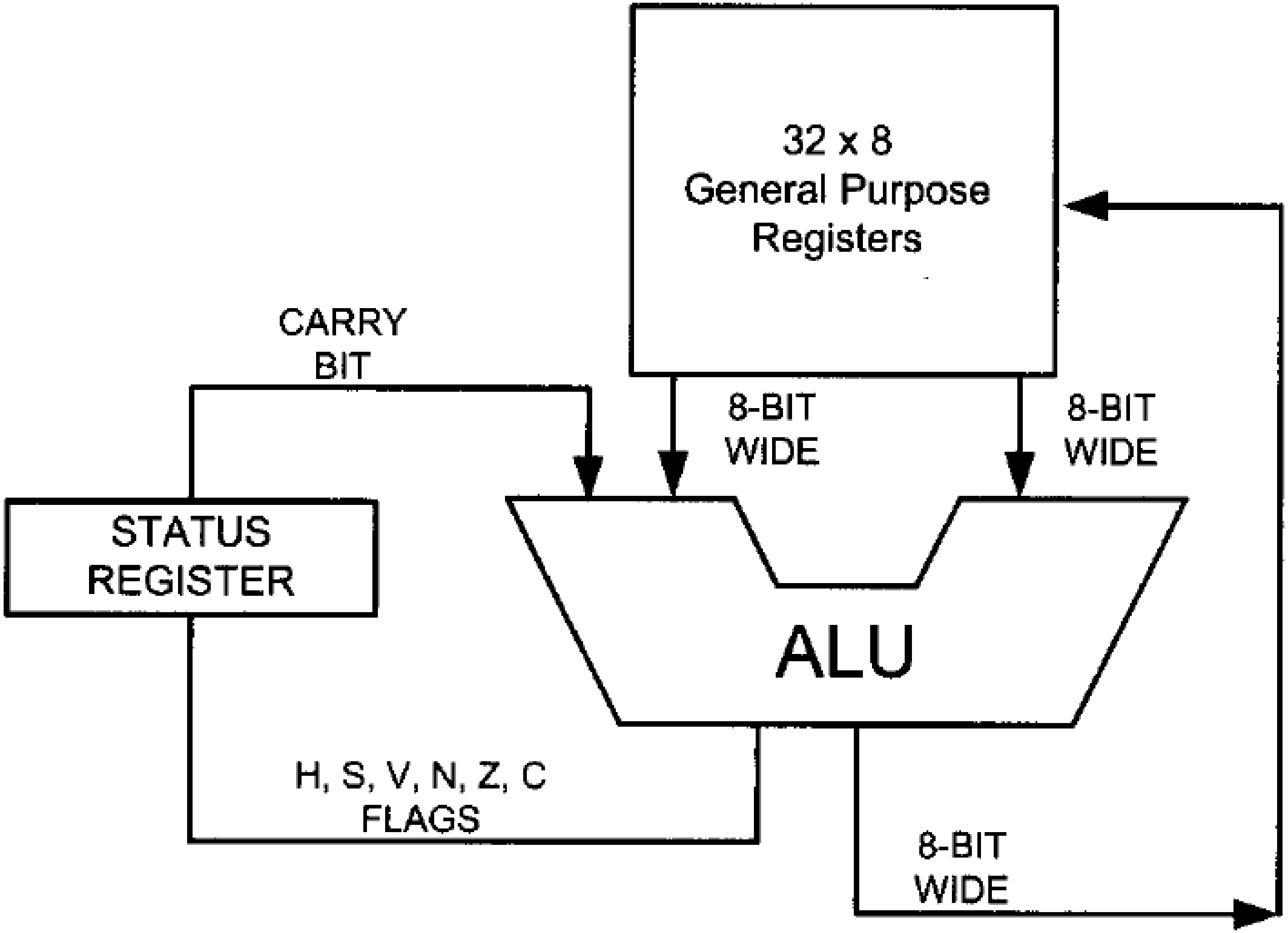
به نام خدا

آشنایی با زبان اسمبلی AVR

دستورات پایه

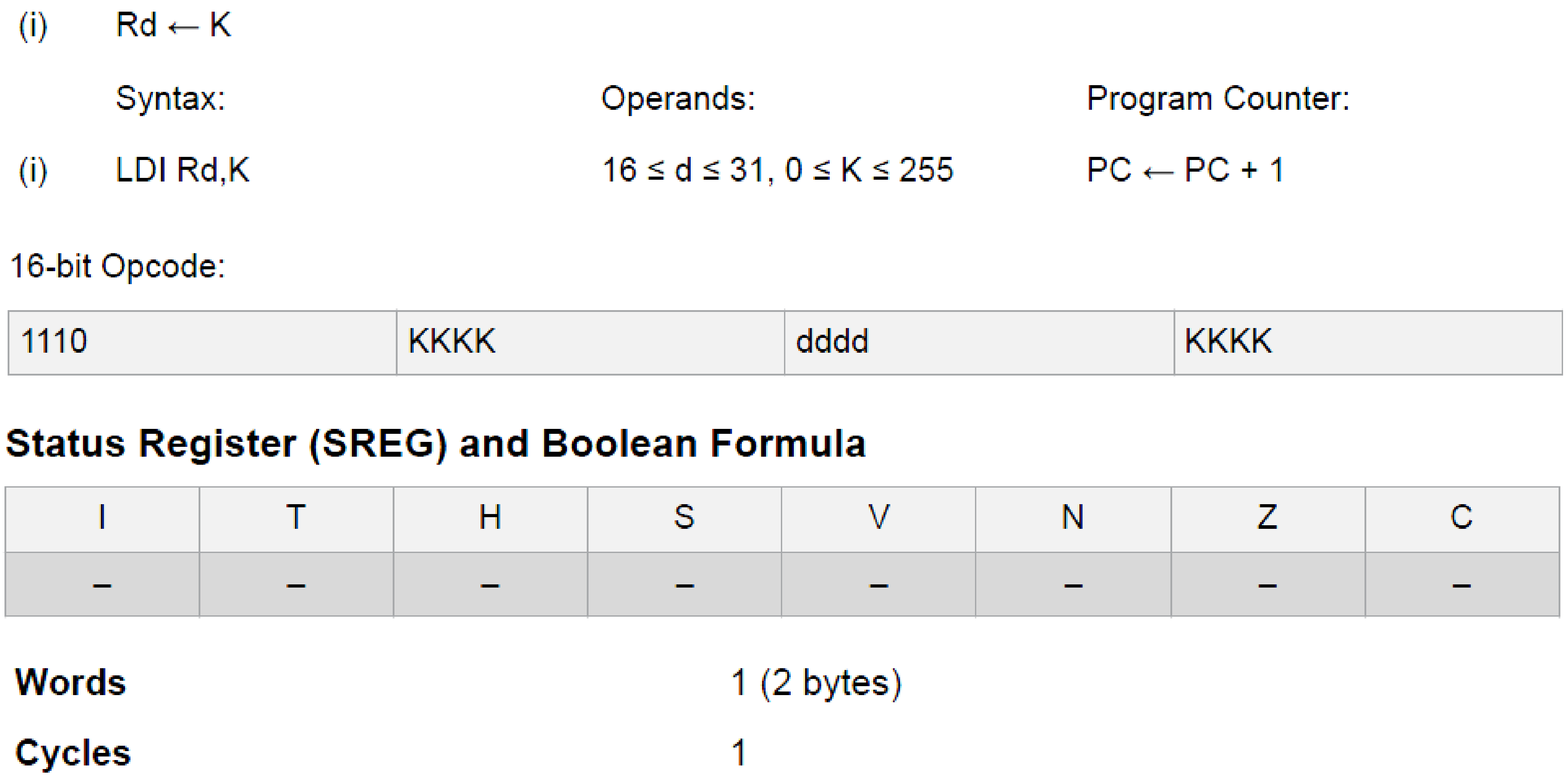
Dr. Aref Karimiafshar
A.karimiafshar@ec.iut.ac.ir

AVR GPR and ALU



LDI – Load Immediate

- Loads an 8-bit constant directly to register 16 to 31



(i) Rd ← K

	Syntax:	Operands:	Program Counter:
(i)	LDI Rd,K	$16 \le d \le 31$, $0 \le K \le 255$	PC ← PC + 1

16-bit Opcode:

1110	KKKK	dddd	KKKK

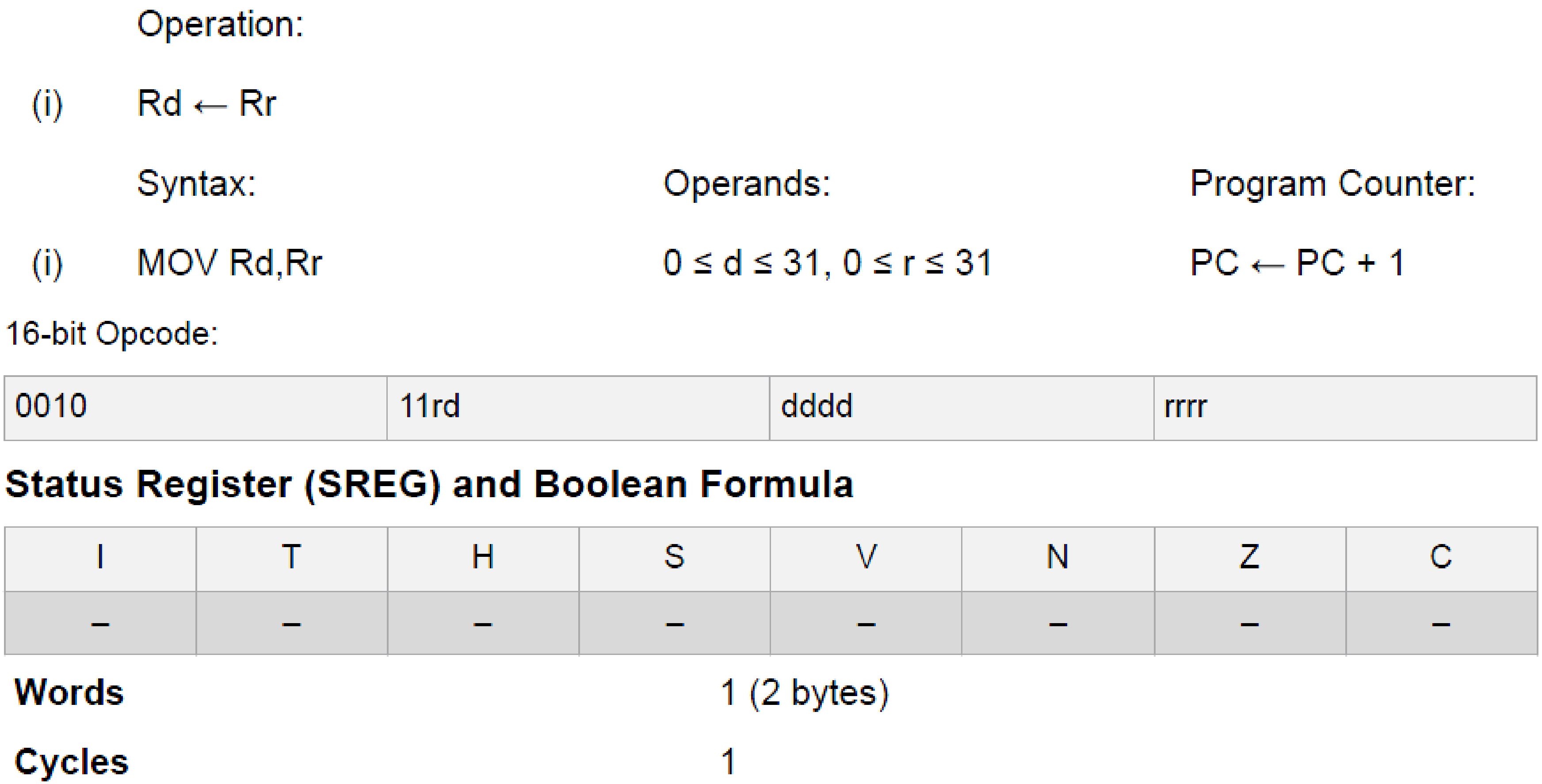
Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	–	–

Words	1 (2 bytes)
Cycles	1

MOV – Copy Register

This instruction makes a copy of one register into another. The source register Rr is left unchanged, while the destination register Rd is loaded with a copy of Rr.



Operation:

(i) $Rd \leftarrow Rr$

Syntax:	Operands:	Program Counter:
(i) MOV Rd,Rr	$0 \le d \le 31$, $0 \le r \le 31$	$PC \leftarrow PC + 1$

16-bit Opcode:

0010	11rd	dddd	rrrr

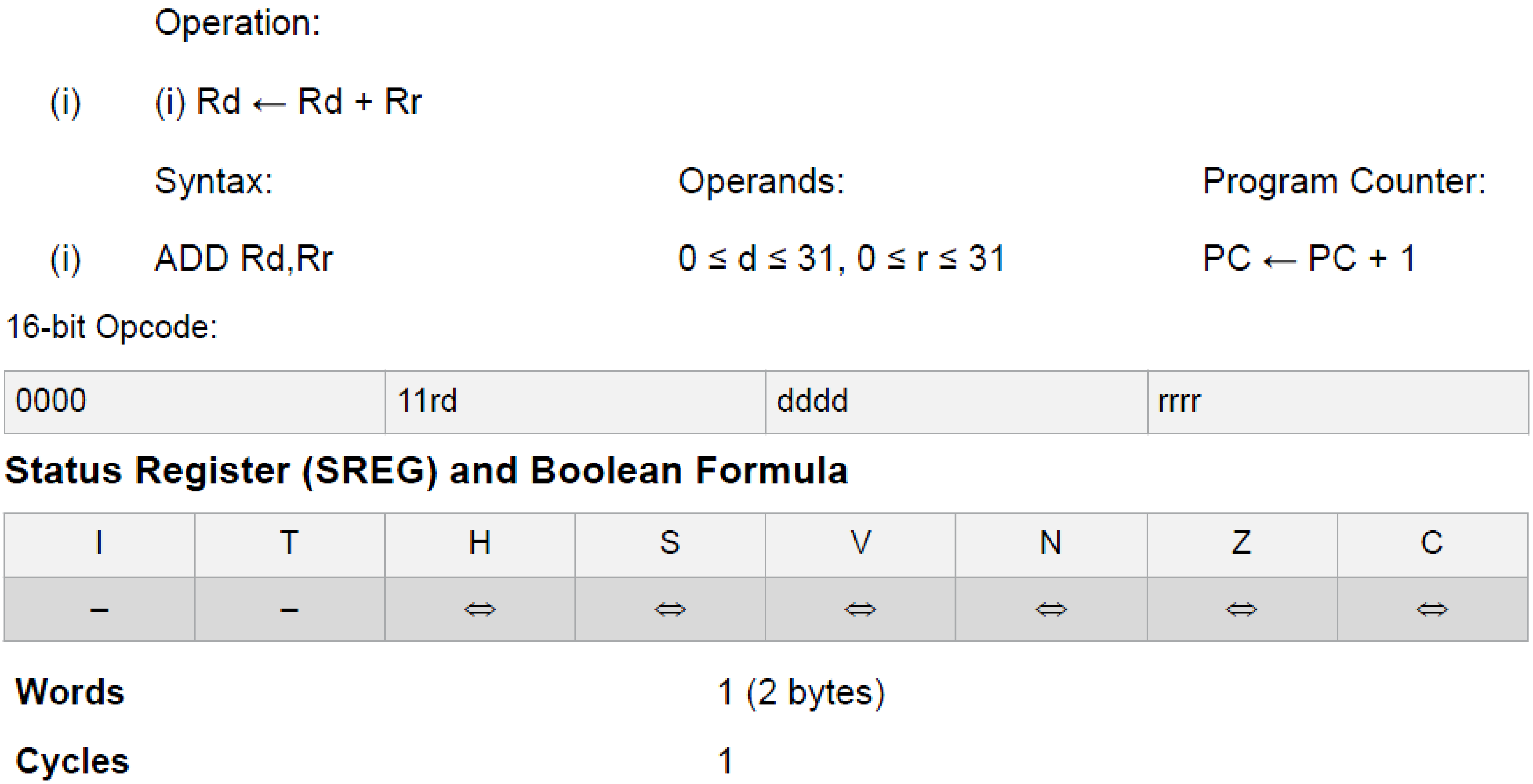
Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	–	–

Words	1 (2 bytes)
Cycles	1

ADD – Add without Carry

Adds two registers without the C Flag and places the result in the destination register Rd



Operation:

(i) (i) Rd ← Rd + Rr

	Syntax:	Operands:	Program Counter:
(i)	ADD Rd,Rr	$0 \le d \le 31$, $0 \le r \le 31$	PC ← PC + 1

16-bit Opcode:

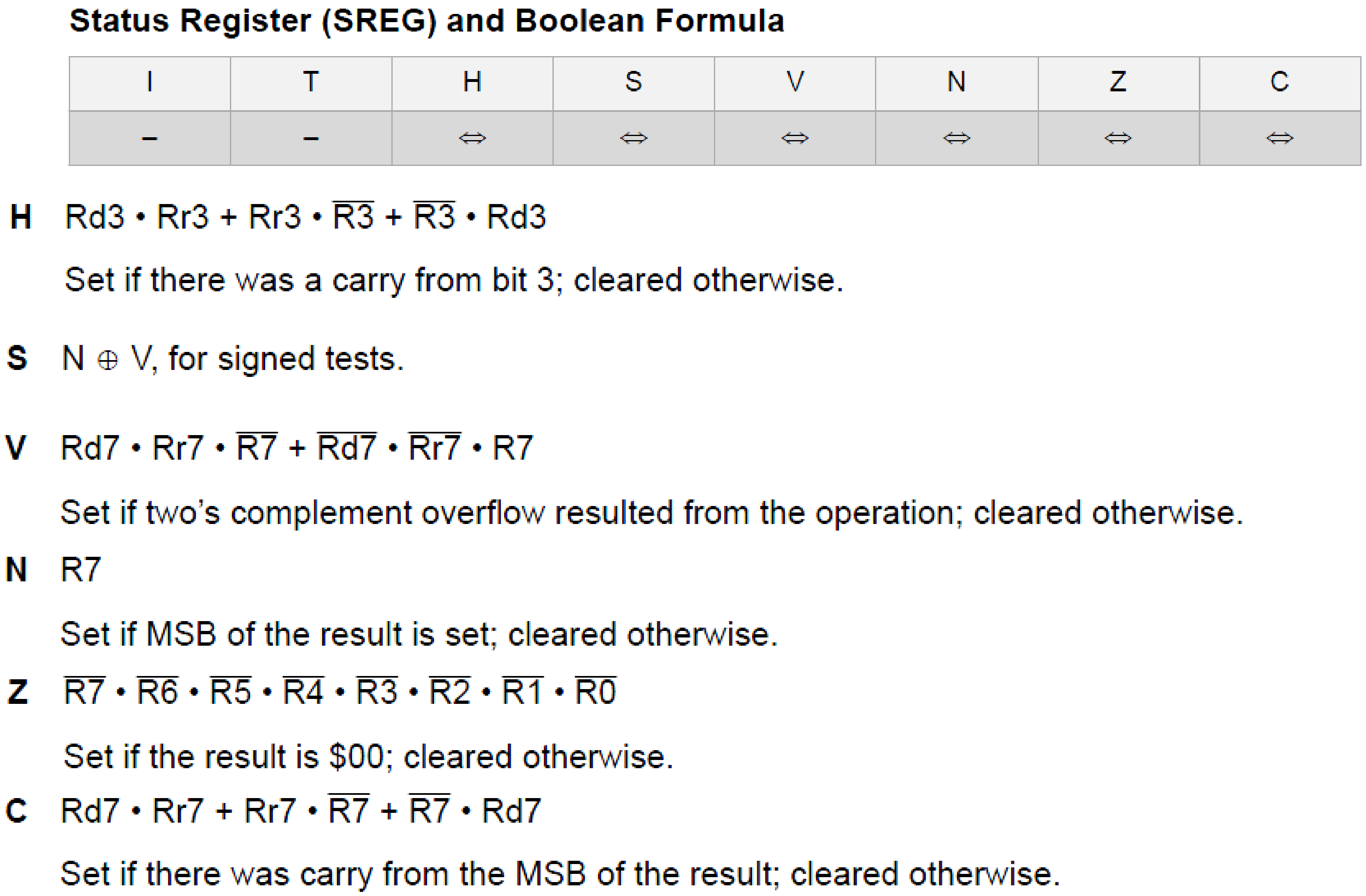
0000	11rd	dddd	rrrr

Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	⇔	⇔	⇔	⇔	⇔	⇔

Words	1 (2 bytes)
Cycles	1

ADD – Add without Carry



Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	⇔	⇔	⇔	⇔	⇔	⇔

H $Rd3 \cdot Rr3 + Rr3 \cdot \overline{R3} + \overline{R3} \cdot Rd3$

Set if there was a carry from bit 3; cleared otherwise.

S $N \oplus V$, for signed tests.

V $Rd7 \cdot Rr7 \cdot \overline{R7} + \overline{Rd7} \cdot \overline{Rr7} \cdot R7$

Set if two's complement overflow resulted from the operation; cleared otherwise.

N $R7$

Set if MSB of the result is set; cleared otherwise.

Z $\overline{R7} \cdot \overline{R6} \cdot \overline{R5} \cdot \overline{R4} \cdot \overline{R3} \cdot \overline{R2} \cdot \overline{R1} \cdot \overline{R0}$

Set if the result is \$00; cleared otherwise.

C $Rd7 \cdot Rr7 + Rr7 \cdot \overline{R7} + \overline{R7} \cdot Rd7$

Set if there was carry from the MSB of the result; cleared otherwise.

Comment

- Use " ; " to write a comment in Assembly language
  - Like "//" in C language
- To present number
  - In hex
    - $ or 0x
  - Decimal
    - No thing
  - Binary
    - 0b

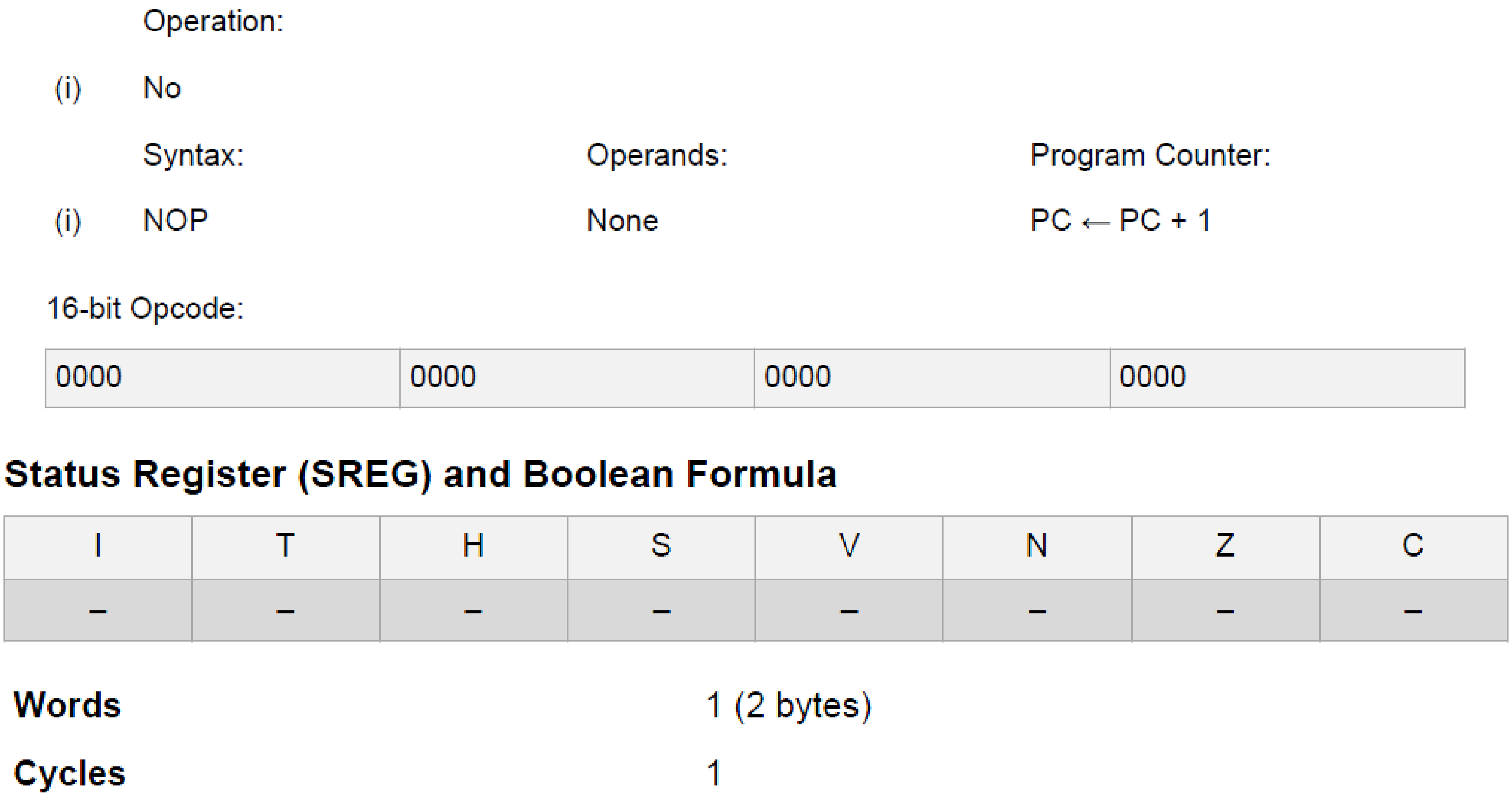
```
LDI R16,$50    LDI R16,0x50

LDI R16, 50

LDI R16,0b00110010
```

NOP – No Operation

- This instruction performs a single cycle No Operation



Operation:

(i) No

	Syntax:	Operands:	Program Counter:
(i)	NOP	None	PC ← PC + 1

16-bit Opcode:

0000	0000	0000	0000

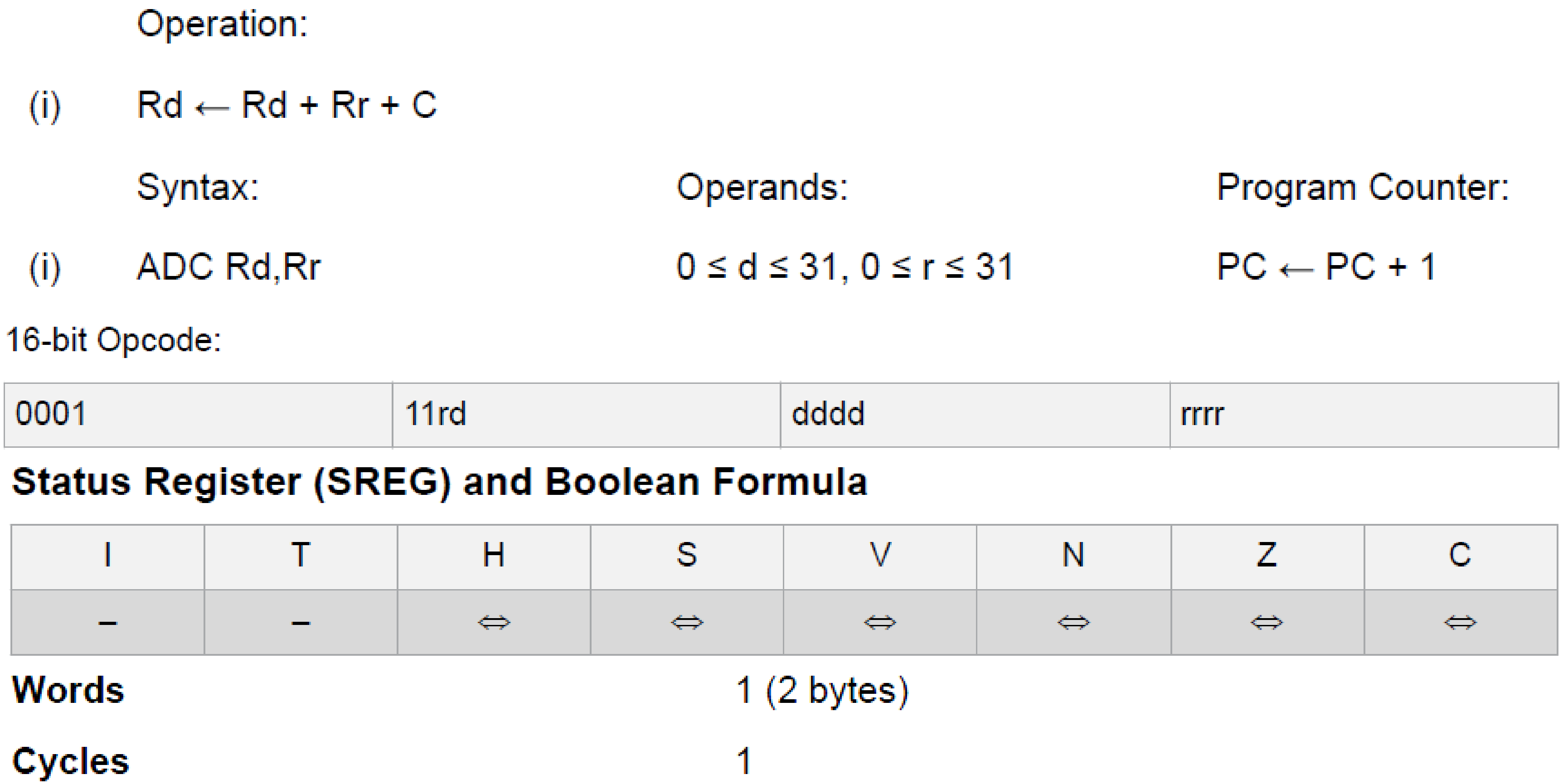
Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	–	–

Words	1 (2 bytes)
Cycles	1

ADC – Add with Carry

- Adds two registers and the contents of the C Flag and places the result in the destination register Rd



Operation:

(i) Rd ← Rd + Rr + C

	Syntax:	Operands:	Program Counter:
(i)	ADC Rd,Rr	0 ≤ d ≤ 31, 0 ≤ r ≤ 31	PC ← PC + 1

16-bit Opcode:

0001	11rd	dddd	rrrr

Status Register (SREG) and Boolean Formula

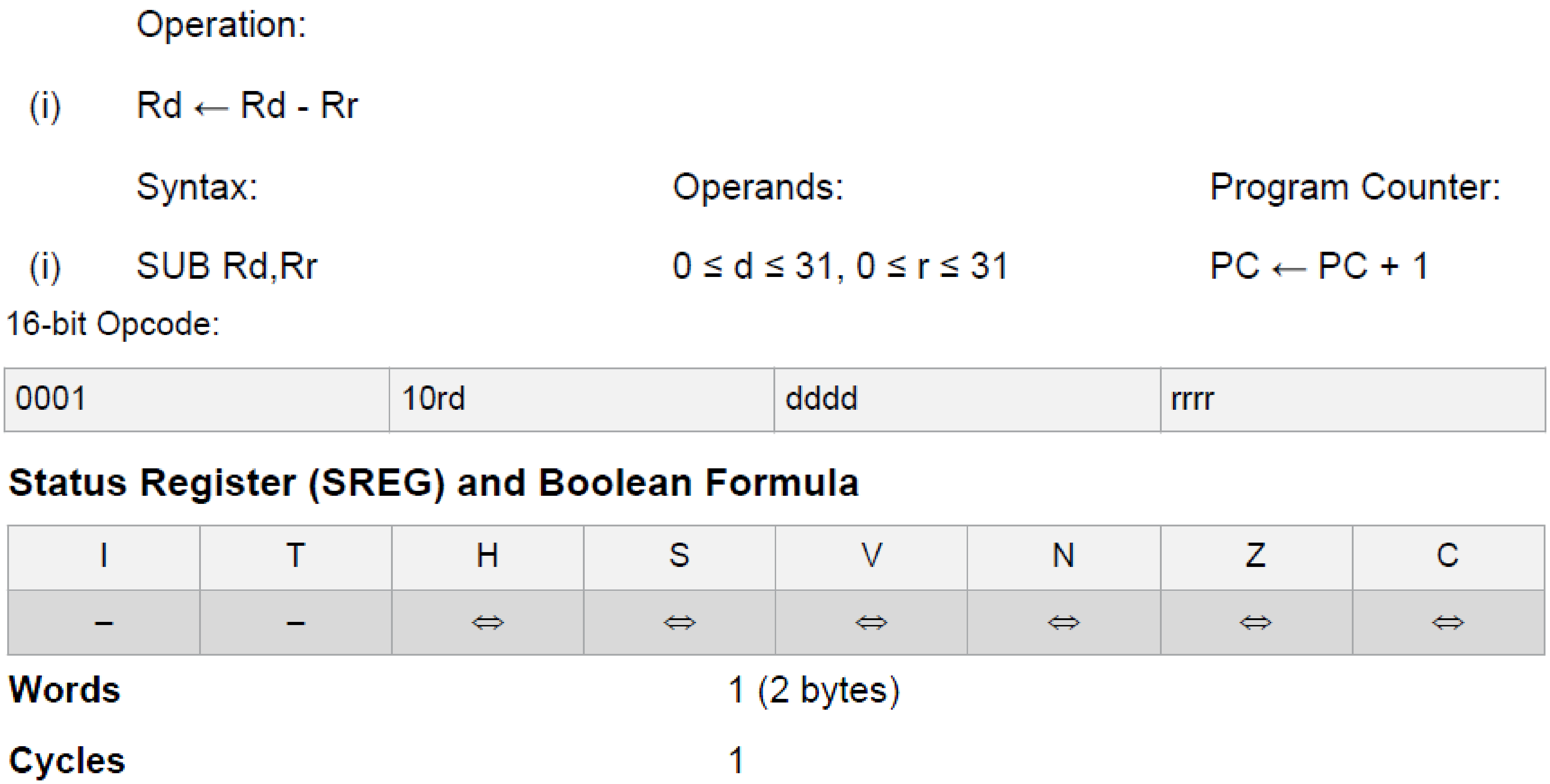
I	T	H	S	V	N	Z	C
–	–	⇔	⇔	⇔	⇔	⇔	⇔

Words 1 (2 bytes)

Cycles 1

SUB – Subtract Without Carry

- Subtracts two registers and places the result in the destination register Rd



Operation:

(i) Rd ← Rd - Rr

	Syntax:	Operands:	Program Counter:
(i)	SUB Rd,Rr	$0 \le d \le 31$, $0 \le r \le 31$	PC ← PC + 1

16-bit Opcode:

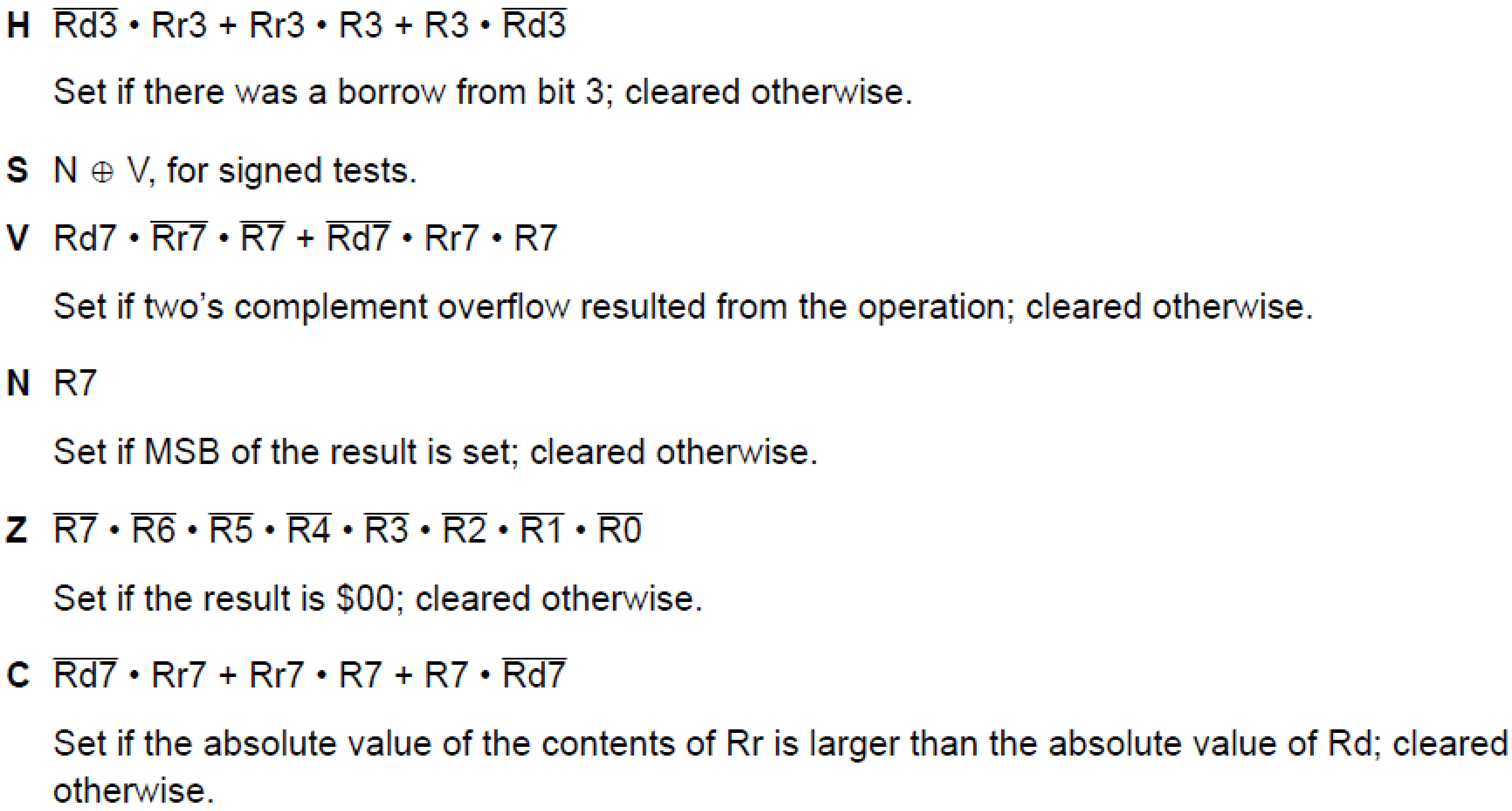
0001	10rd	dddd	rrrr

Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	⇔	⇔	⇔	⇔	⇔	⇔

Words	1 (2 bytes)
Cycles	1

SUB – Subtract Without Carry



H $\overline{\mathrm{Rd3}} \cdot \mathrm{Rr3} + \mathrm{Rr3} \cdot \mathrm{R3} + \mathrm{R3} \cdot \overline{\mathrm{Rd3}}$

Set if there was a borrow from bit 3; cleared otherwise.

S $\mathrm{N} \oplus \mathrm{V}$, for signed tests.

V $\mathrm{Rd7} \cdot \overline{\mathrm{Rr7}} \cdot \overline{\mathrm{R7}} + \overline{\mathrm{Rd7}} \cdot \mathrm{Rr7} \cdot \mathrm{R7}$

Set if two’s complement overflow resulted from the operation; cleared otherwise.

N R7

Set if MSB of the result is set; cleared otherwise.

Z $\overline{\mathrm{R7}} \cdot \overline{\mathrm{R6}} \cdot \overline{\mathrm{R5}} \cdot \overline{\mathrm{R4}} \cdot \overline{\mathrm{R3}} \cdot \overline{\mathrm{R2}} \cdot \overline{\mathrm{R1}} \cdot \overline{\mathrm{R0}}$

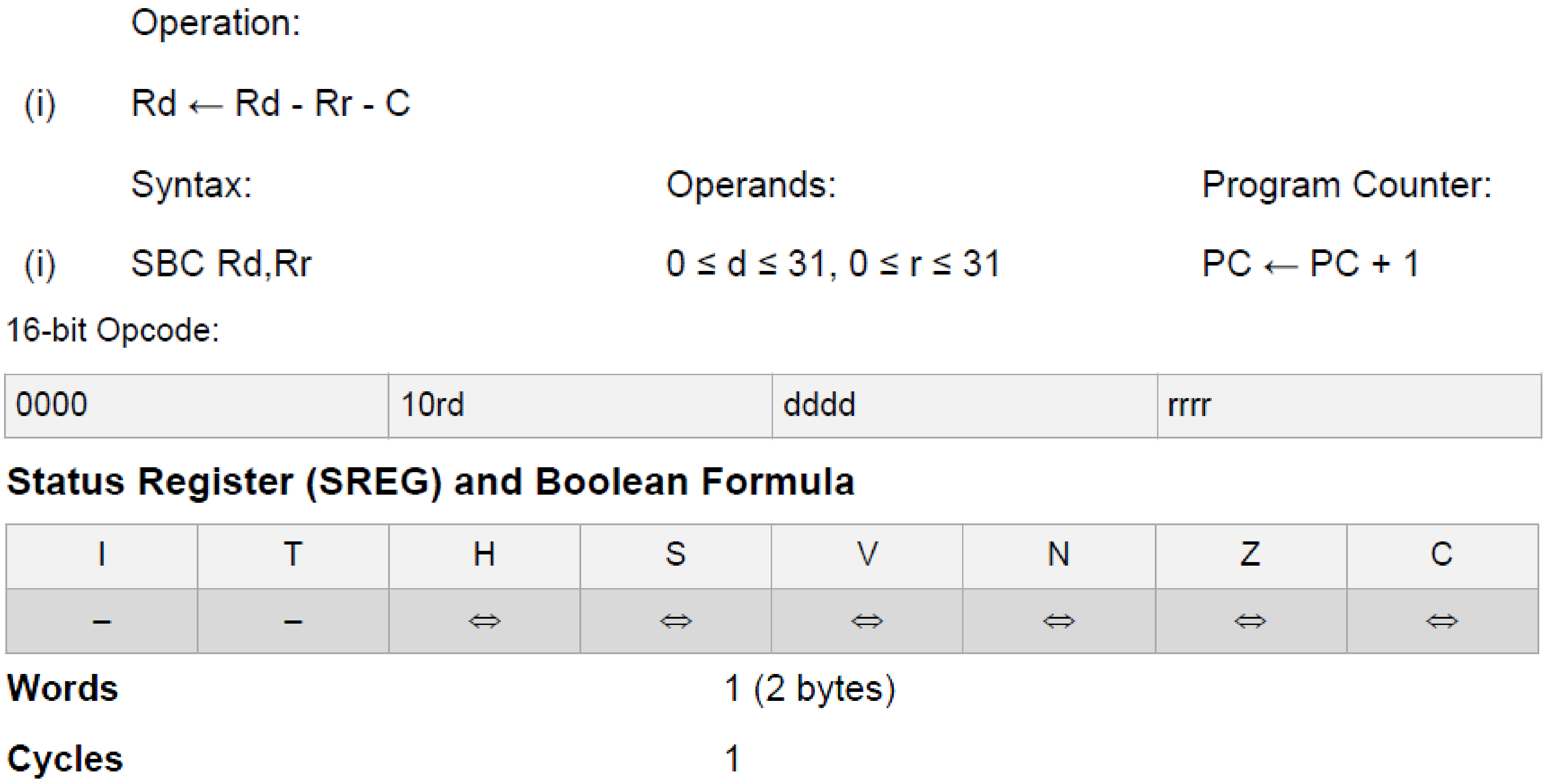
Set if the result is \$00; cleared otherwise.

C $\overline{\mathrm{Rd7}} \cdot \mathrm{Rr7} + \mathrm{Rr7} \cdot \mathrm{R7} + \mathrm{R7} \cdot \overline{\mathrm{Rd7}}$

Set if the absolute value of the contents of Rr is larger than the absolute value of Rd; cleared otherwise.

SBC – Subtract with Carry

- Subtracts two registers and subtracts with the C Flag, and places the result in the destination register Rd.



Operation:

(i) $Rd \leftarrow Rd - Rr - C$

	Syntax:	Operands:	Program Counter:
(i)	SBC Rd,Rr	$0 \le d \le 31, 0 \le r \le 31$	$PC \leftarrow PC + 1$

16-bit Opcode:

0000	10rd	dddd	rrrr

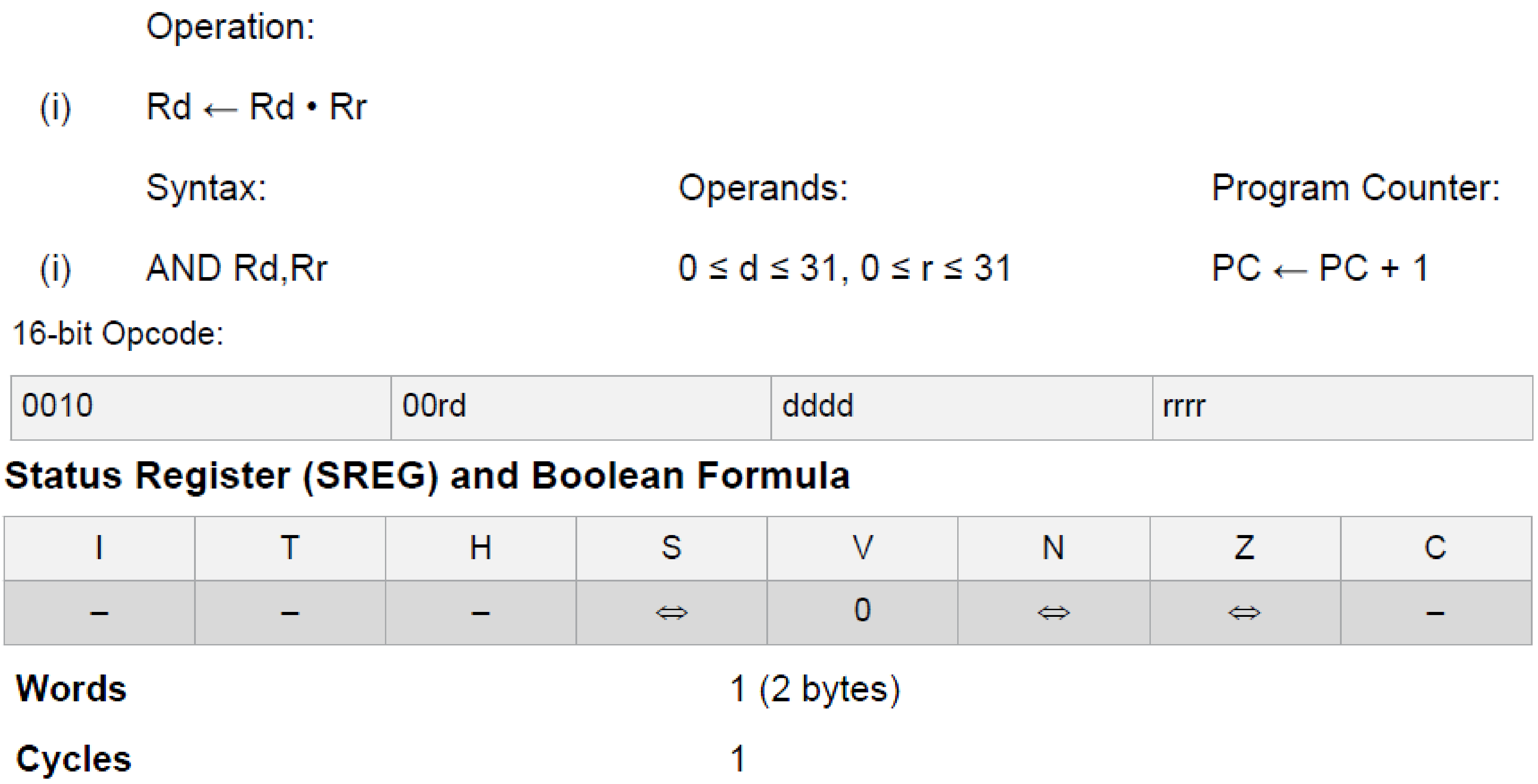
Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	⇔	⇔	⇔	⇔	⇔	⇔

Words	1 (2 bytes)
Cycles	1

AND – Logical AND

- Performs the logical AND between the contents of register Rd and register Rr, and places the result in the destination register Rd



Operation:

(i) Rd ← Rd • Rr

	Syntax:	Operands:	Program Counter:
(i)	AND Rd,Rr	$0 \le d \le 31$, $0 \le r \le 31$	PC ← PC + 1

16-bit Opcode:

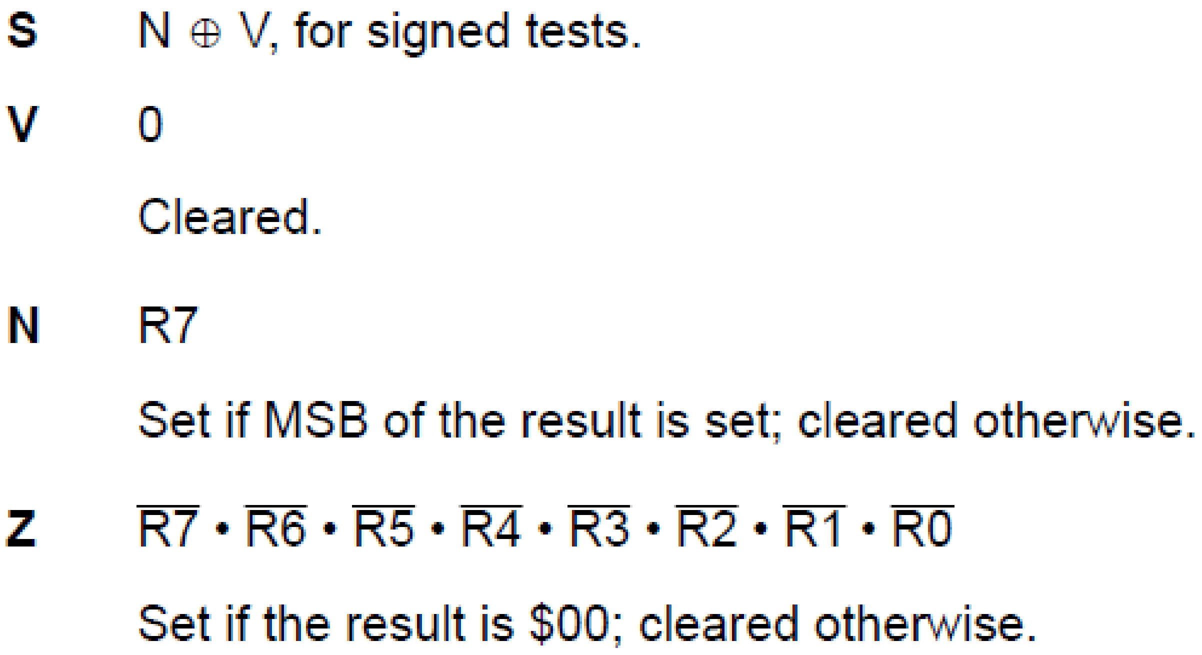
0010	00rd	dddd	rrrr

Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	⇔	0	⇔	⇔	–

Words	1 (2 bytes)
Cycles	1

AND – Logical AND



S N ⊕ V, for signed tests.

V 0

Cleared.

N R7

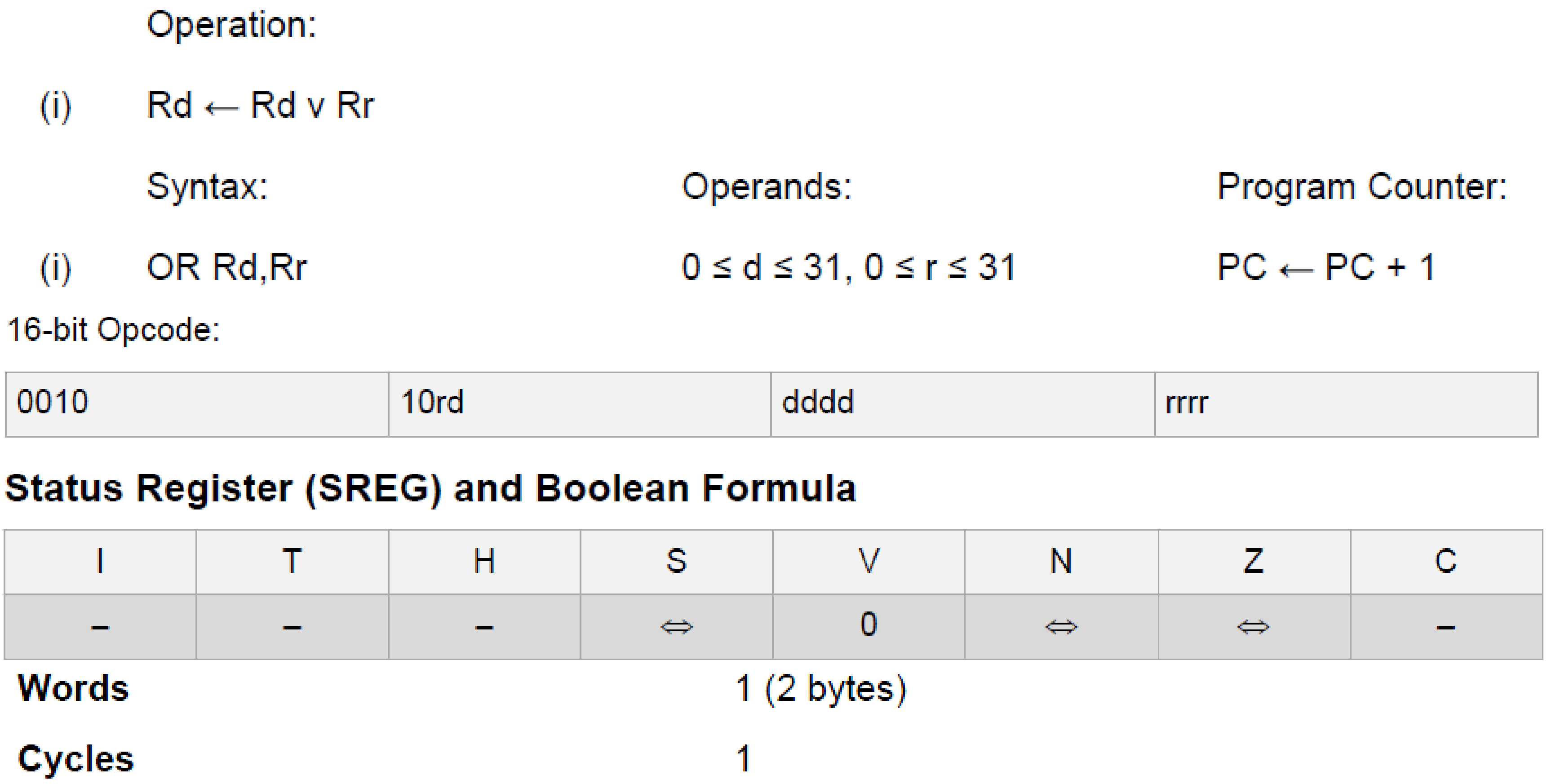
Set if MSB of the result is set; cleared otherwise.

Z $\overline{R7} \bullet \overline{R6} \bullet \overline{R5} \bullet \overline{R4} \bullet \overline{R3} \bullet \overline{R2} \bullet \overline{R1} \bullet \overline{R0}$

Set if the result is $00; cleared otherwise.

OR – Logical OR

- Performs the logical OR between the contents of register Rd and register Rr, and places the result in the destination register Rd



Operation:

(i) $Rd \leftarrow Rd \lor Rr$

	Syntax:	Operands:	Program Counter:
(i)	OR Rd,Rr	$0 \le d \le 31$, $0 \le r \le 31$	$PC \leftarrow PC + 1$

16-bit Opcode:

0010	10rd	dddd	rrrr

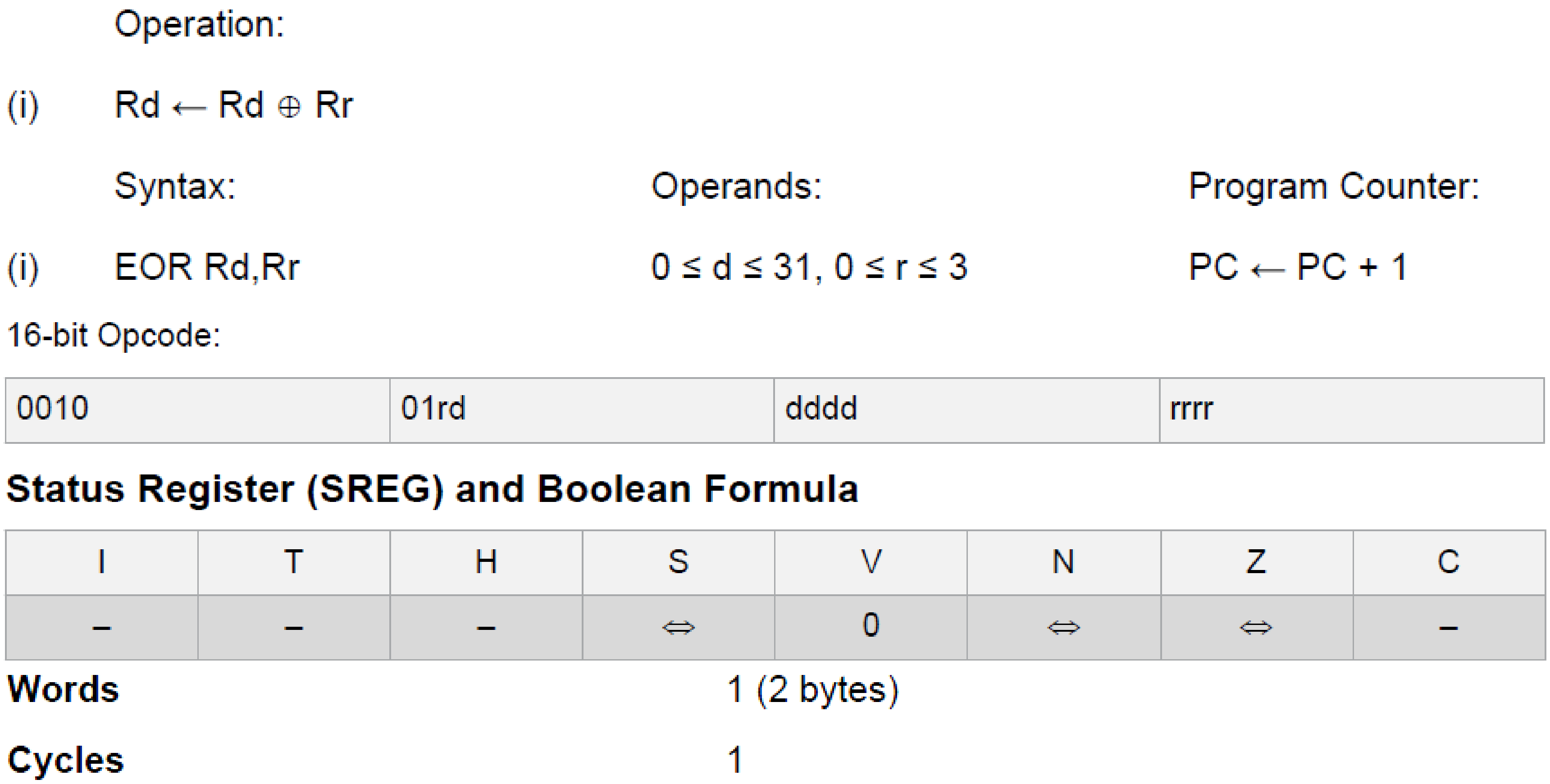
Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	⇔	0	⇔	⇔	–

Words	1 (2 bytes)
Cycles	1

EOR – Exclusive OR

- Performs the logical EOR between the contents of register Rd and register Rr and places the result in the destination register Rd



Operation:

(i) Rd ← Rd ⊕ Rr

	Syntax:	Operands:	Program Counter:
(i)	EOR Rd,Rr	$0 \le d \le 31$, $0 \le r \le 3$	PC ← PC + 1

16-bit Opcode:

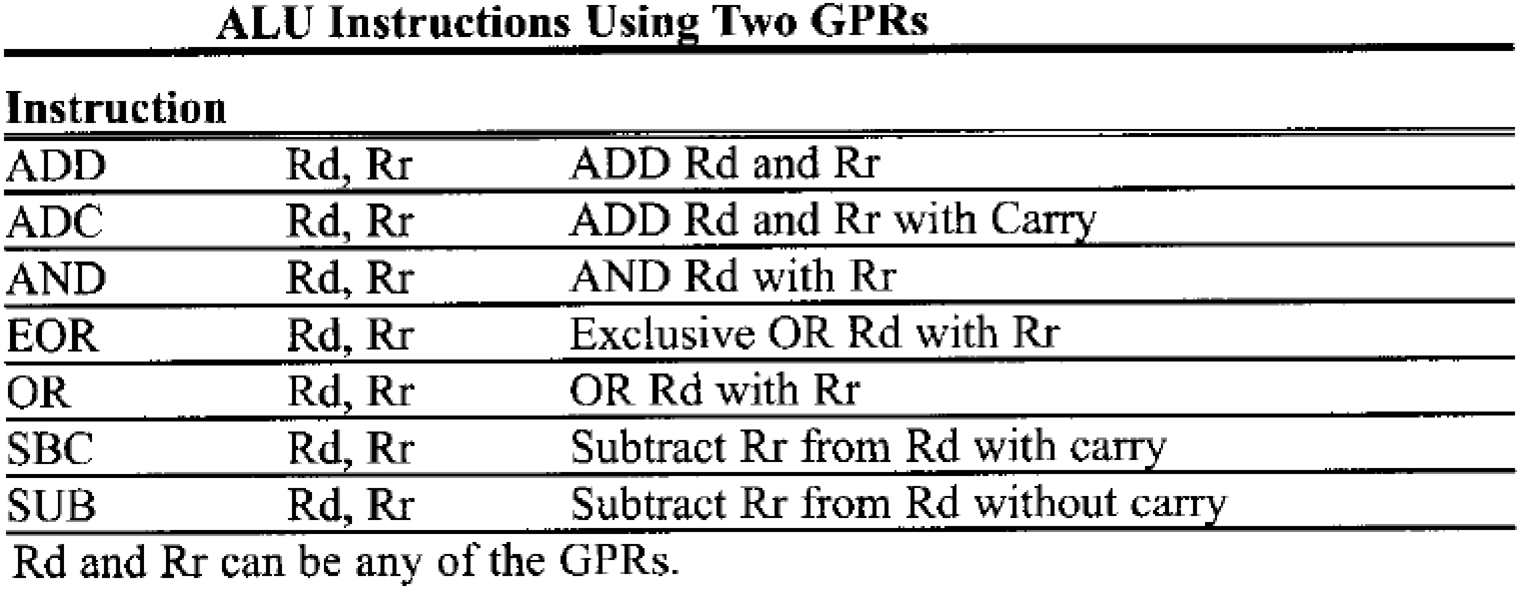
0010	01rd	dddd	rrrr

Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	⇔	0	⇔	⇔	–

Words	1 (2 bytes)
Cycles	1

Summary



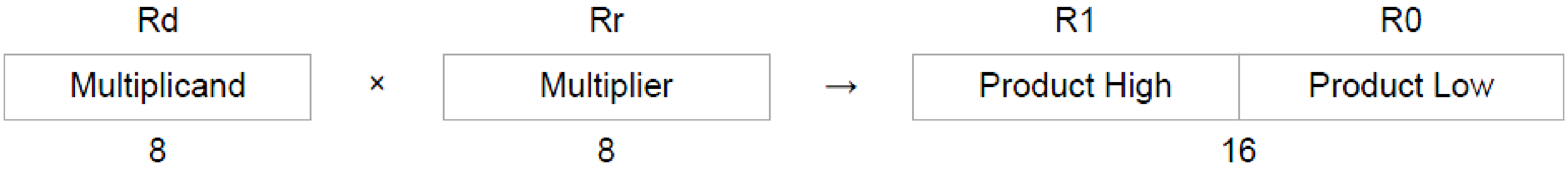
ALU Instructions Using Two GPRs

Instruction		
ADD	Rd, Rr	ADD Rd and Rr
ADC	Rd, Rr	ADD Rd and Rr with Carry
AND	Rd, Rr	AND Rd with Rr
EOR	Rd, Rr	Exclusive OR Rd with Rr
OR	Rd, Rr	OR Rd with Rr
SBC	Rd, Rr	Subtract Rr from Rd with carry
SUB	Rd, Rr	Subtract Rr from Rd without carry

Rd and Rr can be any of the GPRs.

MUL – Multiply Unsigned

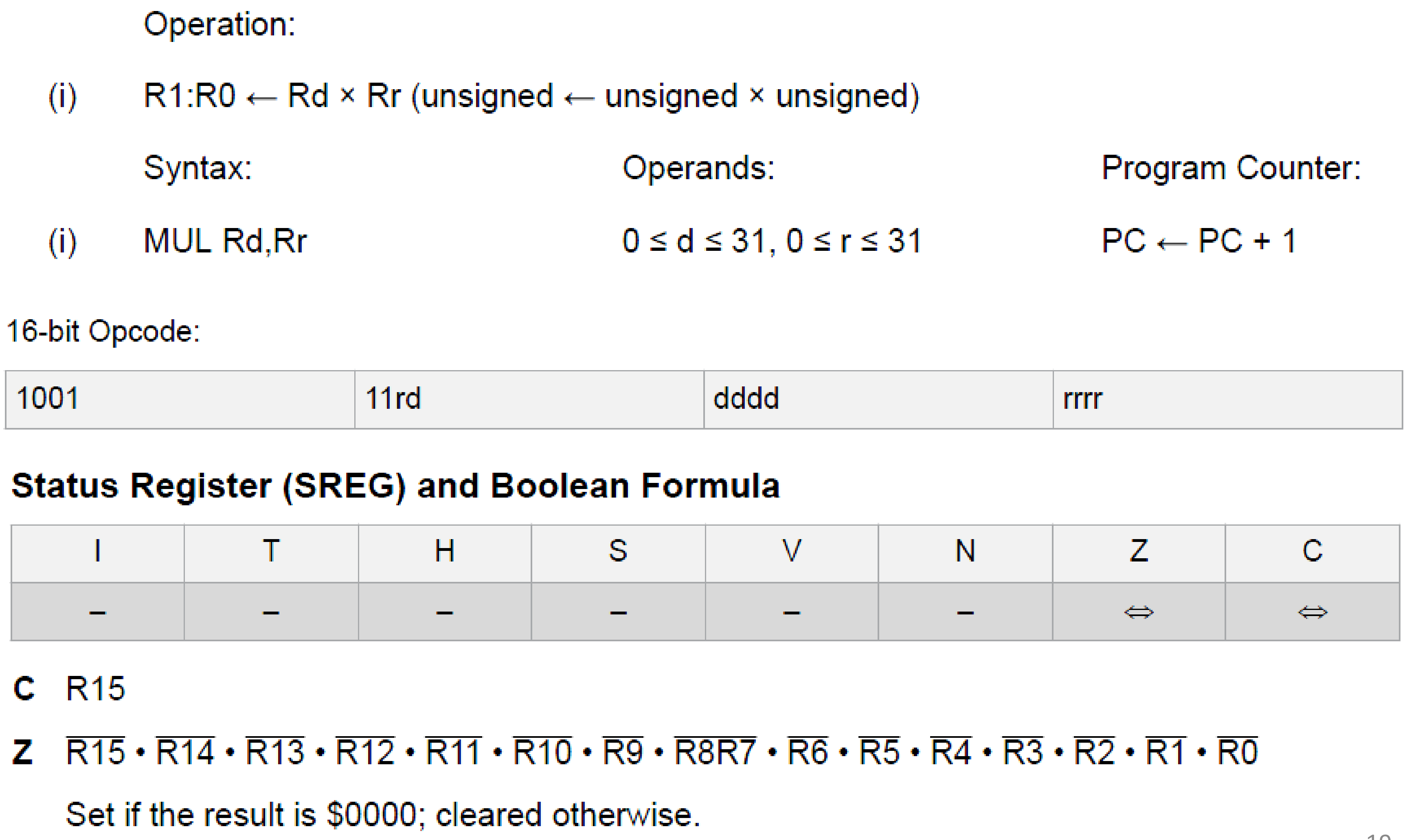
- This instruction performs 8-bit × 8-bit → 16-bit unsigned multiplication



Rd Multiplicand 8 × Rr Multiplier 8 → R1 Product High R0 Product Low 16

- The multiplicand Rd and the multiplier Rr are two registers containing unsigned numbers. The 16-bit unsigned product is placed in R1 (high byte) and R0 (low byte). Note that if the multiplicand or the multiplier is selected from R0 or R1 the result will overwrite those after multiplication.

MUL – Multiply Unsigned



Operation:

(i) R1:R0 ← Rd × Rr (unsigned ← unsigned × unsigned)

	Syntax:	Operands:	Program Counter:
(i)	MUL Rd,Rr	0 ≤ d ≤ 31, 0 ≤ r ≤ 31	PC ← PC + 1

16-bit Opcode:

1001	11rd	dddd	rrrr

Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	⇔	⇔

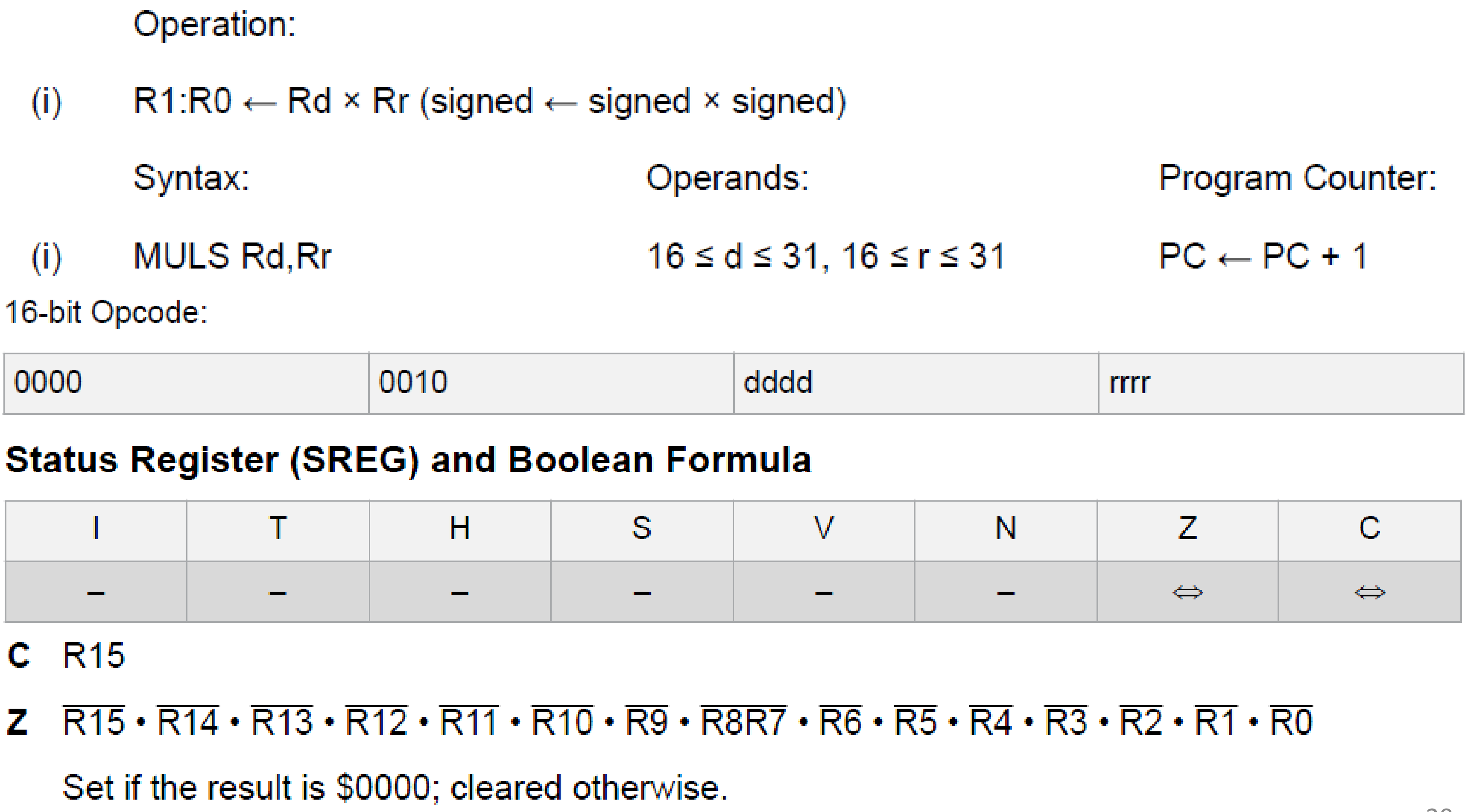
C R15

Z $\overline{R15} \cdot \overline{R14} \cdot \overline{R13} \cdot \overline{R12} \cdot \overline{R11} \cdot \overline{R10} \cdot \overline{R9} \cdot \overline{R8}\overline{R7} \cdot \overline{R6} \cdot \overline{R5} \cdot \overline{R4} \cdot \overline{R3} \cdot \overline{R2} \cdot \overline{R1} \cdot \overline{R0}$

Set if the result is $0000; cleared otherwise.

MULS – Multiply Signed

- This instruction performs 8-bit × 8-bit → 16-bit signed multiplication



Operation:

(i) R1:R0 ← Rd × Rr (signed ← signed × signed)

	Syntax:	Operands:	Program Counter:
(i)	MULS Rd,Rr	$16 \le d \le 31$, $16 \le r \le 31$	PC ← PC + 1

16-bit Opcode:

0000	0010	dddd	rrrr

Status Register (SREG) and Boolean Formula

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	⇔	⇔

C R15

Z $\overline{R15} \bullet \overline{R14} \bullet \overline{R13} \bullet \overline{R12} \bullet \overline{R11} \bullet \overline{R10} \bullet \overline{R9} \bullet \overline{R8R7} \bullet \overline{R6} \bullet \overline{R5} \bullet \overline{R4} \bullet \overline{R3} \bullet \overline{R2} \bullet \overline{R1} \bullet \overline{R0}$

Set if the result is $0000; cleared otherwise.

پایان

موفق و پیروز باشید