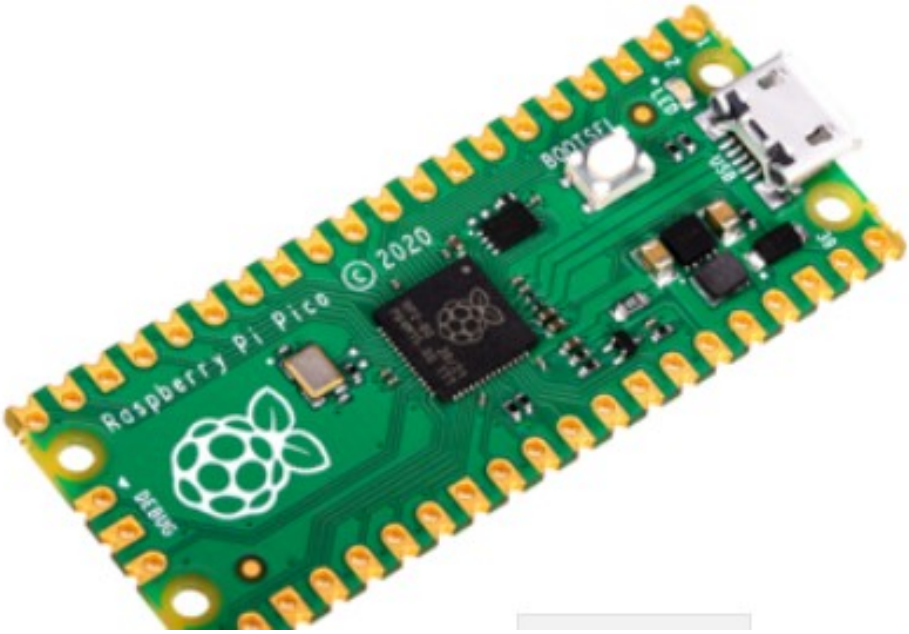
**********Default**********

Programming Pico with Ardiuno IDE
for Nano-RP2040-Connect
Ultra Sonic sensor with PIO

generating trigger
and receiving echo
and a 2nd version using pico-sdk
10/18/22

**********Default**********

Pico

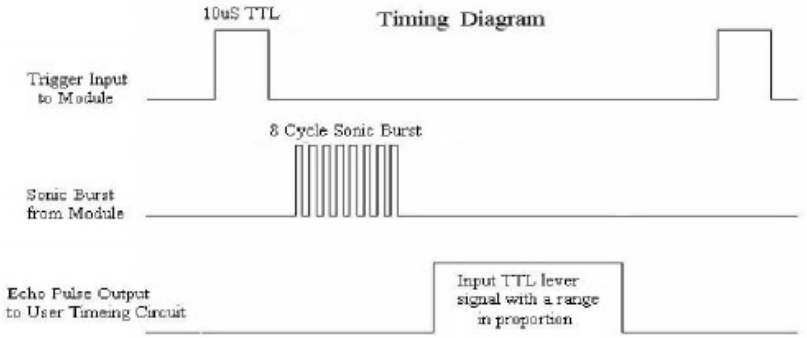


Ultrasonic sensor

Vcc Trig Echo GND

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.




```
diff --git a/CMakeLists.txt b/CMakeLists.txt
index 5fbbfb3..e255b4d 100644
--- a/CMakeLists.txt
+++ b/CMakeLists.txt
@@ -22,6 +22,7 @@ add_subdirectory(Semaphore)
 add_subdirectory(Mutex)
```

```
 add_subdirectory(Scheduling)
 add_subdirectory(first_pwm)
+add_subdirectory(HCSR04)^M
 add_subdirectory(2tasks)
 add_subdirectory(ultibo_blink)
 add_subdirectory(pico-lifting-sf)
```

```
export PICO_SDK_PATH=../pico-sdk/

make

 openocd -f interface/raspberrypi-swd.cfg -f target/rp2040.cfg -c "program HCSR04/HCSR04.elf
verify reset exit"
```

A 2nd source code was used from https://github.com/GitJer/Some_RPI-Pico_stuff..
Minor changes were required to 2 files CmakeLists.txt & HCSR04.cpp.

HCSR04.cpp

```
86,87c86,87
<     // the instance of the HCSR04 (Echo pin = 4, Trig pin = 7)
<     HCSR04 my_HCSR04(4, 7);
---
>     // the instance of the HCSR04 (Echo pin = 14, Trig pin = 15)
>     HCSR04 my_HCSR04(14, 15);
94c94
<         sleep_ms(1000);
---
>         sleep_ms(100);
96c96
< }
---
> }
\ No newline at end of file
```

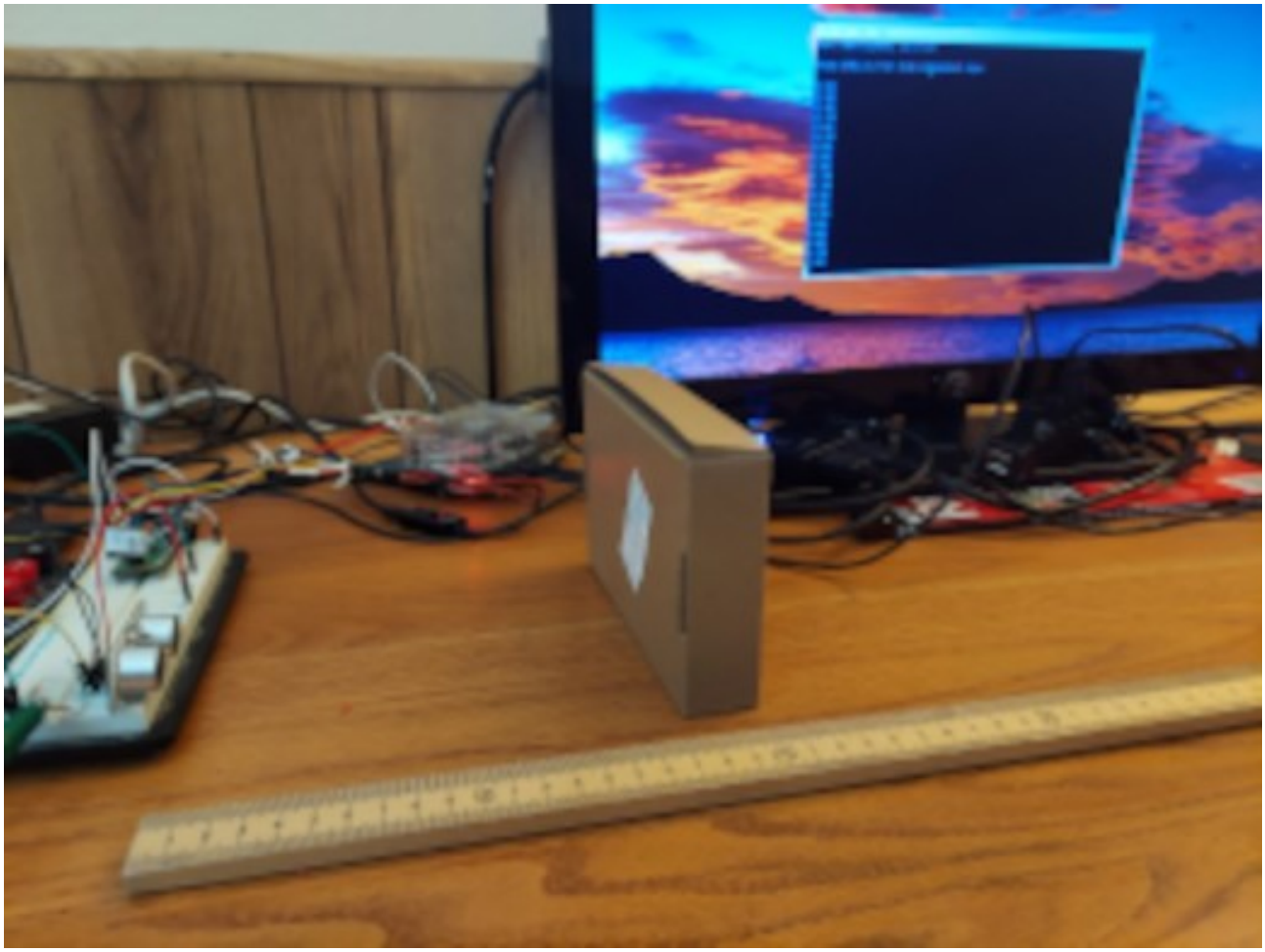
CMakeLists.txt

```
13,14c13
< pico_enable_stdio_usb(HCSR04 1)
< pico_enable_stdio_uart(HCSR04 0)
---
>
16c15
< #example_auto_set_url(HCSR04)
---
> example_auto_set_url(HCSR04)
```

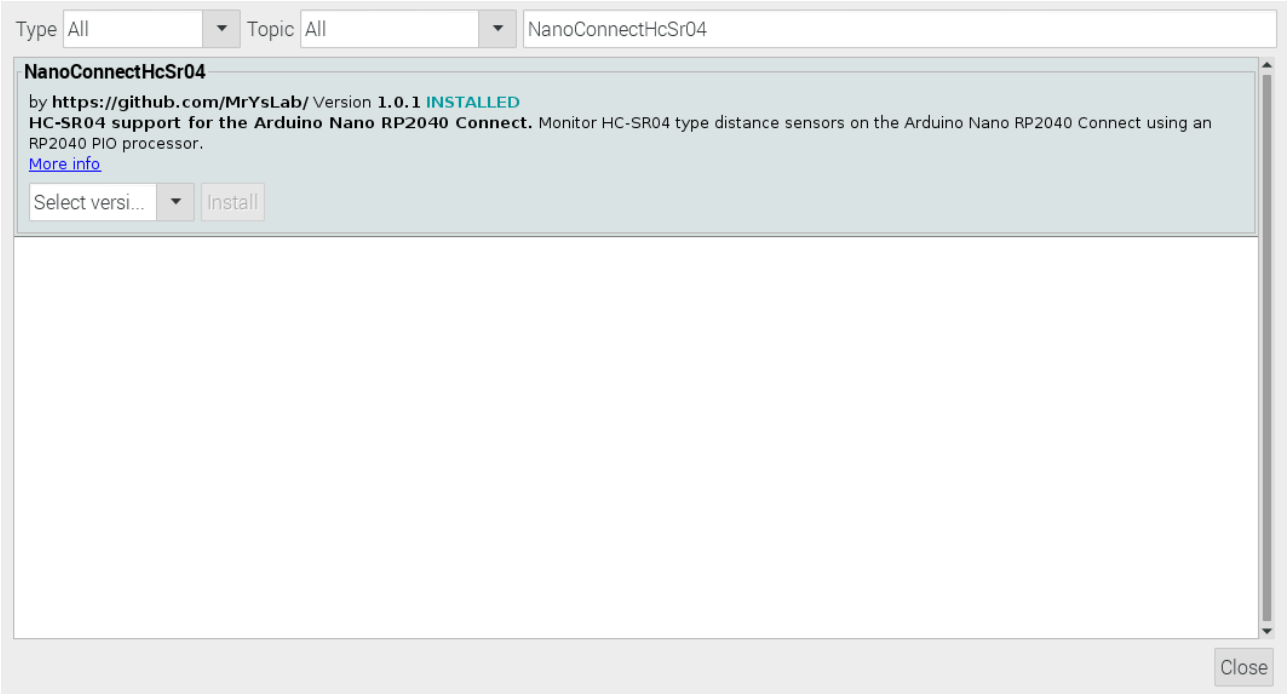
I used 5 volts from the pico to power the HCSR05. This required using 2 10K ohm resistors on the echo signal.

This was the origin of this project.
https://github.com/develone/my-projects-docs/blob/master/HCSR04/ultrasonic-sonar-distance-sensors.pdf

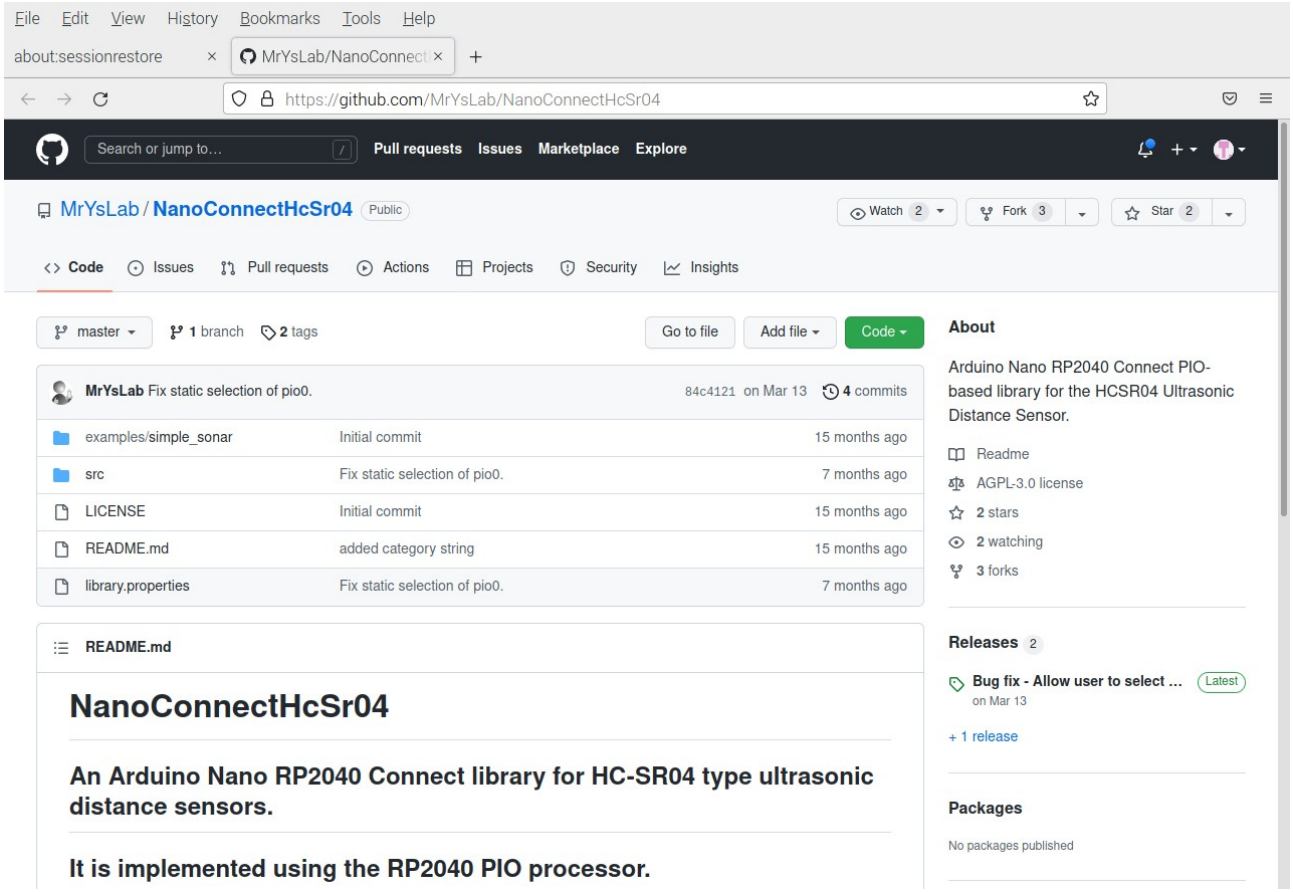
This is sensor sending data over the USB to a serial terminal.



Arduino Library Manager



Github



https://github.com/MrYsLab/NanoConnectHcSr04/blob/master/src/NanoConnectHcSr04.pio.h

https://github.com/MrYsLab/NanoConnectHcSr04/blob/master/src/NanoConnectHcSr04.cpp

This sketch uses NanoConnectHcSr04 library

File Edit Sketch Tools Help

HcSr04

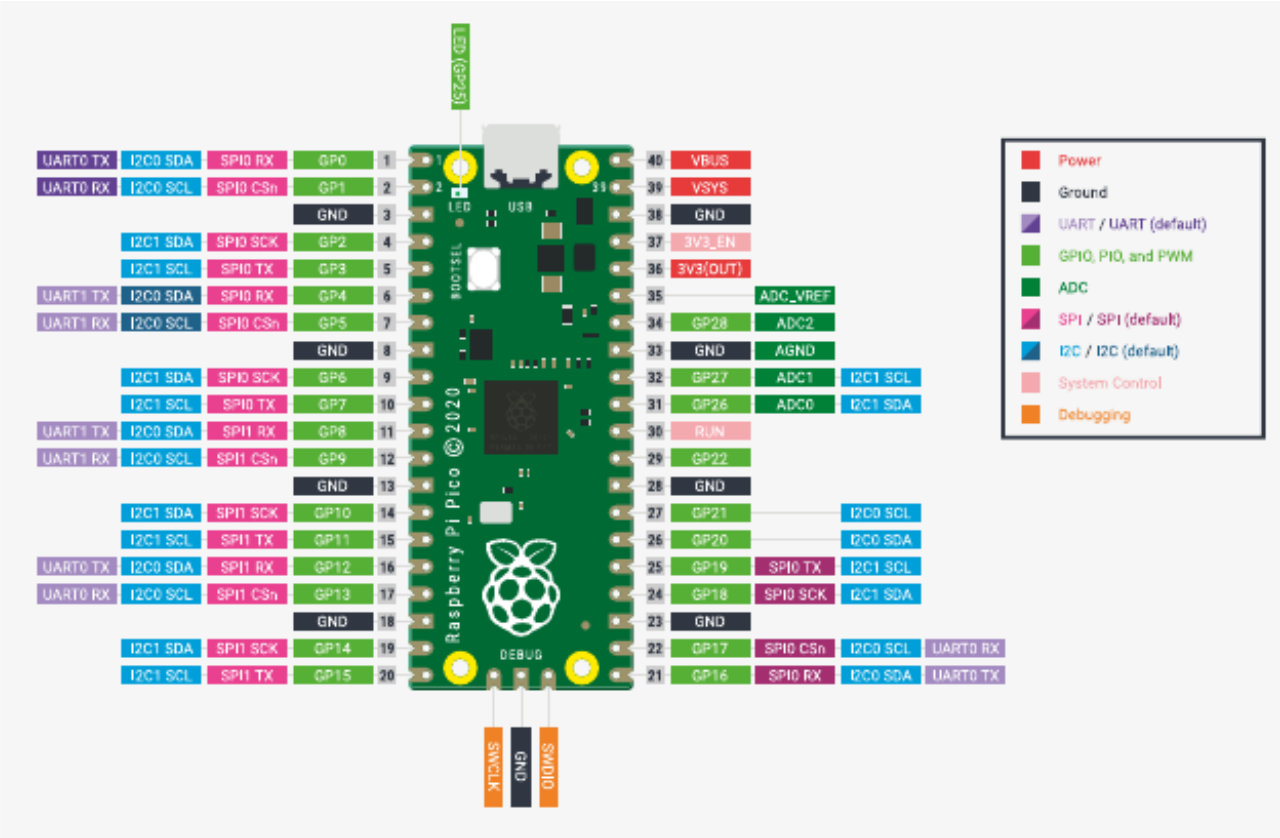
```
#include <NanoConnectHcSr04.h>

// D11 (7) == trigger, D12 (4) == echo
// Using pio0 and sm 0
NanoConnectHcSr04 sonar(7,4, pio0, 0);

// distance value returned
float value = 0.0;

void setup() {
    Serial.begin(115200);


}

void loop() {
    // put your main code here, to run repeatedly:
    value = sonar.readSonar();
    Serial.println(value);
    delay(1000);
}
```

Done compiling.

```
Sketch uses 92240 bytes (0%) of program storage space. Ma
Global variables use 43520 bytes (16%) of dynamic memory,
```

20 Arduino Nano RP2040 Connect on /dev/ttyACM0

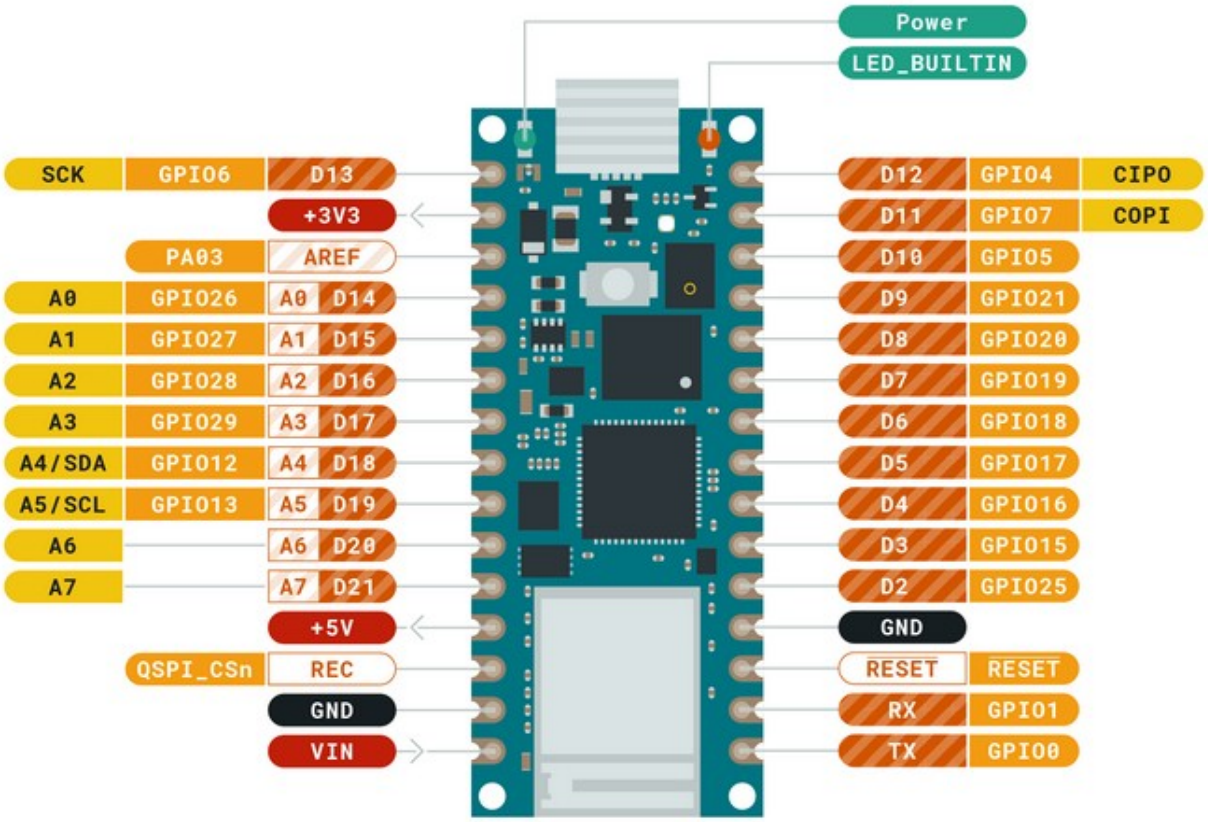
For the pico gpio 7 is pin 10 gpio 4 is pin 6



For the nano-rp2040-connect gpio 7 is pin 12 gpio 4 is pin 11

ARDUINO

ARDUINO NANO RP2040 CONNECT



Ground
Power
LED
Internal Pin
SWD Pin
Other Pin
Digital Pin
Analog Pin
Default
Microcontroller's Port

ARDUINO.CC

```
File  Edit  Tabs  Help
38.77
37.90
38.77
38.75
38.30
38.75
38.30
38.30
38.75
38.75
38.77
38.75
38.75
38.75
38.31
38.74
38.31
38.33
38.75
38.77
38.75
38.31
38.74

CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.8 | VT102 | Offline | ttyACM0
```

Serial Port configuration

```
File  Edit  Tabs  Help
19.10
19.54
19.54
19.5+-----------------------------------------------------------------------+
19.5| A -    Serial Device      : /dev/ttyACM0                              |
19.1| B - Lockfile Location     : /var/lock                                 |
19.5| C -   Callin Program      :                                           |
19.5| D -  Callout Program      :                                           |
19.5| E -    Bps/Par/Bits       : 115200 8N1                                |
19.1| F - Hardware Flow Control : No                                        |
19.5| G - Software Flow Control : No                                        |
19.5| H -     RS485 Enable      : No                                        |
19.5| I -   RS485 Rts On Send   : No                                        |
19.5| J -  RS485 Rts After Send : No                                        |
19.5| K -  RS485 Rx During Tx   : No                                        |
19.5| L -  RS485 Terminate Bus  : No                                        |
19.5| M - RS485 Delay Rts Before: 0                                         |
19.5| N - RS485 Delay Rts After : 0                                         |
19.5|                                                                       |
19.5|    Change which setting?                                              |
19.5+-----------------------------------------------------------------------+
19.54
19.54
```