i2c-tiny-usb

- Main
  - Home
  - EMail
- AVR
  - i2c-tiny-usb
  - Ledmatrix
  - LCD2USB
  - Spi2cf
  - MacAVR
  - NanoVM
  - xu1541
  - BlueMP3
  - BtKit
- Linux
  - Twonky
  - BlueGPS
  - BlueZ
  - Lcd4Linux
- Misc
  - Atari projects
  - C64 projects
  - Tills Palm Pages
  - TheMaze
  - glMaze
  - StonX

Attach any I2C client chip (thermo sensors, AD converter, displays, relais driver, ...) to your PC via USB ... quick, easy and cheap! Drivers for Linux, Windows and MacOS available.

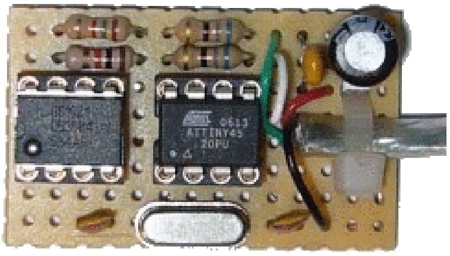
The i2c-tiny-usb project is an open source/open hardware project. The goal of i2c-tiny-usb is to provide a cheap generic i2c interface to be attached to the usb. It is meant as a replacement for those simple and cheap printer port to i2c adapters. A USB solution has several advantages incl. the built-in power supply and a more future proof interface. Furthermore no cpu intense bitbanging is required on the host side. This USB solution even requires less parts than some printer port solutions.

While the i2c-tiny-usb was developed under and for Linux it also works under Windows and MacOS X. A windows demo driver and demo application is included to get you started immediately.

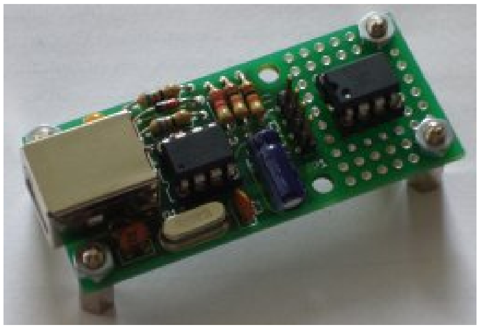
The i2c-tiny-usb project is based on:

- The Linux USB project and the Linux i2c implementation
- The windows driver and test application are based on the libusb-win32
- AVR-USB, a pure software implementation of USB for the AVR platform
- USBtiny, another software usb implementation for the AVR

Hardware



The prototype board including a ds1621 temperature sensor



The final hardware with the same sensor added

The hardware of the i2c-tiny-usb interface consists of the Atmel AVR ATtiny45 CPU, a cheap and easy to obtain microcontroller with 4 KBytes flash (of which ~2k are used in this application) and 256 Bytes RAM. The processor is surrounded by few other parts.

The USB interface

The USB interface of the i2c-tiny-usb interface is based on a pure software implementation and uses two pins of the AVR (PB0 and PB2). This software implementation supports low speed USB only which is signalled to the PC by resistor R1.

The I2C interface is implemented using a bitbanging approach. The hardware supported twi interface of the attiny45 is bound to hardware pins at the chip that are required for USB operation and can thus not be used for I2C. The bitbanging I2C interface being used instead may not be fully I2C compatible and thus not every I2C client chip may function correctly at this bus. No incompatibilities have been reported so far. The i2c-tiny-usb provides a software adjustable i2c clock delay allowing to configure the i2c clock. The default delay is 10us. Due to additional delays in the i2c bitbanging code this results in a i2c clock of about 50kHz.

For simplicity reasons all USB transfers are done via the control endpoint. Since the avr usb library does only support low speed devices it cannot use bulk transfers which are specified for high and full speed devices only. Low speed devices support so called interrupt transfers which are limited to a preset bandwidth while control transfers can use any free bandwidth (if there's any at all).

The device therefore uses control transfers for all of its communication. This requires some additional limitation to prevent multiple driver software (e.g. the kernel driver and the libusb based test application) to access the device at the same time. Under Linux this can be achieved by selecting certain access request types. This kind of access control may not be possible under other operating systems.

Power consumption

The whole device is a so called bus powered device. This means that the complete device is powered directly from USB. Therefore the AVR and one or more I2C client chips are powered from the USB VBUS signal.

The adapter itself draws less than 10mA and reports this to the host via its USB descriptors. The device is able to power I2C client chips as well. But since these chips vary in power consumption it is not possible to correctly include their demands into the device descriptors. It's your responsibility to keep an eye on the total power supply and especially to make sure that the entire device does not exceed the total USB limit of 500mA.

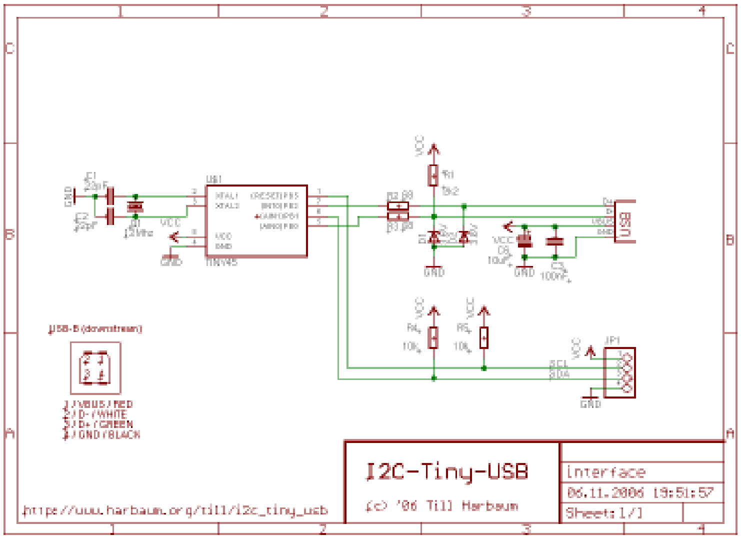
It is planned for future firmware versions to make the reported power consumption software configurable so the value can easily be adopted to the real power demands of the entire device.

Kernel driver

The i2c-tiny-usb is meant to be used with Linux. It comes with a Linux kernel driver that bridges between the USB and I2C subsystems in the Linux kernel. The driver then attaches to the USB device and make the i2c bus available to the i2c subsystem. Thus the entire setup is transparent to client applications like the lm_sensors framework and no special client chip drivers are required. Instead the drivers already present in the linux kernel are used with the i2c-tiny-usb as well. With e.g. the ds1621 temperature sensors used in the prototype the output of sensors may e.g. look like this:

```
ds1621-i2c-2-48
Adapter: i2c-tiny-usb at bus 003 device 017
temp:     +21.50°C (low  = +15.0°C, high = +10.0°C)  ALARM (HIGH)
```

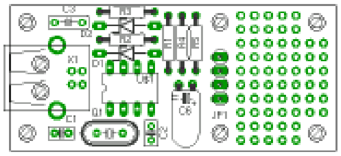
Schematics and PCB

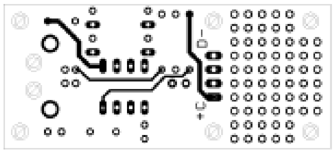


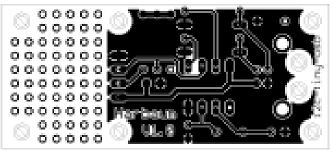
The zener diodes in the schematic are optional. They may be required since the i2c-tiny-usb is directly powered from the USBs VBUS singnal at 5V. The USB data lines (D+ and D-) are supposed to be operated at 3.3V only. Some PCs encounter problems at 5V and limiting the voltage to at most 3.6V may help. My prototype lacks these diodes since my PC works fine with D+ and D- at 5V.

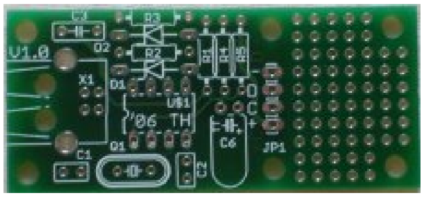
Resistor R1 is 2k2 instead of 1k5 for the same reason. It is meant to pullup to 3.3V. Since we are pulling up to 5V the higher resistance is required.

Below is the final PCB layout. It consists of the USB and I2C parts only and does not include a I2C client chip. Instead it comes with a solder area for easy prototyping. I do have some of these PCBs left. Just drop me an email if you want to buy one (6 EUR per PCB + 4 EUR shipping). You can easily etch a PCB youself. Since most of the connections are on the bottom side even a single sided PCB will work. You'll just have to add the four missing connections using thin wires.

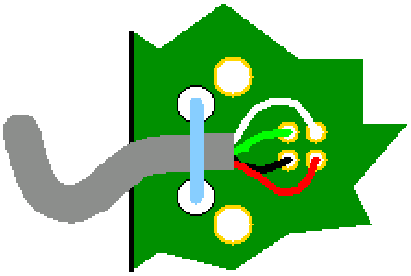

Part placement


Top (click for 600 dpi)

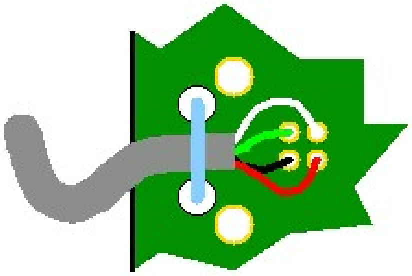

Bottom (click for 600 dpi)


The final PCB

The USB connector space on the PCB provides two additional holes to allow an USB cable to be directly and firmly attached to the device without the use of the USB connector. See the image below for the desired pinout.



Direct cable wiring schema ...



... and in reality (with pcf8574 client).

Compiling the firmware

The ATtiny45 is quite new and thus not all parts of the developement chain support this chip in their current release versions. The latest binutils (linker an assembler) and avrdude (programmer) support the new chips as well but gcc doesn't. The gcc-4.1.0 has to be patched to support some AVR cpus incl. the attiny45. Get the latest avr device patches (patch-newdevices) e.g here.

Uploading the firmware

If you are familiar with Atmel programming you probably know what you are doing. The Makefile in i2c-tiny-usb/firmware/Makefile assumes, that you are using the stk500 for programming the attiny45. Since all 6 user configurable pins of the attiny45 are required, the so called high voltage serial programming (hvsp) mode of the attiny45 has to be used. E.g. the stk500, the AVR-Doper and the AVR Dragon are supporting this mode.

Downloads

- Version 2007-01-05: i2c_tiny_usb-2007-01-05.zip
  - firmware version 1.04/2.04 (i2c clock bug fix)
  - Test application includes pcf8574 support
  - Linux kernel driver for tpa81 temperature array sensor
- Version 2007-01-03: i2c_tiny_usb-2007-01-03.zip
  - Bug fixes in usbtiny version
  - Added PCB layout
  - Added Windows libusb driver and ready-to-run test application
- Version 2006-12-03: i2c_tiny_usb-2006-12-03.zip
  - Added support for usbtiny
  - Firmware version 1.03 (avrusb)/2.03 (usbtiny)
  - Device supports i2c clock expansion
  - Linux kernel driver for cmps03 compass module
  - Updated testapp
- Version 2006-11-06: i2c_tiny_usb-2006-11-06.zip
  - Firmware version 0.01, the first working version
  - Kernel driver 0.9

Links

- AVR-USB, software USB for the AVR platform
- USBtiny, another software usb implementation for the AVR
- USB user space library libusb
- Win32 port of libusb
- The Linux USB Project
- Lm_sensors - Linux hardware monitoring

User contributions

- Henning Pauls i2c-tiny-usb
- Barry Carters feature extended version

Till Harbaum-Impressum http://www.harbaum.org/till