MicroSPS
Quickstart

Einfachstes Erstellen von Steuerungen durch grafische Funktionseingabe in EAGLE

Hardware: SPS-Ctrl

Eine kurze Einführung
&
Inbetriebnahme der Hardware

www.MicroSPS.com
www.mikrocontroller.com

Vorwort

Eine MicroSPS ist eine Mikrocontroller gesteuerte Applikation, die eine Anweisungsliste abarbeitet, welche mit dem EAGLE Schaltungseditor erstellt und übersetzt wurde.

Wir haben eine MicroSPS mit einem AVR-Mikrocontroller der Firma Atmel (ATMEGA32) entwickelt. Als Hardwareplattform dient die Leiterkarte SPS-Ctrl, die der Anwender sich nach seinen Bedürfnissen bestücken kann.

Diese Quickstart-Beschreibung zeigt, welche Schritte erforderlich sind, die Hardware in Betrieb zu nehmen und eine kleine Anwendung zu realisieren. Wir setzen hier Grundkenntnisse mit EAGLE voraus. Für eine genaue Dokumentation der einzelnen Elemente und Parameter sollte man auf das Handbuch „MicroSPS_Handbuch_Vx.pdf“ zurückgreifen.

Wer nicht mit EAGLE vertraut ist, sollte sich zunächst den Grundkurs auf:
http://www.MicroSPS.com
ansehen.

Die Komponenten der MicroSPS-Entwicklung

Folgende Komponenten werden zur Entwicklung einer MicroSPS-Anwendung benötigt:

– Einen Windows-PC mit serieller Schnittstelle oder USB-Seriell-Adapter

– Das Sofwarepaket EAGLE ab Version 4.16. Für kleinere bis mittlelgroße Schaltungen und nicht-kommerzielle Anwendungen kann die Freeware bzw. die Light-Version verwendet werden (Download von www.cadsoft.de).

– Die Installer-Dateien des MicroSPS-Projekts (siehe www.MicroSPS.com).

Zunächst muss EAGLE auf dem PC installiert werden. Erst danach startet man die Installation des MicroSPS-Projekts.

Die SPS-CTRL als MicroSPS-Hardware

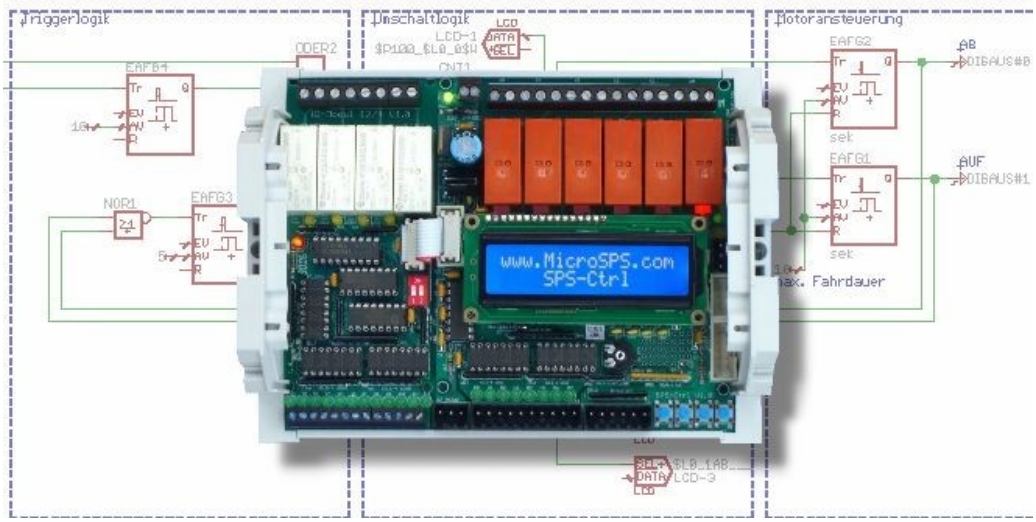


Bild: SPS-Ctrl mit einem 12/4-Erweiterungsmodul in einem Hutschienengehäuse (hier ohne Klarsichtdeckel)

Technische Daten der SPS-CTRL:

- 8 Digitaleingänge, geschützt durch Optokoppler
  Die gemeinsame Masse ist potenzialfrei zur Hauptmasse
- 4 Tasten
- 6 Relais zum Schalten von Lasten
- 2 Digitalausgänge mit Treiberbaustein (Open-Collector) zum Ansteuern von Relais, Leuchten, Motoren,..
- 4 Analogeingänge (0..5 V oder 0..10V) auf Stecker
- Poti zur schnellen Eingabe von Werten oder zum Simulieren eines analogen Eingangs
- PWM-Ausgang z.B. zum Generieren einer analogen Spannung
- LC-Display mit 2*16 Zeichen
- Display und Tastatur per Kabel auch extern im Parallelbetrieb möglich
- kompakte Grösse (ca.12*10cm)
- Einbau im Hutschienengehäuse mit Klarsichtdeckel möglich
- LEDs zeigen Zustand der Digitalausgänge und Eingänge an
- Betriebs-LED zeigt Betriebszustand an
- Eingestellte Digitalwerte werden bei Spannungsausfall abgespeichert (siehe Funktion WERT_VAR) (in Vorbereitung)
- Echtzeituhr mit Batteriepufferung; läuft auch b ei Spannungsausfall weiter
- Versorgung 12V oder 24V Gleichspannung
- Serielle Schnittstelle mit V24-Pegeln
- Infrarotempfänger für Infrarot-Fernbedienungen (RC5-Protokoll, z.B. Phillips)
- Flachbandstecker für Erweiterungsmodule (z.B. Web-Interface (in Vorbereitung))
- die Leiterkarte ist in Industriequalität gefertigt und geprüft

Die Platine „SPS-Ctrl“ ist speziell für die Anwendung als MicroSPS entwickelt worden. Sie lässt sich mittels Erweiterungsmodulen mit zusätzlichen Modulen weiter ausbauen. Wir haben hier besonderen Wert auf viele Ein- und Ausgänge, Erweiterbarkeit und bessere Einbaumöglichkeit gelegt. Durch Features wie Echtzeitur und absetzbares Display und Tastatur ist sie ideal für diese Anwendung.
Zahlreiche Funktionsblöcke der MicroSPS sind speziell auf diese Hardwareplattform zugeschnitten, sodass auch aufwendige Projekte mit Fernbedienungen, LCD-Ausgaben, Menüführung, Zeitsteuerungen, analoger Datenverarbeitung usw. zum Kinderspiel werden.

Die kompakte Leiterkarte (ca.10*12cm) findet, wie im Bild zu sehen, hervorragend Platz in einem Phoenix-Hutschienen-Gehäuse. Die Ein- und Ausgänge, sowie die vier Tasten sind auch noch von außen zugänglich, wenn die restliche Elektronik vom Klarsichtdeckel des Gehäuses verdeckt sind. Sie lässt sich mit 12V, oder 24V DC versorgen (Achtung: Bestückungsvariante). Die Analogeingänge sind mit 0...10 V ebenso „SPS-konform“ wie die Digitaleingänge mit 24 V DC. Sechs Relais bilden die Schnittstelle zu Lampen, Garagentoren, Markisen und anderen Verbrauchern, die gesteuert werden sollen.

Eine SPS-CTRL Leiterkarte ist bei: www.mikrocontroller.com oder www.MicroSPS.com erhältlich. Sie kann je nach Anwendung mit unterschiedlicher Peripherie ausgestattet werden. Dort findet man auch die Firmware (den Bootloader und Schaltungsinterpreter) in *.HEX-Form.

Alle Teile sind bedrahtet und leicht und kostengünstig beschaffbar (z.B. Reichelt).

Digitales Erweiterungsmodul 12/4

Die Hardware der SPS-Ctrl kann mit dem Modul12/4 um digitale I/Os erweitert werden.

Technische Daten des 12/4-Moduls:

– 12 Digitaleingänge, geschützt durch Optokoppler
  Die gemeinsame Masse ist potenzialfrei zur Hauptmasse
– 6 Relais zum Schalten von Lasten
– Anschluss an die SPS-Ctrl mittels 10-poligem Flachbandkabel
– bis zu viel Module anschließbar (dazu entsprechend mehrere Stecker auf die Flachbandleitung krimpen)

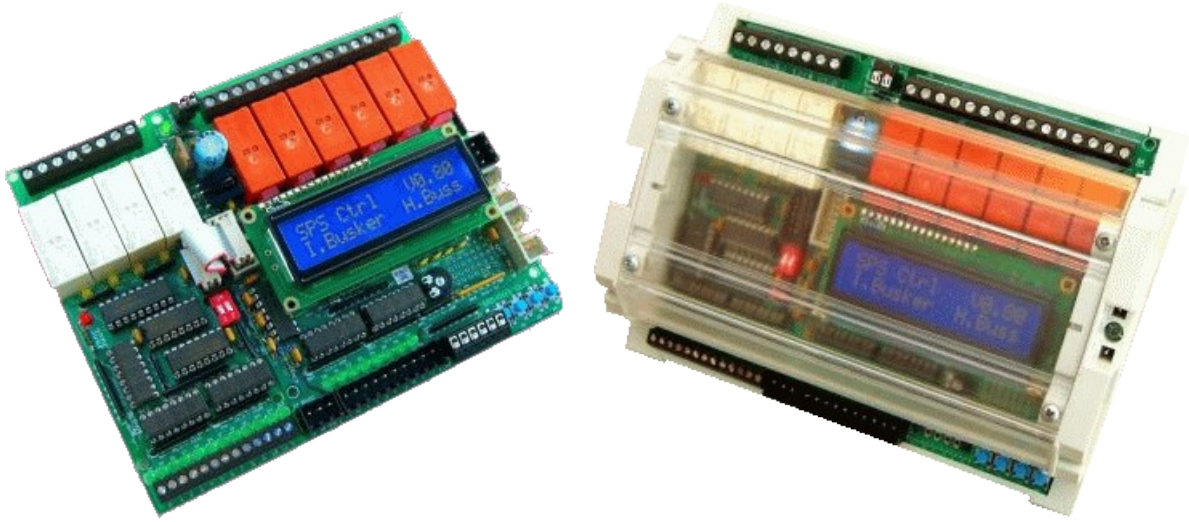


Bild: SPS-Ctrl mit einem Erweiterungsmodul ohne Gehäuse und mit Gehäuse und Klarsichtdeckel

Beschreibung der Hardware

Controller

Ein Atmel ATMEGA32 bildet das Kernstück der MicroSPS. Er speichert und verarbeitet den Funktionsplan und bedient die Peripherie wie Uhr, Display, I/O, Infrarotempfänger usw. Der Funktionsplan (SPS-CODE) wird mittels Downloadtool von einem PC über die serielle Schnittstelle in den Controller geladen. Den ATMEGA32 gibt es auch in Einzelstückzahlen noch preisgünstig im bedrahtetem 40-Pin-Gehäuse. In Vorbereitung ist eine Variante mit ATMEGA644, die dann unter anderem 25% schneller arbeitet.

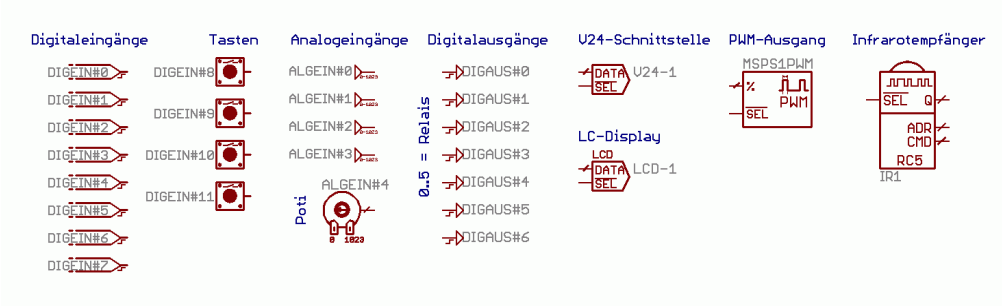


Bild: I/O-Komponenten der SPS-Ctrl im Funktionsplan

Digitaleingänge

Die Digitaleingänge 0 bis 7 sind über Optokoppler realisiert und deshalb potenzialfrei. Die gemeinsame Masse der Digitaleingänge befindet sich auf Stecker X8.1 und X8.10. Die in Reihe geschalteten LEDs zeigen den Zustand des jeweiligen Eingangs an. Sollen die Eingänge in einem Spannungsbereich von 5 V bis 12 V verwendet werden, müssen die Widerstandsnetzwerke RN3 und RN4 680 Ω haben. Bei einer Verwendung mit 12 V bis 30 V sollte 4,7 kΩ eingesetzt werden. Der Funktionsblock eines Digitaleingangs liefert einen binären Wert von 1, wenn Spannung anliegt und 0 bei Spannungsfreiheit.

Relais

Sechs Relais zum Schalten von größeren Lasten können auf der SPS-Ctrl-Platine bestückt werden. Der Schaltzustand der Relais wird mit LEDs angezeigt. Die Relais werden von IC4 getrieben. Je drei Kontakte der Relais K0, K2 und K4 sind als Wechsler auf Anschlussklemmen gelegt. Die Relais K1, K3 und K5 können nur als Schließer verwendet werden. Ein In-Reihe-Schalten eines Schließers mit einem Wechsler bietet sich immer an, wenn man zur Sicherheit eine hardwaremäßige Verriegelung benötigt, wie z.B. beim Steuern von Garagentoren oder Markisen. Wird die Leiterkarte mit 24 V versorgt, müssen 24-V-Relais eingesetzt werden und als Widerstandsnetzwerk RN5 kommt 4,7 kΩ zum Einsatz. Hinter den Funktionsblöcken DIGAUS0 bis DIGAUS5 verbergen sich die Relais, die als Eingangswert einen binären Wert (1 = Ein, 0 = Aus) benötigen. Einen weiteren Digitalausgang bildet der Kontakt an Stecker X10, an den z.B. ein weiteres Relais extern angeschlossen werden kann.

PWM

Der Funktionsblock PWM liefert ein pulsweitenmodulieres Signal am Ausgang X10.2. Als Eingangswert benötigt er eine 16-Bit-Zahl mit einem Wertebereich von 0..100. Aus dem PWM-Signal lässt sich leicht eine analoge Spannung gewinnen, indem man ein RC-Glied nachschaltet. Auch lässt sich die Drehzahl eines kleinen Gleichstrommotors stellen, der direkt zwischen X10.1 und X10.2 angeschlossen wird.

Analogeingänge

Die vier Analogeingänge ALGEIN0 bis ALGEIN3 können je nach Bestückung von RN2 entweder einen Spannungsmessbereich von 0-5V oder 0-10V abdecken. Der Funktionsblock liefert Werte von 0 bis 1023 als 16-Bit-Zahl. Zum Testen des Funktionsplans oder zum Einstellen eines Schwellwertes hat sich das Poti auf der Leiterkarte als praktisch erwiesen. Sein Funktionsblock liefert einen 16-Bit-Wert mit einem Wertebereich von 0 bis 1023. Wer keine Verwendung für ein Poti hat, kann es auch weglassen und den Eingang auf Stecker EXT.5 als weiteren 5-V-Eingang verwenden.

LCD

An die MicroSPS kann ein 2*16-Zeichen Display angeschlossen werden. Alle Displayleitungen incl. Versorgung sind auf Stecker „USR“ geführt. Somit lassen sich Display und Tastatur mittels Flachbandkabel einfach absetzen und auch größere Displaytypen mit z.B. 4 mal 20 Zeichen können zum Einsatz kommen. R2 ist der Kontrastwiderstand, der sich mit 220 Ω für die meisten LCDs bewährt hat. Sollte das verwendete Display zu wenig oder zu viel Kontrast aufweisen, muss der Widerstandswert angepasst werden. Bei der Verwendung der MicroSPS liefert das Digitalsignal der Tasten eine 1, wenn diese Taste betätigt wird. Eine LCD-Ausgabe kann sehr vielfältig formatiert werden. So kann an den Funktionsblock „LCD“ z.B. ein Wert als 16-Bit-Signal angelegt werden, der dann formatiert und skaliert angezeigt wird.

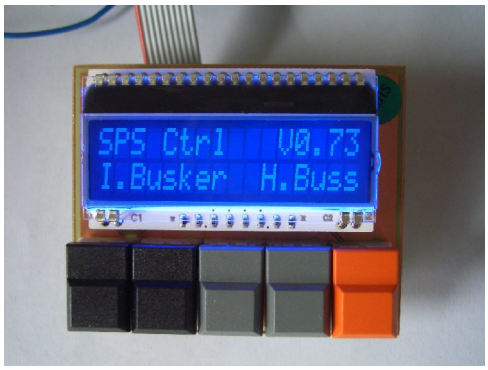


Bild: Abgesetztes Display mit fünf Tasten.

Infrarotempfänger

Ein Infrarotempfänger TSOP1736 empfängt und demoduliert das Signal einer handelsüblichen Fernbedienung. Der Controller verarbeitet Infrarotprotokolle im RC5-Datenformat von Philips. Eine programmierbare Fernbedienung, die z.B. auf „Philips TV“ eingestellt wird, leistet hier brauchbare Dienste. Im RC5-Protokoll gibt es einen Adresscode (z.B. 0 = TV) und einen Funktionscode (z.B. 16 = Power). Der Funktionsblock liefert entweder den Funktionswert und Adresscode als 16-Bitzahl am Q-Ausgang oder getrennt an den ADR- und CMD-Ausgängen.

Echtzeituhr (RTC)

Der Baustein DS1307 bildet zusammen mit einem Quarz und einer Batterie eine Echtzeituhr, die auch nach Spannungsausfall weiter läuft. Das Board kann auch ohne Echtzeituhr aufgebaut werden. In dem Fall nutzt der Controller interne Timer für die Uhrzeitfunktionen. Allerdings führt dann ein Reset oder Spannungsausfall zum Verlust der aktuellen Uhrzeit.

Die serielle Schnittstelle

Die serielle Schnittstelle des Bausteins liefert über den MAX232 normgerechte RS232-Pegel für einen PC. Sie wird zum Einen zum Einspielen des Funktionsplans und des Schaltungsinterpreters benötigt. Zum Anderen können hier sonstige Daten z.B. zum Mitprotokollieren von Messwerten ausgegeben werden. Der Funktionsblock „V24-Ausgabe“ wird genauso bedient, wie ein LCD-Block. Allerdings wird hier die Cursorpositionierung nicht unterstützt.

Inbetriebnahme der Platine

Einspielen des Bootloaders mit PonyProg

Der Bootloader der MicroSPS übernimmt folgende Aufgaben:
- Initialisieren des Controllers
- Starten des Schaltungsinterpreters
- Update des Schaltungsinterpreters per serieller Schnittstelle
- Einspielen des SPS-Codes ins Flash über die serielle Schnittstelle

Achtung:
Nur der Bootloader wird per ISP eingespielt. Ist dann ein Bootloader im Chip vorhanden, müssen Schaltungsinterpreter und SPS-Codes per serieller Schnittstelle programmiert werden.
Wer bereits einen Controller mit Bootloader hat (z.B. eine kommerzielle Version der MicroSPS), kann die folgenden Schritte überspringen!

Falls der Controller noch keinen Bootloader besitzt, muss er per ISP (In-System-Programmer) zunächst eingespielt werden. Nach Installation des MicroSPS-Komplettpakets von www.MicroSPS.com findet man das HEX-File z.B. unter C:\Programme\EAGLE-4.16\projects\MicroSPS\Hex\Bootloader_MEGA32_Vx_yy.hex. Das Einspielen geht am einfachsten mit einem sog. ISP-Kabel, das den Controller entweder über die parallele Schnittstelle oder die serielle Schnittstelle des PCs programmiert. Das ISP-Kabel kann man schnell aus einigen Teilen zusammenbauen. Als Software hat sich die Freeware „PonyProg“ bewährt. Ein Link zu Ponyprog2000 (www.lancos.com) und eine kurze Anleitung zur Bedienung sowie einen fertigen ISP-Adapter findet man ebenfalls unter www.MicroSPS.com.

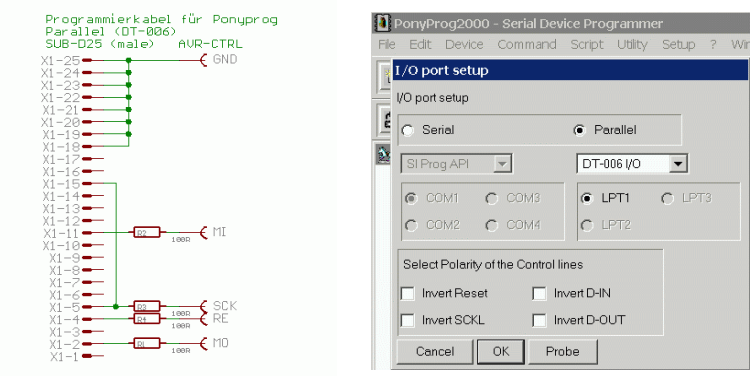


Bild: Schaltplan des parallelen ISP-Adapters und der entsprechenden Einstellung unter Ponyprog

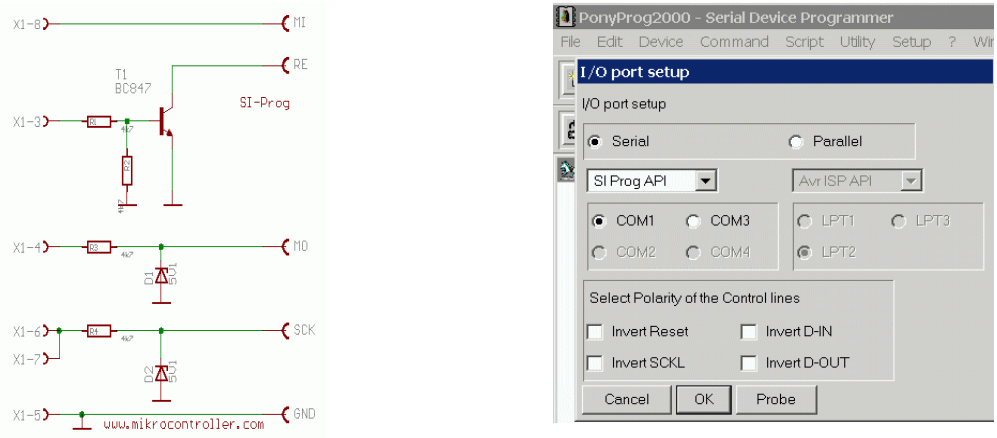


Bild: Schaltplan des seriellen ISP-Adapters und der entsprechenden Einstellung unter Ponyprog

Bedienung von Ponyprog

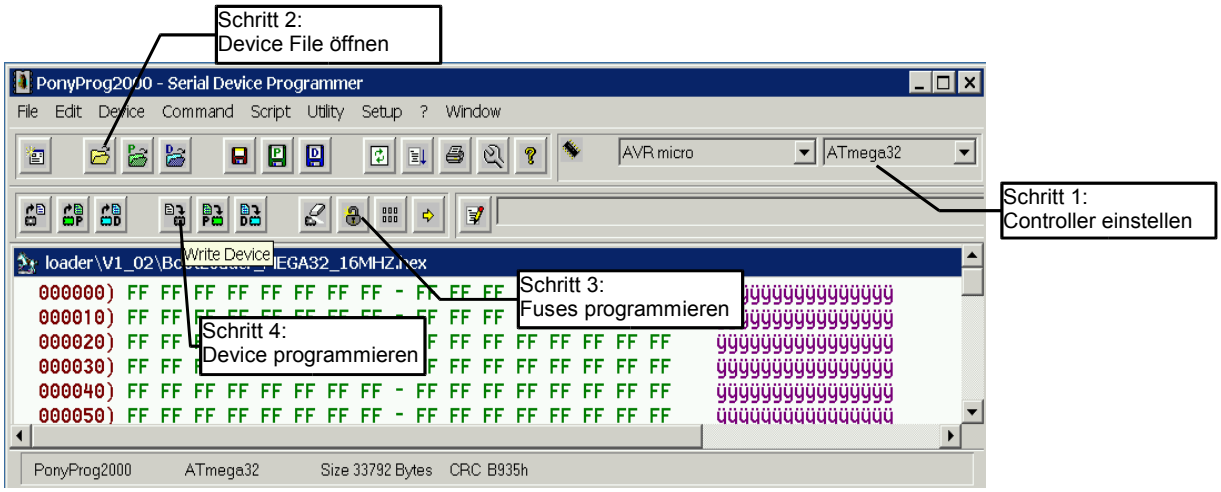


Bild: Schritte zum Programmieren des Bootloaders:

1. Controller: AVR micro --> ATMEGA32 einstellen
2. Das File: „Bootloader_MEGA32_Vx_yy.hex“ (oder ähnlich) laden
3. Fusebits (Konfigurationsbits) des Controllers einstellen (Siehe folgendes Bild) und mit „Write“ schreiben
4. Controller programmieren

Hinweis:
Bei etwaigen Problemen mit Ponyprog, findet man in dem Forum von www.microsps.com bereits viele Tipps

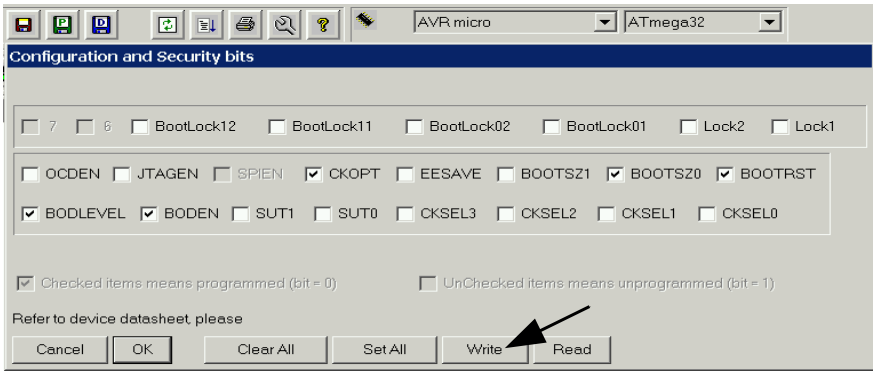


Bild: Ponyprogeinstellung der Fusebits bei Verwendung des externen Quarzes, der Brown-Out-Detection und des Bootloaders.

Einspielen des Schaltungsinterpreters mittels Download-Tool

Nach dem erfolgreichen Einspielen des Bootloaders wird nun der Schaltungsinterpreter per serieller Schnittstelle in den Controller geladen. Dazu verwendet man das Downloadtool „MicroSPS Downloader“, welches man als Verknüpfung auf dem Desktop findet. Unter: „Einstellungen → Firmware updaten“ lädt man das File: MicroSPS-SPS-CTRL_Vx_yy.hex (oder ähnlich) und spielt somit die Software ein.

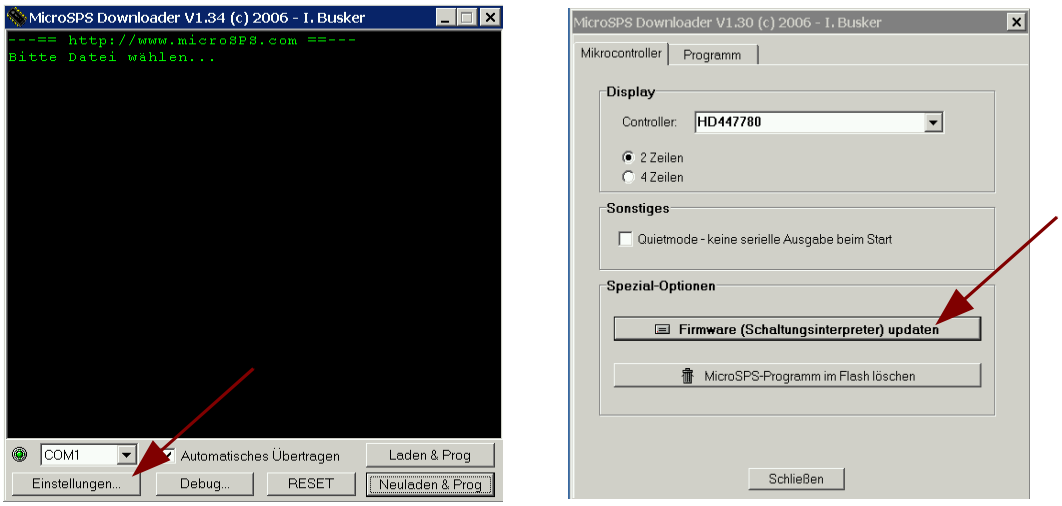
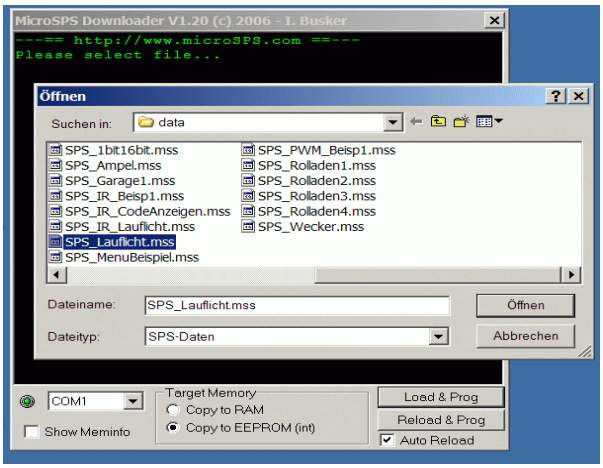


Bild: Einspielen des Schaltungsinterpreters mittels MicroSPS Downloader

Nach dem erfolgreichen Einspielen des Schaltungsinterpreters, meldet sich die MicroSPS im Terminalfenster des Downloaders und eine Ausschrift wird auf dem LCD angezeigt.

Einspielen von Beispiel-Dateien

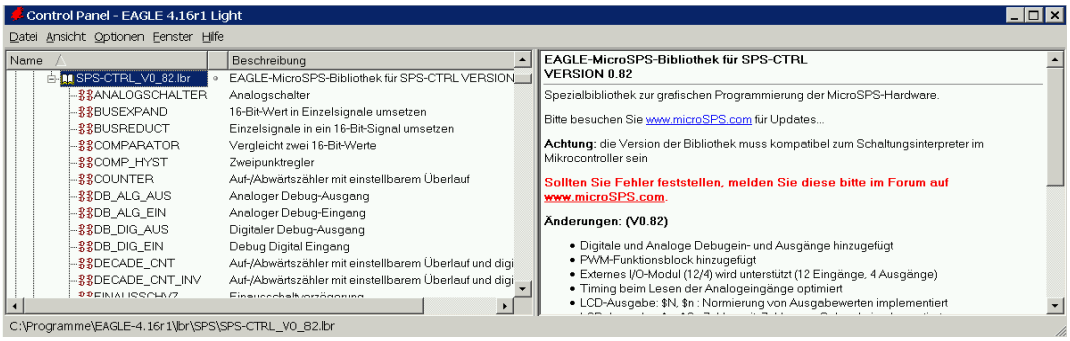
Nun kann das erste MicroSPS-Programm übertragen werden. Dazu dient der Button „Load & Prog“ im MicroSPS-Downloader. Die *.MSS Dateien im „Data“-Verzeichnis sind übersetzte MicroSPS-Beispiele. Die Beispiele, die im Namen mit „SPS_“ anfangen, sind die Dateien für die SPS-Ctrl Hardware.



Vorbereitung

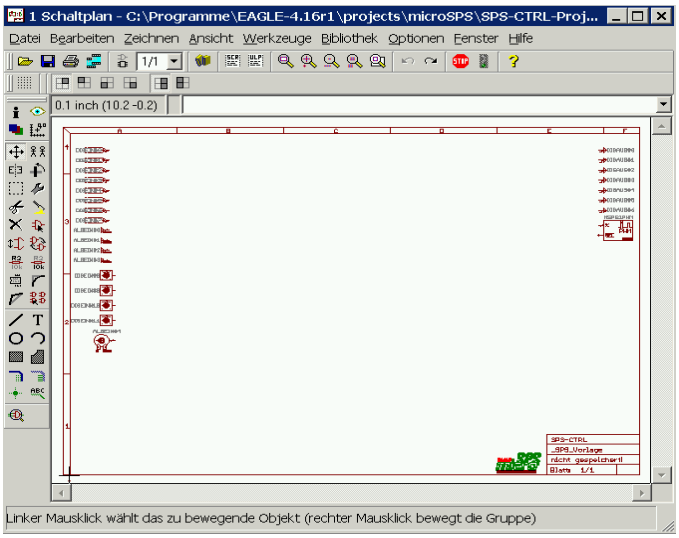
1. Öffnen der Bibliothek: „SPS-Ctrl_Vx_yy.lbr“

Nachdem die Bibliothek geöffnet wurde, stehen alle für die MicroSPS möglichen Funktionsblöcke im Bibliotheksbrowser von EAGLES Control Panel zur Verfügung.



Achtung:
Bauteile aus anderen Bibliotheken können nicht verwendet werden!
Die Version der Bibliothek muss mit dem Schaltungsinterpreter kompatibel sein.

2. Laden einer Vorlage: „_SPS_Vorlage.sch“ (für SPS-Ctrl)



Zum Erstellen eines neuen Projektes sollte man mit unserer Vorlage beginnen.

Falls die MicroSPS-Dateien neu installiert wurden, muss EAGLE durch Starten des ULP-Scriptes initialisiert werden. Dazu betätigt man im Schaltplan-Editor den ULP-Knopf und wählt die Datei
..projects/microsps/programs/init-msps.ulp aus.

Danach muss man einmal F11 drücken, um EAGLE für die Verwendung von Schaltplanelementen ohne Gehäuse zu initialisieren.

In der Vorlage erscheinen die wichtigsten Ein- und Ausgänge der Hardware.

Danach sollte man den Schaltplan unter dem neuen Projektnamen speichern, um sich die Vorlage nicht zu überschreiben.

Hinweis:
Gelöschte Elemente können mit der Taste F11 (Invoke) wieder ausgewählt werden.
Board-Elemente (Ein- und Ausgänge) dürfen nicht durch Kopieren verdoppelt werden.

Falls man die Vorlage nicht verwenden möchte:

1. Eagle initialisieren
Zunächst muss EAGLE für die Verwendung von Schaltplanelementen ohne Gehäuse initialisiert werden. Das geschieht durch Starten des ULP: init-msps.ulp
2. Anlegen eines neuen Schaltplans
3. Platzieren der SPS-Ctrl MicroSPS-Grundeinheit: „MICRO_SPS“

3. Starten des Download-Tools

Mittels diesen Tools wird später das Script in die Zielapplikation geladen.

Laden & Prog lädt ein neues Programm
Neuladen & Prog lädt das Programm erneut

Siehe auch die Anleitung „Downloader & Debugger“ (Separates PDF-File)

Erstellen eines Minimalprogramms

Hier zeigen wir, welche Schritte für die Programmierung erforderlich sind.

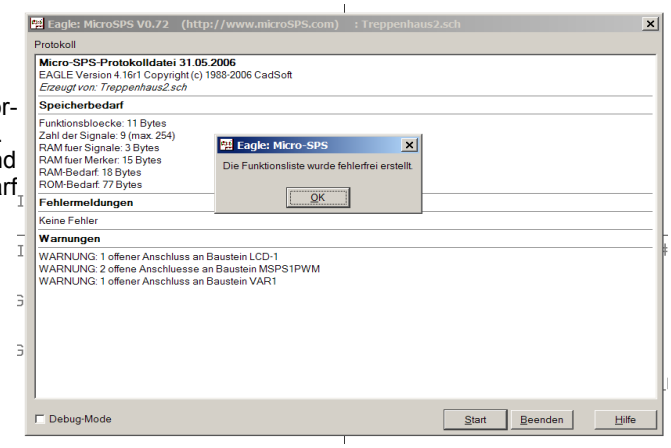
1. Funktionsplan eingeben

Der Funktionsplan kann jetzt eingegeben werden. Wir verbinden einfach einen Ausgang mit einem Eingang um ein Minimalprogramm zu erstellen. Dazu schieben wir die Elemente in Position und verbinden sie mit dem NET-Befehl. Damit bekommt der Taster auf der MicroSPS-Hardware direkten Zugriff auf den Digitalausgang0.

Bild: Minimalprogramm: Der Taster steuert direkt den Digitalausgang

2. Übersetzen des Funktionsplans in eine Anweisungsliste

Jetzt kann der Funktionsplan in das maschinenlesbare Format der Anweisungsliste (SPS-Programm) übersetzt werden. Mit der Taste F12 in EAGLE öffnet sich der Übersetzer und der Button "Start" startet die Übersetzung. Speicherbedarf und etwaige Fehler werden angezeigt.



3. Einspielen des SPS-Codes in die MicroSPS

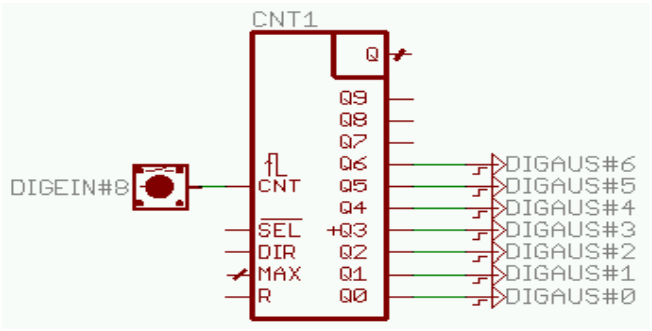
Nun startet man das Übertragungsprogramm zum Überspielen der Anweisungsliste in die Zielhardware. Über "Laden & Prog" wählt man das erzeugte Script aus und lädt es in die Hardware. Das gespeicherte Programm ist auch nach Aus- und Einschalten noch präsent und wird sofort nach dem Einschalten gestartet. Besonders praktisch ist die „Auto Reload“-Funktion des Übertragungsprogramm. Es überträgt automatisch im Hintergrund die neue Anweisungsliste, wenn diese aktualisiert wurde.

4. Test und Modifikation

Das Minimalprogramm läuft jetzt in der Hardware und kann getestet werden. Dazu drückt man auf der MicroSPS die Taste S1. Nun muss Relais0 schalten. In der folgenden Entwicklung modifiziert man lediglich die Schaltung und übersetzt es erneut mit der Funktionstaste F12. Das Download-Tool erkennt das Aktualisieren der Anweisungsliste und spielt sie automatisch in die MicroSPS ein. Der grafischen Entwicklung der eigenen Schaltung steht jetzt nichts mehr im Wege.

Beispiel: Lauflicht

Funktionsplan eingeben



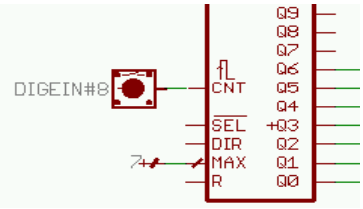
Wir wählen das Element „DECADE CNT“ aus der Bibliotek aus und platzieren es. **Die Symbole holt man am besten mittels des Control Panels von EAGLE aus der Bibliothek**, weil man dann unter anderem zu jedem Symbol eine Beschreibung hat.
An die Ausgänge Q0 bis Q7 schliessen wir die Digitalausgänge 0..7 an.
Den Digitaleingang0 verbinden wir mit dem Zählereingang CNT.
An die einfachen Eingänge eines Symbols können nur digitale Signale angeschlossen werden, die nur die Werte 0 und 1 annehmen.
Ein- und Ausgänge mit Balken (hier MAX und Q) können 16-Bit Variablen oder 1-Bit Variablen angeschlossen werden.

Hinweis:
Sollte die Meldung „Bauteil XY hat kein Package“ kommen, muss einmalig die Taste F11 betätigt werden und das sich öffnende Fenster geschlossen werden.

Test:

Wir übersetzen das Projekt mit F12 und spielen es ein. Bei jedem Tastendruck auf die erste Taste auf der SPS-Ctrl schreitet der Zähler um eins weiter und der nachfolgende Digitalausgang wird geschaltet. Da dem Zähler kein MAX-Wert zugeordnet wurde, läuft er immer weiter, auch wenn dem Zählerstand kein Ausgang zugeordnet wurde. Das bedeutet, dass nach Erreichen des Zählerstandes 7 alle LEDs aus sind.

Zuordnen eines MAX-Wertes für den Zähler

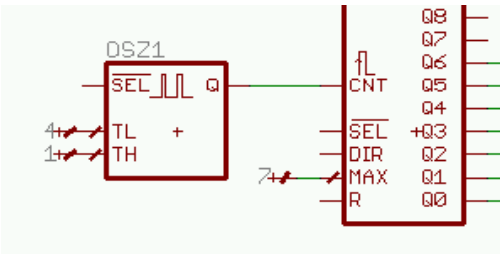


Wir wollen, dass der Zähler bei Erreichen des Zählerstandes 7 wieder bei Null anfängt.
Dazu wählen wir aus der Bibliotek das Symbol: „WERT“ aus und schliessen diesen an den MAX-Eingang an. Mit dem EAGLE-Befehl: Value legen wir den Wert der Variablen auf 7 fest. Mit dem Symbol „WERT“ können 16Bit-Konstanten angeschlossen werden.

Test:

Wir übersetzen das Projekt mit F12 und spielen es ein. Der Zähler schreitet nun bis 7 voran und setzt sich danach selbst wieder auf 0. Danach geht das Spiel von vorne los.

Anschluss eines Oszillators

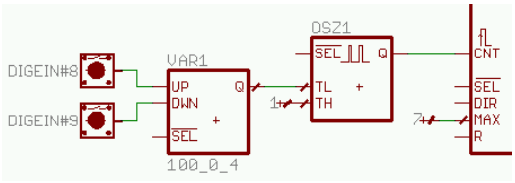


Unser Lauflicht soll selbständig durchlaufen, dazu wählen wir als Taktgeber den OSZILLATOR aus. Den Digitaleingang löschen wir mit dem Delete-Befehl von EAGLE, mit F11 können wir ihn uns später wiederholen. Der Oszillator bekommt an den Eingängen für die Low-Pulsbreite TL den Wert 4 und die High-Pulsbreite TH den Wert 4 (dazu holen wir uns wieder die Symbole „WERT“ aus der Bibliothek). Der Oszillator läuft standardmäßig mit einer Zeitbasis von 10ms. Die Summe der TH und TL-Zeiten sind demnach 50ms, was 20Hz entspricht. Man kann die Zeitbasis des Oszillators z.B. auch auf eine Sekunde ändern, indem man als Value „sek“ eingibt (dann „rappeln“ die Relais auch nicht so).

Test:

Wir übersetzen das Projekt mit F12 und spielen es ein. Nun läuft das Lauflicht schon selbständig.

Einstellbare Werte

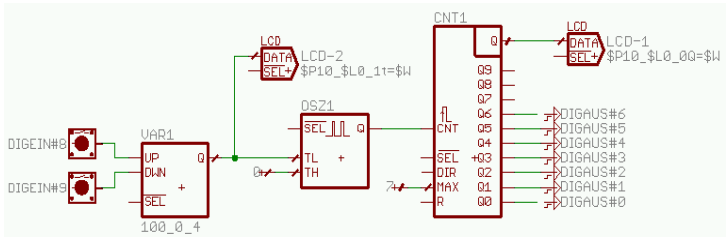


Die Geschwindigkeit soll einstellbar sein. Dazu ersetzen wir die TL-Zeit durch das Element WERT_VAR. An dessen Eingängen schließen wir zwei Digitaleingänge an. Dem Element WERT_VAR können wir Maximalwert, Minimalwert und Defaultwert im Value mitgeben, die durch Unterstriche getrennt werden müssen.
Hier: Max = 100 Min = 0 Default = 4
Die Tasten an den UP und DWN Eingängen können nun den Wert dieser einstellbaren Variablen ändern.

Test:

Wir übersetzen das Projekt mit F12 und spielen es ein. Nun können wir die Geschwindigkeit des Lauflichts mit den Tasten einstellen.

Anzeige auf dem Display



Die Geschwindigkeit und der Zählerstand sollen am LCD angezeigt werden. Dazu platzieren wir das Modul LCD_AUSGABE und schließen den Eingang DATA an die Signale an, deren Werte wir anzeigen möchten.

Mit Value übergeben wir dem Display einige Parameter.
Display1:
 $P10_$L0_0Q=$W

Bedeutet:
$P10_ Anzeige alle 10*10ms aktualisieren
$L0_0 Anzeige in Zeile 0 von Spalte 0 beginnen
Q= Es soll der Text „Q=“ angezeigt werden
$W Es soll ein Wert dezimal ausgegeben werden

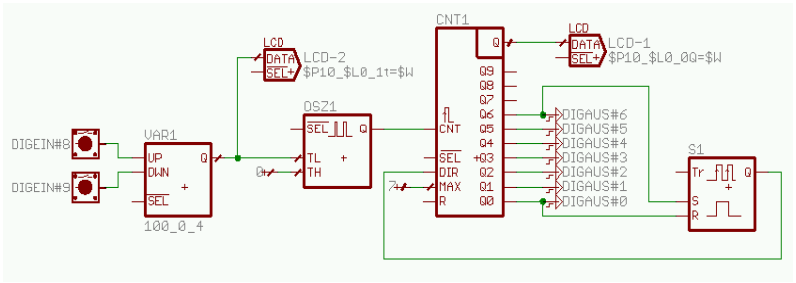
Display2:
 $P10_$L0_1t=$W
Genau wie oben; mit dem Unterschied, dass der Wert in Zeile 1 angezeigt wird (untere Zeile) und die Zeile mit „t=“ beginnt.

Die verschiedenen Parameter des LCDs können in dem Dokument „Handbuch zur MicroSPS“ nachgelesen werden.

Test:

Wir übersetzen das Projekt mit F12 und spielen es ein. Nun können wir die TL-Zeit und den aktuellen Zählerstand am Display ablesen.

Knight Rider:



Das Lauflicht soll beim Erreichen der oberen LED rückwärts zurücklaufen. Dazu muss bei Erreichen des obersten Zustandes die Richtung umgeschaltet werden. Wir verwenden dafür ein RS-Flipflop, das wir unter der Bezeichnung: „STROMSTOSS-REL“ in der Bibliotek finden. Der Eingang S (Set) setzt den Ausgang des Flipflops, der so lange gesetzt bleibt, bis der Eingang R (Reset) gesetzt wird. Den Ausgang des FlipFlops verbinden wir mit dem Richtungseingang DIR unseres Dekadenzählers.

Test:

Wir übersetzen das Projekt mit F12 und spielen es ein. Das Lauflicht wechselt nun nach Erreichen des oberen bzw. unteren Ausgangs die Richtung. Die Geschwindigkeit und der Zählerstand lassen sich am LC-Display ablesen. Die Geschwindigkeit kann über Tasten verändert werden.