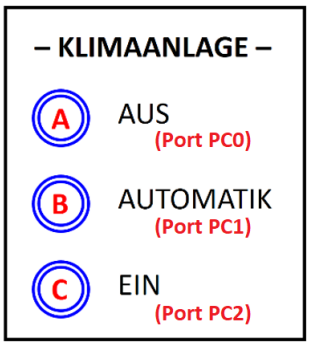
Hochschule München FK03	Embedded Systems		Prof. Dr.-Ing. T. Küpper
Zugelassene Hilfsmittel: alle eigenen, Taschenrechner	**Matr.-Nr.:**	**Name, Vorname:**	
	Hörsaal:	**Unterschrift:**	

WiSe 2023/24
Viel Erfolg!!

1	2	3	4	5	6		Σ	N

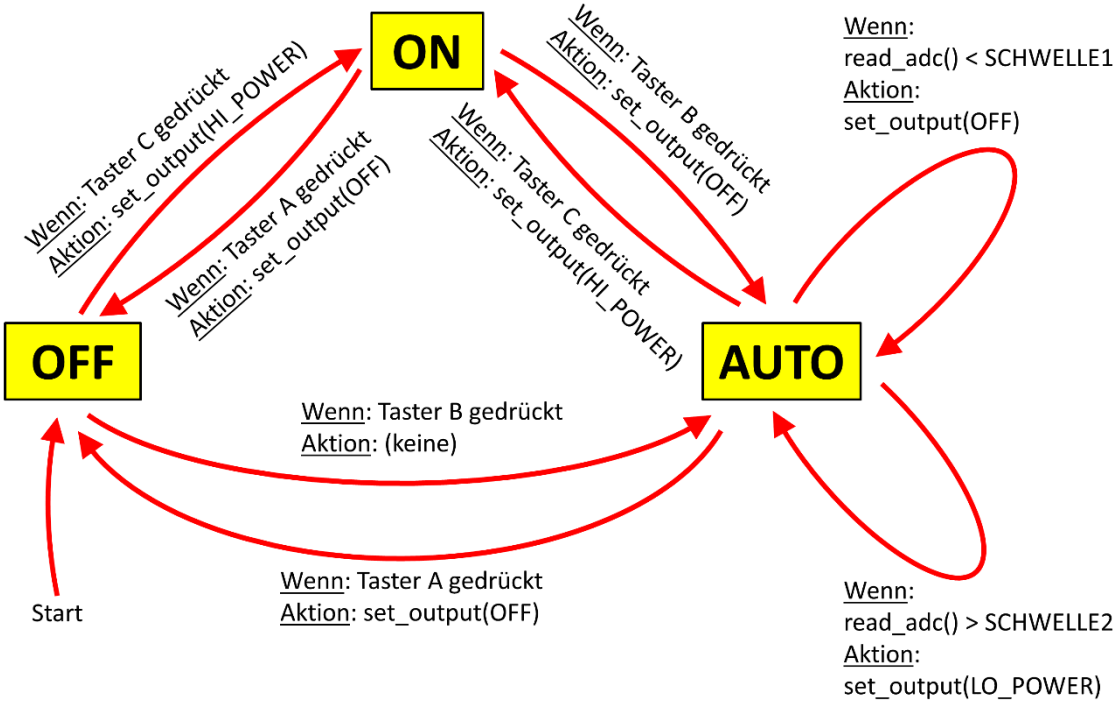
Aufgabe 1 von 5: Programmierung (ca. 26 Punkte ≙ 26 Minuten)

Es soll die Software zur Steuerung einer Klimaanlage entwickelt werden.



- Es wird ein ATmega328P-Mikrocontroller verwendet (Takt = 16 MHz).
- Der Motor der Klimaanlage wird über den Anschluss PB1 angesteuert: Wenn an PB1 eine logische „1“ ausgegeben wird, dann läuft die Klimaanlage. Durch Ausgabe eines PWM-Signals an PB1 kann die Klimaanlage mit reduzierter Leistung betrieben werden.
- An den Anschlüssen PC0, PC1 und PC2 sind die Taster „A“, „B“ und „C“ angeschlossen (siehe nebenstehende Abbildung). Das Drücken eines Tasters führt zu einer logischen „0“ an dem entsprechenden Eingang (wie im Praktikum).
- Im Automatikmodus geht die Klimaanlage je nach aktueller Temperatur an oder aus. Am Eingang des Analog-Digital-Wandlers ist dazu ein Temperatursensor angeschlossen.
- Hinweis: In diesem Programm wird der 16-Bit-Timer/Counter1 verwendet und nicht der 8-Bit-Timer/Counter0.

Das Verhalten der Steuerung wird durch einen endlichen Automaten beschrieben:



1.1. Nach Aufruf der Funktion „init_timer“ (vergl. Seite 4 und Tab. 20-6) zählt der Timer/Counter1 automatisch immer wieder von null bis zu einem bestimmten Maximalwert. Wie groß ist dieser Maximalwert?

20.14.1. TC1 Control Register A

COM1A1/ COM1B1	COM1A0/ COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM1[3:0] = 14 or 15: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on Compare Match, set OC1A/OC1B at BOTTOM (non-inverting mode)
1	1	Set OC1A/OC1B on Compare Match, clear OC1A/OC1B at BOTTOM (inverting mode)

Table 20-6. Waveform Generation Mode Bit Description

Mode	WGM13	WGM12 (CTC1)(1)	WGM11 (PWM11)(1)	WGM10 (PWM10)(1)	Timer/ Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM

Table 20-7. Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0		1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	clkI/O/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)

1.2. Welche Frequenz wird an PB1 ausgegeben, wenn das PWM-Signal aktiv ist?

1.3. Welcher Tastgrad wird nach dem Aufruf von „set_output(LO_POWER)“ an PB1 ausgegeben?

28.9.1. ADC Multiplexer Selection Register

MUX[3:0]	Single Ended Input
0000	ADC0
0001	ADC1
0010	ADC2
0011	ADC3
0100	ADC4
0101	ADC5
0110	ADC6
0111	ADC7
1000	Temperature sensor

1.4. Wozu dient die Zuweisung an „PORTC“ in der Funktion „init_buttons“?

1.5. Der AD-Wandler wird durch Aufruf der Funktion „init_adc“ initialisiert. An welchem Eingang muss das Analog-signal (= der Temperatursensor) angeschlossen werden? (Kurze Begründung erforderlich!)

1.6. Die Funktion „read_adc“ dient zum Einlesen eines Analogwerts durch den Analog-Digital-Wandler. In welchem Wertebereich (min…max) kann sich der Rückgabewert dieser Funktion befinden?

1.7. Programmieren Sie den fehlenden Rest des Hauptprogramms (Funktion „main“).

```
// Takt = 16 MHz
#define F_CPU 16000000UL
#include <avr/io.h>

// Externe Anschlüsse
#define KLIMAANLAGE PB1
#define TASTER_A    PC0
#define TASTER_B    PC1
#define TASTER_C    PC2

// Temperatur-Schwellenwerte für Automatikmodus
#define SCHWELLE1 0.4
#define SCHWELLE2 0.6
```

```
// Timer/Counter1: PWM-Modus einschalten
void init_timer1(void)
{
    TCCR1A = _BV(WGM11) | _BV(WGM10) | _BV(COM1A1);
    TCCR1B = _BV(WGM12) | _BV(CS11);
}

// Taster "A" an PC0, Taster "B" an PC1, Taster "C" an PC2
void init_buttons(void)
{
    PORTC = _BV(TASTER_A) | _BV(TASTER_B) | _BV(TASTER_C);
}

// Analog-Digital-Wandler initialisieren
void init_adc(void)
{
    ADMUX  = _BV(REFS0) | _BV(MUX0) | _BV(MUX1) | _BV(MUX2);
    ADCSRA = _BV(ADEN) | _BV(ADPS2) | _BV(ADPS1) | _BV(ADPS0);
}

// Einen Wert vom Analog-Eingang einlesen
double read_adc(void)
{
    ADCSRA |= _BV(ADSC);           // Wandlung starten
    while(ADCSRA & _BV(ADSC)) { }  // Auf das Ende der Wandlung warten
    return ADC / 1023.0;           // Ergebnis normieren und zurückgeben
}

// Klimaanlage auf hohe/mittere Leistung schalten bzw. ausschalten
#define OFF      0
#define LO_POWER 1
#define HI_POWER 2

void set_output(int power)
{
    if(power == OFF     ) { DDRB &= ~_BV(KLIMAANLAGE);                }
    if(power == LO_POWER) { DDRB |=  _BV(KLIMAANLAGE); OCR1A =  256; }
    if(power == HI_POWER) { DDRB |=  _BV(KLIMAANLAGE); OCR1A = 1023; }
    // Hinweis: Timer/Counter1 ist ein 16-Bit-Timer,
    // OCR1A hat eine Größe von 16 Bits!
}

// Hauptprogramm, endlicher Automat mit drei Zuständen
#define STATE_OFF  0
#define STATE_ON   1
#define STATE_AUTO 2

int main(void)
{
    init_timer1();
    init_buttons();
    init_adc();

    int state = STATE_OFF;
    set_output(OFF);
```

```
while (1)
{
    switch(state)
    {
    case STATE_OFF:
```

Aufgabe 2 von 5: Verschiedenes (ca. 15 Punkte ≙ 15 Minuten)

2.1. Wie lauten die Ergebnisse der folgenden Berechnungen mit 8-Bit-Zahlen?

Ergebnisse als Binärzahl schreiben!

a) 0b00001101 | 0b00001011 =

b) 0b00001101 + 0b00001011 =

c) 0b00011101 | 0b00001101 | 0b00001100 =

d) 0b00011101 + 0b00001101 + 0b00001100 =

e) _BV(3) | _BV(3) =

f) _BV(3) + _BV(3) =

g) ~_BV(3) | _BV(3) =

h) ~_BV(3) & _BV(3) =

2.2. Skizzieren Sie eine Schaltung zum Anschluss eines einfachen (Tast-)Schalters an den Port PC1 eines ATmega-328P-Mikrocontrollers. Der Schalter, der Eingang PC1 des Mikrocontrollers sowie der Pullup-Widerstand am bzw. im Mikrocontroller sollen in Ihrer Skizze erkennbar sein.

Beantworten Sie auch die folgenden beiden Fragen:

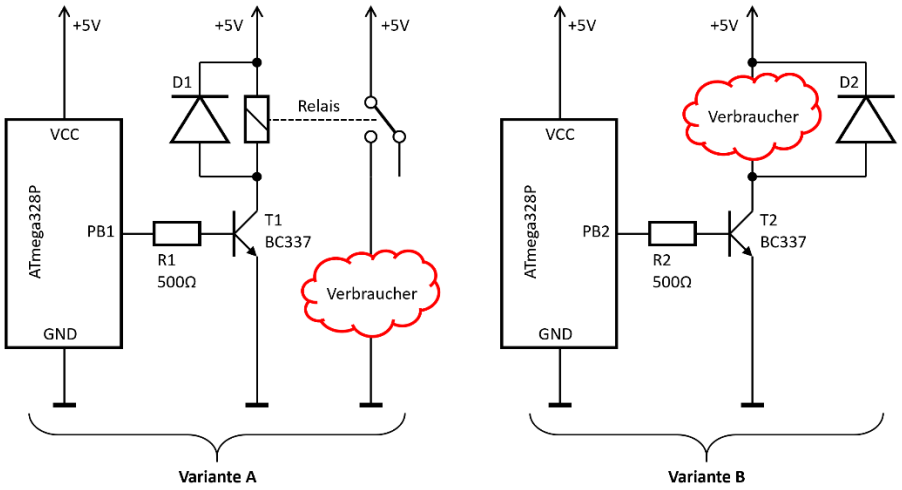
a. Warum ist der Pullup-Widerstand erforderlich?
b. Mit welchem Befehl wird der Pullup-Widerstand am Eingang PC1 aktiviert?

2.3. Der Zustand eines (Tast-)Schalters am Eingang PB1 eines ATmega328P-Mikrocontrollers soll eingelesen werden. Die folgenden Befehle funktionieren allerdings nicht. Korrigieren Sie den abgebildeten Quelltext!

```
// Hinweis: Die int-Variable "taster_gedrueckt" ist weiter oben definiert...
if((PORTB & 0b00000001) == 0)
{
    // Taster gedrückt
    taster_gedrueckt = 1;
}
else
{
    // Taster nicht gedrückt
    taster_gedrueckt = 0;
}
```

Aufgabe 3 von 5: GPIO-Ports (ca. 12 Punkte ≙ 12 Minuten)

Die linke Abbildung zeigt die Ansteuerung eines Verbrauchers durch ein Relais (Variante A), die rechte Abbildung zeigt die Ansteuerung eines Verbrauchers durch einen einzelnen Transistor (Variante B). Für die Basis-Emitter-Spannung gilt bei beiden Transistoren: U_{BE} = 0,5 Volt. Die Spannung an den Anschlüssen PB1 und PB2 beträgt 4,5 Volt (falls diese eingeschaltet sind). Der Stromverstärkungsfaktor („Großsignalverstärkung“) der Transistoren ist B = 100.



3.1. Welchen Strom I_{PB1} muss der Anschluss PB1 (Var. A) zum Einschalten des Verbrauchers liefern?

3.2. Wie verändert sich der Strom durch die (eingeschaltete) Relais-Spule, wenn der Widerstand an der Basis von T1 auf R1 = 250 Ω verkleinert wird? (Begründung oder kurze Rechnung!)

3.3. Wie verändert sich das Verhalten der (linken) Schaltung A, wenn die Diode D1 aus Versehen falsch herum eingebaut wird (= wenn die beiden Anschlüsse der Diode vertauscht werden)?

3.4. Welchen Strom I_{PB2} muss der Anschluss PB2 (Var. B) zum Einschalten des Verbrauchers liefern?

3.5. Ist die Diode D2 erforderlich (Begründung!), wenn in der Schaltungsvariante B …

- … als Verbraucher eine Leuchtdiode (LED) angeschlossen ist?

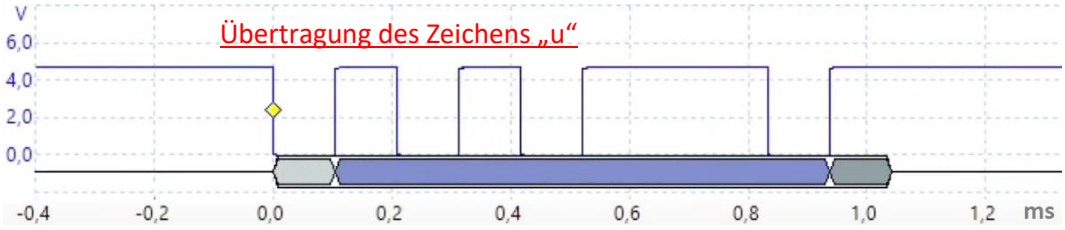
- … als Verbraucher ein kleiner DC-Motor angeschlossen ist?

3.6. Nennen Sie jeweils einen Vorteil und einen Nachteil von Variante A (Verbraucher mit Relais schalten) und von Variante B (Verbraucher mit Transistor schalten).

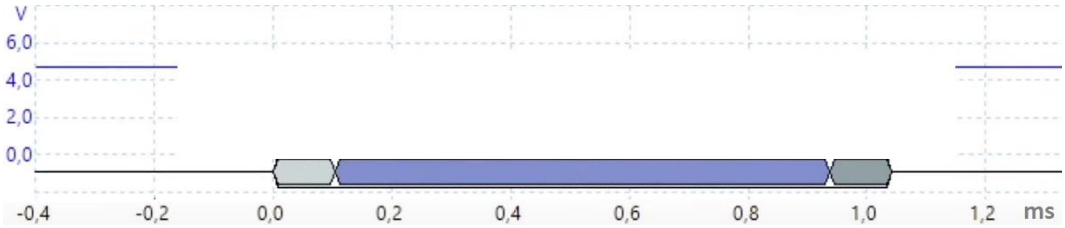
	Variante A	Variante B
Vorteil		
Nachteil		

Aufgabe 4 von 5: Serielle Schnittstelle (ca. 13 Punkte ≙ 13 Minuten)

4.1. Über die serielle Schnittstelle eines ATmega328P wird zunächst das Zeichen „u“ (der ASCII-Code als Dezimalzahl ist 117_{DEZ}) übertragen. Anschließend wird das Zeichen „D“ gesendet, der ASCII-Code dieses Zeichens ist 68_{DEZ}. Die Übertragungsparameter der Schnittstelle sind: 9,6 kbit/s; 8 Datenbits; kein Paritätsbit; 1 Stoppbit.



Skizzieren Sie den Spannungsverlauf, der sich an TXD/PD1 (= Ausgang der seriellen Schnittstelle) beim Senden des zweiten Zeichens ergibt.



4.2. Wie lange dauert die Übertragung einer mp4-Datei mit einer Größe von 500.000 Bytes über eine serielle Schnittstelle minimal? Die Datenübertragungsrate der Schnittstelle betrage 9,6 kbit/s, es werden acht Daten-, ein Start- und ein Stoppbit gesendet (und kein Paritätsbit).

4.3. Durch einen Konfigurationsfehler des Senders werden 2 Stoppbits gesendet (korrekt wäre 1 Stoppbit gewesen). Beschreiben Sie den Effekt dieses Konfigurationsfehlers auf den Ablauf der seriellen Datenübertragung.

- Funktioniert die Datenübertragung noch? Wer erkennt ggf. den Fehler?

4.4. An den Anschlüssen TXD und RXD des Mikrocontrollers werden während einer seriellen Datenübertragung die einzelnen Bits mit Spannungen von 0 Volt (logische Null) bzw. 5 Volt (logische Eins) übertragen.

Serielle Datenübertragung über größere Distanzen basiert häufig auf dem RS-232-Standard. Welche Spannungen werden hier beim Senden eines Datenbits verwendet?

„Logische 0“: _____ „Logische 1“: _____

4.5. Auf der Arduino-Platine aus dem Praktikum befindet sich ein ATmega328P-Mikrocontroller. Es ist problemlos möglich, Daten über die serielle Schnittstelle des Microcontrollers an einen PC zu senden. Dies funktioniert sogar, obwohl moderne PCs über gar keine „klassischen“ seriellen Schnittstellen mehr verfügen, sondern nur über einige USB-Anschlüsse.

Beschreiben Sie in wenigen Stichworten (oder mit einer passenden Skizze), wie die serielle Datenübertragung vom Mikrocontroller bis zur PC-Applikation funktioniert.

Aufgabe 5 von 5: C-Programmierung auf Mikrocontrollern (ca. 14 Punkte ≙ 14 Minuten)

In einem C-Programm für den Mikrocontroller ATmega328P, der mit 16 MHz Taktfrequenz betrieben wird, soll die Anzahl der Millisekunden seit dem Programmstart ermittelt werden. Dazu wird der Timer/Counter0 so eingerichtet, dass er 1000 Interrupts pro Sekunde auslöst. Die Anzahl der Millisekunden seit dem Programmstart kann über die Funktion „Millis“ abgefragt werden.

```
#define F_CPU 16000000UL
#include <inttypes.h>
#include <avr/io.h>
#include <avr/interrupt.h>

// Globale Variable: Millisekunden seit Programmstart
volatile uint32_t _millis_intern = 0;

ISR(TIMER0_COMPA_vect)  // Interrupt-Frequenz = 1 kHz
{
    _millis_intern++;
}

// Millisekunden seit Programmstart abfragen
uint32_t Millis(void)
{
    uint32_t result = 0;
    cli(); result = _millis_intern; sei();
    return result;
}

// Timer initialisieren, Interrupt-Frequenz = 1 kHz
void init_timer(void)
{
    TCCR0A = _BV(WGM01);             // CTC-Modus
    TCCR0B = _BV(CS01) + _BV(CS00);  // Prescaler = 64
    OCR0A  = 249;                    // Vergleichswert für Timer
    TIMSK0 = _BV(OCIE0A);            // Compare Match Interrupt
    sei();                           // Interruptsystem einschalten
}
```

5.1. Warum muss die Variable „_millis_intern“ als „volatile“ deklariert werden?

5.2. Wozu dienen die Befehle „cli()“ und „sei()“ in der Funktion „Millis“?

5.3. Nach welcher Zeit erreicht die Variablem „_millis_intern“ ihren Maximalwert und es kommt zu einem Überlauf?

5.4. Nach welcher Zeit würde es zu einem Überlauf kommen, wenn „_millis_intern“ als uint64_t definiert wäre?

5.5. Mit welcher Frequenz werden die Interrupts ausgelöst, wenn das Register „OCR0A“ auf den Wert 49 (statt auf den Wert 249) gesetzt wird?

5.6. Nennen Sie jeweils einen Vorteil, der sich ergibt, wenn ein Rechner nach der Von-Neumann-Architektur bzw. nach der Harvard-Architektur aufgebaut wird. Welche der beiden Architekturen eignet sich eher für einen Universal-PC, welche für ein Mikrocontroller-basiertes Steuergerät („Embedded System“)?

	Vorteil	für Universal-PC oder für Steuergerät geeignet?
Von-Neumann		
Harvard		