Skriptum Mikrocontroller

Teil 7 Analog-Digital Wandler

DI(FH) Andreas Pötscher, HTL Litec

Inhaltsverzeichnis

Datum	Beschreibung der durchgeführten Änderung	Autor
10.09.2024	V1.0 . . . 1.Version	Andreas Pötscher, HTL Linz–Paul-Hahn-Straße (LiTec)

Analoge Signale

Analoge Signale bei einem Microcontroller sind elektrische Signale, die kontinuierlich variiert werden und Werte innerhalb eines bestimmten Bereichs repräsentieren, im Gegensatz zu digitalen Signalen, die nur diskrete Werte (normalerweise 0 und 1) haben. Analoge Signale können eine breite Palette von Werten annehmen.

Sie sind in vielen Anwendungen wichtig, da sie es ermöglichen, kontinuierliche und variable Daten von der physischen Welt zu erfassen und zu verarbeiten. Beispiele für Anwendungen von analogen Signalen in Microcontrollern sind Temperatursensoren, Lichtsensoren, Messung von Batteriespannungen und Motorsteuerungen, um nur einige zu nennen.

Microcontroller sind in der Regel mit analogen Eingabe- und Ausgabefunktionen ausgestattet, um mit analogen Signalen zu arbeiten.

Analoge Eingänge

Ein Microcontroller verfügt über analoge Eingangspins, die in der Lage sind, analoge Spannungen von externen Sensoren oder Schaltungen zu erfassen. Diese Eingänge wandeln die kontinuierliche Spannung in ein digitales Signal um, das vom Microcontroller verarbeitet werden kann. Dieser Vorgang wird als Analog-Digital-Umwandlung (ADC) bezeichnet.

Analoge Ausgänge

Einige Microcontroller bieten auch analoge Ausgangspins, die in der Lage sind, variable Spannungen auszugeben. Diese werden oft als Digital-Analog-Umwandlung (DAC) bezeichnet.

Typische Spannungspegel

Analoge Signale können eine Vielzahl von Spannungspegeln haben, abhängig von den Anforderungen der spezifischen Anwendung und der Elektronik, die verwendet wird. Im Allgemeinen können analoge Spannungen über einen breiten Bereich variieren. Hier sind einige der gebräuchlichsten Spannungspegel für analoge Signale:

- **0 bis 5 Volt (0-5V):** Dies ist einer der häufigsten Spannungsbereiche für analoge Signale in vielen Elektronik- und Mikrocontroller-Anwendungen. Es wird oft in Logikpegeln und Sensoren gefunden.
- **0 bis 3,3 Volt (0-3,3V):** Dieser Spannungsbereich ist besonders verbreitet in Low-Power-Anwendungen und in Verbindung mit Mikrocontrollern und Mikroprozessoren, die mit 3,3V betrieben werden.
- **± 10 Volt (±10V):** In einigen industriellen Anwendungen, wie der Steuerung von Motoren oder der Messung von Signalen in Labors, werden ±10V-Signale verwendet.
- **± 5 Volt (±5V):** Ähnlich wie ±10V-Signale, aber mit einem kleineren Bereich. Dies wird auch in verschiedenen industriellen und wissenschaftlichen Anwendungen eingesetzt.
- **0 bis 10 Volt (0-10V):** Dieser Bereich ist in einigen Steuerungssystemen und Audioanwendungen üblich.

- **4 bis 20 Milliampere (4-20mA):** In der Prozessautomatisierung werden oft Stromsignale anstelle von Spannungssignalen verwendet. 4-20mA-Signale sind gängig und können eine breite Palette von Messungen darstellen.
- **± 1 Volt (±1V):** In einigen Präzisionsanwendungen, wie Messinstrumenten und Präzisionsverstärkern, werden ±1V-Signale verwendet.

Es ist wichtig zu beachten, dass die Wahl des Spannungsbereichs von den Anforderungen der Anwendung, der Genauigkeit und der verfügbaren Elektronik abhängt. Bei der Verwendung von Mikrocontrollern oder anderen Elektronikkomponenten sollten Sie immer die Datenblätter und Spezifikationen konsultieren, um sicherzustellen, dass Sie die richtigen Spannungsbereiche für Ihre Anwendung verwenden.

Analog Digital Converter

Ein Analog-Digital-Umsetzer (ADC) ist eine elektronische Schaltung oder ein Bauteil in einem Mikrocontroller, der analoge Eingangssignale in digitale Werte umwandelt. Dieser Prozess ist entscheidend, um analoge Informationen, wie sie von Sensoren erfasst werden, in eine digitale Form zu bringen, die von einem Mikrocontroller oder einem Computer verarbeitet werden kann.

Es gibt verschiedene Arten von ADCs, darunter sukzessive Approximations-ADCs, Delta-Sigma-ADCs und Pipeline-ADCs, die jeweils unterschiedliche Methoden für die Abtastung und Quantisierung verwenden. Die Wahl des ADC-Typs hängt von den Anforderungen der Anwendung hinsichtlich Geschwindigkeit, Genauigkeit und Kosten ab.

Hier ist eine grundlegende Erklärung, wie ein ADC funktioniert:

Abtastung (Sampling)

Der ADC beginnt damit, das analoge Eingangssignal in regelmäßigen Abständen zu “abtasten”. Das bedeutet, dass es zu bestimmten Zeitpunkten den aktuellen Wert des analogen Signals misst. Die Frequenz, mit der diese Abtastungen durchgeführt werden, wird als Abtastrate bezeichnet und ist ein wichtiger Parameter des ADC.

Quantisierung

Nachdem das analoge Signal abgetastet wurde, wird der gemessene analoge Wert in eine begrenzte Anzahl von digitalen Werten umgewandelt. Dieser Schritt wird Quantisierung genannt. Die Genauigkeit der Quantisierung wird durch die Bit-Anzahl (Auflösung) des ADC bestimmt. Ein ADC mit einer höheren Bit-Anzahl kann das analoge Signal präziser quantisieren.

Referenzspannung

Um das analoge Signal zu quantisieren, vergleicht der ADC den gemessenen analogen Wert mit einer Referenzspannung, die im ADC eingestellt ist. Der ADC bestimmt dann, in welchem Bereich dieser Referenzspannung der gemessene Wert liegt. Dieser Bereich entspricht einem bestimmten digitalen Wert.

Beispiel

Um die Quantisierung eines 0 bis 5 Volt analogen Signals mit einem ADC mit 10 Bit Auflösung zu verstehen, müssen wir zuerst verstehen, wie der ADC den Spannungsbereich auf die verfügbaren digitalen Werte aufteilt.

Ein ADC mit 10 Bit Auflösung kann 2^{10} (also 1024) verschiedene digitale Werte darstellen. Der Spannungsbereich, den der ADC quantisiert, reicht von 0 Volt bis 5 Volt.

Um die Schritte der Quantisierung zu berechnen, teilen Sie den gesamten Spannungsbereich (5 Volt) durch die Anzahl der verfügbaren digitalen Werte (1024). Das ergibt:

$Schrittweite = \frac{5V}{1024} = 0,00488V$

Das bedeutet, dass jeder digitale Wert, den der ADC ausgibt, eine Spannungsänderung von 4,88 mV repräsentiert. Dies ist die kleinste auflösbare Spannungsänderung.

Wenn Sie nun ein beliebiges analoges Eingangssignal an den ADC anlegen, wird es in Schritten von 4,88 mV quantisiert. Hier sind einige Beispiele:

- Wenn das analoge Signal 0 Volt beträgt, gibt der ADC den digitalen Wert 0 aus.
- Wenn das analoge Signal 2,44 Volt beträgt, gibt der ADC den digitalen Wert 500 aus (2,44 V / 0,00488 V = 500).
- Wenn das analoge Signal 5 Volt beträgt, gibt der ADC den digitalen Wert 1023 aus (da dies der höchste digitale Wert ist).

Auf diese Weise kann der ADC das analoge Eingangssignal in 1024 diskrete Werte quantisieren, wobei jeder Wert einem bestimmten Spannungsbereich entspricht. Die Auflösung von 10 Bit ermöglicht eine relativ hohe Genauigkeit, da Sie viele verschiedene Spannungsniveaus innerhalb des 0-5 V-Bereichs unterscheiden können.

Um einen digitalen Wert in eine Spannung zurückzurechnen, verwenden Sie die Schrittweite des ADC (Analog-Digital-Umsetzers), die wir zuvor berechnet haben, zusammen mit dem digitalen Wert selbst. Hier ist ein Beispiel:

Angenommen, Sie haben einen ADC mit 10 Bit Auflösung, der ein analoges Signal von 0 bis 5 Volt quantisiert. Wir haben bereits herausgefunden, dass die Schrittweite (der kleinste auflösbare Unterschied) 4,88 mV beträgt.

Jetzt haben Sie einen digitalen Wert aus dem ADC, sagen wir 500. Um diesen digitalen Wert in die entsprechende Spannung umzurechnen, verwenden Sie die folgende Formel:

$Spannung = DigitalerWert * Schrittweite$

In diesem Fall:

$Spannung = 500 * 0.00488V = 2,44V$

Der digitale Wert 500 entspricht also einer Spannung von 2,44 Volt aufgrund der Schrittweite des ADC.

Auf die gleiche Weise können Sie jeden anderen digitalen Wert, den der ADC ausgibt, in eine entsprechende Spannung umrechnen. Dies ermöglicht es Ihnen, das ursprüngliche analoge Signal basierend auf den vom ADC erfassten digitalen Werten wiederherzustellen.

ADC beim Atmega 2560

Der Arduino Mega ist ein Mikrocontroller-Board, das über eine Vielzahl von analogen Eingangspins verfügt, die als "AnalogPins" oder "Analogeingangspins" bezeichnet werden. Diese Pins sind speziell für die Erfassung analoger Signale ausgelegt und können verwendet werden, um Spannungen oder andere analoge Größen zu messen. Im Fall des Arduino Mega gibt es insgesamt 16 analoge Eingangspins, die mit den Bezeichnungen A0 bis A15 gekennzeichnet sind.

Sie befinden sich auf den GPIOs PF0 bis PF7 und PK0 bis PK7.

ADC Channel auswählen

Das ADMUX-Register hat mehrere Bits, die verwendet werden, um den ADC-Kanal auszuwählen. Diese Bits sind normalerweise als MUX0, MUX1, MUX2, MUX3, MUX4 und MUX5 bezeichnet.

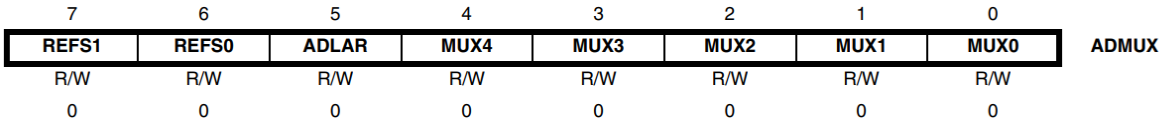

Abbildung 1: ADMUX Register

Bei einem ATmega2560 oder Arduino Mega stehen Ihnen 16 analoge Eingangskanäle zur Verfügung, die von ADC0 bis ADC15 nummeriert sind. Um einen bestimmten Kanal auszuwählen, setzen Sie die entsprechenden Bits in ADMUX auf die gewünschten Werte.

MUX2	MUX1	MUX0	CH
0	0	0	ADC0
0	0	1	ADC1
0	1	0	ADC2
0	1	1	ADC3
1	0	0	ADC4
1	0	1	ADC5
1	1	0	ADC6
1	1	1	ADC7

Zum Beispiel, wenn Sie ADC3 (Analog Pin A3 auf dem Arduino) auswählen möchten, setzen Sie die Bits MUX0 und MUX1 auf 1 und lassen die anderen Bits auf 0:

```
ADMUX |= (1 << MUX0) | (1 << MUX1);
```

Referenzspannung festlegen

Das ADMUX-Register enthält auch Bits, um die Referenzspannung für den ADC festzulegen. Die Referenzspannung ist wichtig, da sie die Skala bestimmt, in der die analoge Spannung gemessen wird. Die Bits für die Auswahl der Referenzspannung heißen REFS0 und REFS1.

REFS1	REFS0	CH
0	0	AREF Pin
0	1	AVCC (5V)
1	0	1,1V
1	1	2,56V

Zum Beispiel, um die Referenzspannung auf 5V-Referenzspannung umzuschalten, setzen Sie die Bits wie folgt:

```
ADMUX |= (1 << REFS0);
```

Prescaler

Die Auswahl des richtigen Prescalers für den Analog-Digital-Wandler (ADC) auf einem AVR-Mikrocontroller, wie dem ATmega2560 (verwendet auf dem Arduino Mega), ist entscheidend, um die Abtastrate und die Genauigkeit der ADC-Messungen zu steuern. Der Prescaler bestimmt, wie schnell der ADC die analoge Spannung messen wird. Dabei wird mit einem höheren Prescaler die Genauigkeit höher aber die Geschwindigkeit langsamer.

Um die vollen 10 Bit Auflösung zu erhalten benötigt der ADC Converter eine Eingangsfrequenz zwischen 50 kHz und 200 kHz. Somit muss bei einer CPU Taktfrequenz von 16Mhz ein Prescaler von 128 eingestellt werden. Das ergibt dann eine ADC Frequenz von 125kHz. Da eine Sampling Zyklus ca. 13 Takte benötigt, lässt sich damit bei einem Kanal eine Abtastrate von ca. 9kHz erreichen. Für genauere Infos siehe Datenblatt.

Der ADC-Prescaler wird im ADCSRA-Register (Analog-Digital Converter Control and Status Register A) festgelegt. Die relevanten Bits für die Auswahl des Prescalers sind ADPS0, ADPS1, und ADPS2. Die Kombination dieser Bits bestimmt den Prescaler, der auf die Systemtaktfrequenz des Mikrocontrollers angewendet wird, um die ADC-Frequenz zu erhalten

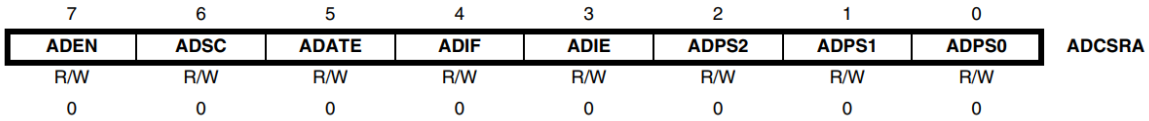

Abbildung 2: ADCSRA Register

ADPS2	ADPS1	ADPS0	PRESCALER
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Dementsprechend wird ein Prescaler von 128 wie folgt eingestellt.

```
ADCSRA |= (1 << ADPS0) | (1 << ADPS1) | (1 << ADPS2);
```

ADC Starten

Als ersten Schritt muss der ADC aktiviert werden. Dies geschieht mit dem Bit ADEN im Register ADCSRA.

Um eine einzige Wandlung zu starten muss das Bit ADSC im Register ADCSRA gesetzt werden. Dieses Bit bleibt so lange High so lange die Wandlung dauert und wird dann automatisch auf 0 zurückgesetzt. Folgender Code startet eine Wandlung und wartet dann ab bis diese Fertig ist.

```
ADCSRA |= 1 << ADSC;
while(ADCSRA >> ADSC & 0x01);
```

Alternativ dazu kann die Wandlung automatisch gestartet werden. Dazu muss das Autotrigger Bit ADATE im Register ADCSRA gesetzt werden.

Standardmäßig ist dabei als Modus Free-Running Mode eingestellt. In diesem Modus muss mit dem Bit ADSC die erste Wandlung gestartet werden. Die weiteren Wandlungen erfolgen dann automatisch. Um den ADC im Freerunning Modus zu starten müssen folgende Zeilen ausgeführt werden.

```
ADCSRA |= 1 << ADEN; //Enable the ADC
ADCSRA |= 1 << ADATE; //Auto trigger
ADCSRA |= 1 << ADSC; // Start
```

Folgende weitere Autotrigger Modi können auch verwendet werden. Diese werden im Register ADCSRB mit den Bits ADTS0 bis ADTS2 eingestellt. Im Free-Running-Mode wird der ADC kontinuierlich eingelesen. Mit den anderen Modi kann die AD Wandlung bei bestimmten Ereignissen gestartet werden.

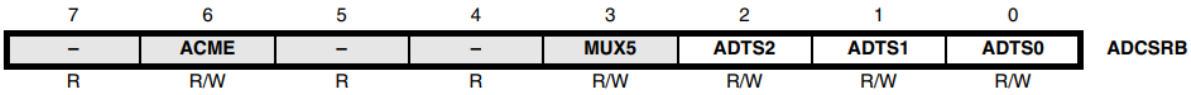

Abbildung 3: ADCSRB Register

ADTS2	ADTS1	ADTS0	Modus
0	0	0	Free-Running-Mode
0	0	1	Analog-Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match A
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter1 Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

Der Free-Running-Mode ist standardmäßig eingestellt oder wird dementsprechend gesetzt.

```
ADCSRB = 0;
```

Auslesen der gewandelten Spannung

Die Register ADCH (Analog-Digital Converter High Byte) und ADCL (Analog-Digital Converter Low Byte) werden verwendet, um das Ergebnis einer Analog-Digital-Wandlung (ADC) auf einem AVR-Mikrocontroller zu speichern. Diese Register speichern das Ergebnis der Wandlung als 10-Bit-Wert (0 bis 1023).

15	14	13	12	11	10	9	8	
–	–	–	–	–	–	ADC9	ADC8	ADCH
ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL

Abbildung 4: ADCL und ADCH Register

Um das Ergebnis einer ADC-Wandlung aus den Registern ADCH und ADCL auszulesen, müssen Sie die beiden Register zusammenlesen und den Wert entsprechend zusammensetzen.

```
uint16_t result = ADCL + ADCH * 256;
```

Elektrischer Anschluss

Es können eine vielzahl an Sensoren an einen analogen Eingang angeschlossen werden. Beispielhaft wird hier der Anschluss eines Potentiometers gezeigt. Der Poti wird, als Spannungsteiler, mit den Aussenkontakten an GND und 5V verbunden der Schleiferkontakt wird auf den analogen Eingang gelegt.

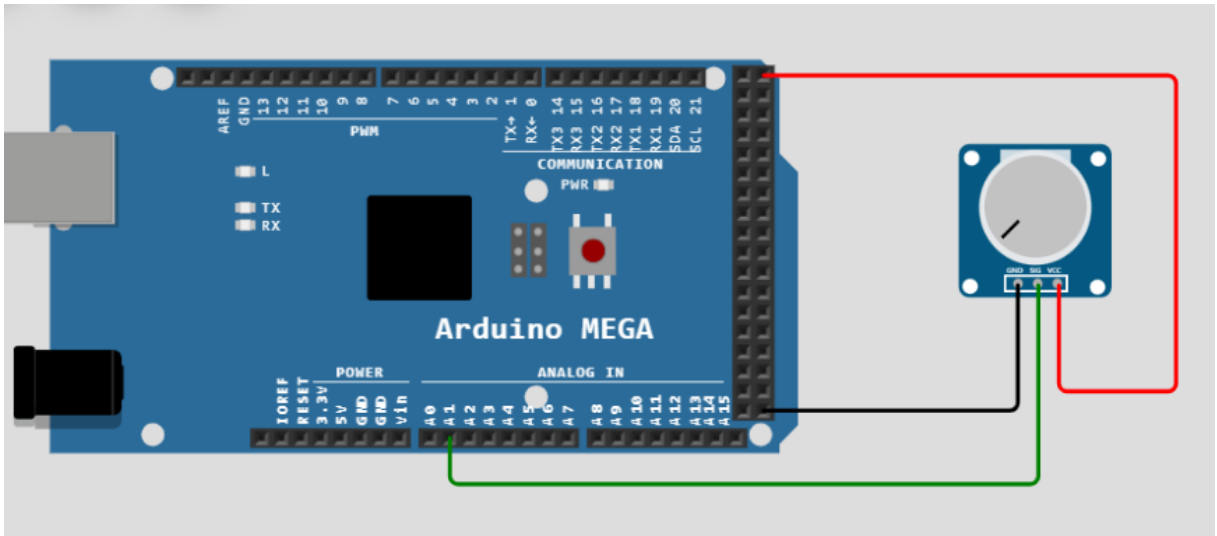

Abbildung 5: Anschluss eines Potentiometers an den Arduino Mega