Pollin „ATMEL Evaluations-Board Version 2.0.1“

Erfahrungsbericht von Jens Sieler-Hornke, Mai 2011

Mir liegt zu diesem Zeitpunkt die Version 2.0.1 des Atmel Evaluations-Board's vor mit gleichnamiger Anleitung in der 6. Auflage.

Die Anleitung

Die Anleitung zum Pollin Atmel Evaluations-Board ist ausführlich und übersichtlich gestaltet (wenn man vom kleinen DIN A5-Format absieht). Mit Tips und Tricks gibt Pollin auch einem Anfänger guten Rat bei der Montage des Bausatzes.
Im Anschluß der Anleitung befinden sich Tabellen die die Bezüge zwischen den Tastern, den LEDs, dem „Buzzer“ und dem Anschluß des 40-poligen Expansions-Port's im Zusamenhang mit den Anschlüssen der zu verwendenden Microcontroller-Typen darstellen.
Die Platine ist entsprechend der Bestückung beschriftet. Die Bauteile sind darauf lediglich entsprechend des Schaltplans nummeriert. Anhand der Bestückungsliste lassen sich die Werte der entsprechenden Bauteile entnehmen.
Ich habe den Schaltplan größer kopiert und darin die Werte der Bauteile eingetragen um eine zügigere Montage durchzuführen. In meinem Bausatz waren alle Bauteile vollzählig, lediglich die Stiftleiste für die Jumper P9-12 war zu kurz und hätten anstatt 3x2reihig 4x2reihig sein müssen. Dementsprechend fehlte auch der Jumper. Ich hatte noch eine längere Leiste und habe mir durch Kürzung eine passende Stiftleiste daraus kreiert. Schnell waren die Jumper von der zu kurzen auf die neue Stiftleiste gesetzt und der Fehlende durch einen Vorhandenen ergänzt.

Der Aufbau

Die Anleitung weist auf die Bestückungs-Reihenfolge in Abhängigkeit der Höhe der Bauteile hin. Hält man sich daran läßt sich der Bausatz bequem aufbauen und einfach löten. Ich habe dazu eine alte Weller Lötstation mit einer 1mm breiten Spitze verwendet.
Oft liegen einige Durchkontaktierungen sehr dicht neben einer Lötstelle, daher ist hier mit etwas Fingerspitzengefühl zu arbeiten um beim Löten keinen Kurzschluß zu verursachen.
Die Taster ließen sich etwas widerspenstig in die Platine einfügen. Auch hier ist etwas Geschick gefordert. Nach ca. 2 gemütlichen Stunden war das Board fertig aufgebaut. Anschließend erfolgte eine optische Kontrolle meinerseits.

Die Versorgungsspannung (J5)

In einigen MicroController-Foren wird die Frage gestellt wie denn die Platine mit dem Computer zu verbinden sei und welche Art der Versorgungsspannung zu wählen ist. Dies geht aus meiner Sicht *(bei genauem Hinsehen)* eindeutig aus der Anleitung hervor: Im Schaltplan befindet sich ganz unten links das „Netzteil“. Links vor dem Gleichrichter liegt der Anschluß **J5** der mit **9V~** beschriftet ist. An die Anschlußklemme J5 ist daher eine Wechselspannung von 9V anzulegen.
In einigen Foren ist zu lesen daß einige Anwender auch eine Gleichspannung von 9V dort angelegt haben. Dies ist durchaus möglich. Die Polarität spielt dabei keine Rolle da der Gleichrichter am Spannungseingang in diesem Fall als Verpolungsschutz arbeitet und damit den benötigten Stromfluß vorgibt.

Der Programmier-Anschluß (J3)

Die Anleitung weist darauf hin daß „über die auf dem Board befindliche ISP-Schnittstelle per RS232-Schnittstelle“ programmiert werden kann. Dies bedeutet daß zum flashen des Microcontroller's der SubD-Anschluß J3 (mit der Beschriftung ISP) zu verwenden ist. Hier ist einfach nur ein 1:1 Verlängerungskabel (Kupplung/Stecker) zum Anschluß an die serielle Schnittstelle des Computers zu verwenden! - Kein Null-Modemkabel oder Ähnliches!

Der RS232-Kommunikations-Anschluß (J6)

Der Sub-D-Anschluß J6 mit der der Beschriftung RS232 dient zur Kommunikation eines Microcontrollers mit der „Außenwelt“ und nicht zum Programmieren des Chips!
Dieser Anschluß ist ebenfalls mittels eines 1:1 Verlängerungskabels mit dem Computer zu verbinden!

Die Programmierung

Zur Programmierung (zum Flashen) eines Microcontroller's ist das Board über seinen Anschluß J3 zur seriellen Schnittstelle eines PC's herzustellen. Ebenfalls ist die Versorgungsspannung zum Board anzuschließen.
Ich verwende ein Netzteil das sekundärseitig 9V~/500mA abgibt. Eine enorme Wärmeentwicklung am Spannungsregler 7805 war bisher nicht festzustellen, entgegen in anderen Foren beschriebenen Beobachtungen. Beim Anlegenen z.B. von 12V erwärmt sich der 7805 natürlich mehr.

Die Anleitung zum Pollin „Testtool“ des Evaluations-Board's empfiehlt zur Übertragung eines Kompilats (.HEX-File) zum Microcontroller das Programm PonyProg2000, das kostenlos im Internet zum Download bereitsteht.

1. Praxis-Test

Ich habe zum ersten Test ein simples Assembler-Programm unter AVR-Studio 4.13 Build 528 geschrieben und dessen Komplilat per Ponyprog2000 in einen ATtiny2313 „ge-flasht“. Der Funktions-Test verlief einwandfrei.

Das Pollin-TESTTOOL

Pollin stellt zum Atmel Evaluations-Board auf der Internet-Site des Produkts eine einfache Test-Routine („D810038S.ZIP“) zum Download bereit. Nach Entpacken des ZIP-File's entstehen die Dateien Testtool.bas (der BASIC-Quell-Code), TESTTOOL.HEX (das Kompilat das in den Microcontroller „ge-flasht“ wird) und die Anleitung Testtool.pdf (Die Anleitung beschreibt die Einstellung der seriellen Schnittstelle des PC's zwecks Kommunikation mit dem Board und der Anleitung zum Daten-Transfer des HEX-Files in den Microcontroller mit PonyProg2000).

Ich habe in einigen Foren gelesen daß man sich nach dem „Brennen“ des Pollin-Testtool-HexFiles in einen ATmega8 wunderte warum der Test erfolglos verlief. Dies kann folgende Ursachen haben:
Im Normalfall wird ein Programm explizit nur für einen bestimmten Microcontroller geschrieben. Explizit ist das Kompilat dazu nur für diesen Typ lauffähig für den es entwickelt wurde! In der Anleitung zum Pollin-Testtool wird ausdrücklich darauf hingewiesen daß das HEX-File für einen ATmega16 geschrieben wurde und wie es in diesen zu übertragen ist, inklusive Einstellungen der Fuse-Bits!

Nachdem das Pollin-Testtool in einen Atmega16 ge-flasht wurde sollen mit dem Pollin Atmel-Evaluations-Board folgende Funktionen ausgeführt werden:

1. Durch Drücken des Taster1 wird die LED1 ein geschaltet.
2. Durch Drücken des Taster2 wird die LED2 ein geschaltet.
3. Durch Drücken des Taster3 wird der Buzzer für 0.5 sec aktiviert, anschließend wird der Text „Atmel-Evaluation-Board“ über die serielle Schnittstelle ausgegeben.
   Der über die serielle Schnittstelle ausgegebene Text, kann unter Windows mit dem Hyperterminal ausgelesen werden. Dabei ist vor dem Start von Hyperterminal die serielle Schnittstelle des PC's durch ein Verlängerungskabel! mit dem Anschluß J6 (RS232) des Boards zu verbinden.

2. Praxis-Test

Mein erster Test mit dem Pollin-TESTTOOL verlief im Frust: Das File TESTTOOL.HEX habe ich mit PonyProg2000 geladen, die Configuration und Security Bits entsprechend der Anleitung zum Testtool gesetzt und die Daten in den ATmega16 ge-flasht.

- Funktion 1 verlief erfolgreich
- Funktion 2 verlief fast erfolgreich nur daß in unregelmäßigen Abständen LED1 und LED2 sowie der „Buzzer“ beim Drücken irgendeiner der beiden Taster beeinflußt wurden. Wobei der „Buzzer“ lediglich nur ein klägliches Knacken und kein Summen von sich gab.
- Funktion 3 ließ zwar den Buzzer knacken aber es erschien kein Text im Hyperterminal trotz korrekter Einstellungen und Anschluß der seriellen Schnittstelle an J6 des Pollin-Board's. Außerdem wurden wieder sporadisch beide LEDs ein- und ausgeschaltet.

Ursache: 2x Fehler im BASIC-Source-Code

Ich überprüfte das Board erneut, fand aber keinen Fehler. Somit entschloß ich mich den Source-Code des Pollin-Testtool's zu untersuchen.

- Mit BASCOM (die im Internet vom Hersteller bereitgestellte kostenlose Testversion reicht dazu aus) habe ich den BASIC-Quell-Code „Testtool.bas“ geladen. Dabei fiel mir auf daß in Zeile 22 die Anweisung `$crystal = 8000000` nicht korrekt ist, denn auf dem Pollin-Evaluations-Board sind die Pins 12 und 13 des IC6 (=Fassung für den ATmega16) mit einem 16MHz- und nicht mit einem 8MHz-Quarz verbunden sind! Der Wert 8000000 ist hier also falsch.
- Dann ist der sogenannte „Buzzer“ kein selbst-tönender Summer sondern zumindest in meinem Board ein Piezo-Schallwandler vom Typ ET121P22. Dieser knackt natürlich nur beim Ein- und Ausschalten. Würde man dies in schnellerer Folge (als wie im Programm mit 500ms) wiederholen entstünde ein Ton. Dieser Effekt muß damit per Software gelöst werden.

Fehlerbeseitigung

Ich entschloß mich den Test gleich mit einem anderen Controller, einem Atmega8, zu wiederholen. Den ATmega16 habe ich der Fassung IC6 entnommen (da jeweils nur EIN Controller auf dem Board zu verwenden ist!) und einen ATmega8 in die Fassung IC4 gesteckt. Auch der ATmega8 wird mit einem 16MHz-Quarz im Pollin-Evaluations-Board getaktet. Anschließend habe ich den BASIC-Code unter BASCOM folgendermaßen den tatsächlichen Bedingungen entsprechend angepaßt:

```
$regfile = "m8def.dat" 'definieren des verwendeten Chips (ATmega8)
$crystal = 16000000 'definieren des verwendeten externen Quarz (16MHz)
```

Damit auch der Piezo-Schallwandler summt muß der Port7 impulsartig getaktet werden. Im Testtool von Pollin wird der Port jedoch nur eingeschaltet, 500ms gewartet und wieder ausgeschaltet. Dies erzeugt jedoch kein Summen sondern lediglich ein schnödes Knacken. Um den Piezo-Schallwandler zum Summen zu bringen ist daher das Programm folgendermaßen an entsprechender Stelle zu ändern:

Das Unterprogramm Beep muß geändert werden und ungefähr folgendermaßen aussehen:

```
Beep:              'Untermenü "Beep"
Dim I As Integer   'Zähler für Impuls-Schleife der Summertaktung

For I = 1 To 100   'Impuls-Schleife Pi x Auge
Toggle PORTD.7     'Status des Pin 7 von Port D wechseln
Waitms 2           '2ms ist „Pi x Auge“ – war vorher Waitms 500
Next               'Ende der Impuls-Schleife
Return             'zurück zum Hauptmenü
```

Anschließend habe ich das Programm neu kompiliert und damit das HEX-File neu erzeugt. Dieses habe ich mit PonyProg2000 unter vorherigem Setzen der Configuration und Security Bits (Fuse-Bits) in den ATmega8 ge-flasht

Erfolgreicher Test

Die LED-Tests 1 und 2 verliefen ordnungsgemäß. Der Buzzer summt kurz vor sich hin und im Hyperterminal erscheint nun der Text „Atmel-Evaluation-Board“. Sicher funktioniert das Programm auch mit einem Atmega16 wenn im Source-Code das Register-File $regfile = "m16def.dat" anpaßt wird.