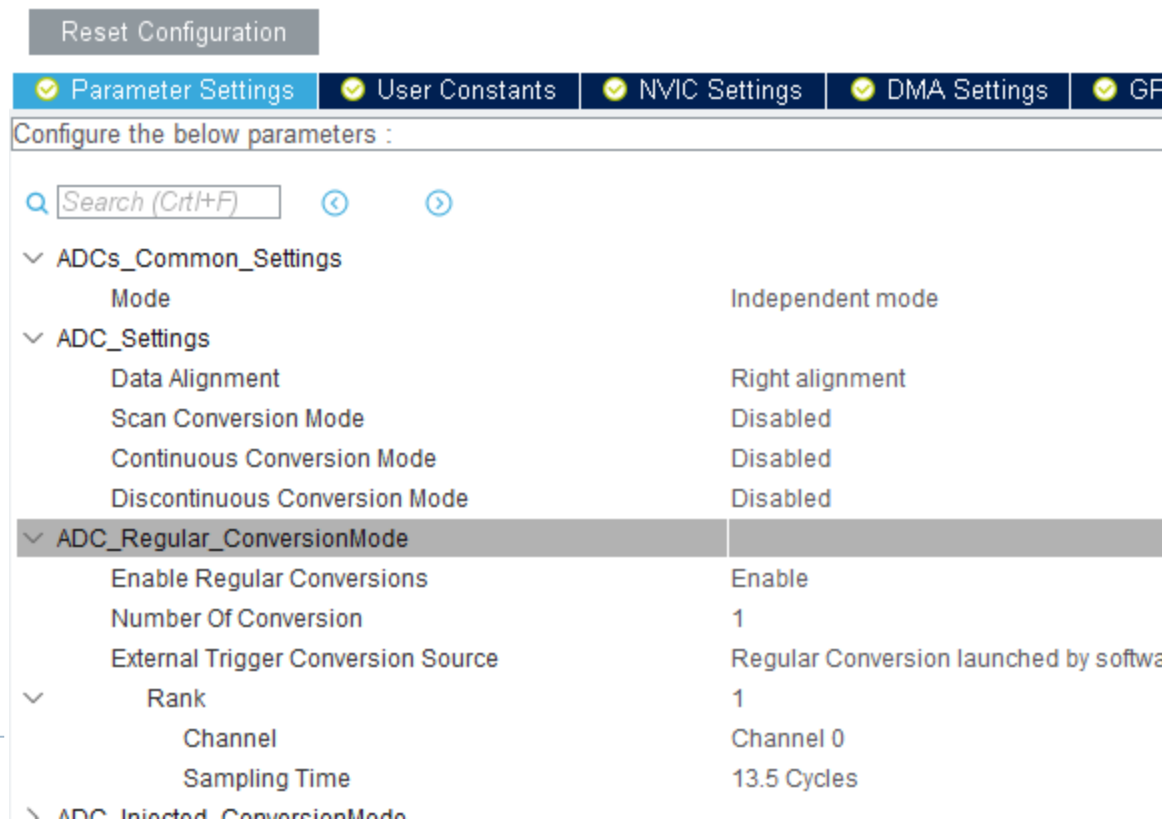
```
HAL_ADC_Start(&hadc1); // start the adc
```

```
HAL_ADC_PollForConversion(&hadc1, 100); // poll for conversion
```

```
adc_val = HAL_ADC_GetValue(&hadc1); // get the adc value
```

```
HAL_ADC_Stop(&hadc1); // stop adc
```

```
HAL_Delay(500); // wait for 500ms
```

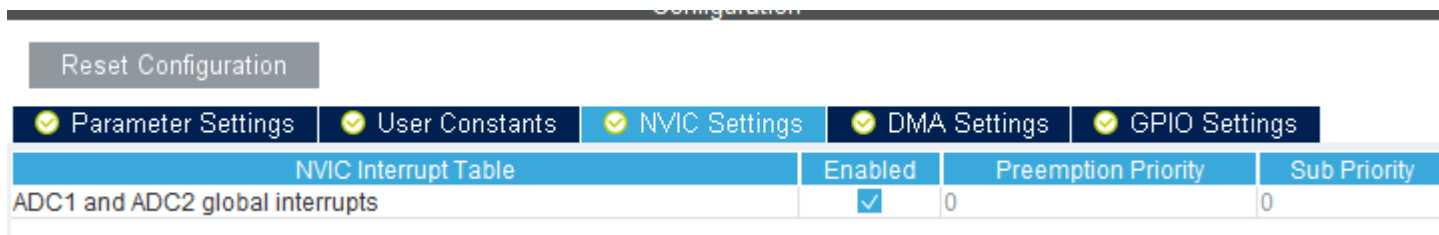
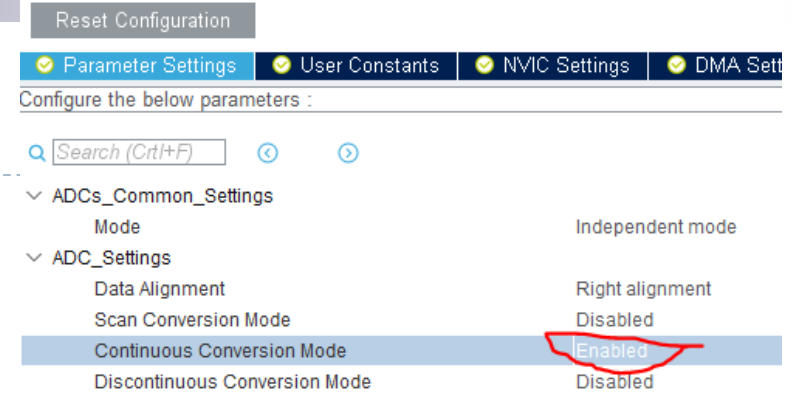


Primer: STM32 – ADC na 3 načine

❑ Prekinitve (angl. Interrupts)

`HAL_ADC_Start_IT (&hadc1);`

```
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc) {  
    adc_val = HAL_ADC_GetValue(&hadc1);  
    /*If continuousconversion mode is DISABLED uncomment below*/  
    //HAL_ADC_Start_IT (&hadc1);  
}
```



Primer: STM32 – ADC na 3 načine

❑ DMA (angl. Direct Memory Access)

`HAL_ADC_Start_DMA (&hadc1, &buffer, 1);`

```
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc) {  
    adc_val = buffer;  
}
```

DMA Request Settings

Mode: Circular

Increment Address: ☐

Peripheral: ☐ Memory: ☒

Data Width: Word Word

Configuration			
Reset Configuration			
✓ Parameter Settings	✓ User Constants	✓ NVIC Settings	✓ DMA Settings
✓ GPIO Settings			
DMA Request	Channel	Direction	Priority
ADC1	DMA1 Channel 1	Peripheral To Memory	Low

Primer: STM32 – UART na 3 načine

□ Programsko spraševanje (angl. Software Polling)

```
uint8_t Rx_data[10]; // creating a buffer of 10 bytes
```

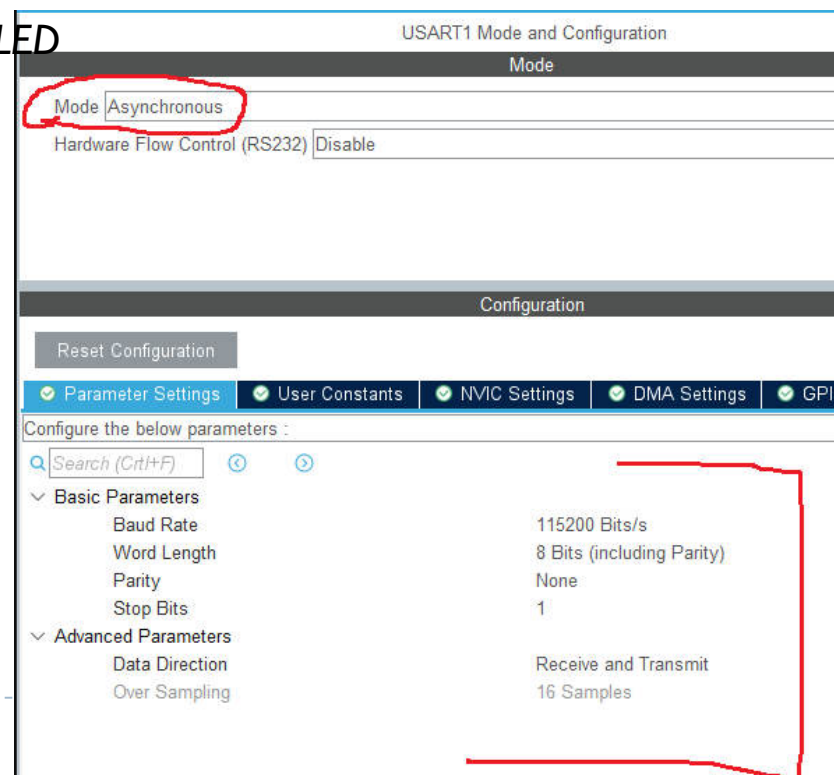
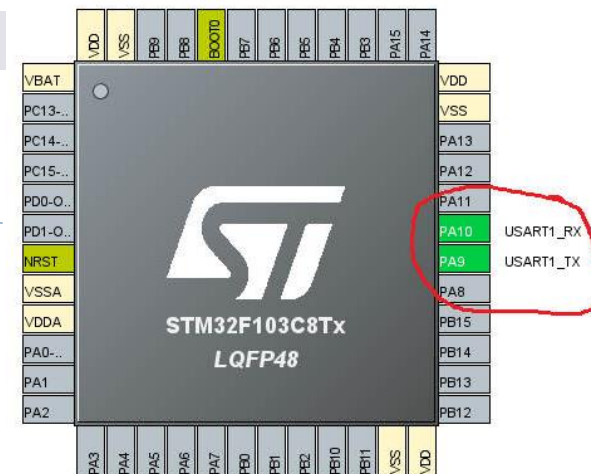
```
while (1) {
```

```
    HAL_UART_Receive (&huart2, Rx_data, 4, 100); // receive 4 bytes of data
```

```
    HAL_GPIO_TogglePin (GPIOA, GPIO_PIN_5); // toggle LED
```

```
    HAL_Delay (250);
```

```
}
```



Primer: STM32 – UART na 3 načine

❑ Prekinitve (angl. Interrupts)

....

```
uint8_t Rx_data[10]; // creating a buffer of 10 bytes
```

```
void HAL_UART_RxCpltCallback(UART_HandleTypeDef  
*huart)
```

```
{
```

```
HAL_UART_Receive_IT (&huart2, Rx_data, 4);
```

```
while (1)
```

```
{
```

```
HAL_GPIO_TogglePin (GPIOA, GPIO_PIN_5);
```

```
HAL_Delay (250);
```

```
}
```

USART1 Mode and Configuration

Mode

Mode Asynchronous

Hardware Flow Control (RS232) Disable

Configuration

Reset Configuration

✔ Parameter Settings ✔ User Constants ✔ NVIC Settings ✔ DMA Settings ✔ GPIO Settings

NVIC Interrupt Table

Enabled

Preemption Priority

Sub Priority

USART1 global interrupt

✔

0

0

Primer: STM32 – UART na 3 načine

❑ DMA (angl. Direct Memory Access)

```
....  
uint8_t Rx_data[10]; // creating a buffer of 10 bytes  
void HAL_UART_RxHalfCpltCallback(UART_HandleTypeDef  
*huart) {  
    HAL_GPIO_TogglePin (GPIOA, GPIO_PIN_0); // toggle PA0  
}  
void HAL_UART_RxCpltCallback(UART_HandleTypeDef*huart)  
{  
    HAL_UART_Receive_DMA(&huart2, Rx_data, 4);  
}
```

```
int main ()
```

```
.....  
    HAL_UART_Receive_DMA (&huart2, Rx_data, 4); // Receive 4  
    Bytes of data
```

```
while (1)  
{  
    HAL_GPIO_TogglePin (GPIOA, GPIO_PIN_5);  
    HAL_Delay (250);  
}
```

Add Delete

DMA Request Settings

Mode	Circular	Increment Address	☐	Peripheral		Memory	☒
		Data Width	Byte			Byte	

Configuration

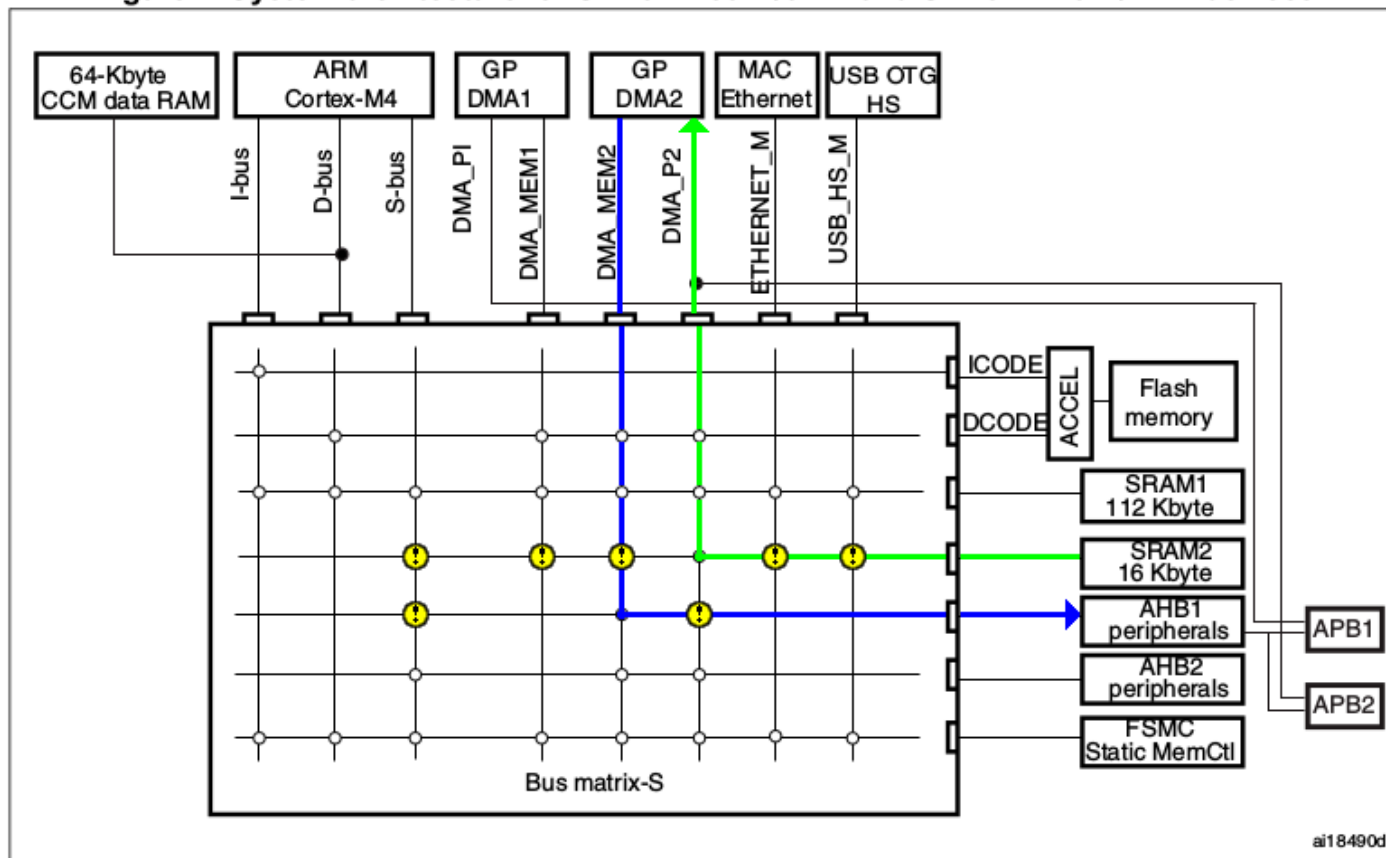
Reset Configuration

Parameter Settings User Constants NVIC Settings **DMA Settings** GPIO Settings

DMA Request	Channel	Direction	
USART1_RX	DMA1 Channel 5	Peripheral To Memory	Low

Primer: STM32 – Interna zgradba – stikalna matrika

Figure 1. System architecture for STM32F405xx/07xx and STM32F415xx/17xx devices



<http://cliffle.com/blog/glitch-in-the-matrix/>

<https://github.com/cbiffle/m4vgalib/>

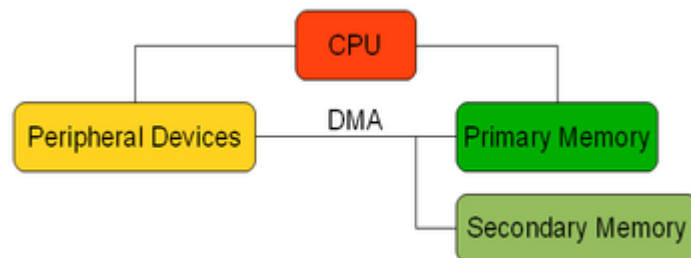
Primer: Analiza delovanja datotečnega sistema

When Poll is Better than Interrupt
Jisoo Yang Dave B. Minton Frank Hady
{jisoo.yang | dave.b.minton | frank.hady} (at) intel.com
Intel Corporation

Performance Summary

Method	Latency (μ s)	Throughput (40 KB/sec)	CPU Utilization (%)
Naive	100	10	5
Batching	~ 1000	21	10.5
Interrupts	106	~ 18	8.9
DMA + Interrupt	61	91	45
DMA + Polling	56	167	84

A **file system** is a data structure which is used to store, retrieve and update a set of files. It can live in memory or secondary storage, so when designing this data structure we need to do it with an eye to speed. We want good performance from our file systems.



Metrics for Performance

Latency*	Response time. Delay from particular request to response.
Throughput*	Rate of request completion (I/O's/sec).
Utilization (closely related to throughput)	Fraction of time that the I/O system is actually doing I/O, can get as close as you can to 100%.

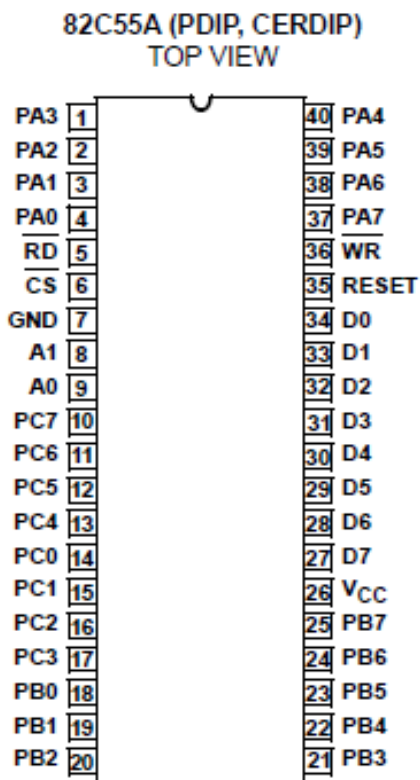
<http://web.cs.ucla.edu/classes/spring13/cs111/scribe/10b/>

Primer: Povezava zaslona in tipkovnice na V/I krmilnik

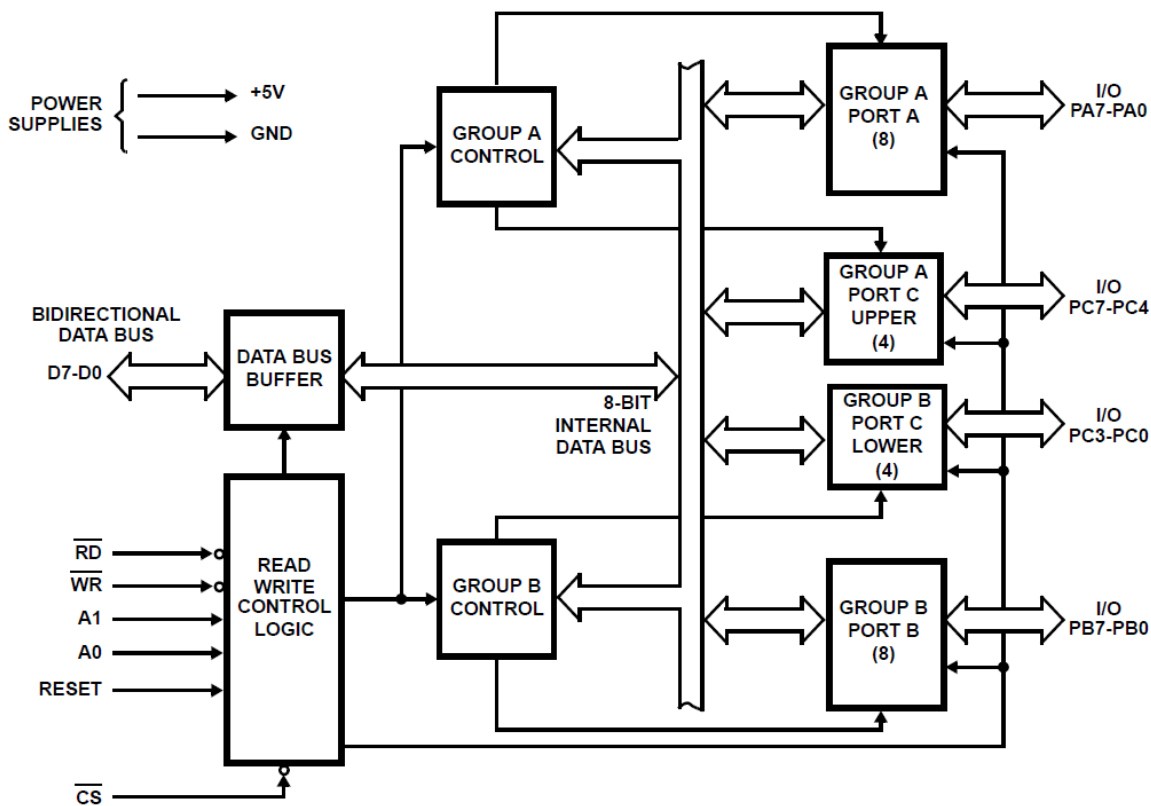
□ Programabilni V/I krmilnik - Intel 82C55A Programmable Peripheral Interface

CMOS tehnologija, kompatibilen s TTL vezji

Oznake pinov



Funkcijski diagram



<https://www.renesas.com/us/en/www/doc/datasheet/82c55a.pdf>

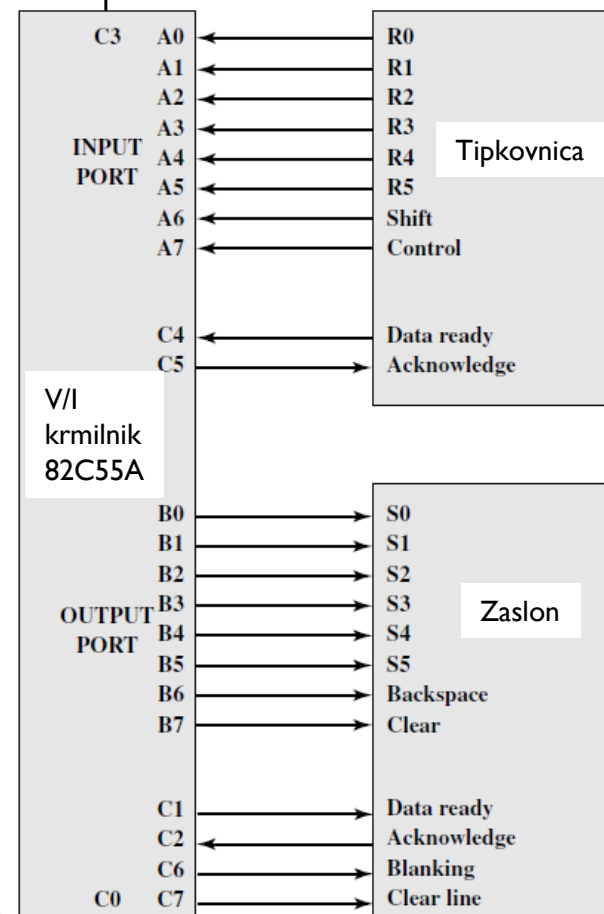
❑ Povezava tipkovnice

- 8-bitna V/I vrata: A0-A7 (A6-Shift, A7-Control)
- Prekinitvena zahteva: C3
- Usklajevanje:
 - Data ready: C4
 - Acknowledge: C5

❑ Povezava zaslona

- 8-bitna V/I vrata: B0-B7 (B6-Backspace, B7-Clear)
- Prekinitvena zahteva: C0
- Usklajevanje:
 - Data ready: C1
 - Acknowledge: C2
 - Blanking: C6
 - Clear Line: C7

Prekinitvena zahteva



<https://www.renesas.com/us/en/www/doc/datasheet/82c55a.pdf>

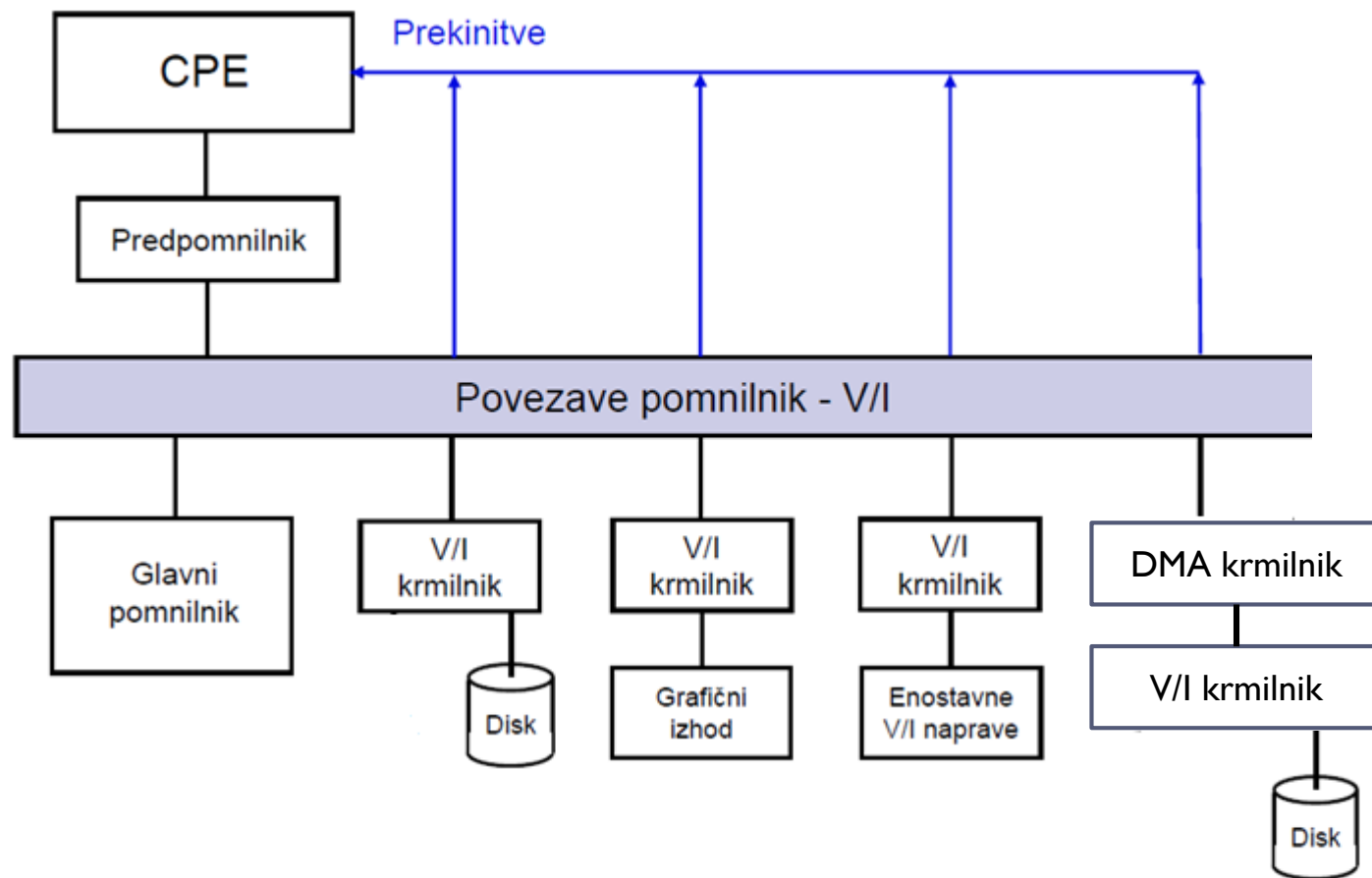
Prekinitvena zahteva

Naloga 1

- ❑ Imamo von Neumannov model računalnika s CPE, predpomnilnikom in glavnim pomnilnikom. Upoštevamo naslednje zahteve:
 - Vsaka V/I naprava je na vodilo povezana preko V/I krmilnika.
 - Na V/I krmilnik je lahko povezanih več enakih naprav.
 - Delovanje V/I naprav poteka s prekinitvenim mehanizmom.

- ❑ Narišite arhitekturno shemo, ki po vodilu povezuje s CPE in pomnilnik z naslednjimi V/I napravami:
 - Trdi disk HDD
 - Trdi disk HDD z DMA prenosom DMA
 - Grafični izhod (zaslon, projektor)
 - Enostavne V/I naprave: miška, tipkovnica

Naloga 1: Rešitev

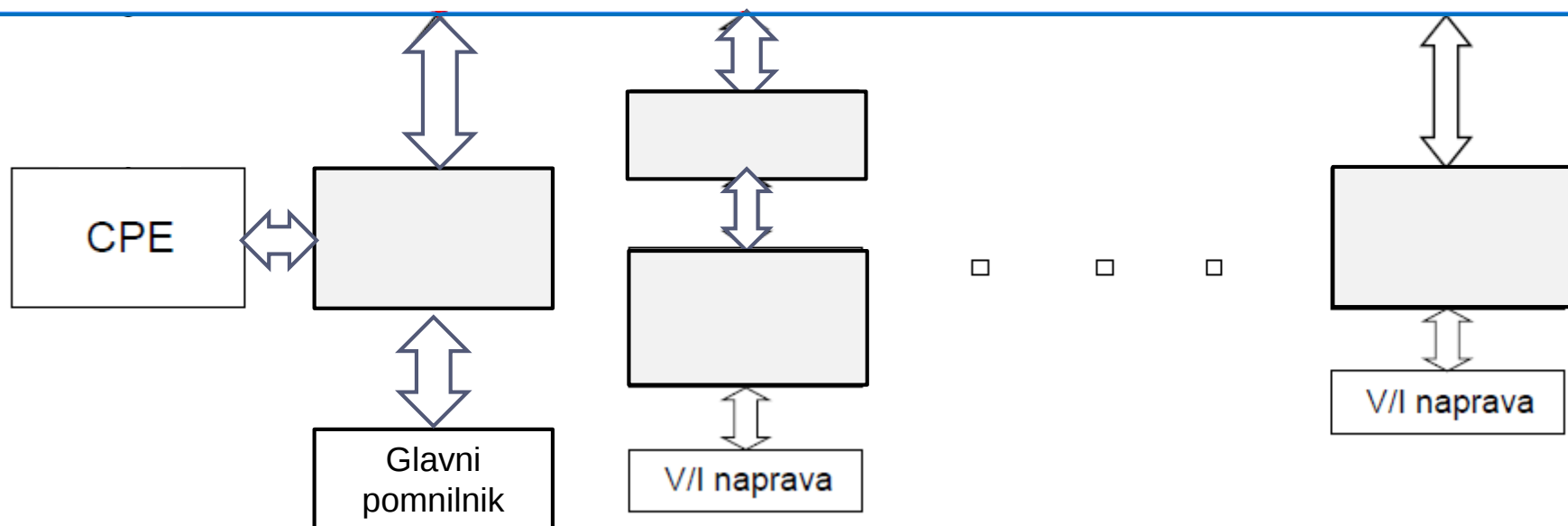


Naloga 2

Pomnilnik in V/I naprave so povezane na vodilo na tri načine, če uporabimo:

- DMA krmilnik kot samostojni modul
- V/I krmilnik z vgrajenim DMA
- Krmilnik pomnilnika, ki direktno komunicira s CPE
- **Dopolnite shemo V/I sistema s povezavami.**

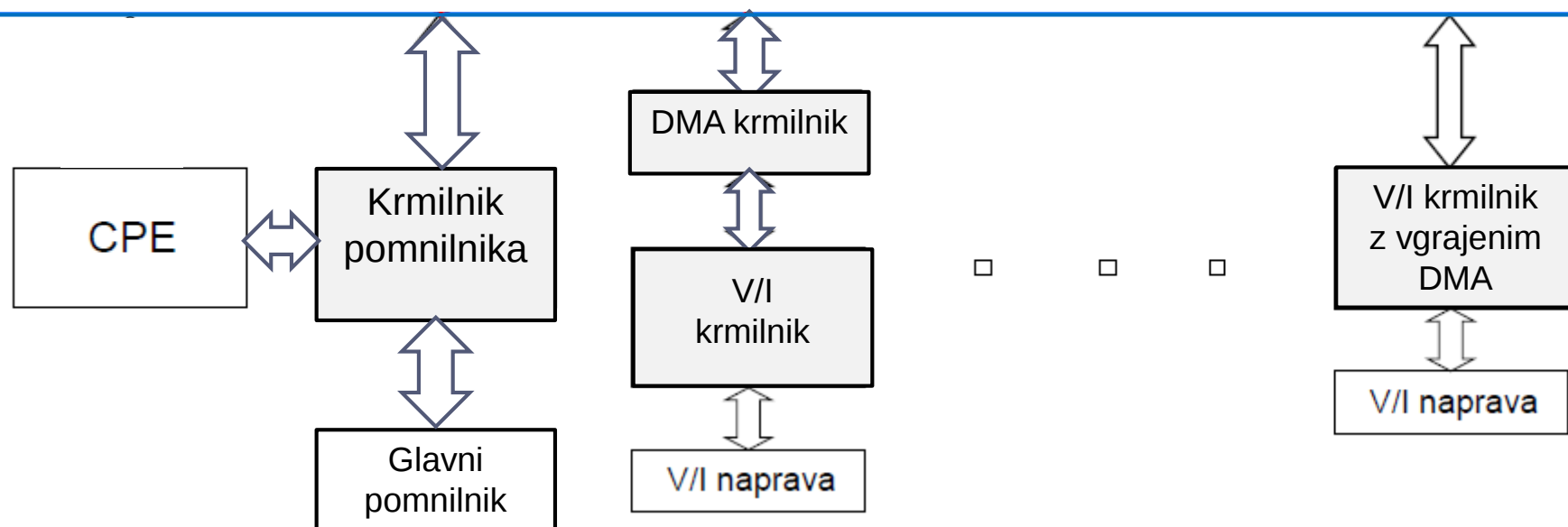
vodilo – povezuje več naprav



Naloga 2 - Rešitev

- ❑ Krmilnik pomnilnika – direktna komunikacija med glavnim pomnilnikom in CPE
- ❑ DMA krmilnik, V/I krmilnik, V/I naprava,
- ❑ V/I krmilnik z vgrajenim DMA, V/I naprava

vodilo – povezuje več naprav



-
- ▶ `HAL_ADC_Start(&hadc1); // start the adc`
 - ▶ `HAL_ADC_PollForConversion(&hadc1, 100); // poll for conversion`
 - ▶ `adc_val = HAL_ADC_GetValue(&hadc1); // get the adc value`
 - ▶ `HAL_ADC_Stop(&hadc1); // stop adc`
 - ▶ `HAL_Delay (500); // wait for 500ms`