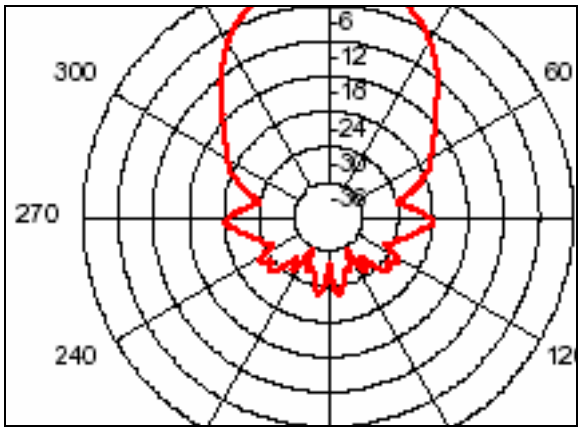
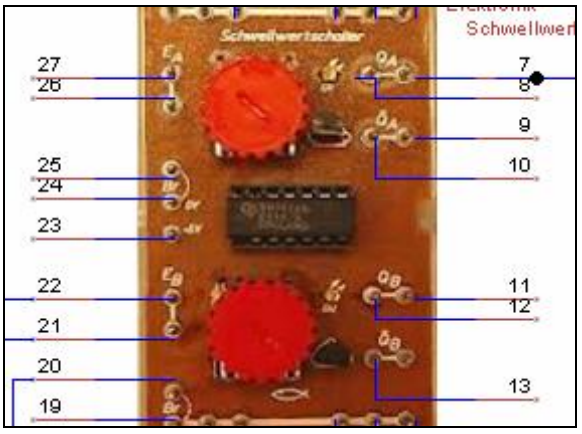
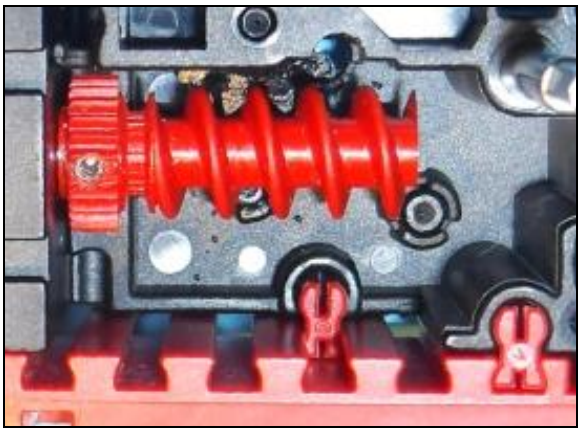
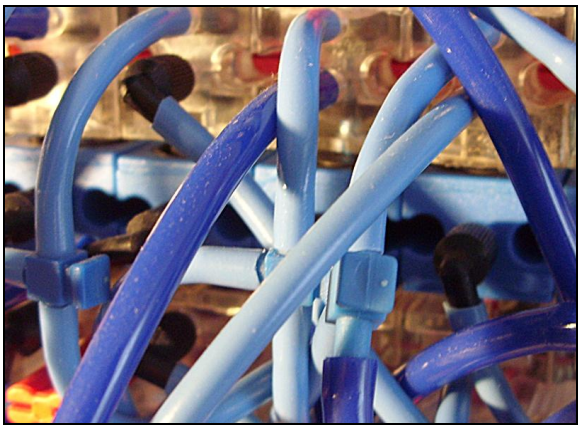
ft:pedia

Heft 4/2013









Herausgegeben von
Dirk Fox und Stefan Falk

ISSN 2192-5879

Editorial

Ausgedünnt

Dirk Fox, Stefan Falk

Wer sich über Jahre mit fischertechnik beschäftigt, hat nicht nur die aktuellen Kästen in seiner Sammlung. Sondern auch kleine oder größere „Konvolute“, ersteigert bei technikfernen Dachbodenaufräumern, die Schätze von Papi (oder aus der eigenen Jugend) und das eine oder andere erworbene Einzelteil.

Solcherart weitgehend in die Abwesenheit von Mangel gebettet, fällt es schwer, sich die Nöte eines ft-Neulings vorzustellen. Die sind aber tatsächlich erheblich – und sollten dem fischertechnik-Team vielleicht einmal eine gründliche Analyse wert sein.

Viele der aktuellen, auch der – völlig zu recht – gelobten Baukästen erlauben vor allem den Nachbau der in der Anleitung vorgeschlagenen Modelle. Zum Teil allerdings wenig mehr. Selbst die Kombination verschiedener Kästen eröffnet nur einen Ausschnitt der Möglichkeiten des fischertechnik-Systems. Denn einige der für ein „universelles“ Bauen und Konstruieren wesentlichen Teile sind nur noch in wenigen Kästen und in kleiner Anzahl verfügbar. Fünf prominente Beispiele:

- **Statik-Bogenstücke** gibt es nur noch in der 60°-Version im Kasten Universal 3 (sechs) – sogar das Riesenrad des Super Fun Park muss ohne sie auskommen.
- Ein **Kardan-Gelenk** für die Rastachsen gibt es nur noch im Motor Set XS.
- **Differentiale** sucht man im Kasten Mechanic + Static und in Cars & Drives vergebens – sie sind ausschließlich mit dem Motor Set XM (eins) erhältlich. Selbst die Computing-Kästen mit Encodermotor enthalten keines.
- **Kettenglieder** fanden sich lange nur als Raupenantrieb im Kasten XL Bulldozer. Als Transportkette gibt es sie im Kasten Dynamic, aber nicht im Universal 3 oder im Mechanic + Static.
- **Statik-Laufschienen** gibt es nur noch im Kasten Super Cranes (drei) – dabei sind sie für die Konstruktion von Hafenkränen, Containerbrücken und Schienenfahrzeugen aller Art elementar.

Selbst in der Creative Box ist außer Kettengliedern keines der aufgeführten Bauteile enthalten. Und weitere haben sich komplett aus dem Sortiment verabschiedet, wie z. B. die wunderbaren roten S-Platten (wie will man ohne sie Container oder Brücken mit Fahrweg bauen?).

Beim Versuch, dem Lego-Erfolg mit Modellkästen nachzueifern, sollte das Alleinstellungsmerkmal von fischertechnik nicht aus dem Blick geraten: die Universalität des Systems, mit dem sich nicht nur Modelle nachbauen, sondern vor allem eigene technische Ideen umsetzen lassen. Nur wer diese Universalität versteht wird auch beginnen, sein fischertechnik-Sortiment gezielt zu vergrößern. Vielleicht ist es an der Zeit, das Bauteilsortiment der fischertechnik-Kästen einmal didaktisch „gegenzubürsten“?

Schöne Weihnachten, Euer ft:pedia-Team.

P.S.: Am einfachsten erreicht ihr uns unter ftpedia@ftcommunity.de oder über die Rubrik *ft:pedia* im Forum der ft-Community.

Inhalt

Ausgedünnt 2
Mini-Modelle (Teil 1): Gabelstapler 4
Perlentauchen (Teil 5) 6
ft-Spezialteile made by TST (Teil 6).....16
Automatik zur Steuerung eines Krans 18
I²C mit dem TX – Teil 7: Real Time Clock (RTC).....28
I²C mit dem TX – Teil 8: Ultraschall-Sensor.....35

Termine

Was?	Wann?	Wo?
Spielwarenmesse 2014	29.01.-03.02.2014	Nürnberg Convention Center
RoboCup German Open 2014	03.-05.04.2014	Messe Magdeburg
fischertechnik Convention 2014	27.09.2014	Erbes-Büdesheim

Hinweise

Mit dem Umzug des ft-community-Servers auf ein neues System werden möglicherweise einige Download-Links aus bisherigen ft:pedia-Ausgaben nicht mehr funktionieren. Die Dateien gibt es aber weiterhin – erreichbar über das Portal der ft:c.

Impressum

http://www.ftcommunity.de/ftpedia

Herausgeber: Dirk Fox, Ettlinger Straße 12-14, 76137 Karlsruhe und Stefan Falk, Siemensstraße 20, 76275 Ettlingen

Autoren: Stefan Falk (steffalk), Dirk Fox (Dirk Fox), Werner Hasselberg, Andreas Tacke (TST), René Trapp (H.A.R.R.Y.).

Modell

Mini-Modelle (Teil 1): Gabelstapler

René Trapp

Erinnert ihr euch an die Mini-Modelle der fischertechnik-„GiveAways“, wie die Straßenwalze [1] oder der Oldtimer [2]? In einer kleinen Serie werden wir weitere solcher charmanten Kleinstmodelle vorstellen. Den Anfang macht ein Gabelstapler im GiveAway-Format.

Ein fischertechnik-Gabelstapler aus lediglich 12 Bauteilen? Kaum zu glauben, aber das geht tatsächlich. Abb. 1 zeigt das fertige Arbeitsgerät: Nur noch mal vom Profi fotografieren lassen und schon kann es in den Verkaufskatalog.

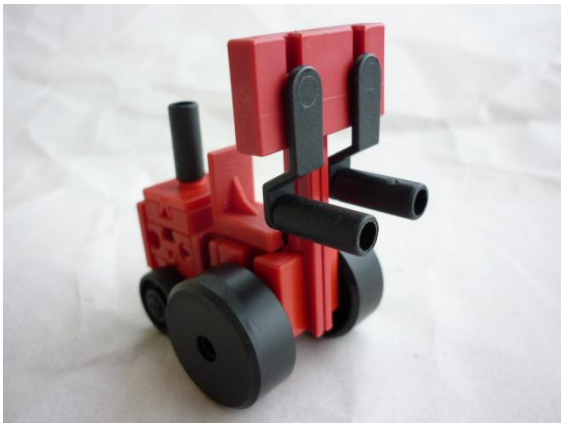


Abb. 1: Der Mini-Gabelstapler

In der Seitenansicht (Abb. 2) sieht man schon fast alle interessanten Verbindungen.

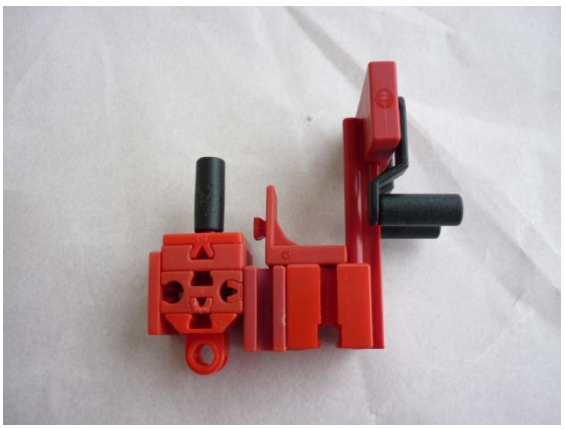


Abb. 2: Fahrgestell ohne Räder und Achsen

Aus den Einzelteilen in Abb. 3 besteht die Hubgabel; das Auspuffrohr hat sich auch noch dazwischengemogelt.

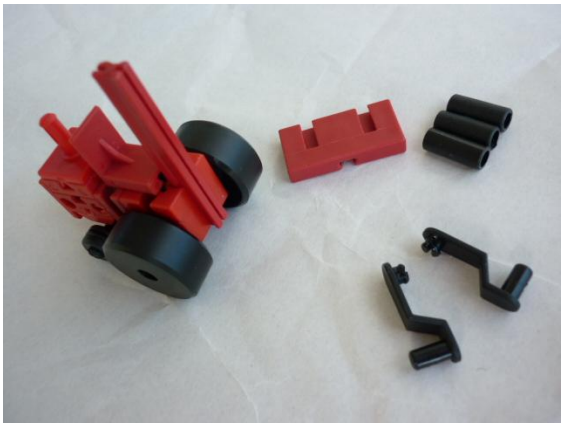


Abb. 3: Einzelteile der Hubgabel

Es erfordert ein bisschen Fummelei, die Schalthebel (31994) in die Bauplatte (38428) einzuschieben, aber es geht zerstörungsfrei.

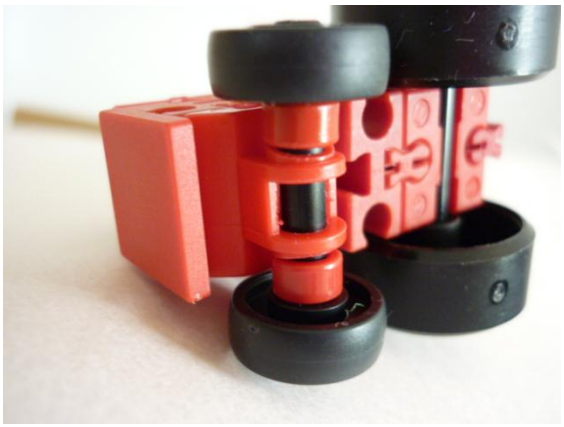


Abb. 4: Der Unterboden

Ein Blick auf die Unterseite enthüllt auch noch die vorletzten Geheimnisse des Zusammenbaus (Abb. 4).

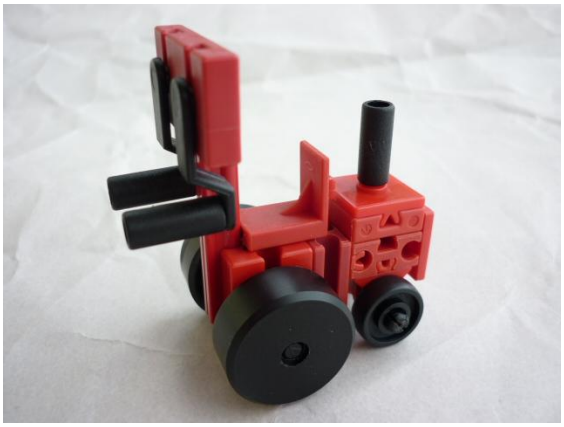


Abb. 5: Seitenansicht

Und zum Schluss noch eine Perspektive die die Position der verwendeten Achsen gut erkennen lässt (Abb. 5).

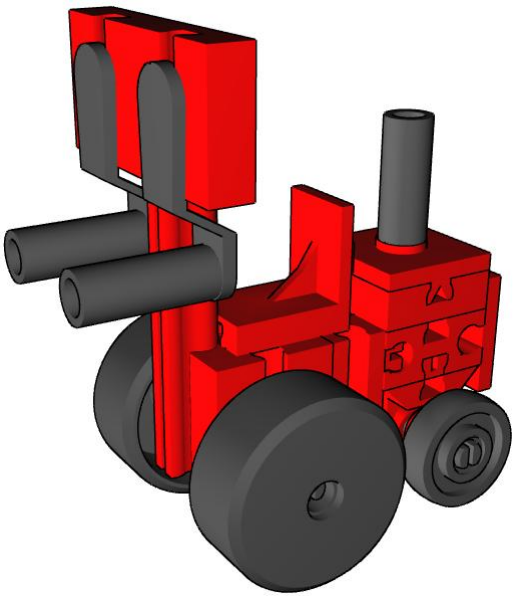
Schließlich darf die Stückliste nicht fehlen:

Stück	ft-Nr.	Bezeichnung
2	31994	Schalthebel
1	38428	Bauplatte 15x15 mit 3 Nuten
3	36819	Lagerhülse
1	31330	Verbindungsstück 45
1	116252	Baustein 15, rot
1	38423	Winkelstein 10x15x15
1	31982	Verbindungsstück 15
2	37468	Baustein 7,5
2	31982	Federnocken
1	31426	Gelenkwürfel Zunge
1	38246	Bauplatte 15x15, 1Z
1	37237	Baustein 5
1	31124	Radachse mit Platte
2	31597	Abstandsring
2	36573	Rad 14
1	36919	V-Achse 28
1	38413	Kunststoffachse 30
2	36574	Rad 23, schwarz

Quellen

[1] Fischer-Werke: *GiveAway Straßen-walze*. Waldachtal, 1990.

[2] Fischer-Werke: *GiveAway Oldtimer*. Waldachtal, 2009.



fischertechnik-Basiswissen

Perlentauchen (Teil 5)

Stefan Falk

In diesem Teil der Reihe tauchen wir ab in die faszinierende Welt der fischertechnik-Pneumatik, stellen dar, was es schon alles gab, wie sich die Dinge weiterentwickelt haben – und was davon heute noch erhältlich ist.

Wo kommt die Druckluft her?

Die fischertechnik-Pneumatik, also das Bauen von mit Druckluft *betriebenen* und auch damit *gesteuerten* Maschinen, begann schon 1981 mit den gleichnamigen Ergänzungs-Baukästen:

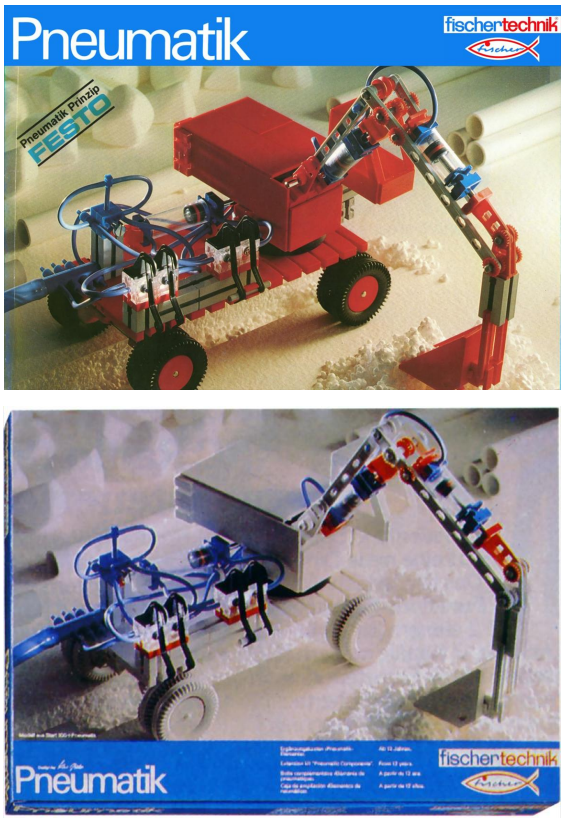


Abb. 1: Die ersten Pneumatik-Kästen

Diese Kästen enthielen neben Pneumatik-Zylindern auch verschiedene Ventile, auf die wir gleich noch zu sprechen kommen.

Allerdings war keine Druckluftquelle Bestandteil des Baukastens, sondern es standen separat zwei Varianten von mit 220 V betriebenen fischertechnik-Kompressoren zur Auswahl:

Abb. 2: Die großen fischertechnik-Kompressoren

Es gab sogar eine 30865 Handpumpe, mit der man ebenfalls ganz ordentlich Druckluft für die Modelle erzeugen konnte (Abb. 3). Diese großen Druckluftquellen gaben ihre Druckluft in Schläuche mit ca. 1 cm Durchmesser.

Abb. 3: Druckluft-Handpumpe

Die passten auf den *Druckluftverteiler* (Abb. 4), an dem man dann bis zu acht fischertechnik-Pneumatikschläuche direkt anschließen konnte. Die unbenutzten Anschlüsse verschloss man mit den später noch aufzuführenden *Stopfen*.

Abb. 4: Druckluft-Verteiler

Um auch mit geringeren Kosten, weniger groß und weniger laut zumindest ein bisschen Druckluft bereitstellen zu können, gab es zwei verschiedene Kurbel-Anbauteile für den damals noch aktuellen Ur-fischertechnik-Motor (Abb. 5).

Diese Teile wurden auch in den Ergänzungskästen für aus fischertechnik gebauten Kompressoren verwendet (Abb. 6).

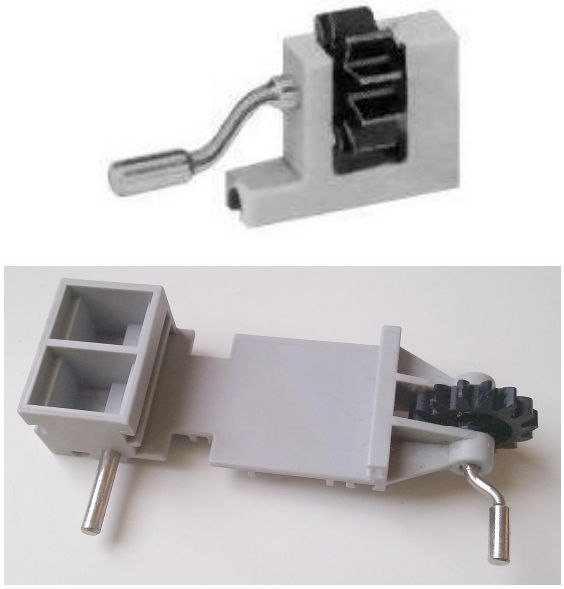


Abb. 5: Kompressoradapter für den Ur-Motor

Abb. 6: Verschiedene Kompressor-Bausätze

Erst später kam der „Pneumatik+“ heraus, der neben den Bauteilen des „Pneumatik“-Kastens ebenfalls die Teile für einen solchen Selbstbau-Kompressor enthielt (Abb. 7).

Abb. 7: Pneumatik+ mit Kompressorteilen

In der späteren Generation der Pneumatik (mit schwarzen anstatt blauen Pneumatikzylindern – siehe weiter unten) gab es neben dem Kompressorzylinder drei neue Teile zum Selbstbau eines kleinen Kompressors (Abb. 8):

Abb. 8: Teile für einen Selbstbaukompressor

Damit konnte man einen ‚Standardkompressor' bauen, der ab dem ersten „Profi Pneumatic"-Baukasten viele Jahre verwendet wurde (Abb. 9).

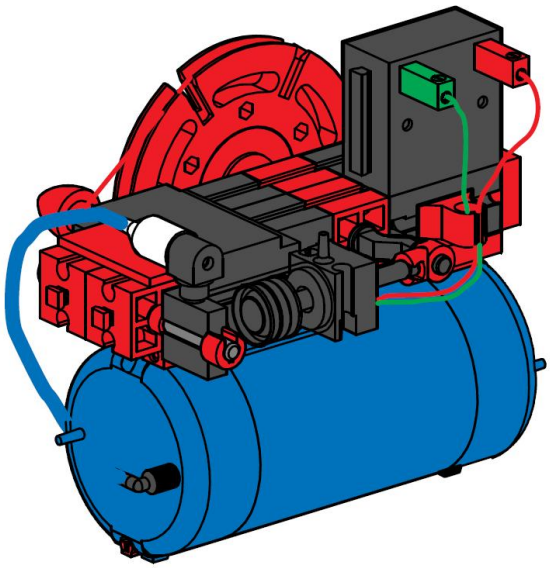


Abb. 9: Selbstbaukompressor mit S-Motor [1]

Viel Freude hat fischertechnik den Fans mit der Einführung des heute aktuellen Kompressors gemacht: Es gab sogar eine Umfrage im fischertechnik-Forum, in dem jede(r) sagen konnte, welche Kriterien wichtig sind. Heraus kam ein toller kleiner Kompressor, wunderbar an mehreren Seiten anbaubar, im ft-Raster liegend, nicht zu laut, aber sehr kräftig. Bis zu 1 bar Druck und ein für viele Modelle sehr brauchbarer Durchsatz stehen damit zur Verfügung (Abb. 10).

Abb. 10: Der aktuelle fischertechnik-Kompressor

Ungezählte Fans haben auch immer wieder selbst neue Druckluftquellen gebastelt und Dritthersteller-Kompressoren an fischertechnik adaptiert – im Bilderpool der ft-Community findet sich mehr dazu.

Was macht man mit Druckluft?

Nachdem wir also geklärt haben, wo wir Druckluft herbekommen, stellt sich die Frage: Was fangen wir damit an? Und schon landen wir beim wichtigsten Pneumatik-Bauelement: Dem *Pneumatik-Zylinder*. Abb. 11 zeigt in der oberen Reihe in Hellblau die ersten Zylinder mit Metall-Kolbenstange in Längen von 60 mm, 45 mm sowie 45 mm mit Rückstellfeder (dessen Kolben also auch ohne Druckluft von alleine wieder eingezogen wird). In der unteren Reihe sieht man die 60- und 45-mm-Pendants der heutigen Zeit mit blauer Kunststoff-Kolbenstange. Diese sind eine Spur weniger leichtgängig als die mit Metall, dafür aber vermutlich günstiger zu produzieren und absolut Rostfrei.

Abb. 11: Pneumatik-Zylinder

Der dritte Zylinder der unteren Reihe ist der 45 mm lange *Kompressor-Zylinder*. Optisch unterscheidet er sich vom normalen 45-mm-Zylinder durch die schwarze Kolbenstange. Technisch sind zwei wichtige Details anders: Erstens ist er am oberen Anschluss undicht (!) und zweitens, weil so die Kolbenstange weniger eng geführt werden muss, deutlich leichtgängiger als der normale Zylinder.

Für seinen Einsatzzweck im S-Motor-Kompressor ist das aber genau richtig: Dort wird ohnehin nur der untere Anschluss verwendet, und die Leichtgängigkeit ist bei der hohen Drehzahl des Kompressors besonders wichtig und nützlich.

Rechts unten in Abb. 11 findet sich der jüngere 60-mm-Zylinder mit Rückholfeder, und rechts oben schließlich ein „halber Zylinder“ als Spezialteil zum Überstülpen eines Luftballons – den kann man dann aufblasen lassen.

Zubehörteile

Neben den Pneumatikschläuchen gab und gibt es bis heute die verschiedenen Anschluss-Stücke sowie den Achsadapter für die Zylinder in im Wesentlichen unveränderter Form (Abb. 12).

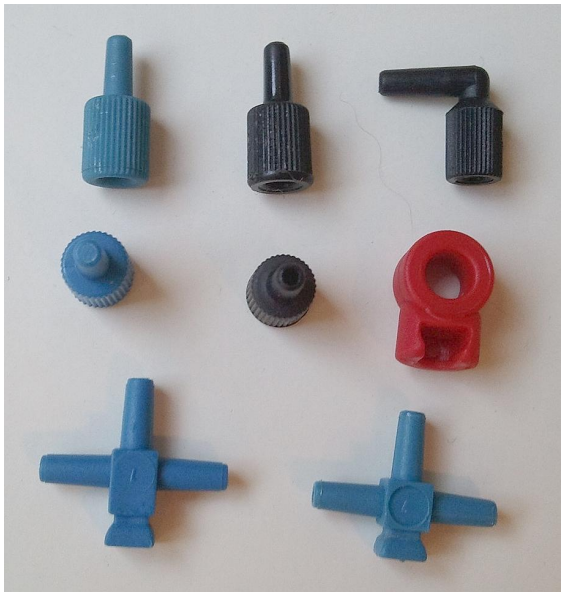


Abb. 12: Stopfen (blau), Anschlussstücke gerade und gewinkelt (schwarz), Achsadapter für Zylinder, T-Stücke

Die blauen Teile links oben sind reine *Stopfen*, die einen Anschluss, z. B. vom Luft-Tank, dicht abschließen. Nicht zu verwechseln damit sind die schwarzen Teile rechts daneben, mit denen man auf einen Anschluss gerade oder um 90 ° abgewinkelt einen Schlauch stecken kann (man beachte das Loch im schwarzen Teil

in der Mitte, welches im blauen fehlt). Wenn man eine Druckluftleitung mit mehr als einem Ziel verbinden muss, verwendet man die blauen Verteilerstücke (unten in Abb. 12). Es gab sie in zwei leicht unterschiedlichen Baulängen, wie man sieht.

Für den Bau von Kompressoren, aber auch für pneumatische Logikschaltungen unverzichtbar, sind die *Rückschlagventile*:

Abb. 13: Rückschlagventile

Die erste Fassung war blau, die aktuelle ist schwarz, aber die Funktion ist dieselbe geblieben: Druckluft (beispielsweise, aber nicht zwingend, vom Kompressorzylinder eines Selbstbau-Kompressor) kommt am in Abb. 13 untenliegenden Anschluss herein und kann am oberen Ende ausströmen. Nur zurück geht es nicht, denn eine Feder drückt ein Bauelement im Inneren auf den unteren Anschluss. Druck von unten kann gegen die Feder arbeiten und so eine Öffnung herstellen, Druck von oben schließt das Ventil aber nur umso fester.

Das ist also das pneumatische Gegenstück zur elektronischen Diode. Wir werden in einem separaten Beitrag noch ausführlich vorstellen, was für raffinierte Druckluft-Schaltungen man mit Rückschlagventilen herstellen kann.

Ventile – Schalter für Druckluft

Festo-Ventile

Beim Anblick von Abb. 14 wird es Kennern der ersten Pneumatik-Generation von fischertechnik warm ums Herz: Sämtliche dieser Bauteile werden nicht mehr hergestellt und hinterlassen eine schmerzliche, weil praktisch unausgefüllte Lücke.

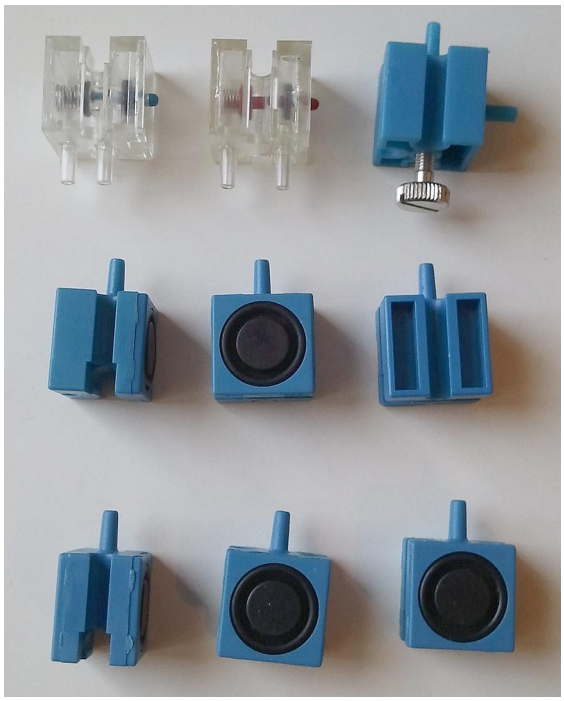


Abb. 14: Festo-Ventile, Drossel, Einfach- und Doppelbetätiger

Im Einzelnen sieht man auf diesem Bild:

- Rechts oben eine einfache *Drossel*, die den Luftstrom durch sie hindurch per Stellschraube mehr oder weniger stark hemmt. Das ist wichtig, wenn ein Pneumatikzylinder nicht schlagartig, sondern schön langsam ein- und ausfahren soll. Ebenso können mit Drosseln einstellbare Zeitglieder gebaut werden, was wir ebenfalls in einem späteren Beitrag ausführlich behandeln werden.
- Links daneben finden sich zwei *Festo-Ventile*. Die heißen so, weil sie von der in Pneumatik-Kreisen weltbekannten Firma Festo für fischertechnik entwickelt wurden (Abb. 15).

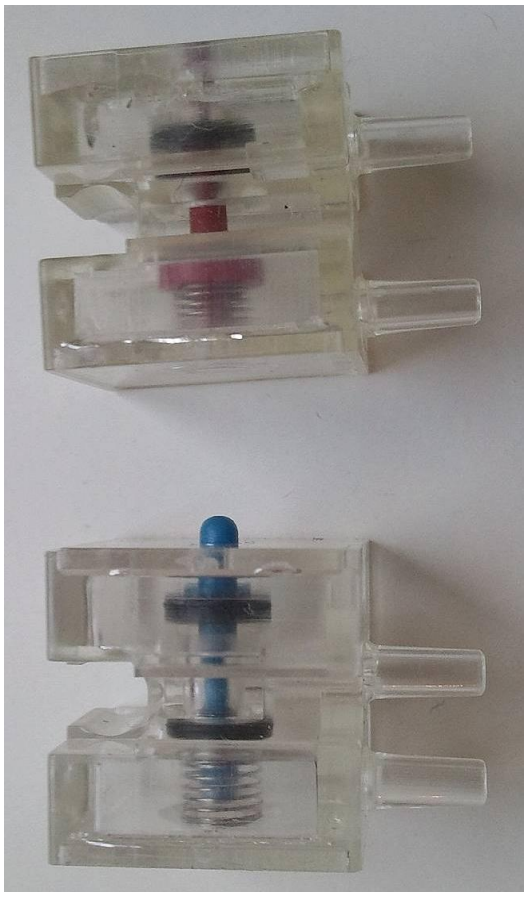


Abb. 15: Detailansicht der Festo-Ventile

Es gibt – Verzeihung, gab – sie mit blauem und rotem *Stößel*, der jeweils ca. 1 mm aus dem Gehäuse in der Größe eines Bausteins 15 heraus ragt. An einem Anschluss legt man Druckluft an. Die kommt am anderen Anschluss aber nur dann an, wenn der Stößel gedrückt (blaues Ventil, „Öffner“) bzw. gerade nicht gedrückt (rotes Ventil, „Schließer“) ist. Diese Bauteile erfüllen also für Druckluft genau denselben Zweck, wie es der Arbeits- (blaues Ventil) bzw. der Ruhekontakt (rotes Ventil) eines normalen fischertechnik-Tasters für den elektrischen Strom tun: Es handelt sich um mechanisch betätigte Schalter für das Durchlassen der Druckluft mit automatischer Rückstellung durch Federkraft, sobald der Stößel nicht mehr gedrückt wird.

- Diese Ventile lassen sich nicht nur mit einfachen mechanischen Mitteln betätigen, sondern mit den eben dazu geschaffenen *Betätigern*, die Abb. 14 in den unteren beiden Reihen zeigt. Auch diese Bauteile haben etwa die Größe eines Bausteins 15 und passen im ft-Raster direkt neben ein Festo-Ventil.

Die Betätiger haben nur einen Druckluftanschluss, der einen Gummibalg aufbläst, sobald Druckluft anliegt. Der Balg hebt sich dann einige wenige Millimeter aus dem Gehäuse – gerade genug, um den Stößel eines Festo-Ventils oder einen ft-Taster zu betätigen. Auf diese Weise können wir per Druckluft ein Druckluft-Ventil betätigen. Das erst eröffnet den Weg zu rein pneumatisch betriebenen Schaltungen, die mit Druckluft ebenso funktionieren wie die vielen verschiedenen Schaltungen aus der ‚Motorsteuerungen’-Artikelserie in der ft:pedia [3].

Ebenso gelingt es mit Taster und Betätiger leicht, per Druckluft ein elektrisches Signal abzugeben: Der mit Druckluft „beaufschlagte“ Betätiger drückt den Taster, der in geeignetem Abstand zum Gummibalg montiert werden muss. Ein wichtiges Anwendungsgebiet dafür ist die Abschaltung eines Kompressors, sobald genügend Druck anliegt.

Die Betätiger gab es in zwei Bauformen: Beim *Einfach-Betätiger* geht der Gummibalg nur in eine Richtung aus dem Gehäuse, beim *Doppel-Betätiger* auf zwei Seiten des Bauteils. Mit letzterem kann man mit einem einzigen Druckluftsignal gleich zwei Festo-Ventile steuern. Wenn das je ein (blauer) Öffner und ein (roter) Schließer ist und man deren Ausgänge an einen Zylinder anschließt, kann man durch Anlegen oder Abschalten von Druckluft mit einer einzigen Steuerleitung kontrollieren, ob der Zylinder aus- oder eingefahren werden soll.

Abb. 16 zeigt eine kleine pneumatische Logik-Schaltung: Links steuert ein (schmaler) Doppelbetätiger zwei Festo-Ventile an, weiter rechts wird ein beweg-

licher Hebel von zwei (etwas breiteren) Einfach-Betätigern durch kurzen Druckluftimpuls nach links oder rechts bewegt.

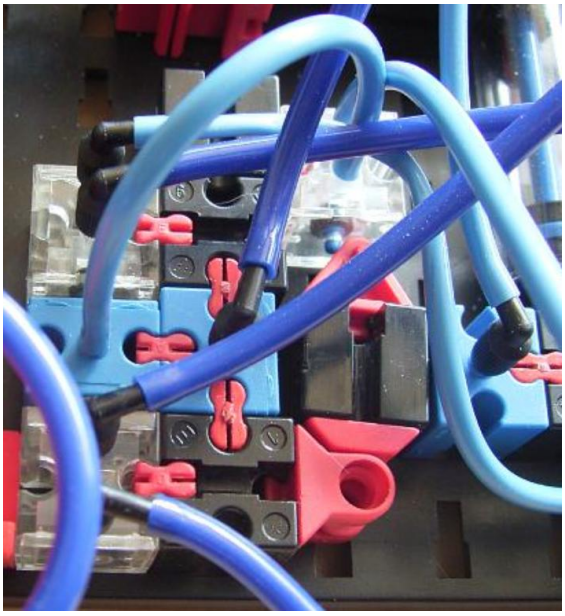


Abb. 16: Einfach- und Doppelbetätiger zur Ansteuerung von Festo-Ventilen

Er bleibt in der jeweiligen Stellung stehen und drückt bzw. drückt nicht auf das Festo-Ventil oben im Bild – fertig ist ein pneumatisches Flip-Flop, ein 1-Bit-Speicher.

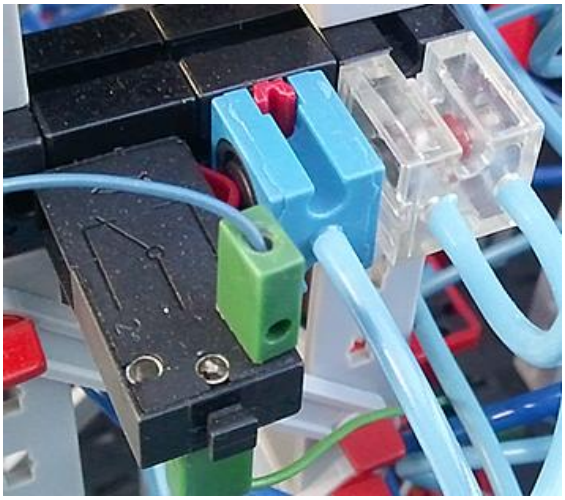


Abb. 17: Doppelbetätiger steuert Festo-Ventil und Elektro-Taster an

Genau passend zu den Festo-Ventilen gab es die *Rollenhebel* (Abb. 18). Diese leichtgängig federnden Bauteile kann man verwenden, um die Festo-Ventile (mit nur ca. 1 mm Schaltweg) auch von Maschinenteilen oder von Hand zu betätigen, die viel größere Wege zurücklegen. Durch die Federwirkung wird das Festo-Ventil dennoch nicht beschädigt. Zudem kann man durch das 4-mm-Loch am Ende des Hebels ft-Achsen stecken und auf diese Weise auch mehrere Rollenhebel miteinander koppeln – alle werden dann gleichzeitig gedrückt bzw. losgelassen und steuern mehrere Ventile mit derselben Bewegung.

Zum allergrößten Bedauern des Autors und vieler ft-Pneumatik-Fans werden aber die Festo-Ventile, Betätiger und Rollenhebel allesamt schon lange nicht mehr hergestellt. Man kann sie allerdings noch auf dem Gebrauchtmarkt finden.

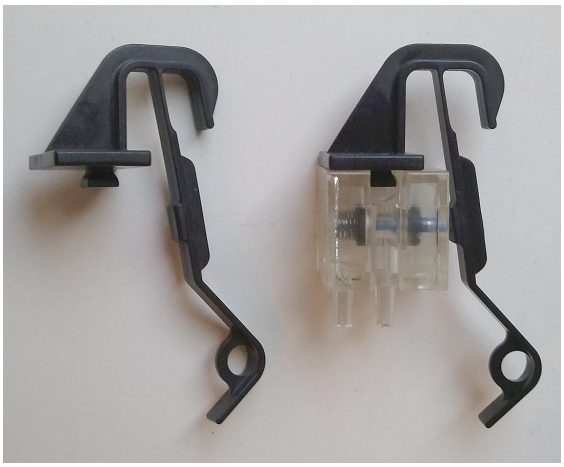


Abb. 18: Rollhebel zu Festo-Ventilen

Wenngleich sie traurigerweise unersetzt sind, werden wir in weiteren Pneumatik-Artikeln dennoch sehen, wie man mit den aktuellen Ventilen zumindest einige der Verwendungsmöglichkeiten realisiert.

Das heutige Pneumatik-Ventil

Heute gibt es das in Abb. 19 gezeigte *Pneumatik-Handventil*. Durch den blauen Drehschieber kann man am mittleren Eingang anliegende Druckluft stoppen oder auf einen der beiden anderen Anschlüsse verteilen. So kann man einen Pneumatikzylinder steuern, und natürlich kann man diese Ventile mit ein paar Verrenkungen auch z. B. per Motorkraft automatisiert betätigen.

Abb. 19: Handventil und Unterdruck-Saugnapf

Ebenfalls auf dieser Abbildung sieht man den noch relativ jungen *Saugnapf*, mit dem man leichte Bauteile mit Unterdruck anheben kann.

Magnetventile

In der ersten Pneumatik-Generation von fischertechnik gab es noch keine eigens hergestellten Elektro-Ventile.

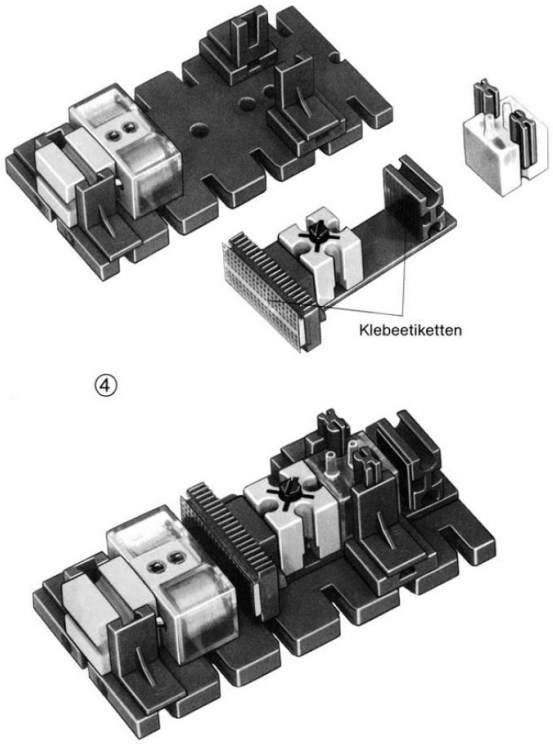


Abb. 20: Klassisch: selbst gebautes Elektro-Ventil

Stattdessen bediente man sich der klassischen Kombination aus Elektromagnet und Rückschlussplatte, um ein Festo-Ventil elektromagnetisch zu betätigen. Abb. 20 zeigt einen gut funktionierenden Aufbau von Seite 69 der Anleitung zum ersten Pneumatik-Kasten [2], Abb. 21 einen Modellausschnitt.

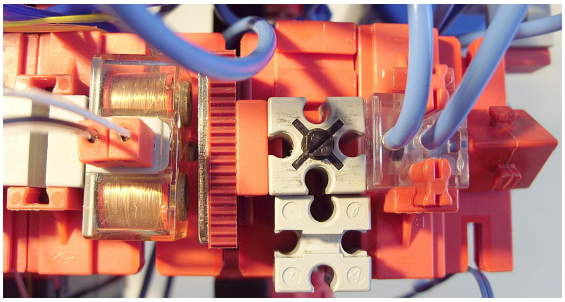


Abb. 21: Einsatz eines Selbstbau-Elektroventils im Modell

1986 folgte das erste spezielle Elektro-Ventil, welches aber immer noch einen normalen ft-Elektromagneten und ein Festo-Ventil verwendete. Einige Spezialteile machten die Sache aber kompakter:

Abb. 22: Elektromagnetisches Ventil von 1986

Abb. 23: Elektromagnetisches Ventil von 2002

Ab 2002 schließlich gab es das heute noch lieferbare fertige Elektro-Ventil (Abb. 23). Ein unabhängig von fischertechnik produziertes Ventil wird hier lediglich mit den weißen Adaptern auf ft-Schlauchmaß gebracht und mit simplem Klebeband am Modell befestigt – nicht gerade ‚fischertechnik-like', aber es funktioniert sehr gut und ist relativ kompakt.

Sensoren

Die erste Pneumatik-Generation begnügte sich nicht damit, Pneumatik-Zylinder mehr oder weniger manuell zu steuern. Vielmehr wurde ein tatsächlicher Einsatzzweck, die Steuerung pneumatisch betriebener Maschinen, auch wirklich wie in der Industrie mit pneumatischen Sensoren nachgebildet. Zwei besondere Sensoren gab es – beide verlangen allerdings eine ordentlich leistungsfähige Luftquelle (die Selbstbaukompressoren mit ft-Motoren scheiden dafür leider aus).

Luftschranke

Zum einen kann man, genau wie mit Lampe und Fotozelle Lichtschranken hergestellt werden, mit zwei langen Düsen eine *Luftschranke* bauen:

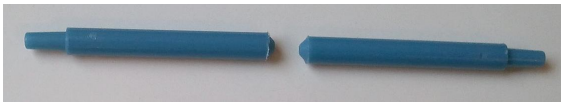


Abb. 24: Düsenpaar für Luftschranken

Das funktioniert so: Eine der Düsen wird mit Druckluft beaufschlagt. Durch ihre Länge (30 mm) wird der Luftstrom darin „laminar", das heißt wirbelfrei. Die Luft strömt also an der Spitze (an der sich eine kleine Öffnung befindet) einfach direkt ins Freie, und zwar als ein recht gerader Luftstrahl. In nicht allzu weiter Entfernung (mit den ft-Düsen sind nur ein paar Millimeter realisierbar) sitzt nun genau fluchtend eine zweite Düse, die diesen Luftstrahl wieder auffängt. Natürlich geht auf dem Weg dahin Druckluft verloren, sodass hier nicht mehr genug Druck ankommt, um einen Pneumatikzylinder zu betreiben. Aber für einen der schon aufgeführten *Betätiger* genügt der Druck: Solange die Luftschranke nicht unterbrochen wird, schlägt der Betätiger aus und kann ein Festo-Ventil ansteuern. Dieses wiederum kann dann tatsächlich einen Pneumatikzylinder versorgen.

Unterbricht man den Luftstrom z. B. durch Einbringen einer dünnen Bauplatte zwischen den beiden Düsen, fällt der Betätiger drucklos ab und lässt das Festo-Ventil frei. So kann man ‚berührungsfrei' (bis auf die strömende Luft) feststellen, ob sich etwas in der Luftschranke befindet oder nicht – gerade wie es eine Lichtschranke auch tut.

Staudüsen

Ähnlich funktioniert die *Staudüse*. Am in Abb. 25 unteren Anschluss kommt Druckluft herein, die oben (dort ist eine kleine Öffnung) ins Freie strömt. Sobald die Öffnung aber abgedeckt wird, staut sich die Druckluft im Inneren des Bausteins, und am oberen Anschluss kommt die Druckluft heraus. Die kann man wie bei der Luftdüse mit einer Betätiger/Festo-Ventil-Kombination verstärken und so in einer Steuerung nutzbar machen.

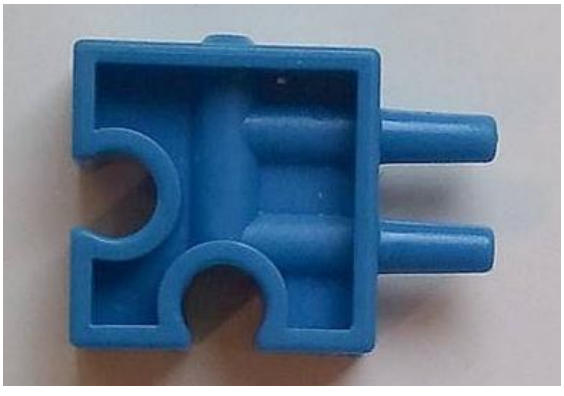


Abb. 25: Staudüse

Ein Anwendungsbeispiel zeigt Abb. 26: Eine ft-Schwingfeder aus dem älteren Elektromechanik-Programm – eine leicht biegbare 15 mm bereite Metall-Lasche – kann die Öffnung der Staudüse abdecken.

Diese wird zur Vermeidung von übermäßigem Luftverbrauch über eine Drossel mit Druckluft versorgt. Der Ausgang ist mit einem (Doppel-) Betätiger verbunden, der über zwei Festo-Ventile das schwache Signal der Staudüse auf ein brauchbares Maß verstärkt.

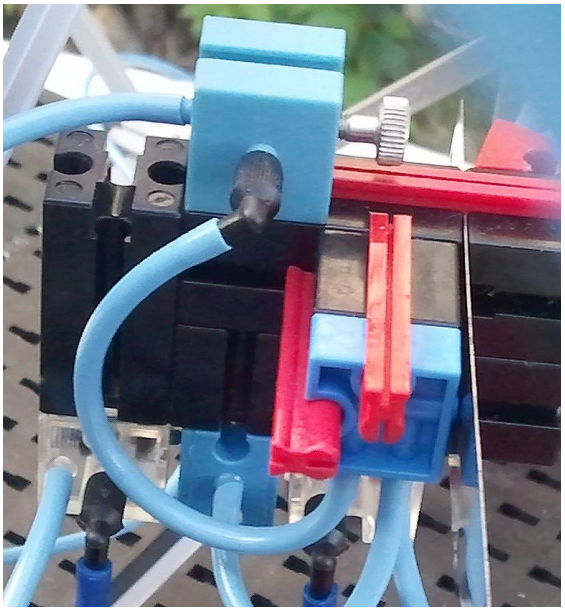


Abb. 26: Gedrosselte Versorgung der Staudüse mit nachgeschalteter Signalverstärkung

Ausblick

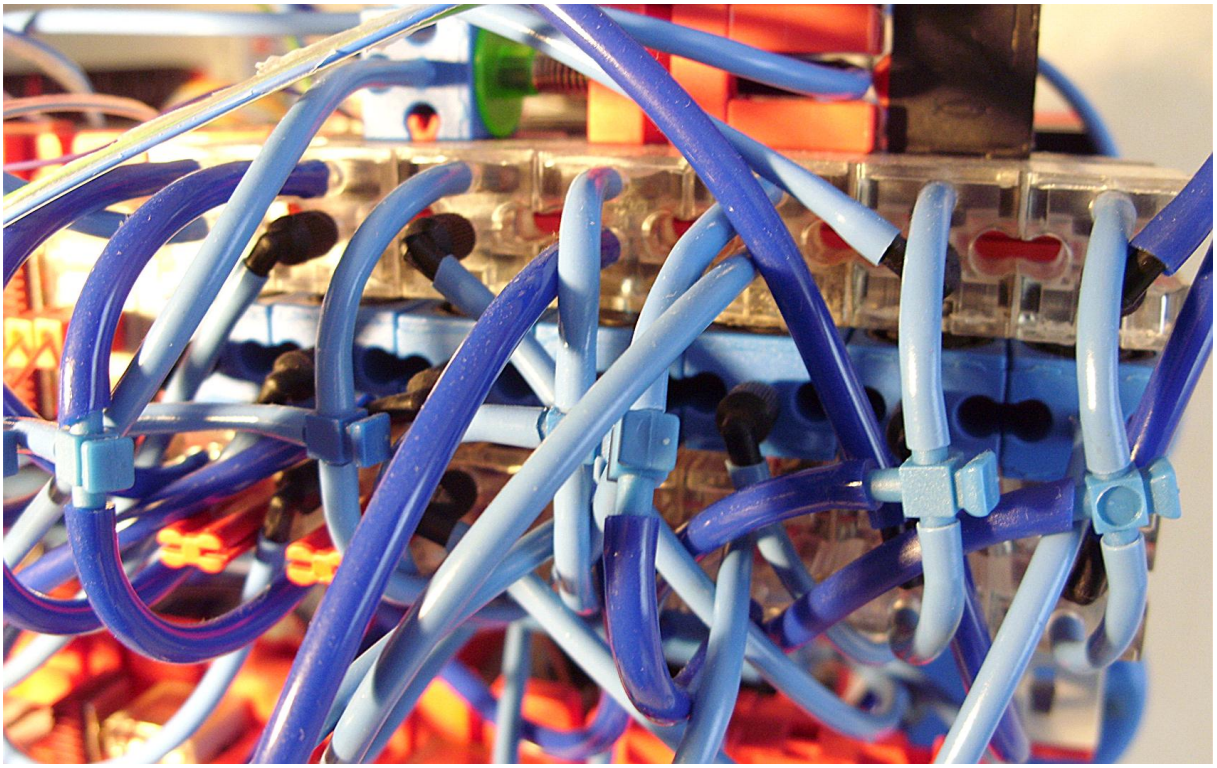
In nachfolgenden Beiträgen werden wir uns den Möglichkeiten all dieser Bauteile im Modell ausführlich zuwenden. Dabei werden Modelle vorgestellt, die nicht nur mit Druckluft betrieben, sondern auch mit Druckluft gesteuert werden – sowohl mit den Festo-Ventilen als auch, soweit möglich, nur mit den heute noch produzierten Handventilen. Auf dem Gebiet pneumatischer Steuerungen gibt es genau so viel zu sagen und genau so viele reizvolle Modelle zu bauen wie mit Elektrik und Elektronik. Versprochen.

Quellen

[1] Fischer-Werke: *Profi Pneumatik*, Tumlingen, 1995.

[2] Fischer-Werte: *Pneumatik Anleitung*. Tumlingen, 1981.

[3] Stefan Falk: *Motorsteuerungen*. ft:pedia 1-4/2011.



Tipps & Tricks

ft-Spezialteile made by TST (Teil 6)

Andreas Tacke

In einer lockeren Reihe stellt TST einige von ihm entwickelte Spezialteile vor, die so manche Lücke beim Bauen mit fischertechnik schließen. Im heutigen Beitrag geht es um Zubehör für den XM- bzw. Encoder-Motor.

In diesem Beitrag geht es um das eine oder andere Teil, mit dem sich die Verwendung bzw. der Einbau des XM- bzw. Encoder-Motors verbessern lässt.

Das schöne an diesem relativ jungen Motor von fischertechnik ist, dass er sich besser verbauen lässt als der Powermotor, da er durch sein Außenmaß von 30 x 30 x 60 mm perfekt ins ft-Raster passt. An drei der umliegenden Seiten sind jeweils zwei ft-Nuten und an der vierten Seite eine ft-Feder vorhanden, was den Anbau sehr komfortabel macht.

Leider sind an der Stirnseite keine Befestigungsmöglichkeiten wie bei den alten M-Motoren (31039/32618) vorgesehen. In einigen Einbaufällen fehlt das tatsächlich, also habe ich mir zu diesem Thema Gedanken gemacht. Die Idee war, eine ähnliche Anbauplatte zu fertigen, wie ich sie auch schon für den Powermotor entwickelt habe (siehe ft-pedia 03/2012 [1]).

Nach einigen Versuchen hatte ich die Lösung. Zum Befestigen der Platte wird die Schraubverbindung an der Stirnseite verwendet, mit der der Motor im Gehäuse befestigt ist. Die Originalschraube wird durch eine Schlitzschraube M3 x 10 ersetzt, mit der sich die Anbauplatte am Motor befestigen lässt (Abb. 1).

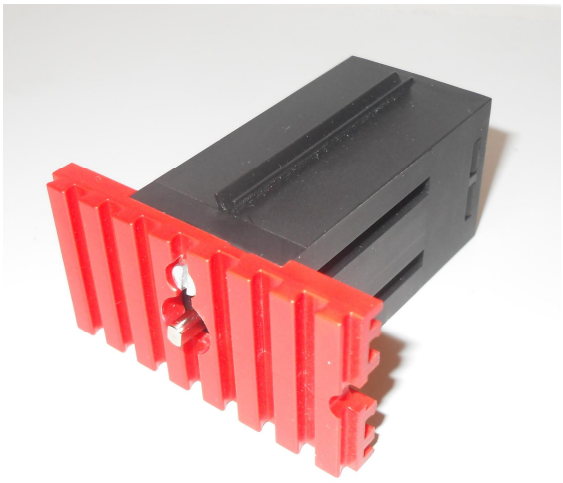


Abb. 1: XM-Motor mit angebauter Platte

Damit besteht nun die Möglichkeit, auch an der Stirnseite des Motors anzubauen. Auf der Rückseite der Anbauplatte befinden sich neben dem Motor jeweils zwei Nuten, die ebenfalls zum Befestigen genutzt werden können.

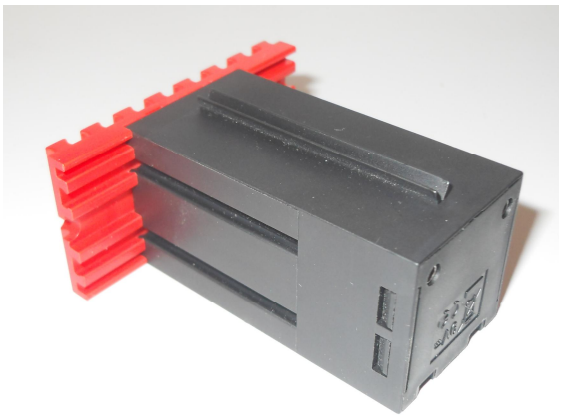


Abb. 2: Rückansicht

Perfekt lässt sich so mit sehr wenigen Bauteilen eine stabile Befestigung zum Differential herstellen (Abb. 3).

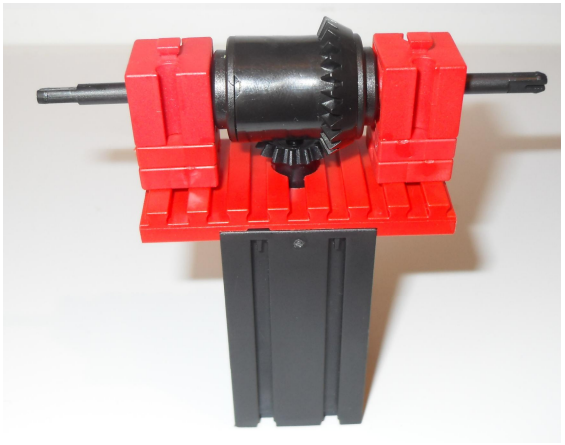


Abb. 3: XM-Motor mit angebautem Differential

Die M-Motoren (31039/32618) besaßen zudem eine fest angebaute rote Schnecke zum Antrieb diverser Getriebeteile. Wäre es nicht schön, dies auch mit dem XM Motor machen zu können? Dafür muss eine Schnecke her, die sich fest auf der Achse befestigen lässt (Abb. 4).

Abb. 4: Modifizierte Schnecke mit Madenschraube zum Anbau an den XM-Motor

Damit lässt sich nun auch der Getriebewinkel 32619 unter Verwendung einer Bodenplatte (z. B. 32859) mit dem XM-Motor verbinden. Jetzt haben wir für den XM-Motor auch noch ein stabiles Drei-Gang-Getriebe.

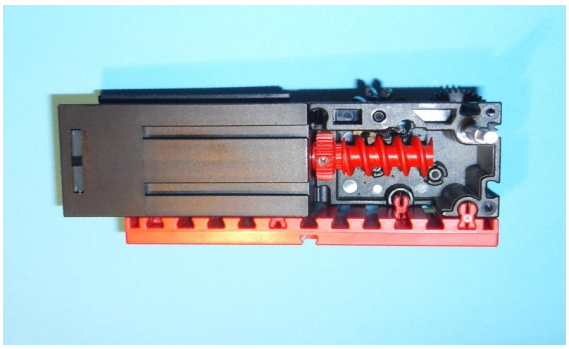


Abb. 5: Motor mit Drei-Gang-Getriebe

Alle Abbildungen in diesem Beitrag zeigen den XM-Motor. Natürlich lassen sich die Anbauplatte und auch die Schnecke mit dem Encoder-Motor kombinieren. da dieser äußerlich baugleich mit dem XM-Motor ist.

Der XM- und auch der Encoder-Motor sind durch ihre kompakte Bauweise und die vielen Befestigungsmöglichkeiten eine echte Bereicherung für das fischertechnik-Sortiment.

Es zeigt sich aber auch, man Gutes durchaus noch etwas verbessern kann …

Quellen

[1] Tacke, Andreas: *ft-Spezialteile made by TST (Teil 2)*, ft:pedia 3/2012, S. 27-28.

Elektronik

Automatik zur Steuerung eines Krans

Werner Hasselberg

fischertechnik und der Kranbau sind seit der Erfindung der Statik-Elemente untrennbar miteinander verbunden. Eigentlich kein Wunder: Mit keinem anderen Spielsystem lassen sich bessere und vielseitigere Kräne bauen. Und weil sie so schön zu bauen sind, beschäftigt sich dieser Beitrag mit der Frage, wie sie vollautomatisch – und ohne PC-Hilfe – gesteuert werden können. Die hier gezeigte Steuerung ist aber noch vielseitiger. Sie kann, etwas erweitert, sogar einen dreiachsigen Roboter steuern, ohne dass dazu ein Computer erforderlich wäre.

Selbstverständlich kann man einen Roboter mit PC-Steuerung sehr vielseitig programmieren und auch fast unbegrenzt viele Arbeitsschritte exakt ausführen. Will man das mit reiner Elektronik realisieren, stößt man irgendwann an logistische und technische Grenzen. Trotzdem macht gerade das Ausprobieren verschiedener Schaltungsvarianten, das Kombinieren der unterschiedlichen Funktionsweisen und der ganze Entstehungsprozess, bis am Ende die Modellsteuerung steht, sehr viel Spaß, weil man näher an die Technik rückt als beim Programmieren eines Interfaces.

Am Anfang war der Kran. Zu seiner vollen Funktionsfähigkeit gehören mindestens die folgenden Funktionen:

1. Start des Kranmotors für das Herablassen des Hakens (unser Kran verfügt über einen Elektromagnet als Haken) und Stopp am Ende.
2. Einschalten des Elektromagneten und Last aufnehmen.
3. Erneuter Start des Kranmotors (die Last wird gehoben) und Stopp am Ende.
4. Start der Krandrehung nach rechts.
5. Stopp der Krandrehung.
6. Magnethaken mit Last wieder senken und Stopp am Ende.
7. Elektromagnet geht aus und löst die „Verriegelung“.
8. Haken geht wieder nach oben und Stopp am Ende.
9. Kran dreht sich wieder zurück in die Ausgangsposition.
10. Fertig.

Wer die ‚Automatik für weichen Motorstart und –stopp’ aus der ft:pedia 3/2013 [1] nachgebaut hat und über eine Eisenbahn verfügt, kann den Kran automatisch starten und so z. B. einen Container verladen.

Die vielen Schritte zeigen aber, dass es durchaus eine kleine Herausforderung für unsere ft-Elektronik ist, sie alle mit möglichst wenig Aufwand zu bewerkstelligen.

Folgende Elemente werden verwendet:

- Vier em3-Relais (nicht die aus dem ec1 Baukasten, da jene nur mit Gleichrichter arbeiten) und ein Eigenbaurelais (spart Bausteine)
- Ein Grundbaustein aus ec2 (für den Kranschwenk)

- Aus dem Elektronik-Ergänzungskasten:
  - Zwei Schwellwertschalter (SWS)
  - Zwei Leistungsstufen (LST)
  - Zweimal Spannungsversorgung (SPV, notfalls genügt auch eine)
  - Vier Minitaster
  - Fünf Elkos 470 µF
  - Zwei 10 kΩ Widerstände
  - Fünf Dioden
  - Falls möglich: Zwei Trafos als Stromquelle

Zeitschalter

In praktisch jedem der neun Arbeitsschritte muss ein Motor oder der Elektromagnet ein- und nach festgelegter Laufzeit ausgeschaltet werden.

Ein einfacher Zeitschalter oder Monoflop (MF), wie im Begleitbuch des Ergänzungskastens auf Seite 46 beschrieben [2], startet durch kurze Betätigung eines Tasters. Der Taster vollzieht also zwei Arbeitsschritte: Drücken und Loslassen.

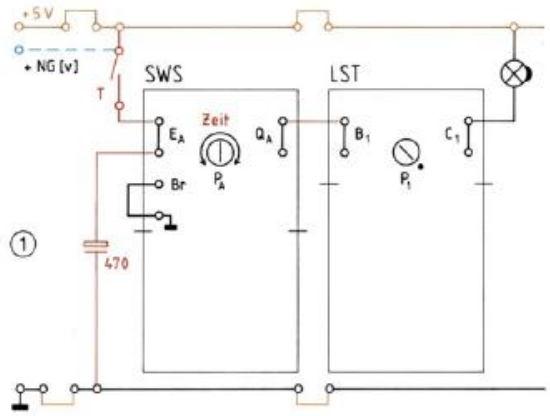


Abb. 1: Zeitschalter (MF) aus Seite 46 des Elektronik-Begleitbuchs [2]

Es handelt sich hier um ein nachtriggerbares Monoflop. Wie lange es eingeschaltet, also SWS1 „an“ ist, bestimmt die Entladung des Elkos. Dazu wird er durch den kurzen Impuls des Tasters geladen und entlädt sich, wenn der Taster gelöst (d. h. die Verbindung zu (+) unterbrochen) ist. Besser wäre ein Monoflop, das nur einen Arbeitsschritt (z. B. „Taster drücken“) benötigt, um sich einzuschalten, und nach einiger Zeit trotz gedrückten Tasters ausgeht. Damit könnte ein Motor im Prinzip über eine gewisse Zeit ein- und wieder ausgeschaltet werden, was bereits die halbe Miete ist. Zwar ist es möglich, ein nicht nachtriggerbares Monoflop mit der ft-Elektronik zu bauen (wie z. B. das Glücksrad auf Seite 52 [2]), es benötigt aber gleich zwei SWS und ist daher sehr kostspielig. Also müssen wir einen anderen Weg finden.

Das in der Abb. 1 beschriebene ‚einfache’ Monoflop ist schon ein guter Anfang, denn auf diese Art können zwei Zeitschalter in einem Baustein verwirklicht werden. Die gesamte Anlage wird am Ende mit vier Bauteilen (zwei SWS-Bausteine, zwei LST-Bausteine und SPV) alle Arbeitsschritte bis auf die Krandrehung bewerkstelligen können.

Gottlob gibt es noch die guten alten em3-Relais, deren Hilfe wir jetzt unbedingt brauchen. Schade, dass Fischertechnik kein so umfangreiches Programm an Elektromechanik- und Elektronikteilen wie früher im Sortiment hält. Deren Kombination erlaubte viele tolle Modellsteuerungen.

Schritt 1: Seilmotor ein- und ausschalten

Zuerst bauen wir ein MF nach Abb. 1. Der Anschluss des SWS an die SPV und LST wurde in der Abb. 2 nicht dargestellt; das ist relativ einfach und sehr leicht im Begleitbuch des Ergänzungskastens nachzulesen [2].

Der Elko 470 µF wird zwischen zwei Lampenfassungen platziert und kann so außerhalb des SWS liegen. Damit ist der SWS1 zugänglicher und nicht so vollgepackt. Der 10 kΩ-Widerstand von der (+)

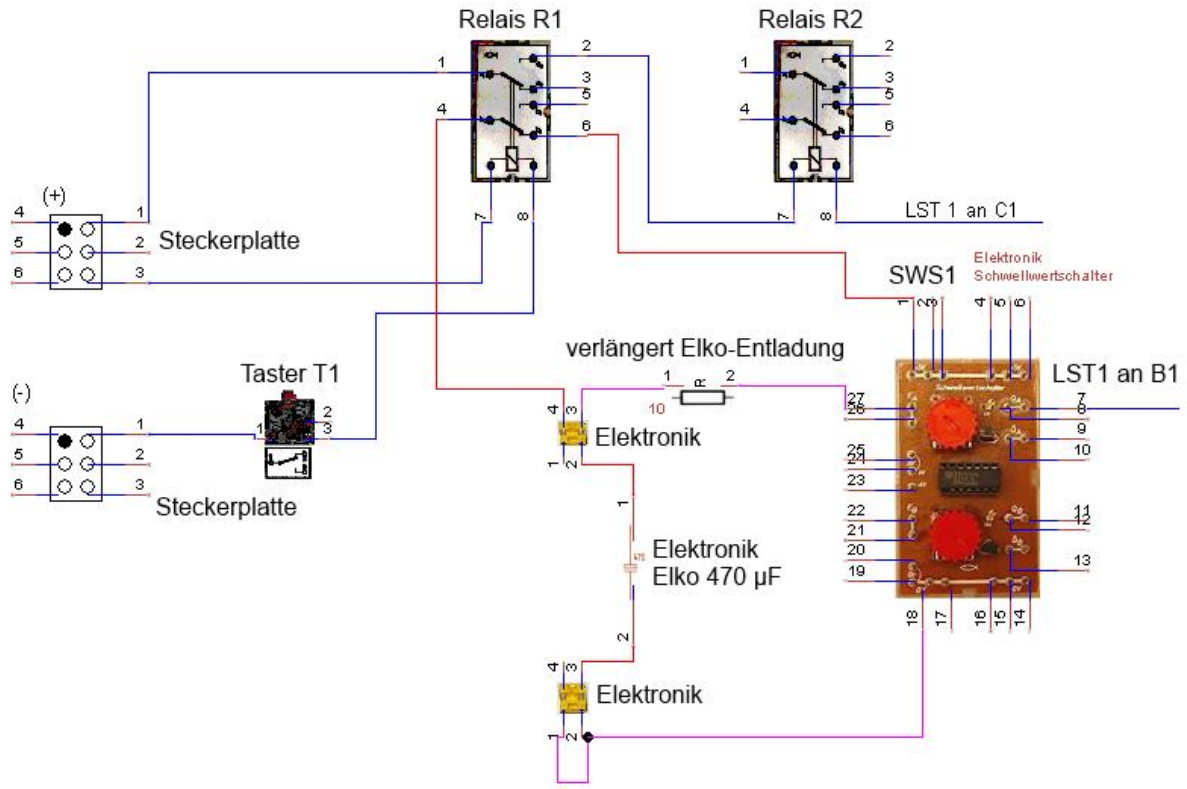


Abb. 2: Zeitschalter für Kranhaken ab

Seite des Elkos wird mit EA verbunden. Das verzögert die Entladezeit.

Mit dem Poti wird die benötigte Dauer fein justiert. Eine noch längere Verzögerung würde der Anschluss des Elko mit (+) direkt an am Poti Br des SWS1 bewirken. Allerdings reagiert der SWS dann noch sensibler auf Poti-Änderungen, und das erschwert es, die richtige Einstellung zu finden. Besser ist deshalb der Anschluss wie in Abb. 2.

Das zweite em3-Relais wird R3 genannt, demzufolge muss es auch ein R2 geben, das aber erst später in die Schaltung integriert wird. Wir beginnen mit dem R3, weil dessen Funktion etwas einfacher zu beschreiben ist.

R1 ist ausgeschaltet, solange der Starttaster T1 nicht gedrückt wird. Bei Druck auf T1 zieht es an und trennt damit die (+) Verbindung des Elko. Augenblicklich beginnt seine Entladezeit; der Zeitschalter beginnt zu arbeiten. Gleichzeitig, und das ist der Trick, wird über den blauen Anschluss von R1 auch R3 eingeschaltet. Der SWS1 wird bereits eingeschaltet, sobald die Anlage ans Netz geht. R3 schaltet sich aber erst jetzt ein. Die Zeit läuft, und ist sie zu Ende, schaltet sich der SWS1 aus. Damit schaltet sich die LST1 aus. R3 an Minus (C1) wird geblockt und geht ebenso wieder aus.

Mit nur einem Klick werden somit zwei Arbeitsschritte erledigt. Würde anstelle von R3 ein Motor verwendet so wäre das Resultat bereits eine bestimmte Laufzeit. Das genügt uns aber noch nicht, denn er muss auch seine Laufrichtung ändern können, und dazu ist ein Polwendeschalter erforderlich. Außerdem sollte er am Ende der Laufzeit schnell gebremst werden, um eine möglichst exakte Magnethaken-Positionierung zu erreichen. Beide Aufgaben übernimmt das R3 zusammen mit dem noch folgenden R2.

Schritt 2: Seil-Motor an R3 anschließen

R3 ein muss also den Motor für das Herablassen des Hakens einschalten. Die Motorsteuerung wird am besten mit einer

eigenen Stromversorgung betrieben, da alles zusammen für einen Trafo etwas zu viel ist und Motorschwankungen verursachen kann, die eine genaue Steuerung des Magnethakens sehr schwer machen. Das ist sehr wichtig, denn nur wenn er bei jedem Durchlauf so exakt wie möglich dieselbe Strecke zurücklegt, kann er einen Container sicher aufnehmen.

Da R3 vor dem Klick auf T1 aus ist, verläuft die Schnellbremsung über die Ruhekontakte. Die eingezeichnete Diode ist unbedingt erforderlich und muss polrichtig wie im Schaltbild eingebaut werden – sie spielt für das Polwenden eine wichtige Rolle. Sobald sich R3 einschaltet, startet der Seilmotor. Der Magnethaken geht nach unten und bleibt am Ende erst einmal stehen. Die Polung des Motors muss dafür entsprechend gewählt werden und ist abhängig davon, wie der Seilzug und die Rolle gebaut sind.

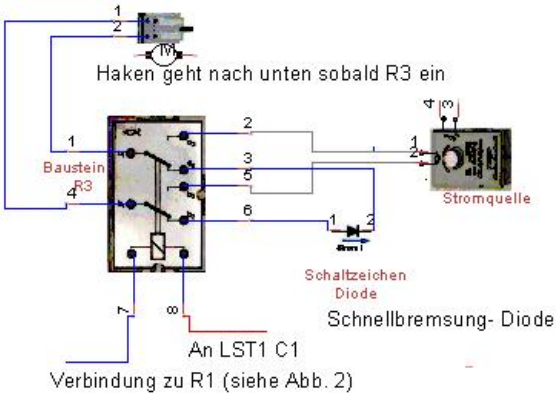


Abb. 3: Seilmotor an R3 für Haken ab

Schritt 3: Nach Motorstopp Containeraufnahme

Ist der Magnethaken am Container angekommen, schadet eine kurze Wartezeit nicht. So kann er sich besser auspendeln und den Container leichter aufnehmen. Da wir zwei SWS Bausteine brauchen, können wir uns diesen Luxus leisten. Der wichtigste Vorteil der Pause ist aber, dass SWS2 das Polwenden des Seil Motors vornimmt und uns die nötige Pause für das Lösen des Magnethakens verschafft (dazu später mehr).

Die Pause erfordert den SWS2 wiederum als Zeitschalter, der sich einschaltet, sobald der SWS1 abschaltet (der Haken befindet sich dann unten). Der SWS1 ist an, bis der Elko sich nach Ablauf der Zeit entladen hat. Dieses Prinzip behalten wir auch für den SWS2 bei, nur dass dieser über SWS1 anstatt (wie SWS1 selbst) über ein Relais angesteuert wird. Solange er an ist, liefert sein Ausgang QA (+). Diese Spannung kann dazu verwendet werden, den Elko 2 des SWS2 aufzuladen. Er kann direkt an den SWS2 (EB) angebracht werden. Die Verbindung von QA (+) braucht aber noch eine Diode, damit dessen Ladung nicht über QA (-) abfließen kann, sobald SWS1 aus geht und sich der Elko 2 entlädt.

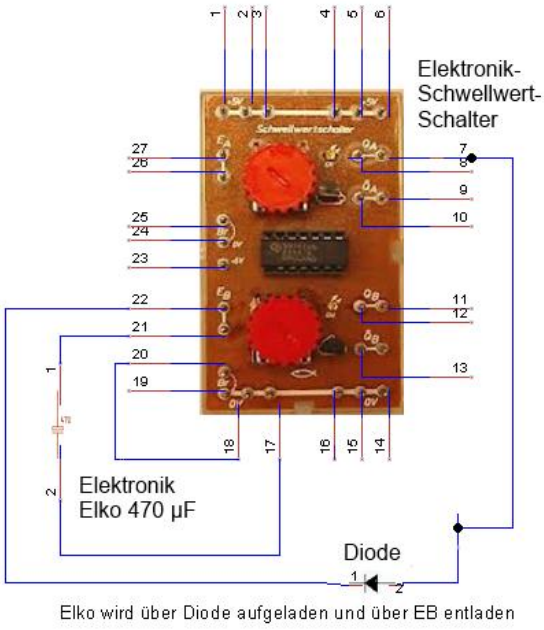


Abb. 4: SWS2 Pause

Sobald SWS1 abschaltet bricht die Versorgung des Elko ab, seine Entladung über EB beginnt. Mit dem Poti wird die Entladezeit eingestellt. Sie muss nicht besonders lang sein, weshalb im Gegensatz zu SWS1 kein Widerstand am Elko notwendig ist. Nach der Pause schaltet sich auch SWS2 aus. SWS1 und SWS2 sind damit beide abgeschaltet.

Schritt 4: SWS2 führt Polwende-Schaltung aus

Jetzt kommt Relais R2 ins Spiel. Solange SWS2 ein ist, ist R2 ein. In diesem Zustand (siehe Abb. 5) kann R2 den Seil-Motor nicht versorgen. Das tat bis jetzt R3. Dessen Part ist aber, sobald der Haken unten ist, fürs erste vorüber. Der Motor steht, die Pause beginnt und SWS2 ist noch an. Die Zeit läuft, sobald sich SWS1 ausschaltet (siehe Schritt 3).

Ist SWS2 aus, schaltet das R2 aus und damit den Seilmotor wieder ein. R2 ist so geschaltet, dass der Motor gleichzeitig umgepolt wird – der Haken geht nach oben. R2 wird ebenfalls mit einer Diode zur Schnellbremsung ausgestattet. Abweichend von R3 liegt sie hier an a3-b3 (in R2 dagegen an a2-b2). Die Abbildung zeigt uns, weshalb für das Schnellbremsen (oder Kurzschlussbremsen) des Motors unbedingt Dioden gebraucht werden. Betrachten wir folgende Situation: R3 ist aus (nach Haken unten) und R2 ist aus (Haken geht wieder nach oben). Die grüne Linie in Abb. 5 zeigt den Lauf des Stroms. Aus b1 von R2 (der Baustein ist spiegelverkehrt dargestellt) geht die Ladung über die Leitung nach R3 (b1), verläuft ohne Diode weiter nach R3 (a1) und somit in einen Kurzschluss. Die beiden Dioden an R2 und R3 verhindern dies und sorgen gleichzeitig für das Schnellbremsen (R3 bei Haken unten, R2 bei Haken oben).

Am Ende muss der Motor noch gestoppt werden. Der Haken ist dann oben angekommen. Dazu brauchen wir den SWS3 wiederum als Zeitschalter.

Schritt 5: Magnethaken rauf und am Ende stoppen

Warum eigentlich zwei Zeitschalter für ‚Haken ab' und ‚Haken hoch'? Grund dafür ist, dass der Elektromotor je nach Richtung unterschiedlich schnell laufen könnte. Deshalb ist es mit nur einem Zeitschalter kaum möglich, den Haken wieder auf dieselbe Höhe zu bekommen wie vor dem Herablassen. Mit zwei Schaltern kann die unterschiedliche Laufleistung des Motors berücksichtig werden. Leider berücksichtigt das noch nicht das

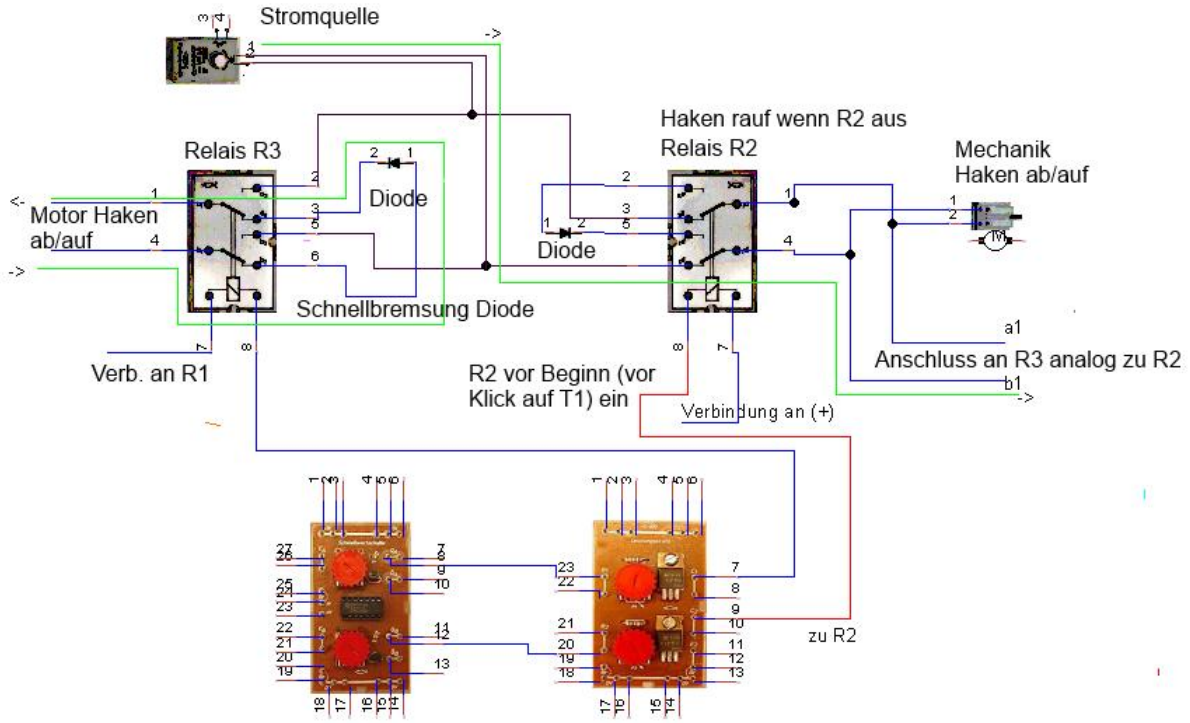


Abb. 5: SWS1/2 Anschluss an LST1/2 und R2/3 mit Selbsthaltedioden

Gewicht der Last. Deshalb darf die auch nicht zu schwer sein. Mit Hilfe eines Flaschenzugs kann aber auch dieser Effekt minimiert werden.

Der Zeitschalter für den Magnethaken nach oben befindet sich am SWS3 und wird vom SWS2 gestartet, genau wie der SWS1 den SWS2 startet. Der Unterschied ist, dass für den SWS3 zwei Elkos verwendet werden, weil die Spannung von QA(+) nicht so hoch ist wie von der (+)-Schiene selbst, was nur dem SWS1 zu Gute kam. Alle anderen werden über die Q-Ausgänge angesteuert. Der 10 kΩ-Entladewiderstand bremst zusätzlich das Entladen, und so kommen wir auf eine vernünftige Entladezeit. Der Anschluss über den Br-Pin ist auch hier nicht ratsam, weil es das genaue Einstellen der Ladezeit über das Poti erschwert.

SWS2 „aus“ schaltet wie festgestellt R2 aus. Der Haken geht solange nach oben, bis R2 wieder eingeschaltet wird, und das erledigt nach Ablauf der Einschaltzeit der SWS3. Von LST3 kann die Leitung direkt zur LST2 verlaufen, an der R2 bereits liegt.

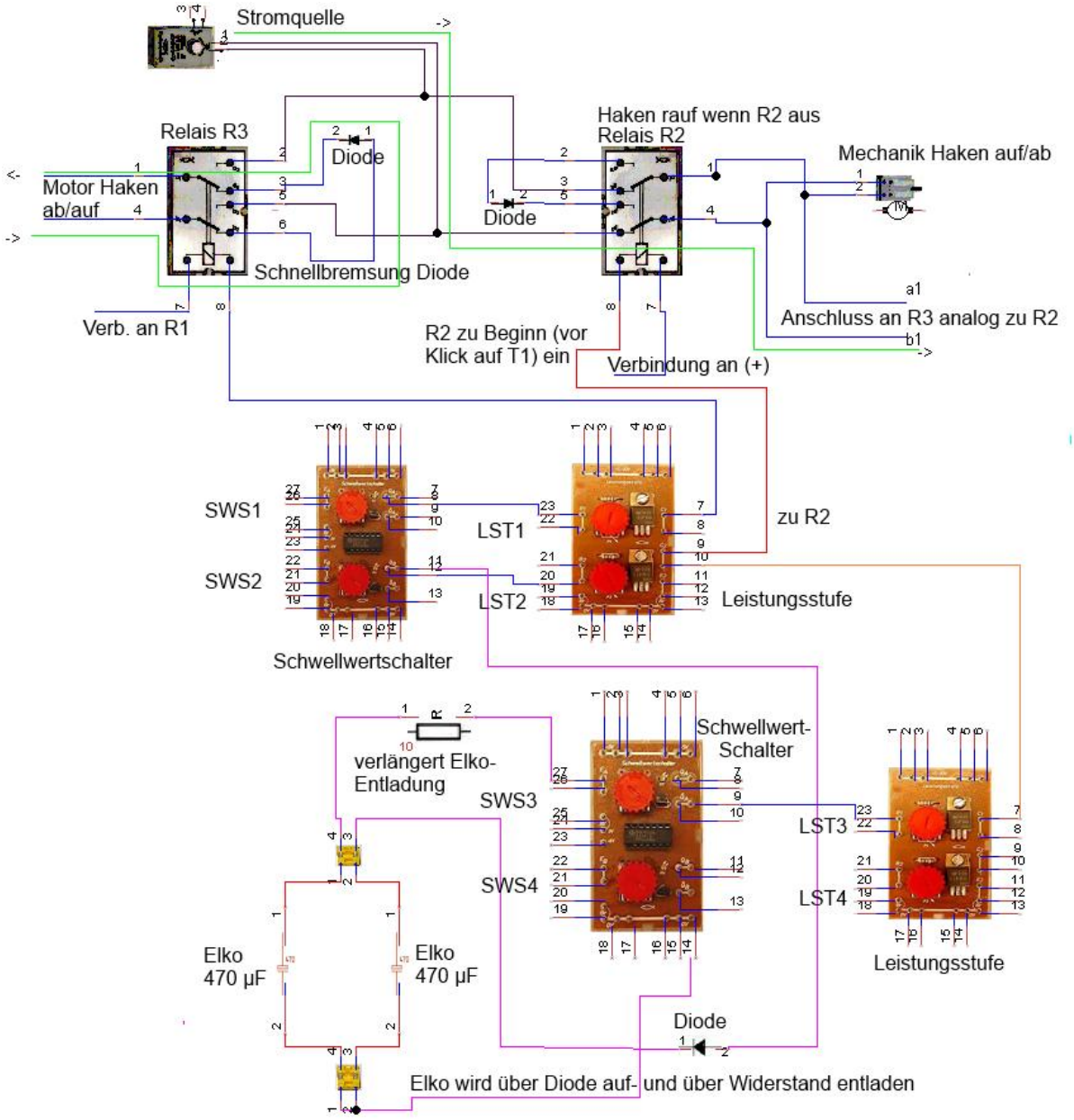


Abb. 6: Magnethaken oben stoppt

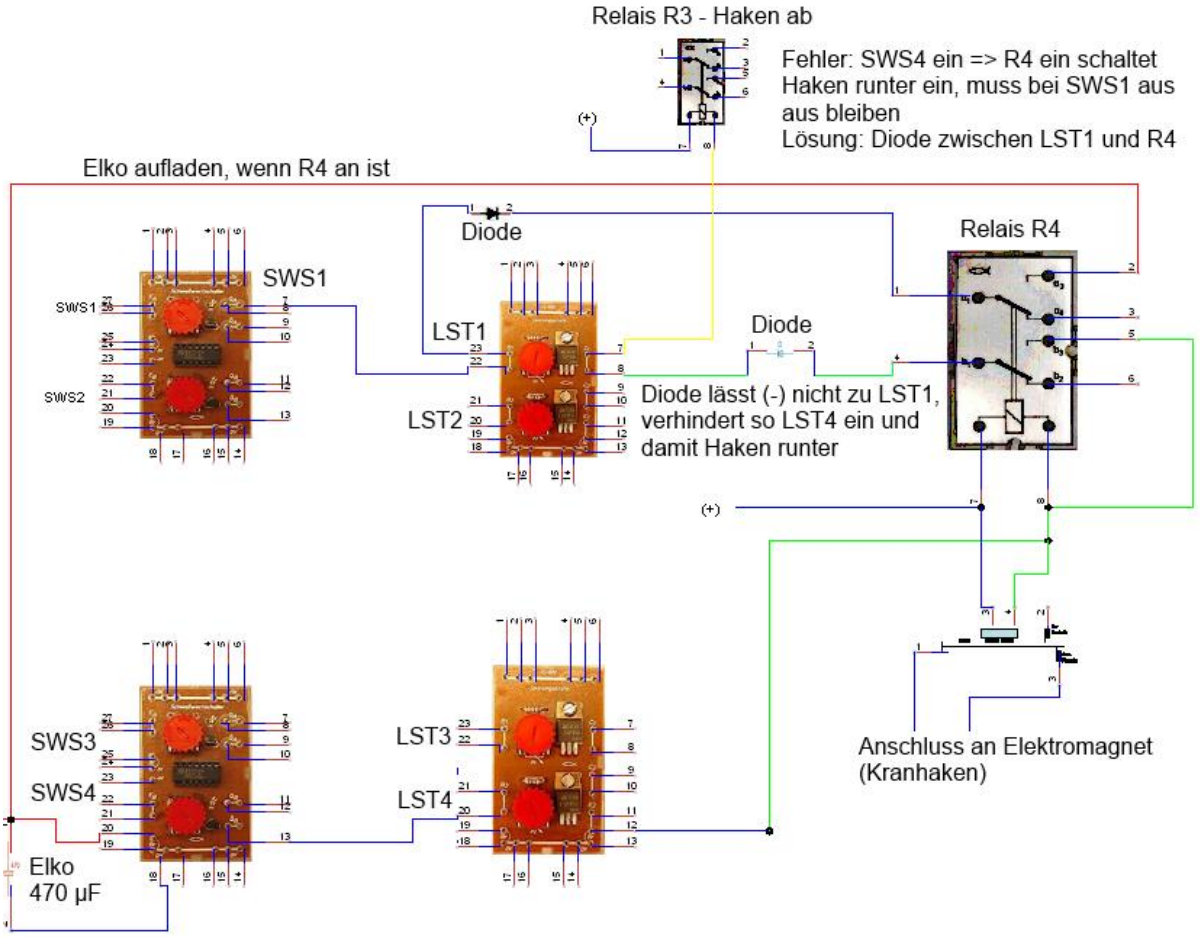


Abb. 7: Magnethaken ein/ausschalten

Da LST2 zu dem Zeitpunkt bereits aus ist, kann (-) von LST3 direkt R2 einschalten das sich ja immer an (+) befindet.

Schritt 6: Magnet Ein/Aus

Der Seilmotor hat alle seine Arbeitsschritte zur Aufnahme einer Last abgeschlossen. Was noch fehlt ist, dass der Magnet auch Strom erhält, um sie an sich zu heften und beim Abstellen auch wieder lösen zu können. Hier kommt uns die Motorpause von Schritt 3 zugute.

Der Ablauf sieht also wie folgt aus:

1. Haken ab
2. Motorstopp und Wartezeit
3. Einschalten des Elektromagneten zur Containerannahme
4. Haken rauf
5. Kran dreht
6. Haken wieder ab
7. Motorstopp und Wartezeit, Elektromagnet ausschalten, Container wird abgestellt
8. Haken wieder hoch

Ein kompletter Durchlauf des bisherigen Systems endet mit Punkt 4. Die nachfolgende Krandrehung startet durch automatisches Betätigen von T1 bei der Drehung. Das heißt aber auch, dass T1 beim Schwenk in die andere Richtung wieder gelöst und das System so wie geplant in den Ausgangszustand versetzt wird. Der Elektromagnet muss über diesen Zustand hinaus eingeschaltet bleiben, bis der Container abzustellen ist.

Wieder verwenden wir einen Zeitschalter, diesmal auf SWS4. Das Anbringen mit (+) direkt an dem Br-Pin sorgt für eine lange Einschaltzeit, die wir jetzt brauchen. Der Elko wird vom SWS1 (QA) aufgeladen.

Die Entladung beginnt deshalb, sobald SWS1 ausgeht, was just in dem Augenblick geschieht, wenn der Haken unten ist.

Einfacher wäre es natürlich, wenn der Elektromagnet die ganze Zeit eingeschaltet ist und nur ausgeht, sobald ein Container abgestellt wird. Dabei würde ein Klick auf einen Taster genügen, der sich z. B. unter der Plattform befindet, auf der er abgestellt wird. Aber die rein elektronische Steuerung erspart diese zusätzliche Konstruktion und ist eleganter.

SWS1 „aus“ startet also das Entladen des Elkos, des Zeitschalters von SWS4 durch Mithilfe eines weiteren em3-Relais R4. SWS1 „aus“ schaltet R4 aus. Das trennt zeitgleich die (+) Versorgung des Elkos über die Arbeitskontakte a1-a3 und schaltet den Elektromagneten ein. SWS1 schaltet sich jedoch nach dem Lösen von T1 durch den Kranschwenk wieder ein – was an dieser Stelle nicht sein darf, da sich ja der Container am Magneten befindet. Deshalb verläuft die (-) Leitung von LST1 nach R4 durch seine Arbeitskontakte b1-b3. Einmal ausgeschaltet, kann die LST1 es nicht mehr einschalten.

Erst mit Ablauf des Zeitschalters SWS4 schaltet LST4 R4 wieder ein und damit den Magneten aus. Das muss exakt in der Motorpause erfolgen, nach dem Kranschwenk und erneutem Herablassen des Hakens (Schritt 3). Es ist also etwas Geduld nötig, um SWS4 über seinen Poti richtig einzustellen; das hängt davon ab, wie lange die Schritte „Haken ab, auf, Krandrehung, Haken wieder ab“ dauern.

Leider sind alle Arbeitskontakte von R4 damit belegt. Infolge des Aufbaus des gesamten Systems kann sich R4 bei SWS1/LST1 „aus“ nicht einschalten, sondern geht aus. Das ist leider nicht zu ändern, denn LST1 muss am Ende ausgehen, um R3 für den Motorstopp nach dem Ablassen des Hakens auszuschalten.

Es wäre allerdings praktisch, denn so könnte der Elektromagnet einfach parallel an R4 angeschlossen werden. Es würde sich dann ganz einfach mit ihm ein und ausschalten. Da das nicht so ist, benötigen wir einen weiteren Schalter: das Eigenbau-Relais, das parallel an R4 geschaltet wird. Bei R4 „aus“ schaltet es sich ebenso aus und den Magneten damit ein. Abb. 7 zeigt die Schaltung.

Unbedingt zu beachten ist die Diode zwischen LST1 und Arbeitskontakt b1 von R4. Ohne sie würde nach dem Einschalten von LST4 das Relais R4 mit LST1 verbunden sein und darüber R3 und somit den Seilmotor ungewollt einschalten.

Schritt 7: Die ganze Anlage starten und den Kran drehen

Jetzt kommen wir endlich zum letzten Schritt der Kransteuerung: das Drehen des Krans und das Starten der bisherigen Anlage überhaupt, sobald ein Container anrollt.

Beginnen wir mit dem Schwenken des Krans. Nach dem Abstellen des letzten Containers befindet sich der Haken wieder oben. Bei der Ankunft eines neuen Containers (angeliefert z. B. durch die Bau-Spiel-Bahn) muss der Kran als erstes zum neuen Container gedreht werden. Dazu verwenden wir die – wie dafür gemachte – Polwendeschaltung auf Seite 54 des em3-Handbuchs ([3], Abb. 8).

Die Taster werden so am Kran angebracht, dass er sie selbst beim Drehen betätigt – am besten gegenüberliegend, dann ist ein 180°-Schwenk möglich.

In dieser Form würde der Kran jedoch ewig hin- und herschwenken. Deshalb sind noch ein paar Modifikationen nötig. Die wichtigste ist ein Reedkontakt. Das Einschalten des Reeds übernimmt ein Magnet am Container, sobald dieser anrollt. Dieser Magnet muss so am Container angebracht

werden, dass er nahe am Reed stehenbleibt.

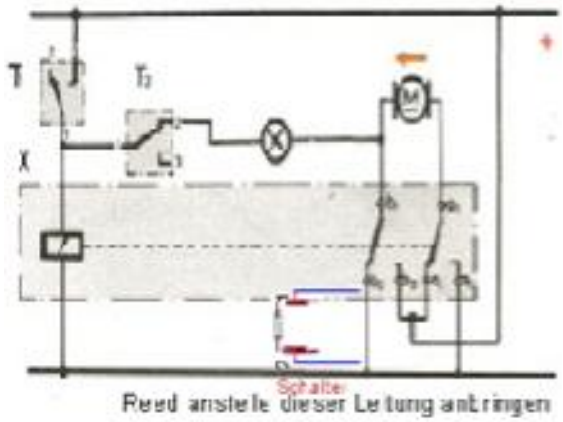


Abb. 8: Polwenden em3 ([3], S. 54)

Je nach Polung (der Anschluss von R4 sollte am gleichen Trafo wie der Seilmotor erfolgen) ist er entweder am b2- oder b3-Kontakt anzubringen, sodass er den Kontakt zum Schwenk-Motor unterbricht, sobald der Container abgesetzt wird, und durch den Schwenk zuvor die Polwendung erfolgt.

Zwar wird der Reed schon ausgeschaltet, sobald der Container hochgezogen wird, das ist aber kein Problem, denn der Schwenk hin zum Container erfolgte vorher und wird jetzt erst geblockt. Beim Absetzen und Umpolen, damit der Schwenk zurück zum nächsten anrollenden Container erfolgen kann, wirkt das Blockieren, bis der nächste Container eingetroffen ist und der Schaltmagnet dort die Blockierung aufhebt. Nach Absetzen der Last und Hochziehen des Magnethakens verbleibt der Kran in dieser Position.

Beim Schwenk zurück wird ein weiterer Taster am Ende gleichzeitig mit dem Polwende-Taster gedrückt, der die Anlage startet (der Haken beginnt sich zu senken). Der Kran darf sich aber erst nach der Aufnahme drehen. Wir brauchen deshalb einen Impulsverkürzer wie auf Seite 56 im ec2-Handbuch ([4], Abb. 9):

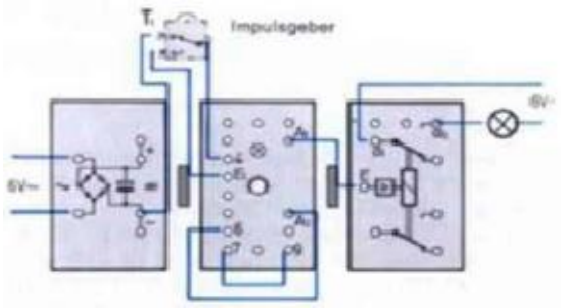


Abb. 9: Impulsverkürzer von Seite 56 des ec2-Handbuchs [4]

Durch den Schwenk muss er aber von zwei Tastern gestartet werden, wie Abb. 10 zeigt.

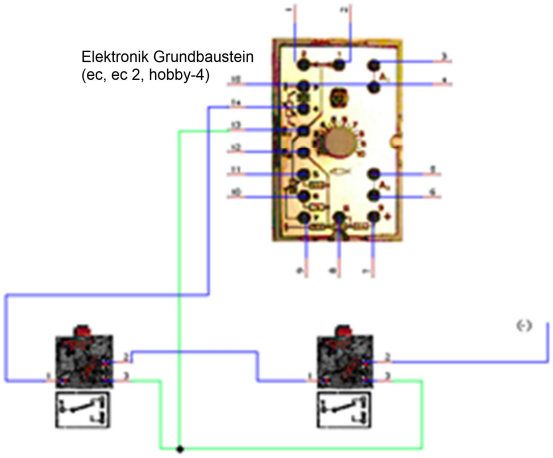


Abb. 10: Impulsverkürzer von zwei Tastern aus starten

Anstatt die Anlage direkt über einen Taster am Kran zu starten, verwenden wir die beiden Impulstaster zum Starten vom Impulsverkürzer und der Anlage.

Das Relais verwenden wir als Schalthilfe. T1 von Abb. 2 ersetzen wird durch eine der beiden Relais-Schaltungen. Die andere verwenden wir, um den Schwenkmotor für die Dauer seiner Einschaltzeit zu unterbrechen. Dazu muss nur ein Kabel des Schwenkmotors z. B. durch die Kontakte a1-a2 verlaufen. Sobald sich das ec-Relais einschaltet, wird der Drehmotor des Krans für diese Dauer unterbrochen. Weil der Impulsverkürzer (IP) das Einschalten der Anlage übernimmt, brauchen wir also zusammen vier Taster: zwei für den Polwender (Abb. 7) und zwei für den IP.

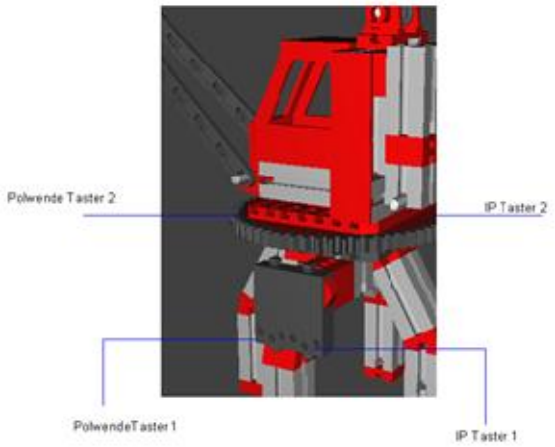


Abb. 11: Schalttaster am Kran

Abb. 11 zeigt grob, in welcher Position sie sich am Kran befinden sollten. Auf jeder Seite sollten je ein Taster aus Abb. 7 und Abb. 8 so miteinander verbunden werden, sodass sie gleichzeitig gedrückt werden.

So wie in der Abbildung grob dargestellt müssen auf beiden Seiten die Taster angebracht werden (Taster auf der gegenüberliegenden Seite im Bild nicht sichtbar). Ein Nocken oder Baustein direkt am Kran drückt dann die Taster am Ende jeder Drehung zu einer der Seiten.

Quellen

[1] Werner Hasselberg: *Automatik für weichen Motorstart und –stopp*, ft:pedia 3/2013, S. 30-35.

[2] Fischer-Werke: *Anleitung zum Elektronik-Ergänzungskasten*, Tumlingen, 1981.

[3] Fischer-Werke: *Anleitung zum em3 Elektromechanik-Grundkasten*, Tumlingen, 1975.

[4] Fischer-Werke: *Anleitung zum ec2 Elektronik-Grundkasten*, Tumlingen, 1975.

Computing

I²C mit dem TX – Teil 7: Real Time Clock (RTC)

Dirk Fox

Seit der Einführung in die Grundlagen des I²C-Protokolls in ft:pedia 3/2012 [3] haben wir in unserer I²C-Serie schon einige Sensoren und Aktoren vorgestellt, die sich an den TX anschließen und in Robo Pro-Programmen nutzen lassen. In diesem Beitrag stellen wir einen Aktor vor, der z. B. unsere ft-Funkuhr aus ft:pedia 3/2012 [1] perfekt ergänzt: eine Batterie gepufferte Echtzeituhr.

Ein Praxistipp vorweg

Nachdem ich inzwischen zahlreiche Varianten ausprobiert habe, um I²C-Sensoren möglichst elegant mit dem EXT-2-Anschluss des TX zu verbinden, möchte ich euch eingangs eine besonders einfache und universell einsetzbare Lösung vorstellen.

Für diese Variante benötigt ihr lediglich eine 6polige Pfostenbuchse (ca. 0,25 €) und vier verschiedenfarbige Female-Jumper (F/F, Abb. 1). Wenn ihr die Jumper in der Mitte durchschneidet, könnt ihr die zweite Hälfte für ein weiteres Anschlusskabel verwenden.

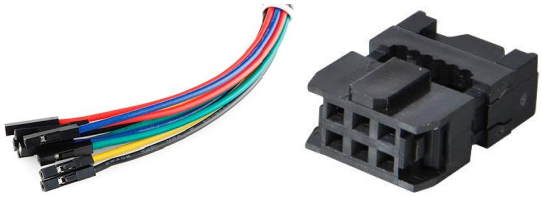


Abb. 1: F/F-Jumper, 6polige Pfostenbuchse

Wählt die Jumper-Kabel möglichst in den von fischertechnik gewohnten Farben (rot für VCC, grün für GND und schwarz für den Datenanschluss SDA, sowie eine beliebige vierte Farbe für den Takt SCL), damit ihr die Anschlüsse auch ohne Beschriftung sofort zuordnen könnt. Dann klemmt ihr die abgeschnittenen Kabelenden in der richtigen Reihenfolge in den Pfostenstecker, und ihr haltet nach wenigen Minuten ein Universal-I²C-Anschlusskabel in den Händen (Abb. 2).

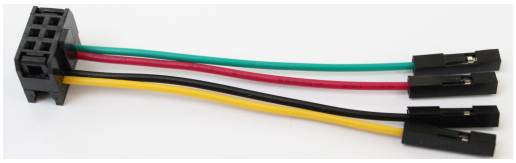


Abb. 2: Universal-I²C-Anschlusskabel

Vorteil dieser Lösung: Sofern das I²C-Modul mit angelöteter Siftleiste an den Anschlüssen geliefert wird, kommt ihr gänzlich ohne Lötkolben aus, das Anschlusskabel kommt mit jeder Anordnung der Kontakte am I²C-Modul klar und ist zudem konkurrenzlos günstig.

Real Time Clock (RTC)

Für zahlreiche Anwendungen, die mit Zeitangaben arbeiten oder Zeiten messen, ist eine exakte Uhr hilfreich. Der Timer von Robo Pro ist dafür zu ungenau; zudem kennt er weder Datum noch Uhrzeit und „vergisst“ die Zeit nach dem Ausschalten. Eine Echtzeit-Uhr sollte daher über eine unabhängige Stromversorgung verfügen und die Möglichkeit bieten, Weckzeiten (Alarme) zu speichern. Sie ließe sich zum Beispiel mit dem Atom-Zeit-Empfänger exakt einstellen, den wir in ft:pedia 3/2012 vorgestellt haben [1]. Zwei solcher I²C-Echtzeituhren werden im Folgenden vorgestellt.

Echtzeit-Uhr DS1307

Ein sehr häufig anzutreffender *Real Time Clock* (RTC) IC mit I²C-Schnittstelle ist der DS1307 von Dallas Semiconductor aus dem Jahr 2001. Er speichert Uhrzeit und Datum, korrigiert Monate mit weniger als 31 Tagen, erkennt Schaltjahre bis zum Jahr 2100 und berechnet den aktuellen Wochentag. Die Uhrzeit kann wahlweise in 24- oder in 12-Stunden-Darstellung (mit AM/PM-Anzeige) gespeichert werden. Zeit- und Datumsangaben werden im BCD-Format dargestellt [4].

Abb. 3: I²C-Board mit RTC-Controller DS1307 (Quelle: watterott.com)

Für etwa 15 € bieten z. B. watterott electronic und EXP unter der Bezeichnung BOB-00099 ein Board von Sparkfun mit dem RTC-Chip DS1307 an (Abb. 3). Es arbeitet mit einer Eingangsspannung von 5 V, lässt sich also ohne Spannungs- und Pegelwandler direkt an den TX anschließen. Die I²C-Kontakte sind auf der Platine gekennzeichnet, sodass der Anschluss nach Einlöten der Stiftleiste keine Herausforderung darstellt. Der Stromverbrauch des DS1307 ist mit 0,5 mA äußerst gering; die Lebensdauer der auf der Rückseite der Platine befestigten Lithium-Knopfzelle CR1225 wird vom Hersteller daher mit 9-17 Jahren angegeben – für ein ft-Modell höchstwahrscheinlich ausreichend.

Zeit und Datum werden in einem sieben Byte großen Register (Adresse 0x00 bis 0x06) gespeichert: Sekunden, Minuten, Stunden, Wochentag, Datum, Monat und Jahr, gefolgt von einem (für uns nicht relevanten) *Control Byte* (Abb. 4). Das Sekunden-Byte enthält als höchstwertiges Bit ein *Clock Halt* (CH) Bit – ist es gesetzt, wird der Oszillator angehalten. Bit 6 des Stunden-Bytes enthält die Auswahl des 12/24-Stunden-Formats; falls Bit 6 = 1 (12-Std.-Format) ist Bit 5 das AM/PM-Bit (0/1). Vom Wochentag-Byte werden nur die niederstwertigen drei Bits genutzt. Das Jahresbyte enthält die letzten beiden Dezimalstellen der Jahreszahl (Abb. 4).

	BIT7							BIT0
00H	CH	10 SECONDS			SECONDS			
	0	10 MINUTES			MINUTES			
	0	12/24	10 HR A/P	10 HR	HOURS			
	0	0	0	0	0	DAY		
	0	0	10 DATE		DATE			
	0	0	0	10 MONTH	MONTH			
	10 YEAR				YEAR			
07H	OUT	0	0	SQWE	0	0	RS1	RS0

Abb. 4: 8-Byte-Register des DS1307 [4]

Alle Zeitangaben sind BCD-kodiert gespeichert. Unter den Adressen 0x08 bis 0x3F stehen weitere 56 Byte RAM zur Verfügung, in denen eine Anwendung z. B. Zwischen- oder Alarmzeiten speichern kann.

I²C-Protokoll

Der RTC-Controller DS1307 unterstützt das I²C-Protokoll im *Standard Mode* (100 kHz). Der Controller hat die feste *Slave*-Adresse 0x68 (= 1101 000); an einem I²C-Bus können daher (ohne Multiplexer) nicht mehrere RTC-Module getrennt angesprochen werden.

Der DS1307 beherrscht lediglich zwei Befehle (Modi): einen Schreib- und einen Lese-Modus, in denen Uhrzeit und Datum eingestellt bzw. ausgelesen werden. In beiden Modi wird das gerade geschriebene

bzw. auszulesende Datenbyte gepuffert; die Puffer werden beim I²C START-Kommando synchronisiert.

Auf diese Weise wird ein Überschreiben der Daten durch die Uhr während eines Schreib- oder Lesevorgangs verhindert. Allerdings schließt das die Übertragung von mehreren aufeinander folgenden Bytes mit dem *repeated START*-Kommando (in Robo Pro aktiviert durch das Kästchen „Offen lassen“, siehe ft:pedia 3/2012 [3]) aus, da der Puffer sonst vom nächsten Eintrag überschrieben wird.

Stellen der RTC

Zum Stellen der Uhr wird Byte für Byte ein *Write*-Befehl an die *Slave*-Adresse 0x68 geschickt, gefolgt von der Register-Adresse, in die geschrieben werden soll (*Subaddress*) und dem BCD-kodierten zugehörigen Bytewert des Uhrzeit- bzw. Datumselements: die Sekunden in Register 0x00, die Minuten in 0x01, die Stunden in 0x02, der Wochentag in 0x03 und Datum, Monat und Jahr in die Register 0x04 bis 0x06. Liegen die Zeit- und Datumsangaben nicht im BCD-Format vor, müssen sie zuvor konvertiert werden.

Soll die Zeit im 12-Stunden-Format gespeichert werden, ist im Stunden-Byte Bit 6, für eine Stundenangabe nach Mittag außerdem Bit 5 für ‚PM’ zu setzen. Die (sequentielle) Nummerierung der Wochentage von 1 bis 7 ist nicht vorgegeben; nach dem Stellen der Uhr wird der Wochentag jeweils um Mitternacht erhöht.

Das *Clock Halt*-Bit (CH) im Sekunden-Byte schaltet den Oszillator ab. Damit kann das RTC-Modul mit eingebauter Batterie geliefert werden, ohne dass es Strom verbraucht. Das CH-Bit ermöglicht es außerdem, die Uhr nach dem Stellen zu einem exakten Zeitpunkt zu starten. Diese Option ist interessant, wenn man die Uhr nach der Normalzeit des DCF77-Signals stellen möchte: Damit kann man beim Empfang des ersten Sekunden-Signals der neuen Minute die RTC exakt starten. Oder man nutzt das CH-Bit, um die RTC als Stoppuhr zu starten (siehe unten).

Auslesen der RTC

Das Auslesen der Zeit erfolgt mit einem *Read*-Befehl an die *Slave*-Adresse 0x68, an den sich die Angabe der auszulesenden Register-Adresse anschließt. So können die BCD-Werte Byte für Byte (und Befehl für Befehl) ausgelesen werden.

Liegen die Daten in 12-Stunden-Darstellung vor, wird dies über ein Flag zurückgemeldet und das AM/PM-Flag gesetzt (0 = AM, 1 = PM).

Speicher

Die Verfügbarkeit von 56 Byte nicht-flüchtigen, direkt adressierbaren Speichers ist besonders interessant, da die Adressen nicht im Adressbereich des TX-EEPROMs (0x50-0x57) liegen und daher ohne Adresskollision genutzt werden können.

Zum Speichern von Zeiten (Stunden, Minuten und Sekunden) genügen drei Byte. Will man die RTC beispielsweise zur Messung von Rundenzeiten vier verschiedener Rennwagen nutzen, kann man je Fahrzeug 14 Runden speichern.

Die Speicherung eines Alarm-Zeitpunkts (Zeit und Tag) erfordert sieben Byte; daher lassen sich bis zu acht solcher Alarme im Speicher ablegen.

Robo Pro-Treiber

Ein sehr einfacher Robo Pro-Treiber für den DS1307 von Rei Vilo wird mit der aktuellen Version von Robo Pro in der Bibliothek (Rubrik I²C) ausgeliefert. Der Treiber kann die RTC stellen und auslesen; die Werte werden automatisch in das bzw. aus dem BCD-Format umgewandelt. Der Treiber funktioniert allerdings nur bei Zeitangaben im 24-Stunden-Format korrekt.

Im Downloadbereich der ft:c findet sich ein angepasster DS1307-Treiber, der auch

das 12-Stunden-Format beherrscht und die Zeitangaben im BCD-Format erhält bzw. ausgibt (siehe Abb. 5). Das ist von Vorteil, wenn man beispielsweise die Echtzeituhr nach dem DCF77-Signal (Normalzeit) stellen oder Uhrzeit und Datum auf einem 7-Segment-LED-Display ausgeben möchte, da in beiden Fällen BCD-kodierte Werte benötigt werden.

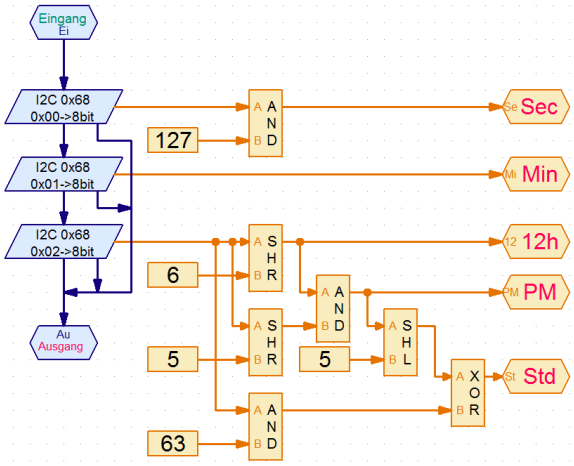


Abb. 5: Auslesen der Zeit sowohl im 12 h- als auch im 24 h-Format als BCD-Werte

Als Anwendungsbeispiel liegt dem Robo Pro-Treiber ein ft-Funkuhr-Programm bei, das auf dem in ft:pedia 3/2012 vorgestellten Programm für einen DCF77-Empfänger aufbaut und auch bei schwachem DCF77-Signal weiterläuft: Wie die originalen Junghans-Funkuhren wird die RTC zu jeder vollen Stunde mit der von Mainflingen bei Frankfurt ausgesandten Normalzeit abgeglichen, sofern das DCF77-Signal fehlerfrei empfangen werden kann. Die Zeitanzeige erfolgt sowohl auf dem Display des TX als auch – sofern angeschlossen – auf bis zu drei LED-Displays (SAA1064) [2], deren Helligkeit mit den Steuertasten des TX verstellt werden kann.

Ein weiteres kleines Beispielprogramm, das ebenfalls dem Treiber beiliegt, zeigt den Nutzen des CH-Bits: Es startet die RTC auf Knopfdruck als Stoppuhr (Abb. 6). Das Programm lässt sich leicht erweitern, z. B. zu einer automatischen Zeitmessung für die Carrerabahn: Reed-Sensoren oder Lichtschranken stoppen die Rundenzeiten, der TX berechnet die Durchschnittswerte und zeigt die schnellste Runde im LED-Display an.

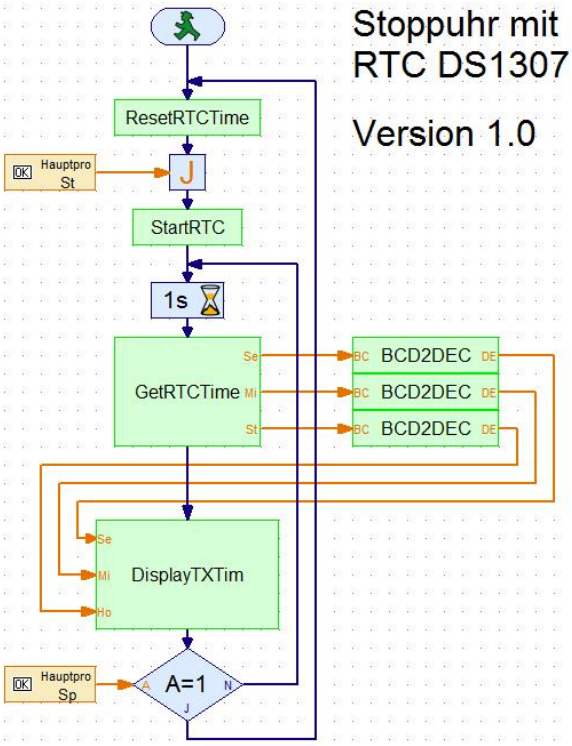


Abb. 6: Beispielprogramm „Stoppuhr“

Echtzeit-Uhr DS3231

Ein ‚große Bruder’ des DS1307 ist der RTC-Controller DS3231 aus dem Jahre 2005. Für knapp 17 € bietet EXP das *Breakout Board* CronoDot v2.1 an.

Abb. 7: ChronoDot (DS3231) (Quelle: exp-tech)

Auch dieses RTC-Board (Abb. 7) wird mit einer Knopfzelle (CR2016) geliefert, deren Lebenszeit mit mindestens acht Jahren angegeben wird. Der DS3231 verträgt Eingangsspannungen von 2,3 bis 5,5 V und kann daher ebenfalls direkt am TX betrieben werden. Er schaltet automatisch auf Batteriebetrieb, wenn die Betriebsspannung unter den Mindestwert fällt. Der Controller gleicht über einen Temperatursensor alle 64 Sekunden Temperatur bedingte Schwankungen des Oszillators aus und erreicht so eine deutlich höhere Genauigkeit. Außerdem verfügt er über zwei Alarm-Register.

Mit 0,84 mA ist sein Ruhestromverbrauch etwas höher als der des DS1307. Dank eines zusätzlichen Century-Bits können sogar Datumswerte bis zum 31.12.2199 dargestellt werden.[1]

I²C-Protokoll

Der RTC-Controller DS3231 unterstützt den I²C *Fast Mode* (400 kHz). Der Controller hat dieselbe – ebenfalls unveränderliche – I²C-Adresse wie der DS1307 (0x68 = 1101 000) und kann daher nicht mit diesem an demselben Bus betrieben werden. Auch der DS3231 kennt nur zwei Modi: Schreiben (zum Stellen der Uhr bzw. eines Alarms) und Lesen (zum Auslesen der Uhrzeit bzw. des Status-Bytes).

Nach dem Einschalten soll man dem RTC-Controller vor dem ersten Befehl 256 bis 300 ms Zeit zur Stabilisierung der Stromversorgung einräumen.

Stellen der RTC

Die Kodierung der Uhrzeit in den Registern 0x00 bis 0x06 entspricht – bis auf das nicht mehr unterstützte *Clock Halt*-Bit – der des DS1307 (Abb. 8). Daher kann der Schreibbefehl des DS1307-Treibers auch für die Eintragung der Uhrzeit im DS3231 verwendet werden. Will man die höhere Busgeschwindigkeit ausreizen, müssen die I²C-Befehle auf 400 kHz Übertragungsgeschwindigkeit umgestellt werden. Die höhere Taktrate kollidiert nicht mit langsameren Sensoren auf demselben Bus, wie zum Beispiel den SAA1064-LED-Displays, auf denen die ft:pedia-Funkuhr Zeit und Datum ausgibt und die nur den I²C *Standard Mode* beherrschen [2].

Alarme

Statt eines frei belegbaren RAM-Bereichs verfügt der DS3231 über sieben Register zur Speicherung von zwei Alarmen: einem sekunden- und einem minutengenauen. Alarm-Tag und -Zeitpunkt werden jeweils in den Registern 0x07 bis 0x0A (Alarm 1) bzw. 0x0B bis 0x0D (Alarm 2) gespeichert. Die Struktur des Registerinhalts entspricht bei Alarm 1 (A1) den ersten vier (Sekunden, Minuten, Stunden und Tag), bei Alarm 2 (A2) dem zweiten, dritten und vierten Byte (Minuten, Stunden und Tag) des RTC-Registers. Die Angaben werden ebenfalls BCD-kodiert gespeichert.

Statt des Wochentags kann im vierten Register von A1 bzw. im dritten Register von A2 jeweils auch das Datum gespeichert werden; in diesem Fall wird zusätzlich Bit 6 gesetzt. Außerdem wird der Alarm-Typ in vier (A1) bzw. drei (A2) Konfigurationsbits – den höchstwertigen Bits der Alarm-Register – festgelegt:

- Sind alle vier Bits gesetzt, so erfolgt der Alarm jede Sekunde (A1).
- Sind die ersten drei Bits gesetzt, wird der Alarm ausgelöst, wenn der Sekundenwert übereinstimmt (A1) bzw. erfolgt der Alarm jede Minute (A2).
- Sind die ersten beiden Bits gesetzt, wird der Alarm ausgelöst, wenn Sekunden- und Minuten- (A1) bzw. nur die Minu-

[1] Um den Robo Pro-Treiber nicht zu kompliziert zu gestalten, wird darin das Century-Bit allerdings nicht ausgewertet.

ADDRESS	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0 LSB	FUNCTION	RANGE
00h	0	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Hours	1–12 + $\overline{AM}$/PM 00–23
03h	0	0	0	0	0	Day			Day	1–7
04h	0	0	10 Date		Date				Date	01–31
05h	Century	0	0	10 Month	Month				Month/ Century	01–12 + Century
06h	10 Year				Year				Year	00–99
07h	A1M1	10 Seconds			Seconds				Alarm 1 Seconds	00–59
08h	A1M2	10 Minutes			Minutes				Alarm 1 Minutes	00–59
09h	A1M3	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Alarm 1 Hours	1–12 + $\overline{AM}$/PM 00–23
0Ah	A1M4	DY/$\overline{DT}$	10 Date		Day				Alarm 1 Day	1–7
					Date				Alarm 1 Date	1–31
0Bh	A2M2	10 Minutes			Minutes				Alarm 2 Minutes	00–59
0Ch	A2M3	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Alarm 2 Hours	1–12 + $\overline{AM}$/PM 00–23
0Dh	A2M4	DY/$\overline{DT}$	10 Date		Day				Alarm 2 Day	1–7
					Date				Alarm 2 Date	1–31
0Eh	$\overline{EOSC}$	BBSQW	CONV	RS2	RS1	INTCN	A2IE	A1IE	Control	—
0Fh	OSF	0	0	0	EN32kHz	BSY	A2F	A1F	Control/Status	—
10h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	Aging Offset	—
11h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	MSB of Temp	—
12h	DATA	DATA	0	0	0	0	0	0	LSB of Temp	—

Abb. 8: Register des DS3231 [5]

tenangabe (A2) des Alarmzeitpunkts mit der Uhrzeit übereinstimmen.

- Ist nur ein Bit gesetzt, muss außerdem die Stundenangabe übereinstimmen.
- Ist kein Bit gesetzt, muss auch der Wochentag (bzw. das Datum) übereinstimmen.

Das Erreichen eines Alarm-Zeitpunkts wird in Bit 0 (A1) und Bit 1 (A2) des Status-Registers 0x0F signalisiert: Hat eines dieser beiden Status-Bits den Wert eins, so wurde der entsprechende Alarm-Zeitpunkt erreicht. Ein Wecker muss daher regelmäßig (mindestens einmal je Sekunde) das Status-Byte via I²C auslesen.

Auslesen der RTC

Empfängt der Controller einen Auslesebefehl, dann werden die sieben Byte des Registers in ein „Ausleseregister“ kopiert, damit Aktualisierungen der RTC während des Auslesevorgangs nicht zu fehlerhaften Zeitangaben führen.

Temperatur

Die vom integrierten Temperatur-Sensor gemessene Temperatur, mit der der Ausgleich thermischer Oszillatorschwankungen berechnet wird, kann auch via I²C-Protokoll ausgelesen werden. Beim Einschalten der Stromzufuhr enthalten die Temperatur-Register den Wert 0°; anschließend wird alle ca. 64 Sekunden eine Temperaturmessung durchgeführt. Sie kann durch ein Setzen des CONV-Flags – Bit 6 im Control-Byte 0x0E – auch zwischen den automatischen Berechnungen ausgelöst werden. Der 10-bit-Temperaturwert mit einer Auflösung von 0,25° steht in den Registern 0x11 und 0x12: Der ganzzahlige Teil in Register 0x11, der Nachkommawert in Bit 6 und 7 von Register 0x12, kodiert als Zahl der 0,25°-Schritte. Die Bestimmung der Temperatur und die Umrechnung des Einflusses auf den Oszillator benötigen 125-200 ms. Die Auswertung des Temperaturwerts in Robo Pro zeigt Abb. 9.

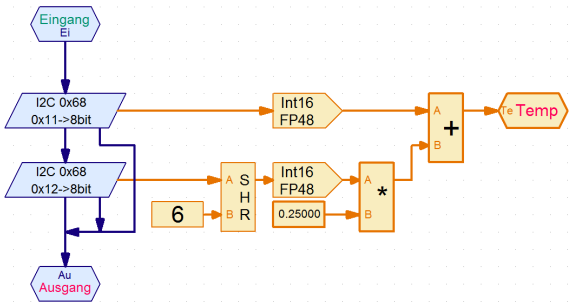


Abb. 9: Auslesen des Temperatur-Messwerts des DS3231 in Robo Pro

Die Genauigkeit des Temperatur-Sensors ist allerdings nicht umwerfend – nach Datenblatt liegt sie bei +/- 3°C [5].

Robo Pro-Treiber

Ein RoboPro-Treiber für den DS3231 ist ebenfalls im Downloadbereich der ft:c zu finden, zusammen mit einer angepassten Erweiterung der fischertechnik-Funkuhr (DCF77-Empfänger [1]). Wie bei der Variante für den DS1307 wird die RTC stündlich mit der Normalzeit abgeglichen; außerdem zeigt die Funkuhr für den DS3231 die Raumtemperatur an.

Die Nutzung der Alarm-Funktion ist wegen der Beschränkung der ‚Benutzerschnittstelle' auf die beiden roten Knöpfe des TX allerdings sehr umständlich – hier wäre ein numerisches Tastenfeld als Eingabe-Einheit eine feine Sache. Daher verschieben wir die Einstellung eines Alarms an der RTC auf einen späteren Beitrag – in dem wir ein über das I²C-Protokoll ansprechbares Tastenfeld vorstellen werden.

Jetzt aber erstmal viel Vergnügen mit dieser Mechanik freien fischertechnik-Uhr.

Vielleicht gelingt es ja auch jemandem von euch, damit eine mechanische ft-Uhr exakt zu stellen?

Quellen

[1] Dirk Fox, Dirk Ottersmeyer: *Bau einer ft-Funkuhr*. ft:pedia 3/2012, S. 4-10.

[2] Dirk Fox: *I²C mit dem TX – Teil 2: LED-Display*. ft:pedia 4/2012, S. 32-37.

[3] Dirk Fox: *I²C mit dem TX – Teil 1: Grundlagen*. ft:pedia 3/2012, S. 32-37.

[4] Dallas Semiconductors: *DS1307 64 x 8 Serial Real-Time Clock*. Datasheet, Rev. 5, February 2008.

[5] Maxim Integrated: *DS3231 Extremely Accurate I²C-Integrated RTC/TCXO/Crystal*. Datasheet, Rev. 9, January 2013.

Computing

I²C mit dem TX – Teil 8: Ultraschall-Sensor

Dirk Fox

Der Abstandssensor des TX misst die Distanz zu einem Objekt via Ultraschall und liefert das Ergebnis in cm. Der Sensor lässt sich sehr einfach aus Robo Pro ansprechen und gut im ft-Raster verbauen [1]. Allerdings gibt es Ultraschall-Sensoren mit I²C-Schnittstelle, die über die eine oder andere Zusatzfunktion verfügen und sich zudem in größerer Zahl an den I²C-Bus des TX anschließen und auswerten lassen. Für autonome Roboter sind sie eine Bereicherung.

Hintergrund

Abstandssensoren sind vor allem für autonome Roboter-Modelle von großer Bedeutung: Sie ermöglichen eine räumliche Orientierung im Nahfeld (wenige Zentimeter bis mehrere Meter) und die Erkennung von Hindernissen und Objekten. Der Ultraschall-Abstandssensor von fischertechnik muss sich den Anschluss am TX jedoch mit anderen Sensoren teilen – und ist in seiner Funktionalität auf die Bestimmung des Abstands zum nächsten reflektierenden Objekt in *cm*-Schritten begrenzt.

Daher lohnt ein Blick auf andere, im Modellbau verbreitete I²C-Ultraschallsensoren, die sich mit dem TX nutzen lassen, zumal über den fischertechnik-Abstandssensor – abgesehen vom Messbereich (3 bis 400 *cm*) [2] – keine technischen Details bekannt sind: weder das Signalisierungsprotokoll[2] noch die Dauer des Messvorgangs, die Stromaufnahme oder die Richtcharakteristik des Sensors.

Von der britischen Firma Devantech Ltd. werden unter der Marke Robot Electronics zahlreiche Ultraschall-Sensoren (*Sonic Range Finder*, SRF) angeboten, die unterschiedliche Leistungsmerkmale aufweisen und sich preislich zwischen 15 und 100 € bewegen. Sie werden in Deutschland von verschiedenen Elektronik-Shops vertrieben, darunter Manu Systems, nodna und EXP. Über einen I²C-Controller, der den *Fast Mode* (400 *kHz*) beherrscht, verfügen die folgenden vier Devantech-SRF-Sensoren:

- der kostengünstige SRF02: Messbereich 16 *cm* bis 6 *m*, Preis ca. 15 €;
- der SRF08 mit Lichtsensor und der Möglichkeit, mehrere hintereinander angeordnete Objekte zu erkennen: Messbereich 3 *cm* bis 6 *m*, Preis ca. 35 €;
- der weltweit kleinste Dual-Ultraschallsensor SRF10[3]: Messbereich 6 *cm* bis 6 *m*, Preis ca. 35 €; und
- der *Pencil Beam* SRF235 mit einem Öffnungswinkel von nur 15°: Messbereich 10 *cm* bis 1,2 *m*, Preis ca. 100 €.

Alle vier Sensoren arbeiten mit einer Ultraschallfrequenz von 40 *kHz* und einer Ver-

[2] Nach einer Analyse von Ad2 und ft-ninja wird das Messergebnis mit 115.200 Baud seriell an den TX übertragen. Der Sensor stammt wahrscheinlich von Maxim.

[3] Für den SRF10 findet sich im Download-Bereich der ft:c ein von ftDirk entwickelter Robo Pro-I²C-Treiber.

sorgungsspannung von 5 *V*. Sie lassen sich ohne einen *Level Shifter* direkt mit dem EXT-2-Anschluss des TX verbinden.

Für einen Einsatz in einem fischertechnik-Roboter sind vor allem der SRF02 – wegen des günstigen Preises – und der SRF08 aufgrund seiner zusätzlichen Funktionen interessant. Beide werden im Folgenden vorgestellt.

Ultraschallsensor SRF02

Der SRF02 wird mit der voreingestellten 7-Bit-I²C-Adresse 0x70 ausgeliefert. Sie kann auf einen Wert zwischen 0x70 und 0x80 geändert werden. Damit lassen sich bis zu 16 dieser Sensoren ohne Multiplexer auf demselben I²C-Bus einzeln adressieren.

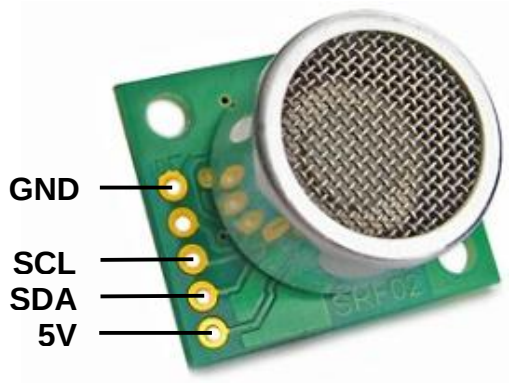


Abb. 1: SRF02 von Devantech, Anschlussbelegung [3]

Die I²C-Anschlussbelegung des Sensors zeigt Abb. 1.

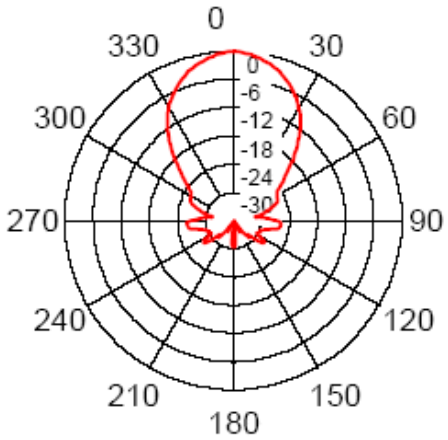


Abb. 2: Richtcharakteristik des SRF02 (Quelle: Devantech Ltd. [3])

Der Sensor hat wie seine ‚großen Brüder' einen Messbereich von bis zu sechs Metern. Eine hohe Messgenauigkeit erreicht der Sensor bei einem Öffnungswinkel von bis zu 60° (Abb. 2). Für Messungen im Nahbereich unterhalb von etwa 16 *cm* ist er allerdings nicht geeignet. Dafür liegt seine Stromaufnahme mit 4 *mA* ausgesprochen niedrig.

Befehls-Register

Die Entfernungsmessung wird durch einen Befehl im *Command Register* (0x00) ausgelöst. Es ist das einzige beschreibbare Register des Sensors. Tabelle 1 enthält eine Übersicht der zulässigen Kommandos, die über das *Command Register* an den Sensor übermittelt werden können.

Wert	Kommando
0x50	*Ranging Mode: Result in* inch
0x51	*Ranging Mode: Result in* cm
0x52	*Ranging Mode: Result in* µs
0x56	*Sync Mode: Result in* inch
0x57	*Sync Mode: Result in* cm
0x58	*Sync Mode: Result in* µs
0x5C	*40* kHz *Impulse*
0x60	*Auto Calibration*
0xA0	*Change I²C-Address (first)*
0xA5	*Change I²C-Address (third)*
0xAA	*Change I²C-Address (second)*

Tab. 1: Kommandos

Die **Befehle 0x50, 0x51 und 0x52** lösen eine Entfernungsmessung aus. Die Befehle legen jeweils die Maßeinheit des Messergebnisses (*inch*, *cm*, *µs*) fest. Jedes Mal, wenn ein Signalton ausgesendet wird, leuchtet eine rote LED kurz auf. Eine einzelne Messung kann bis zu 0,065 *s* dauern; spätestens nach 70 *ms* ist der Sensor wieder bereit. Damit ermöglicht der

Sensor bis zu 15 Messungen pro Sekunde. Diese Mess-Zeitspanne sollte entweder vor dem Auslesen der Messwerte abgewartet werden [4], oder aber das Ergebnisregister wird so lange abgefragt, bis es nicht mehr den Wert 0xFF enthält.

Die **Befehle 0x56, 0x57 und 0x58** lösen eine Messung *ohne* Signalton aus – und erlauben so die Messung eines von einem anderen Sensor abgegebenen Signaltons, auch hier für drei unterschiedliche Maßeinheiten. Der **Befehl 0x5C** erzeugt den für eine solche kollaborative Entfernungsmessung erforderlichen 40-kHz-Signalton.

Der **Befehl 0x60** leitet eine Kalibrierung des Sensors ein. Dabei wird die kürzeste messbare Entfernung bestimmt und im Register *Autotune Minimum* abgespeichert. Der Wert ist stark temperaturabhängig und variiert nach Angaben des Herstellers zwischen 11 und 16 *cm*. Der Kalibrierungsbefehl wird auch automatisch beim Einschalten der Stromzufuhr vom Sensor ausgeführt und benötigt weniger als eine halbe Sekunde.

Mit der **Befehlsfolge 0xA0, 0xAA und 0xA5**, gefolgt von einem Wert zwischen 0x70 und 0x80, wird dem Sensor eine neue I²C-Adresse zugeordnet. Bei der Umstellung der Adresse darf nur ein Sensor an den Bus angeschlossen sein, anderenfalls wird bei allen Sensoren die neue Adresse eingestellt. Beim Einschalten der Stromzufuhr (Anstecken des Sensors an den TX) wird die letzte Stelle der Adresse durch eine entsprechende Anzahl Blink-Signale der roten LED angezeigt; man muss sich die eingestellte Adresse also nicht unbedingt notieren.

Daten-Register

Der Sensor verfügt über sechs (Byte-) Datenregister, die die in Tabelle 2 zusammengestellten Werte enthalten:

Register	**Inhalt (*read*)**
0x00	*Software Version*
0x01	*0x80 (unused)*
0x02-03	*Echo (high/low)*
0x04-05	*Autotune Minimum (high/low)*

Tab. 2: Read-Register

Das von mir getestete Exemplar des SRF02 liefert beim Auslesen von Register 0x00 die Software-Version 6 zurück.

Register 0x01 ist ungenutzt und liefert den konstanten Wert 0x80. Der folgende 16-Bit-Wert *Echo* (Register 0x02 und 0x03) enthält die zuletzt gemessene Entfernungsangabe; die Einheit ist durch das vorausgegangene Messkommando festgelegt.

Während das Messergebnis in der Einheit *cm* – genau wie der fischertechnik-Sensor – eine Genauigkeit von einem Zentimeter liefert, lässt sich aus der Signallaufzeit in *µs* ein wesentlich genaueres Ergebnis berechnen. So wissen wir, dass sich der Schall bei 20°C und 1 bar Luftdruck in Luft mit einer Geschwindigkeit von etwa 343 *m/s* ausbreitet. In einer *µs* legt er also eine Strecke von 0,343 *mm* zurück; das entspricht einem Abstand von 0,1715 *mm*. Sofern die Laufzeitangabe eine Genauigkeit von 1 *µs* aufweist, ließe sich die Entfernung *D* eines Gegenstands aus der Signallaufzeit *t* auf 0,2 *mm* genau bestimmen (1):

$$D = t \cdot 343 / (2 \cdot 10^5)\ cm \qquad (1)$$

Die Register 0x04 und 0x05 enthalten die kürzeste vom Sensor bestimmbare Entfernung (*Autotune Minimum*), die bei der letzten Kalibrierung (nach Einschalten der Stromzufuhr oder nach Auslösen des Kommandos 0x60, s.o.) ermittelt wurde. Die Einheit entspricht der letzten mit einem Messkommando gewählten Messeinheit (*inch*, *cm*, *µs*).

Robo Pro-Treiber

Ein einfacher Robo Pro-Treiber für den SRF02 findet sich im Download-Bereich der ft:c. Der Treiber misst den Abstand – wahlweise in *cm* oder in *µs* –, liest den Mindestabstand in der zuletzt verwendeten Einheit aus und gibt die Versionsnummer der Sensor-Firmware zurück.

Das enthaltene Beispielprogramm erlaubt eine Vergleichsmessung von SRF02 und fischertechnik-Abstandssensor. In meinen Tests zeigte der SRF02 eine leichte Abweichung (1-2 cm), dafür war der Fokus besser – bei größeren Abständen streut der fischertechnik-Sensor deutlich stärker.

Die Messung ohne Signalton ist im Treiber nicht enthalten – aber schnell implementiert, sollte jemand damit experimentieren wollen. Auch die Adressänderung ist nicht dabei: Nach jeder Änderung der Device-Adresse müssen alle I²C-Befehle des Treibers angepasst werden, da man die Device-Adresse nicht als Parameter an die I²C-Kommandos übergeben kann.

Ultraschallsensor SRF08

Der SRF08 wird mit derselben voreingestellten I²C-Adresse 0x70 wie der SRF02 ausgeliefert. Auch beim SRF08 ist eine Adressänderung im Bereich zwischen 0x70 und 0x80 möglich. Die Anschlussbelegung stimmt ebenfalls mit der des SRF02 überein; daher kann dasselbe Anschlusskabel verwendet werden (Abb. 3).

Abb. 3: SRF08 von Devantech, Anschlussbelegung [5]

Im Unterschied zum SRF02 verfügt der Sensor zusätzlich über einen Helligkeitssensor. In einer Fahrzeugsteuerung kann dieser beispielsweise dazu genutzt werden, bei beginnender Dämmerung oder einer Tunnelfahrt automatisch die Fahrzeugbeleuchtung zu aktivieren.

Zudem speichert der Sensor bis zu 17 Entfernungswerte, die aus unterschiedlichen Reflexionen (Echos) des Signaltons gewonnen werden.

Schließlich kann die (maximale) Reichweite des Sensors begrenzt und damit die Dauer einer einzelnen Messung verkürzt werden. Auch die Empfindlichkeit des Sensors lässt sich einstellen, um beispielsweise in geschlossenen Räumen den störenden Einfluss von verzögerten Echos zu begrenzen.

Der Stromverbrauch des SRF08 liegt im Bereitschaftsmodus (Warten auf ein Kommando) bei 3 *mA* und steigt während einer Messung auf 15 *mA*. Eine kurze Lastspitze von 3 *µs* mit 275 *mA* tritt beim Empfang des Messkommandos auf.

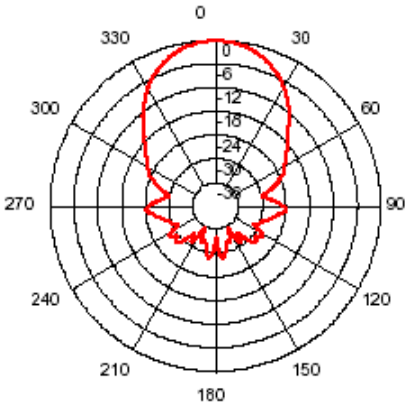


Abb. 4: Richtcharakteristik des SRF08 (Quelle: Devantech Ltd. [5])

Reichweite

Die (hardwareseitig) maximal mögliche Reichweite des SRF08 liegt bei etwa 6 *m*. Die tatsächliche Reichweite wird bestimmt durch einen internen Timer, der die Empfangszeit von Echos eines Signalton begrenzt. Dieser Timer ist beim Einschal-

ten des Sensors auf 65 *ms* voreingestellt; das entspricht einer (theoretischen) Reichweite von 11 *m*. Reduziert man den Wert dieses Timers, so verkürzt sich neben dem Messbereich auch die Antwortzeit des Sensors – damit erhöht sich die Zahl der pro Sekunde möglichen Messungen.

Der Timer kann über das beschreibbare *Range Register* (0x02, Tab. 3) des Sensors eingestellt werden, indem man den Default-Wert (0xFF) durch einen niedrigeren Wert *t* ersetzt. Die Reichweite *R* berechnet sich daraus nach der einfachen Formel (2):

$$R = t \cdot 4{,}3\ cm + 4{,}3\ cm \qquad (2)$$

In jedem Fall sollte man den Wert des *Range Registers* an die tatsächliche maximale Reichweite von 6 *m* anpassen – also auf $t = 0x88$. Die Dauer einer Messung reduziert sich damit auf etwa 36 *ms*, die Zahl der möglichen Messungen pro Sekunde steigt auf 28. Doch Vorsicht: Damit keine „vagabundierenden" Echos nachfolgende Messungen verfälschen, muss für Messungen die Empfindlichkeit des Sensors angepasst werden.

Empfindlichkeit

Die Empfindlichkeit des Sensors wird durch eine Analogverstärkung bestimmt, die während eines Messvorgangs automatisch etwa alle 70 *µs* schrittweise erhöht wird: Sie beginnt bei 94 und endet bei 1.025. In der Default-Einstellung erreicht der Sensor die höchste Empfindlichkeit bei einer Signallaufzeit, die etwa einer Objektentfernung von 39 *cm* entspricht.

Die maximale Empfindlichkeit kann im *Max Gain Register* (0x01, Tab. 3) auf einen Wert von 0 (≡ 94) bis 31 (≡ 1.025) festgelegt werden. Das Verhältnis zwischen dem Register-Wert und dem Verstärkungswert ist nicht linear; für eine konkrete Anwendung sollte man mit unterschiedlichen Werten experimentieren, bis man eine zur eingestellten maximalen Reichweite und gewünschten Messfrequenz passende Verstärkung gefunden hat.

Register	Inhalt (*write*)
0x00	*Command Register*
0x01	*Max Gain Register*
0x02	*Range Register*

Tab. 3: Steuerungs-Register

Die Einstellungen in beiden Registern werden jedoch im RAM gespeichert und nach einer Unterbrechung der Stromversorgung wieder auf die Default-Werte 0x25 bzw. 0xFF zurückgesetzt. Die Einstellung der Werte sollte daher in einer Initialisierungsroutine erfolgen, die immer zu Programmbeginn durchlaufen wird.

Lichtsensor

Der SRF08 ist mit einem einfachen Lichtsensor ausgestattet, der automatisch nach jeder Entfernungsmessung ausgewertet wird. Er liefert einen Wert zwischen 2 (absolute Dunkelheit) und 248 (0xF8, helles Licht) zurück. Dieser Wert lässt sich z. B. für einen Dämmerungsschalter verwenden, der die Beleuchtung eines Fahrzeugs aktiviert.

Befehls-Register

Auch beim SRF08 erfolgt die Steuerung der Abstandsmessvorgänge über das *Command Register* (0x00). Tabelle 4 gibt eine Übersicht der unterstützten Kommandos. Die **Befehle zum Starten eines Messvorgangs** (**0x50 bis 0x52**) entsprechen denen des SRF02; dasselbe gilt für die Befehlssequenz zur Umstellung der I²C-Adresse.

Wert	Kommando
0x50	*Ranging Mode: Result in* inch
0x51	*Ranging Mode: Result in* cm
0x52	*Ranging Mode: Result in* µs
0x53	*ANN Mode: Result in* inch

Wert	Kommando
0x54	*ANN Mode: Result in* cm
0x55	*ANN Mode: Result in* µs
0xA0	*Change I²C-Address (first)*
0xA5	*Change I²C-Address (third)*
0xAA	*Change I²C-Address (second)*

Tab. 4: Kommandos

Neu beim SRF08 sind die **Befehle 0x53 bis 0x55**: Sie aktivieren die *Artificial Neural Networks* Messung – getrennt für die Messeinheiten *inch*, *cm* und *µs*. Dabei werden das Ergebnis der ersten Messung im 16-Bit-Datenregister 0x02/0x03 abgelegt und alle empfangenen Echos „Kategorien" von 2048 *µs* (entsprechend 352 *mm*) den Registern 0x04 bis 0x35 zugeordnet. Wurde in einer Entfernungskategorie ein Echo empfangen, wird das Register auf einen Wert ungleich Null gesetzt: also Register 0x04 bei einem Objekt im Abstand von 0-352 *mm*, Register 0x05 bei einem Objekt im Abstand von 353-705 *mm* usf.

Daten-Register

Der SRF08 verfügt über insgesamt 36 Byte-Register (Tab. 5):

Register	Inhalt (*read*)
0x00	*Software Version*
0x01	*Light Sensor*
0x02-03	*1st Echo (high/low)*
0x04-05	*2nd Echo (high/low)*
…	…
0x22-23	*17th Echo (high/low)*

Tab. 5: Daten-Register

Wie beim SRF02 enthält Register 0x00 die Firmware-Version; mein Sensor meldet 10 zurück. Im Register 0x01 wird der Wert des Lichtsensors abgelegt. Die folgenden Register enthalten bis zu 17 vom Sensor empfangene Abstandswerte. Abhängig vom erteilten Mess-Kommando geben sie den Abstand in *inch*, in *cm* oder in *µs* an. Enthält ein Registerpaar den Wert 0, so folgen keine weiteren Werte.

Eine Messung benötigt bis zu 65 *ms*, bei Einstellung geringerer Reichweite und Empfindlichkeit ggf. weniger. Diese Zeit muss vor Auslesen der Register abgewartet werden – entweder durch eine Warteschleife oder durch wiederholtes Prüfen, ob im Register 0x02 der Wert 0xFF steht – so lange ist der Messvorgang nicht abgeschlossen.

Robo Pro-Treiber

Der Robo Pro-Treiber für den SRF08 findet sich ebenfalls im Download-Bereich der ft:c. Er liest fünf der möglichen 17 Abstandswerte des Sensors nach einer Messung aus und speichert sie in einer (globalen) Liste, liefert die Versionsnummer der Firmware und den Wert des Helligkeitssensors.

Ein Initialisierungskommando stellt die Empfindlichkeit des Sensors ein. Der für sehr spezielle Anwendungen gedachte ANN-Modus des Sensors wird vom Treiber nicht unterstützt.

Quellen

[1] Dirk Fox: *Radar und Sonar*. ft:pedia 2/2011, S. 4-8.

[2] fischertechnik: *Begleitheft Robo TX Explorer*. Fischer-Werke, Waldachtal, 2012.

[3] Devantech: *SRF02 Ultrasonic range finder*. Technical Specification.

[4] Devantech: *SRF02 Ultrasonic range finder*. Technical Specification I2C Mode.

[5] Devantech: *SRF08 Ultrasonic range finder*. Technical Specification.

Tower Bridge von Johann Fox auf der fischertechnik-Convention 2013