TABLICA NAREDABA PROCESORA ARM (za AR1R)

	Naziv	Asembler	Operacija	N	Z	C	V
Obrada podataka	**Aritmetičke naredbe**						
	Add	ADD{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn + oprnd2	n	z	c	v
	Add with carry	ADC{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn + oprnd2 + C				
	Subtract	SUB{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn - oprnd2				
	Subtract with carry	SBC{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn - oprnd2 - not(C)				
	Reverse subtract	RSB{cond}{S} Rd, Rn, <oprnd2>	Rd := oprnd2 - Rn				
	Reverse subtract with carry	RSC{cond}{S} Rd, Rn, <oprnd2>	Rd := oprnd2 - Rn - not(C)				
	Logičke naredbe						
	Bitwise AND	AND{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn AND oprnd2	n	z	x	-
	Bitwise inclusive OR	ORR{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn OR oprnd2				
	Bitwise Exclusive OR	EOR{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn XOR oprnd2				
	Bit clear	BIC{cond}{S} Rd, Rn, <oprnd2>	Rd := Rn AND NOT(oprnd2)				
	Ispitivanje vrijednosti						
	Compare	CMP{cond} Rn, <oprnd2>	Rn - oprnd2 i osvježi CPSR	n	z	c	v
	Compare negative	CMN{cond} Rn, <oprnd2>	Rn + oprnd2 i osvježi CPSR				
	Test	TST{cond} Rn, <oprnd2>	Rn and oprnd2 i osvježi CPSR	n	z	x	-
	Test equivalence	TEQ{cond} Rn, <oprnd2>	Rn xor oprnd2 i osvježi CPSR				
	Punjenje registra						
	Move	MOV{cond}{S} Rd, <oprnd2>	Rd := oprnd2	n	z	x	-
	Move not	MVN{cond}{S} Rd, <oprnd2>	Rd := NOT(oprnd2)				
	No operation	NOP	R0 := R0				
	Množenje						
	Multiply	MUL{cond}{S} Rd, Rm, Rs	Rd := (Rm * Rs)[31:0]	n	z	?	-
	Multiply accumulate	MLA{cond}{S} Rd, Rm, Rs, Rn	Rd := (Rm * Rs + Rn)[31:0]				
CPSR	**Naredbe za registar stanja**						
	Move CPSR to register	MRS{cond} Rd, CPSR	Rd := CPSR				
	Move register to CPSR	MSR{cond} CPSR, Rd	CPSR := Rd				
	Move immediate to CPSR	MSR{cond} CPSR, #<imm_8r>	CPSR := imm_8r				
Skokovi	**Naredbe skoka**						
	Branch	B{cond} address	PC := address				
	Branch and link	BL{cond} address	LR := PC-4; PC := address				
	Druge naredbe kao upravljačke naredbe						
	Software interrupt	SWI 0x123456	Završava simulaciju u SSPARCSS simulatoru				
Memorijske	**Prijenos jednog podatka - 1. grupa**						
	Load word	LDR{cond} Rd, <mode_2>	Rd := mem_32[mode_2]				
	Store word	STR{cond} Rd, <mode_2>	mem_32[mode_2] := Rd				
	Load byte	LDR{cond}B Rd, <mode_2>	Rd := zero_ext mem_8[mode_2]				
	Store byte	STR{cond}B Rd, <mode_2>	mem_8[mode_2] := Rd[7:0]				
	Prijenos jednog podatka - 2. grupa						
	Load halfword	LDR{cond}H Rd, <mode_3>	Rd := zero_ext mem_16[mode_3]				
	Store halfword	STR{cond}H Rd, <mode_3>	mem_16[mode_3] := Rd[15:0]				
	Load signed halfword	LDR{cond}SH Rd, <mode_3>	Rd := sign_ext mem_16[mode_3]				
	Load signed byte	LDR{cond}SB Rd, <mode_3>	Rd := sign_ext mem_8[mode_3]				
	Prijenos bloka podataka i stog						
	Load multiple	LDM{cond}<mode_4L> Rd{!},<reglist>	Napuni listu registara iz bloka počevši od mem[Rd]				
	Store multiple	STM{cond}<mode_4S> Rd{!},<reglist>	Spremi listu registara u blok s početkom od mem[Rd]				

{cond} = uvjet u naredbama

Naziv	Asembler	Uvjet
Equal	EQ	Z
Not equal	NE	**!**Z
Minus / Negative	MI	N
Plus / Positive	PL	**!**N
Overflow set	VS	V
Overflow clear	VC	**!**V
Carry set / Unsigned higher or same	CS/HS	C
Carry clear / Unsigned lower	CC/LO	**!**C
Unsigned higher	HI	C and **!**Z
Unsigned lower or same	LS	**!**C or Z
Signed greater than or equal	GE	N == V
Signed less than	LT	N **!**= V
Signed greater than	GT	**!**Z and (N == V)
Signed less than or equal	LE	Z or (N **!**= V)
Always true	AL/ništa	-

Oznaka	Objašnjenje
R*	Jedan od registara R0-R15
<reglist>	Zarezima odvojena lista registara i/ili raspona registara. Piše se u vitičastim zagradama. Npr. {R9,R2-R4,R1}
{!}	Opcionalni znak ! u naredbama LDM/STM. Ako se napiše ! onda naredba mijenja registar Rd
{S}	Opcionalni sufiks S. Ako se napiše S, onda naredba mijenja zastavice u CPSR-u
mem_N[A]	Označava N-bitni podatak u memoriji na adresi A
zero_ext	Operacija proširivanja ništicama do širine od 32 bita
sign_ext	Operacija predznačnog proširivanja do širine od 32 bita
n, z, c, v	Stanja ALU koja se kopiraju u zastavice. Redom: negative (isto što i sign), zero, carry i overflow
?, -, x	Stanja koja se kopiraju u zastavice. ? = nedefinirano stanje, - = zastavica se ne mijenja, x = izlaz iz barell-shiftera ili bez promjene ako nema pomaka/rotacije

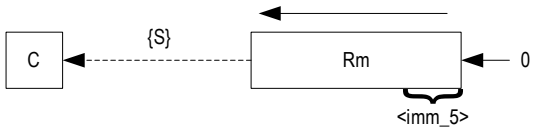
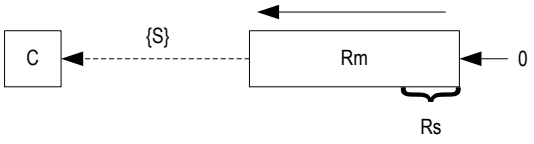
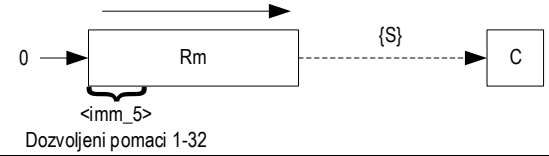
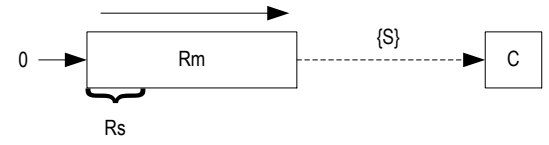
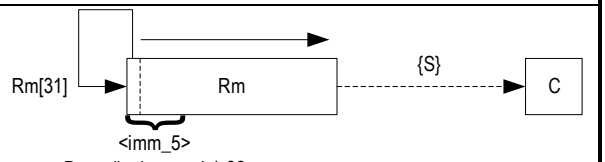
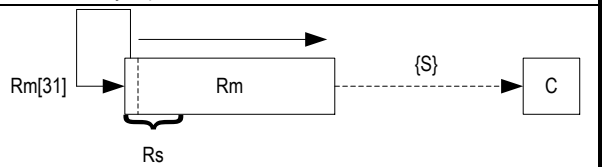
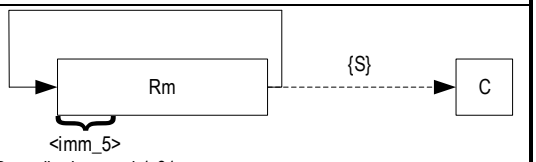
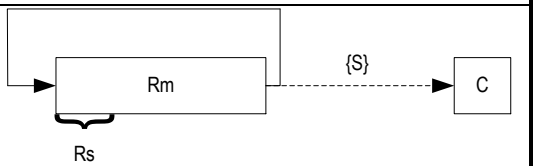
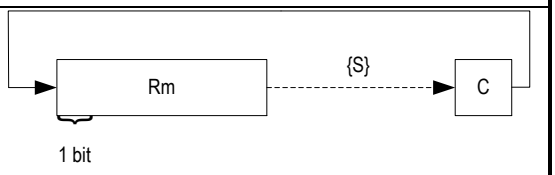
Statusni registar CPSR i načini rada (mode)

31	30	29	28	...	7	6	5	4:0
N	**Z**	**C**	**V**	-	**I**	**F**	-	**M**

I i F = maskiranje prekida IRQ i FIQ (1 znači maskiran)

M = processor mode: SVC (10011), IRQ (10010), FIQ (10001), USR (10000), SYS (11111), ABT (10111), UND (11011)

<oprnd2> - način adresiranja 1

Operand i adresiranje	Objašnjenje
Rm Registarsko	
#<imm_8r> Neposredna vrijednost (sastoji se od dva polja <imm_8> i <rotate_imm> od kojih se dobije 32-bitna konstanta)	8 bitova se koristi za neposrednu vrijednost <imm_8>, 4 bita <rotate_imm> definiraju broj rotacija udesno 8-bitne neposredne vrijednosti: <imm_8> rotirano udesno za (2*<rotate_imm>)
Rm, LSL #<imm_5> Registarsko s neposrednim logičkim pomakom u lijevo (LSL=ASL)	
Rm, LSL Rs Registarsko s registarskim logičkim pomakom u lijevo (LSL=ASL)	
Rm, LSR #<imm_5> Registarsko s neposrednim logičkim pomakom u desno	
Rm, LSR Rs Registarsko s registarskim logičkim pomakom u desno	
Rm, ASR #<imm_5> Registarsko s neposrednim aritmetičkim pomakom u desno	
Rm, ASR Rs Registarsko s registarskim aritmetičkim pomakom u desno	
Rm, ROR #<imm_5> Registarsko s neposrednim rotiranjem u desno	
Rm, ROR Rs Registarsko s registarskim rotiranjem u desno	
Rm, RRX Registarsko s proširenim rotiranjem u desno	

<imm_N> = broj ili labela čija vrijednost je pozitivan N-bitni broj u opsegu od 0 do 2^N-1

<mode_2> - način adresiranja 2

Naziv	Asembler
Obično indeksiranje (bez mijenjanja Rn)	
Neposredni odmak	[Rn, #±<imm_12>]
Bez odmaka	[Rn]
S labelom	Labela
Registarski odmak	[Rn, ±Rm]
Regist. skalirani odmak	[Rn, ±Rm, <SHIFT> #<imm_5>]
Predindeksiranje (mijenja Rn zbog !)	
Neposredni odmak	[Rn, #±<imm_12>] !
Registarski odmak	[Rn, ±Rm] !
Regist. skalirani odmak	[Rn, ±Rm, <SHIFT> #<imm_5>] !
Postindeksiranje (uvijek mijenja Rn)	
Neposredni odmak	[Rn], #±<imm_12>
Registarski odmak	[Rn], ±Rm
Regist. skalirani odmak	[Rn], ±Rm, <SHIFT> #<imm_5>

<mode_3> - način adresiranja 3

Naziv	Asembler
Obično indeksiranje (bez mijenjanja Rn)	
Neposredni odmak	[Rn, #±<imm_8>]
Bez odmaka	[Rn]
S labelom	Labela
Registarski odmak	[Rn, ±Rm]
Predindeksiranje (mijenja Rn zbog !)	
Neposredni odmak	[Rn, #±<imm_8>] !
Registarski odmak	[Rn, ±Rm] !
Postindeksiranje (uvijek mijenja Rn)	
Neposredni odmak	[Rn], #±<imm_8>
Registarski odmak	[Rn], ±Rm

<mode_4> - način adresiranja 4

Asembler (Naziv)	Asembler (Naziv)
mode_4L (LDM, load block, pop)	
IA (increment after)	FD (full descending)
IB (increment before)	ED (empty descending)
DA (decrement after)	FA (full ascending)
DB (decrement before)	EA (empty ascending)
mode_4S (STM, store block, push)	
IA (increment after)	EA (empty ascending)
IB (increment before)	FA (full ascending)
DA (decrement after)	ED (empty descending)
DB (decrement before)	FD (full descending)

Skraćena heksadekadska tablica ASCII-kodova (kontrolni znakovi imaju kodove od 0 do 1F i 7F)

NUL	BEL	BS	TAB	LF	CR	ESC	US		!	"	#	$
00	07	08	09	0A	0D	1B	1F	20	21	22	23	24
%	&	'	(	)	*	+	,	-	.	/	0	1
25	26	27	28	29	2A	2B	2C	2D	2E	2F	30	31
2	3	4	5	6	7	8	9	:	;	<	=	>
32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E
?	@	A	B	C	D	E	F	G	H	I	J	K
3F	40	41	42	43	44	45	46	47	48	49	4A	4B
L	M	N	O	P	Q	R	S	T	U	V	W	X
4C	4D	4E	4F	50	51	52	53	54	55	56	57	58
Y	Z	[	\	]	^	_	`	a	b	c	d	e
59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65
f	g	h	i	j	k	l	m	n	o	p	q	r
66	67	68	69	6A	6B	6C	6D	6E	6F	70	71	72
s	t	u	v	w	x	y	z	{	|	}	~	DEL
73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F