SRF02 Datenblatt (Deutsch/Englisch)

SRF02 – LowCost Ultraschallsensor der neusten Generation

Betriebsspannung	5V (stabilisiert)
Stromaufnahme	nur 4mA (typisch)
Ultraschallfrequenz	40khz
Maximale Reichweite	15cm bis 6 Meter
Schnittstelle	RS232 (TTL) und I2C-Bus
Ausgabeeinheit	Wahlweise mm, inch oder uS
Anmerkung	Einfachste Verwendung, keine Kalibration/Justierung notwendig, einfach Spannung anlegen und schon kann gemessen werden. Die neue Firmware machts möglich!
Gewicht	Nur 4,6 g
Größe	24mm x 20mm x 17mm

Der SRF02 ist eine preiswerte Alternative zu den weit verbreiteten Ultraschallsensoren SRF04, SRF05 und SRF10. Die kompakte Größe, große Messbereich und die hohe Genauigkeit dürfte n dieser Klasse wohl einmalig sein.

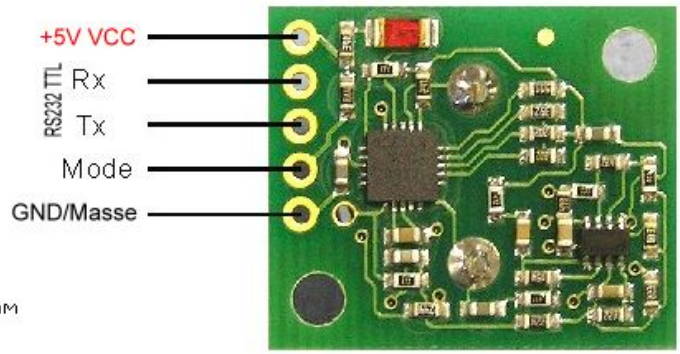
Diese Dokumentation und das Beispiel wird zeigen das die Handhabung sehr einfach ist. Wenige Zeilen, z.B. in Basic, reichen aus um die Entfernung zu ermitteln. Natürlich kann der Sensor auch leicht in Assembler oder C angesprochen werden.

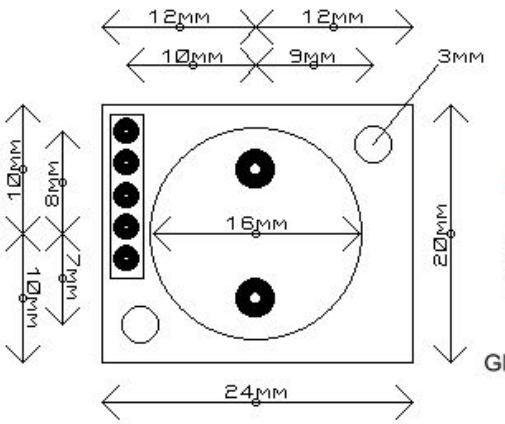
Deutsche Anleitung

SRF05 - Low Cost Utraschall Entfernungsmesser

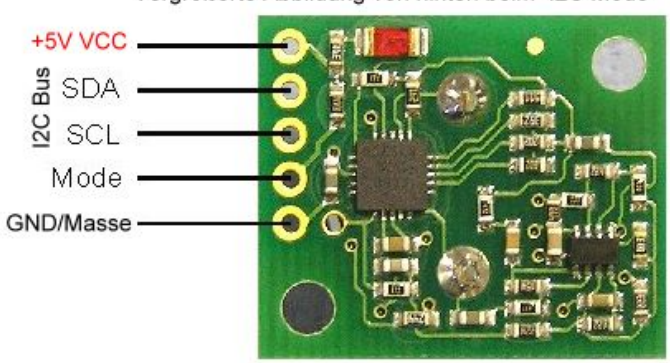
SRF02 neuer sehr kompakter Ultraschallsensor mit I2C und RS232 Interface

Vergrößerte Abbildung von hinten beim RS232 Mode





Vergrößerte Abbildung von hinten beim I2C Mode



Foto/Bezugsquelle: robotikhardware.de

Überblick

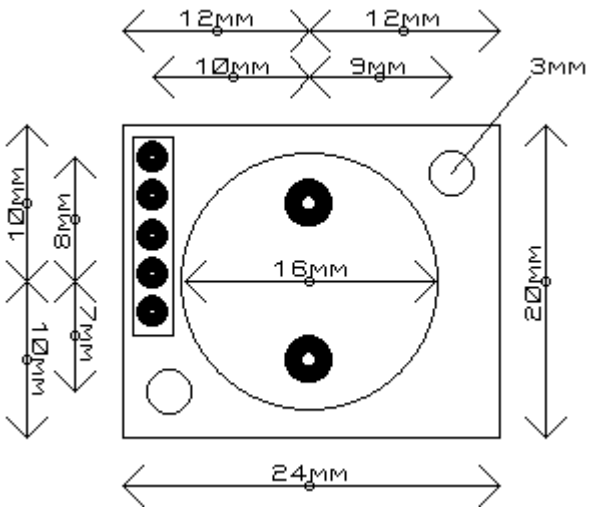
Der neue SRF02 ist ein sehr kompakter Ultraschallsensor der mit I2C-Bus und serieller RS232 Schnittstelle ausgestattet ist. Die serielle Schnittstelle arbeitet im üblichen TTL-Pegel (5V), der Sensor kann somit direkt an Microcontroller (z.B. RN-Boards) angeschlossen werden. Er darf jedoch nicht direkt am PC oder an den dreipoligen RN-RS232 Steckern angesteckt werden, da dort der Rs232 Pegel +12/-12V genutzt wird. Also im RS232 Mode immer direkt an die entsprechenden Ports (oder 4 polige RN-RS232TTLBuchse) anschließen.
Die Baudrate beträgt 9600 Baud, 1 Start, 2 Stop und kein Parity-Bit.
Bis zu 16 Ultraschallsensoren vom Typ SRF02 können an einem einzigen I2C-Bus ode ran einer einzigen RS232 Schnittstelle angeschlossen werden.

Die minimale Messtrecke vom SRF02 ist ca. 15 cm, die maximale Entfernung die gemessen werden kann beträgt ca. 6 Meter. Kleinere Entfernungen als 15cm sind nicht messbar, dafür wären dann Sensoren wie SRF05 oder SRF10 besser geeignet.
Die Entfernung kann wahlweise in cm, inch oder in einer Zeitangabe uS (Zeit bis Echo) ausgegeben werden. Festgelegt wird die Maßeinheit durch einen Befehl.

Betriebsmode

Die Festlegung ob man den Sensor per I2C oder RS232 betreiben möchte erfolgt durch den Mode Pin (siehe Foto, vierte Pin von oben). Wird dieser Pin überhauptnicht beschaltet, dann ist der Sensor automatisch im I2C-Mode. Verbindet man diesen Pin dagegen mit GND (also Minuspol/Masse), so wird der RS232 Mode aktiviert. Man könnte ihn natürlich auch mit einem Controllerport verbinden und so die Schnittstelle per Software umschalten, jedoch ist das in der Praxis wohl kaum interessant.

Um Kabel zu sparen, könnte man also für den RS232 Betrieb einfach eine Brücke zwischen MODE (Pin 4) und GND (Pin5) einlöten. Dadurch reichen dann vier Drähte für die Verbindung zu einem Controllerboard.



I2C Mode Ansteuerung

Wie schon geschildert müssen Sie für diesen Betriebsmode den Mode-Pin **nicht beschalten**, einfach ignorieren. Nur dadurch wird der I2C-Mode aktiviert.

Der I2C-Bus ist sehr beliebt, nicht nur bei allen Roboternetz-Boards (RN-Boards) sondern auch vielen anderen Controller- oder Robotikschaltungen. Der große Vorteil dieses Busses besteht darin das er eigentlich nur aus zwei Leitungen besteht und dennoch bis zu 127 verschiedenste Schaltungen , Sensoren oder Chips daran angeschlossen werden können. Sollten Sie den I2C-Bus noch nicht näher kennen, empfehlen wir die Webseite:

http://www.roboternetz.de/wissen/index.php/I2C

http://www.roboternetz.de/wissen/index.php/RN-Definitionen

http://www.roboternetz.de/wissen/index.php/Kategorie:Projekte

Die Ansteuerung und Handhabung des I2C-Busses ist sehr einfach wie wir auch später in einem Beispielprogramm darstellen.

Wichtig ist zu wissen das jedes Gerät/Sensor/Board (man spricht von **SLAVE**) usw. das am I2C-Bus angeschlossen wird, eine sogenannte Adresse (man spricht von SLAVE-ID) besitzt. Dies ist eine Art Hausnummer über die der **Slave** gezeilt angesprochen werden kann. Jede Slave darf nur einmal vorkommen, ansonsten würd ees erhebliche Probleme kommen. Ist ja auch klar, wo soll der Postbode den Brief einwerfen wenn es zwei Häuser mit gleicher Hausnummer gibt?

Die Standard-Hausnummer vom SRF02 ist Hex 0xE0, also Dezimal 224. Man kann die Slave ID durch einen Befehl jedoch ändern. Folgende Slave-ID´s wären möglich:

E0, E2, E4, E6, E8, EA, EC, EE, F0, F2, F4, F6, F8, FA, FC or FE

Also 16 verschiedene Slave-ID´s sind möglich, daher lassen sich auch 16 Ultraschallsensoren anschließen.

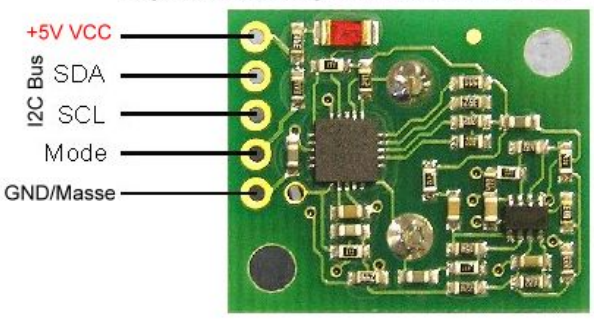
Anschlüsse / Verbindung

Die Pinbelegung des SRF02 ist kompatibel zu den anderen Sensoren wie SRF08 oder SRF10, er kann somit das gleiche Anschlußkabel verwendet werden. Der Mode-Pin war schon bei den anderen Sensoren vorhanden, wird jedoch erst beim SRF02 genutzt.

Zu wissen ist noch das SDA und SCL keinen Pullup-Widerstand auf dem Sensor besitzen, das bedeutet irgendwo auf dem Bus müssen noch zwei Widerstände (ca. 1,8k bis 10k) SDA und SCL jeweils mit 5V verbinden. In der Regel ist das bei den meisten Controllerboards (wie z.B. bei den RN-Boards, RN-Control usw.) bereits vorhanden, so das man sich darum nicht kümmern muss.

Der SRF02 wird immer als normaler Slave also niemals als Master auf dem I2C-Bus betrieben.

Vergrößerte Abbildung von hinten beim I2C Mode



Foto/Bezugsquelle: robotikhardware.de

Register

Der Sensor SRF02 hat 6 Register:

Registernummer	Rückgabe wenn Register gelesen	Funktion wenn Register beschrieben wird
0	Software Firmware Revision	Befehlsregister
1	Ungenutzt (liefert immer den Wert Hex 80)	nicht möglich
2	Entfernungswert (High Byte	nicht möglich
3	Entfernungswert (Low Byte)	nicht möglich
4	Autotune Minimum (High Byte)	nicht möglich
5	Autotune Minimum (Low Byte)	nicht möglich

Register 1:

Das erste Register, also 0 ist das Befehlsregister an das man unterschiedliche Befehle senden kann, zum Beispiel auch den Start-Befehl für die Entfernungsmessung.
Wird das Register dagegen gelesen, so erhält man als Rückgabewert immer die Versionsnummer der derzeitigen Firmware (internes Betriebsprogramm des SRF02)

Eine gestartete Entfernungsmessung kann bis zu 65 ms (0,065 Sekunden) dauern, in dieser Zeit kann der Sensor nicht die I2C-Schnittstelle abfragen, schneller als 15 Messungen pro Sekunde sind also bei diesem Sensor nicht drin. Aber das ist ja auch schon recht flott und dürfte für vieles mehr als ausreichen.

Register 2 und 3

Diese beiden Register geben zusammen die Entfernung in cm, Inch oder uS zurück, je nachdem welche Maßeinheit über das Befehlsregister festgelegt wurde. Aus den beiden Byte-Registern bildet man den Entfernungswert durch die übliche Umrechnung:

Entfernung = (Register2 + 256) + Register3

Register 4 und 5

Diese Register enthalten die kleinste meßbare Entfernung. In der Regel ist das ca. 15cm, es kann jedoch je nach Umgebung und Umgebungstemperatur minimal davon abweichen.

Befehle für SRF02

Um den Messvorgang zu Starten besitzt SRF02 drei Befehlscodes, je nachdem in welcher Einheit gemessen werden soll. In Deutschland wird man wohl in der Regel in Zentimetern messen, daher braucht man sich hier nur den Code Hex 51 (Dezimal 81) wirklich merken.

Die Befehle Dezimal 86 bis 88 starten ebenfalls einen Messvorgang in einer der drei Einheiten. Jedoch wird bei diesen Befehlen nicht automatisch ein Ultraschallton erzeugt sondern davon ausgegangen das ein anderer Sensor das Signal erzeugt. Man kann den Ultraschallton mittels Befehlscode 92 erzeugen. Diese Methode kann zum synchronisieren von mehreren Sensoren genutzt werden.

Befehlscode		**Funktion**
Decimal	Hex	
80	0x50	Startet Messvorgang (gemessen wird in der Einheit **inches**)
81	0x51	Startet Messvorgang (gemessen wird in der Einheit **Zentimeter**)
82	0x52	Startet Messvorgang (gemessen wird in der Einheit **Microsekunden**)
86	0x56	Synchron Messvorgang (gemessen wird in der Einheit **inches**)
87	0x57	Synchron Messvorgang (gemessen wird in der Einheit **Zentimeter**)
88	0x58	Synchron Messvorgang (gemessen wird in der Einheit **Microsekunden**)
92	0x5C	Erzeugt einen 8 zyklischen 40khz Impuls/Ton
96	0x60	Startet automatische Kalibrierung – Befehl kann ignoriert werden da dies bereits beim einschalten (Power on) automatisch erfolgt
160	0xA0	Erste Sequence zum ändern der I2C-Bus Slave ID
165	0xA5	Dritte Sequence zum ändern der I2C-Bus Slave ID
170	0xAA	Zweite Sequence zum ändern der I2C-Bus Slave ID

Messvorgang
Um einen Messvorgang durchzuführen muss zuerst das Register 0 durch den Startbefehl (zum Beispiel Code 81) gestartet werden. Danach fügt man einfach einen Wartebefehl von 65 Millisekunden ein und fragt anschließend Register 2 und 3 nach der Entfernung ab. Das war´s schon, einfacher gehts kaum.

Hier eine Bascom-Basic Beispielfunktion für das Board **RN-Control**:

```
Function Srf02_entfernung(byval Slaveid As Byte) As Integer
Local Lob As Byte
Local Hib As Byte
Local Firmware As Byte
Local Temp As Byte
Local Slaveid_read As Byte

   slaveid_read = Slaveid + 1

   'Messvorgang in starten
   I2cstart
   I2cwbyte Slaveid
   I2cwbyte 0
   I2cwbyte 81                                          'in Zentimetern messen
   I2cstop

   Waitms 65                                            'minimale Wartezeit für Messung
bei SRF02

   I2cstart
   I2cwbyte Slaveid
   I2cwbyte 2                                           'Leseregister festlegen
```

```
   I2cstop


   I2cstart
   I2cwbyte Slaveid_read
   I2crbyte Hib , Ack
   I2crbyte Lob , Nack
   I2cstop


   Srf02_entfernung = Makeint(lob , Hib)
End Function
```

Prüfen ob eine Messung auch wirklich fertig ist

Wie schon erläutert daert ein Messvorgang bis zu 65 Millisekunden. Wenn man also nach dem Start des Messvorganges 65 Millisekunden wartet bevor man das Ergebnis ausliest, so gibt es keine Probleme.

Bei kleinen Entfernungen kann der Messvorgang jedoch deutlich schnell gehen. Möchte man keine Zeit verschenken, so kann man mit einem Trick abfragen ob der Sensor fertig mit der Messung ist. Man fragt dazu einfach die Firmwareversion nach dem Start des Messvorganges ab. Da der Sensor während der Messung nicht auf I2C-Befehle reagiert, wird immer 255 statt der Firmware zurückgeliefert. Man muss dieses also solange wiederholen bis ein anderer Wert als 255 zurück kommt.

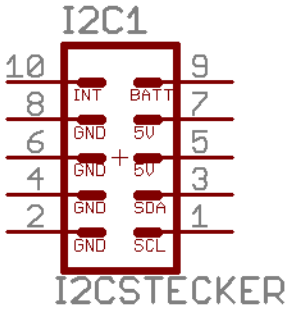
Der komplette **RN-Control** Beispielcode würde dann so aussehen:

```
'####################################################
'SRF02_rncontrol_i2c_beispiel2.bas
'für  RoboterNetz Board RN-CONTROL (ab Version 1.1)
'und das SRF02 Ultraschallmodul für Entfernungsmessung
'Datenblatt zu SRF02:
'http://www.roboternetz.de/phpBB2/dload.php?action=file&file_id=357
'Anschlussbeschreibung:
'http://www.roboternetz.de/wissen/index.php/Sensorarten

'Aufgabe:
' Gibt die Firmware-Version vom SRF02 aus und
' anschließend in einer Endlosschleife die
' Entfernung von Objekten in Zentimetern
' Die Entfernung wird 1 mal pro Sekunde über RS232
' ausgegeben
' Hier wird noch genau gescheckt ob ein Sensor die
' Messung schon beendet hat. Dadurch sind noch
' schnellere Messungen als nur 15 pro Sekunde möglich

'Weitere Beispiele und Beschreibung der Hardware
'unter http://www.Roboternetz.de oder robotikhardware.de
'####################################################

Declare Function Srf02_firmware(byval Slaveid As Byte) As Byte
Declare Function Srf02_entfernung(byval Slaveid As Byte) As Integer

$regfile = "m32def.dat"
$framesize = 42
$swstack = 42
$hwstack = 42

$crystal = 16000000                                    'Quarzfrequenz
$baud = 9600
Config Scl = Portc.0                                   'Ports fuer IIC-Bus
Config Sda = Portc.1
Const Srf02_slaveid = &HE0                             'Standard I2C Adresse von SRF02

Dim Entfernung As Integer
Dim V As Byte
```

```
   Wait 3                                                  'Warte 3 Sekunden
   I2cinit
   Print "SRF02 Testprogramm von robotikhardware.de"
   Print "SRF02 Ultraschall-Firmware Version:" ; Srf02_firmware(srf02_slaveid)

   V = 1
   Do
     Entfernung = Srf02_entfernung(srf02_slaveid)
     Print "Entfernung:" ; Entfernung ; "cm"
     Wait 1
   Loop
End

'------------- Hilfsfunktionen für SRF02 ----------

Function Srf02_firmware(byval Slaveid As Byte) As Byte
Local Firmware As Byte
Local Slaveid_read As Byte

   slaveid_read = Slaveid + 1
   I2cstart
   I2cwbyte Slaveid
   I2cwbyte 0                                              'Leseregister festlegen
   I2cstop

   I2cstart
   I2cwbyte Slaveid_read
   I2crbyte Firmware , Nack
   I2cstop

   Srf02_firmware = Firmware
End Function



Function Srf02_entfernung(byval Slaveid As Byte) As Integer
Local Lob As Byte
Local Hib As Byte
Local Firmware As Byte
Local Temp As Byte
Local Slaveid_read As Byte

   slaveid_read = Slaveid + 1

   'Messvorgang in starten
   I2cstart
   I2cwbyte Slaveid
   I2cwbyte 0
   I2cwbyte 81                                             'in Zentimetern messen
   I2cstop

Warteaufmessung:
   Waitms 1
   Firmware = Srf02_firmware(slaveid)
   If Firmware = 255 Then Goto Warteaufmessung

   I2cstart
   I2cwbyte Slaveid
   I2cwbyte 2                                              'Leseregister festlegen
   I2cstop


   I2cstart
   I2cwbyte Slaveid_read
   I2crbyte Hib , Ack
   I2crbyte Lob , Nack
   I2cstop

   Srf02_entfernung = Makeint(lob , Hib)
End Function
```

Sensor an RN-Control anschließen

Ein oder mehrere SRF02 können wie alle anderen SRF-Sensoren sehr einfach an Controllerboards wie RN-Board (z.B. RN-Control, RN-MiniCOntrol, RN-Mega128Funk usw.) angeschlossen werden. Da der I2C Bus bei den RN-Boards auf einem 10 poligen Wannenstecker liegt, muss man einige Leitungen ungenutzt lassen und am anderen Ende nur folgende Drähte am Sensor anlöten: +5V , GND, SDA , SCL

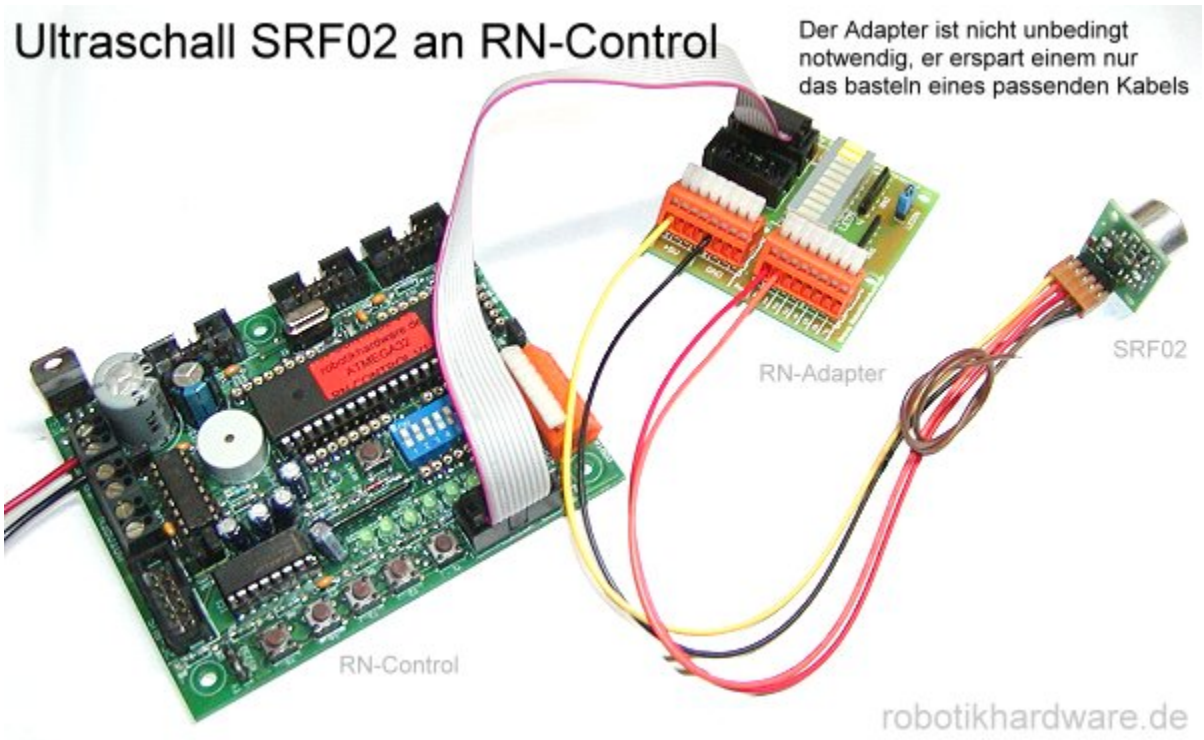


Wenn man über eine Crimpzange mit konfektionierbaren Steckern verfügt, kann man sich auch ein kleinen Kabeladapter basteln.

Alternative über RN-Adapter

Eine andere bequeme Möglichkeit ist ein kleiner RN-Adapter. Das ist eine kleine Platine welche den Wannenstecker auf zwei Steckklemmen verteilt und zudem noch die Leitungen durch LED´s überwacht. Einen solchen Adapter kostet in paar Euro bei Robotikhardware.de aber dafür lassen sich viele Verdrahtungen dann komfortabler und schneller durchführen. Über eine Stiftleiste und ein 5 poliges Standardkabel (aus Elektonikhandel) kann man dann einfach den oder die Sensoren mit dem Board verbinden. Den Adapter steckt man dazu an **Port C** bei RN-Control, da dort auch die I2C-Leitungen liegen (nicht den RN-Adapter an I2C-Buchse anschließen da dort die Pinbelegung anders ist).

Das ganze sieht dann in etwa so aus:



LED
Die LED leuchtet blinkt beim Anlegen der Spannung einige mal. Aus dem Blinkien kann man die eingestellte I2C-Bs Slave ID ersehen, siehe unten. Zudem blinkt diese bei jedem Messvorgang.

Änderung der I2C Slave ID

Die Slave ID muss nur geändert werden wenn mehrere SRF02 an einem I2C Bus betrieben werden sollen oder aber wenn ein anderer Busteilnehmer zufällig die gleiche Slave ID besitzt.

Um die Slave ID zu ändern, darf lediglich nur ein SRF02 am I2C-Bus angeschlossen sein. Die Slave ID wird geändert indem man eine 3 Byte-Sequenz (Hex A0 AA A5) und die neue Slave ID selbst an das Modul sendet.

Die einzelnen Bytes dieser Sequenz müssen an das Register 0 gesendet werden. Man muss also 4 getrennte I2C Schreibbefehle nutzen, wobei der Abstand zwischen jeder Registerbeschreibung 50ms sein sollte.

Um Beispielsweise die Standard ID E0 auf F2 umzustellen, müsste nacheinander das Register 0 mit den Werten A0, AA, A5, F2 beschrieben warden.

Die neue Slave ID sollte man sich gut merken, am besten irgendwo notieren. Falls man die ID mal vergessen hat, so kann man diese durch das blinken der roten LED nach dem Einschalten des Modules entnehmen (siehe Tabelle):

Slave ID		Langes Blinken	Kurzes Blinken
Dezimal	Hex		
224	E0	1	0
226	E2	1	1
228	E4	1	2
230	E6	1	3
232	E8	1	4
234	EA	1	5
236	EC	1	6
238	EE	1	7
240	F0	1	8
242	F2	1	9
244	F4	1	10
246	F6	1	11
248	F8	1	12
250	FA	1	13
252	FC	1	14
254	FE	1	15

Nicht vergessen:
Es darf keine Busteilnehmer mit der gleichen Slave ID an einem Bus geben.

Automatische Kalibrierung

Der Sensor SRF02 muss nicht manuell kalibriert werden. Die Kalibrierung erfolgt automatisch nachdem der Sensor mit Spannung versorgt wurde. Dies läuft völlig automatisch im Hintergrund ab, so das Sie sich nicht darum kümmern müssen. Nähere Details dazu in der englischen Beschreibung weiter hinten.

Serieller Betriebsmode

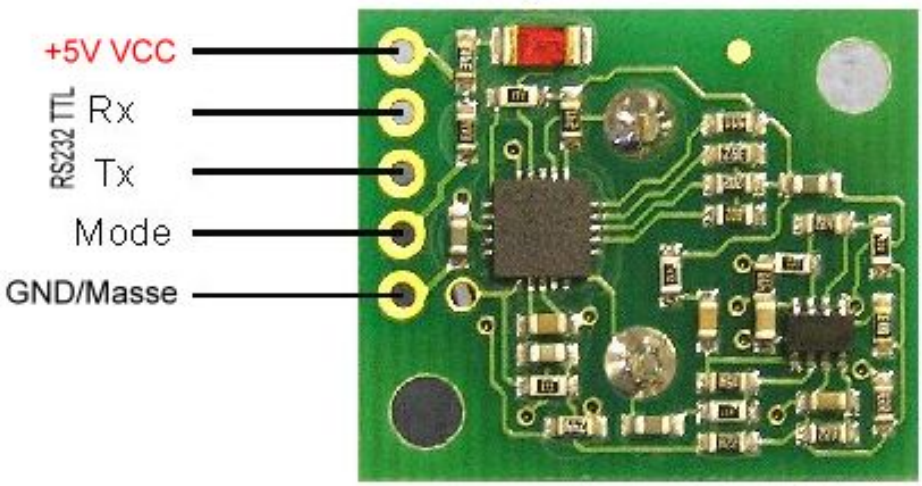
Damit der SRF02 im seriellen RS232-Mode arbeitet muss der Pin MODE (vierte von oben) auf GND gelegt werden. Da er gleich neben GND liegt, könnte man auch eine Brücke einlöten.

Die Übertragungsrate ist fest eingestellt auf 9600 Baud 1 Start, 2 Stop und kein Parity Bit. Nicht vergessen das es sich um ein TTL-Signal handelt, also nicht an ein V24 Signal mit +/- 12V anschließen (PC oder 3 polige RN-Stecker). RN-Boards haben gewöhnlich einen passenden vierpoligen RS232TTL Anschluss. Alternativ kann man RX und TX auch direkt mit dem Microcontroller verbinden. In Verbindung mit Bascom Basic wäre sogar jeder beliebige Port für den Sensor geeignet, da eine RS232 Schnittstelle auch per Software emuliert werden kann.
Für den Anschluss von SRF02 direkt an den PC wäre allerdings noch ein Pegelwandler (Max232) notwendig. Alternativ gingen auch Funkmodule wie RN-Funk.

Befehlssyntax
Die Steuerung des SRF02 ist auch per RS232 sehr ähnlich wie über I2C. Es werden pro Befehl nur 2 Byte übertragen. Das erste Byte beinhaltet die Sensor ID (also eine Art SLAVE ID) und das zweite Byte den Befehlscode.
Die Slave ID´s haben im RS232 Mode die Nummern 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, oder 15. Die bei der Auslieferung eingestelle ID ist 0.
Also auch hier können 16 Sensoren an einer RS232 Verbindung angeschlossen werden.

Anschluss
Wie bereits geschildert, der Mode Pin muss in diesem Mode mit GND verbunden werden.
Der RX-Pin nimmt Daten entgegen, muss also mit TX von einem Controllerboard verbunden werden. TX sendet die Rückgabewerte und muss demnach mit RX eines Controllerboardes verbunden werden.
Wenn sie mehrere Sensoren (bis zu 16) an einer Schnittstelle anschließen möchten, dann verbinden Sie alle RX Pin´s von SRF02 mit TX vom Controllerboard. Zudem alle TX-Pin´s von SRF02 mit dem RX-Pin ihres Controllerboardes. Dies ist beim SRF02 ausdrücklich möglich! Es funktioniert weil TX automatisch hochohmig geschaltet wird wenn SRF02 nicht sendet. Und da aufgrund der unterschiedlichen ID immer nur ein SRF02 gerade sendet, gibt es keinerlei Konflikte.

Vergrößerte Abbildung von hinten beim RS232 Mode



Befehle

Alle Befehle bestehen immer aus 2 Byte. Das erste Byte stelle die Slave ID da (0 bis 15) und das zweite den Befehl. Die vorgegebene Standard ID ist 0.
Die Befehle 80 bis 83 starten den Messvorgang, jeweils in einer anderen Maßeinheit. Der wohl wichtigste Befehl dürfte 81 (Hex 51) sein, er ermittelt die Entfernung in Zentimetern.
Nachdem man den Messvorgang gestartet hat muss man 70 Millisekunden warten und kann dann den Befehlscode 94 senden. Durch diesen Befehlscode wird der SRF02 dazu veranlasst die Entfernung in High und Low Byte zurückzusenden.
vom Sensor zurückgesendet.
Die weiteren drei Befehle 83 bis 85 machen genau das gleiche, jedoch wird hier automatisch die Entfernung zurückgesendet sobald gemessen wurde. Man kann sich hier also den Abrufbefehl 94 sparen.
Die Befehle 86 bis 91 arbeiten wie die zuvor beschriebenen, jedoch wird hier kein Ultraschallsignal ausgesendet. Der SRF02 erwartet dabei das ein anderer Sensor den Ultraschallton aussendet. Dieser Mode kann zur besseren synchronisation von mehreren Sensoren genutzt werden.
Der Befehlscode 92 erzeugt einen 8 zyklischen 40khz Impuls/Ton.
Befehlscode 93 liefert die aktuelle Versionsnummer der SRF02 Firmware
Befehlscode 94 veranlasst SRF02 zum senden der gemessenen Entfernung als High und Low Byte
Die Befehle 95 und 96 werden für Autokalibrierung benutzt.

Befehlscode		
Decimal	**Hex**	**Funktion**
80	0x50	Startet Messvorgang (gemessen wird in der Einheit **inches**)
81	0x51	Startet Messvorgang (gemessen wird in der Einheit **Zentimetern**)
82	0x52	Startet Messvorgang (gemessen wird in der Einheit **Microsekunden**)
83	0x53	Startet Messvorgang (gemessen wird in der Einheit **inches**). Das Ergebnis wird automatisch zurückgesendet sobald Messvorgang abgeschlossen
84	0x54	Startet Messvorgang (gemessen wird in der Einheit **Zentimeter**). Das Ergebnis wird automatisch zurückgesendet sobald Messvorgang abgeschlossen.
85	0x55	Startet Messvorgang (gemessen wird in der Einheit **Microsekunden**). Das Ergebnis wird automatisch zurückgesendet sobald Messvorgang abgeschlossen
86	0x56	Synchron Messvorgang (gemessen wird in der Einheit **inches**)
87	0x57	Synchron Messvorgang (gemessen wird in der Einheit **Zentimetern**)
88	0x58	Synchron Messvorgang (gemessen wird in der Einheit **Microsekunden**)
89	0x59	Synchron Messvorgang (gemessen wird in der Einheit **inches**). Das Ergebnis wird automatisch zurückgesendet sobald Messvorgang abgeschlossen
90	0x5A	Synchron Messvorgang (gemessen wird in der Einheit **Zentimetern**). Das Ergebnis wird automatisch zurückgesendet sobald Messvorgang abgeschlossen
91	0x5B	Synchron Messvorgang (gemessen wird in der Einheit **Microsekunden**). Das Ergebnis wird automatisch zurückgesendet sobald Messvorgang abgeschlossen
92	0x5C	Erzeugt einen 8 zyklischen 40khz Impuls/Ton
93	0x5D	Gibt die Firmwareversion als 1 byte Wert zurück
94	0x5E	Gibt die Entfernung als High und Low Byte zurück (high byte first)
95	0x5F	Gibt die minimale mögliche messbare Entfernung zurück (High und Low Byte). Wird von der Autokalibrierung festgelegt.
96	0x60	Startet automatische Kalibrierung – Befehl kann ignoriert werden da dies bereits beim einschalten (Power on) automatisch erfolgt
160	0xA0	Erste Sequence zum ändern der Slave ID
165	0xA5	Dritte Sequence zum ändern der Slave ID
170	0xAA	Zweite Sequence zum ändern der Slave ID

LED
Die LED leuchtet blinkt beim Anlegen der Spannung einige mal. Aus dem Blinkien kann man die eingestellte I2C-Bs Slave ID ersehen, siehe unten. Zudem blinkt diese bei jedem Messvorgang.

Änderung der Slave ID

Die Slave ID muss nur geändert werden wenn mehrere SRF02 an einer Schnittstelle betrieben werden sollen und wenn ein anderer Sensor die gleiche Slave ID besitzt.

Um die Slave ID zu ändern, darf lediglich nur ein SRF02 angeschlossen sein. Die Slave ID wird geändert indem man eine 3 Byte-Sequenz (Hex A0 AA A5) und die neue Slave ID selbst an das Modul sendet.

Die einzelnen Bytes dieser Sequenz müssen an das Register 0 gesendet werden. Man muss also 4 getrennte I2C Schreibbefehle nutzen, wobei der Abstand zwischen jeder Registerbeschreibung 50ms sein sollte.

Um Beispielsweise die Standard ID E0 auf F2 umzustellen, müsste nacheinander das Register 0 mit den Werten A0, AA, A5, F2 beschrieben warden.

Die neue Slave ID sollte man sich gut merken, am besten irgendwo notieren. Falls man die ID mal vergessen hat, so kann man diese durch das blinken der roten LED nach dem Einschalten des Modules entnehmen (siehe Tabelle):

Slave ID		Langes Blinken	Kurzes Blinken
Dezimal	Hex		
0	00	1	0
1	01	1	1
2	02	1	2
3	03	1	3
4	04	1	4
5	05	1	5
6	06	1	6
7	07	1	7
8	08	1	8
9	09	1	9
10	0A	1	10
11	0B	1	11
12	0C	1	12
13	0D	1	13
14	0E	1	14
15	0F	1	15

Nicht vergessen:
Es darf keine ID doppelt vorkommen!

Automatische Kalibrierung

Der Sensor SRF02 muss nicht manuell kalibriert werden. Die Kalibrierung erfolgt automatisch nachdem der Sensor mit Spannung versorgt wurde. Dies läuft völlig automatisch im Hintergrund ab, so das Sie sich nicht darum kümmern müssen. Nähere Details dazu in der englischen Beschreibung weiter hinten.

Beispiel SRF02 im RS232 Mode an RN-Control

Das nachfolgende Beispiel demonstriert wie einfach sich ein oder mehrere SRF02 Ultraschallsensoren an RN-Control anschließen lassen. Da wir die normale RS232 an RN-Control weiterhin benötigen um Daten und Ergebnisse von RN-Control zum PC zu senden, definieren wir per Softwar einfach eine neue Software RS232 Schnittstelle auf die Ports PA0 und PA1. Dadurch können die SRF02´s sehr einfach an die orange Stecklemme von RN-Control angeschlossen werden. Wir können uns so den Stecker-Adapter sparen. Es wäre natürlich auch jeder andere freie Port verwendbar.

Das ganze sieht dann so aus:

Und hier gleich das passende Beispielprogramm in Bascom-Basic. Es macht genau das gleiche wie das I2C-Programm zuvor, jedoch diesmal alles über RS232 Schnittstelle.

```
'#####################################################
'SRF02_rncontrol_rs232_beispiel.bas
'für
'RoboterNetz Board RN-CONTROL (ab Version 1.1)
'und das SRF02 Ultraschallmodul für Entfernungsmessung
'Datenblatt zu SRF02:
'http://www.roboternetz.de/phpBB2/dload.php?action=file&file_id=357
'Anschlussbeschreibung:
'http://www.roboternetz.de/wissen/index.php/Sensorarten

'Aufgabe:
' Gibt die Firmware-Version vom SRF02 aus und
' anschließend in einer Endlosschleife die
' Entfernung von Objekten in Zentimetern
' Die Entfernung wird 1 mal pro Sekunde über RS232
' ausgegeben
' Dieses Beispiel nutzt den RS232 Mode vom SRF02
' Um die Ergebnisse auch gleichzeitig an PC
' übermitteln zu können, wird in dem Beispiel eine
' zweite RS232 Schnittstelle per Software eingerichtet.
' Dadurch kann der Sensor (oder auch mehrere) bequem
```

```
' an der Steckklemme angeschlossen werden

'Autor: Frank  (Roboternetz)
'Weitere Beispiele und Beschreibung der Hardware
'unter http://www.Roboternetz.de oder robotikhardware.de
'#############################################################



Declare Function Srf02_firmware(byval Slaveid As Byte) As Byte
Declare Function Srf02_entfernung(byval Slaveid As Byte) As Integer


$regfile = "m32def.dat"
$framesize = 42
$swstack = 42
$hwstack = 42

$crystal = 16000000                         'Quarzfrequenz
$baud = 9600                               'Normale Hardware RS232 (hier hängt PC dran)

Open "COMA.0:9600,8,N,2" For Output As #1          'Port PA0 wird als TX definiert
Open "COMA.1:9600,8,N,2" For Input As #2           'Port PA1 wird als RX definiert


Const Srf02_slaveid = 0                     'Standard RS232 Slave ID von SRF02
Dim Entfernung As Integer


  Wait 3                                    'Warte 3 Sekunden
  Print "SRF02 RS232 Testprogramm von robotikhardware.de"
  Print "SRF02 Ultraschall-Firmware Version:" ; Srf02_firmware(srf02_slaveid)
  Print "PA0 wird TX und PA1 wird als RX genutzt"


  V = 1
  Do
    Entfernung = Srf02_entfernung(srf02_slaveid)
    Print "Entfernung:" ; Entfernung ; "cm"
    Wait 1
  Loop
End



'------------- Hilfsfunktionen für SRF02 ----------


Function Srf02_firmware(byval Slaveid As Byte) As Byte

  Print #1 , Chr(slaveid) ; Chr(93);
  Srf02_firmware = Waitkey(#2)
End Function



Function Srf02_entfernung(byval Slaveid As Byte) As Integer
Local Lob As Byte
Local Hib As Byte

  Print #1 , Chr(slaveid) ; Chr(84);          'Messvorgang in cm starten
  Inputbin #2 , Hib , Lob                   'Warte auf Ergebnis
  Srf02_entfernung = Makeint(lob , Hib)
End Function
```

* Alle Beispiele auch auf der robotikhardware.de-CD

Gewährleistung

Beschränkte Garantie und Haftung

Robotikhardware.de übernimmt keine Garantie das das Modul für eine bestimmte Anwendung oder Zweck geeignet ist.
Die Haftung ist maximal auf den Austausch des Modules beschränkt.
Module die durch falsche Anwendung (Verpolung, falsche Spannungen, Überspannungen etc.) können weder ersetzt noch repariert werden.

Es besteht keine Haftung für Schäden die durch die Anwendung des Moduls oder dessen Ausfall entstehen. Wird das Modul zum Bau von Geräten eingesetzt, so muss der Anwender sicherstellen das dafür alle notwendigen Prüfungen und Zulassungen durch ihn erfolgen

Hersteller:

Devantech Lmd

Deutscher Distributor / Bezugsquelle:

www.robotikhardware.de (Brall Software GmbH)

Englische Originalbeschreibung:

SRF05 - Low Cost Ultrasonic Range Finder

SRF02		*Specification*
Voltage	-	*5v only required*
Current	-	*4mA Typ.*
Frequency	-	*40KHz*
Range	-	*15cm - 6m.*
Analogue Gain	-	*Automatic 64 step gain control*
Connection Modes - 1		*Standard I2C Bus.*
2	-	*Serial Bus - connects up to 16 devices to any uP or UART serial port*
Full Automatic Tuning	-	*No calibration, just power up and go*
Timing	-	*Fully timed echo, freeing host controller of task.*
Units	-	*Range reported in uS, mm or inches.*
Light Weight	-	4.6gm
Small Size	-	*24mm x 20mm x 17mm height.*

Best in class performance!

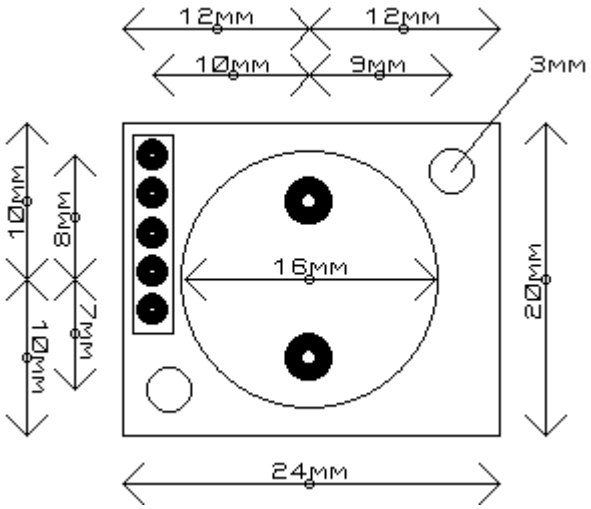
Lowest cost ultrasonic ranger with both I2C and Serial interfaces. New ***AutoTune*** algorithms intelligently back the minimum range right up to the transducer ringdown for the very best performance possible, automatically, in the background, and with no calibration cycles necessary. Additional new commands allow for separate burst and ranging control.

Overview

The SRF02 is a single transducer ultrasonic rangefinder in a small footprint PCB. It features both I2C and a Serial interfaces. The serial interface is a standard TTL level UART format at 9600 baud,1 start, 2 stop and no parity bits, and may be connected directly to the serial ports on any microcontroller. Up to 16 SRF02's may be connected together on a single bus, either I2C or Serial. New commands in the SRF02 include the ability to send an ultrasonic burst on its own without a reception cycle, and the ability to perform a reception cycle without the preceding burst. This has been as requested feature on our sonar's and the SRF02 is the first to see its implementation. Because the SRF02 uses a single transducer for both transmission and reception, the minimum range is higher than our other dual transducer rangers. The minimum measurement range is around 15cm (6 inches). Like all our rangefinders, the SRF02 can measure in uS, cm or inches.

Operating Modes

There are two operating modes for the SRF02. I2C mode and Serial Mode. This is set with the Mode pin, connected to 0v Ground for Serial Mode and left unconnected (or tied to +5v Vcc) for I2C Mode.

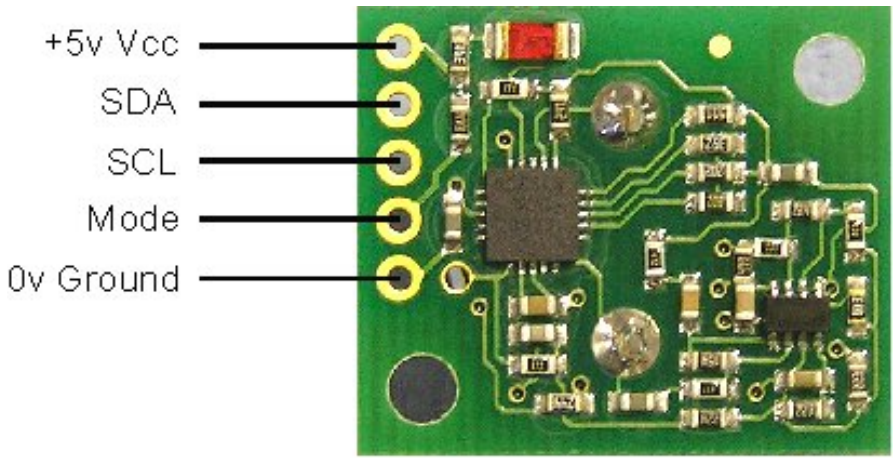


I2C Communication

To use the SRF02 in I2C mode, make sure nothing is connected to the mode pin, it must be left unconnected.
The I2C bus is available on popular controllers such as the OOPic, Stamp BS2p, PicAxe etc. as well as a wide variety of micro-controllers. To the programmer the SRF02 behaves in the same way as the ubiquitous 24xx series EEPROM's, except that the I2C address is different. The default shipped address of the SRF02 is 0xE0. It can be changed by the user to any of 16 addresses E0, E2, E4, E6, E8, EA, EC, EE, F0, F2, F4, F6, F8, FA, FC or FE, therefore up to 16 sonar's can be used.

Connections

The connections to the SRF02 are identical to the SRF08 and SRF10 rangers. The "Mode" pin should be left unconnected, it has an internal pull-up resistor. The SCL and SDA lines should each have a pull-up resistor to +5v somewhere on the I2C bus. You only need one pair of resistors, not a pair for every module. They are normally located with the bus master rather than the slaves. The SRF02 is always a slave - never a bus master. If you need them, I recommend 1.8k resistors. Some modules such as the OOPic already have pull-up resistors and you do not need to add any more.



Registers

The SRF02 appears as a set of 6 registers.

Location	Read	Write
0	Software Revision	Command Register
1	Unused (reads 0x80)	N/A
2	Range High Byte	N/A
3	Range Low Byte	N/A
4	Autotune Minimum - High Byte	N/A
5	Autotune Minimum - Low Byte	N/A

Only location 0 can be written to. Location 0 is the command register and is used to start a ranging session. It cannot be read. Reading from location 0 returns the SRF02 software revision. The ranging lasts up to 65mS, and the SRF02 will not respond to commands on the I2C bus whilst it is ranging.

Locations, 2 and 3, are the 16bit unsigned result from the latest ranging - high byte first. The meaning of this value depends on the command used, and is either the range in inches, or the range in cm or the flight time in uS. A value of 0 indicates that no objects were detected. Do not initiate a ranging faster than every 65mS to give the previous burst time to fade away.

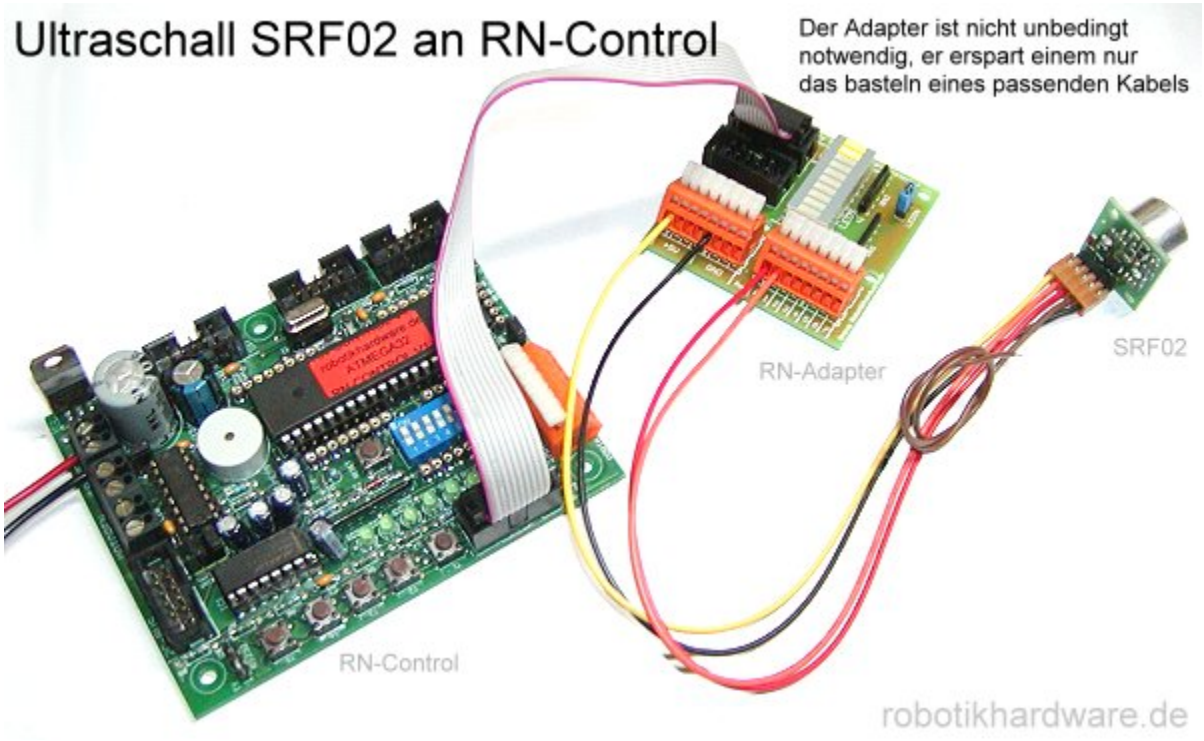
Locations, 4 and 5, are the 16bit unsigned minimum range. This is the approximate closest range the sonar can measure to. See the Autotune section below for full details.

Commands
The are three commands to initiate a ranging (80 to 82), to return the result in inches, centimeters or microseconds. Another set of three commands (86 to 88) do the same, but without transmitting the burst. These are used where the burst has been transmitted by another sonar. It is up to you to synchronize the commands to the two sonar's. There is a command (92) to transmit a burst without doing the ranging and also a set of commands to change the I2C address.

Command		Action
Decimal	Hex	
80	0x50	Real Ranging Mode - Result in inches
81	0x51	Real Ranging Mode - Result in centimeters
82	0x52	Real Ranging Mode - Result in micro-seconds
86	0x56	Fake Ranging Mode - Result in inches
87	0x57	Fake Ranging Mode - Result in centimeters
88	0x58	Fake Ranging Mode - Result in micro-seconds
92	0x5C	Transmit an 8 cycle 40khz burst - no ranging takes place
96	0x60	Force Autotune Restart - same as power-up. You can ignore this command.
160	0xA0	1st in sequence to change I2C address
165	0xA5	3rd in sequence to change I2C address
170	0xAA	2nd in sequence to change I2C address

Ranging
To initiate a ranging, write one of the above commands to the command register and wait the required amount of time for completion and read the result. The echo buffer is cleared at the start of each ranging. The ranging lasts up to 66mS, after this the range can be read from locations 2 and 3.

Checking for Completion of Ranging
You do not have to use a timer on your own controller to wait for ranging to finish. You can take advantage of the fact that the SRF02 will not respond to any I2C activity whilst ranging. Therefore, if you try to read from the SRF02 (we use the software revision number a location 0) then you will get 255 (0xFF) whilst ranging. This is because the I2C data line (SDA) is pulled high if nothing is driving it. As soon as the ranging is complete the SRF02 will again respond to the I2C bus, so just keep reading the register until its not 255 (0xFF) anymore. You can then read the sonar data. Your controller can take advantage of this to perform other tasks while the SRF02 is ranging. The SRF02 will always be ready 70mS after initiating the ranging.



LED
The red LED is used to flash out a code for the I2C address on power-up (see below). It also gives a brief flash during the "ping" whilst ranging.

Changing the I2C Bus Address
To change the I2C address of the SRF02 you must have only one sonar on the bus. Write the 3 sequence commands in the correct order followed by the address. Example; to change the address of a sonar currently at 0xE0 (the default shipped address) to 0xF2, write the following to address 0xE0; (0xA0, 0xAA, 0xA5, 0xF2). These commands must be sent in the correct sequence to change the I2C address, additionally, No other command may be issued in the middle of the sequence. The sequence must be sent to the command register at location 0, which means 4 separate write transactions on the I2C bus. When done, you should label the sonar with its address, however if you do forget, just power it up without sending any commands. The SRF02 will flash its address out on the LED. One long flash followed by a number of shorter flashes indicating its address. The flashing is terminated immediately on sending a command the SRF02.

Address		Long Flash	Short flashes
Decimal	Hex		
224	E0	1	0
226	E2	1	1
228	E4	1	2
230	E6	1	3
232	E8	1	4
234	EA	1	5
236	EC	1	6
238	EE	1	7
240	F0	1	8
242	F2	1	9
244	F4	1	10
246	F6	1	11
248	F8	1	12
250	FA	1	13

252	FC	1	14
254	FE	1	15

Take care not to set more than one sonar to the same address, there will be a bus collision and very unpredictable results.

Note - there is only one module address stored in the SRF02. If you change it, the equivalent Serial Mode address will also change:
0xE0, 0xE2, 0xE4, 0xE6, 0xE8, 0xEA, 0xEC, 0xEE, 0xF0, 0xF2, 0xF4, 0xF6, 0xF8, 0xFA, 0xFC, 0xFE I2C addresses
0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F Equivalent Serial addresses

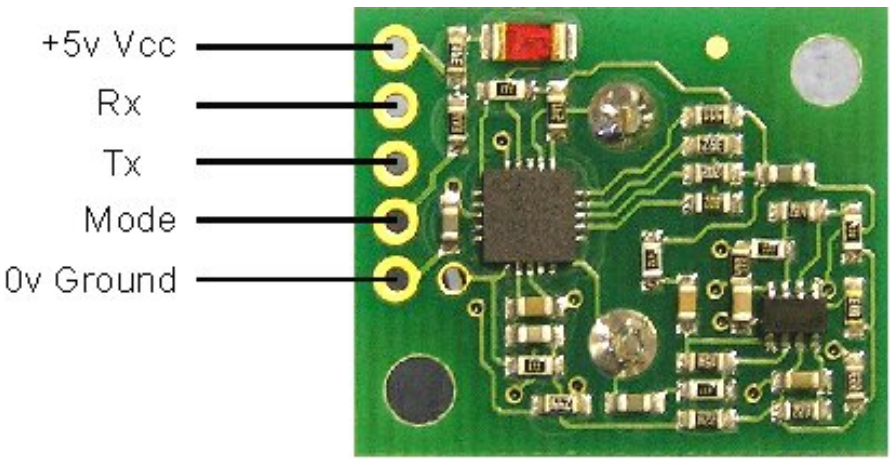
AutoTune
The SRF02 does not require any user calibration. You power up and go right ahead and use the SRF02.
Internally, there are tuning cycles happening automatically in the background. After the ultrasonic burst has been transmitted, the transducer keeps on ringing for a period of time. It is this ringing which limits the closest range the SRF02 can measure. This time period varies with temperature and from transducer to transducer, but is normally the equivalent of 11 to 16cm (4" to 6"), a bit more if the transducer is warm. The SRF02 is able to detect the transducer ring time and move its detection threshold right up to it, giving the SRF02 the very best performance possible. On power up, the detection threshold is set to 28cm (11"). The tuning algorithms quickly back this right up to the transducer ring. This happens within 5-6 ranging cycles - less than half a second at full scan speed. After this the tuning algorithms continue to monitor the transducer, backing the threshold up even further when possible or easing it out a bit when necessary. The tuning algorithms work automatically, in the background and with no impact on scan time.
The minimum range can be checked, if required by reading registers 4 and 5. This value is returned in uS, cm or inches, the same as the range. It is also possible to make the SRF02 re-tune by writing command 96 but you can ignore this command. It is used during our testing.

Serial Communication

To use the SRF02 in Serial mode, make sure the Mode pin is connected to 0v Ground.
Serial data is fixed at 9600 baud 1 start, 2 stop and no parity bits. Serial data is a TTL level signal - It is NOT RS232. Do not connect the SRF02 to an RS232 port - you will destroy the module! If you would like to connect the SRF02 to your PC's RS232 port, you must use a MAX232 or similar device. It can also be used (in I2C mode) with the USBI2C module to make a self powered USB ranger, see the examples page for details. Many small controllers such as the OOPic, Stamp BS2p, PicAxe etc. as well as a wide variety of micro-controllers have serial ports. To communicate with the SRF02, you simply need to send two bytes, the address of the SRF02 (factory default is 0) and the command. The default shipped address can be changed by the user to any of 16 addresses 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, or 15, therefore up to 16 sonar's can be used.

Connections

The connections to the SRF02 are shown below. The "Mode" pin must be connected to 0v ground to place the SRF02 in serial mode. The Rx pin is data into the SRF02 and should be connected to the Tx pin on your controller. The Tx pin is data out of the SRF02 and should be connected to the Rx pin on your controller. If you're using multiple SRF02's, you can connect them all up to the same serial port on your controller. Connect the Tx from your controller to all the Rx pins on the SRF02's and connect the Rx pin on your controller to all the Tx pins on the SRF02's. This works because the Tx pins are high impedance (just a weak pull-up to 5v), except when actually sending data. Just make sure all the SRF02's are programmed to different addresses.



Commands

To send a command to the SRF02, you need to send two bytes. The first is the SRF02's address 0 to 15, (0x00 to 0x0F) and then the actual command itself - see below. The are three commands to initiate a ranging (80 to 82), to produce the result in inches, centimeters or microseconds. These three commands don't Tx the result back to your controller. You should wait 70mS and then use command 94 to get the result of the ranging. Another set of three commands (83 to 85) do the same, but also transmits the result of the ranging back to your controller as soon as it is available. Together, these six commands (80 - 85) are called "Real" because they perform a complete ranging. There is another set of six commands (86 - 91) called "Fake". They are the same as the "Real" commands except that they do not send the 8-cycle burst out. These are used where the burst has been transmitted by another sonar. It is up to you to synchronize the commands to the two sonar's. There is a command (92) to transmit a burst without doing the ranging.
Command 93 is used to get the firmware revision of the SRF02.
Command 94 gets returns two bytes (high byte first) from the most recent ranging. Put them together to make a 16-bit result.
Commands 95 and 96 are used by the Autotune algorithms - See the Autotune section below for details.

Commands		
Decimal	**Hex**	**Action**
80	0x50	Real Ranging Mode - Result in inches
81	0x51	Real Ranging Mode - Result in centimeters
82	0x52	Real Ranging Mode - Result in micro-seconds
83	0x53	Real Ranging Mode - Result in inches, automatically Tx range back to controller as soon as ranging is complete.
84	0x54	Real Ranging Mode - Result in centimeters, automatically Tx range back to controller as soon as ranging is complete.
85	0x55	Real Ranging Mode - Result in micro-seconds, automatically Tx range back to controller as soon as ranging is complete.
86	0x56	Fake Ranging Mode - Result in inches
87	0x57	Fake Ranging Mode - Result in centimeters
88	0x58	Fake Ranging Mode - Result in micro-seconds
89	0x59	Fake Ranging Mode - Result in inches, automatically Tx range back to controller as soon as ranging is complete.
90	0x5A	Fake Ranging Mode - Result in centimeters, automatically Tx range back to controller as soon as ranging is complete.
91	0x5B	Fake Ranging Mode - Result in micro-seconds, automatically Tx range back to controller as soon as ranging is complete.
92	0x5C	Transmit an 8 cycle 40khz burst - no ranging takes place
93	0x5D	Get software version - sends a single byte back to the controller
94	0x5E	Get Range, returns two bytes (high byte first) from the most recent ranging.
95	0x5F	Get Minimum, returns two bytes (high byte first) of the closest range measurable - see Autotune section
96	0x60	Force Autotune Restart - same as power-up. You can ignore this command.
160	0xA0	1st in sequence to change I2C address
165	0xA5	3rd in sequence to change I2C address
170	0xAA	2nd in sequence to change I2C address

LED

The red LED is used to flash out a code for the I2C address on power-up (see below). It also gives a brief flash during the "ping" whilst ranging.

Changing the SRF02 Address

To change the address of the SRF02 you must have only one sonar connected. Write the 3 sequence commands in the correct order followed by the address. Example; to change the address of a sonar currently at 0 (the default shipped address) to 5, write the following to address 0; (0xA0, 0xAA, 0xA5, 0x05). These commands must be sent in the correct sequence to change the I2C address, additionally, No other command may be issued in the middle of the sequence. The sequence must be sent as four separate commands to the current address of the sonar. i.e. 0x00, 0xA0 then 0x00, 0xAA, then 0x00, 0xA5 and finally 0x00, 0x05. When done, you should label the sonar with its new address, however if you do forget, just power it up without sending any commands. The SRF02 will flash its address out on the LED. One long flash followed by a number of shorter flashes indicating its address. The flashing is terminated immediately on sending a command the SRF02.

Address			Long Flash	Short flashes
Decimal	Hex			
0	00	1	0	
1	01	1	1	
2	02	1	2	
3	03	1	3	
4	04	1	4	
5	05	1	5	
6	06	1	6	
7	07	1	7	
8	08	1	8	
9	09	1	9	
10	0A	1	10	
11	0B	1	11	
12	0C	1	12	
13	0D	1	13	
14	0E	1	14	
15	0F	1	15	

Take care not to set more than one sonar to the same address, there will be a bus collision and very unpredictable results.

Note - there is only one module address stored in the SRF02. If you change it, the equivalent I2C address will also change:

0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F Serial addresses

0xE0, 0xE2, 0xE4, 0xE6, 0xE8, 0xEA, 0xEC, 0xEE, 0xF0, 0xF2, 0xF4, 0xF6, 0xF8, 0xFA, 0xFC, 0xFE Equivalent I2C addresses

AutoTune

The SRF02 does not require any user calibration. You power up and go right ahead and use the SRF02.

Internally, there are tuning cycles happening automatically in the background. After the ultrasonic burst has been transmitted, the transducer keeps on ringing for a period of time. It is this ringing which limits the closest range the SRF02 can measure. This time period varies with temperature and from transducer to transducer, but is normally the equivalent of 11 to 16cm (4" to 6"), a bit more if the transducer is warm. The SRF02 is able to detect the transducer ring time and move its detection threshold right up to it, giving the SRF02 the very best performance possible. On power up, the detection threshold is set to 28cm (11"). The tuning algorithms quickly back this right up to the transducer ring. This happens within 5-6 ranging cycles - less than half a second at full scan speed. After this the tuning algorithms continue to monitor the transducer, backing the threshold up even further when possible or easing it out a bit when necessary. The tuning algorithms work automatically, in the background and with no impact on scan time.

The minimum range can be checked, if required by sending command 95. This will return the closest

measurable range in uS, cm or inches, the same as the range. It is also possible to make the SRF02 re-tune by writing command 96 but you can ignore this command. It is used during our testing.

Hersteller:
Devantech Lmd

Deutscher Distributor / Bezugsquelle:
www.robotikhardware.de (Brall Software GmbH)

Deutsche Doku im vorderen Teil dieses Dokumentes
Beispielprogramme im Quellcode auf der robotikhardware.de CD-Rom