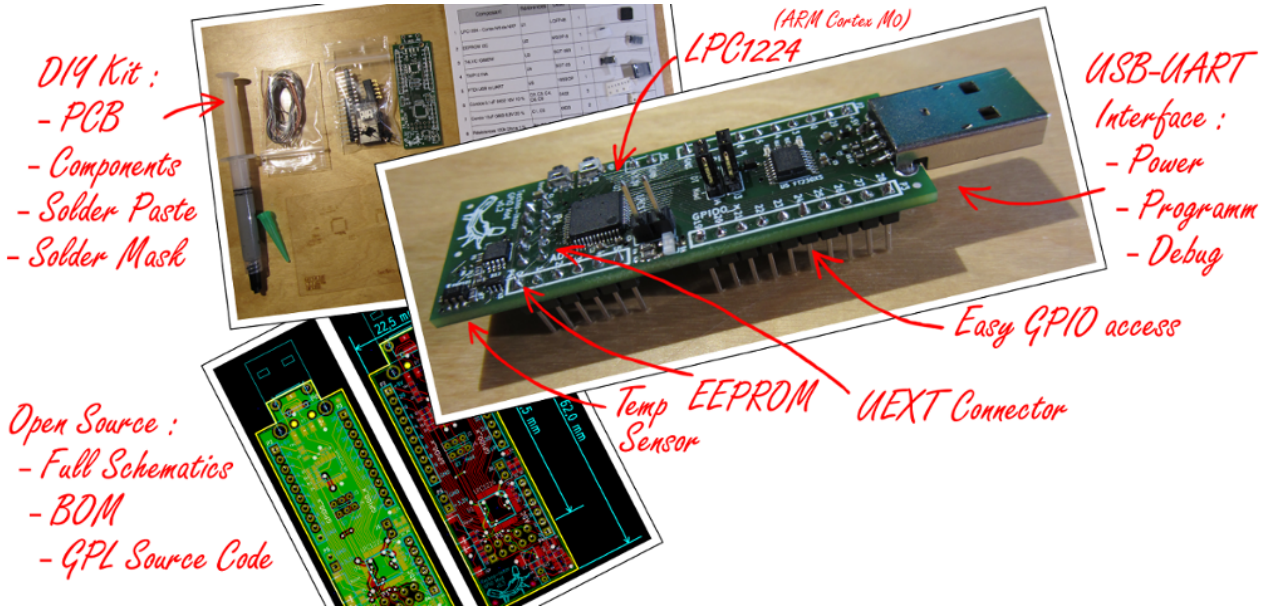


GPIO Demo module
System
Reference Manual

GPIODemo_SRM for Module v0.3

September 12, 2013 - Doc rev 0.3a

Author: Nathaël Pajani

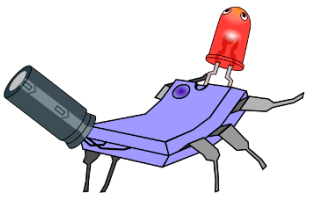


Table des matières

1 Introduction

You are reading the **System Reference Manual** for the GPIO Demo module. This manual covers the module use and design.

The GPIO Demo module is an electronics development and prototyping platform using the LPC1224 micro-controller from NXP [1].
The LPC1224 micro-controller has a Cortex-M0 ARM core, 32KB of flash memory, 4KB of internal SRAM, and multiple interfaces.
The module also includes an EEPROM memory and a temperature sensor on the I²C bus, a bi-color user LED (Red / Green), a reset button, an ISP mode select / User button, an USB-to-UART bridge (used for programming and easy communication with the module), 24 GPIO available on 2.54mm pins dispatched for easy use on prototyping boards and an UEXT connector.

Binaries for the GPIO Demo module can be generated using a gcc ARM toolchain and uploaded using the serial line (over USB thanks to the integrated USB-to-UART bridge) and our lpcprog tool.

The GPIO Demo module is designed for users interested in embedded ARM micro-controller development using free, libre and open source softwares only.
Every information about the design is available and all documentations are freely accessible. You can download the source files for the GPIO Demo module and modify them using KiCad EDA (GPL) according to the license terms found in the license section. You can create your own GPIO Demo module or a modified version.

In this document the GPIO Demo module will be referred as **the module**.

2 Licenses

2.1 Documentation

It is written in LaTeX and the PDF version is generated using pdflatex.

2.2 Hardware

You can produce your own original or modified version of the GPIO Demo module but you cannot sell them without our consent, even without profit.

2.3 Software

The lpcprog tool used to program the module is also under GPLv3 License.

1. http://www.nxp.com/products/microcontrollers/cortex_m0_m0/LPC1224FBD48.html

3 Hardware

3.1 Dimensions

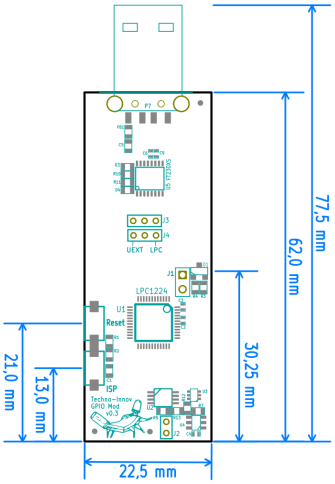


Fig 1 – USB A board type

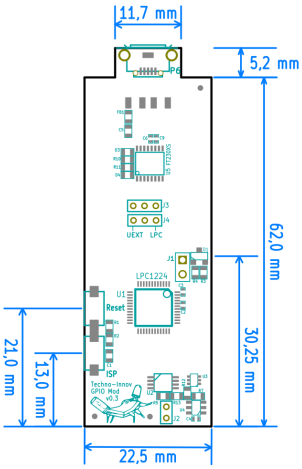


Fig 2 – micro USB board type

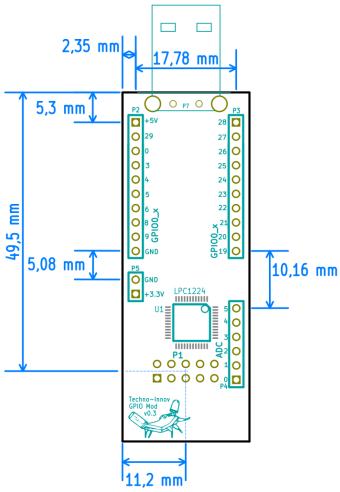


Fig 3 – Headers and UEXT connector (2.54mm)

Figures 1, 2 and 3 give the different dimensions and the position of the main elements (connectors, buttons and user led) of the module.

The only difference between the "*USB A*" and the "*micro USB* board types is the USB connector. A part of the Printed Circuit Board (PCB) is removed on the "*USB A*" board type to give way for the USB type A male connector.
Note : Not all components are shown on each figure for readability. All components but the USB connector are the same on both board types.

3.2 Connectors

The module has five 2.54mm pitch headers numbered P1 to P5 and one USB connector, either P6 or P7 depending on the board type. Refer to figure 4 for connectors position and to table 1 for a short description. Detailed description of the signals found on each connector pin follow.

Name	Description
P1	2x5 pins, 2.54mm pitch header.UEXT [a] connector.
P2	10 pins, 2.54mm pitch header. Provides +5V from USB, ground, and 8 GPIO from port 0.
P3	10 pins, 2.54mm pitch header. Provides 10 GPIO from port 0.
P4	6 pins, 2.54mm pitch header. Provides 6 ADC inputs (can be used as GPIO).
P5	2 pins, 2.54mm pitch header.
P6	USB micro-AB female connector. Available only on micro USB board type.
P7	USB A male connector. Available only on USB A board type.

TABLE 1 – Module Connectors Description

a. https://en.wikipedia.org/wiki/UEXT

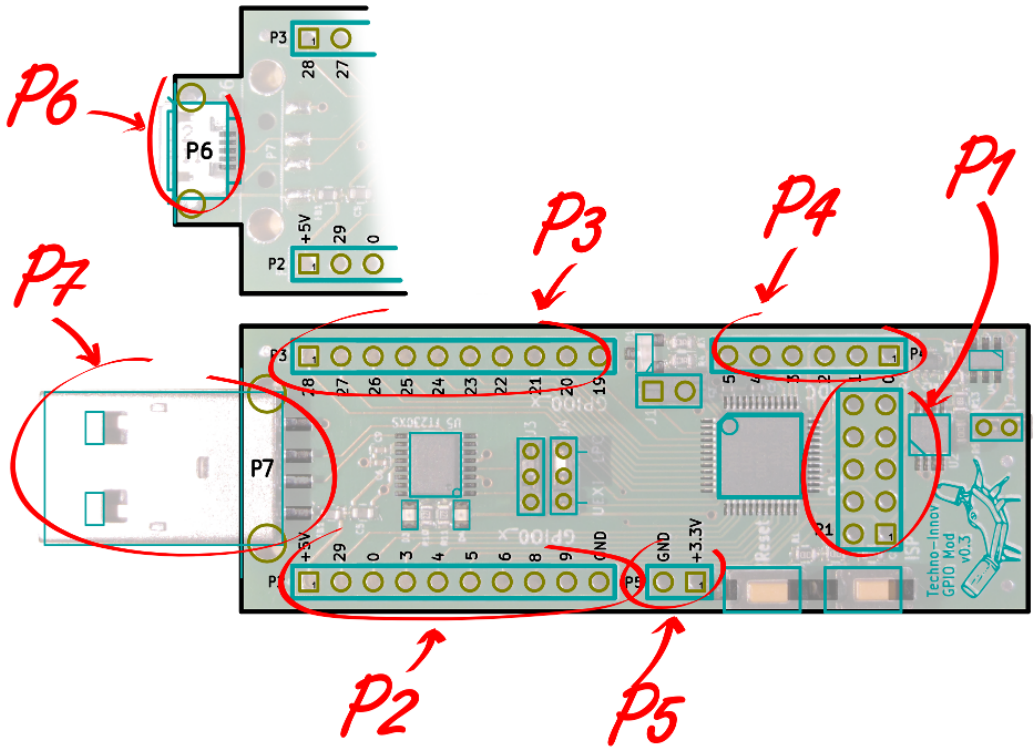


Fig 4 – Module Connectors

3.2.1 P1 Connector

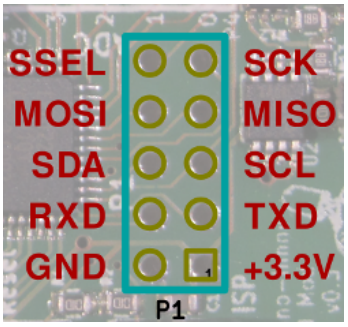


Fig 5 – P1 Connector

Pin #	Description
1	+3.3V : +3.3 Volt
2	GND : Ground
3	TXD : Transmit Data for UART, connected to LPC RXD on UART 0 (LPC pin 16 : PIO0_1).
4	RXD : Receive Data for UART, connected to LPC TXD on UART 0 (LPC pin 17 : PIO0_2).
5	SCL : Clock for I²C bus (LPC pin 25 : PIO0_10).
6	SDA : Bidirectional Serial Data for I²C bus (LPC pin 26 : PIO0_11).
7	MISO : Master In Slave Out for SPI bus (LPC pin 31 : PIO0_16).
8	MOSI : Master Out Slave In for SPI bus (LPC pin 32 : PIO0_17).
9	SCK : Clock for SPI bus (LPC pin 29 : PIO0_14).
10	SSEL : Slave Select for SPI bus (LPC pin 30 : PIO0_15).

TABLE 2 – P1 Connector Pinout

P1 connector is an UEXT [2] connector, as specified by Olimex [3]. The UEXT connector presents power and three serial buses : UART, I²C and SPI.
P1 connector uses a standard 2.54mm (0.1 inch) pitch header, with 2 rows of 5 pins. It can be either male or female, and mounted either on top or on bottom of the board. If mounted on top, other components prevent the use of the common plastic keyed-shroud used for common UEXT connectors. The key would be on the odd pins side, with

2. https://en.wikipedia.org/wiki/UEXT
3. http://www.olimex.com/

the module name and Techno-Innov mascot (tibug).

All P1 pins are connected to related pins on the LPC micro-controller. UART Pins are connected to UART 0 and crossed.

Note : The module is designed as an example for module development for the DomoTab platform, thus the module has the position of a device for the UART pins. This is transparent for the SPI and I²C bus.

3.2.2 P2 Connector

P2 connector is a standard 2.54mm (0.1 inch) pitch header, with 1 row of 10 pins, and can be mounted either on top or on bottom side of the board.
P2 connector provides access to USB +5V power supply and to 8 GPIO pins from port 0 of the LPC micro-controller.

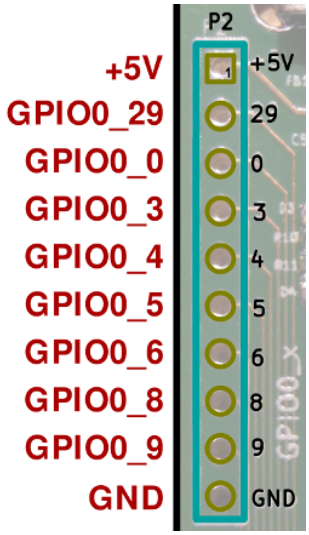


Fig 6 – P2 Connector

Pin #	Description
1	+5V : +5 Volt from USB connector
2	GPIO0_29 : LPC pin 14 : PIO0_29.
3	GPIO0_0 : LPC pin 15 : PIO0_0.
4	GPIO0_3 : LPC pin 18 : PIO0_3.
5	GPIO0_4 : LPC pin 19 : PIO0_4.
6	GPIO0_5 : LPC pin 20 : PIO0_5.
7	GPIO0_6 : LPC pin 21 : PIO0_6.
8	GPIO0_8 : LPC pin 23 : PIO0_7 / RXD1.
9	GPIO0_9 : LPC pin 24 : PIO0_8 / TXD1.
10	GND : Ground

TABLE 3 – P2 Connector Pinout

Note : Most P2 pins also provide alternate capture or match input functions for 32-bit timers. Refer to the LPC1224 documentation from NXP for full documentation of the alternate functions.

Note : When the board is not connected to a power source on the USB port the +5V is not present on pin 1 of P2 connector.

3.2.3 P3 Connector

P3 connector is a standard 2.54mm (0.1 inch) pitch header, with 1 row of 10 pins, and can be mounted either on top or on bottom side of the board.
P3 connector provides access to 10 GPIO pins from port 0 of the LPC micro-controller, including Serial Wire Debug (SWD) pins.

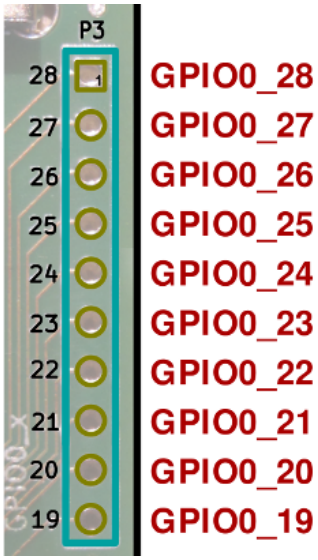


Fig 7 – P3 Connector

Pin #	Description
1	GPIO0_28 : LPC pin 13 : PIO0_28.
2	GPIO0_27 : LPC pin 12 : PIO0_27.
3	GPIO0_26 : LPC pin 11 : PIO0_26 / SWCLK.
4	GPIO0_25 : LPC pin 10 : PIO0_25 / SWDIO.
5	GPIO0_24 : LPC pin 9 : PIO0_24.
6	GPIO0_23 : LPC pin 8 : PIO0_23.
7	GPIO0_22 : LPC pin 7 : PIO0_22.
8	GPIO0_21 : LPC pin 6 : PIO0_21.
9	GPIO0_20 : LPC pin 5 : PIO0_20.
10	GPIO0_19 : LPC pin 4 : PIO0_19.

TABLE 4 – P3 Connector Pinout

Note : Most P3 pins also provide alternate capture or match input functions for 32-bit timers and input or output for comparators. Refer to the LPC1224 documentation from NXP for full documentation of the alternate functions.

3.2.4 P4 Connector

P4 connector is a standard 2.54mm (0.1 inch) pitch header, with 1 row of 6 pins, and can be mounted either on top or on bottom side of the board.
P4 connector provides access to 6 GPIO pins from port 0 and 1 of the LPC micro-controller. These pins are inputs for the A/D converter.

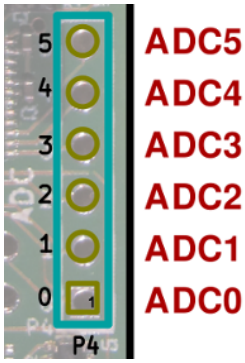


Fig 8 – P4 Connector

Pin #	Description
1	ADC0 : LPC pin 34 : PIO0_30 / AD0.
2	ADC1 : LPC pin 35 : PIO0_31 / AD1.
3	ADC2 : LPC pin 36 : PIO1_0 / AD2.
4	ADC3 : LPC pin 37 : PIO1_1 / AD3.
5	ADC4 : LPC pin 38 : PIO1_2 / AD4.
6	ADC5 : LPC pin 39 : PIO1_3 / AD5.

TABLE 5 – P4 Connector Pinout

3.2.5 P5 Connector

P5 connector is a standard 2.54mm (0.1 inch) pitch header, with 2 pins, and can be mounted either on top or on bottom side of the board.

P5 connector provides access to the +3.3V regulated power output of the FTDI FT230XS USB to UART bridge used to power the board when plugged on a USB port. This is also the power received from UEXT connector pin 1.

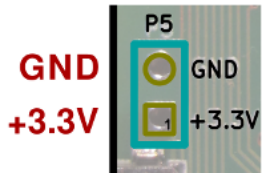


Fig 9 – P5 Connector

Pin #	Description
1	+3.3V
2	GND

TABLE 6 – P5 Connector Pinout

3.2.6 P6 and P7 connectors

P6 and P7 are one-time choice options. Only one of them is present, depending on the board type. Both are standard USB connectors. P6 is a female micro-AB port, and P7 is a male USB-A port.
Refer to the Universal Serial Bus (USB) [4] page on Wikipedia for pinout and more information on the USB bus and connectors.

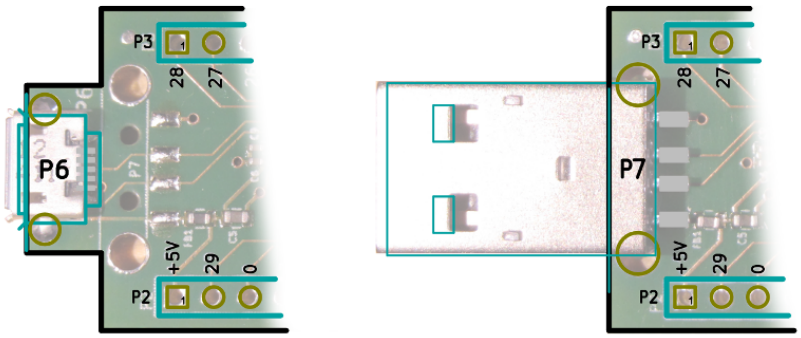


Fig 10 – P6 and P7 Connectors

3.3 Jumpers

The module has 4 configuration jumpers, numbered J1 to J4. Jumpers are common to all board types.

Name	Description
J1	Vref selection.
J2	EEPROM Write protect.
J3 and J4	UART cross for USB to UART bridge.

TABLE 7 – Module Jumpers Description

4. http://fr.wikipedia.org/wiki/Universal_Serial_Bus

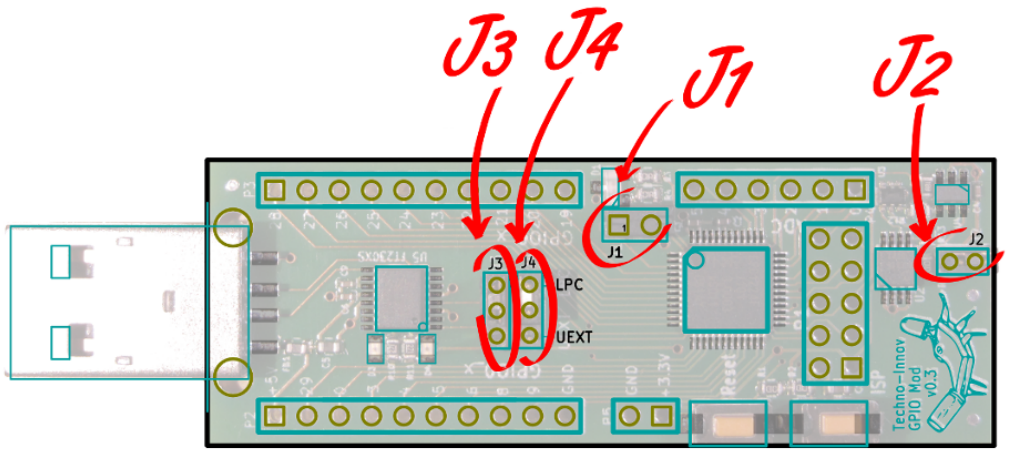


Fig 11 – Module Jumpers

3.3.1 J1 jumper

J1 jumper uses a 2.54mm (0.1 inch) pitch header and is used for Vref selection for the LPC internal Analog to Digital Converter (ADC).

When the jumper is on, Vref is connected to the 3.3V power supply. When the jumper is removed Vref is floating. Pin 1 of the jumper can be connected to an external reference voltage to provide a stable and known reference voltage. See section 4.1.5 for more details on Vref and ADC usage.

3.3.2 J2 jumper

J2 jumper is used to activate the write protection mechanism of the I²C EEPROM. When the jumper is on the write protect is off and the EEPROM can be written. When the jumper is removed the EEPROM is in read-only mode.

3.3.3 J3 and J4 jumpers

J3 and J4 jumpers are used to cross UART pins connected to the FTDI FT230XS USB to UART bridge.

When the jumpers are on the LPC side then the USB to UART bridge is connected to the LPC micro-controller and communication with the micro-controller is enabled (the micro-controller can be programmed if it is in ISP mode).

When the jumpers are on the UEXT side, then the USB to UART bridge can communicate with the UART on the UEXT connector.

Note : LPC Rx and Tx pins of the UART 0 are always connected to the UEXT UART pins, with Rx connected to Tx, so the module acts as a device from the point of view of the UEXT connector.

Note : Both jumpers **must** be on the same side of J3 and J4.

4 Electronics

The GPIO Demo module has been created using KiCad [5] EDA software suite for the creation of the schematics and printed circuit boards.

See page 29 in the annexes for the full schematics. The sources for the schematics are available for download from the module page [6] on Techno-Innov.fr.

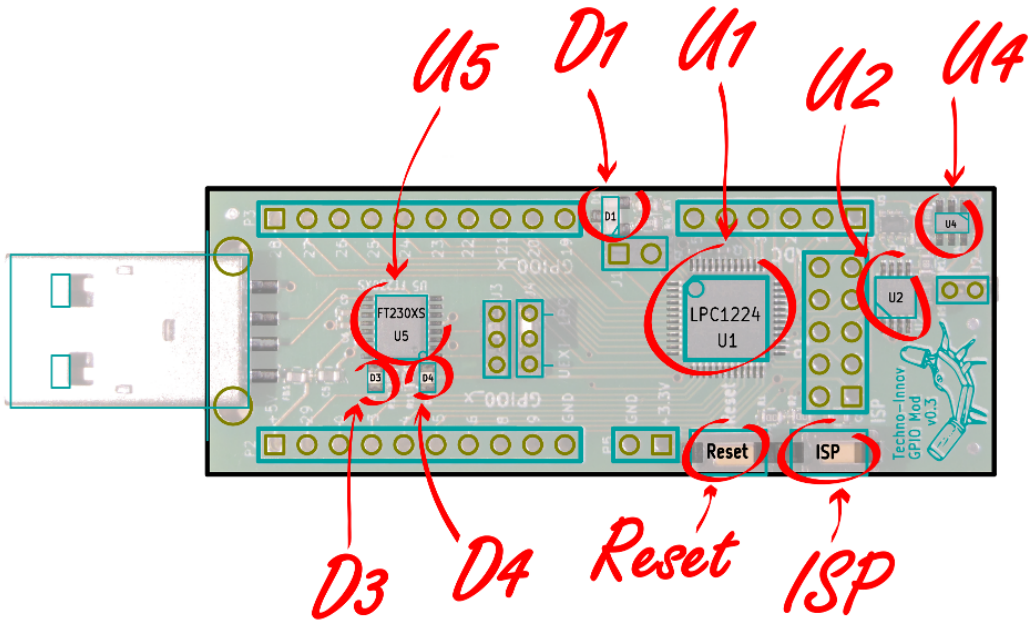


Fig 12 – Module Main Components

Name	Description
U1	LPC1224 ARM Cortex-M0 micro-controller.
U2	I²C EEPROM memory (1kB or 16kB).
U4	I²C TMP101 temperature sensor.
U5	FTDI FT230XS USB to UART bridge.
D1	User led, bicolore (red / green).
D3	Orange led : FTDI Rx activity.
D4	Green led : FTDI Tx activity..
Reset	Reset button for LPC1224 (SW2).
ISP	ISP mode select button for LPC1224 (SW1).

TABLE 8 – Module Main Components Description

4.1 Micro-controller LPC1224

The module's micro-controller is a LPC1224 from NXP [7]. The LPC1224 version used on the module is the LPC1224FBD48/101. All LPC1224 have an ARM Cortex-M0 core running at up to 45 MHz.

5. http://www.kicad-pcb.org/display/KICAD/
6. http://www.techno-innov.fr/technique-module-gpio-demo/
7. http://www.nxp.com/products/microcontrollers/cortex_m0_m0/LPC1224FBD48.html

The module uses the internal 12 MHz RC Oscillator as main clock. Its 1% accuracy is suitable for most applications.

Note : Refer to the LPC1224 documentation from NXP for full list and documentation of the LPC1224 features. Here are only the descriptions of the features used on the module.

4.1.1 Internal RAM

The LPC1224FBD48/101 has 4kB of internal SRAM mapped in one block at address 0x1000 0000.

4.1.2 Internal Flash

The LPC1224FBD48/101 has 32kB of internal FLASH memory, mapped at address 0x0000 0000. The flash memory programming requires no additional hardware thanks to the In-System Programming (ISP) and In-Application Programming (IAP) on-chip bootloader software.

See section 4.1.6 (Reset and ISP mode) or sections 5.2 (Code Compilation) and 5.3 (Uploading binary on target) for more information on internal FLASH memory.

4.1.3 Communication interfaces

The module makes use of the following communication interfaces found on the LPC1224 :

- **Two UARTs** : UART0 is connected to the UEXT connector and to the USB to UART bridge through J3 and J4 jumpers. UART1 Rx and Tx signals are on P2 connector pins 8 and 9.
  UART0 is used for In-System Programming of the LPC1224.
- **One I²C bus interface** supporting full I²C-bus specification and Fast-mode Plus with a data rate of 1 Mbit/s. I²C is connected to the UEXT connector and to the on-board temperature sensor and EEPROM memory. See section 4.2 for more information.
- **one SSP/SPI controller** with FIFO and multi-protocol capabilities. The SPI bus is found on the UEXT connector. Note that the SPI Slave Select signal is also used by the I²C bus to activate the I²C clock to the EEPROM memory.

4.1.4 GPIO

The module gives access to 24 GPIO pins (apart from the signals on the UEXT connector (P1) which can be used as GPIO too), dispatched on P2, P3 and P4 connectors.

Refer to tables 3, 4 and 5 for details of the signals available on these GPIO and to the LPC1224 documentation from NXP for full list of features for each GPIO.

4.1.5 ADC

The six GPIO pins on P4 connector are inputs channels 0 to 5 for the 10-bit ADC of the LPC1224 micro-controller.

The internal ADC uses the voltage on the Vref pin as reference voltage for the conversion. Jumper J1 allows you to use the LPC power supply (approx. 3.3V) as reference voltage (Jumper on). This is the easiest way to use the

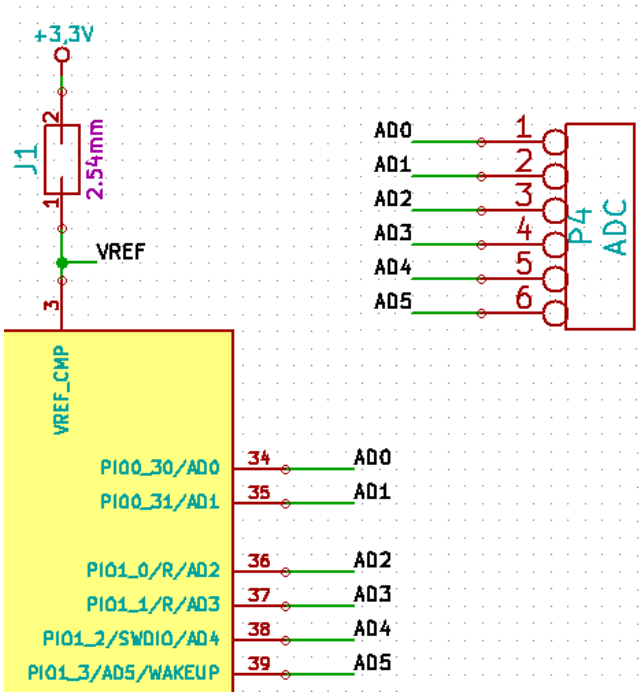


Fig 13 – ADC Input Pins and Vref Pin

ADC, but the measure will not be very accurate as the reference voltage is not precisely known.
If you need more precise conversions you must remove J1 jumper and connect a stable regulated reference voltage to pin 1 of the jumper J1. You should also add a small capacitor (1 to 100 nF) close to the pin to get a more stable reference voltage.

4.1.6 Reset and ISP mode

Reseting the LPC1224 without removing the power can be done with the Reset button (SW2).

To enter In-System Programming (ISP) mode after reset you must hold the ISP button (SW1) when you release the reset button. The LPC1224 bootloader considers a LOW level on the PIO0_12 pin as an external hardware request to enter ISP mode and start the ISP command handler. The sampling of the GPIO0_12 pin may take up to 3ms.

Refer to section 5.3 or to the LPC1224 user manual for more information on ISP mode.

If the ISP button is not held when th Reset button is released (and a valid user code is found in Flash memory) then the execution is transfered to the user program.

4.2 I²C

The module has two elements on the I²C bus : a temperature sensor and an EEPROM memory.

4.2.1 I²C Addresses

I²C Component	I²C Address R / W
1 kB EEPROM	0xA0 / 0xA1
16 kB EEPROM	0xA8 / 0xA9
Temperature sensor	0x94 / 0x95

Table 9 – I²C Addresses

Table 9 shows all the possible I²C Addresses for the components used on the module.
Note : There can be only **one** EEPROM memory chip at a time on the module. Refer to section 4.2.2 bellow for differentiation between the two possible EEPROM sizes.

4.2.2 External EEPROM

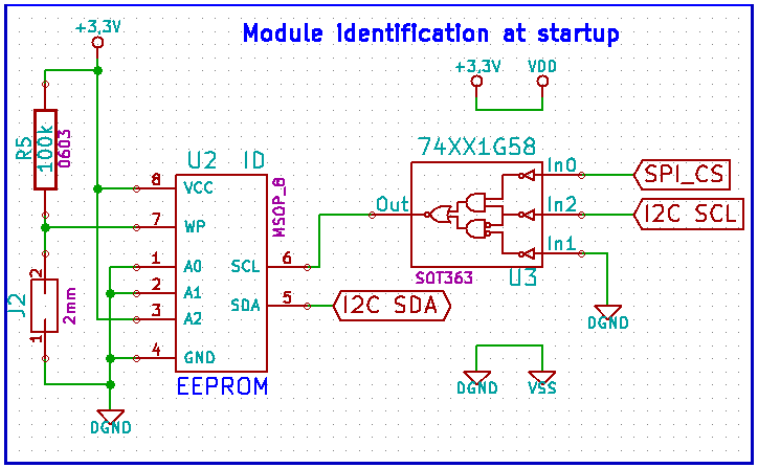


Fig 14 – EEPROM for Module Identification

EEPROMs with different sizes fit in the same footprint and can be used on the module, but they use different protocol depending on the size (one and a half or two memory address words). To allow software to differentiate those and chose the right protocol, we use different addresses depending on the memory size.
This is transparent from the hardware point of view as the small memory chips ignore the three address configuration pins found on the EEPROM chips and answer to eight consecutive I²C address couples (R / W) while the bigger ones take care of "A2" address configuration pin (and only this one in the selected package size).
As shown on figure 14 the hardware sets the A2 address configuration bit to "1", thus only small EEPROMs will answer to address "0xA0 / 0xA1", while bigger ones answer to address "0xA8 / 0xA9".

In addition to the address differentiation for the EEPROM size access to EEPROM memory requires use of a particular "EEPROM select" mechanism.
The EEPROM on the DomoTab modules are used to identify the module function, name, version, and possibly store the task code for the OS to communicate with the module. Many modules can be plugged at the same time on a single device, and the device needs a mechanism to read only one EEPROM at a given time to identify the module physical position when the device has many UEXT connectors. This is done using the SPI Slave Select signal from the UEXT connector. The I²C clock signal (SCL) is not transfered to the EEPROM memory chip when the SPI SSEL signal is not active (LOW).
This is true even if the module is not plugged on a UEXT connector. Thus before any access to the EEPROM memory one must activate the SPI SSEL signal (put the GPIO0_15 pins in LOW output state).

Refer to sample code for an example of how this is done in software and for the routines to read and write EEPROM memory.

4.2.3 Temperature sensor

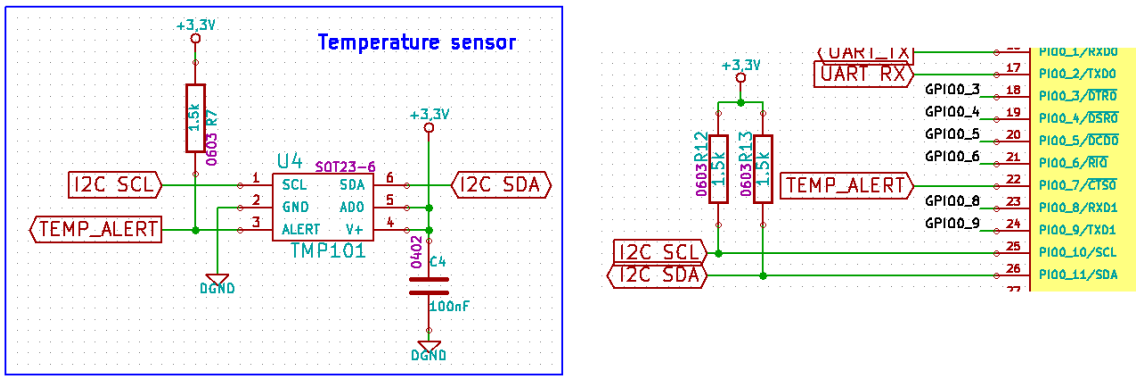


Fig 15 – Temperature sensor

The module has a TMP101 temperature sensor (from Texas Instrument) on the I²C bus (address 0x94 / 0x95).

This temperature sensor has an "alert" function available through a dedicated pin. This pin is routed to a wake-up capable pin of the LPC1224 micro-controller : GPIO0_7 (pin 22) which allows the temperature sensor to wake the micro-controller from "Deep-sleep" mode.
Refer to the LPC1224 User Manual from NXP for more information on the "Deep-sleep" mode and to the TMP101 documentation for the temperature alert signal.

4.3 User Led and Button

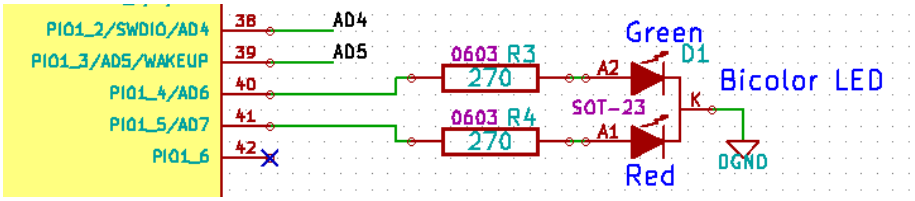


Fig 16 – User Led

The module has three leds and two buttons. The two leds connected to the USB to UART bridge (D3 and D4) and the Reset button have dedicated functions and cannot be assigned other functions.
The remaining led (D1) and button (ISP) can be used as the user wishes.

The D1 led is a bi-color red / green led connected to PIO1_4 (pin 40) and PIO1_5 (pin 41). Both can be turned on at the same time, providing a third color (orange).

Note : The PIO1_4 and PIO1_5 pins are not PWM capable so it's not possible to create shades between red and green without using a lot of processing power.

After reset the ISP button can be used by the user to any purpose. It's state can be read on pin PIO0_12 (pin 27).

4.4 USB to UART bridge

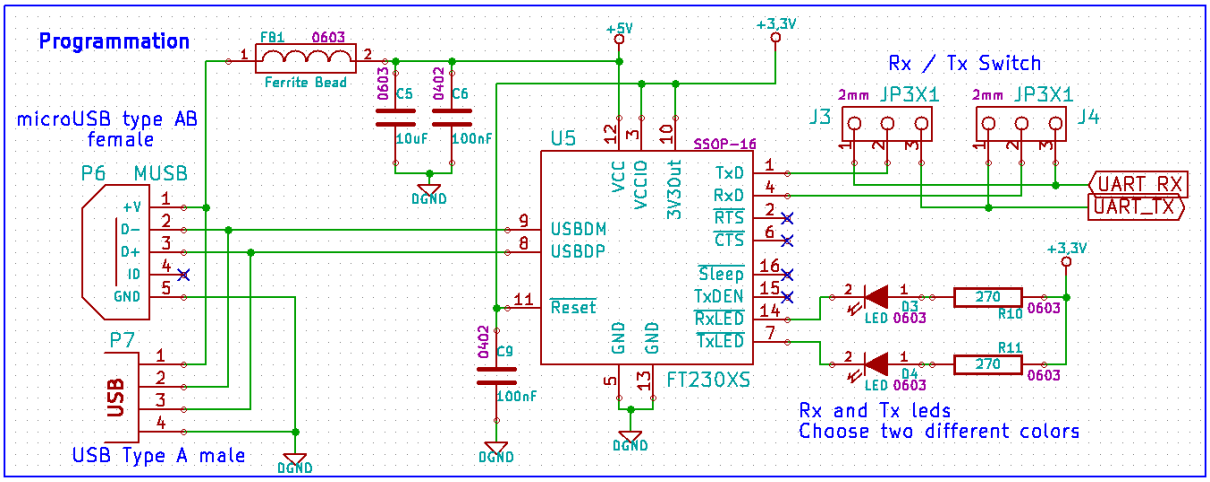


Fig 17 – USB to UART bridge

In order to ease the development process and the use of the module we added a USB to UART bridge on-board. This bridge is made by a FTDI FT230XS chip. It provides a 3.3V regulated voltage for the module and is well supported on most operating systems so there is usually no configuration required to use it as a serial line on the host development system, removing the need of any additional power source or of specific hardware to program the LPC1224 micro-controller and communicate with the module.

The FTDI chip controls two "activity" leds for Rx (D3, the orange one) and Tx (D4, green one) data over the serial link.

5 Software

The LPC micro-controller family uses ARM cores, which make them very easy to use. Apart from a few wrappers, all the code can be written in C and compiled using gcc.

ARM, NXP and other vendors provide sample code, but published under many different licences. The code we provide for the GPIO Demo module is published under the well known GPLv3 licence.

5.1 Sample Source Code

5.1.1 Grab the sources

An example application code can be downloaded from our git repository [8] using the following clone command :

```
user@host:~/sw$ git clone http://gitclone.techno-innov.fr/mod_gpio_demo
```

8. http://git.techno-innov.fr/?p=mod_gpio_demo;a=summary

5.1.2 Sample code content

This code provides the micro-controller definitions (Cortex-M0 specific definitions, registers, interrupts ...) and the routines required to start the micro-controller (bootstrap, vector table, power state, flash, clocks).

At the time of writing it also provides a basic set of library functions and the drivers for the interfaces found on the module. The list of supported features and interfaces is updated as the development goes on, so read the README file for the full list of supported features and interfaces.

The code has been split in three parts : `core/`, `drivers/` and `lib/` (with the associated directories under `include/` for the headers) :

- `core` : Contains all the required parts and system initialisations. Many functions in there are defined as weak aliases of dummy functions, so the code compiles even if no drivers are used. When these functions are redefined in the driver code they override the weak definition.
- `lib` : Contains the implemented parts of the small C library for our code. The micro-controller does not run a full Linux system, so the gnu libc must not be used, and even a µClibc is much more than what's required. Most of the code in these files come from the kernel implementations of libc parts.
- `drivers` : Contains the drivers for the different interfaces found on the module.

5.1.3 Sample code entry point

The main loop is in `main.c` in function `main()`, as with any C program, though `main()` is called by the bootstrap code (`Reset_Handler()` in `core/bootstrap.c`) and could have any name.
The calls to the system initialisation routines have been put together in the `system_init()` function. `SELECTED_FREQ` must be set to one of the `FREQ_SEL_**MHz` defined in `include/core/system.h` :

- `FREQ_SEL_12MHz`
- `FREQ_SEL_24MHz`
- `FREQ_SEL_36MHz`
- `FREQ_SEL_48MHz`
- `FREQ_SEL_60MHz`

Note : The frequency can go up to 60MHz despite what is said in the documentation, but the micro-controller needs much more power at higher frequencies.

Note : `system.h` provides two sleep functions (`msleep()` and `usleep()`). These are active sleeps and are not calibrated for frequencies other than 24 MHz. `usleep()` is **very, very** approximative.

5.2 Building the binary

5.2.1 Get a toolchain

Build has been tested using gcc, and only gcc, in the version provided by the emdebian project [9], but any ARM gcc toolchain should do.

In order to get the emdebian ARM gcc cross-toolchain you should add this repository to your apt sources : `deb http://www.emdebian.org/debian/ unstable main` and then update and install package `gcc-4.7-arm-linux-g`
It may be required that you create a symlink in `/usr/bin` for the arm gcc using one of these :

9. http://www.emdebian.org/

```
root@host:~# cd /usr/bin && ln -s arm-linux-gnueabi-gcc-4.7 arm-linux-gnueabi-gcc
```

or

```
root@host:~# update-alternatives -install /usr/bin/arm-linux-gnueabi-gcc
arm-linux-gnueabi-gcc /usr/bin/arm-linux-gnueabi-gcc-4.7 60
```

There's no need for the related libc package here, the libc does not fit in our micro-controller memory. Instead have a look at the content of the `lib/` directory, and add stuff there.

Alternatively you can download pre-compiled gcc toolchains (many different projects provide their own), or build your own one using crosstool-ng [10] or similar projects. For more information on what is a (cross-)toolchain, have a look at this information page on elinux.org [11].

5.2.2 Build command and options

Once done with the toolchain installation (or if you already have one) you should build using the provided `Makefile` by running the simple "`make`" command. Anyway you may want to change the `CROSS_COMPILE` variable from the `Makefile` and set it to the prefix of your toolchain.

The actual `Makefile` will include all the C files in the build, so if there are drivers you don't want to include in your build you must move the files away.

Note : Some drivers have dependencies, like the eeprom and the temp drivers, which depend on the I²C driver.

5.2.3 Build process

The specific information about the target (LPC1224 micro-controller) memory (Flash and RAM) used by the linker is in the `lpc_link_lpc1224.ld` linker script.

The vector table is defined in the `core/bootstrap.c` file, but the checksum of the first seven entries in the vector table is left unmodified. This checksum must be computed and placed in the eighth vector entry as the bootloader needs to find a valid checksum in the eighth entry to consider the user code as valid and transfer execution to the reset handler (first vector table entry).
This is done by the lpcprog tool before sending the binary to the target.

5.3 Uploading binary on target

5.3.1 Tools

To flash the binary (the one with .bin) to the LPC Flash you can use our lpctools package (not yet packaged for Debian as of 2013-09-10) available in the lpctools git repository [12]. Lpctools is released under GPLv3 licence.
Clone the repository using :

10. http://crosstool-ng.org/
11. http://elinux.org/Toolchains
12. http://git.techno-innov.fr/lpctools

```
user@host:~/sw$ git clone http://gitclone.techno-innov.fr/lpctools
```

Then build (`make`) the tools.

Note : Other tools may be used but have not been tested. No tools were found to be open source when we looked for tools to upload the binaries to the micro-controller. You must check that the tool you chose to use can take care of the checksum computation.

5.3.2 Connection with target and upload

Usual command line to upload a binary to the micro-controller :

```
user@host:~/sw$ ./lpcprog -d /dev/ttyUSB0 -c flash mod_gpio.bin
Part ID 0x3640c02b found on line 18
Part ID is 0x3640c02b
UID: 0x1228f5f5 - 0x4b324307 - 0x08333834 - 0x4d7b2c1a
Boot code version is 1.6
user@host:~/sw$
```

If you want to get information on the connected device use the **id** command of **lpcprog** :

```
user@host:~/sw$ ./lpcprog -d /dev/ttyUSB0 -c id
Part ID 0x3640c02b found on line 18
Flash now all blank.
Checksum check OK
Flash size : 32768, trying to flash 8 blocks of 1024 bytes : 8192
Writing started, 8 blocks of 1024 bytes ...
user@host:~/sw$
```

Note : The part information definition for each supported micro-controller is in the lpctools package. See lpctools readme and lpcprog or isp help (-h option) or manpages for more information.

6 Do It Yourself

This section is dedicated to those who bought the kit to build the module themselves, or to those who bought the components on their own, or created a modified version of the module and need some help to put the parts together and get an operational module.

6.1 Tools

The module assembly requires few tools, but you'll need them :

- Precision tweezer (though any tweezer will do).
- Hot plate (or hot skillet, or equivalent stuff).
- Soldering iron, for the through hole parts (connectors and jumpers).
- Dremel or coping saw for those who want the USB A board type.
- A magnifier may also come in handy, especially for the small temperature sensor (U4) and the multifunction gate (U3).

- A multimeter for the checks, though not required, may be helpful.

For the hot plate, almost any kind of hot plate should do, but it should have a plane surface, and provide a temperature of 250°C or more. Reflow oven, or any oven able to provide 230*C or more should be OK.
We will concentrate our explanations on the use of a hot plate, but you can find information about other approaches to solder SMD components in this sparkfun tutorial [13], which is where I learnt about the hot plate soldering method we now use to manufacture our products.

Note : There are other solutions to solder the board, including hand soldering tricks. Use your favorite search engine on the Internet if you are interested in alternate solutions.

6.2 Materials

All the material required is included in the soldering kit, but if you did not chose to buy one from us, here is what you need :

- Solder paste.
- Desoldering wire. You may not need it if you are very careful with the solder paste application and get the right amount of solder paste, but I often need some.
- Solder (wire) for the through hole parts.

For the solder paste, we tested only one and are happy with it. When you chose solder paste, take care of the melting temperature. The lower the temperature, the less risk to damage the components. But be careful of those with melting temperature below 200°C, they certainly contain lead.

6.3 Components and options

Refer to the Bill Of Material (BOM) page 29 to check that you have all the component. There are two presentations of the BOM with the same content, one with identical references put together, and one ordered by functional blocks.

Our kits contain all the components, including both P6 and P7 connectors (USB-A male and micro-USB AB female) though you need only one of them.

Of course, alternate parts can be used for many of the references from our BOM. If you chose an alternate component, check for the footprint **and** the pinout.
All the pin headers are not necessary either, and you can chose to solder wires or components directly to the Printed Circuit Board (PCB) instead.

If you chose not to populate the EEPROM write protect jumper (J2), remember that you will need a strap to program the EEPROM once.
And of course you can chose to solder only some of the components. For example if you do not need the temperature sensor (U4) for your application you can leave it apart, including the decoupling capacitor (C4) and pull-up resistor (R7) used for it. The "functional blocks" version of the BOM is split in separate functional parts to ease the identification of such groups of components.

Note : If you bought one of our kits, do **not** remove the parts from the paper they are taped on. Many parts look similar and have no clue to differentiate one from another.

13. https://www.sparkfun.com/tutorials/59

6.4 PCB preparation

If you want to solder the micro USB board type (micro-AB USB connector - P6) you have nothing particular to do for this step. Only make sure the PCB is clean and dry.

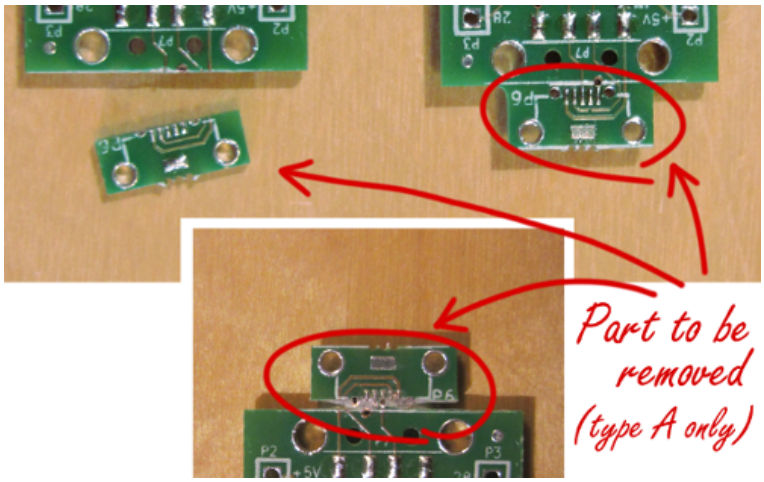


Fig 18 – PCB Part to remove for USB A board type

For those who want to solder the USB A board type (USB A connector - P7), you must remove the PCB part holding the micro-USB connector.
To remove this part the best tool is a dremel, though any coping saw will do perfectly. Be very careful with either of these tools, and use a bench vice or a clamp to hold the board firmly while you work on it, but be careful not to damage the coper tracks (use a small wooden plank or a folded paper to protect the PCB).

6.5 Stencil

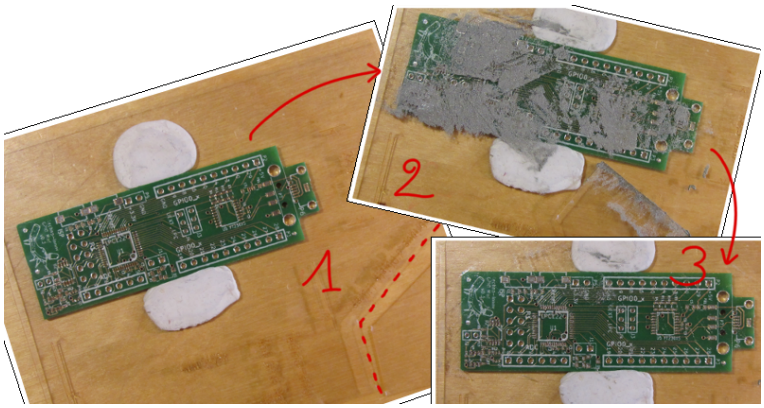


Fig 19 – Place stencil and apply solder paste

To ease the solder paste application on the Printed Circuit Board (PCB) we provide a SMD stencil. This stencil has been laser cut in a Rhodoid sheet (usually used for cooking). It's transparency makes it easy to position the stencil over the PCB.

- First of all, fasten the PCB to the worktop, either with adhesive or with blu-tack (used to fix posters to walls). I prefer the blu-tack solution as it will be possible to use it to maintain the stencil in the right position.

- Tear off the removable part of the stencil, it makes a little squeegee you'll need to apply the solder paste (it's the corner with three small bonds to the rest of the stencil).
- Position the stencil on top of the PCB. The position is OK when the holes fit on the solder pads of the PCB.
- Then use the solution you chose to maintain the PCB for the stencil, so it will not move while you apply the solder paste. The stencil must not bend over the PCB.

Note : If you want to cut your own stencils, keep in mind that the laser will melt the plastic and holes will be bigger than planned. Reduce the size of the holes by 10 to 40% to get the right hole size.
The holes sizes depend on the thickness of the Rhodoid sheet (100 µm to 150 µm), and must be adjusted to get the right amount of solder paste in the hole when you remove the stencil.

6.6 Applying solder paste

Solder paste application is made easy using the SMD stencil. Once the stencil is settled on the PCB, get some solder paste on it. No need to use too much, you'll be able to add more later if you did not get enough in the first place.

Use the removable part of the stencil to spread the solder paste over the PCB (make sure there is some solder in every hole) and remove any excess of solder by scrapping the stencil with the squeegee.

Note : The holes size is proportional to the Rhodoid thickness, so there must not be bumps of solder above the stencil surface.

You can now remove the stencil. Start raising one corner while you hold the rest in place, so the solder stays in the right position. Check **on the PCB** that the solder stayed on the PCB and not in the stencil holes. If some solder stayed in the holes, don't care, so long as there's enough on the PCB. If there's only one or two spots where the solder did not stay, do not start again from scratch, rather add some solder using the point of a needle, or remember these spots and we will see later how you can add solder to a pin of a SMD part easily.

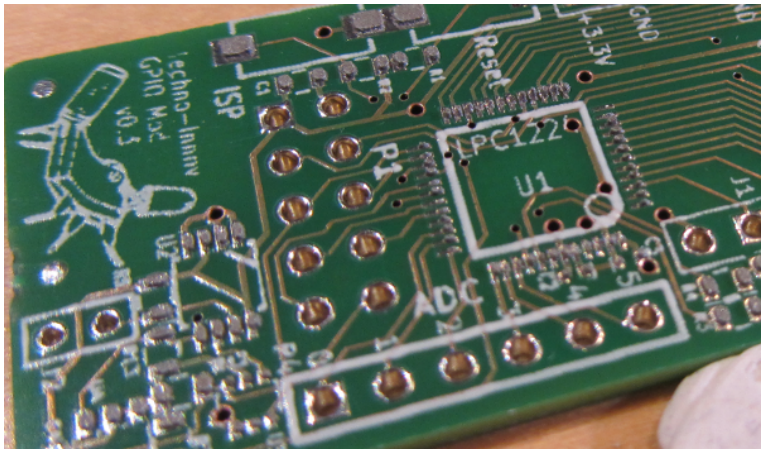


Fig 20 – Solder paste after removing the stencil

Note : Do not panic if the solder chose to stay on the stencil, it is re-usable. Wipe the solder from the PCB with a paper towel and start again.

6.7 Pick and place

You now need to populate the board with the SMD parts (and only these ones). This is an iterative task, starting with one reference from the BOM, getting the parts out of the reel segment (remove the thin plastic cover which holds them in place), and place the parts on the PCB. Make sure you do not place them in the wrong place by referring to the placement sheet included in the Kit. Each pin of the part must lay on some solder.

Be careful of the orientation of the big parts which have a mark for the first pin, and for the Leds for which the mark is on the bottom of the Led !

The placement of the components does not have to be perfect as the liquid solder will gather under each pin and align the part. If the solder makes an uniform layer when you put the part down on it it's no problem either, for the same reason.

Note : The marks for U2, U3 and U4 are hard to see, use oblique light if you do not have a magnifier at hand.

Note : There is no real order for the parts placement, but I prefer starting with the small SMD resistors and capacitors as the solder paste is sticky at the beginning of the process and thus they will stay in the right spot if the board moves (alone, of course ...).

6.8 Hot Plate soldering

Once you are done with the placement of all the SMD components you can move to the effective soldering process. This part is the easiest. Place the board on the hot plate (the parts should stay in place while you move the board, but be careful anyway) and before turning the power on, give a try at removing the board without difficulty. I use the tweezers and pick the board by the holes of the headers.

Turn the power on. Be very careful as the plate is now gonna be very hot !
You will notice easily the start of the reflow process as the solder will turn shiny when getting liquid. Depending on the power of the device the process may take between one to four or five minutes. When every solder spot has melted, turn the hot plate power off and remove the board. Be very careful with this step and do not drop the board or the components will move before the solder had time to cool down.

6.9 Checks and Fixes

Note : For this part, you will need to have an easy access to the Schematics, to prevent destroying the board while trying to remove a connection between two pins which seems to be done by an excess of solder while both pins are connected to the ground plane ... (sounds like I did it ?)

The first check is a visual one. You must make sure that no two consecutive pins got soldered together by an excess of solder paste. These are easy to see (Figure 21), and also easy to remove.

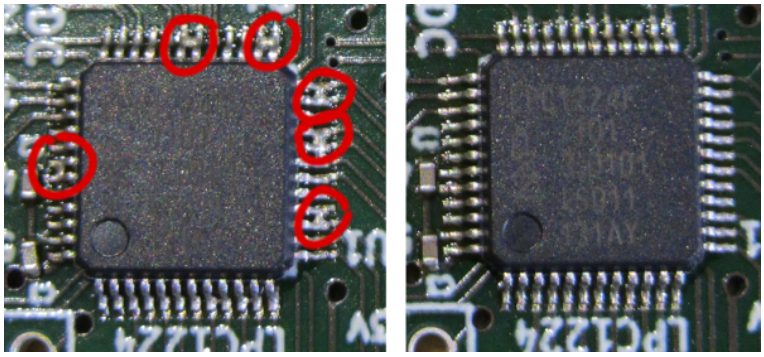


Fig 21 – Short circuits where there was to much solder paste

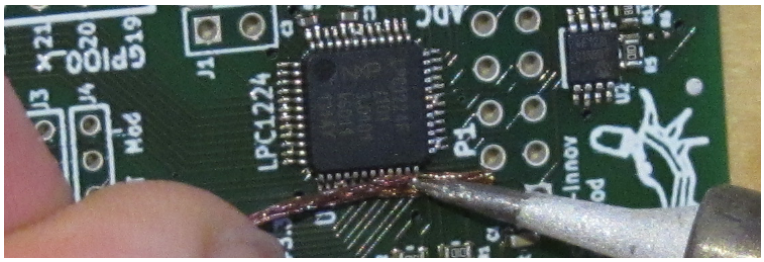


Fig 22 – Using desoldering wire

- For those with only a little amount of solder, clean the iron tip with a paper towel (careful, it's very hot) and apply it between the two pins and push slightly when the solder has melted. If it does not work, move to the second solution.
- For the other, use the desoldering wire (you've got some in the soldering kit, it's the copper braid). Put the end of the desoldering wire on top of the solder bump (do not hold it to close to the end, it will become hot too) and put the iron (cleaned) on top of it and push a little bit so the braid is well in contact with the bump of solder to be removed. Refer to Figure 22

Note : Use a clean part of the desoldering wire, or it will not soak up the solder. When some length of the wire is stained with solder, cut it away with pliers.

The second check is making sure every pin got enough solder. Start by looking at those you noticed when you removed the stencil and were lacking solder paste. A visual inspection may be enough here too. Use of a magnifier may help. The other way to check is with a multimeter if you have one. Use the connectivity check mode and make sure there is contact between the top of the pin and what lays at the other end of the coper track.
When you find a pin with a lack of solder the fix is easy. Do not use solder wire, it will bring too much solder at once, and you'll get short circuits. Use solder paste. Take a small amount of solder paste with the point of the tweezers and put it down where you want it. Then use the solder iron to melt it.

6.10 Through hole

Once you are done with checking the result of the hot plate soldering you can start the last soldering step, the easy one, and solder all the through hole connectors (headers and jumpers).

Use the plastic jumper to hold the headers to avoid burning your fingers. And of course, use the solder wire, not the solder paste.

Do not forget to fill the used USB connector holes, they are here to secure the connector to the PCB, preventing any constraint on the pins when you plug the cable in.
I usually add some solder to the push buttons. This is no required, but makes them more strong.

6.11 Final test

For the final test, plug the board on your computer and look at the bi-color led, it should show two very small glow, one red and one green.

You can also look at your system logs, and get some messages about a new USB serial device.

6.12 Clean and tidy

You think you're done ?
Sounds like you're right. But your workplace will look better with some cleanup :) Give it a thought at least :)

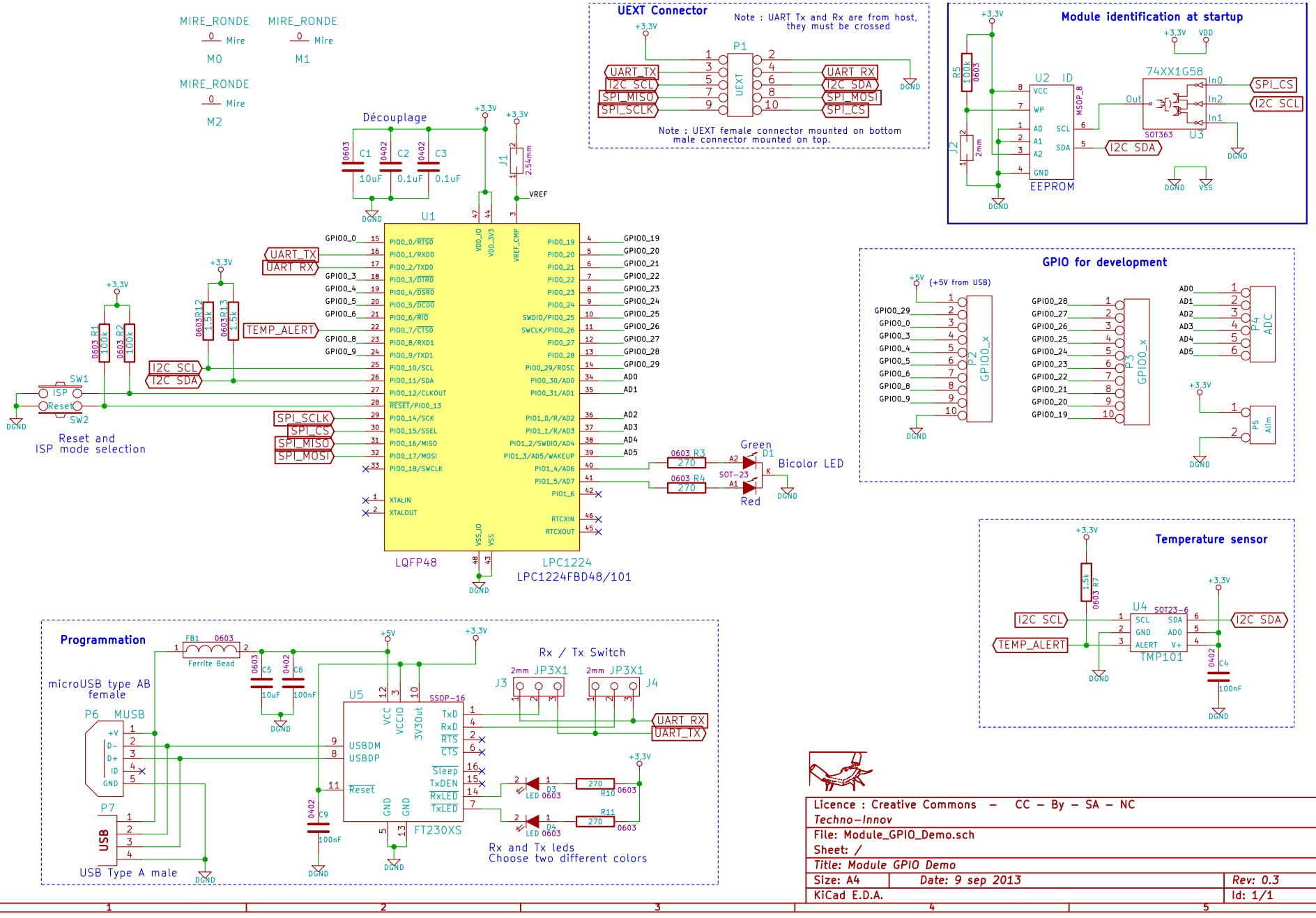
7 Annexes

7.1 Schematics

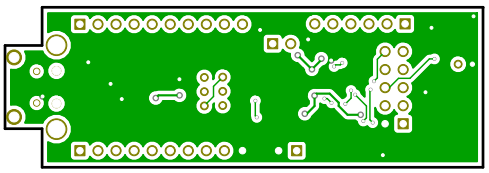
The board schematics and PCB layout have been created using KiCad [14] EDA software suite. You can download the sources on the module page [15] on Techno-Innov.fr.

(See on next pages)

14. http://www.kicad-pcb.org/display/KICAD/
15. http://www.techno-innov.fr/technique-module-gpio-demo/

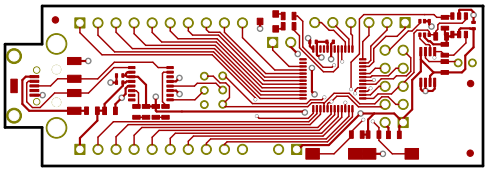

MIRE_RONDE
0 Mire
M0
MIRE_RONDE
0 Mire
M1
MIRE_RONDE
0 Mire
M2
UEXT Connector
Note : UART Tx and Rx are from host, they must be crossed
P1
UART_TX
I2C_SCL
SPI_MISO
SPI_SCLK
UART_RX
I2C_SDA
SPI_MOSI
SPI_CS
UEXT
DGND
Note : UEXT female connector mounted on bottom male connector mounted on top.
Module identification at startup
U2 ID
EEPROM
74XX1G58
U3
SPI_CS
I2C_SCL
I2C_SDA
Découplage
C1 10uF
C2 0.1uF
C3 0.1uF
J1
VREF
U1
LQFP48
LPC1224
LPC1224FBD48/101
UART_TX
UART_RX
TEMP_ALERT
I2C_SCL
I2C_SDA
SPI_SCLK
SPI_CS
SPI_MISO
SPI_MOSI
SW1
ISP
Reset
SW2
Reset and
ISP mode selection
R1 100k
R2 100k
R12 1.5k
R13 1.5k
R3 270
R4 270
D1
Green
Red
Bicolor LED
GPIO for development
P2 GPIO0_x
P3 GPIO1_x
P4 ADC
P5 Alim
Programmation
microUSB type AB female
P6 MUSB
P7 USB
USB Type A male
FB1 Ferrite Bead
C5 10uF
C6 100nF
C9 100nF
U5 FT230XS
SSOP-16
Rx / Tx Switch
J3
J4
Rx and Tx leds
Choose two different colors
D3 LED
D4 LED
R10 270
R11 270
Temperature sensor
U4 TMP101
SOT23-6
C4 100nF
R7 1.5k
Licence : Creative Commons – CC – By – SA – NC
Techno-Innov
File: Module_GPIO_Demo.sch
Sheet: /
Title: Module GPIO Demo
Size: A4
Date: 9 sep 2013
Rev: 0.3
KiCad E.D.A.
Id: 1/1

Bottom copper layer



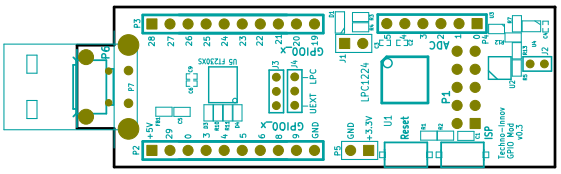
Bottom copper layer

Top copper layer



Top copper layer

Top silk screen



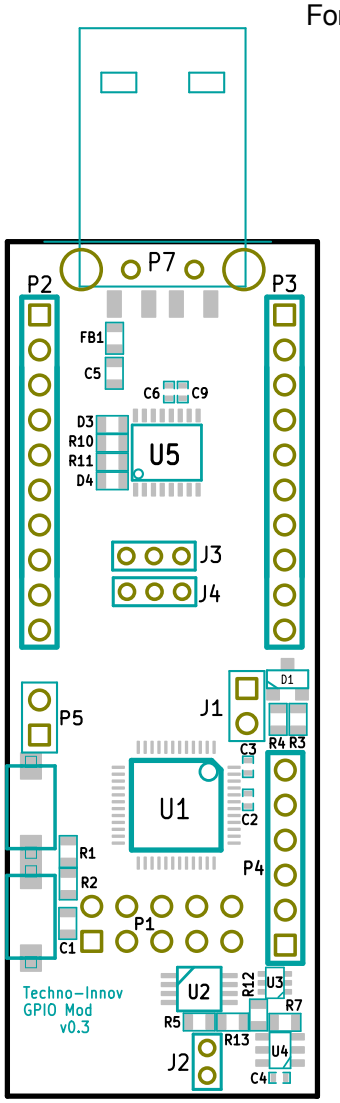
Top silk screen

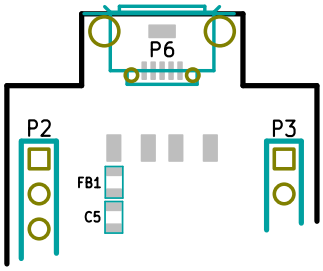
File: Module_GPIO_Demo.brd
Sheet:
Title: Module GPIO Demo
Size: A4 | Date: 6 sep 2013 | Rev: 0.3
KiCad E.D.A. | Id: 1/1

Licence : Creative Commons – CC – By – SA – NC
Techno-Innov
File: Module_GPIO_Demo.brd
Sheet:
Title: Module GPIO Demo
Size: A4 | Date: 6 sep 2013 | Rev: 0.3
KiCad E.D.A. | Id: 1/1

P6 or P7 : chose between
USB A (P7) or USB microB (P6)

For P7 : remove PCB part using
appropriate tools.





D3 and D4 :
The cathode (-) is on the U5 side

P1 to P5 : top or bottom mount
(If you mount P2 to P5 on bottom,
then mount P1 on top for use on
prototyping board)

J1 to J4 : top mount

7.2 BOM

7.2.1 Block version

Part Description	Ref	Module	Nb	Vendor	Vendor ref	Farnell ref
Micro-controller						
LPC1224	U1	LQFP48	1	NXP	LPC1224FBD48/101	1862465
Decoupling capacitors 100nF	C2, C3	0402	2	Multicomp	MCCA000050	1758896
Filter capacitor 10µF	C1	0603	1	TDK	C1608X5R0J106M	2112705
Pull-Up resistors 100k Ohms	R1, R2	0603	2	Multicomp	MC0063W06031%100K	9330402
I²C Pull-Up resistors 1,5k Ohms	R12, R13	0603	2	Multicomp	MC0063W06031%1K5	9330607
Bi-color Led resistors 100 Ohms	R3, R4	0603	2	Multicomp	MC0063W06031%100R	9330364
SMD Led Red / Green	D1	SOT-23	1	Kingbright	KM-23ESGW	1142614
SMD switchs	SW1, SW2		2	Multicomp	DTSM-32S-B	9471898
UEXT connector						
UEXT female connector 2x5	P1	2,54mm	1	Multicomp	2214S-10SG-85	1593490
Temperature sensor						
Decoupling capacitors 100nF	C4	0402	1	Multicomp	MCCA000050	1758896
Pull-Up resistors 100k Ohms	R7	0603	2	Multicomp	MC0063W06031%100K	9330402
I²C temperature sensor TMP101	U4	SOT-23	1	TI	TMP101NA/250G4	1207304
USB Bridge						
Led resistors 270 Ohms	R10, R11	0603	2	Multicomp	MC0063W06031%270R	9330917
Decoupling capacitors 100nF	C6, C9	0402	2	Multicomp	MCCA000050	1758896
Filter capacitor 10µF	C5	0603	1	TDK	C1608X5R0J106M	2112705
SMD chip bead	FB1	0603	1	TDK	MMZ1608R601A	1669700
Rx Led - Orange	D3	0603	1	Vishay	VLMO1300-GS08	2251473
Tx Led - Green	D4	0603	1	Vishay	VLMG1300-GS08	2251461
FT230XS USB to UART	U5	16SSOP	1	FTDI	FT230XS	2081321
Micro-USB type A-B female	P6	SMD	1	Molex	47590-0001	1568022
USB Type A male	J2	SMD	1	Multicomp	MC32605	1696546
Male headers 1x3	J3, J4	2mm	2	Fisher	SLY1.085.50G	9729135
Jumpers 2mm black			2	Harwin	M22-1900005	510932
Identification						
Pull-Up resistors 100k Ohms	R5	0603	1	Multicomp	MC0063W06031%100K	9330402
Multi-function 74LVC1G58DW	U3	SOT-363	1	Diodes Inc.	74LVC1G58DW-7	1893838
EEPROM 1Ko 8x256 400kHz	U2	MSOP-8	1	Microchip	24AA08-I/MS	1331278
GPIO Connectors						
Male headers GPIO (30 pins)	P2, P3, P4, P5, J1	2,54mm	1	Fischer	SL1.025.36Z	9729038
Jumpers 2.54mm black			1	Harwin	M7686-05	3218480

TABLE 10 – BOM by functional block

7.2.2 Easy order version

Part Description	Ref	Module	Nb	Vendor	Vendor ref	Farnell ref
LPC1224	U1	LQFP48	1	NXP	LPC1224FBD48/101	1862465
I²C temperature sensor TMP101	U4	SOT-23	1	TI	TMP101NA/250G4	1207304
FT230XS USB to UART	U5	16SSOP	1	FTDI	FT230XS	2081321
EEPROM 1Ko 8x256 400kHz	U2	MSOP-8	1	Microchip	24AA08-I/MS	1331278
Multi-function 74LVC1G58DW	U3	SOT-363	1	Diodes Inc.	74LVC1G58DW-7	1893838
SMD chip bead	FB1	0603	1	TDK	MMZ1608R601A	1669700
Decoupling capacitors 100nF	C2, C3, C4, C6, C9	0402	2	Multicomp	MCCA000050	1758896
Filter capacitor 10µF	C1, C5	0603	1	TDK	C1608X5R0J106M	2112705
Pull-Up resistors 100k Ohms	R1, R2, R5, R7	0603	2	Multicomp	MC0063W06031%100K	9330402
I²C Pull-Up resistors 1,5k Ohms	R12, R13	0603	2	Multicomp	MC0063W06031%1K5	9330607
Bi-color Led resistors 100 Ohms	R3, R4	0603	2	Multicomp	MC0063W06031%100R	9330364
Led resistors 270 Ohms	R10, R11	0603	2	Multicomp	MC0063W06031%270R	9330917
SMD Led Red / Green	D1	SOT-23	1	Kingbright	KM-23ESGW	1142614
Rx Led - Orange	D3	0603	1	Vishay	VLMO1300-GS08	2251473
Tx Led - Green	D4	0603	1	Vishay	VLMG1300-GS08	2251461
SMD switchs	SW1, SW2		2	Multicomp	DTSM-32S-B	9471898
Micro-USB type A-B female	P6	SMD	1	Molex	47590-0001	1568022
USB Type A male	J2	SMD	1	Multicomp	MC32605	1696546
UEXT female connector 2x5	P1	2,54mm	1	Multicomp	2214S-10SG-85	1593490
Male headers 1x3	J3, J4	2mm	2	Fisher	SLY1.085.50G	9729135
Jumpers 2mm black			2	Harwin	M22-1900005	510932
Male headers GPIO (30 pins)	P2, P3, P4, P5, J1	2,54mm	1	Fischer	SL1.025.36Z	9729038
Jumpers 2.54mm black			1	Harwin	M7686-05	3218480

TABLE 11 – BOM by reference

7.3 Document revision History

Version	Date	Author
0.3a	September 12, 2013	Nathaël Pajani

7.4 Disclaimer

The GPIO Demo moduleis provided "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the GPIO Demo moduleis with you. Sould the GPIO Demo moduleprove defective, you assume the cost of all necessary servicing, repair or correction.