

# Logični ukazi (delo z določenimi biti)

**and** r1, r2, r3 @brisanje z ničlo v maski določenih bitov

r2	00000000	00000000	00000000	01010011
and r3	11111111	11111111	11111111	11001010
=r1	00000000	00000000	00000000	01000010

**bic** r1, r2, r3 @brisanje z enico v maski določenih bitov

r2	00000000	00000000	00000000	01010011
bic r3	11111111	11111111	11111111	11001010
=r1	00000000	00000000	00000000	00010001

**orr** r1, r2, r3 @postavljanje z enico v maski določenih bitov

r2	00000000	00000000	00000000	01010011
or r3	11111111	11111111	11111111	11001010
=r1	11111111	11111111	11111111	11011011

**eor** r1, r2, r3 @invertiranje z enico v maski določenih bitov

r2	00000000	00000000	00000000	01010011
eor r3	11111111	11111111	11111111	11001010
=r1	11111111	11111111	11111111	10011001

# Logični ukazi (preverjanje stanja določenih bitov)

- Preverjanje stanja enega bita (določen je z enico v maski)

**tst r1, r2** @zastavice postavi glede na r1 AND r2

r1	00000000	00000000	00000000	01010011
tst r2	00000000	00000000	00000000	00000100
=	00000000	00000000	00000000	00000000

→ Z=1

r1	00000000	00000000	00000000	01010011
tst r2	00000000	00000000	00000000	00000010
=	00000000	00000000	00000000	00000010

→ Z=0

- Preverjanje stanja večih bitov:

- Najprej izločimo bite, ki nas zanimajo: and
- Primerjamo z želenim stanjem bitov (bite, ki nas ne zanimajo, primerjamo z 0)

## Zgled:

@preveri, da je bit7 v r1 enak 0 in bit 2 v r1 enak 1  
and r2, r1, #0x84      @0x84 = 00...010000100 => r2 = 00..00?0000?00  
cmp r2, #0x04      @0x04 = 00...000000100; ustreza, če Z=1

# Aritmetično-logični ukazi, seznam

- Aritmetični ukazi:

add r0, r1, r2 ; r0 <- r1 + r2	
adc r0, r1, r2 ; r0 <- r1 + r2 + C	(add with C)
sub r0, r1, r2 ; r0 <- r1 - r2	
sbc r0, r1, r2 ; r0 <- r1 - r2 + C - 1	(-not(C) = - (1-C) = C-1)
rsb r0, r1, r2 ; r0 <- r2 - r1	(reverse subtract)
rsc r0, r1, r2 ; r0 <- r2 - r1 + C - 1	(rev. sub -not(C) )

- Logični ukazi:

and r0, r1, r2 ; r0 <- r1 AND r2	
orr r0, r1, r2 ; r0 <- r1 OR r2	
eor r0, r1, r2 ; r0 <- r1 XOR r2	
bic r0, r1, r2 ; r0 <- r1 AND NOT r2	

- Prenos med registri:

mov r0, r2 ; r0 <- r2	
mvn r0, r2 ; r0 <- NOT r2	

- Primerjave:

cmp r1, r2 ; set CPSR flags on r1 - r2	
cnn r1, r2 ; set CPSR flags on r1 + r2	
tst r1, r2 ; set CPSR flags on r1 AND r2	
teq r1, r2 ; set CPSR flags on r1 XOR r2	(equivalence test)