DIGITALNI PROCESORI SIGNALA – Arhitektura TMS320C55x procesora i uvod u CCS

Vladimir Petrović, Aleksandra Lekić

Univerzitet u Beogradu – Elektrotehnički fakultet

2018/2019

Plan rada

- Upoznavanje sa karakteristikama C55X DSP procesora
- Upoznavanje sa radom CCS
- Projektovanje FIR filtra
- Projektovanje IIR filtra
- FFT – softverska i hardverska realizacija
- Obrada signala u realnom vremenu

Literatura - eZdsp

TMS320C5505

http://www.ti.com/lit/ds/symlink/tms320c5505.pdf

Schematics

http://support.spectrumdigital.com/boards/usbstk5505/revd/files/usbstk5505_Schematics_RevD.pdf

Technical Reference

http://focus.ti.com/en/download/dsp/usbstk5505_techref_revb.pdf

Za Rev. B, mi imamo Rev. D. I nisu neke razlike.

User's Guide

http://www.ti.com/lit/ug/sprugh5b/sprugh5b.pdf

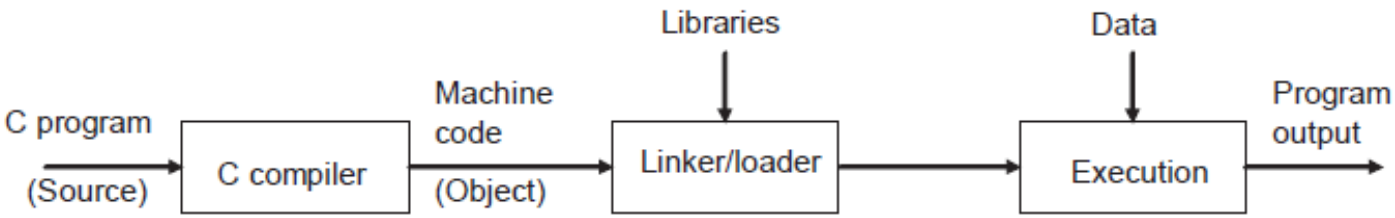
DIGITALNA OBRADA SIGNALA

Za digitalnu obradu signala moguće je koristiti:

- mikrokontrolere opšte namene
- DSP opšte namene
- FPGA opšte (specijalne?) namene
- ASIC specijalne namene

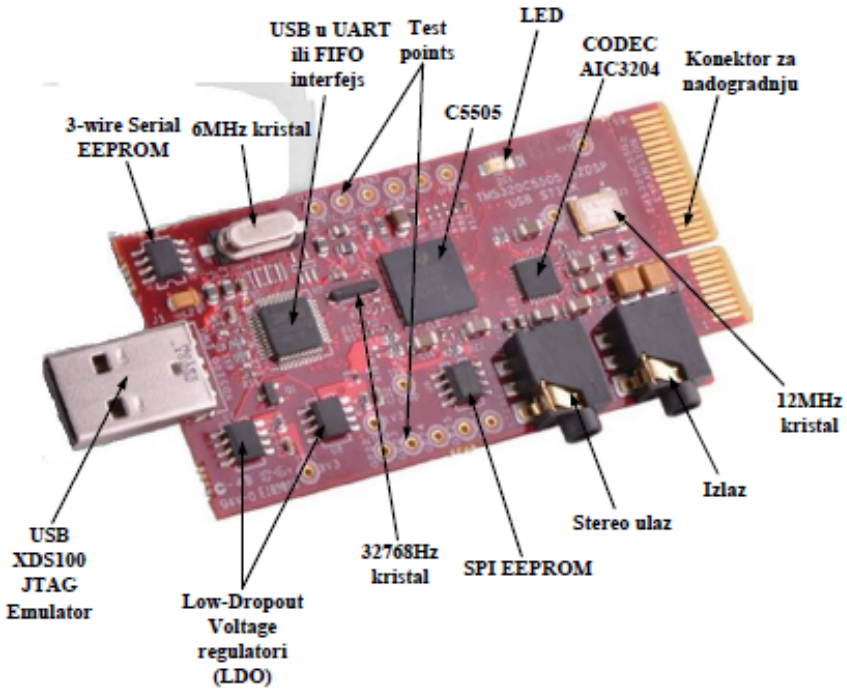
DSP software

- Zasniva se na *high-level* programskim jezicima: C/C++
- Optimizacija brzine izvršavanja koda se postiže pisanjem pojedinih funkcija u asembleru
- Lak je za modifikaciju i debagovanje
- Ulazno/izlazne operacije se lako implementiraju, pa je analiza rada sistema jednostavna
- *Floating-point DSP* vs. *Fixed-point DSP*



TMS320C5505 eZDSP USB Stick

- Malih dimenzija: 8 cm $\times$ 4 cm
- Male potrošnje
- Efikasno računanje skalarnog proizvoda (npr. FIR filtar) – dualna MAC (*multiply-and-accumulate*) arhitektura



TMS320C55X DSP

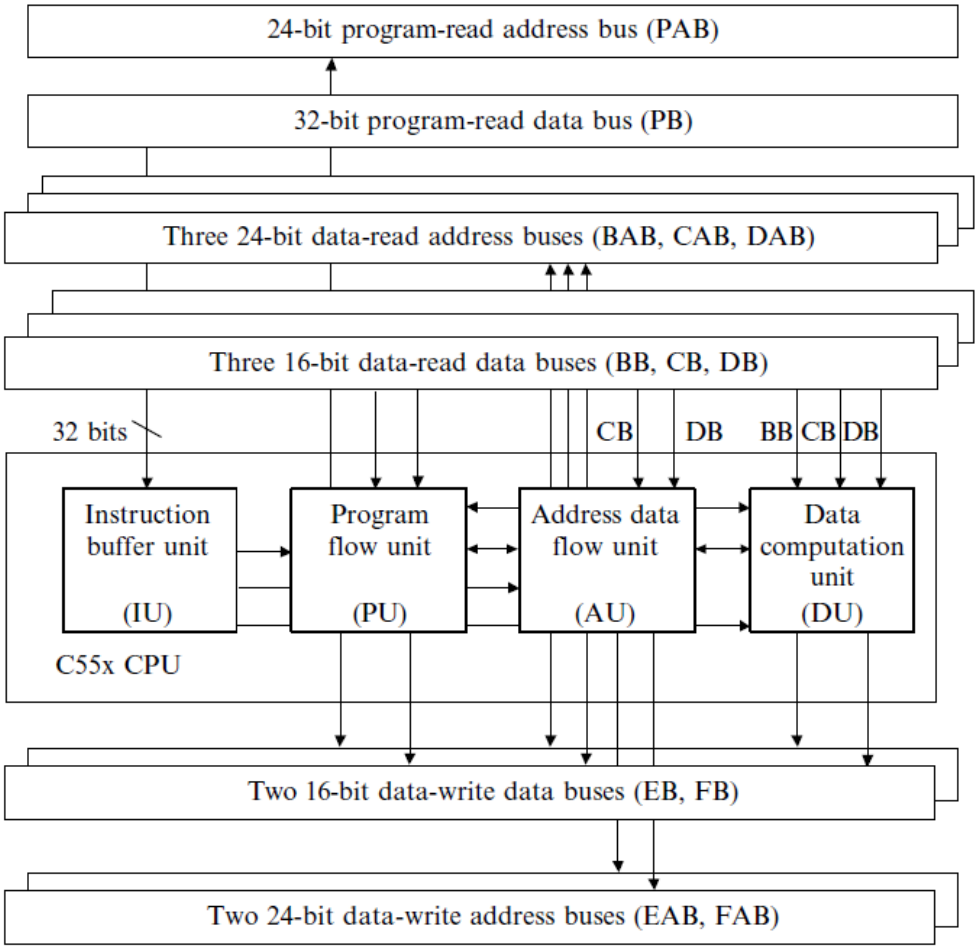
HARDWARE FEATURES			C5505
Peripherals Not all peripheral pins are available at the same time (for more detail, see the Device Configuration section).	External Memory Interface (EMIF)		Asynchronous (8/16-bit bus width) SRAM, Flash (NOR, NAND), SDRAM and Mobile SDRAM (16-bit bus width)(1)
	DMA		Four DMA controllers each with four channels, for a total of 16 channels
	Timers		2 32-Bit General-Purpose (GP) Timers 1 Additional Timer Configurable as a 32-Bit GP Timer and/or a Watchdog
	UART		1 (with RTS/CTS flow control)
	SPI		1 with 4 chip selects
	I^2C		1 (Master/Slave)
	I^2S		4 (Two Channel, Full Duplex Communication)
	USB 2.0 (Device *only*)		High- and Full-Speed Device
	MMC/SD		2 MMC/SD, 256 byte read/write buffer, max 50-MHz clock for SD cards, and signaling for DMA transfers
	LCD Bridge		1 (8-bit or 16-bit asynchronous parallel bus)
	ADC (Successive Approximation [SAR])		1 (10-bit, 4-input, 16-µs conversion time)
	Real-Time Clock (RTC)		1 (Crystal Input, Separate Clock Domain and Power Supply)
	FFT Hardware Accelerator		1 (Supports 8 to 1024-point 16-bit real and complex FFT)
	General-Purpose Input/Output Port (GPIO)		Up to 26 pins (with 1 Additional General-Purpose Output (XF) and 4 Special-Purpose Outputs for Use With SAR)
On-Chip Memory	Size (Bytes)		320KB RAM, 128KB ROM
	Organization		• 64KB On-Chip Dual-Access RAM (DARAM) • 256KB On-Chip Single-Access RAM (SARAM) • 128KB On-Chip Single-Access ROM (SAROM)
JTAG BSDL_ID	JTAGID Register (Value is: 0x1B8F E02F)		see Figure 5-44
CPU Frequency	MHz	1.05-V Core	60 or 75 MHz
		1.3-V Core	100 or 120 MHz
		1.4-V Core	150 MHz
Cycle Time	ns	1.05-V Core	16.67, $13.\overline{3}$ ns
		1.3-V Core	10, 8.33 ns
		1.4-V Core	6.66 ns
Voltage	Core (V)		1.05 V (60, 75 MHz)
			1.3 V (100, 120 MHz)
			1.4 V (150 MHz)
	I/O (V)		1.8 V, 2.5 V, 2.75 V, 3.3 V
LDO	ANA_LDO		1.3 V, 4 mA max current for PLL (V_{DDA_PLL}), SAR, and power management circuits (V_{DDA_ANA})
PLL Options	Software Programmable Multiplier		x4 to x4099 multiplier
BGA Package	10 x 10 mm		196-Pin BGA (ZCH)

TMS320C55X DSP

Ovo je *fixed-point* procesor koji ima:

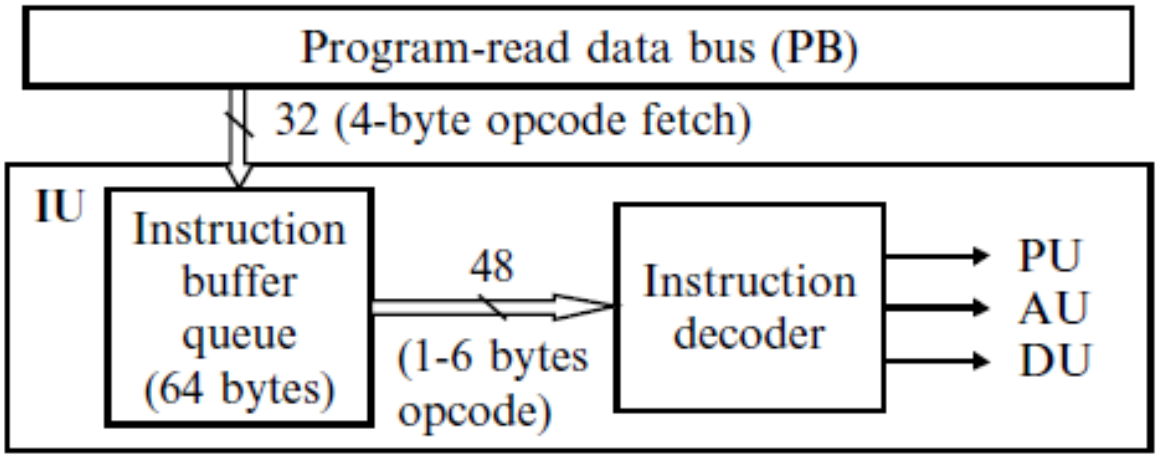
- 64-B bafer koji služi kao programski keš i za efikasno ponavljanje blokovskih operacija.
- dve 17 bit $\times$ 17 bit dualne MAC jedinice koje mogu da izvršavaju dualne *multiply-and-accumulate* operacije u jednom ciklusu.
- 40 bit ALU velike preciznosti + 16 bit ALU za jednostavne aritmetičke operacije koji radi paralelno sa glavnom ALU.
- osam pomoćnih registara.
- cirkularni adresni režim koji podržava do pet cirkularnih bafera.
- operacije za ponavljanje jedne instrukcije kao i bloka instrukcija.

TMS320C55X CPU



CPU - *Instruction buffer unit*

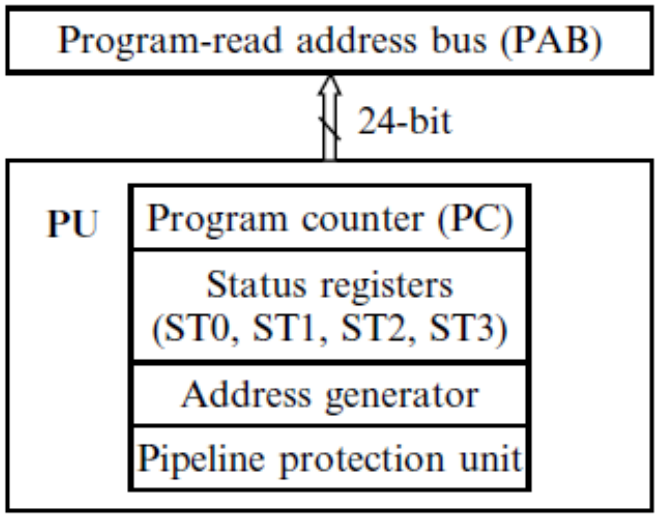
- prebacuje instrukciju iz memorije u CPU
- u svakom ciklusu dopremi 4 B instrukciju i smešta je u 64-bitni *instruction buffer*
- u jednom ciklusu dekodira 6 B programa koje dalje prosleđuje `AU`, `DU` ili `PU`



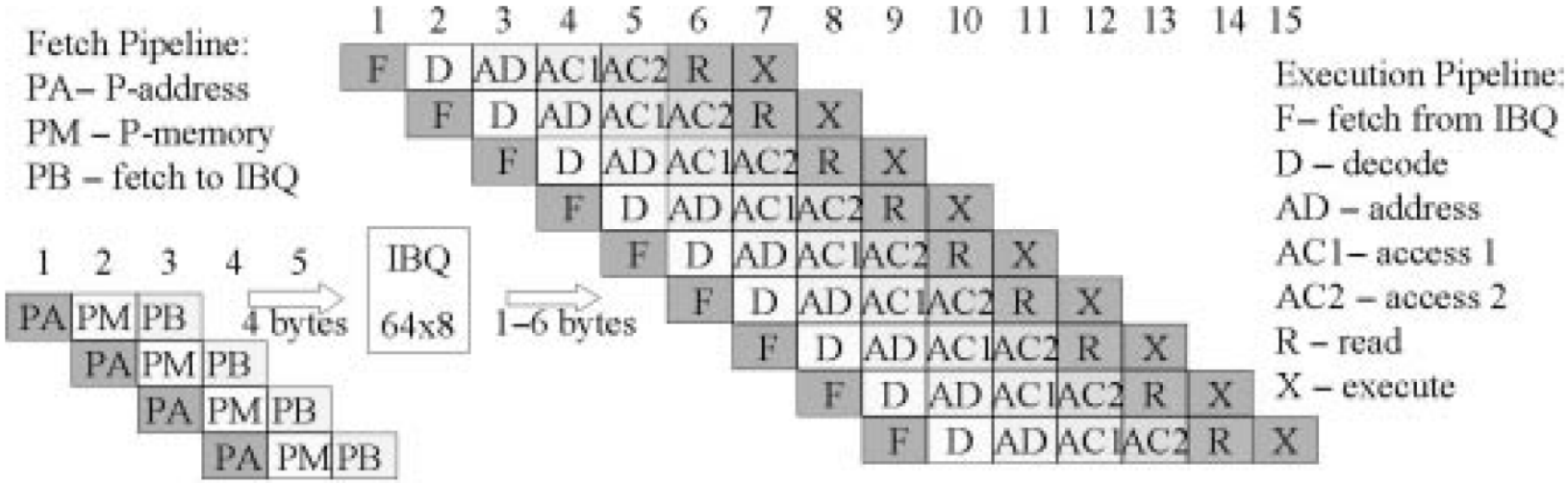
CPU - *Program flow unit*

- prati izvršavanje programa
- generiše 24-bitne adrese za 16 MB memorijski prostor

- `PC` - programski brojač
- `SR0-SR3` - statusni registri
- generator adresa
- *pipeline protection unit*

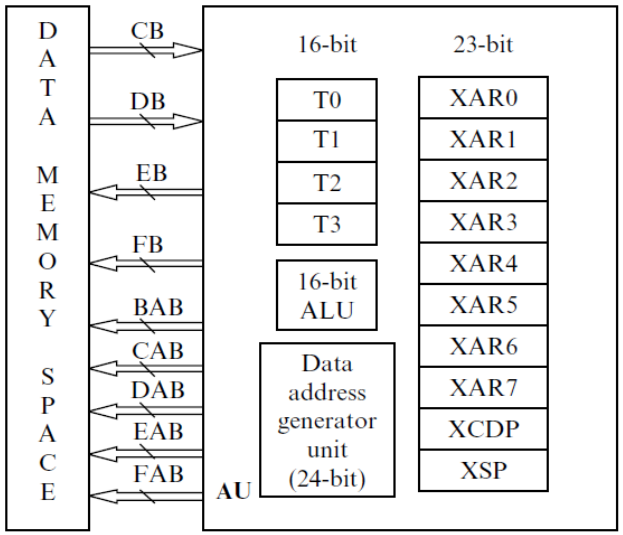


PIPELINE


Fetch Pipeline:
PA– P-address
PM – P-memory
PB – fetch to IBQ
1 2 3 4 5
PA PM PB
PA PM PB
PA PM PB
PA PM PB
4 bytes
IBQ
64x8
1–6 bytes
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
F D AD AC1 AC2 R X
Execution Pipeline:
F– fetch from IBQ
D – decode
AD – address
AC1– access 1
AC2 – access 2
R – read
X – execute

CPU - *Address data flow unit*

- Organizacija pristupa podacima
- 8 pomoćnih 23-bitnih registara (`XAR0-XAR7`) - *extended auxiliary register*
- 4 16-bitnih *temporary* registara (`T0-T3`)
- 23-bitni *extended coefficient data pointer* - `XCDP`
- 23-bitni *extended stack pointer* - `XSP`
- 16-bitna jednostavna ALU

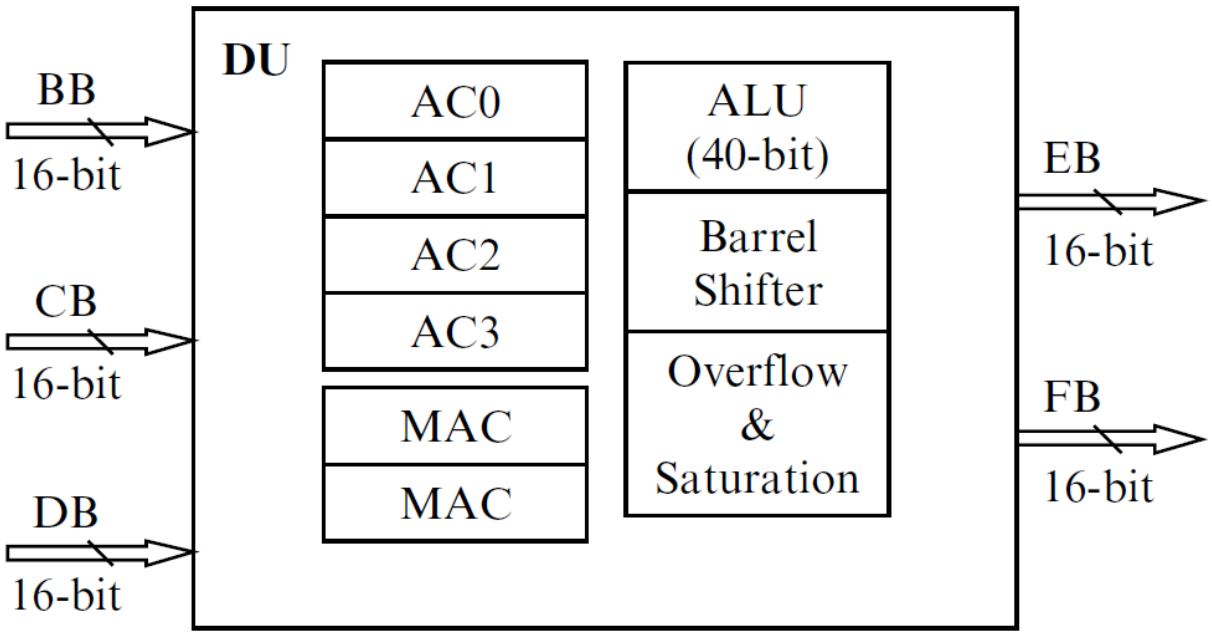


AU dozvoljava korišćenje:

- dve adrese i pokazivač na koeficijent za rad *dual-data* ili
- dve adrese i jedan koeficijent prilikom rada u jednom ciklusu
- najviše 5 cirkularnih bafera

CPU - *Data computation unit*

- Izvršava sve zahtevne aritmetičko-logičke operacije
- dve MAC jedinice
- 40-bitna ALU
- četiri 40-bitna akumulatora `AC0-AC3`
- *barrel shifter*, *rounding* i *saturation* logika



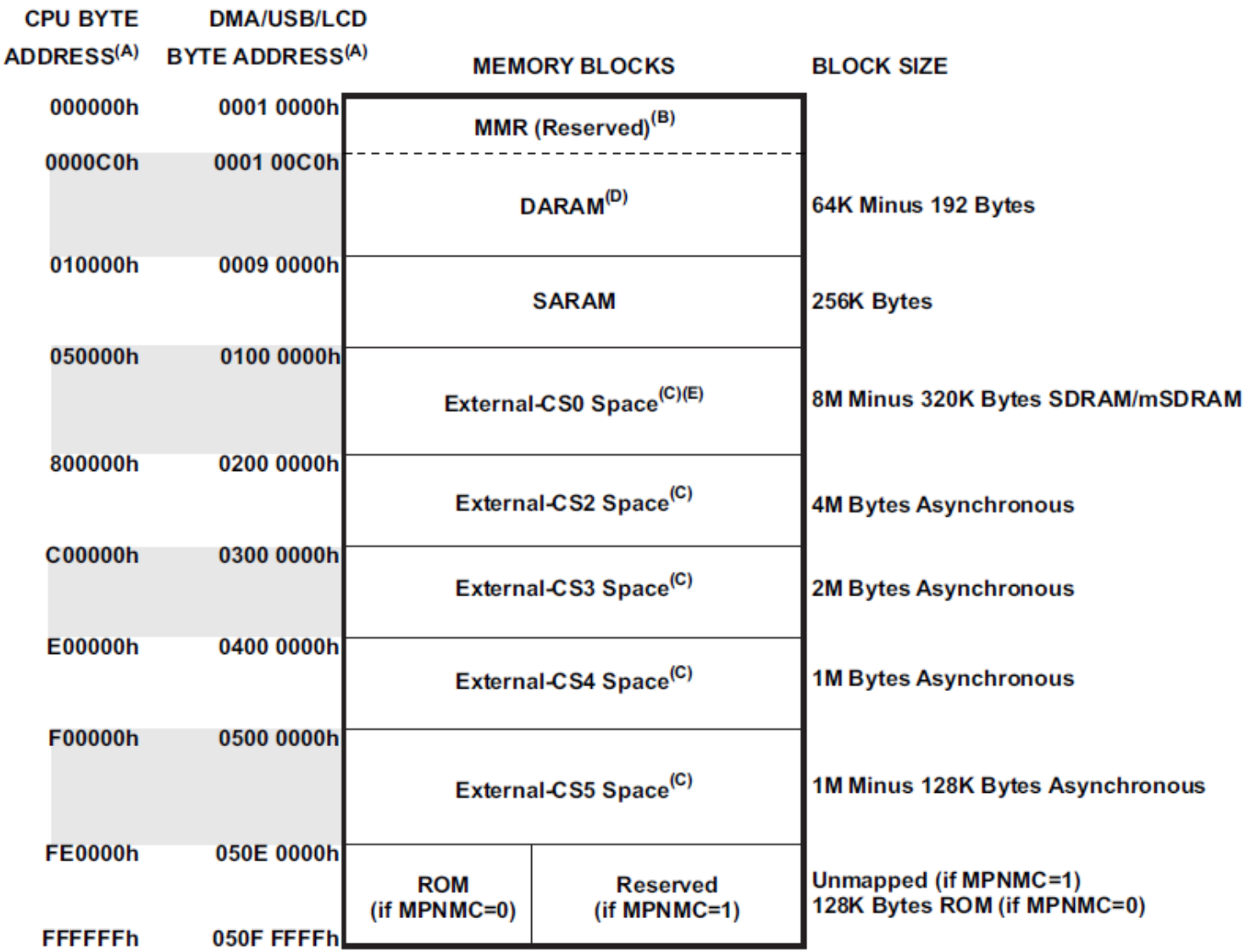
TMS320C55X - MAGISTRALE

- jedna 32-bitna programska magistrala
- tri 16-bitne magistrale podataka
- tri 24-bitne adresne magistrala

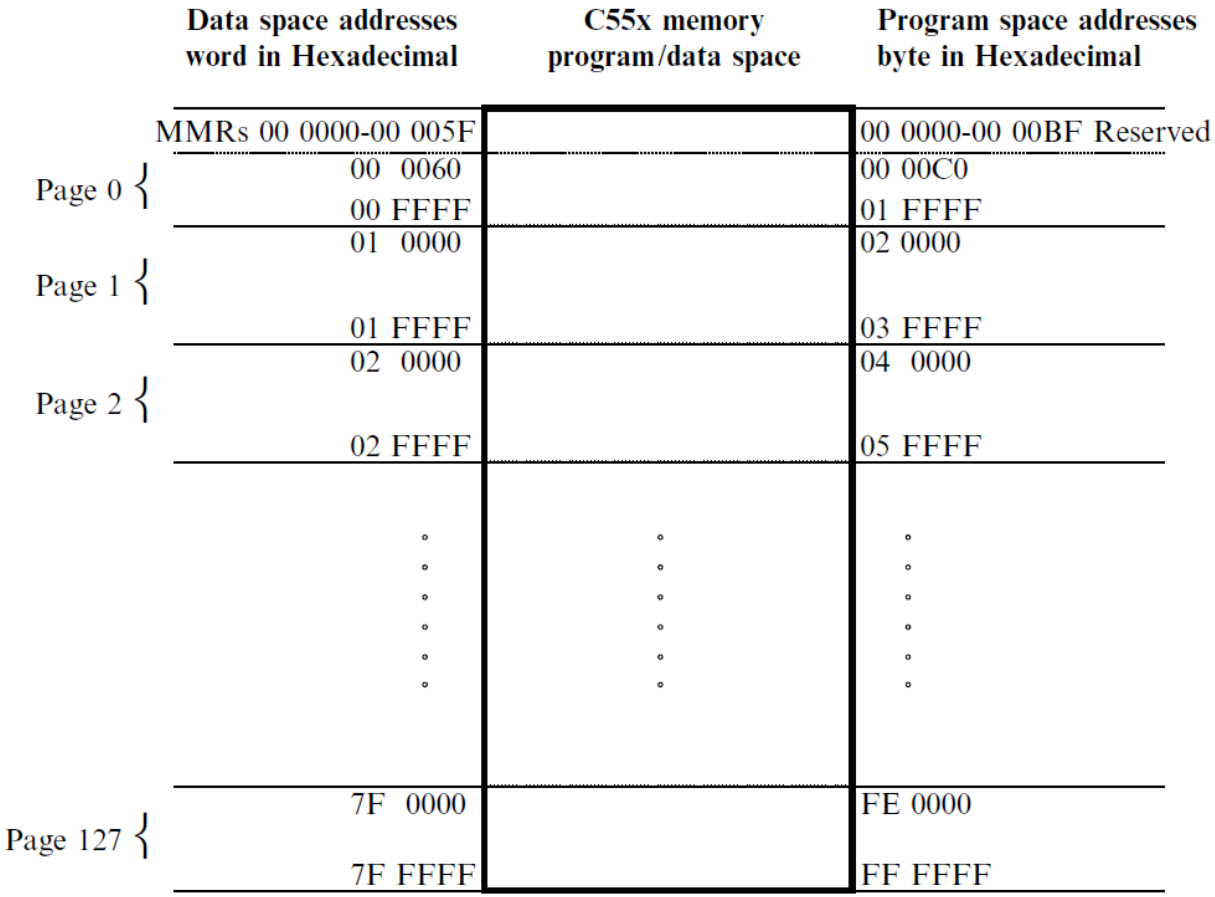
TMS320C55X - MEMORIJA

- 16 MB koji se mogu koristiti kao programska memorija i memorija za skladištenje podataka
- podaci su veličine 2 B = 1 word
- u CPU se dovodi 24-bitna instrukcija koja se dobija maskiranjem LSB
- prostor namenjen za podatke je podeljen na 128 memorijskih stranica po 16 k words
- memorijske lokacije `0x0 - 0x5F` na nultoj strani su memorijski mapirani registri - `MMR`

TMS320C55X - MEMORIJA



TMS320C55X - MEMORIJA



TMS320C55X CPU - ADRESIRANJE

1. direktno
2. indirektno
3. apsolutno
4. kružno (cirkularno)

Akumulatori

Akumulatori AC0-AC3 su 32-bitni.

Primer:

```
   mov *AR3, *AR4, AC0   ; AC0[15:0] = sadržaj na adresi
AR3, AC0[31:16] = sadržaj na adresi AR4
  mov AC0, AC1    ; AC1 = AC0
  mov AC0, *AR3    ;sadržaj na adresi AR3 = AC0[15:0]
  mov HI(AC0), *AR7    ;sadržaj na adresi AR7 = AC0[15:0]
```

Neposredno adresiranje

Operandi su u samom kodu instrukcije.
Dozvoljene su konstante sa 3, 4, 5, 7, 8, 9, 12, 16 i 23 bita.

Primer:

`add #FFFFh, *AR3;` sadržaj lokacije na koju pokazuje `AR3` je sabran sa `#FFFFh` i upisan na lokaciju na koju pokazuje `AR3`

DIREKTNO ADRESIRANJE

1. DP direktno

Direktno adresiranje korišćenjem DP i DPH registara.

DPH = gornjih 7 bita *extended data pointer*-a (XDP - 23 bita) – selektuje koja se od 128 stranica (*page*) memorije adresira
DP = donjih 16 bita XDP-a – selektuje adresu unutar stranice

Primer:

```
X  .set #0x1FFEE     ; specificiranje adrese varijable X
   mov #0x3, DPH     ; prebacuje 7-bitnu konstantu u DPH
   mov #0x0FFEF, DP     ; prebacuje 16-bitnu konstantu u DP
   .dp X
   mov #0x5555, @X     ; @ označava direktno adresiranje
   mov #0xFFFF, @(X+5)     ; upis 0xFFFF na lokaciju X+5
```

DIREKTNO ADRESIRANJE

2. SP direktno

- Direktno adresiranje korišćenjem *stack-pointer*-a
- Koristi se ako je `CPL` bit u statusnom registru `ST1_55` setovan, u suprotnom koristi se DP direktno adresiranje.
- `SP` = donjih 16 bita *extended stack pointer*-a (`XSP` - 23 bita)
- `SPH` = gornjih 7 bita `XSP`

Primer:

```
add *SP(6), AC0   ; AC0 = AC0 + vrednost sa adrese SPH: SP + 6
```

DIREKTNO ADRESIRANJE

3. adresiranje registarskih bita

- Način za pristup određenom bitu registra: `@bitoffset` - pristupa se bitu koji je na lokaciji `bitoffset` u LSB.
- Moguće je pristupiti bitima registara `AC0-AC3`, `AR0-AR7`, `T0-T3`.
- Jedino može za instrukcije `test/set/clear/complement` bita.
- Primer:

```
btstp  @28,AC0
```

AC0	00 3333 5555	AC0	00 3333 5555
ST0	0802	ST0	3802

4. PDP direkno adresiranje

Adrese periferija su 16-bitne. Viših 9-bita registra `PDP` adresira jednu od 512 periferijskih stranica podataka. Donjih 7-bita adresira jednu memorijsku lokaciju od 128 na stranici.
Korisiti se `port()` kvalifikator gde se zadaje 16-bitna vrednost.

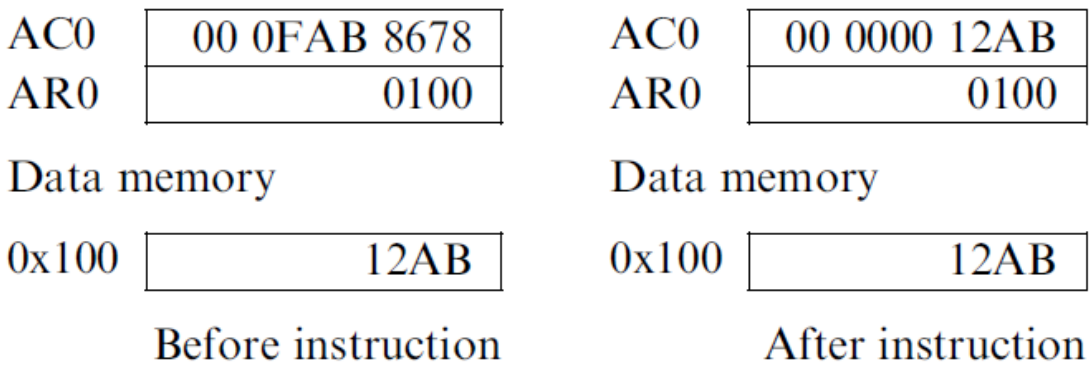
INDIREKTNO ADRESIRANJE

1. AR indirektno

- Koristi jedan od 8 pomoćnih registara (AR0-AR7) kao pokazivač na memorijsku lokaciju, I/O registre ili MMR.
- Viših 7 bita registra XAR pokazuje na memorijsku stranicu, a nižih 16 predstavlja adresu podatka unutar stanice

Primer:

```
mov *AR0, AC0    ; prebacuje podatak sa memorijske adrese na koju pokazuje AR0 u akumulator AC0
```



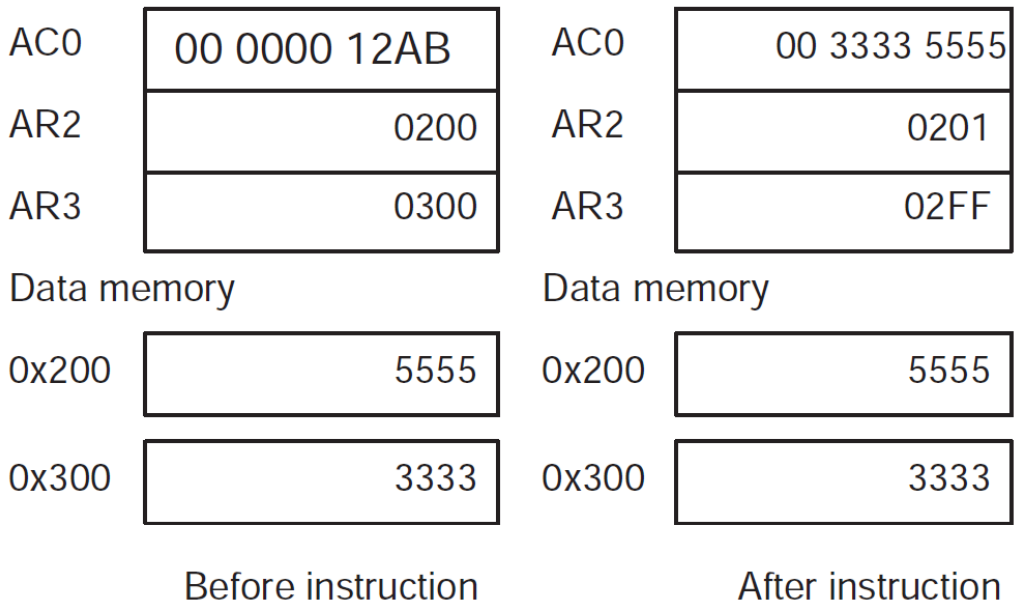
INDIREKTNO ADRESIRANJE

2. Dvostruko AR indirektno

- Omogućava dvostruki pristup memoriji
- Koriste se dva od 8 pomoćnih registara (AR0-AR7) kao pokazivače na memorijsku lokaciju, I/O registre ili MMR.

Primer:

```
mov *AR2+, *AR3-, AC0
```



INDIREKTNO ADRESIRANJE

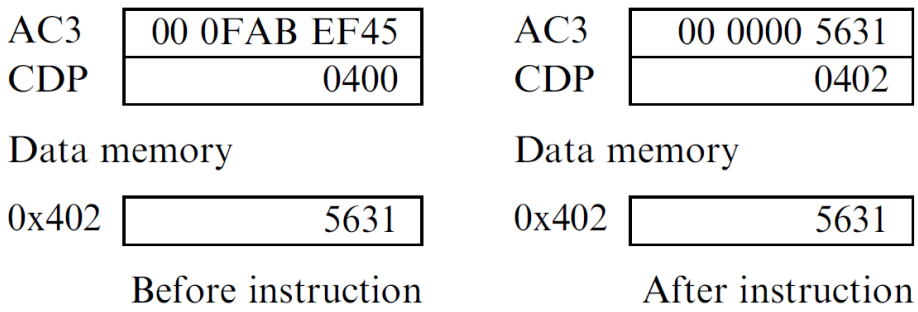
3. CDP indirektno

- Koriste se `CDP` (*coefficient data pointer*) koji pokazuje na memorijsku lokaciju.
- `CDP` predstavlja donjih 16 bita 23-bitnog `XCDP` (gornjih 7 bita je `CDPH`).

Primer:

```
mov *+CDP(#2), AC3    ; prebacuje podatak u akumulator AC3 sa
```

memorijske adrese na koju pokazuje `CDP` nakon sabiranja sa 2



INDIREKTNO ADRESIRANJE

4. Koeficijent indirektno

- Koristi isti postupak generisanja adrese kao kod `CDP`.
- Koristi se u instrukcijama koje imaju tri memorijska operanda, npr. kod FIR filtriranja.

Primer:

```
mpy *AR1+, *CDP+, AC0 ::  mpy *AR2+, *CDP+, AC1
```

Vrednost sa memorijske lokacije na koju pokazuje `AR1` je pomnožena sa vrednošću sa memorijske lokacije na koju pokazuje `CDP` i upisana u `AC0`. Istovremeno se izvršava i druga instrukcija. Nakon toga se inkrementira `AR0`, `AR1` i `CDP`.

INDIREKTNO ADRESIRANJE

Modifikacije pokazivača

Operand	ARn/CDP pointer modifications
*ARn or *CDP	ARn (or CDP) is not modified.
*ARn± or *CDP±	ARn (or CDP) is modified after the operation by: ±1 for 16-bit operation (ARn=ARn±1) ±2 for 32-bit operation (ARn=ARn±2)
*ARn(#k16) or *CDP(#k16)	ARn (or CDP) is not modified. The signed 16-bit constant k16 is used as the offset for the base pointer ARn (or CDP).
*+ARn(#k16) or *+CDP(#k16)	ARn (or CDP) is modified before the operation. The signed 16-bit constant k16 is added as the offset to the base pointer ARn (or CDP) before generating new address.
*(ARn±T0/T1)	ARn is modified after the operation by ±16-bit content in T0 or T1, (ARn = ARn±T0/T1)
*ARn(T0/T1)	ARn is not modified. T0 or T1 is used as the offset for the base pointer ARn.

Memorijski mapirani registri

- Nalaze se na adresama `0x0 - 0x5F` na prvoj strani.
- Najčešće korišćeni registri su `AR0-AR7`, akumulatori `AC0-AC3`, privremeni registri `T0-T3`, razni `BSA` (*buffer start address*) registri... Svi `MMR` su nabrojani u *C55x v3.x CPU Reference Guide* - `http://www.ti.com/lit/ug/swpu073e/swpu073e.pdf`
- Mogu da se adresiraju direktno, indirektno ili apsolutno.

Primer:

```
mov mmap(AR6), BSA67;          BSA67 = vrednost u AR6
mov AR1, AR5;                          AR5 = AR1
```

APSOLUTNO ADRESIRANJE

- Koristi se za pristup memoriji na poznatoj adresi.
- Može biti 16-bitno sa `*abs16(#k16)` i `DPH` ili 23-bitno sa pristupom `*(#k23)`.
- Za apsolutno adresiranje periferijske memorije koristi se `port(#k16)`.

Primer:

```
mov *abs16(#2002h), T2    ; Pod pretpostavkom da je DPH = 0x03.
```

U `T2` se upisuje podatak sa adrese `0x032002`.

```
mov *(0x011234), T2 ; prebacuje podatak koji se nalazi na strani
```

`0x01` na memorijskoj lokaciji `0x1234` u registar `T2`

KRUŽNO ADRESIRANJE

- Služi za implementaciju cirkularnih bafera.
- Registri `AR0-AR7` i `CDP` mogu biti konfigurisani da rade linearno ili cirkularno.
- Setovanjem bita `ARnLC` se označava cirkularni mod rada. (`BSET ARnLC;`
- Dužina kružnog adresiranja zavisi od vrednosti registara `BK03`, `BK47` i `BKC`.
- Veličina bafera je data u `BK03` ako se koristi neki od registara `AR0-AR3`, `BK47` za registre `AR4-AR7` i `BKC` za `CDP`.

KRUŽNO ADRESIRANJE

Pointer	Linear/Circular Configuration Bit	Supplier of Main Data Page	Buffer Start Address Register	Buffer Size Register
AR0	ST2_55(0) = AR0LC	AR0H	BSA01	BK03
AR1	ST2_55(1) = AR1LC	AR1H	BSA01	BK03
AR2	ST2_55(2) = AR2LC	AR2H	BSA23	BK03
AR3	ST2_55(3) = AR3LC	AR3H	BSA23	BK03
AR4	ST2_55(4) = AR4LC	AR4H	BSA45	BK47
AR5	ST2_55(5) = AR5LC	AR5H	BSA45	BK47
AR6	ST2_55(6) = AR6LC	AR6H	BSA67	BK47
AR7	ST2_55(7) = AR7LC	AR7H	BSA67	BK47
CDP	ST2_55(8) = CDPLC	CDPH	BSAC	BKC

KRUŽNO ADRESIRANJE

Primer:

```
amov    #COEFF,XAR2     ; Main data page for COEFF[4]
mov     #COEFF,BSA23    ; Buffer base address is COEFF[0]
mov     #0x4,BK03       ; Set buffer size of 4 words
mov     #2,AR2          ; AR2 points to COEFF[2]
bset    AR2LC           ; AR2 is configured as circular pointer
mov     *AR2+,T0        ; T0 is loaded with COEFF[2]
mov     *AR2+,T1        ; T1 is loaded with COEFF[3]
mov     *AR2+,T2        ; T2 is loaded with COEFF[0]
mov     *AR2+,T3        ; T3 is loaded with COEFF[1]
```

PARALELIZAM

- Postojanje više magistrala omogućava izvršavanje dve instrukcije paralelno. Instrukcije ne smeju da imaju više od 6 B ukupno.
- Može biti implicitan i eksplicitan paralelizam.
- Eksplicitno zadate paralelne instrukcije - *user-built* imaju `||` između.
- Implicitan (*built-in*) paralelizam se označava sa `::` između instrukcija.

Primer:

```
mpym *AR1+, *AR2+, AC0
|| and AR4, T1
```

user-built paralelizam

```
mac *AR0-, *CDP-, AC0
::  mac *AR1-, *CDP-, AC1
```

built-in paralelizam

TMS320C55X CPU - INSTRUKCIJE

TMS320C55x DSP Mnemonic Instruction Set Reference Guide

Poglavlje `Cross-Reference of Algebraic and Mnemonic Instruction Sets`, strane 639-670.

`http://www.ti.com/lit/ug/spru374g/spru374g.pdf`

ARITMETIČKE INSTRUKCIJE

- Instrukcije su: `ADD`, `SUB`, `MPY` i proširene `MAC` - *multiply-and-accumulate*, `MAS` - *multiply-subtraction*.
- Mogu biti dodati sufiksi proširene preciznosti, zasićenja, `add-with-carry`, `subtract-with-borrow`...

Primer:

```
mpym *AR1+, *CDP-, AC0;
```

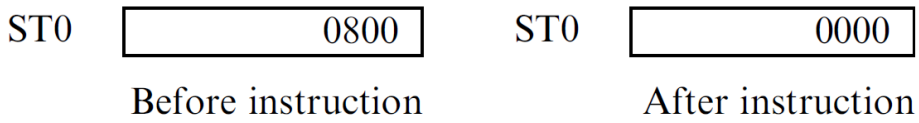
	Before instruction		After instruction
AC0	FF FFFF FF00	AC0	00 0000 0020
FRC	0	FRC	0
AR1	02E0	AR1	02E1
CDP	0400	CDP	03FF
Data memory		Data memory	
0x2E0	0002	0x2E0	0002
0x400	0010	0x400	0010

LOGIČKE INSTRUKCIJE

- NOT, AND, OR, XOR
- BSET, BCLR, BTSTP

Primer:

```
bclr #11, ST0;
```



MOVE INSTRUKCIJA

Primer:

```
move uns(rnd(HI(saturate(AC0 « T2)))), *AR1+;
```

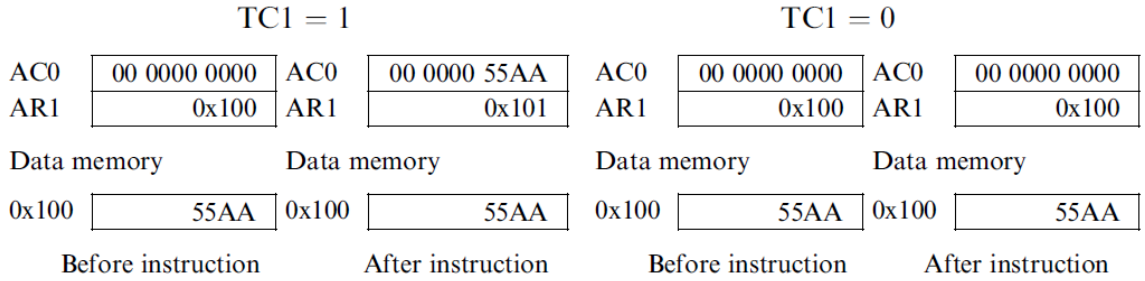
	Before instruction		After instruction
AC0	00 0FAB 8678	AC0	00 0FAB 8678
AR1	0x100	AR1	0x101
T2	0x2	T2	0x2
Data memory		Data memory	
0x100	1234	0x100	3EAE

PROGRAM FLOW INSTRUKCIJE

- Kontrolišu tok izvršavanja programa.
- To su instrukcije grananja `B`, poziv i povratak iz *subroutine*-e `CALL` i `RET`, *loop* operacija `RPTB`.
- Instrukcije uslovnog grananja `BCC` i povratka `RETCC` kontrolišu izvršavanje programa pod zadatim uslovima.
- Instrukcija uslovnog izvršavanja dela programa je `XCC`.

Primer:

```
      xcc label, TC1;
      mov *AR1+, AC0;
label
```



PRIMERI RADA U CODE COMPOSER STUDIO-U

1. Getting started with CCS and eZdsp
2. File Input and Output
3. User Interface for eZdsp
4. Audio Loopback Using eZdsp