

{SITEMAP}

[Timer1CTCInterrupt](#)

C-Programm für den ATMEGA 328P 16MHz

Timer 1 COUNTER COMPARE MATCH

Der Compare Match Interrupt des Timer 1 wird genutzt, um ein Taktsignal zu erzeugen. Die Pins PB0, PB1 und PB2 werden bei Programmstart eingelesen und stellen den Prescaler des Timers ein. Eine Änderung der Taktrate ist also nur nach Neustart möglich. Am Port C liegt das HIGH Byte des Compare Registers. Das LOW Byte ist immer Null. Auch dieses wird nur beim Programmstart eingelesen.

Programmieren Sie den Compare Match Interrupt so, dass der Pin PB3 bei jedem Interrupt togglet. Berechnen Sie die Periodendauer für das langsamste und für das schnellste Signal an Pin PB3.

Beachte: Trace, Kommentare (d/e), Symbole (e), Berechnungen am Zettel.

[ADC 1V 8Bit Timer0](#)

[ADC 1V 8Bit Timer2](#)

C-Programm für den ATMEGA 328P 16MHz

ADU

An Pin ADC0 hängt eine analoge Spannung zwischen 0 und 1 Volt (das Ausgangssignal eines MP3 Players). Dieses Signal soll mit konstanter Abtastrate gesampelt werden. Die Abtastrate wird mit Timer 0 erzeugt und beträgt ca. 1ms. Das Ergebnis der Wandlung wird am Port B als 8Bit Zahlenwert ausgegeben.

Beachte: Trace, Kommentare (d/e), Symbole (e), Berechnungen am Zettel.

[PWM \(Timer0 toggelt OC0A\)](#)

[PWM etwas umständlich \(Timer1 toggelt OC0A\)](#)

C-Programm für den ATMEGA 328P 16MHz

ADU + FAST PWM

Ein Potentiometer hängt auf Pin ADC0 zwischen Masse und der Versorgung VCC (5V). Dieser Wert wird eingelesen steuert die Helligkeit einer Led am Pin OC0A (Led gegen Masse, 0V dunkel, 5V maximale Helligkeit).

Beachte: Trace, Kommentare (d/e), Symbole (e), Berechnungen am Zettel.

[ADU \(manueller Trigger, keine Interrupts\)](#)

C-Programm für den ATMEGA 328P 16MHz

ADU

An Pin ADC4 hängt eine analoge Spannung zwischen 0 und 5 Volt. Diese Spannung soll in einen 10 Bit Zahlenwert umgewandelt werden, wenn am Pin ICP1 (Input Capture Pin) eine fallende Flanke angelegt wird.

Der gewandelte Zahlenwert wird seriell über die Pins PB0 und PB1 ausgegeben.

Ausgabe der Zahlenfolge 01001:

Data (PB0): 000 111 000 000 111

Clock(PB1): 010 010 010 010 010 usw.

Beachte: Trace, Kommentare (d/e), Symbole (e), Berechnungen am Zettel.

CTC zur Takterzeugung; Periode einlesen

C-Programm für den ATMEGA 328P 16MHz

COUNTER COMPARE MATCH

Ein externer Takt hängt am Eingang T0 und läuft mit $f=1\text{MHz}$. Von Port B wird ein Wert zwischen 10 und 99 (dezimal) eingelesen, der einer Periodendauer zwischen 10 und 99ms entspricht. Am Port C (Pin wählbar) wird ein Takt mit der Periodendauer dieser eingelesenen Zeit ausgegeben. Verwende Timer1 mit Compare Match Register. Die Takterzeugung muss im Compare Match Interrupt erfolgen. Es genügt eine Genauigkeit von 10%.

Beachte: Trace, Kommentare (d/e), Symbole (e), Berechnungen am Zettel.

Serial In - Serial 2 Parallel conversion - write to EEPROM

C-Programm für den ATMEGA 328P 16MHz

Serial Data to EEPROM

Daten werden vom Port C seriell eingelesen (PC0=SDA Data, PC1=CLK Clock); die eingelesenen Bits werden zu einem Byte zusammengestellt (Serien – Parallel- Wandlung) und nacheinander auf die Adressen 0xF, 0x10 usw. ins EEPROM geschrieben.

Lesen einer Null: steigende Flanke an CLK und SDA = 0

Lesen einer Eins: steigende Flanke an CLK und SDA = 1

Es wird ohne Interrupt gearbeitet. Beachte: Trace, Kommentare (d/e), Symbole (e), Berechnungen am Zettel.

I2C Slave Slave 1.0

Es soll ein Programm geschrieben werden welches den I²C Bus verwendet um Daten zwischen zwei Arduino Unos zu verschicken. Es soll ein Master und ein Slave programmiert werden. Der Master schickt dem Slave Daten und der Slave gibt diese auf dem Display aus.

Automatisches Tor

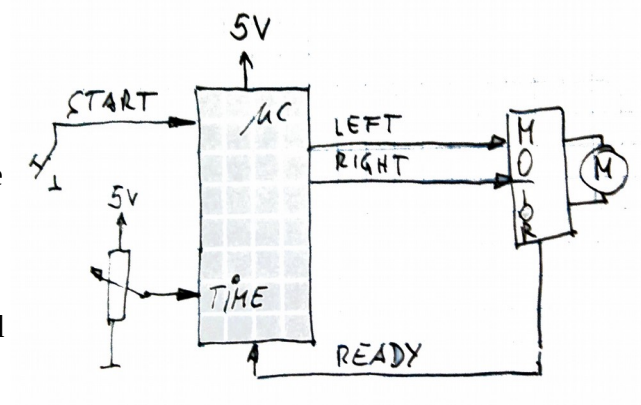
Garagentor

Inputsignale:

Start: öffnet das Tor und schließt es wieder durch kurzen Tastendruck (Ruhezustand: HIGH)

Analogsignal zwischen 0 und 5V: stellt die max. Öffnungszeit ein (0 bis 100s)

Ready: die Öffnung ist vorzeitig beendet, das Signal löst einen Interrupt aus; der Motor wird gestoppt und die bis jetzt vergangene Öffnungszeit als neue Soll-Öffnungszeit abgespeichert.



Output: MotorLeft, MotorRight; Stopp, wenn beide Signale auf Null

Ein kurzer Tastendruck (fallende Flanke an START) öffnet oder schließt das Tor, je nachdem ob es vorher offen oder geschlossen war.

Jetzt wird geprüft, wie lange es offen sein soll. Dazu dreht man am Potentiometer und gibt dadurch eine Zeitspanne zwischen 0 und 100s ein. Sollte das Tor aber geschlossen sein, bevor die am Poti eingestellte Zeit abgelaufen ist, dann wird von der Motorsteuerung ein READY Signal erzeugt, das den Motor abstellt und die bis dahin abgelaufenen Zeit als neue Öffnungszeit abspeichert.