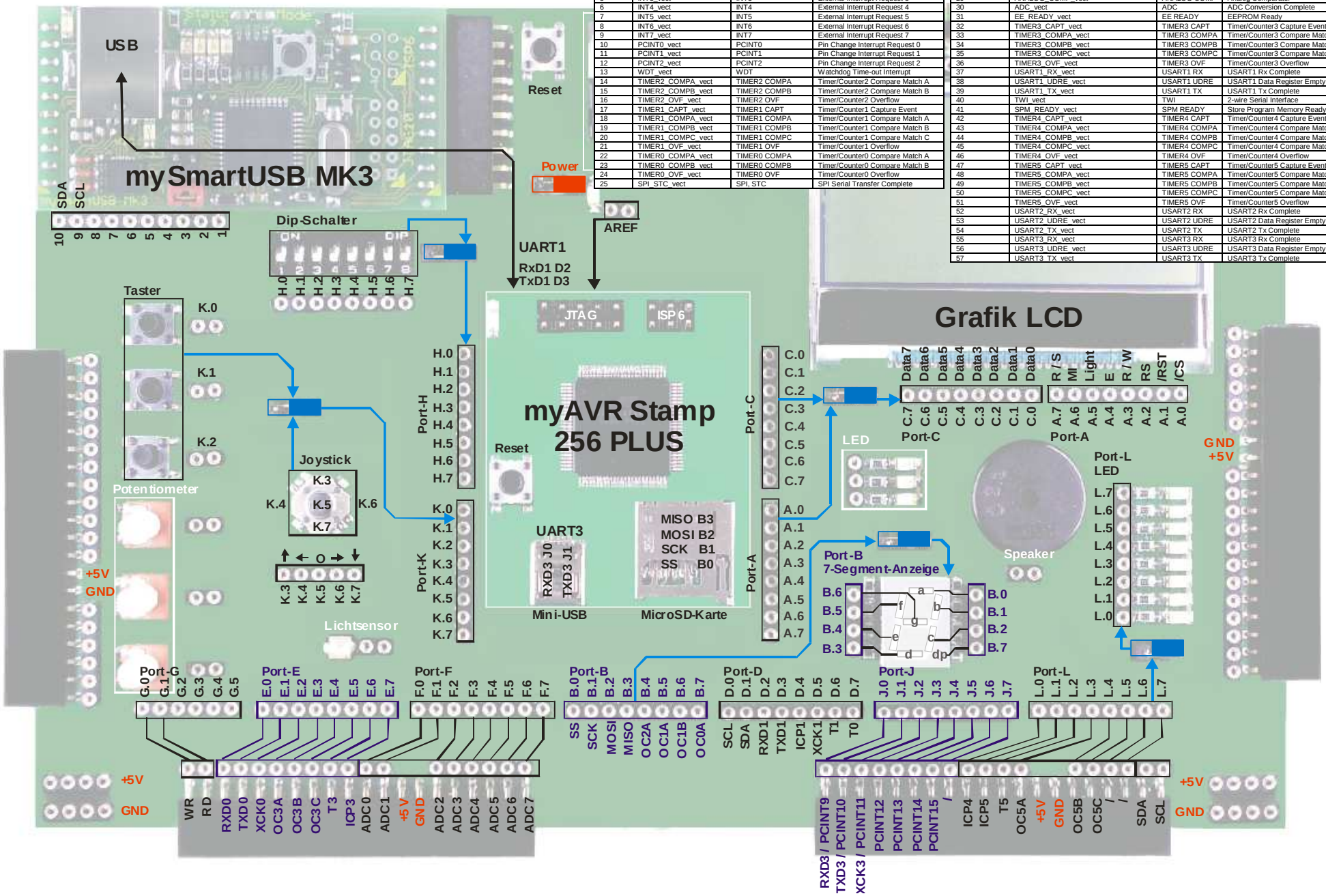
Vekt.-Nr.	C-Vektoren	Source	Interrupt Definition
2	INT0_vect	INT0	External Interrupt Request 0
3	INT1_vect	INT1	External Interrupt Request 1
4	INT2_vect	INT2	External Interrupt Request 2
5	INT3_vect	INT3	External Interrupt Request 3
6	INT4_vect	INT4	External Interrupt Request 4
7	INT5_vect	INT5	External Interrupt Request 5
8	INT6_vect	INT6	External Interrupt Request 6
9	INT7_vect	INT7	External Interrupt Request 7
10	PCINT0_vect	PCINT0	Pin Change Interrupt Request 0
11	PCINT1_vect	PCINT1	Pin Change Interrupt Request 1
12	PCINT2_vect	PCINT2	Pin Change Interrupt Request 2
13	WDT_vect	WDT	Watchdog Time-out Interrupt
14	TIMER2_COMPA_vect	TIMER2 COMPA	Timer/Counter2 Compare Match A
15	TIMER2_COMPB_vect	TIMER2 COMPB	Timer/Counter2 Compare Match B
16	TIMER2_OVF_vect	TIMER2 OVF	Timer/Counter2 Overflow
17	TIMER1_CAPT_vect	TIMER1 CAPT	Timer/Counter1 Capture Event
18	TIMER1_COMPA_vect	TIMER1 COMPA	Timer/Counter1 Compare Match A
19	TIMER1_COMPB_vect	TIMER1 COMPB	Timer/Counter1 Compare Match B
20	TIMER1_COMPC_vect	TIMER1 COMPC	Timer/Counter1 Compare Match C
21	TIMER1_OVF_vect	TIMER1 OVF	Timer/Counter1 Overflow
22	TIMER0_COMPA_vect	TIMER0 COMPA	Timer/Counter0 Compare Match A
23	TIMER0_COMPB_vect	TIMER0 COMPB	Timer/Counter0 Compare Match B
24	TIMER0_OVF_vect	TIMER0 OVF	Timer/Counter0 Overflow
25	SPI_STC_vect	SPI, STC	SPI Serial Transfer Complete

Vekt.-Nr.	C-Vektoren	Source	Interrupt Definition
26	USART0_RX_vect	USART0 RX	USART0 Rx Complete
27	USART0_UDRE_vect	USART0 UDRE	USART0 Data Register Empty
28	USART0_TX_vect	USART0 TX	USART0 Tx Complete
29	ANALOG_COMP_vect	ANALOG COMP	Analog Comparator
30	ADC_vect	ADC	ADC Conversion Complete
31	EE_READY_vect	EE READY	EEPROM Ready
32	TIMER3_CAPT_vect	TIMER3 CAPT	Timer/Counter3 Capture Event
33	TIMER3_COMPA_vect	TIMER3 COMPA	Timer/Counter3 Compare Match A
34	TIMER3_COMPB_vect	TIMER3 COMPB	Timer/Counter3 Compare Match B
35	TIMER3_COMPC_vect	TIMER3 COMPC	Timer/Counter3 Compare Match C
36	TIMER3_OVF_vect	TIMER3 OVF	Timer/Counter3 Overflow
37	USART1_RX_vect	USART1 RX	USART1 Rx Complete
38	USART1_UDRE_vect	USART1 UDRE	USART1 Data Register Empty
39	USART1_TX_vect	USART1 TX	USART1 Tx Complete
40	TWI_vect	TWI	2-wire Serial Interface
41	SPM_READY_vect	SPM READY	Store Program Memory Ready
42	TIMER4_CAPT_vect	TIMER4 CAPT	Timer/Counter4 Capture Event
43	TIMER4_COMPA_vect	TIMER4 COMPA	Timer/Counter4 Compare Match A
44	TIMER4_COMPB_vect	TIMER4 COMPB	Timer/Counter4 Compare Match B
45	TIMER4_COMPC_vect	TIMER4 COMPC	Timer/Counter4 Compare Match C
46	TIMER4_OVF_vect	TIMER4 OVF	Timer/Counter4 Overflow
47	TIMER5_CAPT_vect	TIMER5 CAPT	Timer/Counter5 Capture Event
48	TIMER5_COMPA_vect	TIMER5 COMPA	Timer/Counter5 Compare Match A
49	TIMER5_COMPB_vect	TIMER5 COMPB	Timer/Counter5 Compare Match B
50	TIMER5_COMPC_vect	TIMER5 COMPC	Timer/Counter5 Compare Match C
51	TIMER5_OVF_vect	TIMER5 OVF	Timer/Counter5 Overflow
52	USART2_RX_vect	USART2 RX	USART2 Rx Complete
53	USART2_UDRE_vect	USART2 UDRE	USART2 Data Register Empty
54	USART2_TX_vect	USART2 TX	USART2 Tx Complete
55	USART3_RX_vect	USART3 RX	USART3 Rx Complete
56	USART3_UDRE_vect	USART3 UDRE	USART3 Data Register Empty
57	USART3_TX_vect	USART3 TX	USART3 Tx Complete



© Laser & Co. Solutions GmbH - 03/2012

I/O Register (Auszug) / register set (essentials)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UCSRnC n = 3-0	UMSELn1	UMSELn0	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn
UCSRnB n = 3-0	RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n
UCSRnA n = 3-0	RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn
OCRn[A|B|C][H|L] n = 5/4/3/1	Timer/Counter n - Output Compare Register A|B|C High bzw. Low Byte Bsp.: OCR5CL							
OCRn[A|B] n = 2/0	Timer/Counter n - Output Compare Register A|B							
ICRn[H|L] n = 5/4/3/1	Timer/Counter n - Input Capture Register High bzw. Low Byte							
TCNTn n = 5/4/3/1	Timer/Counter n - Counter Register High bzw. Low Byte							
TCNTn n = 2/0	Timer/Counter n (8 Bit)							
TCCRnC n = 5/4/3/1	FOCnA	FOCnB	FOCnC	-	-	-	-	-
TCCRnB n = 5/4/3/1	ICNCn	ICESn	-	WGMn3	WGMn2	CSn2	CSn1	CSn0
TCCRnA n = 5/4/3/1	COMnA1	COMnA0	COMnB1	COMnB0	COMnC1	COMnC0	WGMn1	WGMn0
TCCRnB n = 2/0	FOCnA	FOCnB	-	-	WGMn2	CSn2	CSn1	CSn0
TCCRnA n = 2/0	COMnA1	COMnA0	COMnB1	COMnB0	-	-	WGMn1	WGMn0
PORT[L-H]	PORT [L-H]7	PORT [L-H]6	PORT [L-H]5	PORT [L-H]4	PORT [L- H]3	PORT [L-H]2	PORT [L-H]1	PORT [-H]0
DDR[L-H]	DD[L- H]7	DD[L-H]6	DD[L-H]5	DD[L-H]4	DD[L-H]3	DD[L-H]2	DD[L-H]1	DD[L-H]0
PIN[L-H]	PIN[L-H]7	PIN[L-H]6	PIN[L-H]5	PIN[L-H]4	PIN[L-H]3	PIN[L- H]2	PIN[L-H]1	PIN[L-H]0
PORT[G-A]	PORT [F-A]7	PORT [F-A]6	PORT [G-A]5	PORT [G-A]4	PORT [G-A]3	PORT [G-A]2	PORT [G-A]1	PORT [G-A]0
DDR[G-A]	DD[F-A]7	DD[F-A]6	DD[G-A]5	DD[G-A]4	DD[G-A]3	DD[G-A]2	DD[G-A]1	DD[G-A]0
PIN[G-A]	PIN[F-A]7	PIN[F-A]6	PIN[G-A]5	PIN[G-A]4	PIN[G-A]3	PIN[G-A]2	PIN[G-A]1	PIN[G-A]0
UDR[0|1|2|3]	USART[0|1|2|3] I/O Data Register							
UBRR[0|1|2|3]H	-	-	-	-	USART[0|1|2|3] Baud Rate Register High Byte			
UBRR[0|1|2|3]L	USART[0|1|2|3] Baud Rate Register Low Byte							
TWAMR	TWAM6	TWAM5	TWAM4	TWAM3	TWAM2	TWAM1	TWAM0	-
TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
TWDR	2-wire Serial Interface Data Register							
TWAR	TWA6	TWA5	TWA4	TWA3	TWA2			
TWSR		TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0
TWBR	2-wire Serial Interface Bit Rate Register							
ASSR	-	EXCLK	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB
DIDR1	-	-	-	-	-	-	AIN1D	AIN0D
DIDR0	ADC7D	ADC6D	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D
DIDR2	ADC15D	ADC14D	ADC13D	ADC12D	ADC11D	ADC10D	ADC9D	ADC8D
ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0
ADCSRB	-	ACME	-	-	MUX5	ADTS2	ADTS1	ADTS0
ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
ADC[H|L]	ADC Data Register High bzw. Low Byte							
XMCRB	XMBK	-	-	-	-	XMM2	XMM1	XMM0
XMCRA	SRE	SRL2	SRL1	SRL0	SRW11	SRW10	SRW01	SRW00
TIMSKn n = 5/4/3/1	-	-	ICIEn	-	OCIEnC	OCIEnB	OCIEnA	TOIEn
TIMSKn n = 2/0	-	-	-	-	-	OCIEnB	OCIEnA	TOIEn
PCMSK2	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16
PCMSK1	PCINT15	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8
PCMSK0	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0
EICRB	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40
EICRA	ISC31	ISC30	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00
PCICR	-	-	-	-	-	PCIE2	PCIE1	PCIE0
OSCCAL	Oscillator Calibration Register							
PRR1	-	-	PRTIM5	PRTIM4	PRTIM3	PRUSART	PRUSART	PRUSART
PRR0	PRTWI	PRTIM2	PRTIM0	-	PRTIM1	PRSPI	PRUSART	PRADC
CLKPR	CLKPCE	-	-	-	CLKPS3	CLKPS2	CLKPS1	CLKPS0
WDTCSR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0
SREG	I	T	H	S	V	N	Z	C
SPH	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8
SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
EIND	-	-	-	-	-	-	-	EIND0
RAMPZ	-	-	-	-	-	-	RAMPZ1	RAMPZ0
SPMCSR	SPMIE	RWWSB	SIGRD	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN
MCUCR	JTD	-	-	PUD	-	-	IVSEL	IVCE
MCUSR	-	-	-	JTRF	WDRF	BORF	EXTRF	PORF
SMCR	-	-	-	-	SM2	SM1	SM0	SE
OCDR	OCDR7	OCDR6	OCDR5	OCDR4	OCDR3	OCDR2	OCDR1	OCDR0
ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0
SPDR	SPI Data Register							
SPSR	SPIF	WCOL	-	-	-	-	-	SPI2X
SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
GPIOR[2|1|0]	General Purpose I/O Register 2|1|0							
TCNT0	Timer/Counter0 (8 Bit)							
GTCCR	TSM	-	-	-	-	-	PSRASY	PSRSYNC
EEARH	-	-	-	-	EEPROM Address Register High Byte			
EEARL	EEPROM Address Register Low Byte							
EEDR	EEPROM Data Register							
EECR	-	-	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE
EIMSK	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
EIFR	INTF7	INTF6	INTF5	INTF4	INTF3	INTF2	INTF1	INTF0
PCIFR	-	-	-	-	-	PCIF2	PCIF1	PCIF0
TIFRn n = 5/4/3/1	-	-	ICFn	-	OCFnC	OCFnB	OCFnA	TOVn
TIFRn n = 2/0	-	-	-	-	-	OCFnB	OCFnA	TOVn

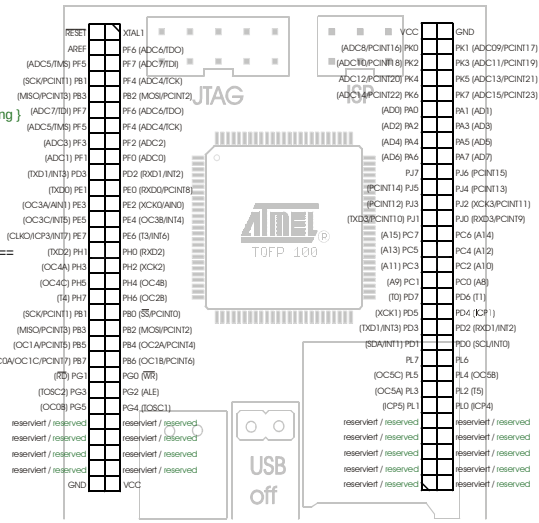
Übersicht AVR-C (Auszug) / instruction set (essentials)

Schlüsselwörter (Auszug) / keyword (essentials)	
case	char
const	continue
default	do
double	else
enum	extern
float	for
int	long
register	return
short	signed
sizeof	struct
static	switch
typdef	union
unsigned	void
volatile	while
Arithmetische Operatoren / arithmetic operators	
binäre Zeichen / binary sign	***Bedeutung / meaning***
+	Addition / addition
-	Subtraktion / subtraction
*	Multiplikation / multiplication
/	Division / division
%	Modulo / modulo
unäre Zeichen / unary signs	***Bedeutung / meaning***
+, -	Vorzeichen / sign
++	Inkrement / increment
--	Dekrement / decrement
Operatoren / operators	
Bit-Operatoren / bitwise operators	***Bedeutung / meaning***
&	und / and
|	oder / or
~	nicht-Operator / nor
^	exklusiv-Oder / xor
>>	Verschiebung nach rechts / shift right
<<	Verschiebung nach links / shift left
log. Operatoren / logical operators	***Bedeutung / meaning***
==	ist gleich / is equal
>	größer als / greater than
<	kleiner als / less than
>=	größer gleich / greater-than-or-equal
<=	kleiner gleich / less-than-or-equal
!=	ungleich / unequal
&&	und / and
||	oder / or
!	nicht / not
Wertzuweisungen / value assignments	
=	Wertzuweisung / value assignment
Trennzeichen / separators	
Symbol / symbole	***Bedeutung / meaning***
;	Befehlsende / end of commands
{	Blockanfang Geltungsbereich / block beginning
}	Blockende Geltungsbereich / block ending
,	Trennzeichen Parameter / separator of parameters
(	Anfang Parameterliste / beginning parameter list
)	Ende Parameterliste / ending parameter list
Zeichenkette / string	
Symbole / symbols	***Bedeutung / meaning***
„Hallo"	konstanter String / constant string
‚A'	konstantes Zeichen / constant character
Zahlen / numbers	
Symbole / symbols	***Bedeutung / meaning***
123	Integerzahl Dezimaldarstellung / integer decimal number
0xA0	Integerzahl Hexa-Darstellung / integer hex-number
1.23	Gleitkommazahl / float
0b001	Integerzahl Binär-Darstellung / integer binary code
Kommentare / comments	
Symbole / symbols	***Bedeutung / meaning***
//	bis Zeilenende / till end of line
/*	Kommentaranfang / comment beginning
*/	Kommentarende / comment ending
Datentypen in AVR C / data types in AVR C	
Bedeutung / meaning	***Beispiel / example***
char	
Datentyp für ein Zeichen; 8 bit; Wertebereich: -128 ... 127 data type for one character; 8 bit; range: -128 ... 127	Beispiel: char buchstabe = ‚K'; Beispiel: char wert1; // 8 bit example: char buchstabe = ‚K'; example: char wert1; // 8 bit
int	
Datentyp für eine Ganzzahl; 16 bit; Wertebereich: 32768 ... +32767 data type for integer; 16 bit; range: -32768 ... +32767	Beispiel: int alter = 37; Beispiel: int wert3; example: int alter = 37; example: int wert3;
float	
Datentyp für eine Kommazahl; 32 bit; Wertebereich: $3.4*10^{-38}$ ... $3.4*10^{38}$ data type for decimal; 32 bit; range: $3.4*10^{-38}$ ... $3.4*10^{38}$	Beispiel: float alter = 37.5; example: float alter = 37.5;

bool	
Wahrheitswert; 8 bit / logical value; 8 bit	Wertebereich: true | false / range: true | false
unsigned	
Darstellung vorzeichenloser positiver Zahlen; representation of unsigned positive numbers;	Beispiel: unsigned char wert2; // äquivalent zu uint8_t Beispiel: unsigned int wert4; // entspricht uint16_t example: unsigned char wert2; // equivalent to uint8_t example: unsigned int wert4; // equal to uint16_t
volatile	
flüchtig volatile	Beispiel: volatile int wert5; // 16 bit, von Optimierung ausgeschlossen example: volatile int wert5; // 16 bit, barred from optimization
Kontrollstrukturen in AVR C / control structures in AVR C	
Abschluss einer Anweisung mit einem Semikolon / ending of command with semicolon	Beispiel: / example: PORTB = 0xFF;
Kontrollstruktur / control structure	***Beispiel / example***
if (Bedingung) Anweisung; / if (condition) statement; if (Bedingungen) Anweisung; else Anweisung; / if (condition) statement; else statement;	if (!(PIND&0x04)) PORTB = 0xFF; if (!(PIND&0x04)) PORTB = 0xFF; else PORTB = 0x00;
/* kopfgesteuerte Schleife / WHILE loop */ while (Bedingung) Anweisung; while (condition) statement;	while (i < 5000) i++; // zählen / count while (i < 5000) { ... } // Anweisungen ausführen / execute statements
/* fußgesteuerte Schleife / DO WHILE loop */ do {Anweisung} while (Bedingung); do (statement) while (condition);	do i++; while (i < 500); // zählen / count do { … } while (true); // Anweisungen ausführen / execute statements
/* Zählschleife / FOR loop */ for (Initialisierung; Wiederholbedingung; Schrittweite) Anweisung; for (initialisation; repeatability conditions; increment) statement	for (int i = 0; i < 10; i++) ... ; for (int i = 0; i > 10; i--) { … }
/* Fallunterscheidung / case differentiation */ switch (switch_variable) { case Konstante1 / constant1: {Anweisung1 / statement1;} break; case Konstante2 / constant2: {Anweisung2 / statement2;} break; default: {Anweisung_X / statement_X} break; }	switch (PIND & 0x0C) // 2 Taster an PORTD2 & 3 / 2 buttons on PORTD2 & 3 { case 0x08 : PORTB = 0x01; // Taster1 → LED1 / button1 → LED1 break; case 0x04 : PORTB = 0x02; // Taster2 → LED2 / button2 → LED2 break; default : PORTB = 0x00; // kein Taster / no button break; }

Codebeispiel / code example

```
#include <avr\io.h>
#include <interrupt.h>
//- Funktion mit Parameter und Rückgabewert / function with parameter and return value -
bool function1 (int para1)
{   // eigener Unterprogrammcode
    // own programm code
    return true;
}
//-------- Interrupt Service Routine --------
ISR (INT0_vect)
{ // Interruptbehandlung / interrupt handling }
//-----------------------------------------------
class LED
{
    public:
        void init() {ddrB.bit0 = 1;}
        void on() {portB.bit0 = 1;}
        void off() {portB.bit0 = 0;}
};
//== Hauptprogramm / main programme ==
// startet bei Power ON & Reset /
// starts at Power On & Reset
int main()
{
    // Initialisierungen / initialisations
    LED led;
    led.init();
    while (true)
    {
        // Code (EVA)
        // code (input-output processing)
        led.on();
        waitMs(100);
        led.off();
        waitMs(100);
    }
    return 0;
}
```



myAVR C spezifische Statements / myAVR C specific statements		
Bedeutung / meaning	***Spezifika / specifics***	***Beispiele / examples***
eine Zeit lang warten / wait a while	waitMs(ms)	waitMs(100);
eine Zeit lang warten / wait a while	waitUs(us)	waitUs(100);
Portinitialisierung / port initialisation	ddrX.bitn	ddrB.bit0 = 1;
Eingabe / input	pinX.bitn	wert = pinD.bit2;
Ausgabe / output	portX.bitn	portB.bit0 = 1;
Bit-Wert Deklaration / bit sign declaration	BITn	BIT2

© Laser & Co. Solutions GmbH - 03/2012