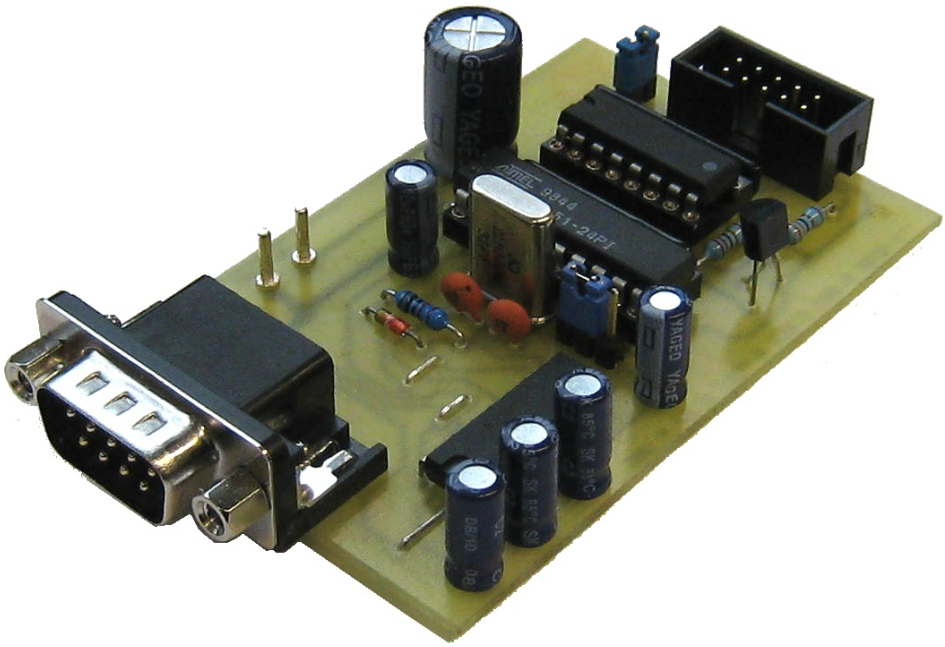
AT89S ISP-Flasher
für Atmel© Microcontroller

Ralph Seelig



Vorwort

Entgegen der Aussagen von Vertretern der AVR-Series Benutzer die MCS-51 Familie sei tot ist dem nicht der Fall. Das dürften auch die Chip-Hersteller so sehen, denn anderst ist es nicht zu erklären weshalb in schöner Regelmäßigkeit neue Bausteine mit immer mehr Features für die MCS-51 Familie erscheinen.

Allen voran ist hier die Fa. Atmel (die auch die AVR-Familie begründet hat) stark vertreten. Die MCS-51 Familie ist seit 1980 eingeführt und kann als sogenannter Industriestandart bezeichnet werden.

Neuen Technologien folgend sind mittlerweile alle Bausteine der MCS-51 Familie „flashbar“, das bedeutet dass sie ihren Codespeicher im Chip mit „an Bord“ haben und dieser Chip mittels elektrischer Spannung auch wieder löschbar und danach neu beschreibbar ist.

Bei den flashbaren Chips kann zwischen parallelen und seriellen Programmiermethoden unterschieden werden. Manche der Chips bieten beide Programmiermethoden an.

Serielle Programmiermethoden können wiederum unterschieden werden in Bootloaderupload und in „Programmieradapterupload“.

Chips mit Bootloader haben parallel zum eigentlich Benutzerprogramm ein Programm (eben den Bootloader) fest installiert. Dieses Programm ist in der Lage, über eine USB,RS-232 oder CAN Schnittstelle neue Codedaten zu empfangen und sich selbst neu zu flashen.

Die Methode über Programmieradapter (im folgenden als ISP-Adapter bezeichnet) soll hier besprochen sein.

Warum nun soll man sich die Mühe machen, Software über einen ISP-Adapter einzuspielen, wenn es Bausteine gibt, die „einfach“ an den PC angeschlossen werden können?

Die Antwort ist einfach:

- die Bausteine gibt es sehr viel preiswerter als die preiswertesten mit Bootloader (der AT89S51 im PLCC-44 Gehäuse kostet bei einem bekannten Distributro 1,25€ - Stand Dezember 2011)
- die kleinsten Bausteine sind kleiner als die mit Bootloader
- es ist keine (manuellle) Steuerung nötig, die den Baustein in den Uploadmodus bringt

Gründe für die Entwicklung des ISP-Adapters

Durchsucht man das Internet nach verfügbaren Adaptern zur Programmierung der AT89S – Serie so wird einem sehr schnell klar, dass es da keine große Auswahl gibt. Atmel selbst bietet einen Adapter an, der an die (wirklich sehr veraltete) parallele Schnittstelle des PC’s angeschlossen wird. Fast alle neueren Computer bieten diese Schnittstelle nicht mehr an, zudem unterstützen neuere Betriebssysteme den Betrieb dieses Adapters nicht mehr.

Verschlimmernd hinzu kommt, dass der Betrieb des Atmel Adapters über eine „USB–Parallel-Brücke“ oft nicht funktioniert.

Im Internet häufig gefundene Adapter sind Ponyprog von Claudio Lanconelli oder AtmelISP von Ulrich Bangert. Ponyprog stellt eine universelle Schnittstelle zu seriell programmierbaren Bausteinen da, AtmelISP ist auf die AT89S – Serie beschränkt. Der „Haken“ an beiden Programmen ist, dass es die zur Programmierung benötigten “Einsen“ und „Nullen“ über sogenanntes „Bitwackeln“ an der RS-232 herstellt, d.h. dass die Daten nicht über ein Protokoll an die Schnittstelle geschickt wird, sondern die Pins der Schnittstelle vom steuernden Anwenderprogramm auf 1 oder 0 gesetzt werden.

Diese Methode funktioniert leidlich und ist bisweilen mit Frust verbunden da es bisweilen langwierig sein kann, die richtigen Einstellungen zu finden. Außerdem sind beide Methoden sehr empfindlich, was die Qualität der Signale angeht. So kann es sein, dass der Programmieradapter der gerade eben noch sein Dienst getan hat nun ebensolchigen verweigert.

Außerdem sind Ponyprog und AtmelISP AUCH von der Rechengeschwindigkeit des PC's abhängig, Einstellungen die auf dem einen PC funktioniert haben, funktionieren auf einem anderen nicht zwingend. Tests haben gezeigt, dass auch viele USB – RS232 Adapter die Zusammenarbeit mit PonyProg oder AtmelISP verweigern.

Es sollte also ein zuverlässiger Adapter entstehen, der von der Rechenleistung des PC's unabhängig ist und über einen USB – RS232 Adapter betrieben werden kann.

Außerdem sollte dieser Adapter nicht über viele Konfigurationsmöglichkeiten verfügen, er soll einfach nur angeschlossen werden und funktionieren.

Um dieses zu erreichen, muß das Timing des ISP-Flash-Protokolls vom PC ausgelagert sein, es liegt also nichts näher, dieses Timing einem eigenständigen Microcontroller zu überlassen.

In der Theorie ist das nicht sonderlich schwierig, aber viele Fehler stecken im Detail (sogar sehr viele).

Schwierigkeiten bei der Entwicklung des ISP-Adapter waren:

- die Signale (insbesondere die auf der MISO – Leitung) sind sehr empfindlich, besonders bezüglich Anstiegszeiten der Taktflanken. Ohne schaltungstechnische Maßnahme könnte ein Adapterkabel nur sehr kurz ausgeführt sein.
- Empfindlichkeit gegen unterschiedlicher Massepotentiale von zu flashendem Baustein und der Masseleitung des flashenden Controllers
- unterschiedliche Timings der unterschiedlichen Bausteine

Alleine die Signalempfindlichkeit hatte die Entwicklung stark behindert, weil eine Funktion zeitweise gegeben war, danach aber nicht (ein Experimentierboard ist nun mal kein sauberer Aufbau).

Es ist ein Adapter entstanden, der zwar ein propietäres Protokoll besitzt (zu nichts kompatibel, dafür jedoch sehr einfach und gut beschrieben), der aber sehr stabil (auch über einen USB – RS232 Adapter) funktioniert.

Derzeit unterstützt werden die Atmel – Bausteine:

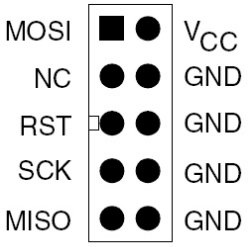
- **AT89S51**
- **AT89S52**
- **AT89S2051**
- **AT89S4051**
- **AT89S8253**

Die Bausteine **AT89S8252** und **AT89S53** werden **NICHT** unterstützt. Der Hauptgrund hierfür ist der, dass die beiden ein komplett anderes Protokoll zum beschreiben des Speichers besitzen und zudem bereits durch den AT89S8253 abgelöst sind.

Inhaltsverzeichnis:

AT89S ISP-Flasher

Das vorliegende Gerät ist ein Adapter mittels diesem es möglich ist MCS-51 Microcontroller der Fa. Atmel ohne Bootloader zu flashen. Ein Zielmicrocontroller erhält hierbei seine Daten über MISO, MOSI, SCK und RES Leitungen.



Die Belegung des ISP-Steckers des Adapters ist linksstehend abgebildet. Ein Stecksystem muß die Leitungen mit den entsprechenden Anschlüssen des Microcontrollers verbinden

Der Adapter ist als sogenanntes „Stand alone Gerät“ konzipiert und verarbeitet über die Bediensoftware Intel Hex-Dateien.

Intel Hex-Dateien werden von allen gängigen Compilern und Assembler erzeugt, sodaß es keine Probleme gibt, erzeugten Code in das Zielsystem zu übertragen

Das Gerät kann über Jumper konfiguriert werden:

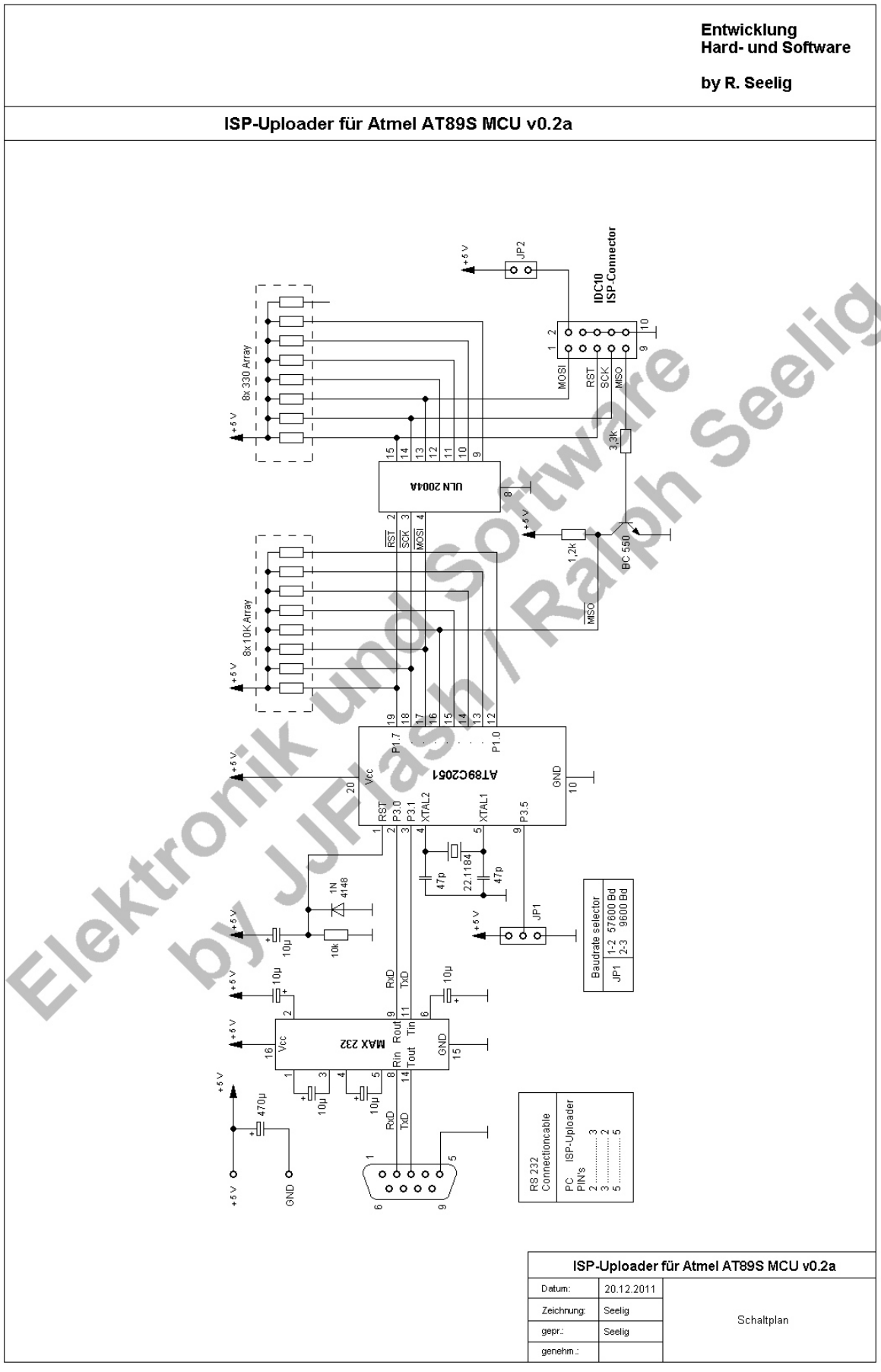
JP2 bestimmt, ob der Adapter seine Betriebsspannung aus der Zielschaltung bezieht

JP1 legt die Baudrate fest:: Jumperverbindung 1-2 entspricht 57600 Bd
Jumperverbindung 2-3 entspricht 9600 Bd

Um den ISP-Flasher an einem PC zu betreiben, muß dieser über eine serielle Schnittstelle verfügen, diese kann jedoch auch mittels einem USB-RS232 Adapter realisiert sein. Bisher sind keine USB-Adapter bekannt, mit denen der Flasher nicht betrieben werden konnte.

Sicherzustellen ist, dass der USB-RS232 Adapter einen Comport von 1 bis 8 belegt. Wie dies zu bewerkstelligen ist, ist der Anleitung des jeweiligen Adapter zu entnehmen.

Die Schaltung:



Die Schaltung kommt sehr unspektakulär daher und besteht aus Standardbauteilen. Als einziges Problem beim Nachbau besteht, einen bereits programmierten AT89C2051 (das ist KEIN ISP-Baustein) zu haben (kann aber beim Autor gegen einen Unkostenaufwand bezogen werden).

Alternativ zum AT89C2051 kann hier auch ein AT89S2051 oder ein AT89S4051 verwendet werden, wenn er zuvor bspw. über ISP geflasht wurde (aber hierfür braucht man eben auch einen funktionierenden ISP-Adapter).

Grundsätzlich wäre es möglich gewesen, den zu programmierenden ISP-Controller direkt über den AT89C2051 anzusteuern. Hier hat sich jedoch gezeigt, dass das mehr als nur kritisch ist. Die ISP-Leitungen (im besonderen die MISO-Leitung) ist so sehr empfindlich, dass die Leitungen die den Controller flashen sollen absichtlich sehr niederohmig gemacht wurden. Hierfür wurde zum „entkoppeln“ vom AT89C2051 ein ULN 2004 Darlingtonarray zwischengeschaltet. Da dieser in Emitterschaltung arbeitet müssen diesem Array alle Signale invertiert zugeführt werden (war bei der Firmware des Controllers zu beachten).

Gleiches gilt für die MISO-Leitung, die dem AT89C2051 über einen Transistor zugeführt wird.

Funktionsbeschreibung:

Der Microcontroller bezieht seine Daten grundsätzlich über eine RS-232, die auch über einen USB zu RS-232 Adapter hergestellt werden kann. Die Daten gelangen über einen MAX-232 zum Controller.

Damit der Adapter auch noch auf alten DOS oder WIN98 Systemen laufen kann, wurde eine Wahlmöglichkeit für die Baudrate integriert (ein DOS Programm steht derzeit nicht zur Verfügung, es diente dem Autor, den Adapter in ein bestehendes DOS-System zu integrieren).

JP1 legt die Baudrate fest (wählbare Baudraten sind 9600Bd und 57600Bd – siehe Schaltplan), das RS-232 Protokoll lautet: 1 Startbit, 8 Datenbits, keine Paritätsprüfung, 1 Stopbit.

Die Portleitungen, die den zu flashenden Controller steuern sollen, werden über den Darlingtontreiber ULN2004 verstärkt.

JP2 dient zur Wahl, ob der Adapter seine Betriebsspannung der Zielschaltung entnehmen soll, oder ob eine eigene Betriebsspannung angelegt wird.

JP2 geschlossen => Betriebsspannung über die Zielschaltung
JP2 offen => externe Betriebsspannungszuführung

Die Software ISPPROG:

Die PC-Software wurde für Microsoft-Windows © Betriebssysteme erstellt und wurde getestet unter den Systemen:

- Windows XP
- Windows Vista
- Windows 7

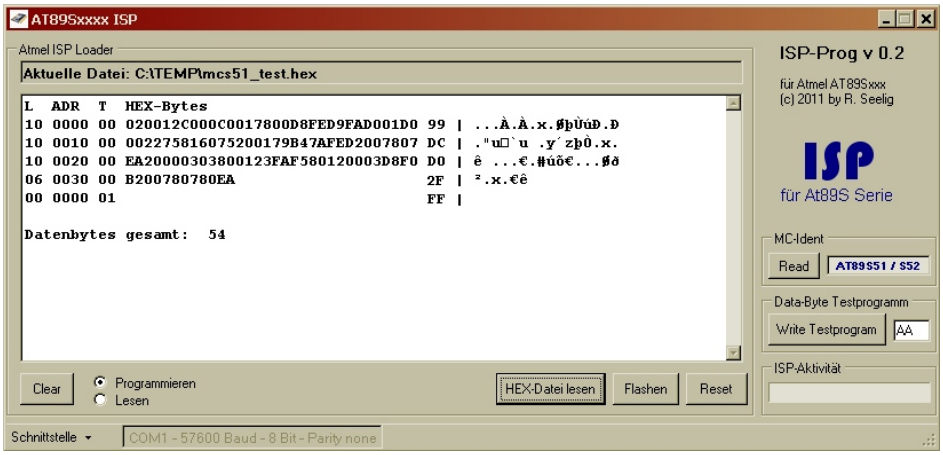
Zum Einsatz kamen unterschiedliche USB-RS232 Adapter und der Adapter der Fa. Logitech (LogiLink) zeigte sich als der schnellste. Allerdings gab es keinen einzigen Adapter der NICHT funktionierte (ein einzelner Adapter jedoch zeigte deutliche Verlangsamung in der Dateiübertragung).

Die Übertragung der Daten geschieht mit einer nicht veränderbaren Baudrate von 57600 Bd

Die Bedienung des ISP-Adapter gestaltet sich als einfach. Prinzipiell kann zwischen dem Schreibbetrieb (Programmieren) und Lesebetrieb (Lesen) umgeschaltet werden. Für die jeweilige Betriebsart wird das Anzeigefenster entsprechend umgeschaltet.

Die Beschriftungen der Bedienmöglichkeiten sind intuitiv gehalten. Um einen Baustein beschreiben zu können, muß die Verbindung zum ISP-Adapter hergestellt werden. Hierfür stehen COM-Ports von 1 – 8 zur Verfügung.

Betriebsart „Programmieren“



Screenshot „Programmieren“

Schnittstelle:

verbindet oder trennt den ISP-Adapter von der Schnittstelle

HEX Datei lesen:

wählt die zu flashende Datei vom Datenträger aus

Flashen:

Beschreibt den Baustein. Hierbei wird zuerst der Baustein gelöscht und anschließend mit der aktuell auf dem Datenträger gespeicherten Datei beschrieben. Wurde diese Datei seit dem letzten Auswählen mit „HEX Datei lesen“ verändert wird automatische diese Datei als Datenquelle benutzt.

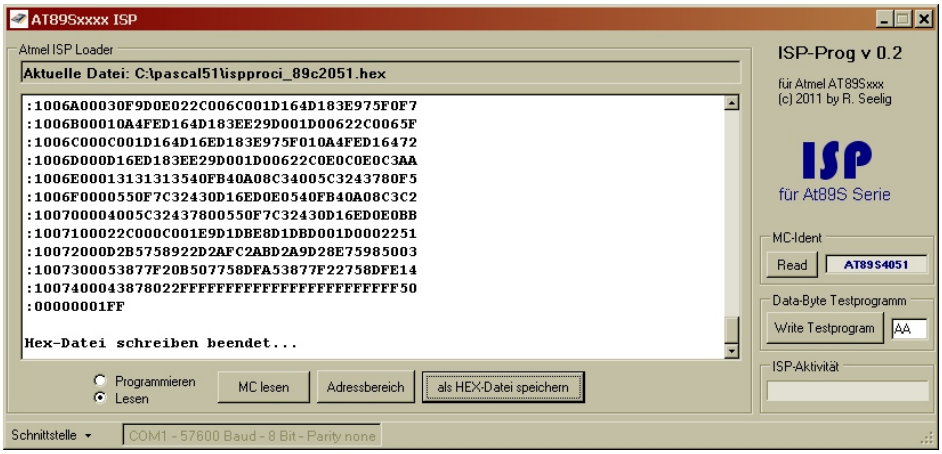
Reset:

Führt einen Reset am Zielcontroller durch.

Clear:

Löscht den Zielcontroller. Nach dem Löschen sind alle Codespeicherzellen mit 0FFh beschrieben.

Betriebsart „Lesen“



Screenshot „Lesen"

Adressbereich:

Legt die Start- und Endadresse des Bereiches fest, der gelesen werden soll

MC lesen:

Liest die mit Adressbereich festgelegten Daten aus dem Codespeicher des Controllers

Als HEX Datei speichern:

Speichert die mit „MC lesen“ gelesenen Daten als Intel-Hex Datei auf den Datenträger (diese Datei kann dann als Quelldatei zum Flashen eines Controllers dienen)

Möglichkeiten, die in beiden Betriebsarten zugänglich sind:

MC Ident: Read

Liest die Signatur des angeschlossenen Zielcontrollers aus und zeigt diese an

Data Byte Testprogramm: Write Testprogramm:

Diese Option dient zum grundsätzlichen Testen, ob eine Datenkommunikation mit dem Zielcontroller möglich ist. Es wird ein 5 Byte langes „Programm“ geschrieben, das lediglich die Aufgabe hat, einen Bytewert auf Port0 auszugeben. Welches Byte dort ausgegeben wird, wird im Eingabefeld festgelegt. Es werden folgende Codebytes geschrieben:

7580xx80FE

xx steht hier für das Byte, das im Eingabefeld angegeben ist.

ISP-Aktivität:

Zeigt an, wie lange die gewählte „Aktion“ dauert.

Die Firmware ISPPROCI_89C2051.ASM

Das Firmware oder auch das „Betriebsprogramm“ des AT89C2051 wurde mit dem Assembler ASEMW von W.W.HEINZ geschrieben.

Nach dem Einschalten der Betriebsspannung wartet der Controller auf Steuerkommandos vom Host (dem PC). Würde anstelle des Bedienprogramms ISPPROG ein Terminalprogramm angeschlossen sein, würde dieses vorerst nichts anzeigen.

Möchte man grundsätzliche Funktionen des Adapters mit einem Terminalprogramm vorab testen, so muß dieses folgende Einstellungen haben:

- 57600 Bd
- 1 Startbit
- 8 Datenbits
- Keine Parität
- 1 Stopbit

Die Firmware kennt 2 Betriebsmodi:

- Flashmodus
- Diagnosemodus

Flashmodus

Der Flashmodus ist der normale Betriebszustand der Firmware. In diesem Modus befindet sich die Firmware direkt nach dem Einschalten. Alle Kommandos der Firmware können auch mit einem Terminalprogramm (wenn auch etwas „ungelenk“) nachvollzogen werden.

Alle Zeichen die an den Controller geschickt werden oder vom Controller geliefert werden sind

!!! ASCII-Zeichen !!!

Es dürfen keine anderen Zeichen als die Ascii-Zeichen gesendet werden. Ein Kommando wird **NICHT** mit einem Return oder sonstwelchigem Zeichen abgeschlossen!

Flashmodus

Die Kommandos der Firmware im Flashmodus sind:

Kommando „1“ :	**Clear**

Der gesamte Codespeicher wird gelöscht. Nach erfolgter Aktion hat jede Codespeicherstelle den Wert 0FFh.

Kommando „2“ :	**Startadresse festlegen**

Struktur **2 Hex3 Hex2 rd3 rd2 Hex1 Hex0 rd1 rd0**

2 = Kommando
Hex3 = Hexziffer des höherwertigen Bytes der Adresse
Hex2 = Hexziffer des höherwertigen Bytes der Adresse
rd3 = rücklesen der Ziffer von Hex3
rd2 = rücklesen der Ziffer von Hex2
Hex1 = Hexziffer des niederwertigen Bytes der Adresse
Hex0 = Hexziffer des niederwertigen Bytes der Adresse

Diese Funktion legt für alle nachfolgende Lese- oder Schreibvorgänge die Startadresse fest. Die Startadresse bleibt so lange erhalten, bis sie durch erneuten Aufruf neu festgelegt wird. Die Funktion erwartet 2 x 2 Hexziffern als weitere Übergabewerte.

Beispiel:

Die Startadresse soll auf 1234h festgelegt werden. Dem Controller ist folgendes zu übermitteln:

Zeichenfolge „212“ senden. Danach werden vom Controller 2 Hexziffern geliefert die die übermittelten Daten beinhalten, in diesem Falle „1“ und „2“. Danach ist die Zeichenfolge „34“ zu senden. Der Controller antwortet dieses mal mit „3“ und „4“. Die Startadresse ist auf 1234h festgelegt.

Kommando „3“ :	**Programmlänge / Datenlänge festlegen**

Struktur **3 Hex3 Hex2 rd3 rd2 Hex1 Hex0 rd1 rd0**

3 = Kommando
Hex3 = Hexziffer des höherwertigen Bytes der Datenlänge
Hex2 = Hexziffer des höherwertigen Bytes der Datenlänge
rd3 = rücklesen der Ziffer von Hex3
rd2 = rücklesen der Ziffer von Hex2
Hex1 = Hexziffer des niederwertigen Bytes der Datenlänge
Hex0 = Hexziffer des niederwertigen Bytes der Datenlänge

Diese Funktion legt für alle nachfolgenden Lese- oder Schreibvorgänge die Anzahl der zu transferierenden Bytes fest. Die Funktion erwartet 2x2 Hexziffern als weitere Übergabewerte.

Da die interne Zählung des ISP-Adapters bei 0 beginnt, wird eine Festlegung der Datenlänge auf bspw. 0004 bei einem späteren Datentransfer 5 Bytes betragen.

Beispiel:

Es sollen insgesamt 0100h Bytes transferiert werden. Dem Controller ist folgendes zu übermitteln:

Zeichenfolge „301“ senden. Danach werden vom Controller 2 Hexziffern geliefert, die die übermittelten Werte beinhalten, in diesem Falle „0“ und „1“. Danach ist die Zeichenfolge „00“ zu senden. Der Controller antwortet dieses mal mit „0“ und „0“. Die Datenlänge ist auf 0100h festgelegt.

Kommando „4“ : Reset

Der Zielcontroller wird „geresetet“. Befindet sich ein lauffähiges Programm im Controller wird dieses nach dem Resetimpuls ausgeführt.

Kommando „5“ : Meldung und Versionsanzeige

Diese Funktion gibt eine Meldung der Firmware auf der seriellen Schnittstelle aus. Sie beinhaltet eine Copyrightmeldung, die Liste der unterstützten Controller sowie die Funktionen der einzelnen Kommandos.

Ist ein Terminalprogramm an den ISP-Adapter angeschlossen, müssen folgende Meldungen erscheinen:

```
AT89Sxx ISP-Flash Programmer / 4-Byte Protokoll
unterstuetzte Controller:  AT89S51   - AT89S52
                           AT89S2051 - AT89S4051
(c) 2011 by R. Seelig      AT89S8253

Steuerkommandos
1 - Clear  \ 2 - Startadresse  \ 3 - Programmlaenge
4 - Reset  \ 5 - Meldung       \ 6 - Flashen
7 - Lesen  \ 8 - MCU-Typ       \ 9 - Diagnosemode
```

Kommando „6“ : Flashen

Struktur 6 HexHi HexLo Antwort … HexHi HexLo Antwort

Diese Funktion beschreibt den Codespeicher des Zielcontrollers. Der ISP-Adapter erwartet einzelne zu flashende Bytes in Hexziffern. Jedes Byte besteht aus zwei Ascii-Hexziffern. Wurde der Zielcontroller korrekt beschrieben, antwortet der ISP-Adapter mit einem großen „O“ als Anzeige dafür, dass das flashen dieses einen Bytes [O]kay war. Trat ein Fehler auf und der Zielcontroller konnte nicht korrekt beschrieben werden, sendet der ISP-Adapter ein „E“ für [E]rror zurück.

Beispiel:

Die Startadresse wurde mit Funktion 2 auf 0000h und die Datenlänge auf 0004h festgelegt. Der ISP-Adapter erwartet nun insgesamt 5 Bytes, die er flashen soll.

Nach jeweils 2 gesendeten Hexziffern (die das zu flashende Byte bilden) antwortet der Adapter jeweils mit „O“ oder mit „E“. Dieser Vorgang wiederholt sich insgesamt 5 mal.

Kommando „7“ :	Lesen

Diese Funktion liest den Codespeicher des Zielcontrollers aus. Das Auslesen beginnt ab der in <Startadresse> angegebenen Speicherstelle und beträgt die in <Programmlänge> angegebene Anzahl+1 Bytes .

Beispiel:

Die Startadresse wurde mit Funktion 2 auf 0000h und die Datenlänge auf 0004h festgelegt. Es werden nun 5 mal 2 Hexziffern aus den ersten 5 Speicherstellen des Zielcontrollers übertragen.

Kommando „8“ :	Zielcontroller identifizieren

Diese Funktion liest die Kennung der unterstützten Microcontrollertypen aus und liefert diese in 2 Hexziffern als Ergebnis zurück.

Zurückgegebene Werte sind:

- 69 für einen AT89S51 oder AT89S52
- 23 für einen AT89S2051
- 43 für einen AT89S4051
- 73 für einen AT89S8253

Kommando „9“ :	Diagnosemodus starten

Diese Funktion startet den Diagnosemodus mithilfe diesem es möglich ist, einzelne Bytes auf der MOSI-Leitung zu senden oder auf der MISO-Leitung einzulesen.

Wird der Diagnosemodus gestartet, sind sämtliche Funktionen des Flashmodus solange nicht erreichbar, bis der Diagnosemodus beendet ist oder der ISP-Adapter ein- und wieder ausgeschaltet wird (entspricht einem Reset des ISP-Adapters).

Diagnosemodus

Der Diagnosemodus macht den meisten Sinn in Verbindung mit einem Terminalprogramm. Mithilfe dieser Option ist der Anwender in der Lage, dem Zielcontroller einzelne Bytes mittels MOSI zu senden oder mittels MISO zu lesen.

Das macht vor allem dann Sinn, wenn einzelne Lock- oder Userbits gesetzt werden sollen, die im Bedienprogramm ISPPROCI nicht realisiert sind.

Weiterhin ist es denkbar, erste Versuche mit eventuell zukünftig erscheinende Chips in diesem Modus zu bewerkstelligen. In diesem Falle wäre gleich eine Plattform vorhanden um mit brandneuen Bauteilen umgehen zu können. Bspw. müßte in diesem Modus auch ein AT89S8252, ein älterer Baustein der ein sogenanntes 3 – Byte Protokoll besitzt, ansprechbar sein.

Außerdem können im Diagnosemodus ohne das Bedienprogramm der angeschlossene Zielcontroller ermittelt oder der Speicher ausgelesen werden.

In den Diagnosemodus gelangt man durch ein Kommando „7“ im Flashmodus. Nach dem Aktivieren des Diagnosemodus wird auf der seriellen Schnittstelle folgende Meldung ausgegeben:

```
Diagnosemodus
Taste in Klammer zur Auswahl

-- m[o]si -- m[i]so -- i[n]it -- [r]ead -- [P]in default -- [e]xit --
```

Die Möglichkeiten sind:

m[o]si: Schickt ein einzelnes Byte seriell auf der MOSI Leitung zum Zielcontroller. Zum Aktivieren dieser Funktion muß ein „o“ an den ISP-Adapter gesendet werden, danach erfolgt eine Eingabeaufforderung des zu sendenden Bytes.

M[i]so: Liest ein einzelnes Byte seriell von der MISO Leitung. Zum Aktivieren dieser Funktion muß ein „i“ an den ISP-Adapter gesendet werden, direkt danach wird das gelesene Byte im HEX-Format angezeigt

i[n]it: Initialisiert den Zielcontroller durch das Senden des „Programm Enable“ Kommando des Controllers und liest im Anschluß daran die Kennung des Controllers die einem Klartext zugeordnet wird und diesen Klartext ausgibt.
Zum Aktivieren dieser Funktion muß ein „n“ gesendet werden.

[r]ead: Liest den Codespeicher des Zielcontrollers aus. Zum Aktivieren dieser Funktion muß ein „r“ gesendet werden. Anschließend wird die Start- und die Endadresse des zu lesenden Codebereichs abgefragt. Der Speicherinhalt wird mit Speicheradresse, Inhalt sowie dem Ascii-Äquivalent angezeigt

[p]in default: Setzt die Pins des ISP-Anschlusses auf ihren Ursprungszustand zurück (MISO, MOSI, RESET, SCK]

[e]xit: Beendet den Diagnosemodus, der ISP-Adapter gelangt in den Flash-Modus zurück.

Anhänge

Vergleich der unterstützten Microcontroller

Controllertyp	Programmspeicher	internes RAM	EEPROM	16-Bit Zähler	Flashdauer / Byte
AT89S51	4k	128 Byte	-	2	1 ms
AT89S52	8k	256 Byte	-	3	1 ms
AT89S2051	2k	256 Byte	-	2	4 ms
AT89S4051	4k	256 Byte	-	2	4 ms
AT89S8253	12k	256 Byte	2k	3	4 ms