SRF10 Ultrasonic rangefinder

A high-quality ultrasonic module for measuring distances. Specifically developed for robotics, but also suitable for numerous other applications. This is the latest generation of ultrasonic modules; the SRF10 is essentially the successor to the SRF04 and SRF08 series, which are already in use in thousands of applications.
Thanks to state-of-the-art technology, the size of the successor to this ultrasonic module has been significantly reduced compared to its predecessors.

Communication with the SRF10 ultrasonic sensor is conveniently achieved via the I2C bus. This bus is used by many microcontroller boards, including almost all robot network boards (RN boards) such as RN-Control, RNBFRA, RN-Mega8, and many others.

The slave ID (I2C default address) of the SRF10 is set to Hex E0 upon delivery. It is also possible to change the slave ID, allowing up to 16 SRF10 ultrasonic modules to be used via one bus (one cable). The user can:

The following 16 slave IDs are possible:

Hex E2, E4, E6, E8, EA, EC, EE, F0, F2, F4, F6, F8, FA, FC und FE.

The broadcast address 0 is also supported. This allows commands to be sent to all connected sensors simultaneously. This allows different distances to be measured simultaneously.

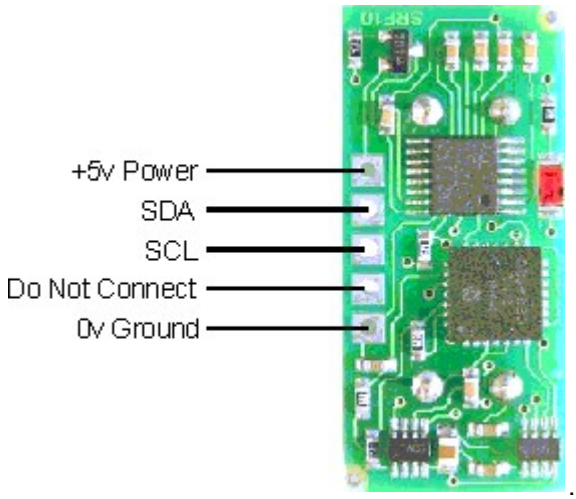
Technical features:

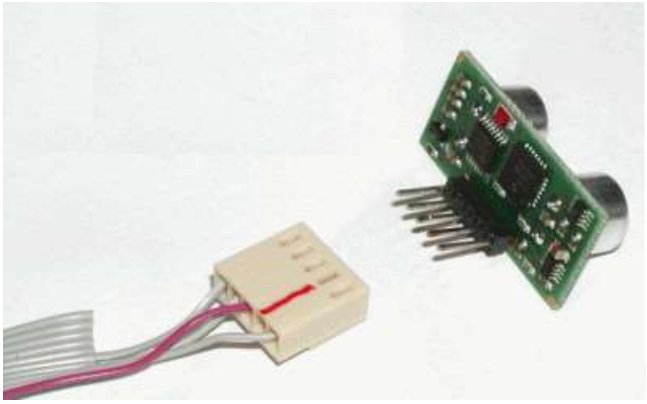
- **Operating voltage:** 5V
- **Current consumption:** approx. 3mA standby, approx. 15mA during measurement
- **Frequency:** 40kHz
- **Maximum range:** 6m
- **Minimum range:** 4cm
- **Data acquisition:** internal, no external controller required for timing. Interface: Standard – I2C (compatible with numerous controller boards, e.g., RN-Control). Output format: µs, mm, or inch.
- **Feature:** Analog gain 40-700 (adjustable, 16 steps).
- **Dimensions:** 32mm x 15mm x 10mm
- **Manufacturer**: Devantech Ltd

-

Connections

The SRF10's connectors are pin-compatible with its predecessor, the SRF08. The "Do Not Connect" pin must not be used; it is used only by the manufacturer. The I2C lines SDA and SCL must be connected to the corresponding I2C or TWI pins of a controller or controller board. All modern controller boards route the I2C bus to the outside via a connector. Many boards, such as RN-Control, also supply the required voltage of 5 volts via the same connector. When using the I2C bus, it is important to ensure that a board on the bus pulls the SCA and SDA lines to high level via two pull-up resistors (approximately 4.7k to 10k). The SRF10 does not have such pull-up resistors; boards like RN-Control already have them.





Register

The SRF10 module is controlled via 4 internal registers:

Register number	Function when register is read	Function when register is written
0	The firmware version will be returned	A command is passed Command register
1	Not used (always returns Hex 80)	Gain factor Default value: 16
2	Determined distance (high byte)	Range Default value: 255
3	Determined distance (low byte)	is not used

Register numbers 0, 1, and 2 can be written to. Commands are transmitted via register 0, thus triggering the measurement. With the default settings, a measurement normally takes about 65 milliseconds. By writing other values to the RANGE register, the measurement time can be shortened. In these cases, the gain register may also need to be adjusted by writing to it. The distance of the last measurement can be read as a whole 16-bit number (unsigned) from registers 2 and 3 (low and high bytes). The distance is therefore calculated as follows:

(Register2 * 256) + Register3 = distance

A value of 0 indicates that no objects were found in the measurement range. The unit in which the distance is read is determined by the actual measurement command.

Commands

There are three different commands that trigger the measurement (decimal 80 to 82). Additional commands are used to set the I2C slave ID.

Command		Action
Decimal	Hex	
80	0x50	The distance is measured in inches
81	0x51	The distance is measured in centimeters
82	0x52	The distance is transmitted by the travel time of the sound (microseconds)
160	0xA0	1 byte to change the slave ID
165	0xA5	3 bytes to change the slave ID
170	0xAA	2 bytes to change the slave ID

-

Checking if a measurement is complete

To determine whether a measurement is complete, register 0 can be read. If the value 255 is returned, the measurement is not yet complete. If a different value is returned (which also indicates the firmware version), the measurement is complete, and the distance can be read via registers 2 and 3.

Changing the Range

The maximum range of the SRF10 is set internally by a timer. The default value is 65 milliseconds, which theoretically allows measurements up to 11 meters. However, this far exceeds the maximum possible measuring range of 6 meters. By decreasing the value in Range Register 2, the time to wait for an echo can be reduced. The range can be set precisely in 43 mm increments. The range is calculated as follows:

Reichweite= (43mm * Register2) + 43mm

A value of 0 in register 2 would therefore result in a range limitation of 43 mm. In practice, however, measurements are only possible from 4 cm. Reducing the range also means that measurements can be taken faster than 65 ms. Therefore, the range is reduced, especially when measurements need to be taken more quickly (increasing the measurement frequency). If the measurement frequency is increased, the gain usually also needs to be reduced.

Analog Gain (Module Sensitivity)

The maximum sensitivity of the measurement process is normally set to the highest possible value of 255. This ensures that even the smallest echoes are detected by the module. In small rooms or rooms with many large objects, additional echoes may occur that distort the measurement result. In such cases, the gain factor in register 1 must be reduced. This is also necessary when measurements are taken very quickly one after the other (high measurement frequency), as otherwise echoes from the previous measurement may interfere with the new measurement. The sensitivity is not permanently stored in SRF10 but only in RAM. Therefore, it must be configured as desired each time the module is switched on.

Gain register		Maximum gain
Decimal	Hex	
0	0x00	Sets the maximum gain to 40
1	0x01	Sets the maximum gain to 40
2	0x02	Sets the maximum gain to 50
3	0x03	Sets the maximum gain to 60
4	0x04	Sets the maximum gain to 70
5	0x05	Sets the maximum gain to 80
6	0x06	Sets the maximum gain to 100
7	0x07	Sets the maximum gain to 120
8	0x08	Sets the maximum gain to 140
9	0x09	Sets the maximum gain to 200
10	0x0A	Sets the maximum gain to 250
11	0x0B	Sets the maximum gain to 300
12	0x0C	Sets the maximum gain to 350
13	0x0D	Sets the maximum gain to 400
14	0x0E	Sets the maximum gain to 500
15	0x0F	Sets the maximum gain to 600
16	0x10	Sets the maximum gain to 700

The ideal values are best determined through experimentation.

LED

The red LED is used when a new I2C slave ID is set. It also flashes briefly during each ultrasonic measurement.

Changing the I2C Slave ID

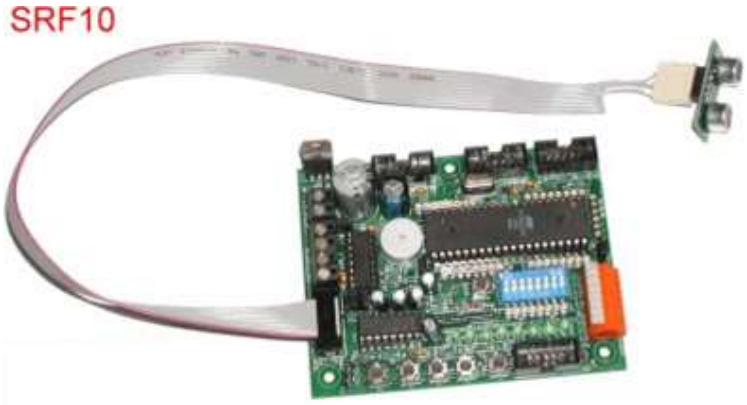
The slave ID only needs to be changed if multiple SRF10s are to be operated on an I2C bus or if another bus device happens to have the same slave ID. To change the slave ID, only one SRF10 may be connected to the I2C bus. The slave ID is changed by sending a 3-byte sequence (hex A0 AA A5) and the new slave ID itself to the module. The individual bytes of this sequence must be sent to register 0. Therefore, four separate I2C write commands are required, with a 50 ms interval between each register write. For example, to change the default ID E0 to F2, register 0 would have to be written with the values A0, AA, A5, and F2 in sequence. The new slave ID should be remembered well, ideally written down somewhere. If you forget the ID, you can find it by the flashing red LED after powering on the module (see table):

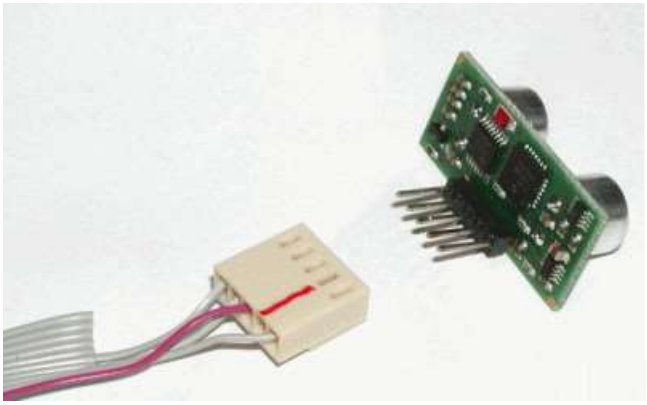
Slave ID		Long blinking	Short flashing
Decimal	Hex		
224	E0	1	0
226	E2	1	1
228	E4	1	2
230	E6	1	3
232	E8	1	4
234	EA	1	5
236	EC	1	6
238	EE	1	7
240	F0	1	8
242	F2	1	9
244	F4	1	10
246	F6	1	11
248	F8	1	12
250	FA	1	13
252	FC	1	14
254	FE	1	15

Don't forget:

There must be no bus nodes with the same slave ID on a bus.

Example program for RN-Control (AVR)





```
'#####################################################
'srf10_ultraschallbeispiel2.bas
'#####################################################


Declare Function Srf10_entfernung(byval Srf10_slaveid As Byte) As Integer Declare Sub
Srf10_reichweite(byval Srf10_slaveid As Byte , Byval Reichweite As Word) Declare Sub
Srf10_verstaerkung(byval Srf10_slaveid As Byte , Byval Srf10_verstaerkung As Byte) Declare
Function Srf10_firmware(byval Srf10_slaveid As Byte) As Byte

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

```
$crystal = 16000000                              'Quartz frequency
$baud = 9600

Config Scl = Portc.0                             'Ports for IIC bus
Config Sda = Portc.1



Dim Entfernung As Integer


   Wait 3                                         'Wait 3 seconds
   I2cinit
   Print "SRF10 Test program "
   Print "SRF 10 Firmware Version:" ; Srf10_firmware(&He0)

   Srf10_reichweite &HE0 , 1000                   'Set range in centimeters
   Srf10_verstaerkung &HE0 , 10                   'Gain factor


   Do
     Entfernung = Srf10_entfernung(&He0)
     Print "Entfernung:" ; Entfernung ; "cm"
     Wait 1
   Loop

End


Function Srf10_entfernung(byval Srf10_slaveid As Byte) As Integer
Local LoB As Byte
Local Hib As Byte
Local Firmware As Byte
Local Temp As Byte
Local Srf10_slaveid_read As Byte

   Srf10_slaveid_read = Srf10_slaveid + 1

   'Messvorgang in starten
   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 0
   I2cwbyte 81                       'measure in centimeters
   I2cstop

Warteaufmessung:
   Waitms 1
   Firmware = Srf10_firmware(&He0)
   If Firmware = 255 Then Goto Warteaufmessung

   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 2                                    'Set reading register
   I2cstop

   I2cstart
   I2cwbyte Srf10_slaveid_read
   I2crbyte Hib , Ack
   I2crbyte Lob , Nack
   I2cstop

   Srf10_entfernung = Makeint(lob , Hib)
End Function



'Messreichweite in cm festlegen
Sub Srf10_reichweite(byval Srf10_slaveid As Byte , Byval Reichweite As Word)
Local Wert As Word
Local Temp As Byte

   Wert = Reichweite / 4    'Approximate register calculation
   Temp = Low(wert)
```

-

```
   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 2                                    'Register
   I2cwbyte Temp
   I2cstop
End Sub


'Verstärung  festlegen
Sub Srf10_verstaerkung(byval Srf10_slaveid As Byte , Byval Srf10_verstaerkung As Byte)
   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 1                                    'Register
   I2cwbyte Srf10_verstaerkung
   I2cstop

End Sub


Function Srf10_firmware(byval Srf10_slaveid As Byte) As Byte
Local Firmware As Byte
Local Srf10_slaveid_read As Byte

   Srf10_slaveid_read = Srf10_slaveid + 1
   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 0                                              'Set reading register'
   I2cstop

   I2cstart
   I2cwbyte Srf10_slaveid_read
   I2crbyte Firmware , Nack
   I2cstop

   Srf10_firmware = Firmware
End Function
```

Warranty

Limited Warranty and Liability

Robotikhardware.de makes no guarantee that the module is suitable for a specific application or purpose. Liability is limited to the replacement of the module. Modules caused by improper use (reverse polarity, incorrect voltages, overvoltages, etc.) cannot be replaced or repaired. There is no liability for damage caused by the use of the module or its failure. If the module is used to build devices, the user must ensure that all necessary tests and approvals are carried out by them.

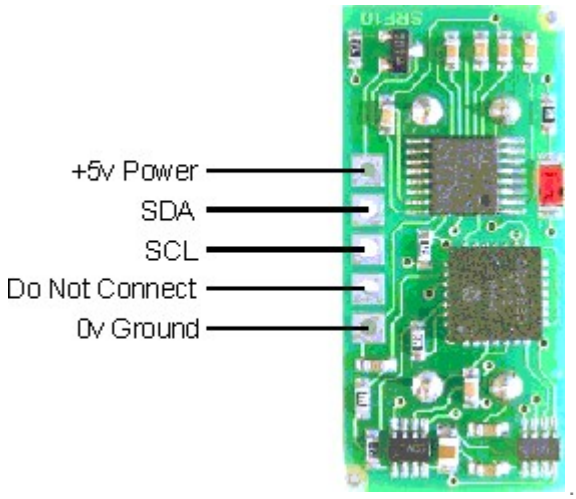
Englische Original Doku vom Hersteller

SRF10 Ultrasonic range finder

Technical Specification

Communication with the SRF10 ultrasonic rangefinder is via the I2C bus. This is available on popular controllers such as the OOPic and Stamp BS2p, as well as a wide variety of micro-controllers. To the programmer the SRF10 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 SRF10 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. We have examples of using the SRF10 module with a wide range of popular controllers.

Connections
The connections to the SRF10 are identical to the SRF08. The "Do Not Connect" pin should be left unconnected. It is actually the CPU MCLR line and is used once only in our workshop to program the PIC16F87 on-board after assembly, and 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 SRF10 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 SRF10 appears as a set of 4 registers.

Location	Read	Write
0	Software Revision	Command Register
1	Unused (reads 0x80)	Max Gain Register (default 16)
2	Range High Byte	Range Register (default 255)
3	Range Low Byte	N/A

Only locations 0, 1 and 2 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 SRF10 software revision. By default, the ranging lasts for 65mS, but can be changed by writing to the range register at location 2. The SRF10 will not respond to commands on the I2C bus whilst it is ranging. See the **Changing Range** and **Analogue Gain** sections below.

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.

Commands

The are three commands to initiate a ranging (80 to 82), to return the result in inches, centimeters or microseconds. There is also a set of commands to change the I2C address.

Command		Action
Decimal	Hex	
80	0x50	Ranging Mode - Result in inches
81	0x51	Ranging Mode - Result in centimeters
82	0x52	Ranging Mode - Result in micro-seconds
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 Mode

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 default and recommended time for completion of ranging is 65mS, however you can shorten this by writing to the range register before issuing a ranging command.

-

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 SRF10 will not respond to any I2C activity whilst ranging. Therefore, if you try to read from the SRF10 (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 SRF10 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 SRF10 is ranging.

Changing the Range

The maximum range of the SRF10 is set by an internal timer. By default, this is 65mS or the equivalent of 11 metres of range. This is much further than the 6 metres the SRF10 is actually capable of. It is possible to reduce the time the SRF10 listens for an echo, and hence the range, by writing to the range register at location 2. The range can be set in steps of about 43mm (0.043m or 1.68 inches) up to 11 metres. The range is ((Range Register x 43mm) + 43mm) so setting the Range Register to 0 (0x00) gives a maximum range of 43mm. Setting the Range Register to 1 (0x01) gives a maximum range of 86mm. More usefully, 24 (0x18) gives a range of 1 metre and 93 (0x5D) is 4 metres. Setting 255 (0xFF) gives the original 11 metres (255 x 43 + 43 is 11008mm). There are two reasons you may wish to reduce the range.

1. To get at the range information quicker
2. To be able to fire the SRF10 at a faster rate.

If you only wish to get at the range information a bit sooner and will continue to fire the SRF10 at 65ms of slower, then all will be well. However if you wish to fire the SRF10 at a faster rate than 65mS, you will definitely need to reduce the gain - see next section.

The range is set to maximum every time the SRF10 is powered-up. If you need a different range, change it once as part of your system initialization code.

Analogue Gain

The analogue gain register sets the *Maximum* gain of the analogue stages. To set the maximum gain, just write one of these values to the gain register at location 1. During a ranging, the analogue gain starts off at its minimum value of 40. This is increased at approx. 96uS intervals up to the maximum gain setting, set by register 1. Maximum possible gain is reached after about 100mm (4inches) of range. The purpose of providing a limit to the maximum gain is to allow you to fire the sonar more rapidly than 65mS. Since the ranging can be very short, a new ranging can be initiated as soon as the previous range data has been read. A potential hazard with this is that the second ranging may pick up a distant echo returning from the previous "ping", give a false result of a close by object when there is none. To reduce this possibility, the maximum gain can be reduced to limit the modules sensitivity to the weaker distant echo, whilst still able to detect close by objects. The maximum gain setting is stored only in the CPU's RAM and is initialized to maximum on power-up, so if you only want do a ranging every 65mS, or longer, you can ignore the Range and Gain Registers. The Gain Register is set to 16 (a gain of 700) at power-up. This can be decreased as required.

Gain Register		Maximum Analogue Gain
Decimal	Hex	
0	0x00	Set Maximum Analogue Gain to 40
1	0x01	As above - Analogue Gain to 40
2	0x02	Set Maximum Analogue Gain to 50
3	0x03	Set Maximum Analogue Gain to 60
4	0x04	Set Maximum Analogue Gain to 70
5	0x05	Set Maximum Analogue Gain to 80
6	0x06	Set Maximum Analogue Gain to 100
7	0x07	Set Maximum Analogue Gain to 120
8	0x08	Set Maximum Analogue Gain to 140
9	0x09	Set Maximum Analogue Gain to 200
10	0x0A	Set Maximum Analogue Gain to 250
11	0x0B	Set Maximum Analogue Gain to 300
12	0x0C	Set Maximum Analogue Gain to 350
13	0x0D	Set Maximum Analogue Gain to 400
14	0x0E	Set Maximum Analogue Gain to 500
15	0x0F	Set Maximum Analogue Gain to 600
16	0x10	Set Maximum Analogue Gain to 700

Note that the relationship between the Gain Register setting and the actual gain is not a linear one. Also there is no magic formula to say "use this gain setting with that range setting". It depends on the size, shape and material of the object and what else is around in the room. Try playing with different settings until you get the result you want. If you appear to get false readings, it may be echo's from previous "pings", try going back to firing the SRF10 every 65mS or longer (slower).
If you are in any doubt about the Range and Gain Registers, remember they are automatically set by the SRF10 to their default values when it is powered-up. You can ignore and forget about them and the SRF10 will work fine, detecting objects up to 6 metres away every 65mS or slower.

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 SRF10 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 SRF10 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 SRF10.

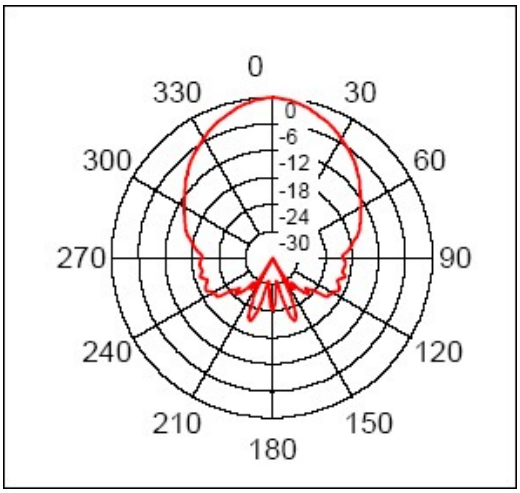
Address		Long Flash	Short flashes
Decimal	Hex		
224	E0	1	0
226	E2	1	1
228	E4	1	2
230	E6	1	3
232	E8	1	4
234	EA	1	5
236	EC	1	6
238	EE	1	7
240	F0	1	8
242	F2	1	9
244	F4	1	10
246	F6	1	11
248	F8	1	12
250	FA	1	13
252	FC	1	14
254	FE	1	15

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

-

Changing beam pattern and beam width

You can't! This is a question which crops up regularly, however there is no easy way to reduce or change the beam width that I'm aware of. The beam pattern of the SRF10 is conical with the width of the beam being a function of the surface area of the transducers and is fixed. It is possible to make the sonar less sensitive to objects off to the side by reducing the maximum gain register from 16 to a lower level. This is a the expense of shorter range, however most small robots don't need 6m of range. A value of 8 (max. gain 140) will reduce the practicable range to about 2m, but it will be much less sensitive to objects off the center line. The beam pattern of the transducers used on the SRF10, taken from the manufacturers data sheet, is shown below.



There is more information in the sonar faq

Mounting the SRF10

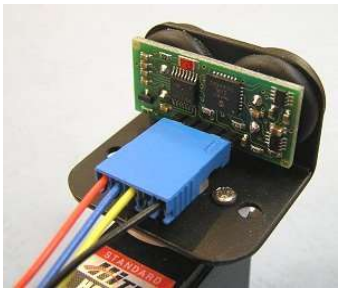
You may have notice that there are no mounting holes on the SRF10 module! That was deliberate to keep the module as small as possible. So how do you mount it? Here are three suggestions:

1. A straight or right angle 0.1 inch connector soldered to your PCB.

2. Using two 9.5mm rubber grommets. Two holes should be drilled into the panel you're mounting the SRF10 to. The hole centers should be 0.7inches (17.78mm) apart and the holes drilled 0.5 inches (12.7mm) in diameter. The two grommets should then be fitted to the panel and the SRF10 gently pushed into them.

3. Using our SRF10 Mounting Kit, shown below.

-

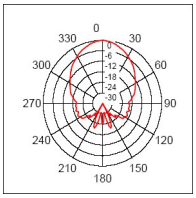


-

Ultrasonic Rangers FAQ

Q. What is the accuracy of the ranging?
A. We quote 3-4cm. Its normally better than this, however so many factors affect accuracy that we won't specify anything better than this. The speed of sound in air is approx. 346m/S at 24 degrees C. At 40KHz the wavelength is 8.65mm. The sonar's detect the echo by listening for the returning wavefronts. This echo has an attack/decay envelope, which means it builds up to a peak then fades away. Depending on which wavefront is the 1st to be strong enough to be detected, which could be the 1st, 2nd or even 3rd, the result can jitter by this much. Another effect which limits accuracy is a phasing effect where the echo is not coming from a point source. Take a wall for example, the ping will bounce off the wall and return to the sonar. The wall is large, however, and there will be reflections from a large area, with reflections from the outside being slightly behind the central reflection. It is the sum of all reflections which the sensor sees which can be either strengthened or weakened by phasing effects. If the echo is weakened then it may be the following wavefront which is detected - resulting in 8.65mm of jitter. It is possible to see changes of distance as small as mm but then get cm of jitter.

Q. How can I narrow the beam width?
A. The beam pattern is conical with the width of the beam being a function of the surface area, frequency and type of transducers and is fixed. The beam patterns of the transducers used on rangers, taken from the manufacturers data sheets, are shown below. beam widths are taken at the -6dB points. There is an interesting article by Harold Carey on reducing the beam width here



Q. What are the units on the vertical axis in the beam pattern diagram?
A. Units are dB, taken from the manufacturers data sheet

Q. What distance above the floor should the sonar be mounted?
A. If you can mount the SRF04/8 12in/300mm above the floor, that should be OK. If you mount them lower, you may need to point them upwards slightly to avoid reflections from the carpet pile or ridges in a concrete floor.

Q. Can we replace the transducers with sealed weatherproof types?
A. No. We have tried these on both the SRF04 and SRF08 and they do not work. The characteristics of the sealed devices requires a new design which is on our future plans list.

Q. What is the RH limit for the transducers?
A. This is not specified by the transducer manufacturers and is not listed in the data sheet. The following is the manufacturers response to an email "The RH here in Taiwan is normally higher than 95%. Just if this sensor(400ST/R160) is used in the air, it should be okay. Don't use in outdoors. Exposing in rainy day or underwater is not allowed."

Q. Is there a need for us to change the SRF08/SRF10 address when using the sensor, can't I just use the default address?
A. Yes, if you only have one sensor you can use the default shipped address of 0xE0. You only need to set addresses if you are using more than one SRF08/SRF10 on the same I2C bus.

Q. Can I fire two or more sonar's at the same time?
A. No! If two or more sonar's are fired together then they could pick up each other "ping" resulting in a

false readings. Fire them sequentially 65mS apart
A. Yes! We do this all the time on our test robot, firing 8 SRF08's at the same time. They are facing outwards and fitted around a 15inch diameter circle. The gain is set to minimum and they are fired using the I2C general call at address 0, and read individually at their set addresses. Under these circumstances there is no direct interference.
A. Possibly! - Try it, and compare the results with firing them sequentially at 65mS intervals..

Q. If I change the SRF08/SRF10 I2C address, will it stay at that address next time I switch on or do I need to set it every time?
A. You only need to set it once and it stays set to the new address - even when you power up again. The I2C address is stored in EEPROM and stays the same until you deliberately change it.

Q. If I change the SRF08 Range and Gain registers, will they stay the same the next time I switch on or do I need to set them every time?
A. Unlike the address, which is permanent, You will need to set the Range and Gain when you power up again.

Q. Can I change the sonar frequency of 40KHz to something else?
A. No. The frequency must be 40KHz, because that is the only frequency the transducers will operate at. Also the circuitry is designed to operate at 40KHz so you cannot change the transducers to other frequency types.

Q. If I reduce the range setting of the SRF08, can I fire the sonar faster?
A. Yes, but be careful. If you fire the sonar and there is nothing in the immediate range, than on the second firing, you may pick up an echo of the first ping which has only just arrived from a distant object. The second ranging will falsely interpret this as an echo from a nearby object. To avoid this, don't fire the sonar more frequently than every 60mS or so.

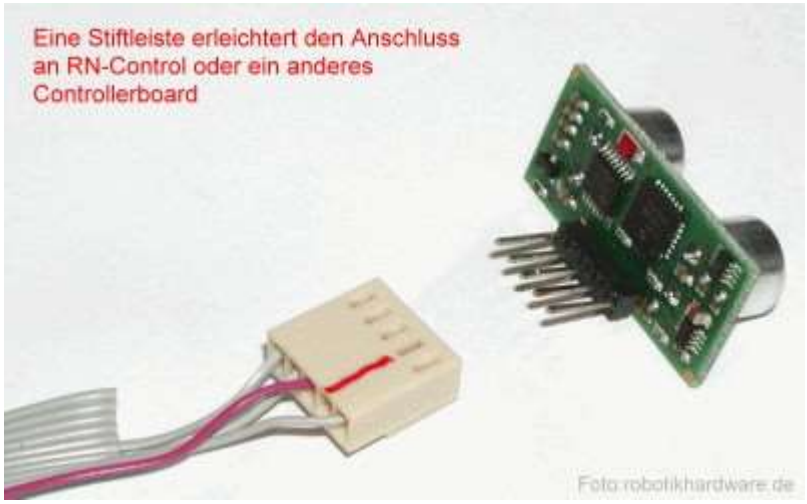
Q. My software master I2C code does not read correct data from the SRF08/SRF10, but its works fine with an I2C EEPROM chip. Why is this?
A. The most likely cause is the master code not waiting for the I2C bus hold. This is where the slave can hold the SCL line low until it is ready. When the master releases SCL (remember it's a passive pull-up, not driven high) the slave may still be holding it low. The master code should then wait until it actually does go high before proceeding.

Q. What is the power output of the ultrasonic burst.
A. On the SRF04, SRF08 and SRF10 its 100-150mW.

Hersteller und englische Doku: Devantech Ltd
Bezug: **www.robotikhardware.de** (Brall Software GmbH)

-

Sample for RN-Control




```
'#####################################################
'srf10_ultraschallbeispiel2.bas
'für
'RoboterNetz Board RN-CONTROL (ab Version 1.1)
'und das SRF10 Ultraschallmodul für Entfernungsmessung
'Datenblatt zu SRF10:
'http://www.roboternetz.de/phpBB2/dload.php?action=file&file_id=310
'Bezug: Robotikhardware.de

'Aufgabe:
' Gibt die Entfernung von Objekten in Zentimetern aus
' Die Messungen werden mit fest vorgegebenen Verstärkungsfaktor
' ermittelt

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


Declare Function Srf10_entfernung(byval Srf10_slaveid As Byte) As Integer
Declare Sub Srf10_reichweite(byval Srf10_slaveid As Byte , Byval Reichweite As Word)
Declare Sub Srf10_verstaerkung(byval Srf10_slaveid As Byte , Byval Srf10_verstaerkung As Byte)
Declare Function Srf10_firmware(byval Srf10_slaveid As Byte) As Byte

$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



Dim Entfernung As Integer


   Wait 3                                         'Warte 3 Sekunden
   I2cinit
   Print "SRF10 Testprogramm "
   Print "SRF 10 Firmware Version:" ; Srf10_firmware(&He0)

   Srf10_reichweite &HE0 , 1000                   'Reichweite in Zentimetern festlegen
   Srf10_verstaerkung &HE0 , 10                   'Verstärkungsfaktor


   Do
     Entfernung = Srf10_entfernung(&He0)
     Print "Entfernung:" ; Entfernung ; "cm"
     Wait 1
   Loop

End


Function Srf10_entfernung(byval Srf10_slaveid As Byte) As Integer
Local LoB As Byte
Local Hib As Byte
Local Firmware As Byte
Local Temp As Byte
Local Srf10_slaveid_read As Byte

   Srf10_slaveid_read = Srf10_slaveid + 1

   'Messvorgang in starten
   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 0
   I2cwbyte 81                                    'in Zentimetern messen
   I2cstop

Warteaufmessung:
   Waitms 1
   Firmware = Srf10_firmware(&He0)
   If Firmware = 255 Then Goto Warteaufmessung

   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 2                                   Leseregister festlegen
   I2cstop

   I2cstart
   I2cwbyte Srf10_slaveid_read
   I2crbyte Hib , Ack
   I2crbyte Lob , Nack
   I2cstop

   Srf10_entfernung = Makeint(lob , Hib)
End Function



'Messreichweite in cm festlegen
Sub Srf10_reichweite(byval Srf10_slaveid As Byte , Byval Reichweite As Word)
Local Wert As Word
Local Temp As Byte
```

-

```
   Wert = Reichweite / 4                        'Ungefähre Registerberechnung
   Temp = Low(wert)

   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 2                                   'Register
   I2cwbyte Temp
   I2cstop
End Sub


'Verstärung  festlegen
Sub Srf10_verstaerkung(byval Srf10_slaveid As Byte , Byval Srf10_verstaerkung As Byte)
   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 1                                   'Register
   I2cwbyte Srf10_verstaerkung
   I2cstop

End Sub


Function Srf10_firmware(byval Srf10_slaveid As Byte) As Byte
Local Firmware As Byte
Local Srf10_slaveid_read As Byte

   Srf10_slaveid_read = Srf10_slaveid + 1
   I2cstart
   I2cwbyte Srf10_slaveid
   I2cwbyte 0                                               'Leseregister festlegen
   I2cstop

   I2cstart
   I2cwbyte Srf10_slaveid_read
   I2crbyte Firmware , Nack
   I2cstop

   Srf10_firmware = Firmware
End Function
```