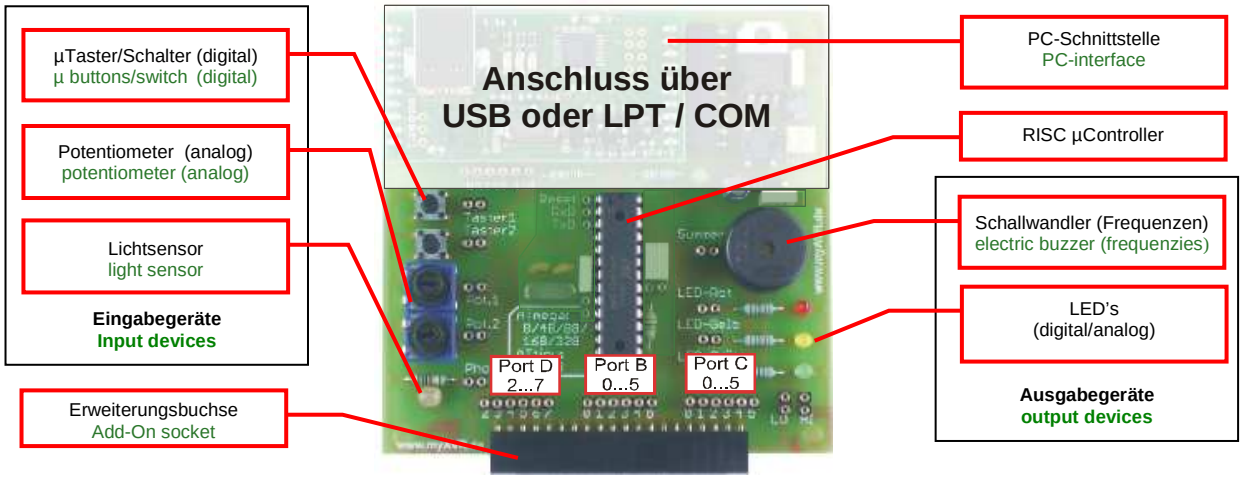
C-Referenzkarte myAVR Board MK2/light/(MK1) mit ATmega8 / C-reference sheet myAVR board MK2/light/(MK1) with ATmega8

Experimentierplattform:

myAVR Board MK2 USB Version 2.2 / myAVR Board light Version 1.1 / (myAVR Board MK1 LPT Version 1.6)

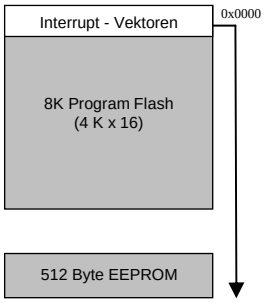


I/O Register

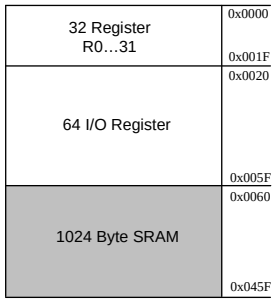
I/O	MEM	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0x3F	0x5F	SREG	I	T	H	S	V	N	Z	C
0x3E	0x5E	SPH	–	–	–	–	–	SP10	SP9	SP8
0x3D	0x5D	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
0x3C	0x5C	Reserved								
0x3B	0x5B	GICR	INT1	INT0	–	–	–	–	IVSEL	IVCE
0x3A	0x5A	GIFR	INTF1	INTF0	–	–	–	–	–	–
0x39	0x59	TIMSK	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	–	TOIE0
0x38	0x58	TIFR	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	–	TOV0
0x37	0x57	SPMCR	SPMIE	RWWSB	–	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN
0x36	0x56	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
0x35	0x55	MCUCR	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00
0x34	0x54	MCUCSR	–	–	–	–	WDRF	BORF	EXTRF	PORF
0x33	0x53	TCCR0	–	–	–	–	–	CS02	CS01	CS00
0x32	0x52	TCNT0	Timer/Counter0 (8 Bits)							
0x31	0x51	OSCCAL	Oscillator Calibration Register							
0x30	0x50	SFIOR	–	–	–	–	ACME	PUD	PSR2	PSR10
0x2F	0x4F	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10
0x2E	0x4E	TCCR1B	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10
0x2D	0x4D	TCNT1H	Timer/Counter1 – Counter Register High byte							
0x2C	0x4C	TCNT1L	Timer/Counter1 – Counter Register Low byte							
0x2B	0x4B	OCR1AH	Timer/Counter1 – Output Compare Register A High byte							
0x2A	0x4A	OCR1AL	Timer/Counter1 – Output Compare Register A Low byte							
0x29	0x49	OCR1BH	Timer/Counter1 – Output Compare Register B High byte							
0x28	0x48	OCR1BL	Timer/Counter1 – Output Compare Register B Low byte							
0x27	0x47	ICR1H	Timer/Counter1 – Input Capture Register High byte							
0x26	0x46	ICR1L	Timer/Counter1 – Input Capture Register Low byte							
0x25	0x45	TCCR2	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20
0x24	0x44	TCNT2	Timer/Counter2 (8 Bits)							
0x23	0x43	OCR2	Timer/Counter2 Output Compare Register							
0x22	0x42	ASSR	-	–	–	–	AS2	TCN2UB	OCR2UB	TCR2UB
0x21	0x41	WDTCR	–	–	–	WDCE	WDE	WDP2	WDP1	WDP0
0x20(1)	0x40(1)	UBRRH	URSEL	–	–	–	UBRR[11:8]			
		UCSRC	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL
0x1F	0x3F	EEARH	–	–	–	–	–	–	–	EEAR8
0x1E	0x3E	EEARL	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0
0x1D	0x3D	EEDR	EEPROM Data Register							
0x1C	0x3C	EECR	–	–	–	–	EERIE	EEMWE	EEWE	EERE
0x1B	0x3B	Reserved								
0x1A	0x3A	Reserved								
0x19	0x39	Reserved								
0x18	0x38	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
0x17	0x37	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
0x16	0x36	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
0x15	0x35	PORTC	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
0x14	0x34	DDRC	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
0x13	0x33	PINC	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
0x12	0x32	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0
0x11	0x31	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
0x10	0x30	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
0x0F	0x2F	SPDR	SPI Data Register							
0x0E	0x2E	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X
0x0D	0x2D	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
0x0C	0x2C	UDR	USART I/O Data Register							
0x0B	0x2B	UCSRA	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM
0x0A	0x2A	UCSRB	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
0x09	0x29	UBRRL	USART Baud Rate Register Low byte							
0x08	0x28	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0
0x07	0x27	ADMUX	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0
0x06	0x26	ADCSRA	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
0x05	0x25	ADCH	ADC Data Register High byte							
0x04	0x24	ADCL	ADC Data Register Low byte							
0x03	0x23	TWDR	Two-wire Serial Interface Data Register							
0x02	0x22	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
0x01	0x21	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0
0x00	0x20	TWBR	Two-wire Serial Interface Bit Rate Register 168							

Speicheraufbau / memory structure

FLASH



SRAM



Interruptvektoren / interrupt vectors

Vector No.	Program Adress	Source	Interrupt Definition
1	0x000	RESET	External Pin,Power-on Reset,Brown-out Reset, and Watchdog Reset
2	0x001	INT0	External Interrupt Request 0
3	0x002	INT1	External Interrupt Request 1
4	0x003	TIMER2 COMP	Timer/Counter2 Compare Match
5	0x004	TIMER2 OVF	Timer/Counter2 Overflow
6	0x005	TIMER1 CAPT	Timer/Counter1 Capture Event
7	0x006	TIMER1 COMPA	Timer/Counter1 Compare MatchA
8	0x007	TIMER1 COMPB	Timer/Counter1 Compare MatchB
9	0x008	TIMER1 OVF	Timer/Counter1 Overflow
10	0x009	TIMER0 OVF	Timer/Counter0 Overflow
11	0x00A	SPI; STC	Serial Transfer Complete
12	0x00B	USART , RXC	USART, Rx Complete
13	0x00C	USART , UDRE	USART, Data Register Empty
14	0x00D	USART , TXC	USART, TX Complete
15	0x00E	ADC	ADC Conversion Complete
16	0x00F	EE_RDY	EEPROM Ready
17	0x010	ANA_COMP	Analog Comperator
18	0x011	TWI	Two-wire serial Interface
19	0x012	SPM_RDY	Store Program Memory Ready

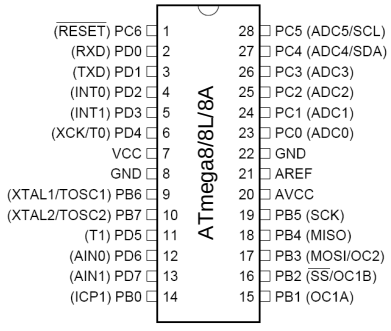
Arbeitsregister / working registers

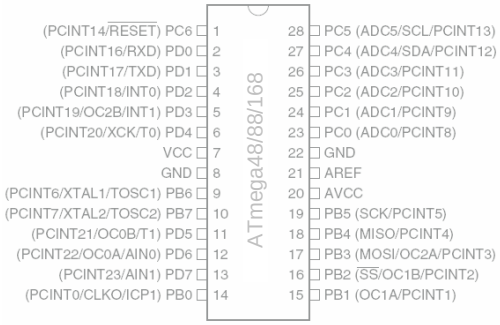
Register			Adresse	Bemerkung
R0			0x00	siehe LPM
R1			0x01	**KEIN** ADIW, SUBI, SUBIW, ANDI, ORI, CPI, LDI,
...			...	
R15			0x0F	
R16			0x10	keine Einschränkungen
...			...	
R26	XL	X	0x1A	siehe LD/ST
R27	XH		0x1B	
R28	YL	Y	0x1C	siehe LD/ST
R29	YH		0x1D	
R30	ZL	Z	0x1E	siehe LD/ST
R31	ZH		0x1F	

Portfunktionen / port functions

Name	Port	Funktion
Externe Interrupts	D2-3	INT0-1
Analog-Digital-Converter	C0-5	ADC0-5
Analog-Komparator	D6-7	AIN0-1
myAVR LCD	D2	R/S
	D3	Enable
	D4-7	DB4-7
	B0 (optional)	R/W
	B1 (optional)	Backlight
TWI / I²C	C4, C5	SDA, SCL
SPI-Bus AVR-ISP	B3	MOSI
	B4	MISO
	B5	SCK
	B2 (optional)	SS
UART (RS232)	D0, D1	RxD, TxD
	D4 (für Sync.Mode)	XCK
Reset	C6	RST
Quarz/Resonator/Takt	B6, B7	XTAL1, XTAL2
Timer/Counter-Output (PWM, Waveform)	B1, B2, B3	OC1A, OC1B, OC2
Timer/Counter Input-Clock	D4, D5	T0, T1
Timer/Counter InputCapture	B0	ICP1

Pinbelegung / pin assignments





C-Referenzkarte myAVR Board MK2/light/(MK1) mit ATmega8 / C-reference sheet myAVR board MK2/light/(MK1)

Die wichtigsten Datentypen in AVR C / The most important data types in AVR C	
Char	Datentyp für 1 Zeichen (Character, Buchstabe) 8 Bit, Wertebereich: -128..127 / Data type for 1 character (character, letter) 8 bits, value range: -128 .. 127 Beispiel / example: char buchstabe = 'K';
int	Datentyp für eine Ganzzahl (Integer). 16 Bit, Wertebereich: -32768...+32767 / Data type for an integer (Integer). 16 bits value range: -32768 ... + 32767 Beispiel / example : int alter = 37;
float	Datentyp für eine Kommazahl. 32 Bit, Wertebereich: $3.4*10^{-38}$...$3.4*10^{38}$ / Data type for a point number 32 bits value range: $3.4 * 10^{-38}$ ... $3.4 * 10^{38}$ Beispiel / example : float alter = 37.5;
unsigned	(vorzeichenlos) legt fest das ein Wert kein Vorzeichen besitzt - also nur positive Zahlen darstellen kann. Der Typ unsigned selbst repräsentiert ein Bit. specifies that a value is not signed - therefore can only represent positive numbers. The unsigned itself represents a bit.
bool	Wahrheitswert 8 Bit, Wertebereich: true, false / logical value; 8 bit, range: true | false
volatile	Flüchtig, 16 bit, von Optimierung ausgeschlossen / volatile, 16 bit, barred from optimization Beispiel: volatile int wert5; / example: volatile int wert5;

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	***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	***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
}	Blockende Geltungsbereich / block ending
,	Trennzeichen Parameter / separator of
(	Anfang Parameterliste / beginning parameter
)	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
0xA0	Integerzahl Hexa-Darstellung / integer hex-
1.23	Gleitkommazahl / float
0b001	Integerzahl Binär-Darstellung / integer binary
Kommentare / comments	
Symbole / symbols	***Bedeutung / meaning***
//	bis Zeilenende / till end of line
/*	Kommentaranfang / comment beginning
*/	Kommentarende / comment ending

Interruptmakros für den ATmega8: ISR(vektor) / Interrupt macros for ATmega8: ISR (vector)
INT0_vect, INT1_vect,
TIMER2_COMP_vect, TIMER2_OVF_vect,
TIMER1_CAPT_vect, TIMER1_COMPA_vect, TIMER1_COMPB_vect, TIMER1_OVF_vect
TIMER0_OVF_vect
USART_RXC_vect, USART_UDRE_vect, USART_TXC_vect, ADC_vect,
SPI_STC_vect ; EE_RDY_vect; ANA_COMP_vect; TWI_vect; SPM_RDY_vect

Kontrollstrukturen in AVR C / control structures in AVR C

Abschluss einer Anweisung mit einem Semikolon / ending of command with

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
{Anweisung1 ;}
break;
case Konstante2 :
{Anweisung2 ;}
break;
default:
{Anweisung_X }
break;
}
	switch (PIND & 0x0C)
	// 2 Taster an PORTD2 & 3 / 2
	{
	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;
	}

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

Codebeispiel / code example

```
#include <avr\io.h>
#include <interrupt.h>
// Funktion mit Parameter und Rückgabewert
bool function1 (int para1)
{    // eigener Unterprogrammcode
     return true;
}

//-------- Interrupt Service Routine --------
ISR (INT0_vect)
{
class LED
{
     public:
          void init() {ddrB.bit0 = 1;}
          void on() {portB.bit0 = 1;}
          void off() {portB.bit0 = 0;}
};

//== Hauptprogramm == startet bei Power ON & Reset
int main()
{     // Initialisierungen
     LED led;
     led.init();
     while (true)
     {          // Code (EVA)
          led.on();
          waitMs(100);
          led.off();
          waitMs(100);
     }
     return 0;
}
```