Ergänzung zur Bedienung des AVR-Transistortester

K.-H. Kübbeler

Gilt nur für Testversion 0.9k

Einleitung:

Basierend auf der Software von Markus Frejek habe ich angefangen die Software zu modifizieren. Der Grund war zu Beginn eigentlich nur ein Problem, das ich mit dem Programmieren des EEprom hatte. Da sich der Flashspeicher aber problemlos programmieren ließ, dachte ich mir, das ich am schnellsten und günstigsten zu einer lauffähigen Version komme, wenn ich die Texte und Parameter einfach aus dem EEprom in den Flashspeicher verlagere.In diesem Zusammenhang habe ich mir die Software genauer angesehen und hatte dabei erst eine Idee, die ich dann auch testweise umgesetzt habe:
Für Ausgaben auf dem Display wird die Spannung in mV gebraucht und nicht die Schritte des AD-Wandlers. Das muss in der Originalsoftware jedes Mal umgerechnet werden. Die original AD-Wandler Routine (ReadADC) liest den Wandler zwanzig Mal, addiert die Ergebnisse und dividiert danach durch 20. Damit ist die Auflösung wieder 5000mV/1023 . Wenn ich nun den AD-Wandler statt 20 Mal 22 Mal lesen lasse, das Ergebnis nochmal verdopple und danach durch 9 dividiere ist mein Maximalwert 22*2*1023/9=5001 was prima zu der gewünschten mV Anzeige passt.
Mit dieser Idee fing dann richtige Arbeit an. Jetzt mußte ich alle Spannungsabfragen im Programm auf die neue Auflösung anpassen. So lernte ich das Programm von Markus Frejek immer besser kennen. Meine aktuelle Version von ReadADC liest 45 Mal, wobei der jeweils erste Messwert vergessen wird , die nächsten 44 Messungen werden addiert und durch 9 geteilt. Da ich anfangs auch noch den Ehrgeiz hatte, den Code der abgespeckten Version in 4K Programmspeicher zu bekommen, habe ich sicher auch Änderungen vorgenommen die bei richtiger Betrachtung nicht nötig waren. So habe ich beispielsweise die Warteschleifen durch Aufrufe einer selbst entwickelten Assembler-Warteschleifenroutine ersetzt, die selbst nur 66 Byte lang ist, jeder Aufruf kostet aber nur 1 Wort Befehl (2 Byte) und es wird ein Bereich von 10µs bis 5s im Raster
1 2 3 5 10 …. zur Verfügung gestellt. Die Routine beinhaltet ab 100ms schon den WatchDog Reset. Zwischenwerte wie z.B. 8ms Wartezeit kosten 2 Aufrufe (bis auf 9ms , hier würde ich 3 Aufrufe brauchen). Mir ist keine Technik bekannt, die bei vielen Warteaufrufen wirtschaftlicher wäre, wobei meine Technik Taktgenau arbeitet, wenn die unterste Routine (10µs) das tut. Das werde ich überprüfen, wenn ich eine Testversion für Quarz-Betrieb fertig habe. Lediglich durch den zusätzlichen WatchDog Reset ist die 100ms um 1µs zu lang. Die Aufrufe der Routinen benötigen keine Register mit Ausnahme des Stack-Pointers für die Verwaltung der Rückkehradressen im RAM. Das waren in meiner ersten Version (10µs bis 1000ms) mal 22Byte Platz, jetzt sind es ein paar Byte mehr. So konnte ich wertvollen Speicherplatz einsparen. Das Programm für die 8MHz Variante wird dadurch kürzer als die 1MHz Variante, da ich dann auch in den LCD-Routinen mit dieser Wartetechnik arbeite (es stehen dann auch Aufrufe für 1µs,2µs,3µs,5µs zur Verfügung).

Neue Features und Änderungen:

- Die Widerstandsmessung wurde so erweitert, das auch Potis oder Trimmer angeschlossen werden können. Die Widerstandsbelegung wird in Zeile 1 des LCD-Displays dann im Format 1:2:3 angezeigt, wobei in der zweiten Zeile dann der Widerstand zwischen Pin 1 und Pin 2 und dahinter der zwischen Pin 2 und 3 angezeigt. Wenn man zu faul ist den Gesamtwiderstandswert auszurechnen, kann man den mittleren Pin entfernen und die Messung neu starten. Wenn der Schleifer des Potis an einem Ende steht, kann der Transistortester den mittleren von dem Endpin nicht mehr unterscheiden!

- Die Messung von Kondensatoren wird durch eine Ladezeitmessung ermittelt. In der Originalsoftware ist das eine Programmschleife, die den jeweiligen Input-Pin abfragt und mitzählt. Nachteil des Softwarezählers ist, dass die Schleifenzeit (Anzahl der Befehle für einen Durchlauf) selbstverständlich in die Berechnung einfließt. Ich habe diese Lösung durch eine technische Möglichkeit des AVR's ersetzt, das ein Zähler, der mit der vollen Taktfrequenz (also 1MHz bzw. auch 8MHz) laufen kann, seinen Zählerstand bei einem äußeren Ereignis zwischenspeichern kann. Dieses äußere Ereignis kann auch der Komparator-Ausgang des AVR's sein. So entlade ich den Kondensator, starte den Zähler bei 0 und starte die Ladung des Kondensators mit dem 680Ω oder 470kΩ Widerstand. Jetzt frage ich in der Schleife ab, ob entweder ein Zählerübertrag oder ein Zwischenspeicher Ereignis eingetreten ist. Die Übertrags-Ereignisse zähle ich brav mit und beim Eintreten des Zwischenspeicher-Ereignisses stoppe ich den Zähler und prüfe, ob ich noch einen Übertrag zusätzlich machen muss, denn leider lässt sich der Zähler nicht automatisch durch das Zwischenspeicher-Ereignis stoppen. Von der so ermittelten Gesamtzeit ziehe ich bei der Messung von kleinen Kondensatoren eine experimentell bestimmte Konstante ab. Inwieweit diese Konstante von Exemplar zu Exemplar angepasst werden muss, kann ich nicht sagen. Die niedrige Vergleichsspannung führt aber zu einer Verkürzung der Messzeit.
- Leider litt die Messgenauigkeit der Kapazitätsmessung unter den Exemplar-Streuungen der AVR's, wie ich aus der bisher einzigen Rückmeldung wusste. Als Grund vermute ich die große Exemplar-Streuung der internen 1,3V Referenz, die ich bei der Kapazitätsmessung als Vergleichswert benutze. Deshalb besitzt die aktuelle Software intern eine Tabelle im 50mV Abstand, in der die theoretische Abhängigkeit der Ladezeit von der Vergleichs-Spannung abgelegt ist. Jetzt wird vor der Messung die Referenzspannung im Verhältnis zu der 5V Betriebsspannung gemessen und entsprechende Umrechnungsfaktoren für die Kapazitätsmessung für beide Widerstandswerte linear interpoliert. Diese Funktion kostet natürlich wertvollen Programmspeicherplatz. Ob damit die Exemplar-Streuungen kompensiert werden können muss noch getestet werden. Alternativ wäre auch eine Lösung denkbar, bei der die Spannung nur einmal vorher gemessen wird und dann die berechneten Faktoren in der individuellen Software vom Compiler eingetragen werden. Das würde aber bedeuten, dass jeder Nutzer seinen Tester kalibrieren muss, wenn er genauere Messergebnisse wünscht. Außerdem ist mir nicht bekannt, wie sich die Referenzspannung mit der Betriebstemperatur ändern. Jetzt wird bei jedem Einschalten neu gemessen.
- Die Entladefunktion von Kondensatoren wurde insofern verändert, das der Kondensator bei Unterschreitung von 1300mV über die Ausgabe Ports der ADC Pinne kurzgeschlossen wird. Das halte ich für vertretbar, da der Innenwiderstand von jedem Port Ausgang etwa 20Ω beträgt. Auch die im Datenblatt in Figure 149 (Seite 258) gezeigten Kurven gehen bis 2V. Garantieren kann ich das natürlich nicht. Ich habe mit meinem Atmega8L die Funktion mit 15mF Kondensatoren ausprobiert, ohne das ich einen Schaden bemerkt hätte. Der Strom sollte nach Rechnung auch unter den spezifizierten 40mA bleiben. Schaden kann der AVR natürlich nehmen, wenn HV Kondensatoren vor der Messung nicht entladen wurden.
- Das Format der Pin-Angabe bei Dioden (A: K:) habe ich durch eine Symboldarstellung ersetzt, die sprachunabhängig jedem Bastler bekannt sein sollte. Leider lässt das Display keine vernünftige Zehnerdioden oder Doppeldiodendarstellung zu, weshalb die Darstellung mit Einzeldioden in Flussrichtung dargestellt wird. Auf die Pin-Angaben achten! Wenn bei zwei Dioden die äußeren Pinnummern gleich sind, kann es sich um Zehnerdioden oder Doppel-LEDs oder ähnliches handeln. Hierbei auf die Flussspannungen achten.

- Die Diodenmessung von Einzeldioden habe ich mit einer Kapazitätsmessung in Sperrrichtung ergänzt. Damit kann man vielleicht die Eignung für bestimmte Zwecke abschätzen. Hier habe ich Werte von wenigen pF bis zu 5nF (Basis Emitter Strecke eines BU508A). Bei einer Parallelschaltung aus Kondensator und Diode kommt es zu einer Verlängerung der Messzeit, die Kombination wird aber bei 2µF und normaler Flussspannung der Diode noch erkannt. Bei größeren Kapazitätswerten wird nur der Kondensator erkannt. Selbstverständlich kann man die Messung auch bei den Transistor-Dioden durchführen, wenn man nur 2 Klemmen also Basis und entweder Kollektor oder Emitter anschließt.
- Die Darstellung der Pinnummern bei bipolaren Transistoren habe ich aus programm-ökonomischen Gründen durch eine EBC=123 Darstellung ersetzt.
- Die Transistormessung wurde durch eine Messung in Kollektorschaltung (Emitterfolger) ergänzt. Bei Leistungstransistoren wird so ein vernünftigerer hFE bestimmt. Bei dieser Messmethode besteht keine Gefahr der Sättigung der Basis, auch wenn die Basis mit dem 680Ω Widerstand gespeist wird. Die übliche Messmethode bei höheren Verstärkungsfaktoren ist aber nach wie vor die Emitterschaltung.

Die Optimierungen bezüglich der Messgenauigkeit sind nach meiner Meinung noch nicht abgeschlossen. Hier wäre ein wenig mehr Informationsrückfluss nützlich.
Ich hoffe, dass ich keinen wichtigen Punkt vergessen habe, und komme damit zum letzten Punkt, dem ich ein eigenes Kapitel widmen möchte.

Selbsttest-Funktion

Bei der neuesten Software 0.9k habe ich eine Selbsttestfunktion eingebaut. Die Bedienung dazu ist sehr einfach. Die meisten werden nach meiner Meinung Prüfschnüre mit Krokodil-Klemmen zum Anschluss von Prüflingen benutzen. Um den Tester in die Selbsttestfunktion zu bringen, werden einfach alle drei Prüfklemmen auf ein Stück blanken Draht (eine Seite eines Widerstandes z.B.) geklemmt und dann der Startknopf gedrückt. Schon startet das Selbsttest-Programm. Nach Ablauf des Programms macht der Transistortester mit dem normalen Messprogramm weiter, endet also in der Regel mit „unbekanntes Bauteil“. Die traurige Seite dieser Selbsttestfunktion ist, dass auch bei der 8K Version des AVRmega der Speicher zu Ende geht. Die Länge des Programms beträgt jetzt mit der Selbsttestfunktion schon 8000 Byte in der Atmega8 Version. Wenn ich das Programm für einen Atmega88 kompiliere, liegt es mit 8122 Byte schon hart am Limit. Und ich sehe nicht mehr sehr viel Luft zum sparen. Ich habe ja schon einige Routinen wie ReadADC und Hilfsfunktionen in Assembler programmiert.
Beim Ablauf der Selbsttest-Schritte wird in der 1. Zeile immer ein T mit fortlaufender Nummer angezeigt, jeder einzelne Test wird 8 Mal wiederholt und dann zum nächsten Schritt oder Ende weitergegangen. Bei den Tests werden nur Messergebnisse dargestellt. Es werden keine Fehleranalysen durchgeführt. Interpretieren muss man die Messergebnisse selbst. Noch ein wichtiger Hinweis für diejenigen, die auf dem Transistortester einen ISP-Stecker verbaut haben. Generell darf bei den Messungen kein Programmiergerät angeschlossen sein!

Doch nun zu den einzelnen **Test**s:

1. Messung der 1.3V Referenzspannung (Bandgap Reference). Dargestellt wird in Zeile 1 der Text Ref= und die gemessene Spannung in mV, in der zweiten Zeile werden die daraus berechneten Faktoren für die Kapazitätsmessungen dargestellt.

2. Vergleich der 680Ω Widerstände.
   Dargestellt wird in Zeile 1 der kryptische Text „L1+2- .3- 2+.“. Das soll Folgendes bedeuten: Das L steht für Low also die 680Ω Widerstände. 1+ steht dafür das der Widerstand von Pin1 nach + (VCC) geschaltet wird. Das nachfolgende 2- steht für die Schaltung des Pin2 Widerstandes nach -. Das Ergebnis der Spannungsmessung steht in der 2.Zeile an der ersten Stelle. Es folgt eine „ .3-“ was heißen soll dass beim Zweiten Messwert der Pin1 Widerstand beibehalten wird, aber anstelle des Pin2 Widerstandes jetzt der Pin3 Widerstand nach Masse geschaltet wird. Die letzte Messung wird entsprechend mit Pin2 Widerstand nach + und dem Pin3 Widerstand nach Masse gemessen. In allen Fällen sollten dabei die halbe Betriebsspannung also 2500 gemessen werden. Bitte beachten Sie, dass die Auflösung des ADC nur knapp 5mV beträgt, also nicht zu kritisch sein.
   Meine Ergebnisse sind hier 2493 2493 2488 , obwohl ich 0,1% Präzisionswiderstände verbaut habe.
3. Vergleich der 470kΩ Widerstände.
   In Zeile 1 wird dargestellt „H1+2- .3- 2+“. Hier wird also die gleiche Messung wie unter dem zweiten Test beschrieben mit den Hochohmigen Widerständen wiederholt.
   Meine Ergebnisse sind hier: 2493 2498 2498
4. Hier wird gar nicht gemessen, es erfolgt nur der Hinweis, das es jetzt Zeit ist die zusammengeschalteten Mess-Klemmen zu trennen (Text: isolate Probe).
   Die Werte in der 2. Zeile sind ohne Bedeutung, es wird nur nicht gelöscht.
5. Es wird geprüft, ob ohne angeschlossenes Bauteil mit dem hochohmigen Widerstand nach – (Masse) geschaltet werden kann. Anzeige in Zeile 1 ist RH- .
   Hier sollte für alle drei Pins eine 0 in Zeile 2 erscheinen.
6. Es wird geprüft, ob ohne angeschlossenes Bauteil mit dem hochohmigen Widerstand nach + geschaltet werden kann. Anzeige in Zeile1 ist RH+ .
   Hier sind für alle drei Pins 5001 der Idealwert. Abweichungen vom Idealwert bei Test 5 und Test 6 deuten auf einen Fehler oder ein Isolationsproblem hin (Lötmittelreste oder ähnlich).
7. Innenwiderstandmessung der Pin-Anschlüsse Richtung Masse.
   Text in der 1. LCD-Zeile ist Ri_Lo = (mV). In der zweiten Zeile erscheinen 3 Spannungen. Die AVR Ausgänge sind natürlich keine idealen Schalter, Die Anschlüsse haben einen Innenwiderstand, der hier versucht wird, mit dem Stromfluß des 680Ω Widerstand gegen + zu bestimmen. Aber allzu große Hoffnung, dass man hiermit die Messgenauigkeit durch kalibrieren weiter steigern kann muss ich dämpfen, es werden nur die 3 Pinne (PC1-3) der ADC-Eingänge (PC0 bis PC2) untersucht. Für einen vollständige Abgleich der individuellen Parameter wäre auch die Messung der 680Ω Ausgänge erforderlich (Pin PB0,PB2 und PB4), was nicht ohne Eingriff möglich ist. Um auf die Ohm-Werte zu kommen müssen die Spannungswerte noch durch ca. 7 geteilt werden.
   Bei mir sind die Ergebnisse: 127 132 135
8. Innenwiderstandsmessung der ADC Pin-Anschlüsse Richtung VCC (+).
   Hier wird der Stromfluss mit den 680Ω gegen Masse erzeugt.
   Test in der 1. LCD-Zeile ist Ri_Hi= (mV). In der zweiten Zeile erscheinen 3 Spannungen, die bis zum Idealwert 5001 fehlen. Also die gleiche Messung wie bei Test 7 zur anderen Seite.
   Meine Ergebnisse sind: 150 151 151

Zum Schluss wird noch der Text „Auto Test End“ in Zeile 1 und die Versionsnummer der Software in Zeile 2 ausgegeben und das Programm fährt mit einer normalen Bauelemente-Messung fort.