Messung von AKKU-(Entlade-)Kapazitäten mit ATTINY24 in C

Kürzlich hatte ich LiPo-Akkus für kleine Euros erworben - und nun wollte ich gerne wissen, ob die eigentlich ihr Geld wert und die Angaben zur Kapazität "belastbar" sind.

Dazu benötigte ich einen Helfer, der die Akkus entlädt, dabei kontinuierlich die Spannung im Auge behält und beim Unterschreiten der Entladeschlussspannung den Akku abwirft.

Da bei der Spannungsmessung der Lastwiderstand bekannt ist, kann der aktuell fließende Strom errechnet und über die Dauer der Entladung die Kapazität des Akkus ermittelt werden.

Da mich auch der zeitliche Verlauf der Entladespannung interessiert, sollten die Messwerte protokolliert und später mit Hilfe von gnuplot visualisiert werden.

Als Controller habe ich wegen der Anzahl der benötigten Pins einen ATtiny24 gewählt.

Die Daten werden in einem 2-wire Serial Eeprom AT24C512 mit 64 kB zwischengespeichert.

Die Entladung erfolgt über einen Lastwiderstand, der über einen FET zugeschaltet wird.

Einzelnen LiPo- bzw. LiFePO4-Zellen können direkt angeschlossen werden.
Für in Serie geschaltete Akkupacks bzw. Bleiakkus ist ein Spannungsteiler erforderlich.

Der ATtiny arbeitet nebenbei als USI-TWI-Slave bzw. als USI-TWI-Master.
Als Slave stellt er die aktuellen Messwerte zum Auslesen zur Verfügung.

Der Slave kann ebenfalls genutzt werden, um einige Einstellungen auf dem ATtiny zu verändern, ohne dass dazu der Controller neu programmiert werden muss. Veränderbar sind:

- die TWI-Adresse des Slaves,
- die Höhe der Referenzspannung für die Spannungsmessung,
- die Höhe der Entladeschlussspannung,
- der Lastwiderstand.

Die Werte sind im Eeprom des ATtiny gespeichert und werden bei einem Neustart übernommen.

Weiterhin kann der Slave genutzt werden, um den Discharger aus einer Benutzeroberfläche heraus zu steuern. Denn er kann Kommandos empfangen und ausführen zum:

- Starten einer Messreihe (manuell durch Betätigen der Start-Taste möglich),
- Unterbrechen der Messreihe,
- Fortsetzen einer unterbrochenen Messreihe,
- Beenden einer Messreihe (vor Erreichen der Entladeschlussspannung).

Nachdem die Entladung abgeschlossen ist (unteres Spannungslimit erreicht), kann das TWI-Eeprom ausgelesen und die Daten können zur Visualisierung weitergereicht werden.

Funktionsweise

Der Entladevorgang selbst wird auf einer abgesetzten und steckbaren Platine ausgeführt.
Der Akku wird über einen elektronischen Schalter (IRLU024) mit der Last verbunden.

Der Lastwiderstand einschließlich aller Übergangswiderstände und des Spannungsabfalls am FET sollte vor Beginn durch eine Messung bestimmt werden.

Mit Kenntnis dieses Widerstandswertes sowie der anliegenden Spannung kann der fließende Strom berechnet werden.

Die Messungen werden jeweils in Abständen von x Sekunden ausgeführt und die errechneten Stromwerte aufaddiert.

Die entladene Kapazität kann berechnet werden, indem die Summe der Stromwerte durch (3600/x) dividiert wird.

Der Discharger benötigt eine Zeitbasis und einen ADC zur Spannungsmessung.
Die Zeitbasis liefert der 16-Bit Timer1, der jeweils nach Ablauf einer Sekunde ein Flag setzt.

Zur Messung der Akkuspannung durch den ADC ist eine bekannte Referenzspannung erforderlich.
Das kann sein:

- die internen 1.1V Referenzspannung,
- eine externen Spannungsreferenz, z.B. VCC oder eine Konstantspannungsquelle.

Die Verwendung der interne Referenz reduziert zwar den erforderlichen Bauteilaufwand.
Es wird allerdings eine Kalibrierung erforderlich - und da die Auflösung dieser Spannungsmessung gering ist, entsteht ein unschöner Treppeneffekt im Verlauf der Spannungskurve.
(Bei der Messung von Spannung bis 5V beträgt die Auflösung bis zu 20mV, bei Spannungen von 14V treten Sprünge von ca. 50mV zwischen benachbarten Messwerten auf.)

Daher setze ich eine externe Referenzspannung ein.
Um von Spannungsschwankungen unterschiedlicher Netzteile/Batterien unabhängig zu sein, verwende ich eine Referenzspannung unterhalb von VCC, aber oberhalb der LiPo-Spannung von 4.2 V. Die Brownout-Detection ist auf 4.3V eingestellt.

Wie bereits erwähnt ist der Entladekreis zusammen mit dem schaltenden FET ist auf einer steckbaren Platine platziert.
So kann mit geringem Aufwand für verschiedene Entladeszenarien (Akku-Spannungen und Belastungen) passende Hardware vorbereitet werden.

Die Software auf dem ATtiny wartet nach dem Start auf ein Kommando zur Ausführung der Messung.
Dabei leuchtet die gelbe LED mit Dauerlicht.
Ein Druck auf die Start-Taste oder ein USI-Kommando startet den Entladevorgang.

Die grüne LED signalisiert nun, dass der FET durchgeschaltet ist.
Die langsam blinkende gelbe LED zeigt an, dass das Programm ordentlich in seiner Endlosschleife arbeitet.

Messungen werden im Abstand von 6 Sekunden ausgeführt, anschließend wird der Messwert mit der definierten Entladeschlussspannung verglichen.

Nach jeweils 5 Messungen wird ein Datensatz mit den aktuellen Werten auf das Serielle Eeprom geschrieben.
Jeder Datensatz enthält 8 Byte, ein Eeprom mit 64KB kann somit 8.192 Datensätze aufnehmen (das entspricht 68.26h = 2.8 Tage Aufzeichnungsdauer).

Nach Erreichen der Entladeschlussspannung wird der FET und die grüne LED abgeschaltet, die gelbe LED zeigt mit Dauerlicht an, dass die nächste Messreihe gestartet werden kann.

Vorher sollten aber die Messwerte ausgelesen und gesichert werden, denn neue Daten werden immer beginnend mit Adresse 0, 0 des Eeproms geschrieben.

Der Betriebszustand des Dischargers wird durch 3 LED's signalisiert.

```
Grüne LED  FET durchgeschaltet, die Entladung läuft
Gelbe LED  vor Entladebeginn: Dauerlicht,
           während der Entladung: Blinken im Sekundentakt
Rote LED   vor Entladebeginn: Daten im Tiny-Eeprom wurden manuell geändert
           während der Entladung: TWI-Fehler bei einem TWI-Master-Write
Gelb+Rot   Dauerlicht: die Entladung wurde via TWI-Kommando unterbrochen
```

Der Entladevorgang wird durch den Start-Taster gestartet (oder durch ein TWI-Kommando).

Folgende Einstellungen können vor dem Start einer Messreihe (Dauerlicht der gelben LED) verändert werden:

```
TWI_ADR, 251, alte TWI-Adresse, neue TWI-Adresse
TWI_ADR, 252, interne Spannungsreferenz (high-Byte), (low-Byte)
TWI_ADR, 253, Entladeschlussspannung (high-Byte), (low-Byte)
TWI_ADR, 254, Entlade-Widerstand (high Byte), (low-Byte)
```

Der TWI-Slave erwartet hier genau 4 Byte, jeweils beginnend mit der aktuellen TWI-Adresse.
Angenommen, die TWI-Adresse des ATtiny sei "8":

```
8, 251, 8, 10 - Ändert die TWI-Adresse von 8 auf 10
8, 252, 9, 96 - Definiert 2400mV als Referenzspannung
8, 253, 9, 46 - Definiert 2350mV als Ladeschlussspannung
8, 254, 0, 95 - Definiert 9.5 Ohm (Faktor 10 !) als Last-Widerstandswert auf
```

Anmerkung:
Der Wert für den Lastwiderstand wird z.Z. durch die Software nicht mehr genutzt, da die Berechnung und Aufsummierung des fließenden Stromes dem auswertenden Programm überlassen wird.

Immer dann, wenn der ATtiny 4 Bytes empfangen hat, toggelt die rote LED als Hinweis.
Änderungen werden aber zunächst nur ins Eeprom geschrieben.
Erst bei einem Reset werden die neuen Werte übernommen.

Der Inhalt der ersten 8 Byte des Eeproms kann durch Auslesen des USI-Slaves kontrolliert werden (siehe weiter unten).

Folgende Aktionen können via USI-Slave ausgelöst werden:

Vor Beginn einer Messreihe (Dauerlicht der gelben LED):

```
8, 100, 200, x Startet eine Messreihe (x = beliebiges Zeichen)
```

Während einer Messreihe (Dauerlicht der grünen LED, gelbe LED blinkend),
der TWI-Slave erwartet hier genau 2 Byte:

```
8, 201 - Unterbricht eine Messreihe
       - (grüne LED aus, gelbe/rote LED Dauerlicht)
8, 202 - Setzt eine unterbrochene Messreihe fort
8, 203 - Beendet eine Messreihe (nachdem der nächste Datensatz geschrieben wird)
```

Der USI-Slave kann ausgelesen werden und liefert folgende Daten:

Nach dem Programmstart (gelbe LED Dauerlicht) wird der Inhalt der ersten 8 Bytes des ATtiny-Eeproms zum Auslesen bereitgestellt.
Hier kann kontrolliert werden, ob die Einstellung für die TWI-Adresse, die Referenzspannung, die Entladeschlussspannung und den Lastwiderstand gegenüber den Voreinstellungen im Programmcode geändert wurden.
Die Reihenfolge der Daten:

```
255, TWI-Adr, Uref_high, Uref_low, Uabschalt_high, Uabschalt_low, R_high, R_low
```

Solange der Anwender Vorgabewerte nicht verändert hat, steht im Eeprom:

```
255, 255, 255, 255, 255, 255, 255, 255
```

Wurde die TWI-Adresse auf '100' geändert, dann sieht die Ausgabe so aus:

```
255, 100, 255, 255, 255, 255, 255, 255
```

Wurde zusätzlich ein Lastwiderstand von 10.1 Ohm definiert, dann sehen wir:

```
255, 100, 255, 255, 255, 255, 0, 101
```

Wurde dann noch eine Ladeschlussspannung von 2500mV definiert, dann wird angezeigt:

```
255, 100, 255, 255, 9, 196, 0, 101
```

Nachdem die Messreihe gestartet wurde (grüne LED Dauerlicht, gelbe LED blinkend), können aus dem TWI-Slave die Messwerte ausgelesen werden, die jeweils im 6 Sekunden-Intervall aktualisiert werden. Die Reihenfolge der Daten:

```
Zeit_high, Zeit_low, Kapazität_high, Kapazität_low, Spannung_high, Spannung_low
```

Als Zeitinformation wird die Anzahl der bereits ausgeführten Messungen ausgegeben.
Bei einem Messintervall von 6 Sekunden muss dieser Wert mit 6 multipliziert werden, um die Dauer der Entladung in Sekunden wiederzugeben.

Jeweils nach 5 Messungen (somit nach 30 Sekunden) wechselt der ATtiny kurz in den USI-Master-Modus und schreibt den letzten Messwert auf das TWI-Eeprom.

Es werden immer Datensätze der Länge 8 Byte in nachfolgendem Format geschrieben:

```
ID, Zeit_high, Zeit_low, C_high, C_low, U_high, U_low, 255
```

Die Werte für die Kapazität C werden z.Z. nicht berechnet, hier steht immer 0, 0.
Aus der Kenntnis der Spannung, des Lastwiderstandes sowie der Dauer des Messintervalls kann eine auswertenden Software den fließenden Strom sowie die Entladekapazität berechnen.

Das erste Byte eines Datensatzes ist eine ID, die für alle Datensätze einer Messreihe identisch ist. Der Wert wird vor dem Start einer neuen Messreihe incrementiert.

Der abschließende Datensatz einer Messreihe hat folgendes Format:

```
ID, Uref_high, Uref_low, Uabschalt_high, Uabschalt_low, R_high, R_low, 0
```

Das letzte Byte '0' unterscheidet diesen Datensatz von allen vorangegangenen Datensätzen mit Messwerten, deren letztes Byte die Konstante '255' enthält.

Noch einmal zusammengefasst:
In alle Datensätzen einer Messreihe steht im ersten Byte der gleiche Wert.
Alle Datensätze mit Messwerten enden mit '255' als 8. Byte.
Der letzte Datensatz und damit der Abschluss der Datenreihe ist an der '0' im 8. Byte erkennbar.

Anmerkung zum USI-Slave / USI-Master:
Der USI-Master auf dem ATtiny beherrscht keinen Multi-Master-Mode. Daher kann es es zu Kollisionen kommen, wenn auf den Slave zugegriffen wird, während der Master gerade schreibt.

Controller

Als Controller hatte ich einen ATtiny24 mit 2k Flash gewählt.
Anstelle eines externen Quarzes (8MHz) kann der interne RC-Oszillator verwendet werden, die Anforderungen an die Genauigkeit der Zeitbasis müssen dann aber relativiert werden.

Die externe Referenzspannung liefert ein TL431 (der wartete zufälligerweise in der Bastelkiste auf seinen Einsatz).

Der Programmcode wollte während der Entwicklung häufiger die engen Grenzen des ATtiny sprengen. Um etwas Platz zu gewinnen, habe ich die ISR-Routinen alternativ in Assembler geschrieben. Und auch die speicherfressenden 32-Bit Berechnungen für die Entladekapazität musste ich häufig auskommentieren.

Nachdem ich mich entschlossen hatte, nicht nur 4.2V LiPo's sondern auch Blei-Gel-Akkus und Akkupacks zu vermessen, schien es mir nicht mehr sinnvoll zu sein, die Kapazitätsberechnung auf dem Controller auszuführen. Zu viele Parameter hätten jeweils angepasst werden müssen.

Der Controller protokolliert nun lediglich noch die am Spannungsteiler gemessene Spannung über die Zeit.
Der auswertende PC kann dann aus dem bekannten Lastwiderstand und dem Faktor, der sich aus der Spannungsteilung ergibt, die Akkuspannung und den fließenden Strom berechnen und über die Zeit aufsummieren.

Zum Auslesen des Eeprom verwende ich meinen "Raspmega" in Verbindung mit einem Python-Script. Damit wird eine Logdatei erzeugt, die von gnuplot in eine Graphik umgesetzt wird.

Fazit

Das Ziel der Übung war eigentlich gewesen, die Kapazität / Qualität der neu gekauften LiPo's zu beurteilen:

Von den versprochen 2200mAh konnte ich bis zu 100% bei einer Entladeschlussspannung von 3.2V wiederfinden.

Ich denke, das Ergebnis sieht doch ganz positiv aus - oder ?
Zumindest die vermessenen Blei-Gel-Akkus zeigen eine wesentlich ungünstigere Ausbeute.

In der Anlage "Entladekurven.pdf" habe ich einige Diagramme ausgedruckt.

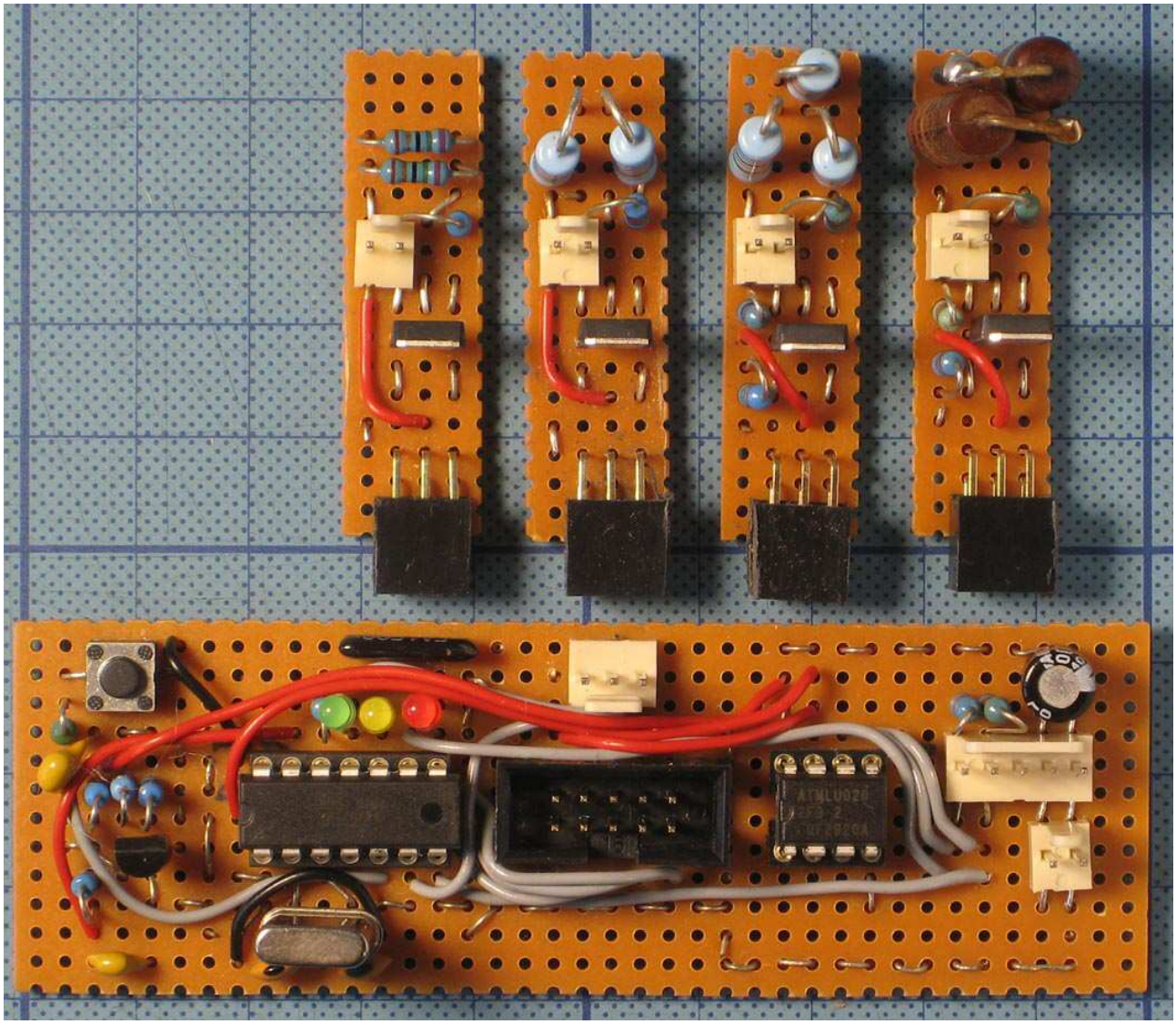
Abb. 1 - zeigt die Spannungsverläufe eines LiFePO4 sowie von 2 baugleichen LiPo's
Abb. 2 - zeigt den Spannungsverlauf eines 3C-LiPos
Abb. 3 - zeigt den Spannungsverlauf an einem 6V Blei-Gel-Akku mit 4.7Ah
Abb. 4 - zeigt den Spannungsverlauf an einem 12V Blei-Gel-Akku mit 7.2Ah.

Viel Spaß beim Messen,

Michael S.

04.01.2014

Hier noch ein Foto meines Dischargers:



Unten die "Hauptplatine" mit Start-Taster und Spannungsreferenz auf der linken Seite, ATtiny24 mit Programmierstecker in der Mitte und Seriellem Eeprom sowie TWI-Bus-Stecker auf der rechten Seite.

Oben vier steckbare Platinen mit unterschiedlichen Lastwiderständen und Spannungsteilern auf den beiden rechten Platinen.