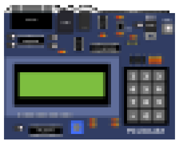


PICSimLab_0_8_9

Luis Claudio Gambôa Lopes <lcgamboa@yahoo.com>

Download: github

HTML version of documentation

June 16, 2021

Contents

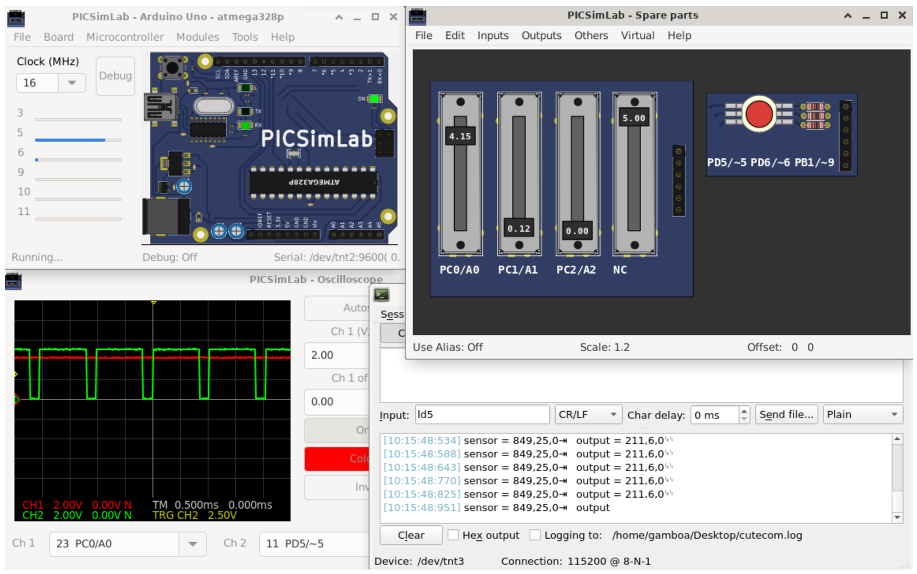
Chapter 1

Introduction

PICSimLab means PIC Simulator Laboratory

PICSimLab is a realtime emulator of development boards with integrated MPLABX/avr-gdb debugger. PICSimLab supports some picsim microcontrollers and some simavr microcontrollers. PICSimLab have integration with MPLABX/Arduino IDE for programming the boards microcontrollers.

The experimental version supports uCsim, gpsim and qemu-stm32 simulators in addition to the stable ones.



Use Alias: Off Scale: 1.2 Offset: 0 0

Input: ld5 CR/LF Char delay: 0 ms Send file... Plain

```
[10:15:48:534] sensor = 849,25,0⇥  output = 211,6,0
[10:15:48:588] sensor = 849,25,0⇥  output = 211,6,0
[10:15:48:643] sensor = 849,25,0⇥  output = 211,6,0
[10:15:48:770] sensor = 849,25,0⇥  output = 211,6,0
[10:15:48:825] sensor = 849,25,0⇥  output = 211,6,0
[10:15:48:951] sensor = 849,25,0⇥  output
```

Logging to: /home/gamboa/Desktop/cutecom.log

Device: /dev/tnt3 Connection: 115200 @ 8-N-1

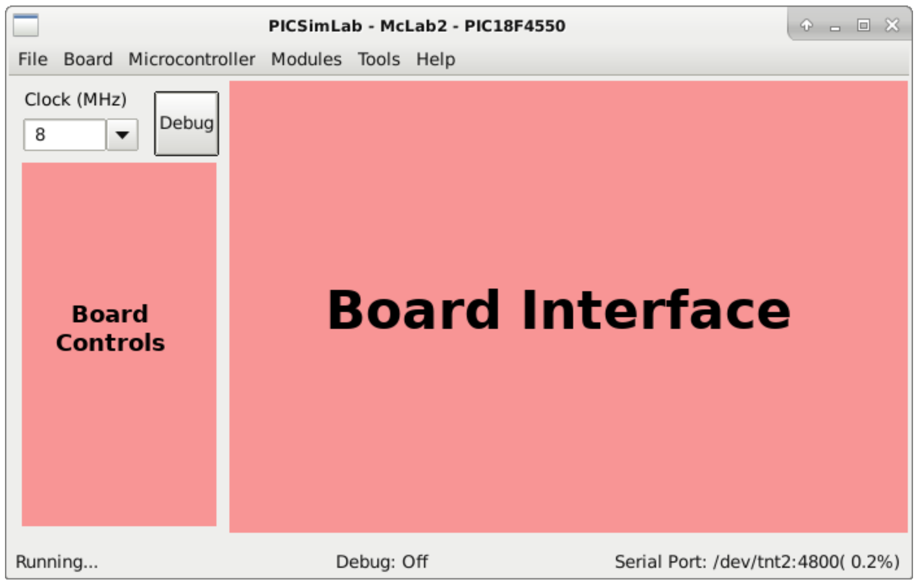
Chapter 2

Simulator Interface

2.1 Main Window

The main window consists of a menu, a status bar, a frequency selection combobox, an on/off button to trigger debugging, some board-specific controls and the part of the board interface itself.

In the title of the window is shown the name of the simulator PICSimLab, followed by the board and the microcontroller in use.



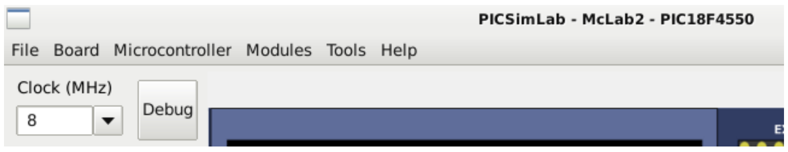
The frequency selection combobox directly changes the working speed of the microcontroller, when the “Clock (MHz)” label goes red indicates that the computer is not being able to run the program in real time for the selected clock. In this case the simulation may present some difference than expected and the CPU load will be increased.

The on/off button to enable debugging is used to enable debugging support, with the active support there is a higher simulation load.

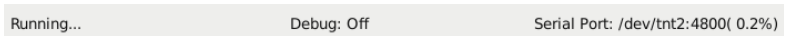
The menus and their functions are listed below:

- File
  - Load Hex - Load .hex files
  - Reload Last - Reload the last used .hex file
  - Save Hex - Save memory in a .hex file
  - Configure - Open the configuration windows
  - Save Workspace - Saves all current workspace settings to a .pzw file
  - Load Workspace - Loads saved settings from a .pzw file
  - Exit
- Board
  - Breadboard - Choose board Breadboard
  - McLab1 - Choose board McLab1
  - K16F - Choose board K16F
  - McLab2 - Choose board McLab2
  - PICGenios - Choose board PICGenios
  - Arduino Uno - Choose board Arduino Uno
- Microcontroller
  - xxxxx - Selects the microcontroller to be used (depends on the selected board)
- Modules
  - Oscilloscope - Open the oscilloscope window
  - Spare parts - Open the spare parts window
- Tools
  - Serial Terminal - Open the serial terminal Cutecom
  - Serial Remote Tank - Open the remote tank simulator
  - Esp8266 Modem Simulator - Open the Esp8266 Modem Simulator
  - Arduino Bootloader - Load microcontroller with Arduino serial bootloader
  - MPLABX Debugger Plugin - Open the web page to download the MPLABX Debugger Plugin
  - Pin Viewer - Open the Pin Viewer

- Help
  - Contents - Open the Help window
  - Board - Open the Board Help window
  - Examples - Load the examples
  - About Board - Show message about author and version of board
  - About PICSimLab - Show message about author and version of PICSimLab



The first part of the status bar shows the state of the simulation, in the middle part the status of the debug support and in the last part the name of the serial port used, its default speed and the error in relation to the real speed configured in the microcontroller.



```
Running...                    Debug: Off              Serial Port: /dev/tnt2:4800( 0.2%)
```

2.2 Interaction with the Board

On the interface area of the board it is possible to interact in some ways:

- Click in ICSP connector to load an .hex file.
- Click in PWR button to ON/OFF the emulator..
- The buttons can be activated through mouse or keys 1, 2, 3 e 4.

2.3 Command Line

PICSimLab supports two command lines format:
One for load a PICSimLab Workspace file (.pzw)

```
picsimlab file.pzw
```

And other for load .hex files

```
picsimlab boardname microcontroller [file.hex] [file.pcf]
```

2.4 Remote Control Interface

The PICSimLab remote control interface supports TCP connections using telnet or nc (netcat).

The default port is 5000 and can be changed in configuration windows.

The 'rlwrap' command can be used for best command edition support in telnet or nc:

```
 rlwrap nc 127.0.0.1 5000
```

The supported commands can be shown using the "help" command:

```
 help
List of supported commands:
  dumpe [a] [s]- dump internal EEPROM memory
  dumpf [a] [s]- dump Flash memory
  dumpr [a] [s]- dump RAM memory
  exit         - shutdown PICSimLab
  get ob       - get object value
  help         - show this message
  info         - show actual setup info and objects
  pins         - show pins directions and values
  pinsl        - show pins formated info
  quit         - exit remote control interface
  reset        - reset the board
  set ob vl    - set object with value
  sync         - wait to syncronize with timer event
  version      - show PICSimLab version
Ok
```

The "info" command show all available "objects" and values:

```
info
Board:     Arduino Uno
Processor: atmega328p
Frequency:   16000000 Hz
Use Spare: 1
    board.out[00] LD_L= 254
  part[00]: LEDs
    part[00].out[08] LD_1= 254
    part[00].out[09] LD_2= 30
    part[00].out[10] LD_3= 254
    part[00].out[11] LD_4= 254
    part[00].out[12] LD_5 254
    part[00].out[13] LD_6= 254
    part[00].out[14] LD_7= 254
  part[01]: Buzzer
    part[01].out[02] LD_1= 140
  part[02]: Push buttons
    part[02].in[00] PB_1= 1
    part[02].in[01] PB_2= 0
    part[02].in[02] PB_3= 1
    part[02].in[03] PB_4= 1
    part[02].in[04] PB_5= 1
    part[02].in[05] PB_6= 1
    part[02].in[06] PB_7= 1
    part[02].in[07] PB_8= 1
Ok
```

The “pins” command show all pins directions and digital values:

```
pins
  pin[01] ( PC6/RST) < 0              pin[15] (  PB1/~9) > 0
  pin[02] (   PD0/0) < 1              pin[16] ( PB2/~10) > 0
  pin[03] (   PD1/1) < 1              pin[17] ( PB3/~11) > 0
  pin[04] (   PD2/2) < 1              pin[18] (  PB4/12) < 0
  pin[05] (  PD3/~3) > 0              pin[19] (  PB5/13) > 0
  pin[06] (   PD4/4) < 1              pin[20] (     +5V) < 1
  pin[07] (     +5V) < 1              pin[21] (    AREF) < 0
  pin[08] (     GND) < 0              pin[22] (     GND) < 0
  pin[09] (  PB6/X1) < 0              pin[23] (  PC0/A0) < 0
  pin[10] (  PB7/X2) < 0              pin[24] (  PC1/A1) < 0
  pin[11] (  PD5/~5) < 1              pin[25] (  PC2/A2) < 0
  pin[12] (  PD6/~6) < 1              pin[26] (  PC3/A3) < 0
  pin[13] (   PD7/7) < 1              pin[27] (  PC4/A4) > 0
  pin[14] (   PB0/8) > 0              pin[28] (  PC5/A5) > 0
Ok
```

The “pinsl” command show all pins info in text formatted output:

```
pinsl
28 pins [atmega328p]:
  pin[01] D I 0 000 0.000 "PC6/RST "
  pin[02] D I 1 200 0.000 "PD0/0   "
  pin[03] D I 1 200 0.000 "PD1/1   "
  pin[04] D I 1 200 0.000 "PD2/2   "
  pin[05] D O 0 007 0.000 "PD3/~3  "
  pin[06] D I 1 200 0.000 "PD4/4   "
  pin[07] P I 1 200 0.000 "+5V     "
  pin[08] P I 0 000 0.000 "GND     "
  pin[09] D I 0 000 0.000 "PB6/X1  "
  pin[10] D I 0 000 0.000 "PB7/X2  "
  pin[11] D I 1 200 0.000 "PD5/~5  "
  pin[12] D I 1 200 0.000 "PD6/~6  "
  pin[13] D I 1 200 0.000 "PD7/7   "
  pin[14] D O 0 000 0.000 "PB0/8    "
  pin[15] D O 0 000 0.000 "PB1/~9   "
  pin[16] D O 0 000 0.000 "PB2/~10  "
  pin[17] D O 0 006 0.000 "PB3/~11  "
  pin[18] D I 0 000 0.000 "PB4/12   "
  pin[19] D O 0 000 0.000 "PB5/13   "
  pin[20] P I 1 200 0.000 "+5V      "
  pin[21] R I 0 000 0.000 "AREF     "
  pin[22] P I 0 000 0.000 "GND      "
  pin[23] A I 0 000 0.875 "PC0/A0   "
  pin[24] A I 0 000 1.925 "PC1/A1   "
  pin[25] A I 0 000 2.700 "PC2/A2   "
  pin[26] A I 0 000 4.275 "PC3/A3   "
  pin[27] D O 1 179 0.000 "PC4/A4   "
  pin[28] D O 1 186 0.000 "PC5/A5   "
Ok
```

You can view one input/output state using the “get” command:

```
get board.out[00]

get part[02].in[01]
```

Its possible use the “get” command to view individual pins state:

```
#digital state
get pin[19]
pin[19]= 0
Ok

#digital mean value (0-200)
get pinm[19]
pin[18]= 100
Ok

#analog state
get apin[25]
apin[25]= 2.700
Ok

#all info
get pinl[13]
pin[13] D I 1 200 0.000 "PD7/7    "
Ok
```

And set value of one input using the “set” command:

```
set part[02].in[01]  0
set part[02].in[01]  1
```

Or set value of one pin using the “set” command:

```
#digital
set pin[10]  2

#analog
set apin[20] 2.345
```

For windows users putty telnet client is a good option.

Chapter 3

Boards

PICSimLab currently supports five backend simulators. The stable version supports picsim and simavr. The experimental version supports uCsim, gpsim and qemu-stm32 in addition to the stable ones.

The Figure 3.1 shows which cards are based on which backend simulator:

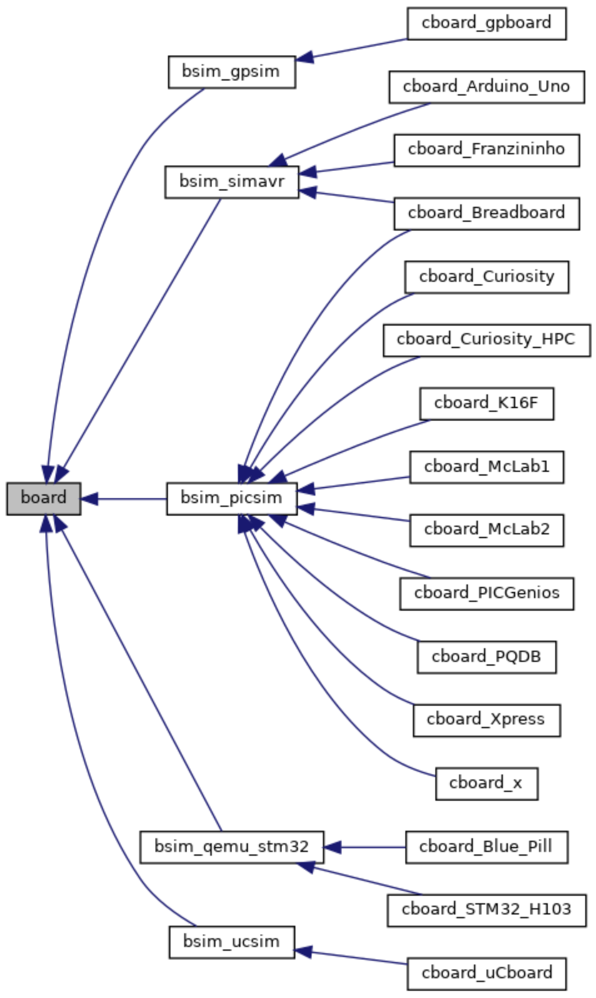


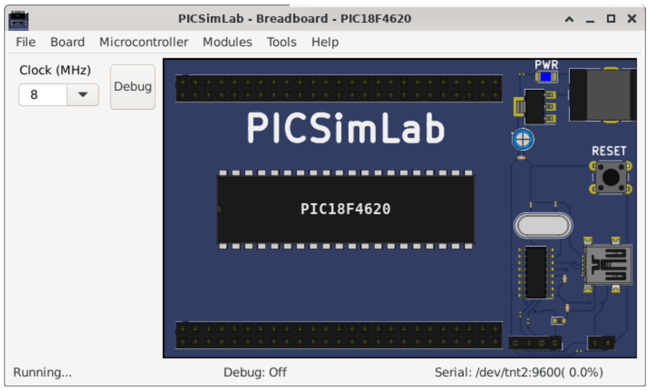
Figure 3.1: Boards backend simulators

The below table show the supported debug interface of each simulator:

Backend	**Debug Support**
picsim	MPLABX Integrated Debug (see section 6.1)
simavr	MPLABX Integrated Debug (see section 6.1) and remote avr-gdb (see section 6.3)
qemu-stm32	remote arm-gdb (see Chapter 6.4)
uCsim	uCsim remote console (telnet) (see section 6.5)
gpsim	none yet

3.1 Breadboard

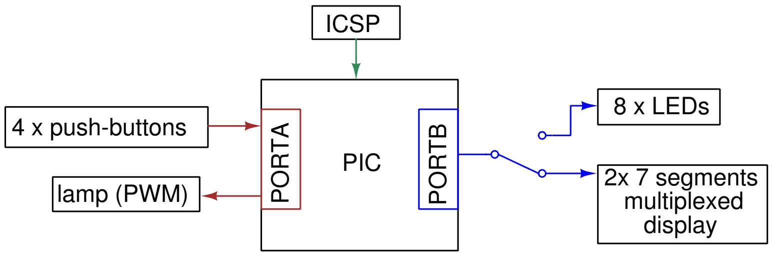
It is a generic board only with reset, serial and crystal circuits and support to multiple microcontrollers of picsim and simavr.

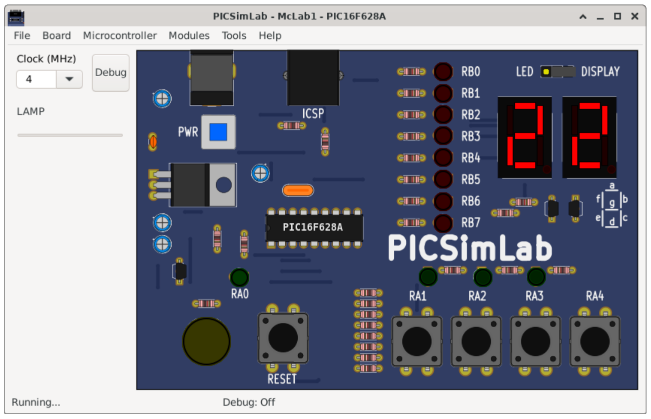

PICSimLab - Breadboard - PIC18F4620
File Board Microcontroller Modules Tools Help
Clock (MHz)
8
Debug
PICSimLab
PIC18F4620
PWR
RESET
Running...
Debug: Off
Serial: /dev/tnt2:9600(0.0%)

Examples

3.2 McLab1

It emulates the Labtools development board McLab1 that uses one PIC16F84, PIC16F628 or PIC16F648 of picsim.





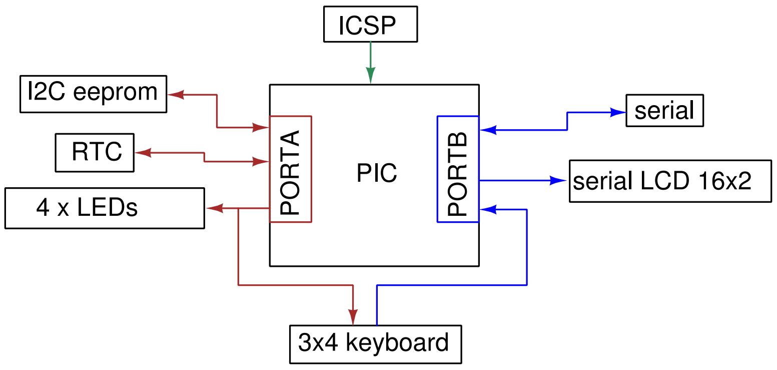
Board McLab1 schematics.

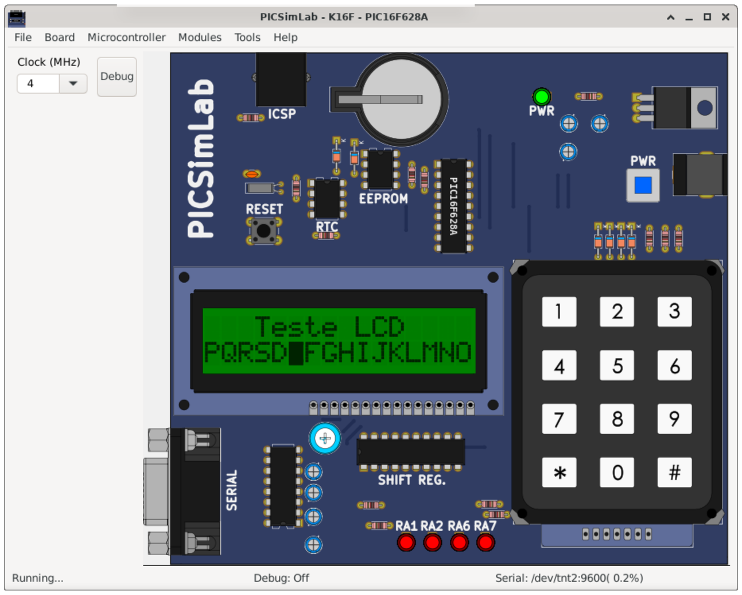
The code examples can be loaded in PICSimLab menu **Help->Examples**.

The source code of board McLab1 examples using MPLABX and XC8 compiler are in the link: board_McLab1.

3.3 K16F

It emulates an didactic board developed by author that uses one PIC16F84, PIC16F628 or PIC16F648 of picsim.



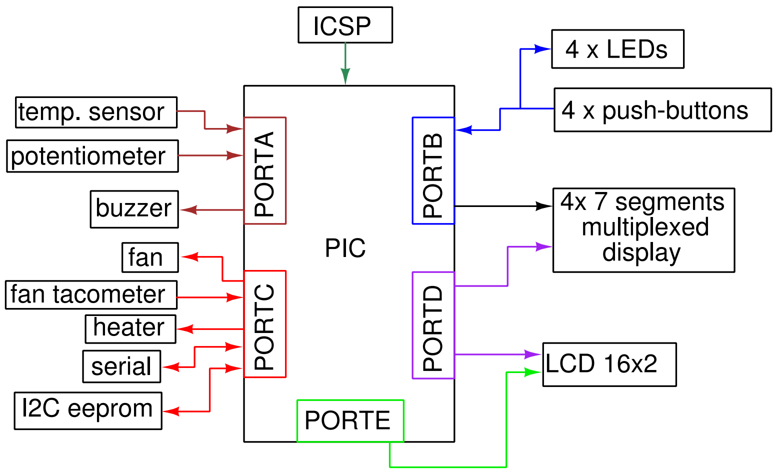


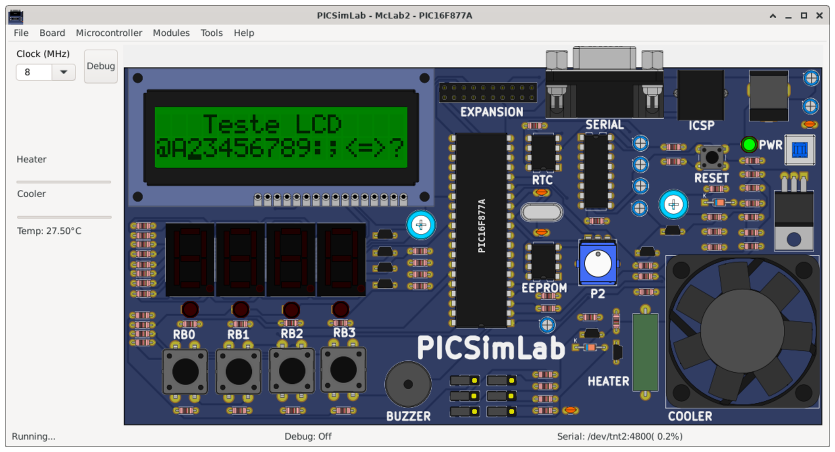
Board K16F schematics.

The code examples can be loaded in PICSimLab menu **Help->Examples**.
The source code of board K16F examples using MPLABX and XC8 compiler are in the link: board_K16F.

3.4 McLab2

It emulates the Labtools development board McLab2 that uses one PIC16F777, PIC16F877A, PIC18F452, PIC18F4520, PIC18F4550 or PIC18F4620 of picsim.





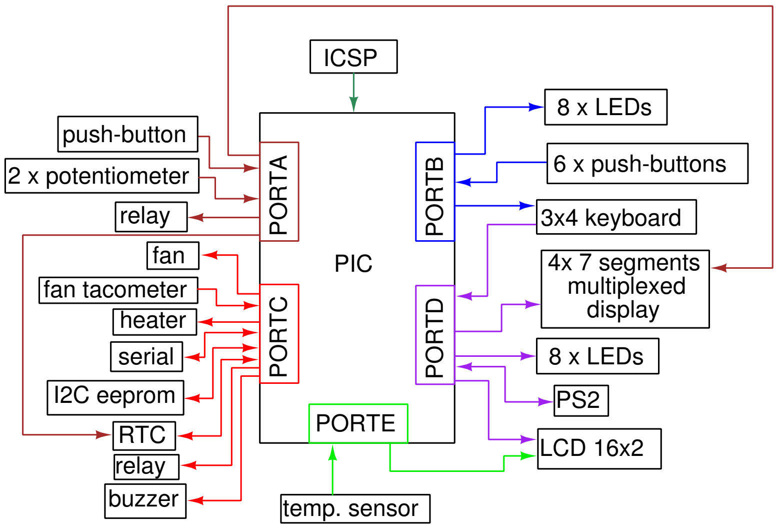
Board McLab2 schematics.

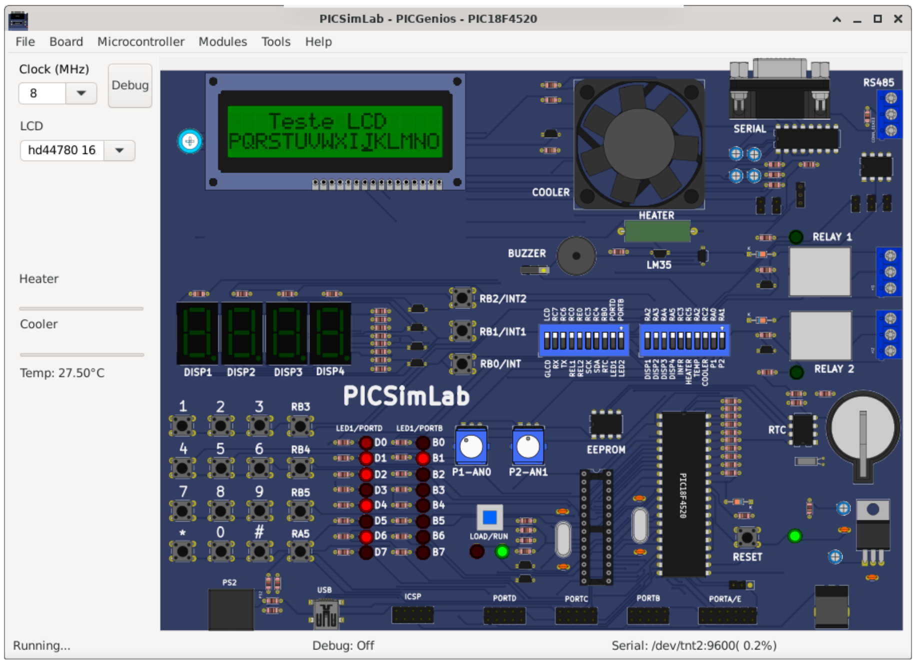
The code examples can be loaded in PICSimLab menu **Help->Examples**.

The source code of board McLab2 examples using MPLABX and XC8 compiler are in the link: board_McLab2.

3.5 PICGenios

It emulates the microgenius development board PICGenios PIC18F e PIC16F Microchip that uses one PIC16F777, PIC16F877A, PIC18F452, PIC18F4520, PIC18F4550 or PIC18F4620 of picsim.





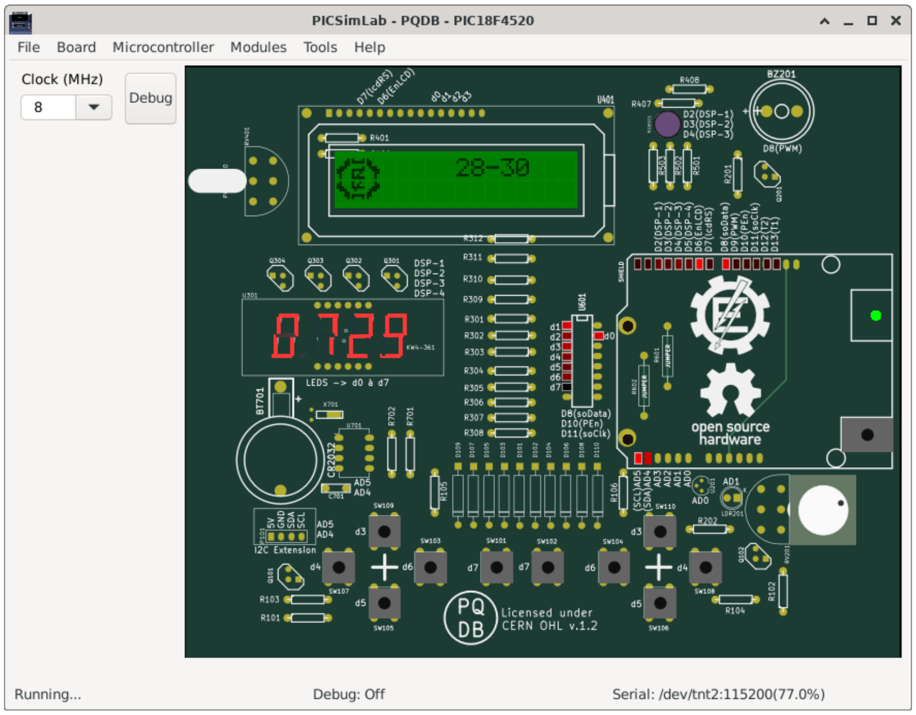
Board PICGenios schematics.

The code examples can be loaded in PICSimLab menu **Help->Examples**.

The source code of board PICGenios examples using MPLABX and XC8 compiler are in the link: board_PICGenios.

3.6 PQDB

The PQDB board is an opensource/openhardware project, more info at https://github.com/projetopqdb/. It was developed to be used with arduino/freedom boards, but adapted to use the microcontroller PIC18F4520 of picsim on PICSImLab.

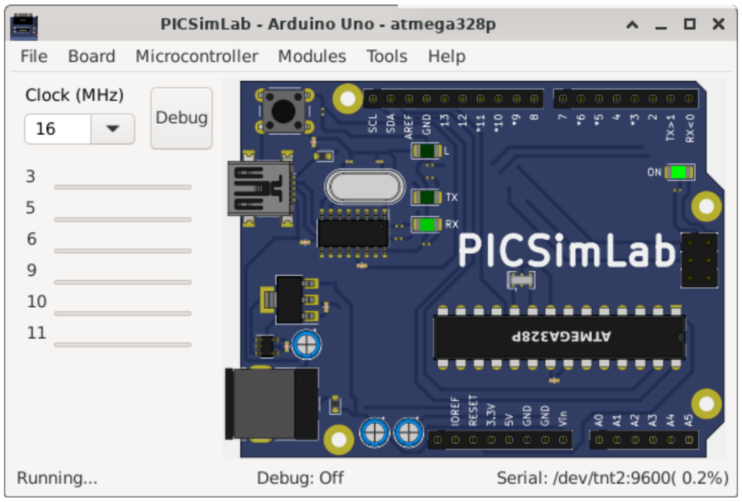


Running...   Debug: Off   Serial: /dev/tnt2:115200(77.0%)

Examples

3.7 Arduino Uno

It emulates the Arduino Uno development board that uses one ATMEGA328P microcontroller of simavr.

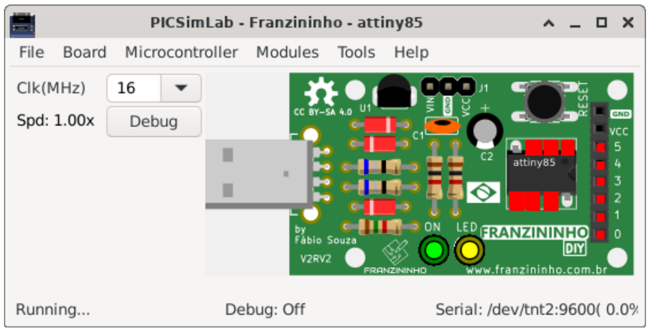

PICSimLab - Arduino Uno - atmega328p
File Board Microcontroller Modules Tools Help
Clock (MHz)
16
Debug
3
5
6
9
10
11
PICSimLab
ATMEGA328P
Running...
Debug: Off
Serial: /dev/tnt2:9600(0.2%)

Board Arduino Uno schematics.

The code examples can be loaded in PICSimLab menu **Help->Examples**.
The source code of board Arduino Uno examples using the Arduino IDE with avr-gcc are in the link: board_Arduino_Uno.
More information about the Arduino in www.arduino.cc

3.8 Franzininho

The Franzininho DIY board is an openhardware project, more info at https://franzininho.com.br/. It was developed to be used with the microcontroller ATtiny85 of of simavr on PICSImLab.


PICSimLab - Franzininho - attiny85
File Board Microcontroller Modules Tools Help
Clk(MHz)
16
Spd: 1.00x
Debug
attiny85
FRANZININHO DIY
www.franzininho.com.br
Running...
Debug: Off
Serial: /dev/tnt2:9600(0.0%

Examples

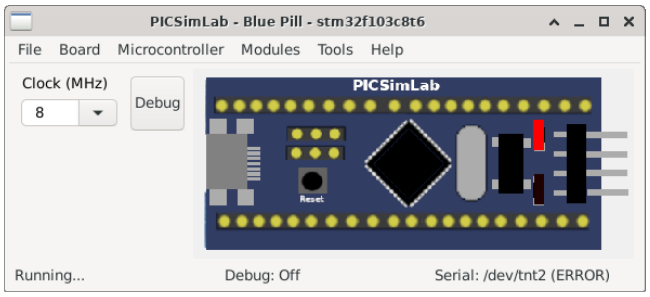
Chapter 4

Experimental Boards

Boards in the experimental phase. Probably with some bugs and missing features.

4.1 Blue Pill

It is a generic board only with reset, serial and crystal circuits and support to stm32f103c8t6 microcontroller of qemu-stm32.

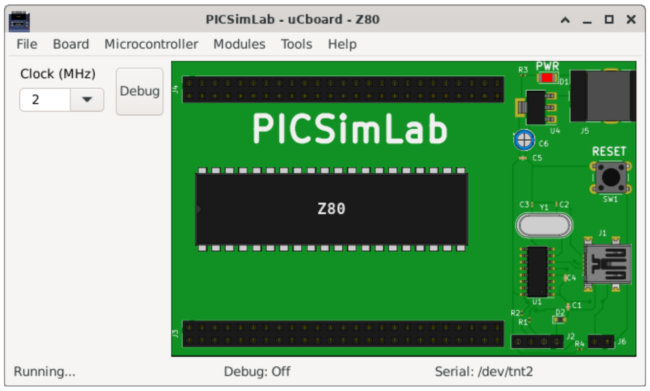


Running... Debug: Off Serial: /dev/tnt2 (ERROR)

Examples

4.2 uCboard

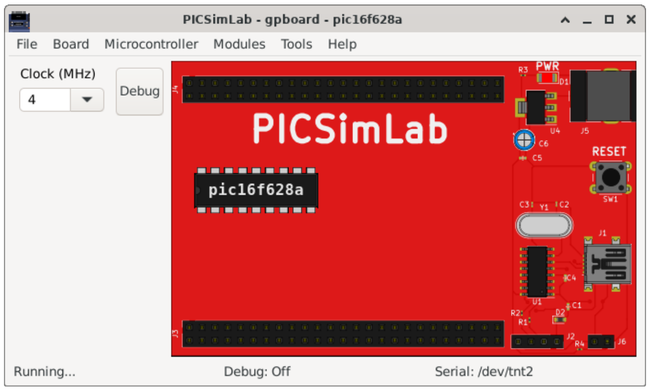
It is a generic board only with reset, serial and crystal circuits and support to multiple microcontrollers (initially C51, Z80 and STM8S103)of uCsim.



Examples

4.3 gpboard

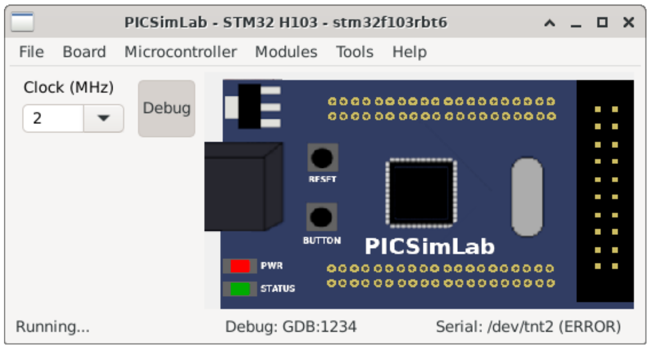
It is a generic board only with reset, serial and crystal circuits and support to multiple microcontrollers of gpsim.



Examples

4.4 STM32 H103

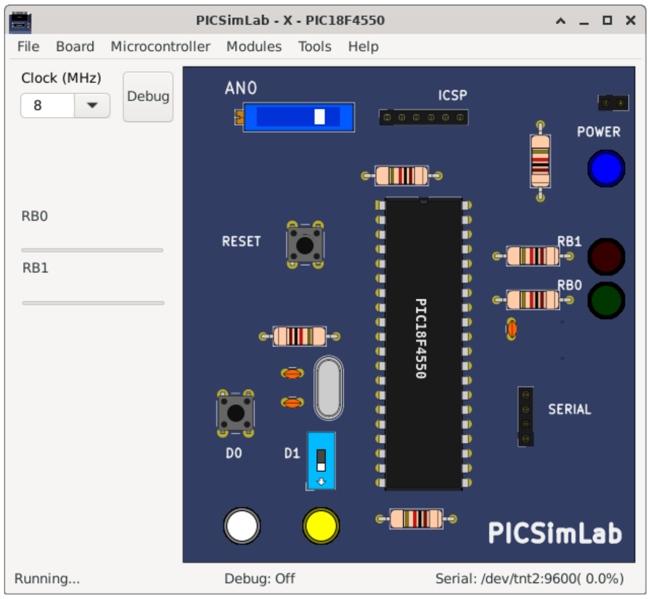
It is a generic board only with reset, one push button, serial and crystal circuits and support to stm32f103rbt6 microcontroller of qemu-stm32.


PICSimLab - STM32 H103 - stm32f103rbt6
File Board Microcontroller Modules Tools Help
Clock (MHz)
2
Debug
RESET
BUTTON
PWR
STATUS
PICSimLab
Running...
Debug: GDB:1234
Serial: /dev/tnt2 (ERROR)

Examples

4.5 X

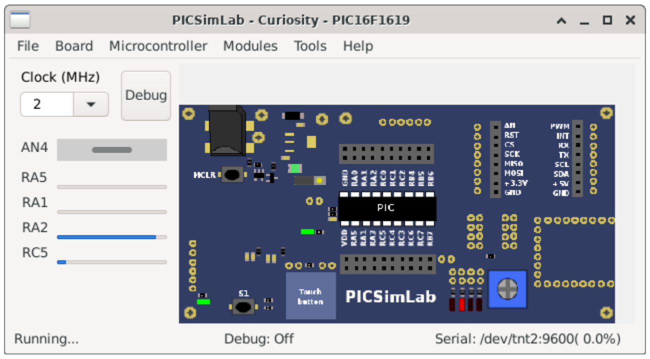
It is a generic board, used as example in How to Compile PICsimLab and Create New Boards.


PICSimLab - X - PIC18F4550
File Board Microcontroller Modules Tools Help
Clock (MHz)
8
Debug
RB0
RB1
AN0
ICSP
POWER
RESET
RB1
RB0
PIC18F4550
SERIAL
D0
D1
PICSimLab
Running...
Debug: Off
Serial: /dev/tnt2:9600(0.0%)

Examples

4.6 Curiosity

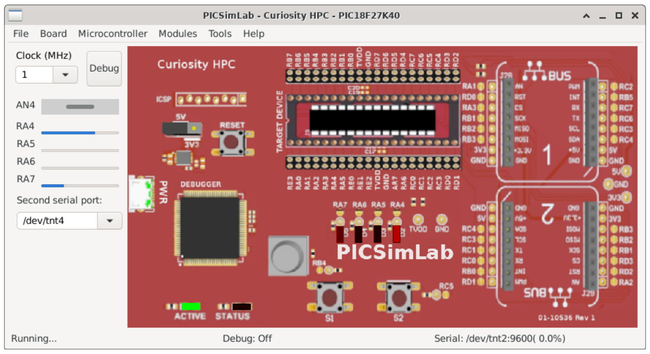
This is a simple PIC microcontroller development board that uses picsim.



Examples

4.7 Curiosity HPC

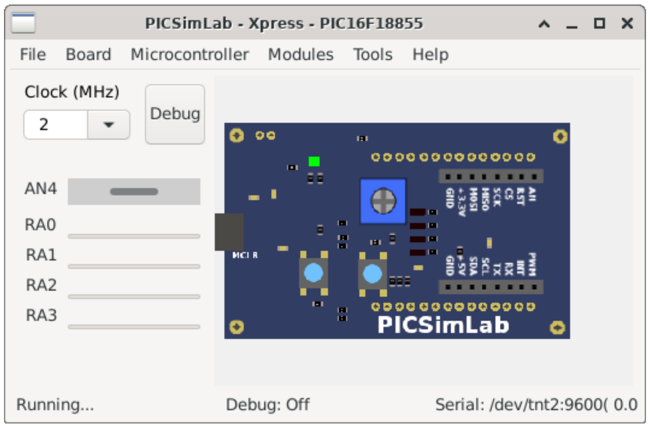
This is a simple PIC microcontroller development board that uses picsim.



Examples

4.8 Xpress

This is a simple PIC microcontroller development board that uses picsim.


PICSimLab - Xpress - PIC16F18855
File
Board
Microcontroller
Modules
Tools
Help
Clock (MHz)
2
Debug
AN4
RA0
RA1
RA2
RA3
PICSimLab
Running...
Debug: Off
Serial: /dev/tnt2:9600(0.0

Examples

Chapter 5

Serial Communication

To use the simulator serial port, install a NULL-MODEM emulator:

- Windows: com0com http://sourceforge.net/projects/com0com/
- Linux: tty0tty https://github.com/lcgamboa/tty0tty

For communication the PICSimLab should be connected in one port of the NULL-MODEM emulator and the other application connected in the other port. Configuration examples linking PICSimLab to Cutecom for serial communication:

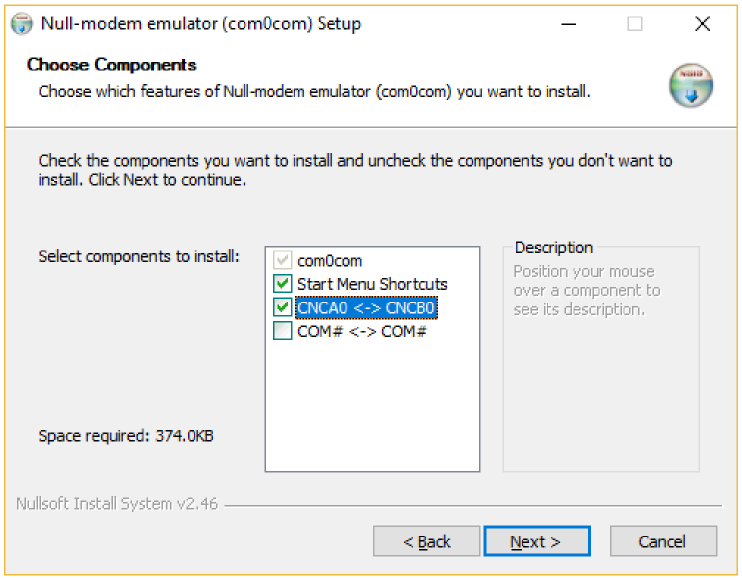
OS	PicsimLab port	Cutecom port	NULL-Modem prog.	Connection
Windows	com1	com2	com0com	com1<=>com2
Linux	/dev/tnt2	/dev/tnt3	tty0tty	/dev/tnt2<=>/dev/tnt3

5.1 Com0com Installation and Configuration(Windows)

Download the signed version of com0com.

Unzip the downloaded .zip file and run the specific installer of your operating system, x86 for windows 32-bit or x64 for windows 64-bit.

Configure the “choose components” window as the figure below:


Null-modem emulator (com0com) Setup

Choose Components

Choose which features of Null-modem emulator (com0com) you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

- [✓] com0com
- [✓] Start Menu Shortcuts
- [✓] CNCA0 <-> CNCB0
- [ ] COM# <-> COM#

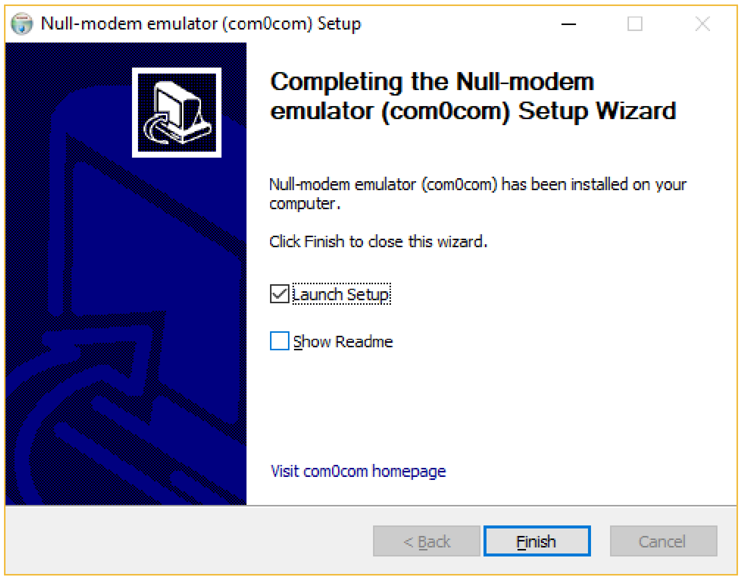
Description

Position your mouse over a component to see its description.

Space required: 374.0KB

Nullsoft Install System v2.46

In the last configuration window, check the "Launch setup" option:


Null-modem emulator (com0com) Setup

Completing the Null-modem emulator (com0com) Setup Wizard

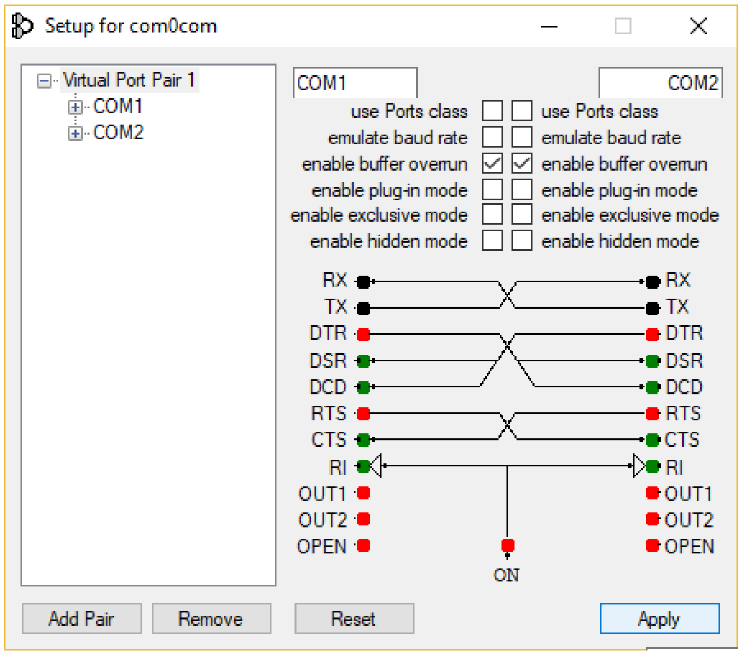
Null-modem emulator (com0com) has been installed on your computer.

Click Finish to close this wizard.

- [✓] Launch Setup
- [ ] Show Readme

Visit com0com homepage

In the setup window, change the port names to COM1, COM2, COM3 .... Just check the "enable buffer overrun" option on the two ports, click in the "Apply" button and close the setup. In the configuration shown in the figure below, the COM1 and COM2 ports form a NULL-MODEM connection, where one port must be used by the PICSimLab and another by the application with serial communication.



Setup for com0com

Virtual Port Pair 1
COM1
COM2

COM1			COM2
use Ports class	☐	☐	use Ports class
emulate baud rate	☐	☐	emulate baud rate
enable buffer overrun	☑	☑	enable buffer overrun
enable plug-in mode	☐	☐	enable plug-in mode
enable exclusive mode	☐	☐	enable exclusive mode
enable hidden mode	☐	☐	enable hidden mode

5.2 tty0tty Installation and Configuration (Linux)

Download the href https://github.com/lcgamboa/tty0tty/archive/master.zip tty0tyy. Unzip the downloaded folder.

Open a terminal and enter in the tty0tty/module/ folder and enter the following commands:

```
sudo apt-get update
sudo apt-get -y upgrade
sudo apt-get -y install gcc make linux-headers-`uname -r`
make
sudo make install
```

The user must be in the **dialout** group to access the ports. To add your user to **dialout** group use the command:

```
sudo usermod -a -G dialout your_user_name
```

after this is necessary logout and login to group permissions take effect.

Once installed, the module creates 8 interconnected ports as follows:

```
/dev/tnt0  <=>  /dev/tnt1
/dev/tnt2  <=>  /dev/tnt3
/dev/tnt4  <=>  /dev/tnt5
/dev/tnt6  <=>  /dev/tnt7
```

the connection between each pair is of the form:

```
TX   ->  RX
RX   <-  TX
RTS  ->  CTS
CTS  <-  RTS
DSR  <-  DTR
CD   <-  DTR
DTR  ->  DSR
DTR  ->  CD
```

Any pair of ports form a NULL-MODEM connection, where one port must be used by the PICSimLab and another by the application with serial communication.

Chapter 6

Programmer and Debugger Support

The type of debug interface depends on the backend simulator utilized.

6.1 MPLABX Integrated Debug (picsim and simavr)

To use the MPLABX IDE for debug and program the PicsimLab, install the plugin com-picsim-picsimlab.nbm in MPLABX.

The plugin connect to Picsimlab through a TCP socket using port 1234 (or other defined in configuration window), and you have to allow the access in the firewall.

Tutorial: how to use MPLABX to program and debug PICsimLab.

It's possible import and debug a Arduino sketch into MPLABX using the Arduino import plugin.

6.2 Arduino IDE Integration (simavr)

For integrated use with the Arduino IDE, simply configure the serial port as explained in the section 5 and load the Arduino bootloader. The bootloader can be loaded from the "Tools->Arduino bootloader" menu.

In Windows, considering com0com making a NULL-MODEM connection between COM1 and COM2, simply connect the PICSimLab on the COM1 port (defined in configuration window) and the Arduino IDE on the COM2 port or vice versa.

On Linux the operation is the same, but using for example the ports /dev/tnt2 and /dev/tnt3.

In Linux for the virtual ports to be detected in Arduino it is necessary to replace the library lib/liblistSerialsj.so of the Arduino with a version which support the detection of tty0tty ports, that can be downloaded in the link listSerialC with tty0tty support.

6.3 avr-gdb Debug (simavr)

With debug support enabled you can use avr-gdb to debug the code used in the simulator. Use the configuration window to choose between MDB (MPLABX) or GDB to debug AVR microcontrollers.

Use avr-gdb with the .elf file as the parameter:

```
avr-gdb compiled_file.elf
```

and the command below to connect (1234 is the default port):

```
target remote localhost:1234
```

Graphic debug mode can be made using eclipse IDE with Sloeber Arduino plugin.

6.4 arm-gdb Debug (qemu-stm32)

With debug support enabled you can use arm-none-eabi-gdb (or gdb-multiarch) to debug the code used in the simulator.

Use arm-none-eabi-gdb with the .elf file as the parameter:

```
arm-none-eabi-gdb compiled_file.elf
```

and the command below to connect (1234 is the default port):

```
target remote localhost:1234
```

Graphic debug mode can be made using eclipse IDE with Eclipse Embedded CDT.

6.5 uCsim Debug

The uCsim debug console can be accessed with the telnet (1234 is the default port):

```
telnet localhost 1234
```

All uCsim commands are supported.

For windows users putty telnet client is a good option to access the uCsim console.

Chapter 7

Tools

7.1 Serial Terminal

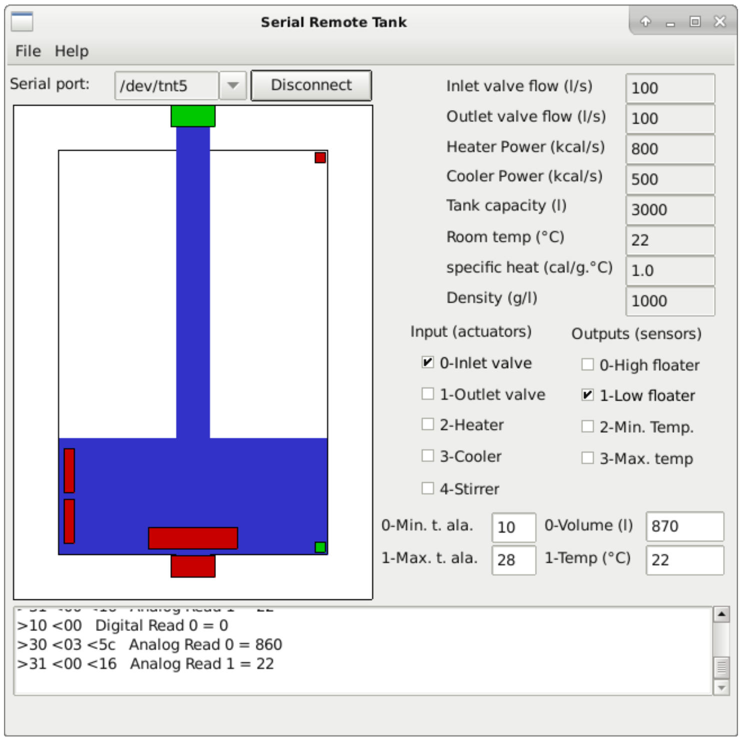
To use this tool with PICSimLab you first need to configure a virtual serial port as described in Chapter: Serial Communication. It is possible to use this tool with a real serial port connected to a real device.

Open the serial terminal Cutecom

7.2 Serial Remote Tank

To use this tool with PICSimLab you first need to configure a virtual serial port as described in Chapter: Serial Communication. It is possible to use this tool with a real serial port connected to a real device.

Serial Remote Tank



Serial port: /dev/tnt5

Inlet valve flow (l/s) 100
Outlet valve flow (l/s) 100
Heater Power (kcal/s) 800
Cooler Power (kcal/s) 500
Tank capacity (l) 3000
Room temp (°C) 22
specific heat (cal/g.°C) 1.0
Density (g/l) 1000

Input (actuators)	Outputs (sensors)
✓ 0-Inlet valve	☐ 0-High floater
☐ 1-Outlet valve	✓ 1-Low floater
☐ 2-Heater	☐ 2-Min. Temp.
☐ 3-Cooler	☐ 3-Max. temp
☐ 4-Stirrer	

0-Min. t. ala.	10	0-Volume (l)	870
1-Max. t. ala.	28	1-Temp (°C)	22

```
>10 <00  Digital Read 0 = 0
>30 <03 <5c  Analog Read 0 = 860
>31 <00 <16  Analog Read 1 = 22
```

7.2.1 Sensors and Actuators

Actuators

Digital inputs

1. Inlet valve
2. Outlet valve
3. Heater
4. Cooler
5. Stirrer

Analog inputs

1. Minimal temperature alarm trigger level
2. Maximal temperature alarm trigger level

Sensors

Digital outputs

1. High floater
2. Low floater
3. Minimal temperature
4. Maximal temperature

Analog outputs

1. Volume
2. Temperature

7.2.2 Communication Protocol

Writing on Digital Input

Sent one byte in 0x0N hexadecimal format where N is the number of input followed by a second byte with value 0x00 for disable or 0x01 for enable.

Example to turn on the input 2:

```
Serial_write(0x02);
Serial_write(0x01);
```

Reading Digital Output

Sent one byte in 0x1N hexadecimal format where N is the number of output and read one byte. The byte readed have value 0x00 for disable or 0x01 for enable.

Example to read output 3:

```
Serial_write(0x13);
valor=Serial_read(0);
```

Writing on Analog Input

Sent one byte in 0x2N hexadecimal format where N is the number of input followed by two bytes with the 16 bits value.

Example to write the value 230 on analog input 1:

```
Serial_write(0x21);
valor=230;
Serial_write((valor&0xFF00)>>8);
Serial_write(valor&0x00FF);
```

Reading Analog Output

Sent one byte in 0x3N hexadecimal format where N is the number of output and read two bytes to form the 16 bits value.

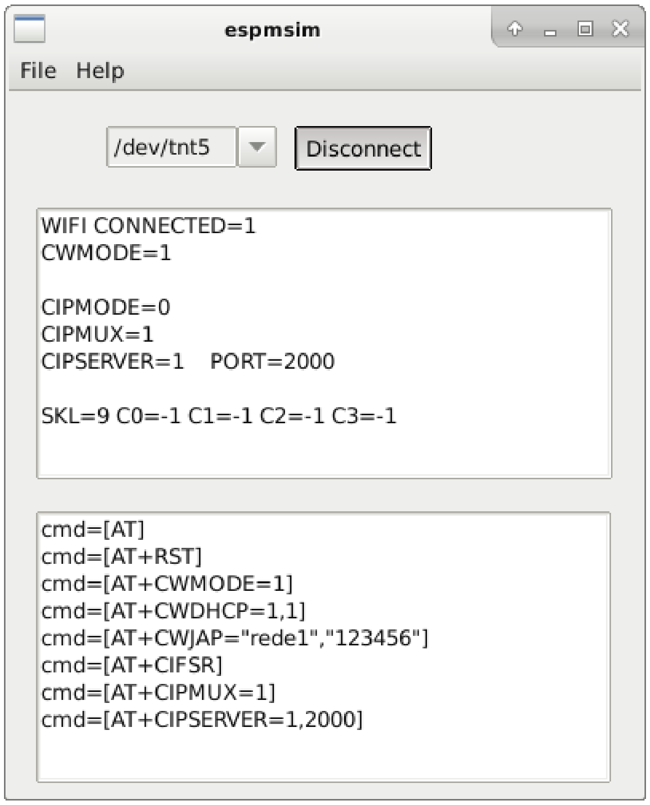
Example to read analog output 2:

```
Serial_write(0x32);
valorh=Serial_read(0);
valorl=Serial_read(0);
valor=(valorh<<8)|valorl;
```

7.3 Esp8266 Modem Simulator

To use this tool with PICSimLab you first need to configure a virtual serial port as described in Chapter: Serial Communication. It is possible to use this tool with a real serial port connected to a real device.

ESP8266 Modem Simulator



```
WIFI CONNECTED=1
CWMODE=1

CIPMODE=0
CIPMUX=1
CIPSERVER=1   PORT=2000

SKL=9 C0=-1 C1=-1 C2=-1 C3=-1
```

```
cmd=[AT]
cmd=[AT+RST]
cmd=[AT+CWMODE=1]
cmd=[AT+CWDHCP=1,1]
cmd=[AT+CWJAP="rede1","123456"]
cmd=[AT+CIFSR]
cmd=[AT+CIPMUX=1]
cmd=[AT+CIPSERVER=1,2000]
```

7.3.1 Supported Commands

- AT
- AT+RST
- AT+GMR
- AT+CWMODE=1
- AT+CWDHCP=1,1
- AT+CWLAP
- AT+CWJAP="rede1","123456"
- AT+CIFSR
- AT+CIPMUX=1
- AT+CIPSERVER=1,2000
- AT+CIPSEND=0,10
- AT+CIPCLOSE=0

7.4 Arduino Bootloader

To use this tool with PICSimLab you first need to configure a virtual serial port as described in Chapter: Serial Communication.

Load microcontroller with Arduino serial bootloader

7.5 MPLABX Debugger Plugin

- Open the web page to download the plugin

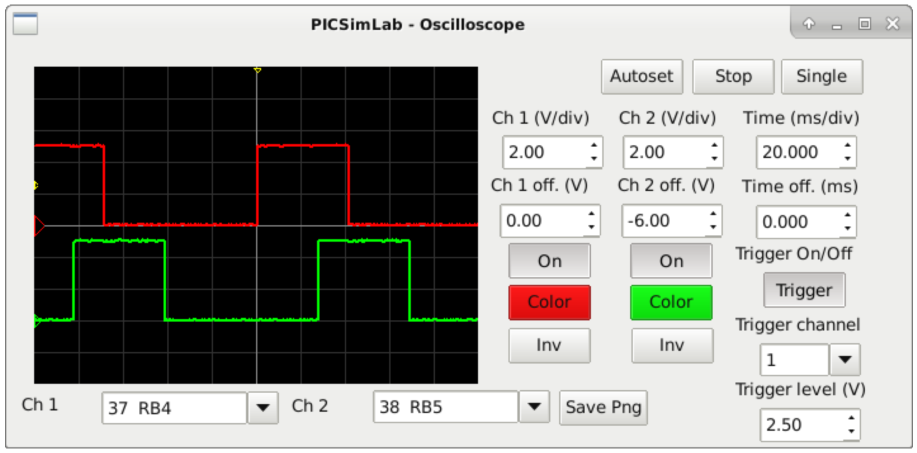
7.6 Pin Viewer

Open the Pin status viewer program

Chapter 8

Oscilloscope

The PICSimLab has a basic two-channel oscilloscope that can be used to view the signal on any pin of the microcontroller. The oscilloscope can be accessed through the “Modules->Oscilloscope” menu.



Ch 1 (V/div) 2.00
Ch 2 (V/div) 2.00
Time (ms/div) 20.000
Ch 1 off. (V) 0.00
Ch 2 off. (V) -6.00
Time off. (ms) 0.000
Trigger channel 1
Trigger level (V) 2.50
Ch 1 37 RB4
Ch 2 38 RB5

Chapter 9

Spare Parts

The PICSimLab has a window that allows the connection of spare parts to the microcontroller, it can be accessed through the menu " Modules-> Spare parts ".

The main window has the menu with the following functions:

- File
  - New configuration - Clear the spare parts window
  - Save configuration - Saves the current settings of the spare parts into .pcf file
  - Load configuration - Loads the settings from .pcf file
  - Save pin alias - Saves the current pin alias to .ppa text file
  - Load pin alias - Loads the pin alias from .ppa file
- Edit
  - Clear pin alias - Clear the pin alias
  - Toggle pin alias - Enable/Disable pin alias use
  - Edit pin alias - Open current pin alias .ppa file in text editor
  - Reload pin alias - Reload the current .ppa pin alias file (need after edit .ppa file)
  - Zoom in - Increase draw scale
  - Zoom out - Decrease draw scale
- Inputs
  - Encoder - Adds a rotary quadrature encoder with push button
  - Gamepad - Adds a gamepad
  - Gamepad (Analogic) - Adds a gamepad with one analogic output
  - Keypad - Adds one matrix keypad

  - MPU6050 - Adds a accelerometer and gyroscope (only raw values)
  - Potentiometers - Adds 4 potentiometers
  - Potentiometers (Rotary) - Adds 4 rotary potentiometers
  - Push Buttons - Adds 8 push buttons
  - Push Buttons (Analogic) - Adds 8 push buttons with analog output
  - Switchs - Adds eight switchs
  - Ultrasonic HC-SR04 - Adds a ultrasonic range sensor

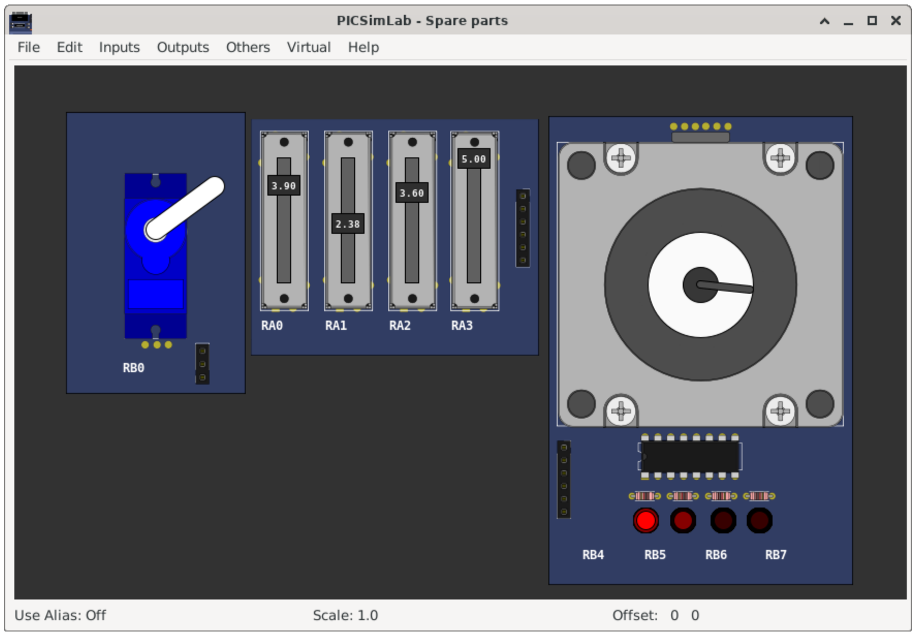
- Outputs
  - 7 Segments Display - Adds four multiplexed 7 segments displays
  - 7 Segments Display (w/dec) - Adds four multiplexed 7 segments displays with decoder
  - Buzzer - Adds a active/passive buzzer
  - DC Motor - Adds a DC motor with H-bridge and quadrature encoder
  - LCD hd44780 - Adds a text display hd44780
  - LCD ili9340 - Adds a color graphic display ili9340 with touchscreen
  - LCD pcd8544 - Adds a monochrome graphic display pcd8544 (Nokia 5110)
  - LCD pcf8833 - Adds a color graphic display pcf8833
  - LCD ssd1306 - Adds a monochrome graphic display ssd1306
  - LED Matrix - Adds a 8x8 LED matrix with MAX72xx controller
  - LEDs - Adds 8 red LEDs
  - RGB LED - Adds one RGB LED
  - Servo Motor - Adds a servo motor
  - Step Motor - Adds a step motor

- Others
  - ETH w5500 - Adds a ethernet shield w5500
  - IO 74xx595 - Adds a 74xx595 SIPO 8 bit shift register
  - IO MCP23S17 - Adds a MCP23S17 serial SPI IO expander
  - IO PCF8574 - Adds a PCF8574 serial I2C IO expander
  - IO UART - Adds a UART serial port
  - Jumper Wires - Adds sixteen jumper wires
  - MEM 24CXXX - Adds a 24CXXX serial I2C EEPROM memory
  - RTC ds1307 - Adds a ds1307 real time clock
  - RTC pfc8563 - Adds a pfc8563 real time clock
  - SD Card - Adds a SD card shield

- Temperature System - Adds a temperature control system

- Virtual
  - D. Transfer Function - Adds a discrete transfer function mathematical model
  - IO Virtual term - Adds a virtual serial terminal
  - Signal Generator - Adds a virtual signal generator
  - VCD Dump - Adds a digital value file dump recorder
  - VCD Dump (Analogic) - Adds a analog value file dump recorder
  - VCD Play - Adds a digital value file dump player

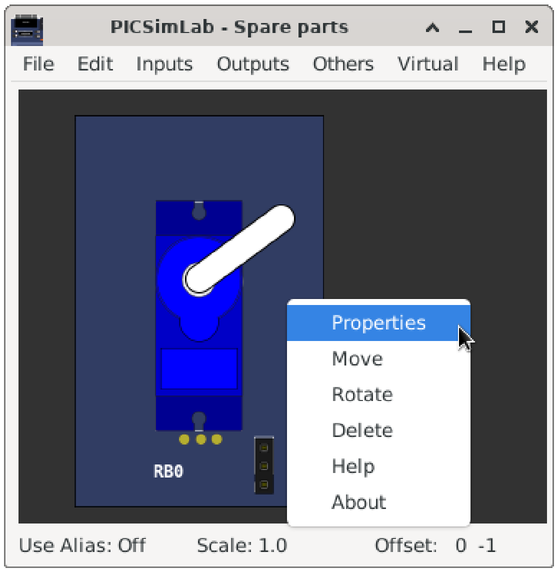
- Help
  - Contents - Open Help window
  - About - Show message about author and version



After adding the part, with a right click of the mouse you can access the options menu of the part with the options:

- Properties - Opens the connection settings window
- Move - Unlocks the part to move
- Rotate - Change the orientation of part

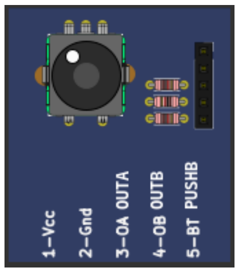
- Delete - Remove part
- Help - Open Help window of part
- About - Show message about author and version of part



9.1 Inputs

9.1.1 Encoder

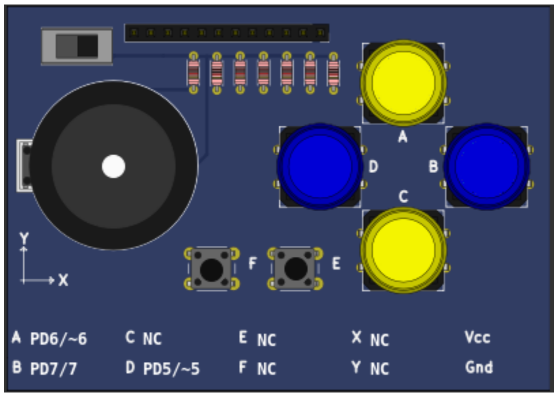
This part is a rotary quadrature encoder with push button. The output is twenty pulses per revolution.

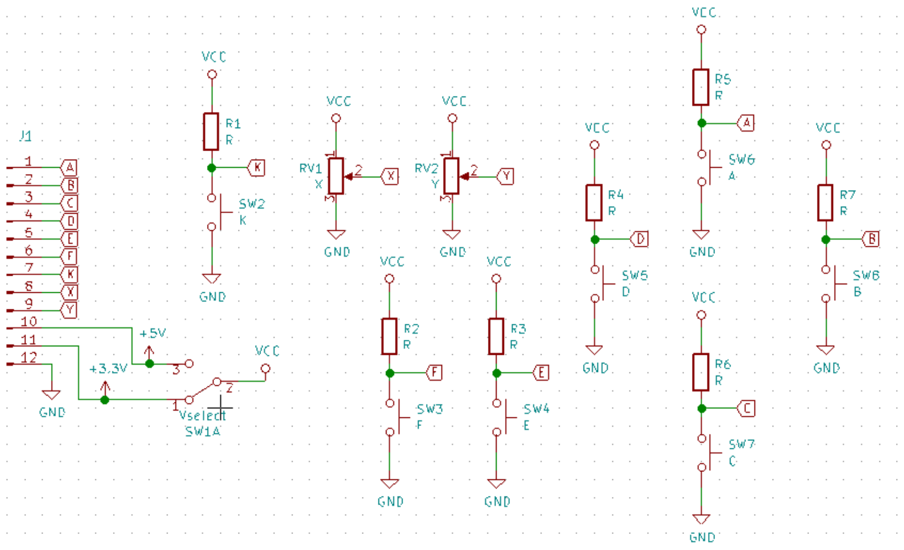


Examples

9.1.2 Gamepad

This part is a gamepad with two analog axis and 7 push buttons.





The gamepad can be controlled by keyboards keys:

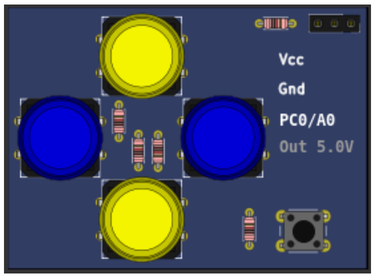
- X axis - keys ’A’ and ’D’
- Y axis - keys ’W’ and ’S’
- Button A - key ’I’
- Button B - key ’L’
- Button C - key ’K’

- Button D - key ’J’
- Button E - key ’E’
- Button F - key ’O’
- Button K - key ’R’

Examples

9.1.3 Gamepad Analogic

This part is a gamepad with 5 push buttons and one analogic output.



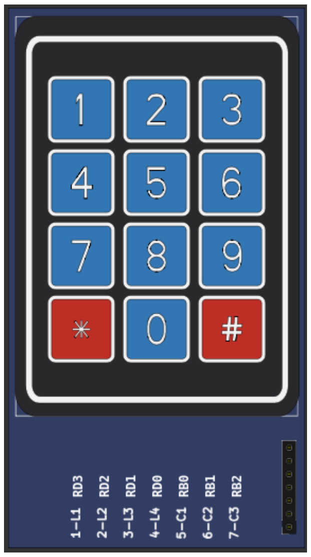
The gamepad can be controlled by keyboards keys:

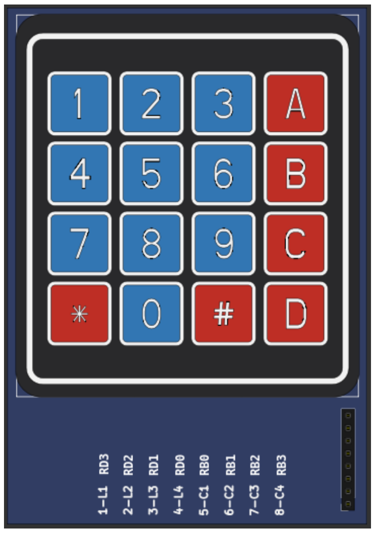
- Button A - key ’L’
- Button B - key ’I’
- Button C - key ’K’
- Button D - key ’J’
- Button E - key ’O’

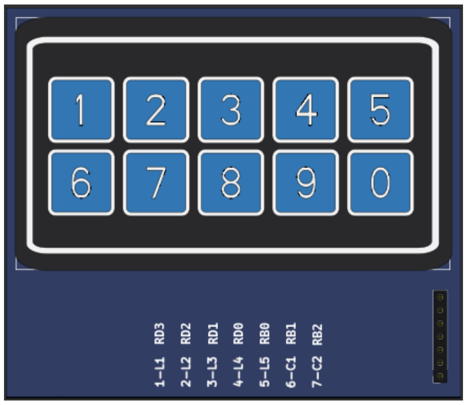
Examples

9.1.4 Keypad

It is a matrix keyboard configurable to 4x3 , 4x4 or 2x5 rows/columns.


1
2
3
4
5
6
7
8
9
*
0
#
1–L1 RD3
2–L2 RD2
3–L3 RD1
4–L4 RD0
5–C1 RB0
6–C2 RB1
7–C3 RB2

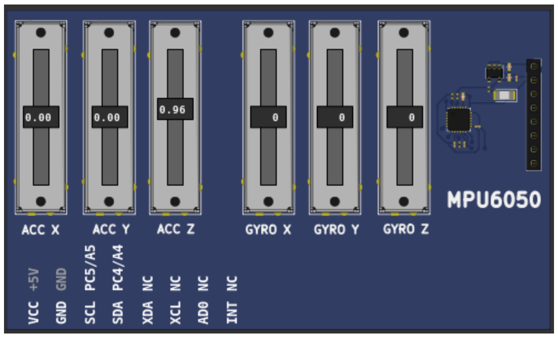

1
2
3
A
4
5
6
B
7
8
9
C
*
0
#
D
1–L1 RD3
2–L2 RD2
3–L3 RD1
4–L4 RD0
5–C1 RB0
6–C2 RB1
7–C3 RB2
8–C4 RB3



Examples

9.1.5 MPU6050

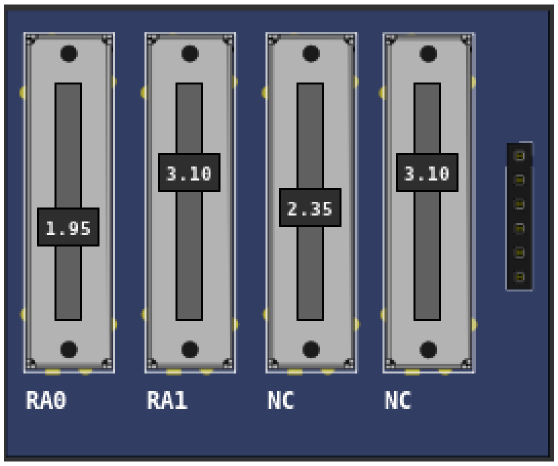
This part is MPU6050 accelerometer and gyroscope with I2C interface. Only raw values are available, DMP is not supported.

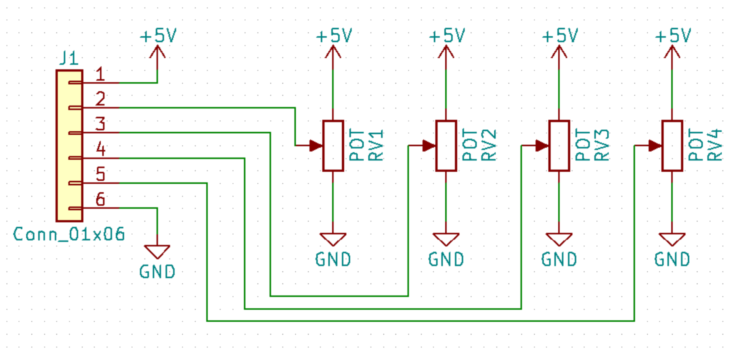


Examples

9.1.6 Potentiometers

This part is formed by 4 potentiometers connected between 0 and 5 volts, the output is connected to the cursor and varies within this voltage range.

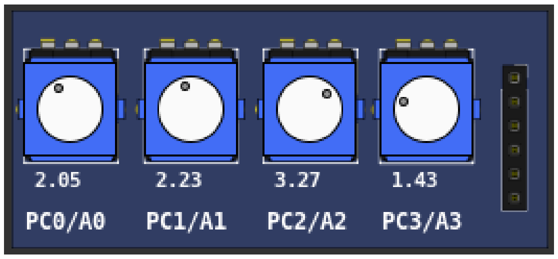


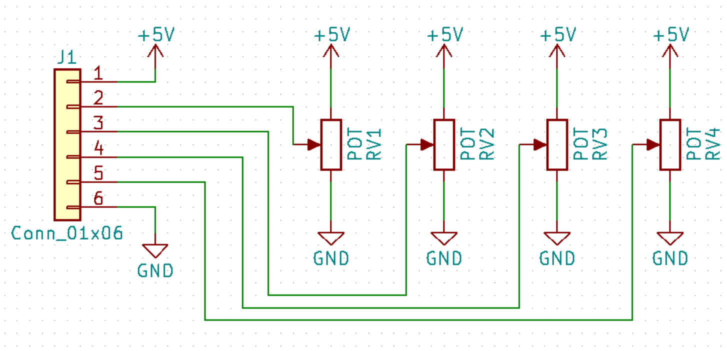


Examples

9.1.7 Potentiometers (Rotary)

This part is formed by 4 rotary potentiometers connected between 0 and 5 volts, the output is connected to the cursor and varies within this voltage range.

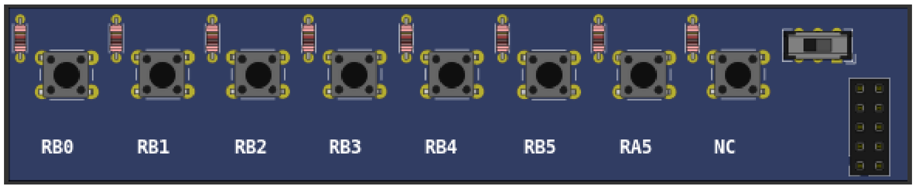


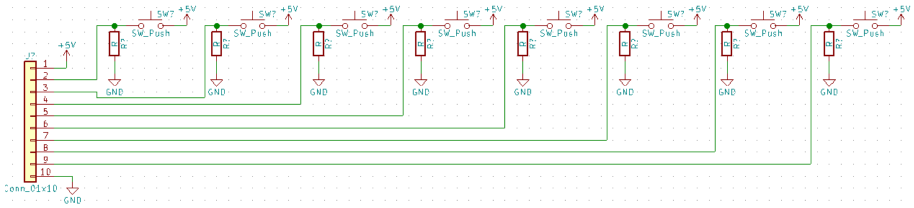


Examples

9.1.8   Push Buttons

This part consists of 8 push buttons. The output active state can be configurable.

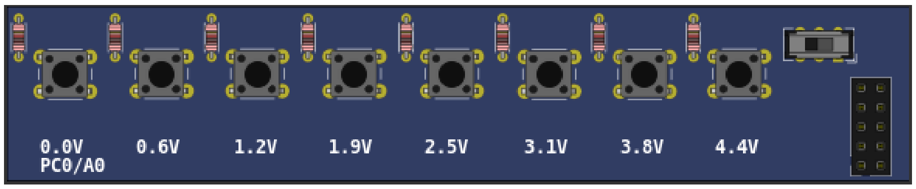




Examples

9.1.9   Push Buttons (Analogic)

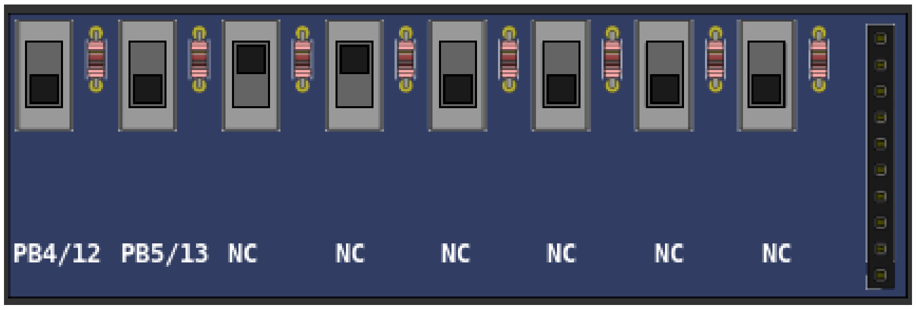
This part consists of 8 push buttons connected in a resistive ladder.

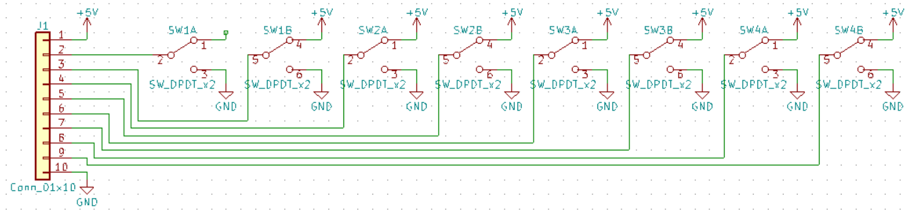


Examples

9.1.10 Switchs

This part consists of 8 keys with on or off position (0 or 1).

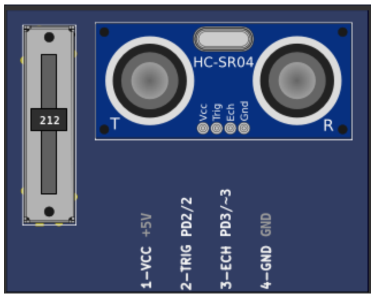




Examples

9.1.11 Ultrasonic HC-SR04

This part is ultrasonic range meter sensor.

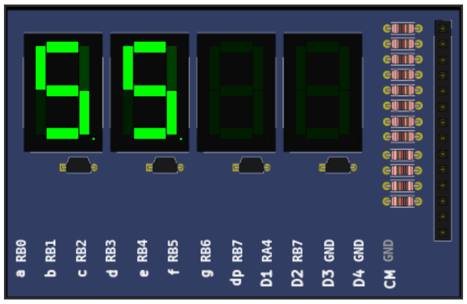


Examples

9.2 Outputs

9.2.1 7 Segments Display

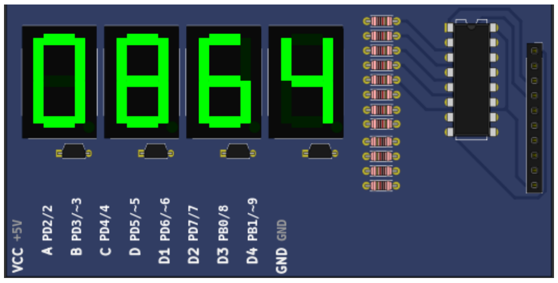
This is a four multiplexed 7 segments displays.

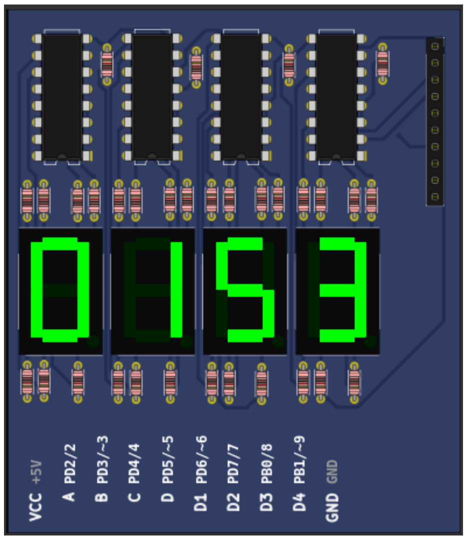


Examples

9.2.2 7 Segments Display (w/dec)

This is a four multiplexed 7 segments displays with BCD to 7 segments decoder (CD4511).

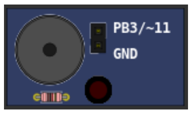




Examples

9.2.3 Buzzer

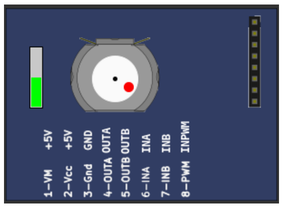
This is a active/passive buzzer.



Examples

9.2.4 DC Motor

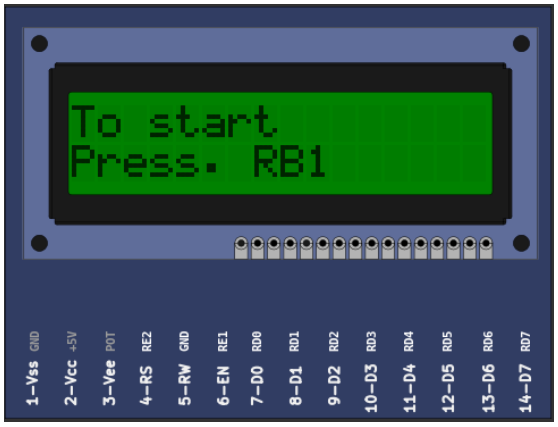
This part is DC motor with H-bridge driver and quadrature encoder.

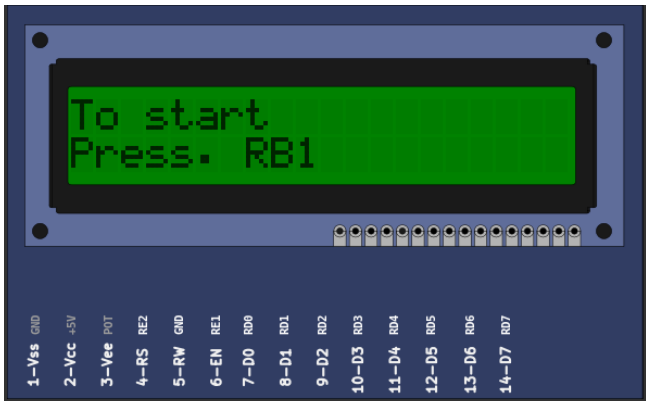


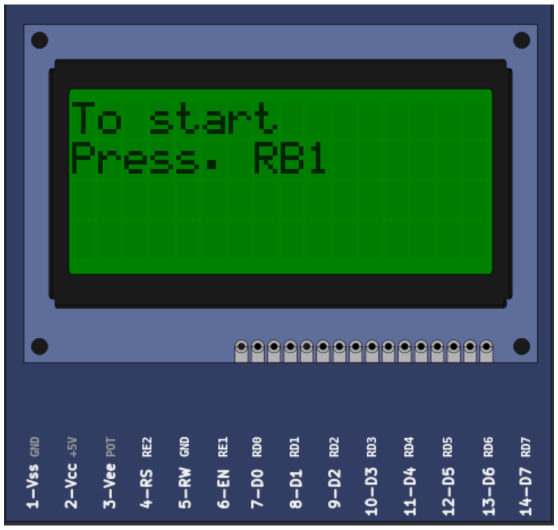
Examples

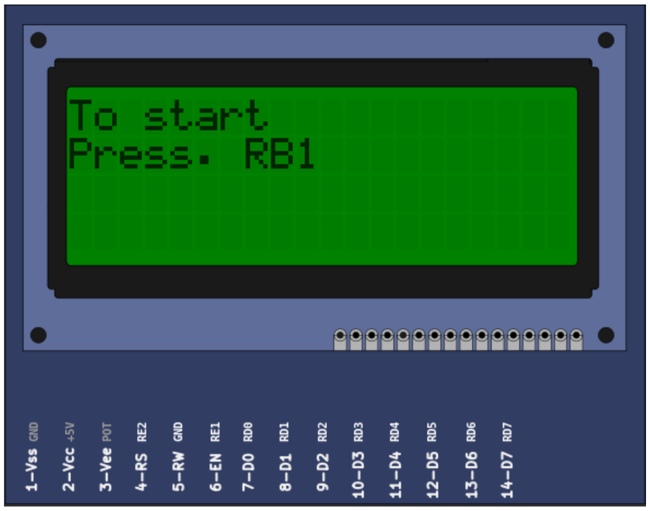
9.2.5 LCD hd44780

This part is a text display with 2 (or 4) lines by 16 (or 20) columns.





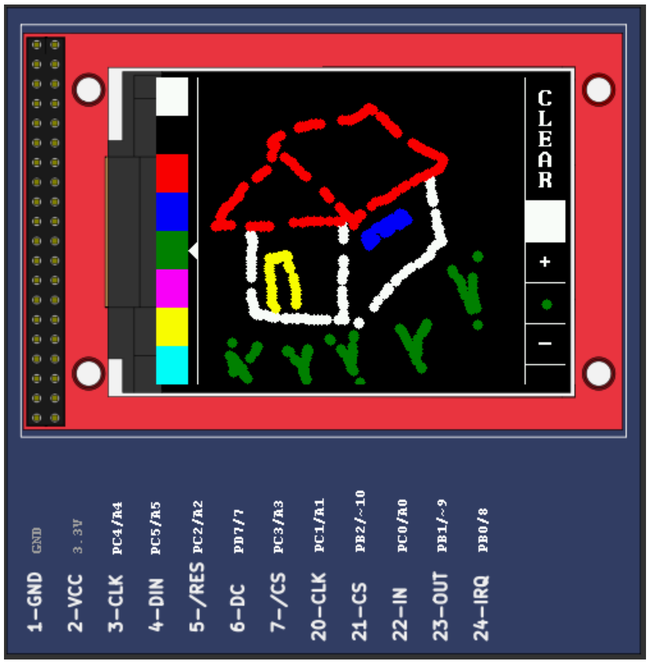




Examples

9.2.6 LCD ili9341

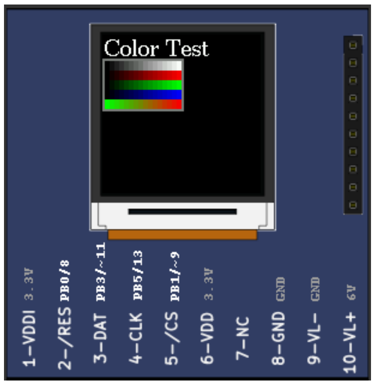
This part is a color graphic display with 240x320 pixels with touchscreen (xpt2046 controller). Only 4 SPI mode and 8 bits parallel mode is avaliable.



Examples

9.2.7 LCD pcf8833

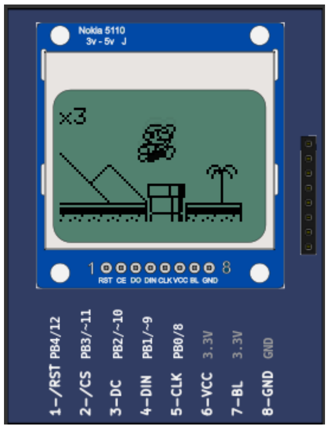
This part is a color graphic display with 132x132 pixels.



Examples

9.2.8 LCD pcd8544

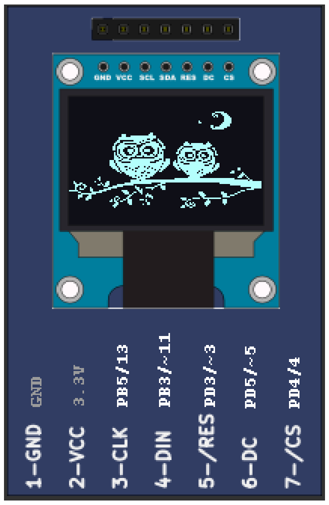
This part is a monochrome graphic display with 48x84 pixels. (Nokia 5110)



Examples

9.2.9 LCD ssd1306

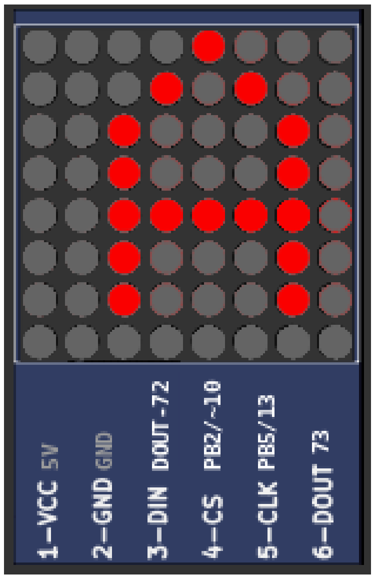
This part is a monochrome oled graphic display with 128x64 pixels. The part suport I2C and 4 SPI serial mode.



Examples

9.2.10 LED Matrix

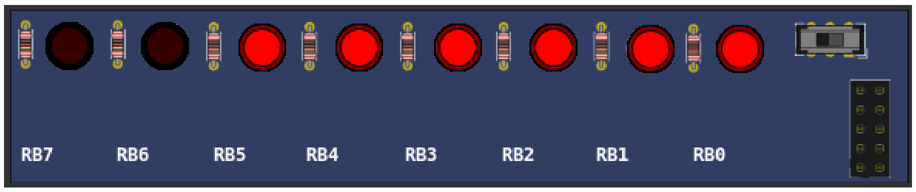
It is a 8x8 LED matrix with MAX72xx controller.

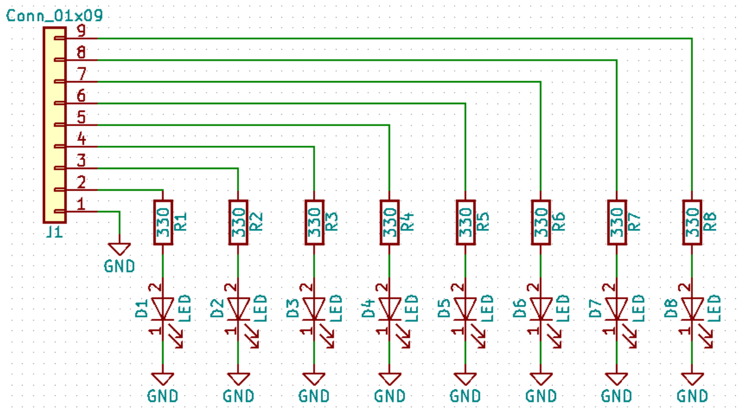


Examples

9.2.11 LEDs

This part is a bar of 8 independent red LEDs.

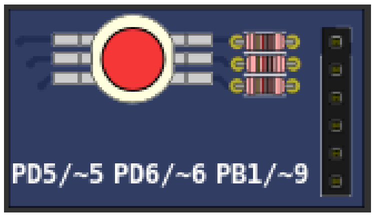


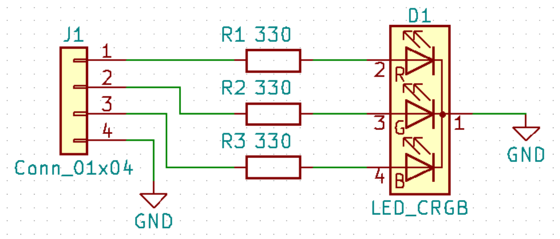


Examples

9.2.12 RGB LED

This part consists of a 4-pin RGB LED. Each color can be triggered independently. Using PWM it is possible to generate several colors by combining the 3 primary colors.

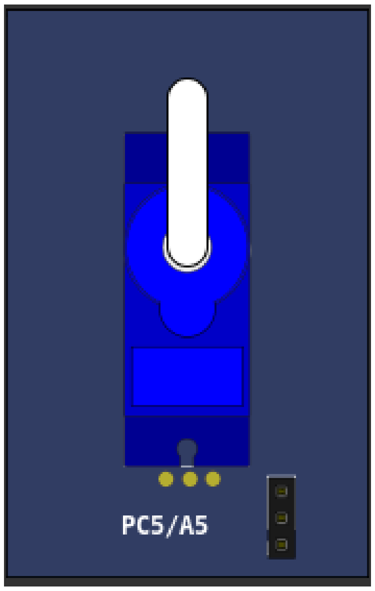


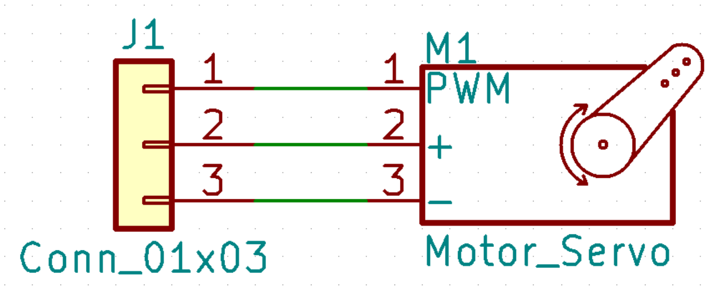


Examples

9.2.13 Servo Motor

The servo motor is a component that must be activated with a pulse of variable width from 1ms to 2ms every 20 ms. A pulse of 1ms positions the servo at -90º, one from 1.5ms to 0º and one from 2ms to 90º.

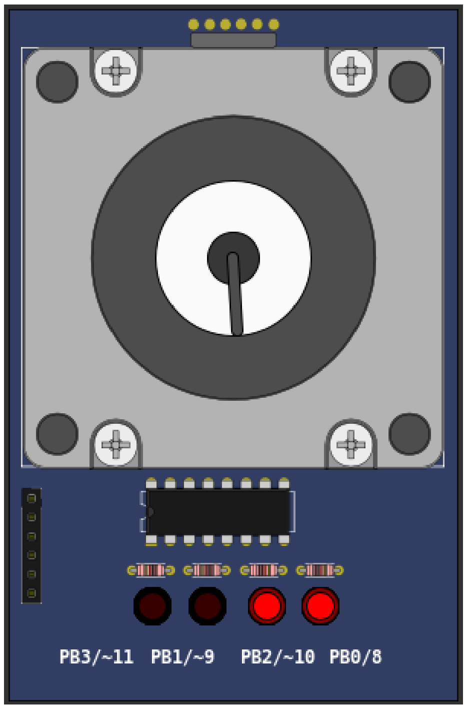


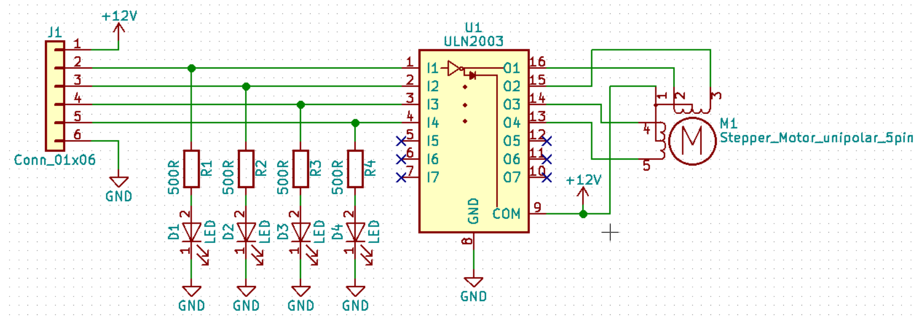


Examples

9.2.14 Step Motor

The stepper motor is a component with 4 coils that must be driven in the correct order to rotate the rotor. Each step of the motor is 1.8°.





Examples

9.3 Others

9.3.1 ETH w5500

This part is a ethernet shield w5500 with support to 8 sockets simultaneously.

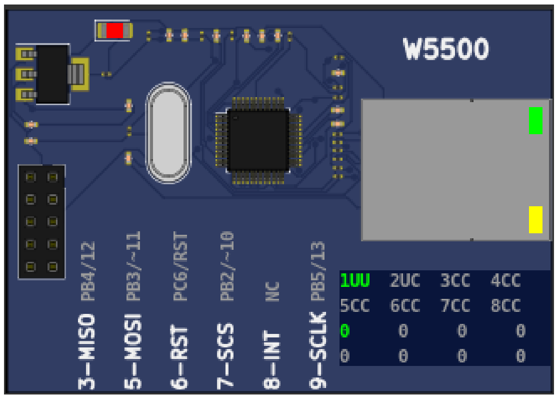
Only TCP/UDP unicast address sockets is supported. DHCP is emulated and return a fake ipv4 address.

All listening ports below 2000 are increased by 2000 to avoid operational system services ports. For example listening on port 80 becomes 2080.

w5500 Status Legend:

1º Letter - Type	2º Letter - Status	3º Letter - Error
C - Closed	C - Closed	B - Bind
T - TCP	I - Initialized	S - Send
U - UDP	L - Listen	R - Receive
M - MACRAW (don't supported)	S - Syn sent	L - Listen
	E - Established	U - Reuse
	W - Close wait	C - Connecting
	U - UDP	D - Shutdown
	M - MACRAW (don't supported)	

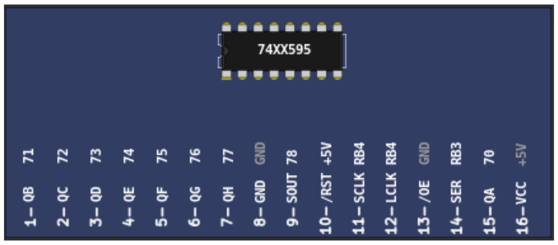
Click on connector to toggle link status.



Examples

9.3.2 IO 74xx595

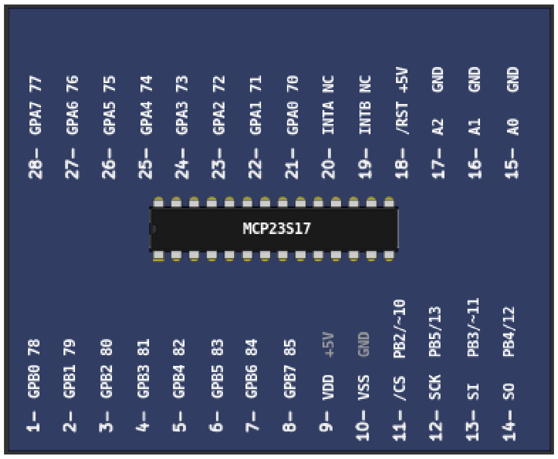
This is one 74xx595 serial input and parallel output 8 bit shift register.



Examples

9.3.3 IO MCP23S17

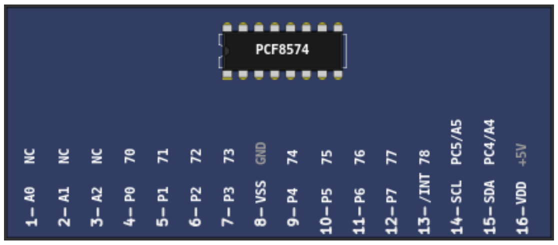
It is a MCP23S17 serial SPI IO expander part.



Examples

9.3.4 IO PCF8574

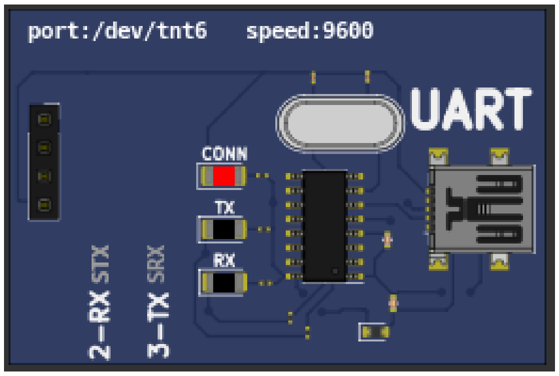
It is a PCF8574 serial I2C IO expander.



Examples

9.3.5 IO UART

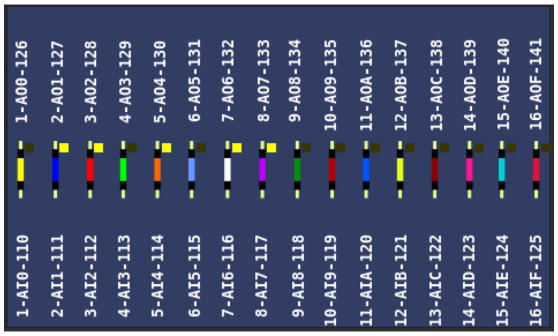
This part is a UART serial port. This part connects the hardware/software UART IO pins of microcontroller to one real/virtual PC serial port. To use virtual port is need to install a virtual port software, as described in 5.



Examples

9.3.6 Jumper Wires

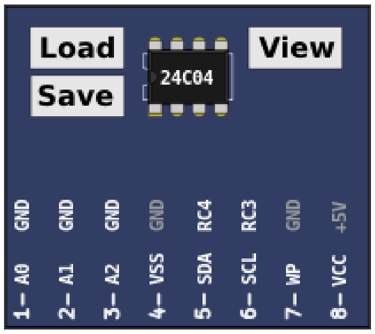
This part are formed by sixteen jumper wires. Each jumper has one input and one output. The jumper input must be connected to one pin output, the jumper output can be connected to multiple pin inputs. The jumper can be used to connect microcontroller pins or make connection between spare parts pins.



Examples

9.3.7 MEM 24CXXX

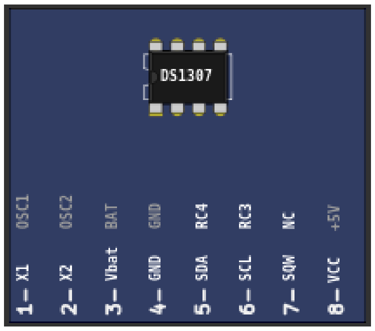
It is a 24CXXX serial I2C EEPROM part. There are support to the models 24C04 and 24C512.



Examples

9.3.8 RTC ds1307

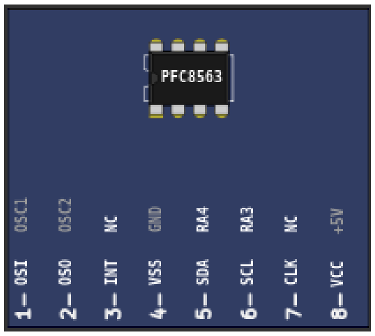
This part is a ds1307 real time clock with serial I2C interface.



Examples

9.3.9 RTC pfc8563

This part is a pfc8563 real time clock with serial I2C interface.



Examples

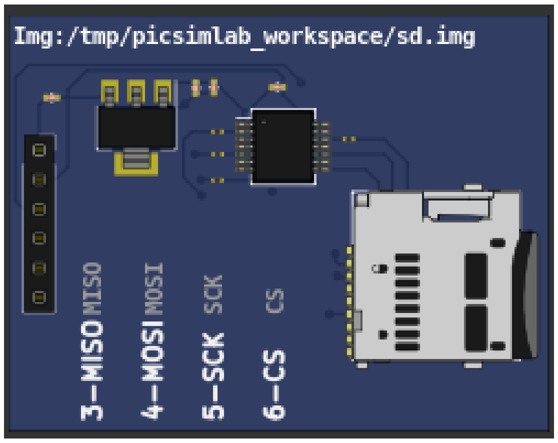
9.3.10 SD Card

This part is a SD Card shield. It's necessary set one sd card file image before use it. (Click on SD card connector to open file dialog)

On Linux one empty image can be created with this command:

```
dd if=/dev/zero of=sd.img bs=1M count=32
```

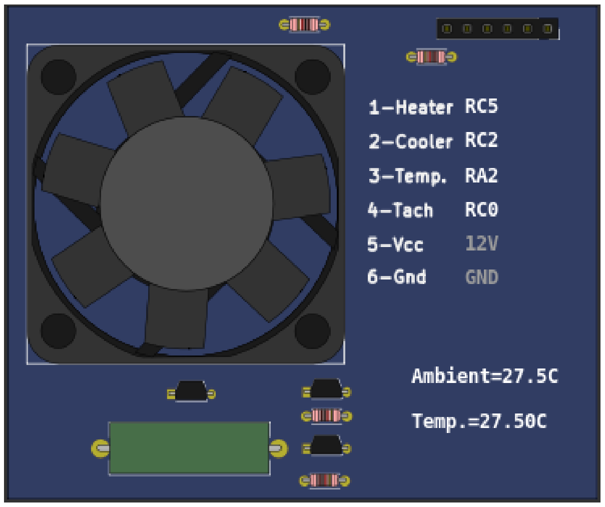
This empty image can be used with raw sd card access, to work with FAT file system the image need to be formatted before the use. (using SdFormatter.ino for example)



Examples

9.3.11 Temperature System

This part is a temperature control system. The temperature control system consists of a heating resistor, an LM35 temperature sensor, a cooler and an infrared tachometer.

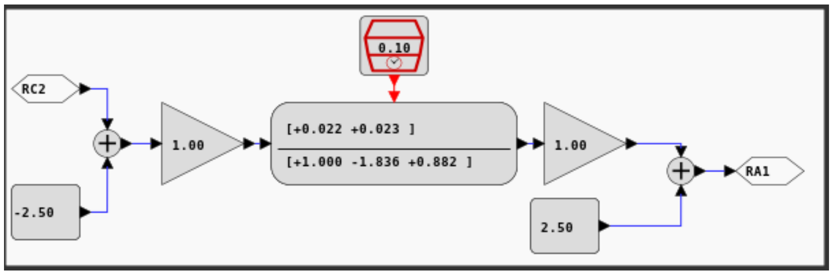


Examples

9.4 Virtual

9.4.1 D. Transfer Function

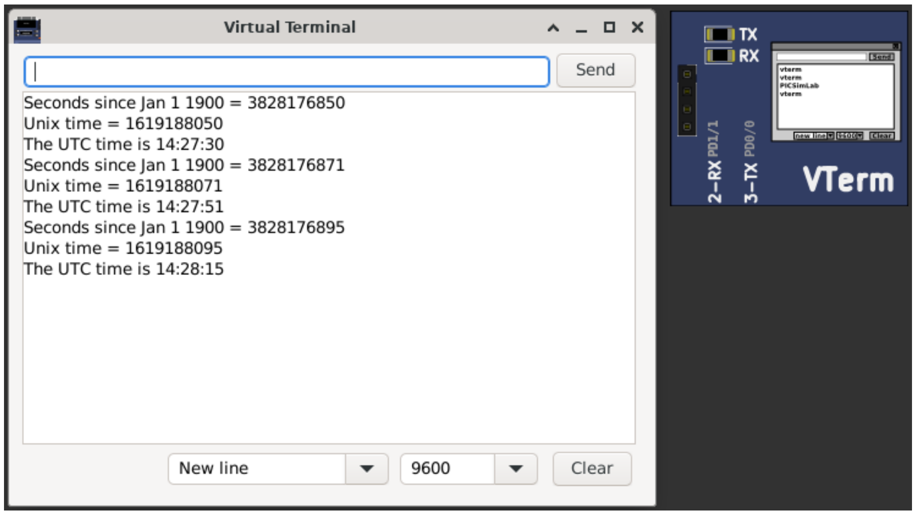
This is a discrete transfer function mathematical model.



Examples

9.4.2 IO Virtual term

This part is a virtual serial terminal. This part can be used to read and write RX/TX pins UART signals. This part don't need the use or install of virtual serial ports on computer. Clik on terminal picture to open the terminal window.

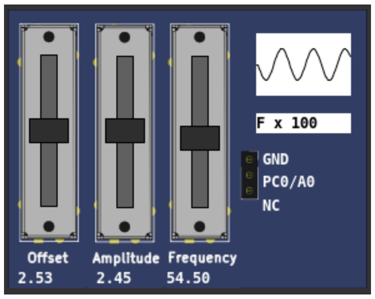


```
Seconds since Jan 1 1900 = 3828176850
Unix time = 1619188050
The UTC time is 14:27:30
Seconds since Jan 1 1900 = 3828176871
Unix time = 1619188071
The UTC time is 14:27:51
Seconds since Jan 1 1900 = 3828176895
Unix time = 1619188095
The UTC time is 14:28:15
```

Examples

9.4.3 Signal Generator

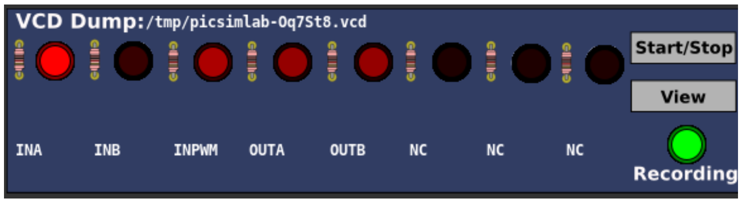
This part is a virtual signal generator with support for sine, square and triangular waves generation with amplitude and frequency adjustment.

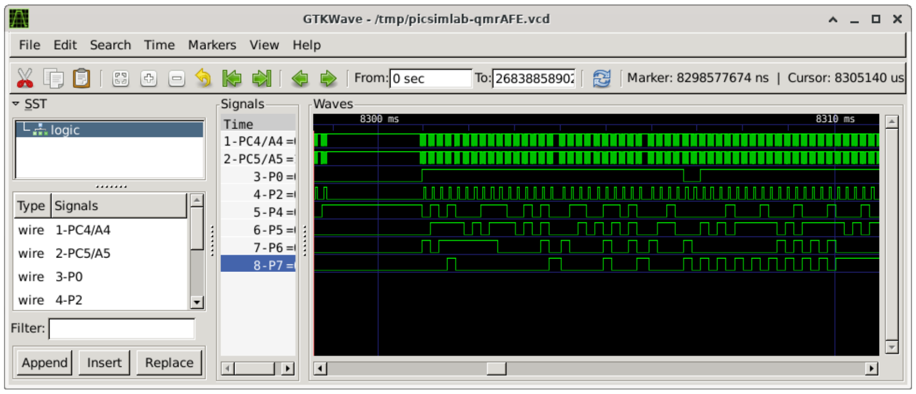


Examples

9.4.4 VCD dump

This part is a digital value file dump recorder. The file can be visualized with gtkwave.

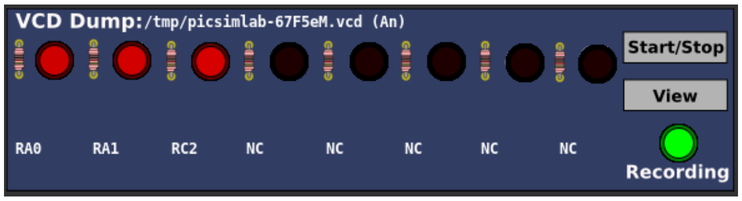


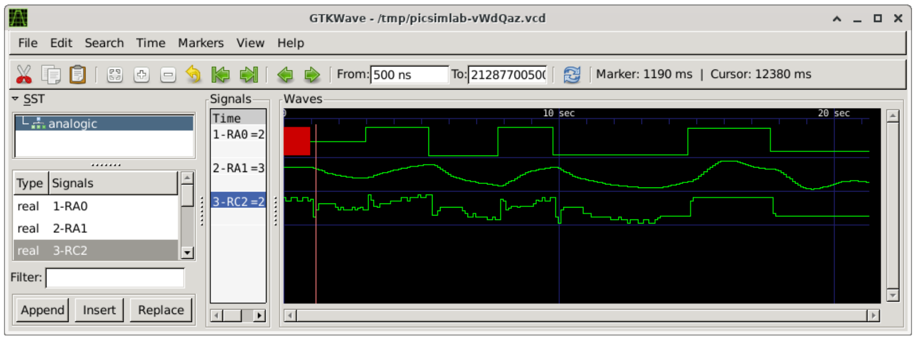


Examples

9.4.5 VCD dump (Analogic)

This part is a analog value file dump recorder. The file can be visualized with gtkwave.

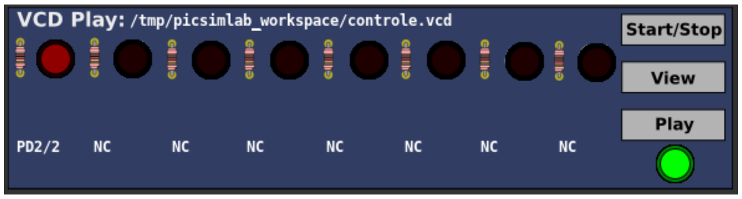



GTKWave - /tmp/picsimlab-vWdQaz.vcd
File Edit Search Time Markers View Help
From: 500 ns To: 21287700500
Marker: 1190 ms | Cursor: 12380 ms
SST
analogic
Type Signals
real 1-RA0
real 2-RA1
real 3-RC2
Filter:
Append Insert Replace
Signals
Time
1-RA0 =2
2-RA1 =3
3-RC2 =2
Waves
10 sec
20 sec

Examples

9.4.6 VCD Play

This part play a VCD file saved from VCD Dump part.


VCD Play: /tmp/picsimlab_workspace/controle.vcd
Start/Stop
View
Play
PD2/2
NC
NC
NC
NC
NC
NC
NC

Examples

Chapter 10

How To's

- How to use MPLABX to program and debug PICsimLab.
- How to Compile PICsimLab and Create New Boards.

Chapter 11

License