

---

## **Noch einmal – Servos steuern: Die Servo-Erweiterungsmodule der Firma Wilke Technology**

Gunther Zielosko

### **1. Grundlagen**

Im Applikationsbericht 3 haben wir das Thema Servosteuerung schon einmal behandelt, aus heutiger Sicht eine eher singuläre und umständliche Lösung. Besser und einfacher geht es nun mit speziellen Bausteinen der Firma Wilke. Die Erweiterungsmodule EP16 und EP17, die 8 oder 16 Servo-Ausgänge sowie 24 Digi-Scan Eingänge mitbringen, sind wie geschaffen für allerlei motorische Antriebe zunächst in der Industrie, aber auch für Experimente mit Robotern, in Modellbahnanlagen, Kamerasteuerungen, in Fahrzeugmodellen aller Art usw. Während Servos normalerweise im Modellbau per Funksteuerung über eine Art Pulsweitenmodulation „analog“ angesteuert werden, geschieht dies in den EP16- und EP17-Modulen rein digital über eine serielle V24-Schnittstelle. Damit steht einer Nutzung der Module innerhalb des BASIC-Tiger<sup>®</sup> Systems aber auch in anderen Systemen mit RS232-Schnittstelle nichts mehr im Wege. Jeder PC kann dann unmittelbar bis zu 16 Servos direkt ansteuern. Interessiert? Dann lesen Sie hier weiter.

### **2. Die Bausteine EP16-Servo8 und EP17-Servo16**

Die Erweiterungsbausteine stecken wie üblich in einem vergossenen BASIC-Tiger<sup>®</sup>-Gehäuse und haben 92 Anschlüsse. Neben der Servo-Ansteuerung gibt es noch einen nahezu unabhängigen Digi-Scan-Teil mit 24 Eingängen, deren logischer Zustand über das Bussystem eingelesen werden kann. Eine willkommene Ergänzung für jene, die zum Beispiel in einem Modell noch Schalter, Sensoren oder andere Kontakte abfragen wollen. Dieser Teil liegt für uns etwas am Rande des Interesses und wird deshalb hier nicht behandelt.

Wenn man sich den Servo-Teil anschaut, findet man bis zu 16 Servoausgänge Servo-0 bis Servo-15 und insgesamt vier Steuerpins, die RxD\_TTL, CTS\_TTL, RxD\_V24 und CTS\_V24 genannt werden. Für unsere Experimente würden wir eigentlich nur die beiden V24-Steuerpins benötigen, da diese ihrer Bezeichnung nach kompatibel zu den COM-Schnittstellen des PC und zum Plug-and-Play-Lab sein sollten (Pegel -3...-15 V und +3...+15V). Leider ist diese Angabe im Tiger-Pocket-Guide aber falsch. Zwar verträgt der Dateneingang RxD\_V24 diese Pegel und wertet sie richtig aus, der sendende Ausgang CTS\_V24 liefert dagegen nur TTL-Pegel (allerdings richtig negiert zu CTS\_TTL). Beim Betrieb mit dem BASIC-Tiger<sup>®</sup> scheint das zwar trotzdem zu funktionieren, mit einem PC-Programm aber nicht. Der CTS-Ausgang des Servo-Teiles erlaubt nämlich nur zu bestimmten Zeiten innerhalb eines Zyklus einen Datentransfer, bei falscher Pegellage weigert sich die COM-Schnittstelle des PC, überhaupt Daten zu senden. Sicherer ist es daher, die TTL-Anschlüsse des EP-Servo-Bausteines zu nutzen und mit einem externen MAX232 für klare Pegel zu sorgen.

Zum Schluß ist natürlich wichtig, Masse (GND) und Betriebsspannung ( $V_{cc} = +5V$ ) am Modul anzuschließen. Vergessen Sie nicht, GND des Moduls mit GND der seriellen Schnittstelle (Pin 5 am SUB-D-Stecker) und GND des Servos zu verbinden. Für den Anschluß des Servos braucht man insgesamt 3 Leitungen, nämlich GND, +5V (möglichst aus einer unabhängigen Quelle) und das Servo-Steuer-Signal. Die genaue Beschaltung aller Komponenten wird später vorgestellt.

Das ist die Hardware – einfacher geht's wirklich nicht. Ist es mit der Software genauso einfach? Leider nicht – die Aufbereitung eines Befehles für den Servo-Baustein ist doch recht kompliziert und erklärungsbedürftig. Wenn Sie den BASIC-Tiger® als Steuercomputer benutzen wollen, lesen Sie bitte die Erklärungen im Abschnitt 6, dort gibt es auch Beispielprogramme für verschiedene Anwendungen. Für den Fall, daß der PC Servo-Motoren steuern soll, gibt es ein komfortables Windows-Programm, mit dem Sie Zugriff auf immerhin 8 Servos haben.

### 3. Aufbereitung der Steuerkommandos

Wir wissen von früher her, daß Modellbau-Servos ihre Position über eine Pulslängensteuerung einstellen (0,552 ms bis 2,596 ms). Bei handelsüblichen Fernsteueranlagen erfolgt das analog über Monoflops. Beim EP-Servomodul wird die Impulslänge für jedes Servo von einem eingebauten Prozessor eingestellt und im Abstand von 20 ms immer wieder aktualisiert. Die Einstellung der Daten erfolgt über eine serielle Schnittstelle. Eine einzige RS232-Schnittstelle steuert bis zu 16 Servos. Das ist möglich, indem jedes Servo quasi eine Adresse hat, so z.B. Servo 0 = 00h, Servo 1 = 01h...und Servo 15 = 0Fh. Jeder Steuerbefehl enthält neben den eigentlichen Positionierungsdaten die Servo-Adresse des einzustellenden Servos. Die Positionierungsdaten selbst sind 10Bit-Werte, das Servo kann also (theoretisch) von der Stelle 0 (z.B. Linksanschlag) bis zur Stelle 1023 (z.B. Rechtsanschlag) gefahren werden, eine sehr feinfühligste Steuerung. Eine Besonderheit gibt es hier im Gegensatz zur analogen Servoansteuerung – die Stellung 1023 ist nicht erreichbar, hier hat man eine Möglichkeit, das Servo stromlos zu schalten, also z.B. Strom zu sparen oder das Servo von Hand bewegbar zu machen. Praktisch gibt es also die Positionen 0 bis 1022, denen jeweils analoge Impulslängen für das Servo von 0,552 ms bis 2,596 ms entsprechen (Bild 1).

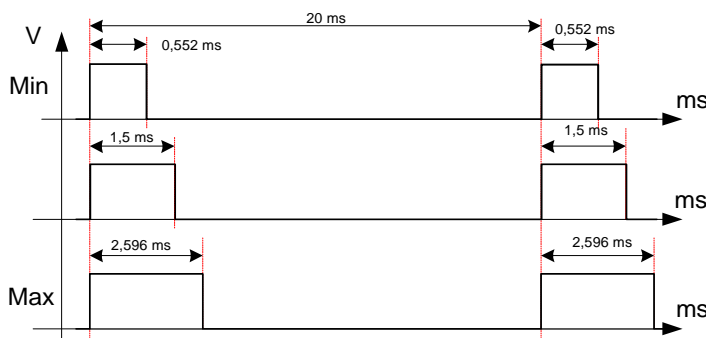


Bild 1 Impulsschema eines Servoausganges (Positionswerte  $Min = 0$ ,  $Max = 1022$ )

Eine Kommando-Sequenz besteht aus zwei Bytes und enthält, wie schon erklärt, die Adresse des Servos und dessen neue Position. Das niederwertige Byte (Low-Byte) wird immer zuerst übertragen. Wie die Daten angeordnet sind, zeigt die Tabelle 1. Die Servo-Adressen werden blau und die Positionsdaten violett dargestellt: Die unteren Zeilen enthalten Beispielwerte, die im Anschluß erklärt werden.

Byte	High-Byte								Low-Byte							
Daten Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Position Bit						9	8	7		6	5	4	3	2	1	0
Servo-Adr. Bit		3	2	1	0											
Beispiel FF 07	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
Beispiel D2 0E	0	0	0	0	1	1	1	0	1	1	0	1	0	0	1	0
Beispiel 96 11	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0

Tabelle 1 Verteilung der Daten- und Adreßbits

Das Bit 7 des Low-Bytes wird immer auf „1“, das Bit 7 des High-Bytes immer auf „0“ gesetzt. Damit vermindert man die Möglichkeit von Fehleinstellungen bei „verstümmelten“ Daten oder anderen Übertragungsproblemen. Der 10-Bit-Einstellwert des Servos verteilt sich auf beide Bytes. Die Bits 3 bis 6 des High-Bytes enthalten die Servoadresse. Unsere 3 Beispiele erklären sich dann so:

```
FF 07      Servo 0 auf Position 1111111111 (dez. 1023 = ausgeschaltet)
D2 0E      Servo 1 auf Position 1101010010 (dez. 850)
96 11      Servo 2 auf Position 0010010110 (dez. 150)
```

Alles klar?

Um die Servopositionen zu aktualisieren, müssen übrigens nicht immer alle Daten neu aufbereitet und übertragen werden. Es genügt, das jeweils zu verstellende Servo neu mit Daten zu versorgen, die anderen Servos bleiben solange stabil in ihrer Stellung stehen. Für die serielle Datenübertragung müssen folgende Parameter eingestellt werden:

```
Baudrate:   38400 Bd
Datenbits:   8
Parity:      No
```

Es soll hier nochmals erwähnt werden, daß die serielle Schnittstelle des Modules nicht zu jeder Zeit Daten empfängt, dazu ist sie nur während 14,2 ms in jedem 20 ms - Zyklus bereit. Die Steuerung erfolgt über das CTS-Signal, der BASIC-Tiger<sup>®</sup> bzw. der PC sollten also immer auch an diesen Pin angeschlossen sein. Das intelligente Zeitmanagement des BASIC-Tigers<sup>®</sup> und auch entsprechende PC-Programme haben Datenpuffer und regeln das meist ordentlich.

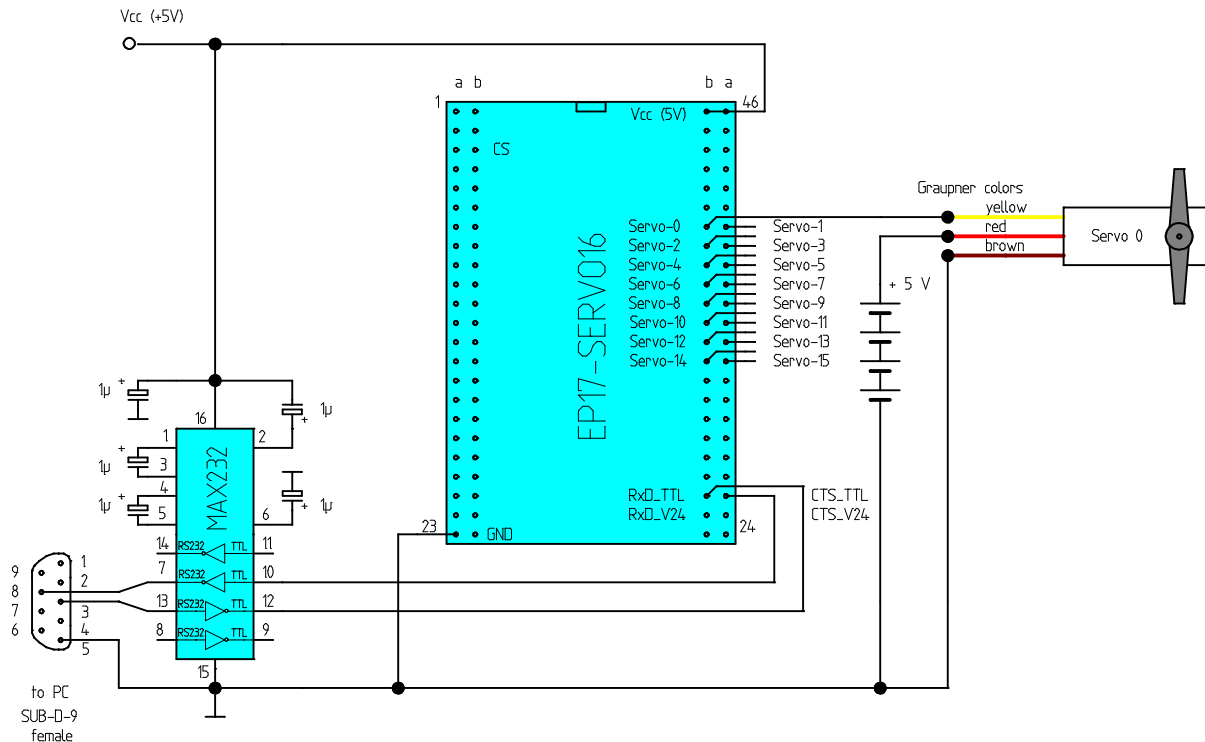
---

#### **4. Das Problem der mechanischen Servobegrenzung**

Normale Modellbau-Servos werden vom Sender her mit Monoflops gesteuert, die ihrerseits mit einem Widerstand (Steuerknüppel, Schieberegler oder Drehknöpfe) in ihrer Ausgangsimpulslänge eingestellt werden. In handelsüblichen Fernsteuereinrichtungen mit Sender, Empfänger und Servo-Elektronik werden dabei üblicherweise nicht die laut Definition maximal möglichen Ausschläge realisiert. Bei der rein digitalen ImpulslängeEinstellung des EP-Servo-Modules ist sehr wohl eine minimale oder maximale Impulslänge möglich. Stellt man diese Extremwerte ein, kann es dazu kommen, daß das Servo bestrebt ist, weiter als bis zu den mechanischen Anschlägen zu fahren. Das kann der Motor zwar nicht, es fließt aber ein erheblicher Strom und das Servo könnte beschädigt werden. Das heißt, der Anwender muß irgendwie dafür sorgen, daß Maximalausschläge nur so groß werden, daß das Servo den Endausschlag noch nicht erreicht. Das gilt sowohl für BASIC-Tiger®- als auch für PC-basierende Programme. Entweder man stellt die Maxima und Minima der Impulsängen so ein, daß mit Sicherheit keines der verwendeten Servos in diese mißliche Lage kommt oder man sorgt für eine automatische Kalibrierung. Wege dazu werden in den folgenden Kapiteln gezeigt.

#### **5. Servo-Steuerung mit dem PC und das Programm „SERVO01.TST“**

Um den Servo-Baustein am PC zu betreiben, ist die Schaltung nach Bild 2 erforderlich. Von der Com-Schnittstelle am PC (meist ein 9-poliger SUB-D-Stecker) geht es über ein Kabel mit 9-poliger SUB-D-Buchse zum Pegelwandler MAX232, der aus den RS232-Signalen TTL-Signale herstellt. Diese beiden Signale und GND übernehmen die Steuerung des Servo-Teiles des Erweiterungsmodules. Von dort aus gehen nur noch die einzelnen Servo-Steuerleitungen an die einzelnen Servos. Wenn Sie Servos der Firma Graupner verwenden, gelten die Farbcodes des Schaltbildes. Ansonsten müssen Sie sich informieren, wie Ihr Servo angeschlossen wird.



*Bild 2 Schaltung EP17-Servo16 für den Betrieb mit einer COM-Schnittstelle des PC*

Nun kann ein einfaches PC-Programm über die COM-Schnittstelle viele Servos in ihrer Position einstellen. Einfaches PC-Programm ist leichter gesagt als getan. Ein solches Programm, natürlich unter Windows, muß zunächst in der Lage sein, irgendwie komfortabel die Einstellwerte für die Servos einzustellen. Diese müssen dann in die Form gebracht werden, die wir im Kapitel 3 dargestellt haben. Abschließend müssen sie in der Form eines 2 Byte-Wortes (nicht etwa als ASCII-Zeichen!) direkt an eine entsprechend voreingestellte COM-Schnittstelle gesendet werden. Für Nicht-Windows-Programmierer schon recht kompliziert. Um die Anwendung der EP-Servo-Module für diese Zeitgenossen ein wenig zu erleichtern, wird hier wieder ein komfortables Windows-Programm auf der Basis von TestPoint von Keithley mitgeliefert, das Sie nur noch installieren müssen. Das Programm **SERVO01.TST** als Runtime-Modul zeigt sich nach der Installation wie folgt:

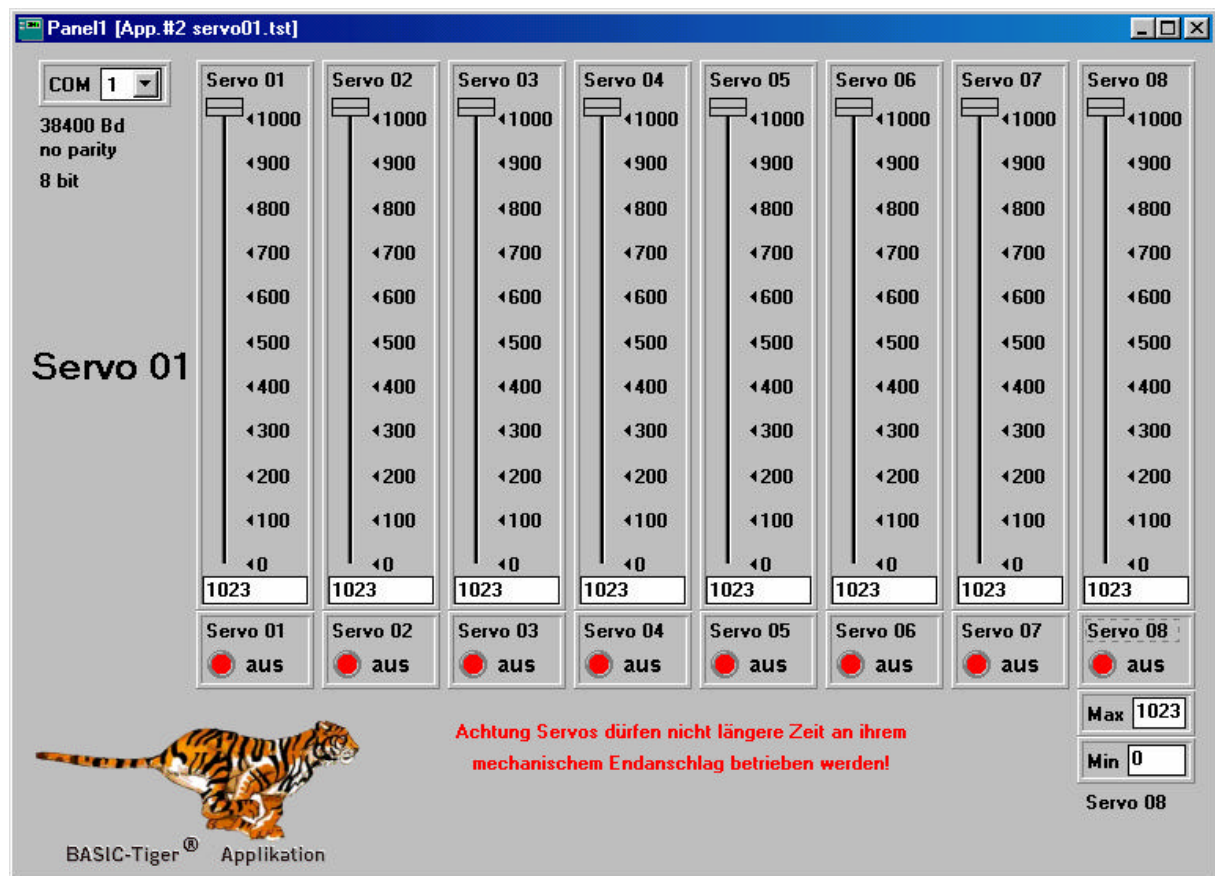


Bild 3 Das Programm SERVO01.TST zur Positionierung von bis zu 8 Servos über den PC

Ein Programm wie SERVO01.TST kann natürlich nicht alle denkbaren Anwendungen für Servos abdecken, dennoch bietet es einige Möglichkeiten zum Experimentieren.

Sie sehen 8 Schieberegler, mit denen Sie komfortabel jedes Servo unabhängig einstellen können. Die Position können Sie auf drei verschiedenen Wegen einstellen:

- einfach mit der Maus (Mauszeiger auf den Knopf, links anklicken und schieben)
- gewünschten Regler mit der linken Maustaste anklicken und mit den Tastatur-Pfeiltasten nach oben oder unten fein regulieren (Einzelschritte möglich)
- gewünschten Regler mit der linken Maustaste anklicken und Wert direkt in das Schriftfeld eintragen

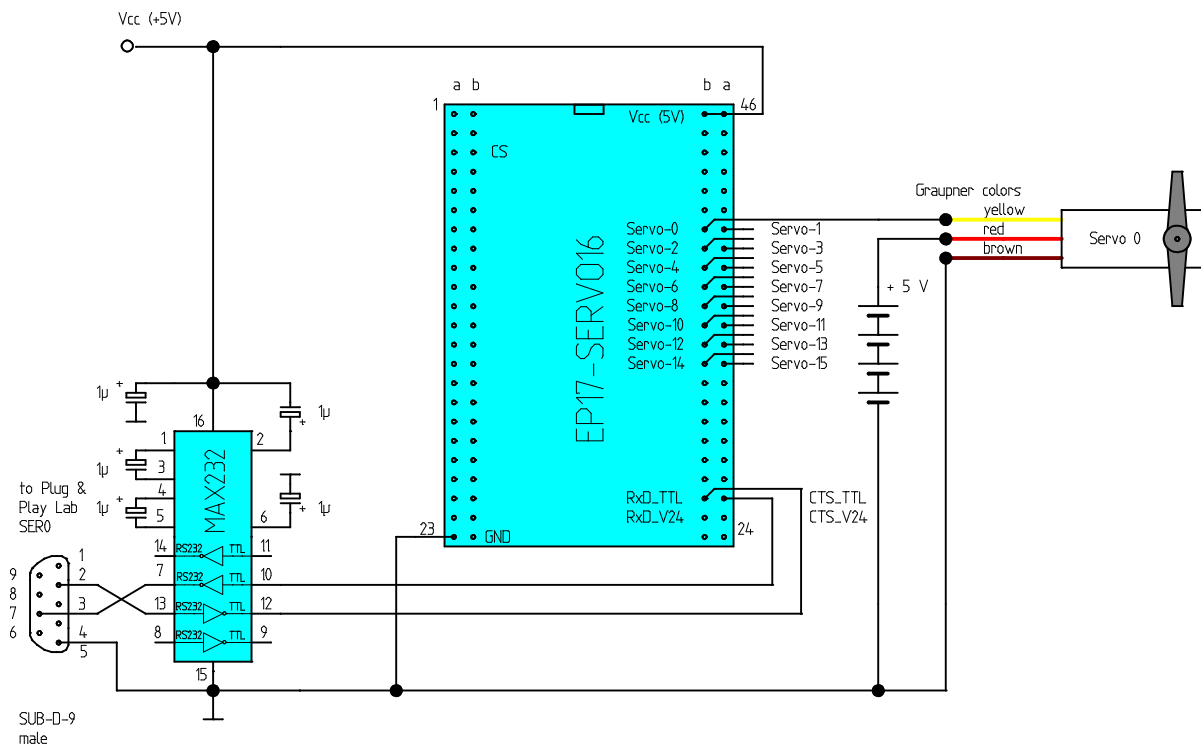
Beachten Sie bei allen Experimenten, daß Sie kein Servo auf Dauer über seine mechanische Endstellung hinaus fahren, in der es einen großen Strom zieht und zerstört werden kann (Abschnitt 4)!

Der Einstellwert 1023 ist, wie schon erwähnt, kein echter Positionswert, sondern eine Sonderfunktion zum kompletten Ausschalten des Servos. Sie können ein wirkliches Abschalten dadurch erreichen, daß Sie es in die obere Endstellung (1023) fahren. Ein aktives Servo wird im Programmfenster mit einer grünen, ein ausgeschaltetes Servo mit einer roten „LED“ signalisiert.

Unter dem Schieberegler für Servo 08 sehen Sie noch zwei Eingabefelder, hier können Sie mit der Maximal- und Minimalstellung (nur bei diesem Servo!) experimentieren. Am Anfang hat Servo 08 dieselben Extremwerte, d.h. 0 und 1023, wie die anderen. Wenn Sie das Servo in die Nähe der Extremwerte fahren, werden Sie feststellen, daß es irgendwo nicht mehr weitergeht. Alternativ können Sie einen Strommesser in die Stromversorgungsleitung für die Servos schalten und den Strom messen. Kommt das Servo in eine mechanische Endstellung, fließt ein erheblicher Dauerstrom. Tragen Sie nun Maximum und Minimum so ein, daß das Servo nicht mehr in diese „gefährlichen“ Bereiche gelangt. Da die Impulszeit bei den Servomodulen digital und damit hochgenau eingestellt wird, gilt der ermittelte Bereich für das betreffende Servo automatisch für alle Einstellregler.

## 6. Servo-Steuerung mit dem BASIC-Tiger<sup>®</sup>

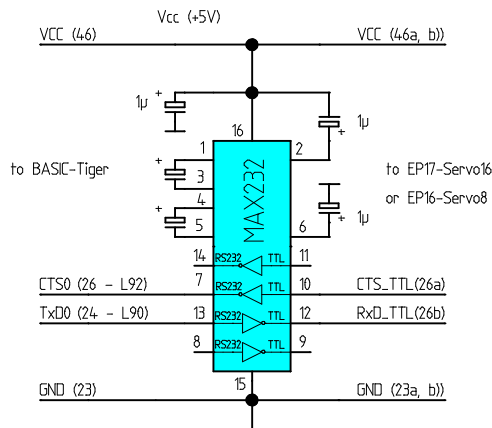
Die Erweiterungsbausteine EP16-Servo8 und EP17-Servo16 wurden natürlich für den Betrieb mit dem BASIC-Tiger<sup>®</sup> entwickelt, auch hier zunächst die Schaltung:



*Bild 4 Schaltung EP17-Servo16 für den Betrieb mit dem Plug and Play Lab (SER0)*

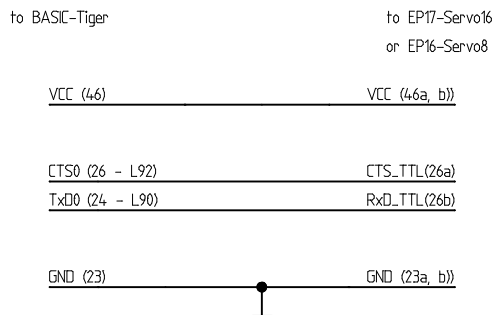
Sie sehen, daß beim Betrieb mit dem Plug and Play Lab ein anderer Steckverbinder erforderlich ist – hier ein 9-poliger SUB-D-Stecker. Außerdem ändert sich dessen Verdrahtung gegenüber der Variante am PC. Beachten Sie, daß wegen des erforderlichen CTS-Anschlusses nur die serielle Schnittstelle SER0 sicher funktionieren kann, da nur diese einen CTS (Clear to Send)-Anschluß hat.

Im Übrigen arbeitet das Plug and Play Lab immer mit RS232-Pegeln – egal, was für einen BASIC-Tiger<sup>®</sup> Sie einsetzen. Für eigenständige Lösungen mit einem BASIC-Tiger<sup>®</sup> müssen Sie etwas mehr überlegen. Für (die teureren) Tiger mit eingebauter RS232-Schnittstelle müssen Sie wieder einen MAX232 opfern. Bei sonst gleicher Schaltung wie im Bild 4 werden im Bild 5 nur die Anschlüsse zum BASIC-Tiger<sup>®</sup> dargestellt.



*Bild 5 Direktanschaltung an BASIC-Tiger<sup>®</sup> mit eingebauter RS232-Schnittstelle*

Tiger ohne RS232-Schnittstelle sind billiger und man spart gleich noch einen MAX232. Bild 6 zeigt die Unterschiede, auch hier wird wieder nur die Anbindung des Tigers dargestellt.



*Bild 6 Direktanschaltung an BASIC-Tiger<sup>®</sup> ohne eingebaute RS232-Schnittstelle*

Soweit zur „Hardware“, die im letzten Falle auf ein paar Drähte reduziert ist.

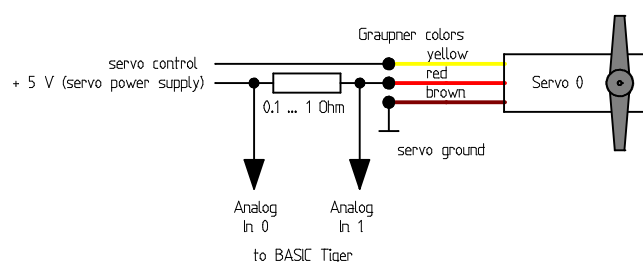
Auf der Seite der BASIC-Tiger<sup>®</sup> Software stehen den meisten von uns, anders als bei der Programmierung unter Windows am PC, wieder nahezu grenzenlose Möglichkeiten offen. Fangen wir mit einem einfachen Testprogramm an.

Das Programm **SERVO01.TIG** steuert alle theoretisch möglichen 16 Servos gleichermaßen an. Dies geschieht in einer Schleife, d.h. die Servo-Adresse (Variable „devAdr“) wird von 0 bis zur höchsten eingestellten Adresse (Variable „LASTSERVO“) durchgezählt. Tragen Sie bei #DEFINE LASTSERVO 0 ein, wird nur ein Servo (Servo 0) angesteuert – in der vorliegenden Form von SERVO01.TIG ist dies so eingestellt. Zunächst wird für alle Servos die Position 1023, d.h. die „Aus“-Stellung eingestellt. Kein Servo ist in Betrieb und kein



Motorstrom fließt. Danach werden alle Servos in eine sehr hohe Position (MAX) gefahren, hier 700, um den gefährlichen Betrieb am mechanischen Anschlag sicher zu vermeiden. Nach einer kurzen Verweilzeit wird eine kleine Position (MIN) angesteuert, die hier mit 100 eingestellt ist. Am Ende werden alle Servos mit dem Wert 1023 wieder ausgeschaltet. Das Ganze wiederholt sich für alle Servos – ein toller Dauertest mit der Option, z.B. die Wiederholgenauigkeit der Einstellwerte zu testen. Nebenbei sehen Sie auf dem LC-Display: die Servo-Adresse (dez.), die aktuelle Position (dez.) und das aus beiden Werten umgerechnete Kommando-Wort (hex., Hier High-Byte zuerst!).

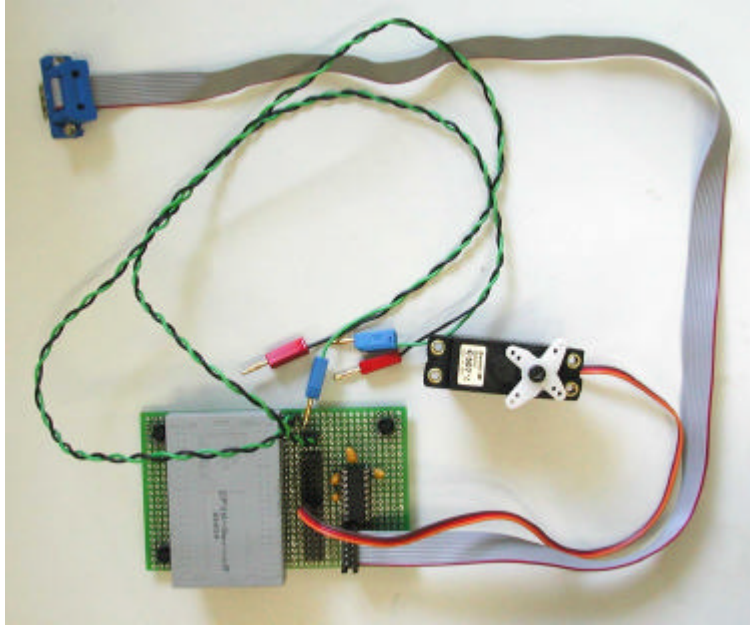
Das Programm **SERVO02.TIG** dient zur automatischen Feststellung der mechanischen Servo-Endpunkte. Wir haben ja erkannt, daß ein Servo mit seiner ganzen Kraft versucht, eine der Steuer-Impuls-Länge entsprechende Position einzunehmen. Kommt es dabei an den mechanischen Anschlag, erhöht sich der Servostrom erheblich. Diesen Effekt nutzen wir aus. SERVO02.TIG geht zunächst von den theoretisch erreichbaren Positionen 0 bis 1022 aus (MIN, MAX). Von der rechnerischen Mittelstellung 512 aus erfolgt eine langsame Erhöhung des Positionswertes. Beim Erreichen des mechanischen Anschlages steigt der Strom in das Servo an. Wir messen das, indem wir einfach in die Zuleitung an die separate +5 V-Quelle für das Servo einen Widerstand von 0,1  $\Omega$  bis 1  $\Omega$  einbauen. Mit zwei Leitungen greifen wir die Spannung über dem Widerstand ab und messen diese mit zwei Analogeingängen AD 0 und AD 1 unseres BASIC-Tigers® (Bild 7). Legen Sie dazu beim BASIC-Tiger® Analog GND (Pin 44) an GND (Pin 23) und A/D Ref-volt-in (Pin 43) an Vcc (Pin 46) Ein Maß für den Strom ist die Spannungs-Differenz zwischen beiden Widerstandsenden gemäß  $I=(U_1-U_2)/R$ . Der BASIC-Tiger® errechnet dies und bricht die Bewegung beim Erreichen eines vorgegebenen Strom-Grenzwertes ab. Er speichert den ermittelten Maximalwert (MAXn). Natürlich ist der ermittelte Wert für den Strom kein echter Stromwert, die Spannungsdifferenz ist ja auch nur als Stromsprung interessant (Endwert erreicht). Dasselbe Spiel beginnt zur Ermittlung des Minimums. Sind beide Extremwerte erreicht, werden das neue „reale“ Maximum und Minimum für das individuelle Servo angezeigt. Abschließend bewegt sich das Servo zwischen diesen beiden Endpunkten hin und her.



*Bild 7 Servo-Strommessung*

Ein tolles, selbst lernendes Servo, nicht wahr? Wenn Sie wollen, können Sie das der Übersichtlichkeit wegen nur auf ein Servo (Servo 0) beschränkte Programm auf alle möglichen Kanäle erweitern und so eine Kalibrierung aller angeschlossenen Servos erreichen...

Zum Abschluß noch einmal der gesamte Laboraufbau des Autors (Bild 8).



*Bild 8 Laboraufbau mit EP16-Servo8, MAX232, Servo, seriellem Kabel und Stromleitungen*

Viel Spaß mit Ihren Servos!