Sklopovski UI prijenos - DMA

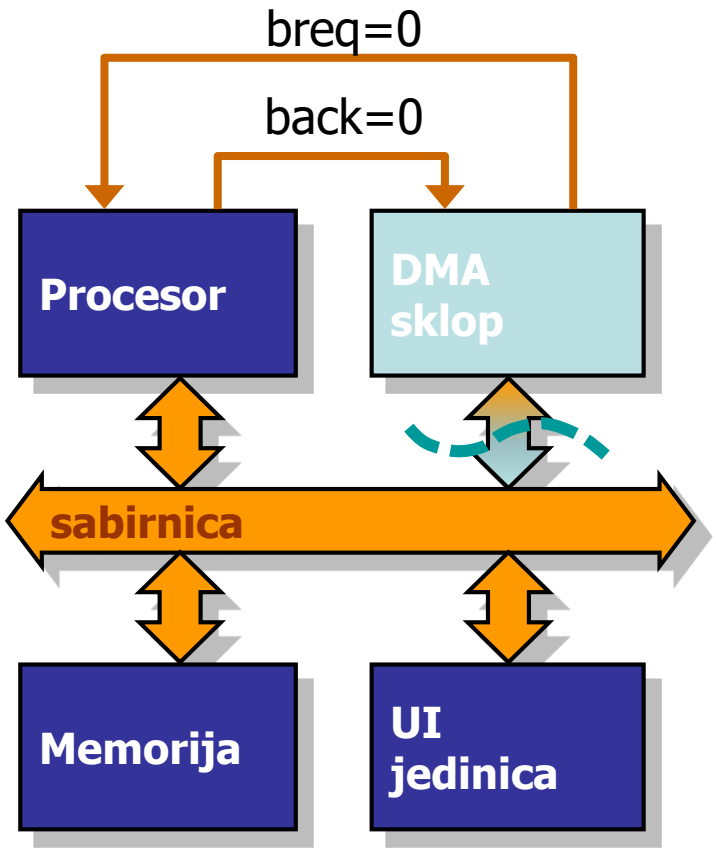
DMA prijenos

- Izravni pristup memoriji ili DMA (Direct Memory Access) je **sklopovski** ulazno-izlazni prijenos
- Prijenos ne obavlja procesor, nego posebna DMA-jedinica (ili DMA-sklop, engl. DMA controller)
- Velika brzina prijenosa (podatci ne prolaze kroz procesor)
- Moguć je prijenos između memorije i UI jedinice ili neka druga kombinacija izvora i odredišta podataka

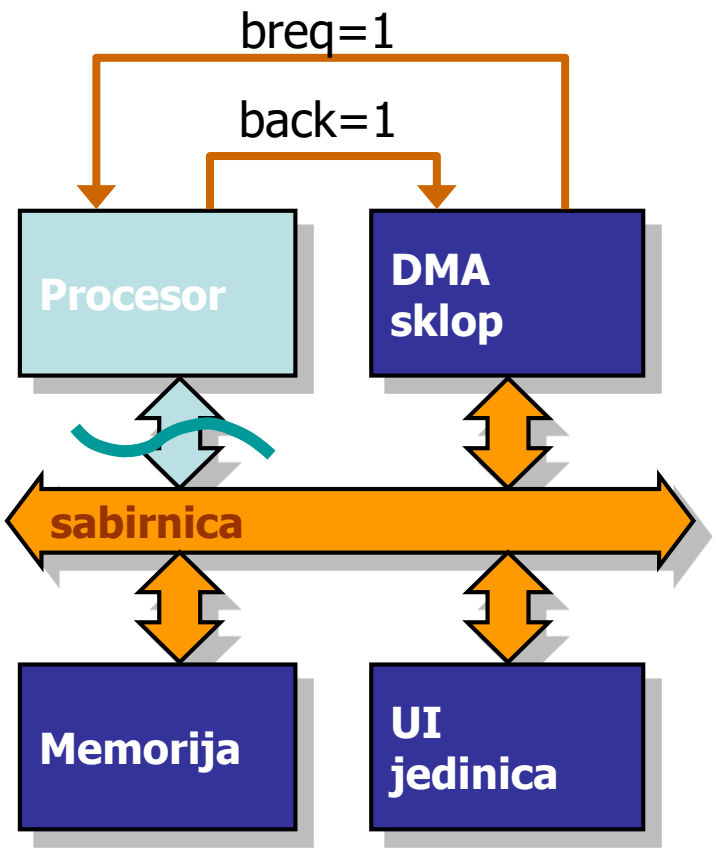
DMA prijenos

- Osnovna ideja:
  - Procesor inicijalizira DMA-sklop, kako bi ovaj znao koliko podataka treba prenijeti, gdje su izvor i odredište podataka itd.
  - DMA-sklop preuzme upravljanje nad sabirnicom i obavlja prijenos
  - Procesor je za vrijeme prijenosa neaktivan
  - Procesor i DMA-sklop moraju se međusobno sinkronizirati kako bi u svakom trenutku samo jedan od njih upravljao sabirnicom: za to se koriste linijama BREQ i BACK
  - Pod upravljanjem sabirnicom misli se na iniciranje sabirničkih transakcija čitanja i pisanja (npr. upravljanje adresnom sabirnicom, rd, wr, itd.)

DMA - Shema



Procesor upravlja sabirnicom*



DMA-sklop upravlja sabirnicom

*Slika simbolično prikazuje stanje na sabirnici. DMA sklop ne upravlja sabirnicom, ali mu procesor može normalno pristupati kao i drugim UI jedinicama i memoriji

DMA - Brzina prijenosa

- Približna usporedba bezuvjetnog prijenosa i DMA za N podataka (npr. iz VJ u memoriju):

	Naredba		trajanje
PETLJA	**LOAD**	**R0, (VJ_READ)**	**2**
	STORE	**R0, (R2)**	**2**
	ADD	**R2, 4, R2**	**1**
	SUB	**R3, 1, R3**	**1**
	JR_NZ	**PETLJA**	**2**

- Vidimo da za N podataka programu treba približno N*8 perioda (stvarno mu treba jedan period manje (N*8 – 1) jer kod zadnjeg prolaska kroz petlju JR traje samo jedan period, ali je to zanemarivo u odnosu na N*8).

- U slučaju prijenosa podataka korištenjem FRISC-DMA, sustavu će trebati N*2 perioda (DMA čita podatak preko sabirnice (prvi period) i onda ga zapisuje (drugi period)).

- Kod mnogih stvarnih VJ čest je slučaj da je DMA dio VJ pa DMA ne treba komunicirati sa VJ preko procesorske sabirnice te je za jedan podatak potreban samo jedan ciklus !!!

DMA - Vrste prijenosa

- Vrste DMA prijenosa:
  - Zaustavljanje procesora (engl. continuous, halting)
  - Krađa ciklusa (engl. cycle stealing, word-at-a-time)
  - Blokovski (engl. burst)

DMA – Zaustavljanje procesora

- DMA zaustavljanjem procesora odvija se ovako:
  - DMA-sklop preuzme upravljanje nad sabirnicom
  - DMA-sklop prenese **sve podatke**
  - Tek tada upravljanje nad sabirnicom se vraća procesoru koji je cijelo vrijeme bio zaustavljen

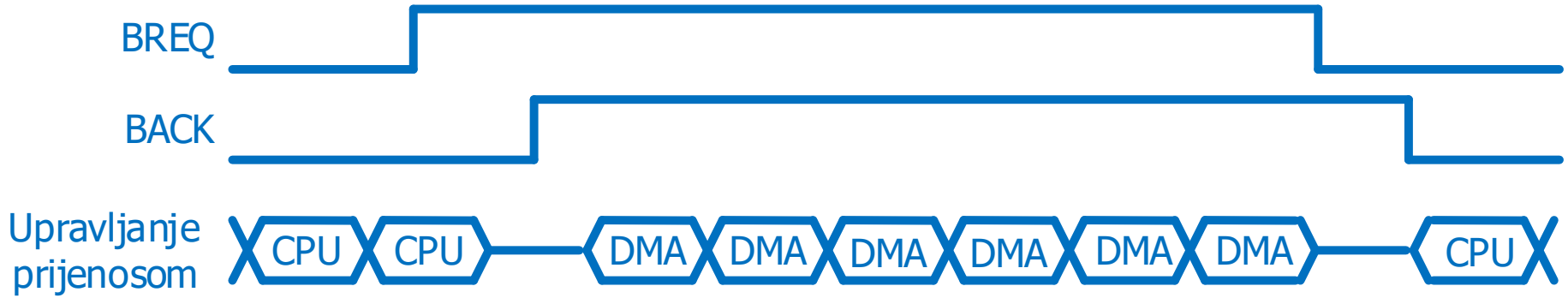
- Dok DMA-sklop upravlja sabirnicom, procesor ne može dohvaćati naredbe iz memorije i potpuno je neaktivan (jedino čeka dojavu od DMA da može nastaviti s radom)

- Nedostatak: procesor može dulje vrijeme biti neaktivan

- Prednost: najbrži prijenos

DMA - Zaustavljanje procesora

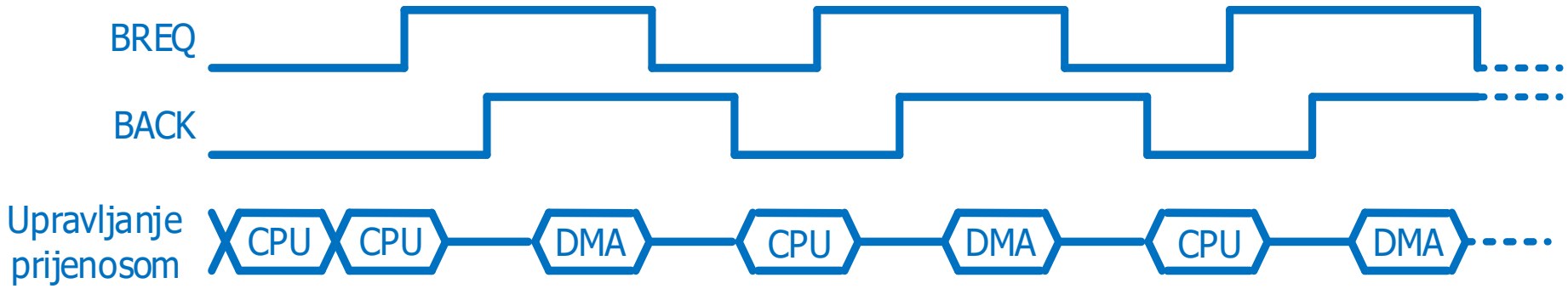
- **breq** je prioritetniji od svih zahtjeva za prekid



DMA - Krađa ciklusa

- DMA krađom ciklusa odvija se ovako:
  - DMA-sklop preuzme upravljanje nad sabirnicom
  - DMA-sklop prenese **jedan podatak**
  - Upravljanje nad sabirnicom se vraća procesoru
  - Gornja tri koraka se ponavljaju dok se ne prenesu svi podatci
- Prednost: procesor se usporava, ali ipak izvodi glavni program
- Nedostatak: sporije od zaustavljanja procesora (dio vremena troši se na sinkronizaciju oko upravljanja sabirnicom)

DMA - Krađa ciklusa



DMA - Blokovski prijenos

- DMA blokovskim prijenosom odvija se ovako:
  - DMA-sklop preuzme upravljanje nad sabirnicom
  - DMA-sklop prenese **nekoliko podataka (tj. blok)**
  - Upravljanje nad sabirnicom se vraća procesoru
  - Gornja tri koraka se ponavljaju dok se ne prenesu svi podatci

- Kompromis između zaustavljanja procesora i krađe ciklusa
- Zaustavljanje procesora može se promatrati kao blokovski prijenos kod kojeg je veličina bloka jednaka svim podatcima
- Krađa ciklusa može se promatrati kao blokovski prijenos kod kojeg je u bloku samo jedan podatak

DMA-sklopovi

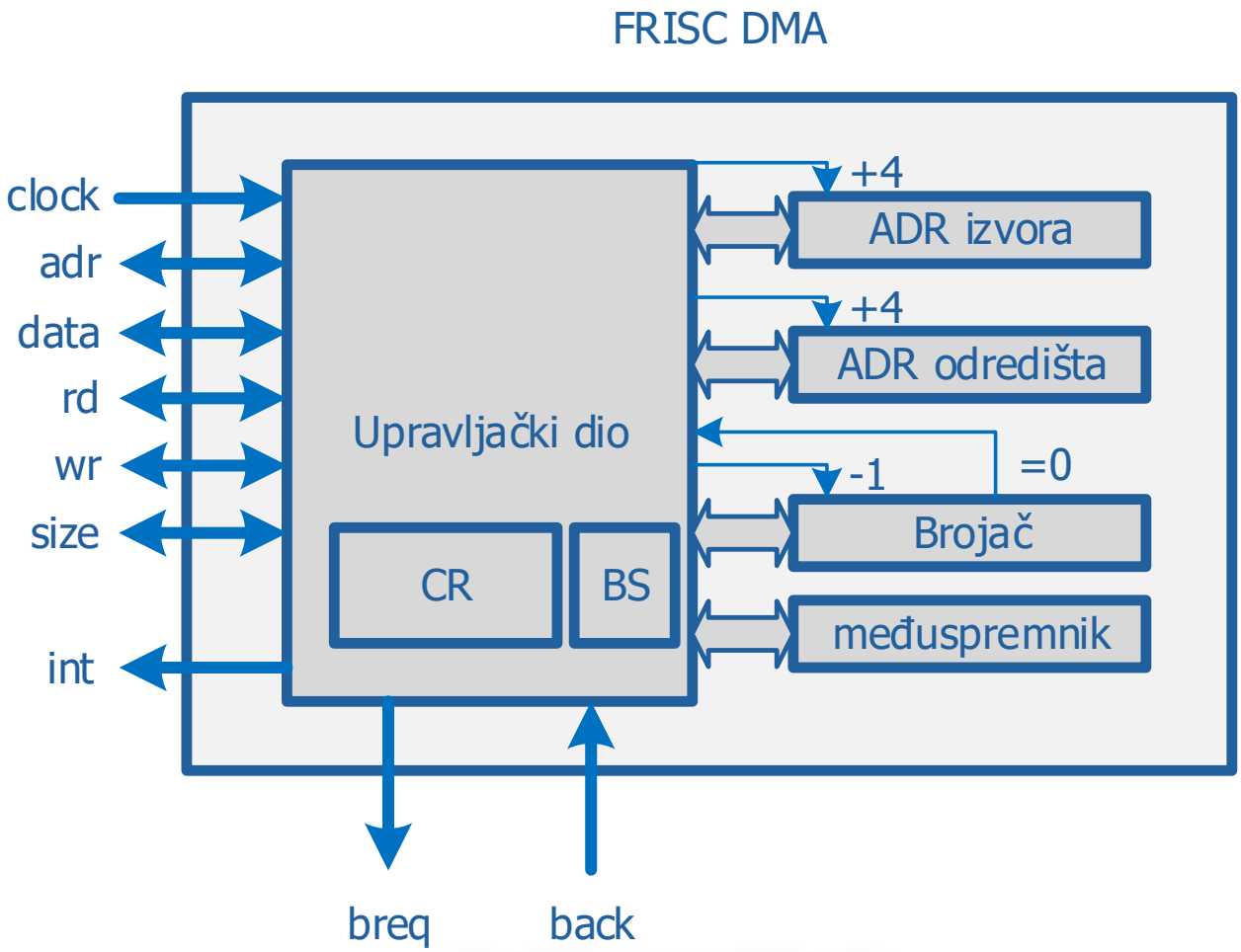
- DMA-sklopovi mogu imati različite mogućnosti:
  - prijenos podataka između različitih izvora i odredišta
    - memorija → VJ
    - memorija → memorija (kopiranje bloka, inicijalizacija bloka)
    - VJ → memorija
    - VJ → VJ
  - traženje podataka u bloku ili kombinirani prijenos s traženjem
  - odabir načina prijenosa
  - dojava kraja prijenosa
    - postavljanje spremnosti
    - prekid
    - generiranje impulsa (mogu se prebrajati npr. CT-om)
  - itd.

Sklop FRISC-DMA

Sklop FRISC-DMA

- Sklop FRISC-DMA ima sljedeće mogućnosti:
  - Prijenos 32-bitnih podataka
  - Odabir vrste prijenosa:
    - zaustavljanje procesora
    - krađa ciklusa
  - Zbog memorijskog preslikavanja, tj. jednakog pristupa memoriji i UI jedinicama, izvor i odredište mogu biti bilo koja kombinacija ovih komponenata
  - Pristup UI jedinicama obavlja se bez provjere (bezuvjetno) što znači da UI jedinica mora biti sposobna primati/slati podatke brzinom DMA-prijenosa
  - Dojava kraja prijenosa obavlja se postavljanjem spremnosti i prekidom

FRISC-DMA


FRISC DMA
clock
adr
data
rd
wr
size
int
Upravljački dio
CR
BS
+4
ADR izvora
+4
ADR odredišta
-1
=0
Brojač
međuspremnik
breq
back

Programski pogled

adresa	Pisanje	čitanje
PA	upis adrese izvora	čitanje adrese izvora
PA + 4	upis adrese odredišta	čitanje adrese odredišta
PA + 8	upis u brojač podataka	čitanje brojača podataka
PA + 12_{10}	upis upravljačke riječi CR	Čitanje CR
PA + 16_{10}	pokretanje prijenosa	-
PA + 20_{10}	potvrda prihvata prekida (tj. brisanje bistabila stanja)	čitanje bistabila stanja BS

Uočite da DMA nema odvojene adrese za dojavu prihvata prekida i kraja posluživanja. DMA ne može postati ponovno spreman u ovisnosti o nekom vanjskom procesu, nego u ovisnosti o FRISC-u (tj. programu). Zato ne treba posebna dojava o kraju posluživanja.

CR

bitovi 31 – 4	bit 3	bit 2	bit 1	bit 0
-	DESTINATION	SOURCE	MODE	INT
	0 – memorija 1 – vanjska jedinica	0 – memorija 1 – vanjska jedinica	0 – zaustavljanje procesora 1 – krađa ciklusa	0 – ne postavlja prekid 1 – postavlja prekid

Bitovi SOURCE i DESTINATION zapravo ne određuju je li izvor/odredište zaista memorija ili vanjska jedinica, nego samo definiraju hoće li se ili neće adresni registri povećavati nakon svakog prenesenog podatka.

Inicijalizacija sklopa FRISC-DMA

- Prilikom inicijalizacije treba zadati (u bilo kojem redoslijedu):
  - broj podataka
  - adresu izvora
  - adresu odredišta
  - upravljačku riječ

- Nakon toga treba pokrenuti DMA-prijenos upisom na adresu PA+16 (odmah nakon toga počinje DMA-prijenos)
  - ako je odabrano zaustavljanje procesora, dio programa iza naredbe za pokretanje DMA-prijenosa neće se izvoditi sve dok se prijenos ne završi
  - ako je odabrana krađa ciklusa, dio programa iza naredbe za pokretanje se izvodi, ali usporeno

DMA Primjeri

Treba prenijeti 1000_{10} podataka iz VJ na adresi FFFF3330 u memoriju od adrese 4000. Adresa DMA-sklopa je FFFF0000. Prije, za vrijeme ili nakon DMA-prijenosa treba se izvesti potprogram POTPR.

Kad je prijenos gotov, a nakon završetka potprograma POTPR, treba pozvati potprogram GOTOVO te nakon njega nastaviti s radom glavnog programa (pretpostavka je da su ovi potprogrami već napisani i da postoje u memoriji).

Prijenos treba obaviti zaustavljanjem procesora.

```
DMA_SRC    EQU 0FFFF0000
DMA_DEST   EQU 0FFFF0004
DMA_SIZE   EQU 0FFFF0008
DMA_CTRL   EQU 0FFFF000C
DMA_START  EQU 0FFFF0010
DMA_BS     EQU 0FFFF0014
```

bitovi 31 – 4	bit 3	bit 2	bit 1	bit 0
-	DESTINATION	SOURCE	MODE	INT
	0 – memorija 1 – vanjska jedinica	0 – memorija 1 – vanjska jedinica	0 – zaustavljanje procesora 1 – krađa ciklusa	0 – ne postavlja prekid 1 – postavlja prekid

```
INIT    MOVE  10000, SP
        ; inicijalizacija DMA-sklopa
        MOVE  0FFFF3330, R0     ; upis adrese
        STORE R0, (DMA_SRC)     ; izvora

        MOVE  4000, R0          ; upis adrese
        STORE R0, (DMA_DEST)    ; odredišta

        MOVE  %D 1000, R0       ; upis broja
        STORE R0, (DMA_SIZE)    ; podataka

        ; Upis upravljačke riječi:
        MOVE  %B 0100, R0
        STORE R0, (DMA_CTRL)                    >>>>
```

```
<<<<
        ; pokretanje DMA-prijenosa
GLAVNI STORE R0, (DMA_START)

        ; dio programa koji će se izvesti tek nakon
        ; završenog DMA-prijenosa
        STORE R0, (DMA_BS) ; brisanje BS DMA

        CALL POTPR
        CALL  GOTOVO
               ...              ; nastavak glavnog programa
```

DMA Primjeri

Prethodni zadatak treba riješiti krađom ciklusa.

Detekciju kraja prijenosa treba napraviti pomoću ispitivanja bistabila stanja.

```
DMA_SRC    EQU 0FFFF0000
DMA_DEST   EQU 0FFFF0004
DMA_SIZE   EQU 0FFFF0008
DMA_CTRL   EQU 0FFFF000C
DMA_START  EQU 0FFFF0010
DMA_BS     EQU 0FFFF0014
```

bitovi 31 – 4	bit 3	bit 2	bit 1	bit 0
-	DESTINATION	SOURCE	MODE	INT
	0 – memorija 1 – vanjska jedinica	0 – memorija 1 – vanjska jedinica	0 – zaustavljanje procesora 1 – krađa ciklusa	0 – ne postavlja prekid 1 – postavlja prekid

```
INIT   MOVE  10000, SP
       ; inicijalizacija DMA-sklopa
       MOVE  0FFFF3330, R0     ; upis adrese
       STORE R0, (DMA_SRC)     ; izvora

       MOVE  4000, R0          ; upis adrese
       STORE R0, (DMA_DEST)    ; odredišta

       MOVE  %D 1000, R0       ; upis broja
       STORE R0, (DMA_SIZE)    ; podataka

       ; Upis upravljačke riječi:
       MOVE  %B 0110, R0
       STORE R0, (DMA_CTRL)
```

>>>>

```
<<<<
        ; pokretanje DMA-prijenosa
GLAVNI STORE R0, (DMA_START)

        ; dio programa koji se izvodi usporeno i istodobno
        ; s DMA-prijenosom

        CALL  POTPR

        ; dio programa koji se mora izvesti NAKON DMA-prijenosa

CEKAJ   LOAD  R0, (DMA_BS) ; učitaj spremnost DMA-sklopa
        AND   R0, 1, R0
        JR_Z  CEKAJ          ; čekaj završetak prijenosa
        STORE R0, (DMA_BS)   ; brisanje BS DMA
        CALL  GOTOVO
        ...             ; nastavak glavnog programa
```