

Arithmetic and Logic Instructions

Arithmetic Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
ADD Rd, Rr	Add without Carry	$Rd \leftarrow Rd + Rr$	Z,C,N,V,S,H	1 1 1 1
ADC Rd, Rr	Add with Carry	$Rd \leftarrow Rd + Rr + C$	Z,C,N,V,S,H	1 1 1 1
ADIW Rd, K	Add Immediate to Word	$Rd \leftarrow Rd+1:Rd + K$	Z,C,N,V,S	2 2 2 N/A
SUB Rd, Rr	Subtract without Carry	$Rd \leftarrow Rd - Rr$	Z,C,N,V,S,H	1 1 1 1
SUBI Rd, K	Subtract Immediate	$Rd \leftarrow Rd - K$	Z,C,N,V,S,H	1 1 1 1
SBC Rd, Rr	Subtract with Carry	$Rd \leftarrow Rd - Rr - C$	Z,C,N,V,S,H	1 1 1 1
SBCI Rd, K	Subtract Immediate with Carry	$Rd \leftarrow Rd - K - C$	Z,C,N,V,S,H	1 1 1 1
SBIW Rd, K	Subtract Immediate from Word	$Rd+1:Rd \leftarrow Rd+1:Rd - K$	Z,C,N,V,S	2 2 2 N/A
NEG Rd	Two's Complement	$Rd \leftarrow \$00 - Rd$	Z,C,N,V,S,H	1 1 1 1
INC Rd	Increment	$Rd \leftarrow Rd + 1$	Z,N,V,S	1 1 1 1
DEC Rd	Decrement	$Rd \leftarrow Rd - 1$	Z,N,V,S	1 1 1 1
MUL Rd,Rr	Multiply Unsigned	$R1:R0 \leftarrow Rd \times Rr$ (UU)	Z,C	2 2 2 N/A
MULS Rd,Rr	Multiply Signed	$R1:R0 \leftarrow Rd \times Rr$ (SS)	Z,C	2 2 2 N/A
MULSU Rd,Rr	Multiply Signed with Unsigned	$R1:R0 \leftarrow Rd \times Rr$ (SU)	Z,C	2 2 2 N/A
FMUL Rd,Rr	Fractional Multiply Unsigned	$R1:R0 \leftarrow Rd \times Rr << 1$ (UU)	Z,C	2 2 2 N/A
FMULS Rd,Rr	Fractional Multiply Signed	$R1:R0 \leftarrow Rd \times Rr << 1$ (SS)	Z,C	2 2 2 N/A
FMULSU Rd,Rr	Fractional Multiply Signed with Unsigned	$R1:R0 \leftarrow Rd \times Rr << 1$ (SU)	Z,C	2 2 2 N/A

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Logic Instructions

Mnemonic	Description	Operation	Flags	Clocks AVR ⁽¹⁾
AND Rd, Rr	Logical AND	$Rd \leftarrow Rd \cdot Rr$	Z,N,V,S	1 1 1 1



By **bladabuska**

Not published yet.

Last updated 14th October, 2024.

Page 1 of 9.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>

Logic Instructions (cont)

ANDI Rd, K	Logical AND with Immediate	$Rd \leftarrow Rd \cdot K$	Z,N,V,S	1 1 1 1
OR Rd, Rr	Logical OR	$Rd \leftarrow Rd \vee Rr$	Z,N,V,S	1 1 1 1
ORI Rd, K	Logical OR with Immediate	$Rd \leftarrow Rd \vee K$	Z,N,V,S	1 1 1 1
EOR Rd, Rr	Exclusive OR	$Rd \leftarrow Rd \oplus Rr$	Z,N,V,S	1 1 1 1
COM Rd	One's Complement	$Rd \leftarrow \$FF - Rd$	Z,C,N,V,S	1 1 1 1
SBR Rd,K	Set Bit(s) in Register	$Rd \leftarrow Rd \vee K$	Z,N,V,S	1 1 1 1
CBR Rd,K	Clear Bit(s) in Register	$Rd \leftarrow Rd \cdot (\$FFh - K)$	Z,N,V,S	1 1 1 1
TST Rd	Test for Zero or Minus	$Rd \leftarrow Rd \cdot Rd$	Z,N,V,S	1 1 1 1
CLR Rd	Clear Register	$Rd \leftarrow Rd \oplus Rd$	Z,N,V,S	1 1 1 1
SER Rd	Set Register	$Rd \leftarrow \$FF$	Z,N,V,S	1 1 1 1
DES K	Data Encryption	if (H = 0) then R15:R0 $\leftarrow$ Encrypt(R15: R0, K) else if (H = 1) then R15:R0 $\leftarrow$ Decrypt(R15: R0, K)	NONE	N/A 1/2 N/A N/A

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Flow Control Instructions

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Jump and Call Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
CALL k	Call Subroutine	$PC \leftarrow k$	NONE	4/5 3/4 3/4 N/A
JMP k	Jump	$PC \leftarrow k$	NONE	3 3 3 N/A
RCALL k	Relative Call Subroutine	$PC \leftarrow PC + k + 1$	NONE	3/4 2/3 2/3 3
ICALL	Indirect Call to (Z)	$PC(15:0) \leftarrow Z$ $PC(21:16) \leftarrow 0$	NONE	3/4 2/3 2/3 3
RJMP k	Relative Jump	$PC \leftarrow PC + k + 1$	NONE	2 2 2 2
IJMP	Indirect Jump to (Z)	$PC(15:0) \leftarrow Z$ $PC(21:16) \leftarrow 0$	NONE	2 2 2 2
EICALL	Extended Indirect Call to (Z)	$PC(15:0) \leftarrow Z$ $PC(21:16) \leftarrow EIND$	NONE	4 3 2/3 N/A
EIJMP	Extended Indirect Jump to (Z)	$PC(15:0) \leftarrow Z$ $PC(21:16) \leftarrow EIND$	NONE	2 2 2 N/A
RET	Subroutine Return	$PC \leftarrow STACK$	NONE	4/5 4/5 4/5 6
RETI	Interrupt Return	$PC \leftarrow STACK$	I	4/5 4/5 4/5 6

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Compare and Skip Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
----------	-------------	-----------	-------	-----------------------



By bladabuska

Not published yet.
Last updated 14th October, 2024.
Page 2 of 9.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>

Compare and Skip Instructions (cont)

CPSE Rd, Rr	Compare, skip if Equal	if (Rd = Rr) then PC ← PC + 2/3 else PC ← PC + 1	NONE	1/2/3 1/2/3 1/2/3 1/2
CP Rd, Rr	Compare	Rd - Rr	Z,C,N,V,S,H	1 1 1 1
CPC Rd, Rr	Compare with Carry	Rd - Rr - C	Z,C,N,V,S,H	1 1 1 1
CPI Rd, K	Compare with Immediate	Rd - K	Z,C,N,V,S,H	1 1 1 1
SBRC Rr, b	Skip if Bit in Register Cleared	if (Rr(b) = 0) then PC ← PC + 2/3 else PC ← PC + 1	NONE	1/2/3 1/2/3 1/2/3 1/2
SBRS Rr, b	Skip if Bit in Register Set	if (Rr(b) = 1) then PC ← PC + 2/3 else PC ← PC + 1	NONE	1/2/3 1/2/3 1/2/3 1/2
SBIC A, b	Skip if Bit in I/O Register Cleared	if (I/O(a,b) = 0) then PC ← PC + 2/3 else PC ← PC + 1	NONE	1/2/3 2/3/4 1/2/3 1/2
SBIS A, b	Skip if Bit in I/O Register Set	if (I/O(a,b) = 1) then PC ← PC + 2/3 else PC ← PC + 1	NONE	1/2/3 2/3/4 1/2/3 1/2

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Branch Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
BRBS s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRBC s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BREQ k	Branch if Equal (Zero)	if (Z = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRNE k	Branch if Not Equal (Zero)	if (Z = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRCS k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2



By **bladabuska**

cheatography.com/bladabuska/

Not published yet.

Last updated 14th October, 2024.

Page 3 of 9.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>

Branch Instructions (cont)

BRCC k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRSH k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRLO k	Branch if Lower	if (C = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRMI k	Branch if Minus	if (N = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRPL k	Branch if Plus	if (N = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRGE k	Branch if Greater or Equal, Signed	if (N ⊕ V = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRLT k	Branch if Less Than, Signed	if (N ⊕ V = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRHS k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRHC k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRTS k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRTC k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRVS k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2
BRVC k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1 else PC ← PC + 1	NONE	1/2 1/2 1/2 1/2



By **bladabuska**

cheatography.com/bladabuska/

Not published yet.

Last updated 14th October, 2024.

Page 4 of 9.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>

Branch Instructions (cont)

BRIE k	Branch if Interrupt Enabled	if (I = 1) then $PC \leftarrow PC + k + 1$ else $PC \leftarrow PC + 1$	NONE	1/2 1/2 1/2 1/2
BRID k	Branch if Interrupt Disabled	if (I = 0) then $PC \leftarrow PC + k + 1$ else $PC \leftarrow PC + 1$	NONE	1/2 1/2 1/2 1/2

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Data Transfer Instructions

Flash Memory Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
LPM	Load Program Memory	$R0 \leftarrow (Z)$	NONE	3 3 3 N/A
LPM Rd, Z	Load Program Memory	$Rd \leftarrow (Z)$	NONE	3 3 3 N/A
LPM Rd, Z+	Load Program Memory and Post-Increment	$Rd \leftarrow (Z)$ $Z \leftarrow Z + 1$	NONE	3 3 3 N/A
ELPM	Extended Load Program Memory	$R0 \leftarrow (RAMPZ:Z)$	NONE	3 3 3 N/A
ELPM Rd, Z	Extended Load Program Memory	$Rd \leftarrow (RAMPZ:Z)$	NONE	3 3 3 N/A
ELPM Rd, Z+	Extended Load Program Memory and Post-Increment	$Rd \leftarrow (RAMPZ:Z)$ $RAMPZ:Z \leftarrow RAMPZ:Z + 1$	NONE	3 3 3 N/A
SPM	Store Program Memory	$(RAMPZ:Z) \leftarrow R1:R0$	NONE	D/D D/D 4 N/A
SPM Z+	Store Program Memory and Post-Increment by 2	$(RAMPZ:Z) \leftarrow R1:R0$ $Z \leftarrow Z + 2$	NONE	D/D D/D 4 N/A

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Data Memory Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
LDS Rd, k	Load Direct from data space	$Rd \leftarrow (k)$	NONE	2 2 3 2
LD Rd, X	Load Indirect	$Rd \leftarrow (X)$	NONE	2 1 2 1/2
LD Rd, X+	Load Indirect and Post-Increment	$Rd \leftarrow (X)$ $X \leftarrow X + 1$	NONE	2 1 2 2/3
LD Rd, -X	Load Indirect and Pre-Decrement	$X \leftarrow X - 1$ $Rd \leftarrow (X)$	NONE	2 2 2 2/3
LD Rd, Y	Load Indirect	$Rd \leftarrow (Y)$	NONE	2 1 2 1/2



By **bladabuska**

Not published yet.
Last updated 14th October, 2024.
Page 5 of 9.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>

Data Memory Instructions (cont)

LD Rd, Y+	Load Indirect and Post-Increment	$Rd \leftarrow (Y)$ $Y \leftarrow Y + 1$	NONE	2 1 2 2/3
LD Rd, -Y	Load Indirect and Pre-Decrement	$Y \leftarrow Y - 1$ $Rd \leftarrow (Y)$	NONE	2 2 2 2/3
LDD Rd, Y + q	Load Indirect with Displacement	$Rd \leftarrow (Y + q)$	NONE	2 2 2 N/A
LD Rd, Z	Load Indirect	$Rd \leftarrow (Z)$	NONE	2 1 2 1/2
LD Rd, Z+	Load Indirect and Post-Increment	$Rd \leftarrow (Z)$ $Z \leftarrow Z + 1$	NONE	2 1 2 2/3
LD Rd, -Z	Load Indirect and Pre-Decrement	$Z \leftarrow Z - 1$ $Rd \leftarrow (Z)$	NONE	2 2 2 2/3
LDD Rd, Z + q	Load Indirect with Displacement	$Rd \leftarrow (Z + q)$	NONE	2 2 2 N/A
STS k, Rr	Store Direct to Data Space	$(k) \leftarrow Rr$	NONE	2 2 2 1
ST X, Rr	Store Indirect	$(X) \leftarrow Rr$	NONE	1 1 1 1
ST X+, Rr	Store Indirect and Post-Increment	$(X) \leftarrow Rr$ $X \leftarrow X + 1$	NONE	1 1 1 1
ST -X, Rr	Store Indirect and Pre-Decrement	$X \leftarrow X - 1$ $(X) \leftarrow Rr$	NONE	2 2 1 2
ST Y, Rr	Store Indirect	$(Y) \leftarrow Rr$	NONE	2 1 1 1
ST Y+, Rr	Store Indirect and Post-Increment	$(Y) \leftarrow Rr$ $Y \leftarrow Y + 1$	NONE	2 1 1 1
ST -Y, Rr	Store Indirect and Pre-Decrement	$Y \leftarrow Y - 1$ $(Y) \leftarrow Rr$	NONE	2 2 1 2
STD Y + q, Rr	Store Indirect with Displacement	$(Y + q) \leftarrow Rr$	NONE	2 2 1 N/A
ST Z, Rr	Store Indirect	$(Z) \leftarrow Rr$	NONE	2 1 1 1
ST Z+, Rr	Store Indirect and Post-Increment	$(Z) \leftarrow Rr$ $Z \leftarrow Z + 1$	NONE	2 1 1 1
ST -Z, Rr	Store Indirect and Pre-Decrement	$Z \leftarrow Z - 1$ $(Z) \leftarrow Rr$	NONE	2 2 1 2
STD Z + q, Rr	Store Indirect with Displacement	$(Z + q) \leftarrow Rr$	NONE	2 2 1 N/A
PUSH Rr	Push Register on Stack	$STACK \leftarrow Rr$	NONE	2 1 1 1
POP Rd	Pop Register from Stack	$Rd \leftarrow STACK$	NONE	2 2 2 3
XCH Z, Rd	Exchange	$(Z) \leftarrow Rd$ $Rd \leftarrow (Z)$	NONE	N/A 1 N/A N/A
LAS Z, Rd	Load and Set	$(Z) \leftarrow Rd \vee (Z)$ $Rd \leftarrow (Z)$	NONE	N/A 1 N/A N/A



By **bladabuska**

Not published yet.

Last updated 14th October, 2024.

Page 6 of 9.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>

Data Memory Instructions (cont)

LAC Z, Rd	Load and Clear	$(Z) \leftarrow (\$FF - Rd) \cdot (Z)$ $Rd \leftarrow (Z)$	NONE	N/A 1 N/A N/A
LAT Z, Rd	Load and Toggle	$(Z) \leftarrow Rd \oplus (Z)$ $Rd \leftarrow (Z)$	NONE	N/A 1 N/A N/A

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Register Transfer Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
MOV Rd, Rr	Copy Register	$Rd \leftarrow Rr$	NONE	1 1 1 1
MOVW Rd, Rr	Copy Register Pair	$Rd+1:Rd \leftarrow Rr+1:Rr$	NONE	1 1 1 N/A
LDI Rd, K	Load Immediate	$Rd \leftarrow K$	NONE	1 1 1 1
IN Rd, A	In From I/O Location	$Rd \leftarrow I/O(A)$	NONE	1 1 1 1
OUT A, Rr	Out To I/O Location	$I/O(A) \leftarrow Rr$	NONE	1 1 1 1

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Bit and Bit-Test Instructions

Shift and Rotation Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
LSL Rd	Logical Shift Left	$Rd(n+1) \leftarrow Rd(n)$ $Rd(0) \leftarrow 0$ $C \leftarrow Rd(7)$	Z,C,N,V,H	1 1 1 1
LSR Rd	Logical Shift Right	$Rd(n) \leftarrow Rd(n+1)$ $Rd(7) \leftarrow 0$ $C \leftarrow Rd(0)$	Z,C,N,V	1 1 1 1
ROL Rd	Rotate Left Through Carry	$Rd(0) \leftarrow C$ $Rd(n+1) \leftarrow Rd(n)$ $C \leftarrow Rd(7)$	Z,C,N,V,H	1 1 1 1
ROR Rd	Rotate Right Through Carry	$Rd(7) \leftarrow C$ $Rd(n) \leftarrow Rd(n+1)$ $C \leftarrow Rd(0)$	Z,C,N,V	1 1 1 1
ASR Rd	Arithmetic Shift Right	$Rd(n) \leftarrow Rd(n+1)$, $n = 0..6$	Z,C,N,V	1 1 1 1
SWAP Rd	Swap Nibbles	$R(3..0) \leftrightarrow R(7..4)$	NONE	1 1 1 1
SBI A, b	Set Bit in I/O Register	$I/O(A,b) \leftarrow 1$	NONE	2 1 1 1
CBI A, b	Clear Bit in I/O Register	$I/O(A,b) \leftarrow 0$	NONE	2 1 1 1
BST Rr, b	Bit Store from Register to T	$T \leftarrow Rr(b)$	T	1 1 1 1



By bladabuska

Not published yet.
Last updated 14th October, 2024.
Page 7 of 9.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>

Shift and Rotation Instructions (cont)

BLD Rd, b	Bit load from T to Register	$Rd(b) \leftarrow T$	NONE	1 1 1 1
-----------	-----------------------------	----------------------	------	---------------

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Status Register Bit Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
BSET s	Flag Set	$SREG(s) \leftarrow 1$	SREG(s)	1 1 1 1
BCLR s	Flag Clear	$SREG(s) \leftarrow 0$	SREG(s)	1 1 1 1
SEC	Set Carry	$C \leftarrow 1$	C	1 1 1 1
CLC	Clear Carry	$C \leftarrow 0$	C	1 1 1 1
SEN	Set Negative Flag	$N \leftarrow 1$	N	1 1 1 1
CLN	Clear Negative Flag	$N \leftarrow 0$	N	1 1 1 1
SEZ	Set Zero Flag	$Z \leftarrow 1$	Z	1 1 1 1
CLZ	Clear Zero Flag	$Z \leftarrow 0$	Z	1 1 1 1
SEI	Global Interrupt Enable	$I \leftarrow 1$	I	1 1 1 1
CLI	Global Interrupt Disable	$I \leftarrow 0$	II	1 1 1 1
SES	Set Signed Test Flag	$S \leftarrow 1$	S	1 1 1 1
CLS	Clear Signed Test Flag	$S \leftarrow 0$	S	1 1 1 1
SEV	Set Two's Complement Overflow	$V \leftarrow 1$	V	1 1 1 1
CLV	Clear Two's Complement Overflow	$V \leftarrow 0$	V	1 1 1 1
SET	Set T in SREG	$T \leftarrow 1$	T	1 1 1 1
CLT	Clear T in SREG	$T \leftarrow 0$	T	1 1 1 1
SEH	Set Half Carry Flag in SREG	$H \leftarrow 1$	H	1 1 1 1
CLH	Clear Half Carry Flag in SREG	$H \leftarrow 0$	H	1 1 1 1

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

MCU Control Instructions

MCU Control Instructions

Mnemonic	Description	Operation	Flags	Clocks ⁽¹⁾
BREAK	Break	-	NONE	1 1 1 1
NOP	No Operation	-	NONE	1 1 1 1
SLEEP	Sleep	-	NONE	1 1 1 1
WDR	Watchdog Reset	-	NONE	1 1 1 1

1. Number of clocks in AVR 8-bit CPU version (AVR | AVRxm | AVRxt | AVRrc)

Assembler Directives

Definição de Segmentos

.CSEG	Defines the start of a Code Segment.
.CSEGSIZE ⁽¹⁾	Specifies the size of the program memory block.
.DSEG	Defines the start of a Data Segment.
.ESEG	Defines the start of an EEPROM Segment.
.ORG	Sets the location counter to an absolute value.
.OVERL- AP/.NO- OVERLAP ⁽²⁾	Marks a section that will be allowed to overlap code/data with code/data defined elsewhere, without any error or warning messages being generated.

1. This directive is specific to AT94K devices, since they have a user configurable memory partition between the AVR Program memory and the data memory.
2. These directives are for projects with special needs and should normally not be used.

Definição de Símbolos

.DEF	Defines a symbol to a register.
.EQU	Assigns a value to a label. This value cannot be reassigned later in the program.
.SET	Assigns a value to a label. This value can be reassigned later in the program.
.UNDEF	Undefines a symbol previously defined with the .DEF directive.

Definição de Variáveis

.BYTE	Reserves blocks of 1 byte of memory resources in the SRAM or EEPROM.
.DB	Reserves blocks of 1 byte of memory resources in the SRAM or EEPROM.
.DD	
.DQ	
.DW	



By **bladabuska**

cheatography.com/bladabuska/

Not published yet.

Last updated 14th October, 2024.

Page 9 of 9.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>