

AN11552

NXP Smartphone Quick-Jack Solution

Rev. 1.1 — 3 June 2014

Application note

Document information

Info	Content
Keywords	LPC8xx, LPC800, LPC812, Smartphone Quick-Jack Solution, HiJack, MCU, Mobile phone, Headset, Audio, smartphone
Abstract	This application note describes the LPC800-based NXP Smartphone Quick-Jack Solution. The NXP Smartphone Quick-Jack Solution repurposes the standard 3.5 mm stereo audio jack found on most smartphones into a self-powered data channel that makes communication with these devices as easy as plugging a headset jack into the audio port.



Revision history

Rev	Date	Description
1.1	20140603	Added remark that Quick-Jack has been tested with iOS 6.1.6 and iOS 7.0.4
1	20140519	First release

Contact information

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

This application note describes the LPC800-based NXP Smartphone Quick-Jack Solution. After this brief introduction, the following topics will be discussed in more detail:

- [Quick-Jack fundamentals](#)
- [Quick-Jack hardware description](#)
- [Quick-Jack software description](#)
- [Measurements and waveforms](#)
- [NXP Quick-Jack Quick Start Guide](#)
- [Building and flashing the LPC812 firmware](#)
- [Quick-Jack smartphone compatibility](#)

1.1 NXP Smartphone Quick-Jack Solution

Inspired by the University of Michigan's Project HiJack, the NXP Smartphone Quick-Jack Solution repurposes the standard 3.5 mm stereo audio jack found on most smartphones into a self-powered data channel that makes communication with these smartphones as easy as plugging a headset jack into the audio port.

The Quick-Jack demo board integrates a joystick, temperature sensor, LEDs and an expansion header. The app running on the smartphone is able to interface with these on-board peripherals via the Quick-Jack interface.

Quick-Jack's main features:

- Demo board uses the low-power LPC812 microcontroller in TSSOP20 package
- Board is powered by either the smartphone or a small button battery
- SE98 I²C temperature sensor integrated on the demo board
- Control the four on-board LEDs from the smartphone
- Read input from on-board joystick
- Quick-Jack app available for iOS and Android smartphones
- Expansion header featuring most of the LPC800 GPIO pins
- Board features standard ARM SWD debug interface (10 pins, 1.27 mm)

The NXP Smartphone Quick-Jack Solution has been tested and verified to work with the iPhone 4, iPhone 4S, iPhone 5, iPhone 5S and the Samsung Galaxy S3 smartphones.

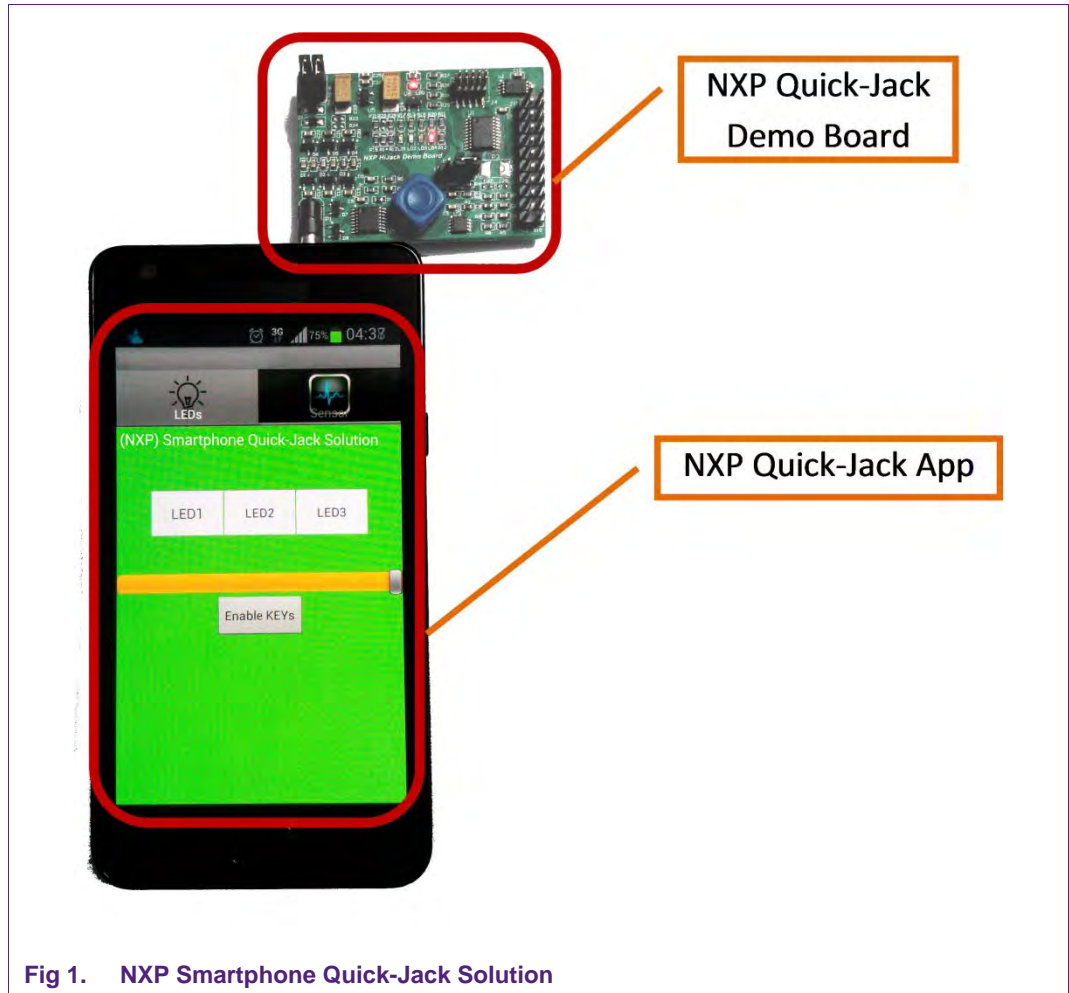


Fig 1. NXP Smartphone Quick-Jack Solution

1.2 LPC800

The LPC800 series of microcontrollers are an ARM Cortex-M0+ based, low-cost 32-bit MCU operating at CPU frequencies of up to 30 MHz. The LPC800 supports up to 16 kB of flash memory and 4 kB of SRAM.

The peripheral complement of the LPC800 includes a CRC engine, one I²C-bus interface, up to three USARTs, up to two SPI interfaces, one multi-rate timer, self wake-up timer, and state-configurable timer, one comparator, function-configurable I/O ports through a switch matrix, an input pattern match engine, and up to 18 general-purpose I/O pins.

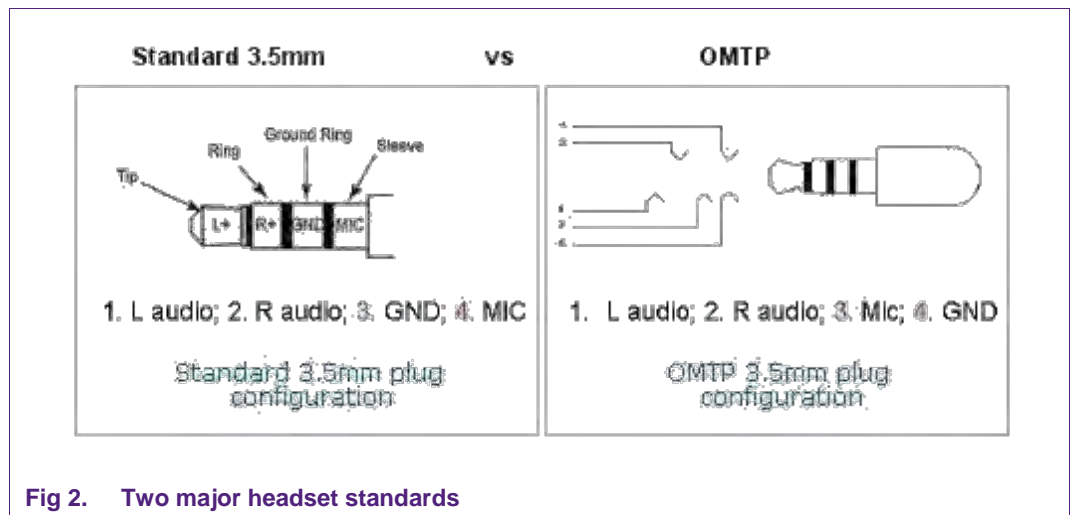
Due to the low power consumption of the LPC800, the LPC800 is an ideal candidate for the Smartphone Quick-Jack Solution.

2. Quick-Jack fundamentals

The Smartphone Quick-Jack Solution allows communication between a smartphone and the LPC800 microcontroller using the smartphone’s standard 3.5 mm stereo audio jack. This chapter explains the physical interface and the line-code used by Quick-Jack.

2.1 Physical interface

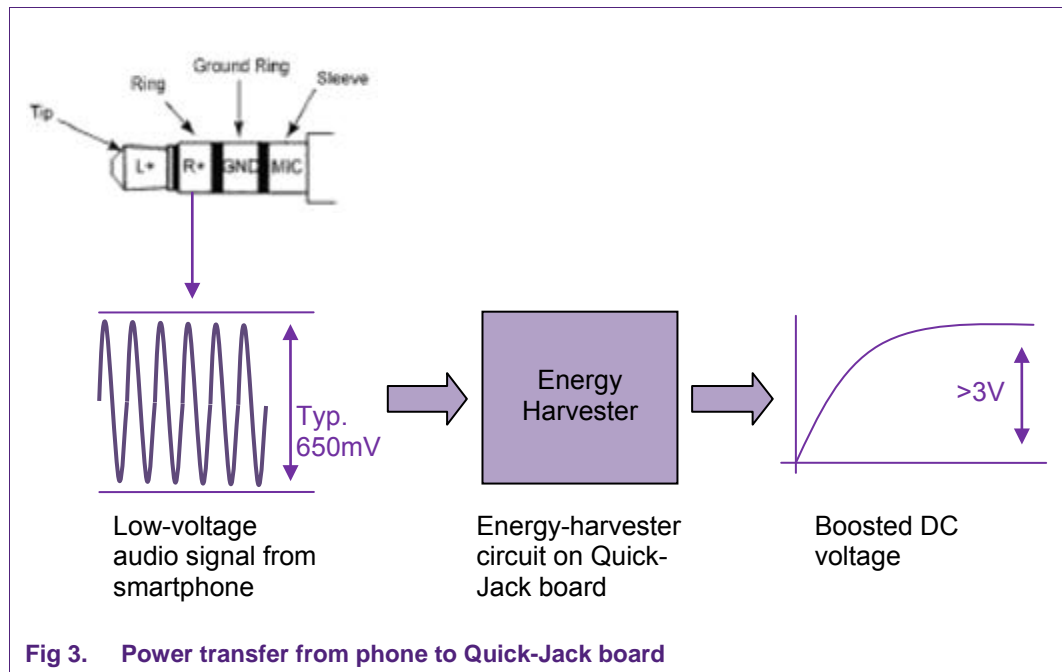
There are two major headset standards used in smartphones, OMTP and CTIA. The main difference between these two standards is which pin carries the MIC and GND signals, as shown in Fig 2. The NXP Smartphone Quick-Jack Solution is compatible with both standards. The hardware identifies the type of headset automatically and configures the hardware accordingly.



2.2 Power transfer

The Quick-Jack board must be able to operate solely from the power supplied by the smartphone.

To facilitate this feature, one of the two audio channels is dedicated to power transfer; a constant stream of audio feeds an energy-harvester circuit on the Quick-Jack board. This energy-harvester circuit boosts the low-voltage AC signal from the audio-jack to a DC-voltage suitable for digital circuitry.



2.3 Manchester line code

The NXP Smartphone Quick-Jack Solution uses Manchester coding as line code to achieve the communication between the smartphone and the LPC800 microcontroller.

Manchester code always has a transition at the middle of each bit period and may (depending on the information to be transmitted) have a transition at the start of the period also. The direction of the mid-bit transition indicates the data. Transitions at the period boundaries do not carry information. They exist only to place the signal in the correct state to allow the mid-bit transition. The existence of guaranteed transitions allows the signal to be self-clocking, and also allows the receiver to align correctly; the receiver can identify if it is misaligned by half a bit period, as there will no longer always be a transition during each bit period.

Manchester coding main features:

- Each bit is transmitted in a fixed time (the "period").
- A 0 is expressed by a low-to-high transition, a 1 by high-to-low transition (according to G.E. Thomas' convention—in the IEEE 802.3 convention, the reverse is true). NXP Quick-Jack uses the IEEE 802.3 convention.
- The transitions which signify 0 or 1 occur at the midpoint of a period.
- Transitions at the start of a period are overhead and don't signify data.

[Fig 4](#) shows a graphical representation of the Manchester line coding.

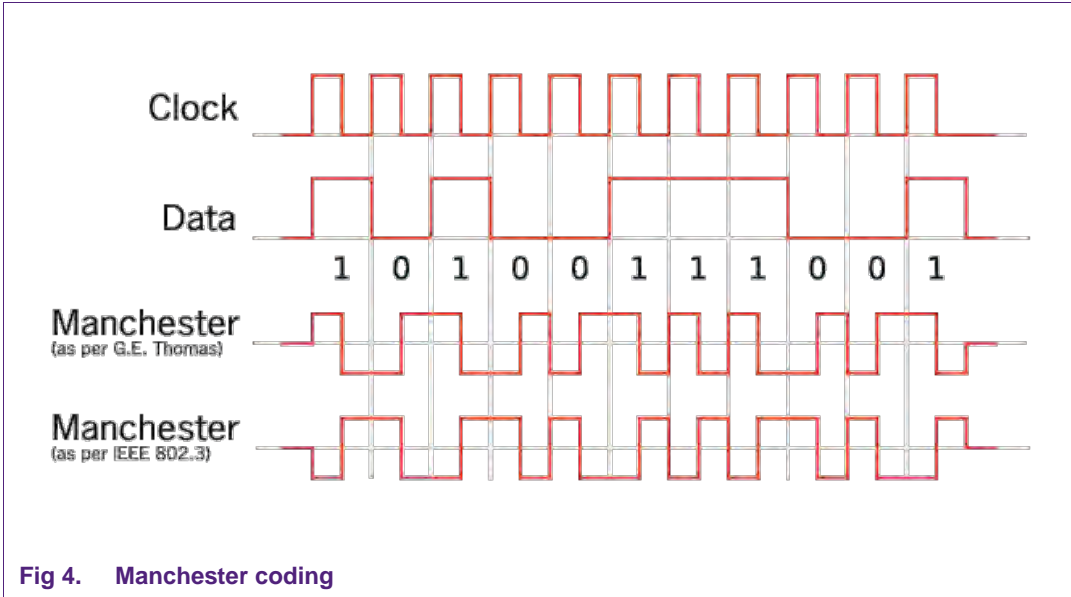


Fig 4. Manchester coding

3. Quick-Jack hardware description

This chapter describes the hardware used in the Smartphone Quick-Jack Solution in more detail. In particular, the following items will be discussed:

- [Hardware block diagram](#)
- [Quick-Jack schematic description](#)

3.1 Hardware block diagram

[Fig 5](#) shows the hardware block diagram of the NXP Smartphone Quick-Jack Solution.

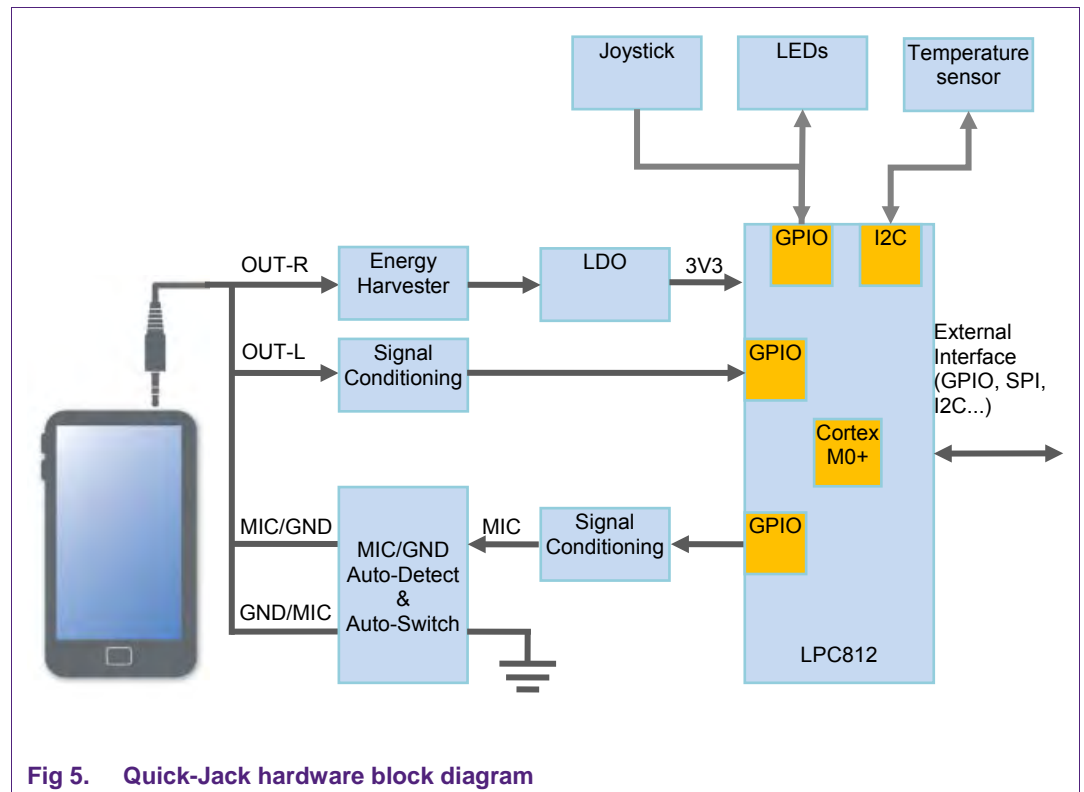


Fig 5. Quick-Jack hardware block diagram

3.2 Quick-Jack schematic description

The Quick-Jack hardware schematic can roughly be divided down into the following sections:

- [Power supply](#)
- [Audio jack MIC/GND auto-connect circuit](#)
- [Audio communication circuit](#)
- [LPC812 microcontroller](#)
- [I/O devices \(LEDs, joystick and temperature sensor\)](#)
- [Connectors \(jack-plug, SWD, expansion header\)](#)

Each of these sections will be briefly discussed in the following paragraphs.

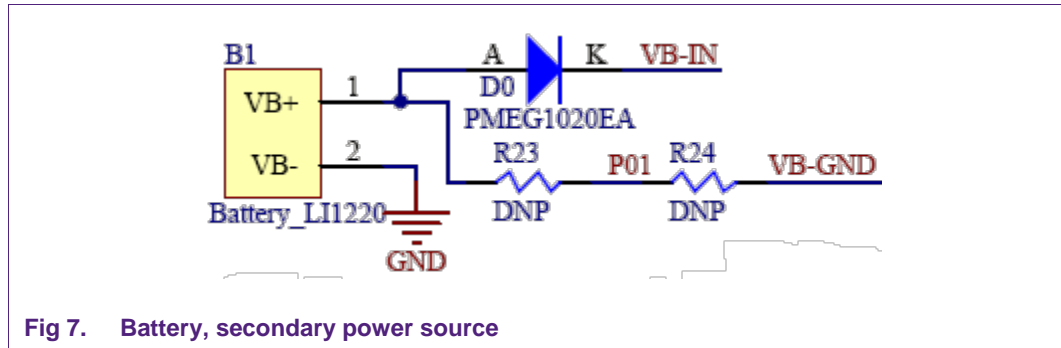


Fig 7. Battery, secondary power source

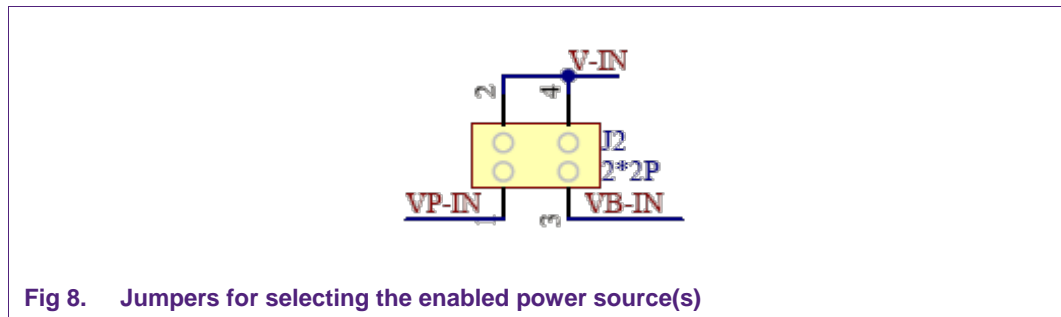


Fig 8. Jumpers for selecting the enabled power source(s)

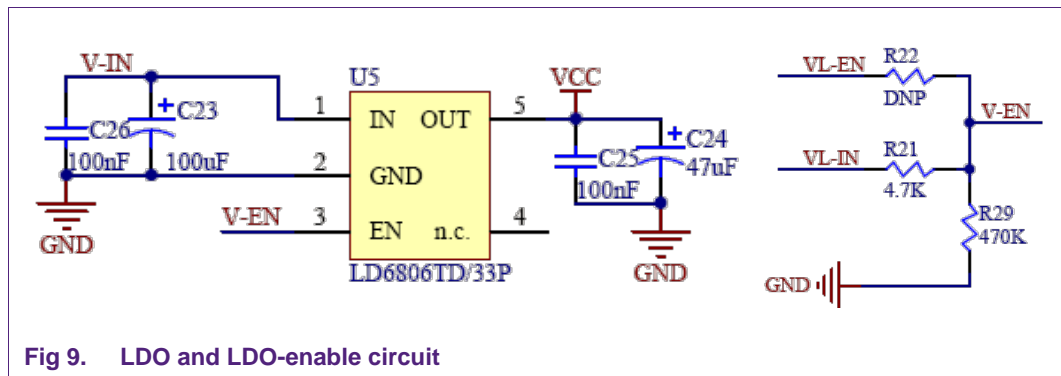


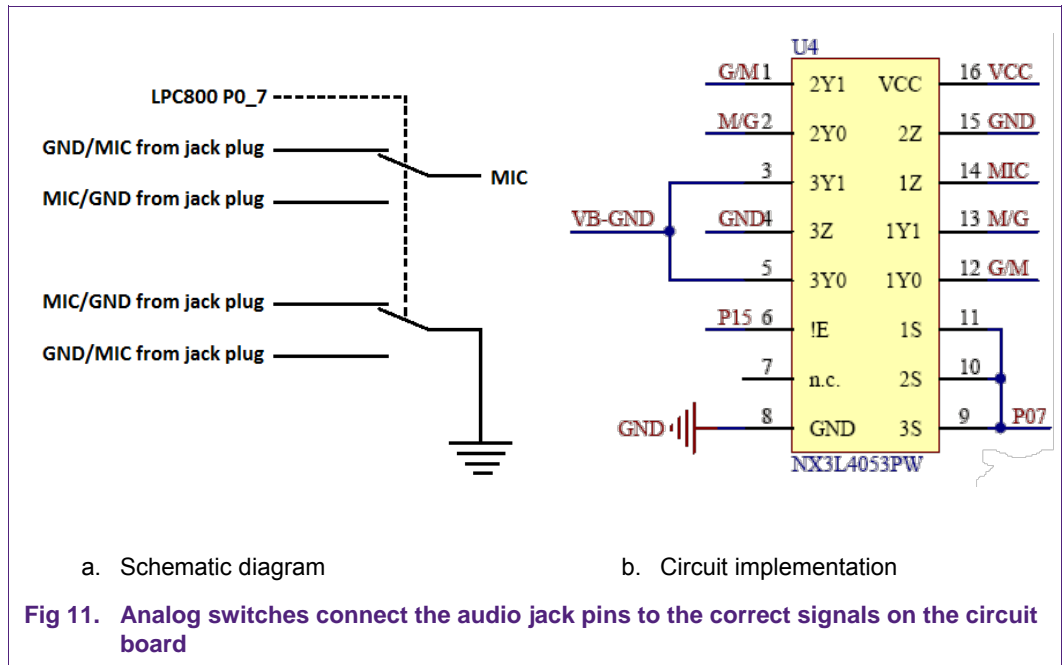
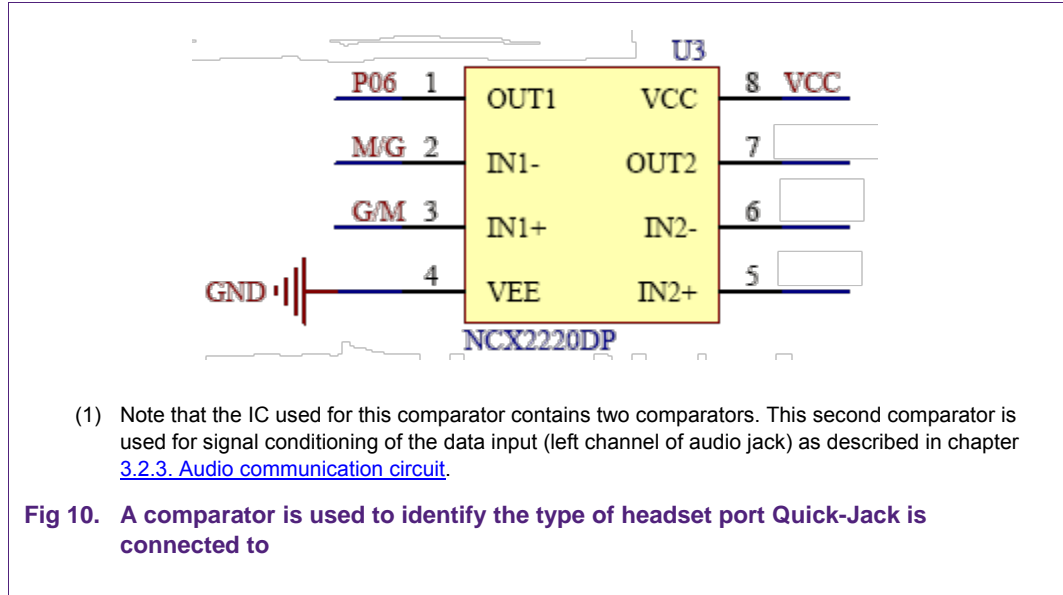
Fig 9. LDO and LDO-enable circuit

3.2.2 Audio jack MIC/GND auto-connect circuit

As mentioned in chapter [2.1. Physical interface](#), there are two major headset standards used in smartphones. The main difference between these two standards is the location of the MIC and GND signals on the jack plug.

To be able to support both standards automatically, a two-stage circuit is used:

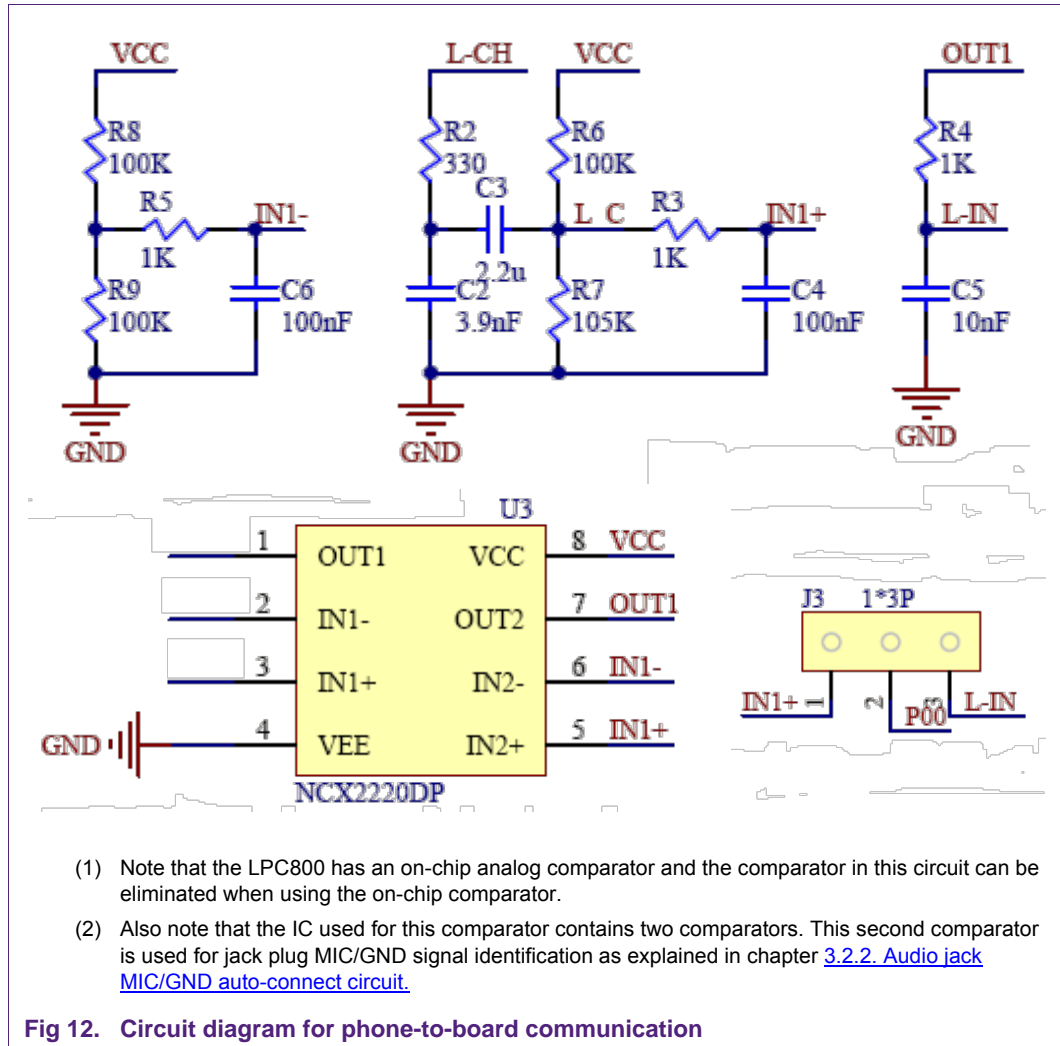
- First, a comparator circuit detects the type of headset port Quick-Jack has been inserted into. The implemented circuit is shown [Fig 10](#).
- The result is interpreted by the LPC800, which then configures an analog switch accordingly. The analog switch connects the right pin of the jack plug to the right signal on the circuit board (GND/MIC). The implemented circuit is shown in [Fig 11](#).



3.2.3 Audio communication circuit

Quick-Jack uses the phone’s left audio-channel as data output and the MIC as data input. A simple circuit is required to be able to adapt the analog audio signals to the microcontroller’s digital signals.

The circuit diagram implemented to allow communication from the phone to Quick-Jack is shown in Fig 12.



The purpose of this schematic is to show the logic that filters and digitizes the analog signal. The inverting input of comparator is connected to $\frac{1}{2}V_{CC}$, and the non-inverting input is connected to the filtered input signal. This causes the comparator's output to reflect a digital representation of the analog input signal.

Pin P0_0 is of the LPC800 takes care of reading the audio input. Jumper J3 allows P0_0 to be connected to either the comparator's output signal (label *L-IN*) or the filtered analog signal (label *IN+*). It is connected to the comparator's output by default.

Since the LPC800 has an on-chip analog comparator, the external comparator U3 can be eliminated from the hardware. The Quick-Jack hardware supports bypassing of the external comparator U3 using jumper J3. However, this mode is currently not supported in the LPC800 firmware. Refer to chapter [6.4. Jumper settings](#) for the location of jumper J3 on the Quick-Jack board.

The circuit diagram implemented to allow communication from Quick-Jack to the phone is shown in [Fig 13](#).

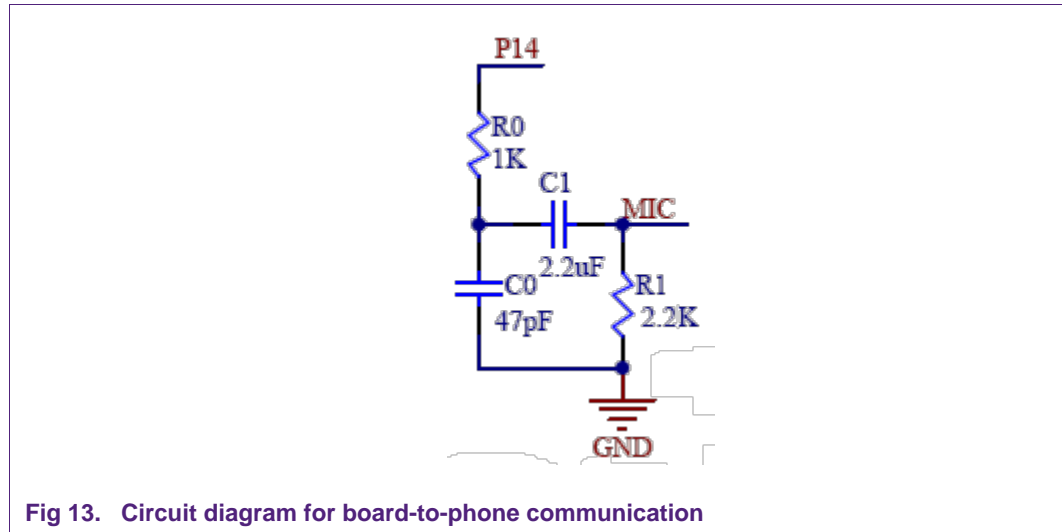


Fig 13. Circuit diagram for board-to-phone communication

The circuit simply filters the digital signal using R0 and C0 and removes any DC offset using C1 and R1.

3.2.4 LPC812 microcontroller

The low-power ARM Cortex-M0+ LPC812 microcontroller handles all the hardware interfacing and software protocol handling, thereby enabling Quick-Jack to communicate with the smartphone (Fig 14). The LPC812 Quick-Jack firmware is configured to use the 12 MHz Internal RC (IRC) oscillator as clock source, though the footprints for the required components to use an external crystal are available on the PCB.

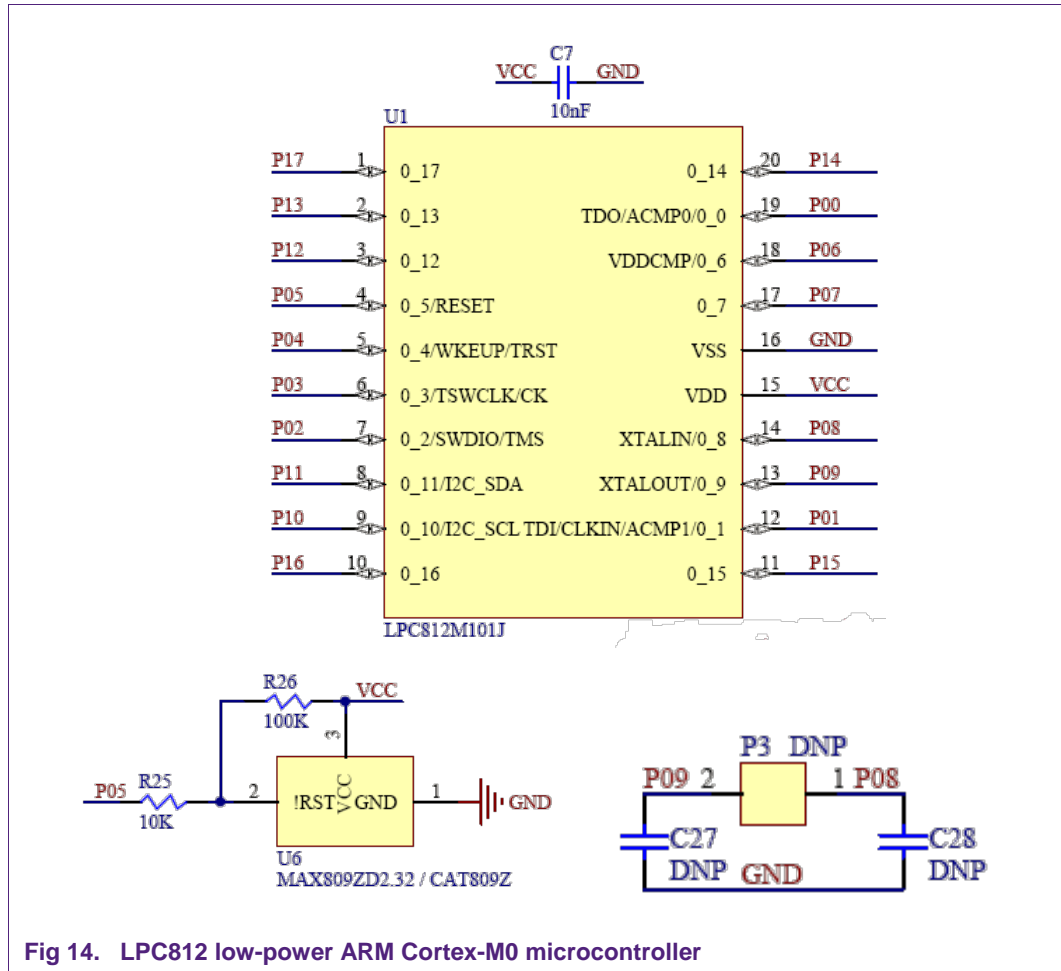


Fig 14. LPC812 low-power ARM Cortex-M0 microcontroller

3.2.5 I/O devices (LEDs, joystick and temperature sensor)

The Quick-Jack board comes with a few on-board devices ([Fig 15](#)) which can be controlled from (or read-out with) the smartphone app:

- Five LEDs as output devices. LD0 indicates power (after the LDO) is present, LD[1..4] can be controlled from the smartphone.
- As first input device, Quick-Jack features a joystick. The state of the on-board joystick is reflected in the app running on the smartphone.
- As second input device, Quick-Jack features the SE98 I²C temperature sensor.

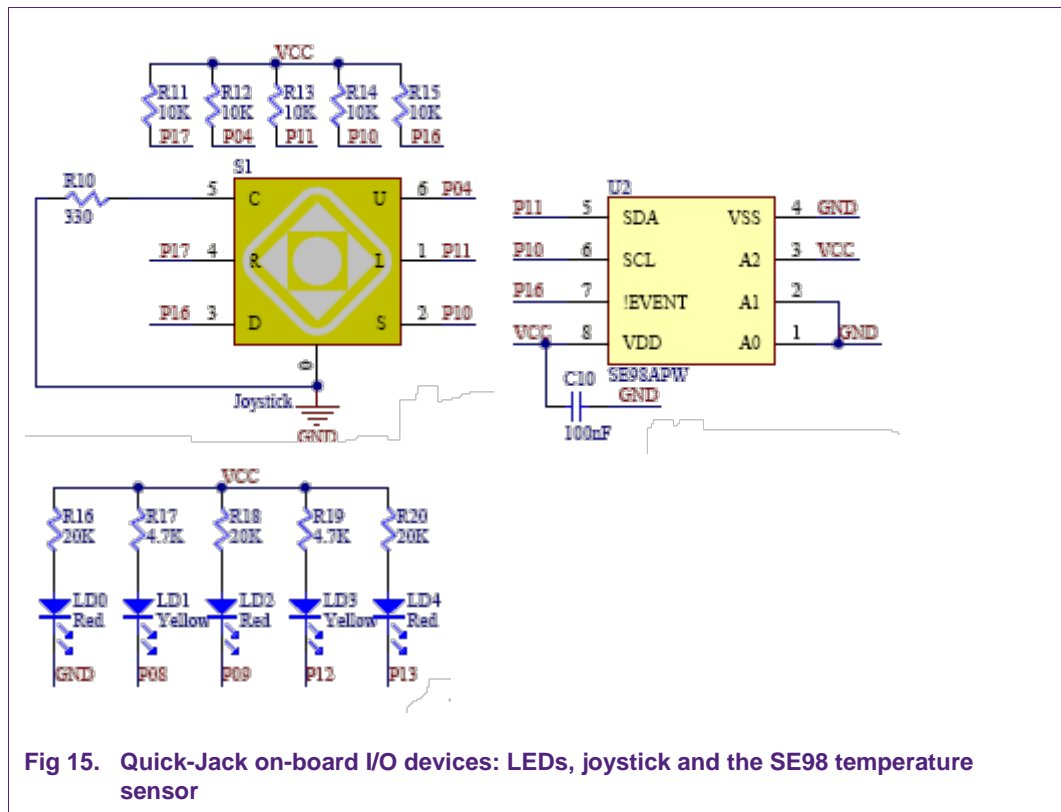


Fig 15. Quick-Jack on-board I/O devices: LEDs, joystick and the SE98 temperature sensor

3.2.6 Connectors (jack-plug, SWD, expansion header)

Three connectors can be found on the Quick-Jack board (Fig 16):

- Jack-plug for connecting Quick-Jack to the smartphone. To protect Quick-Jack from ESD, all of the jack-plug pins have ESD-protection diodes.
- SWD for programming the LPC812.
- 20-pin expansion connector. The 20-pin expansion connector allows connecting external devices (e.g. sensor, I/O boards) to the LPC812 and interfacing with the smartphone.

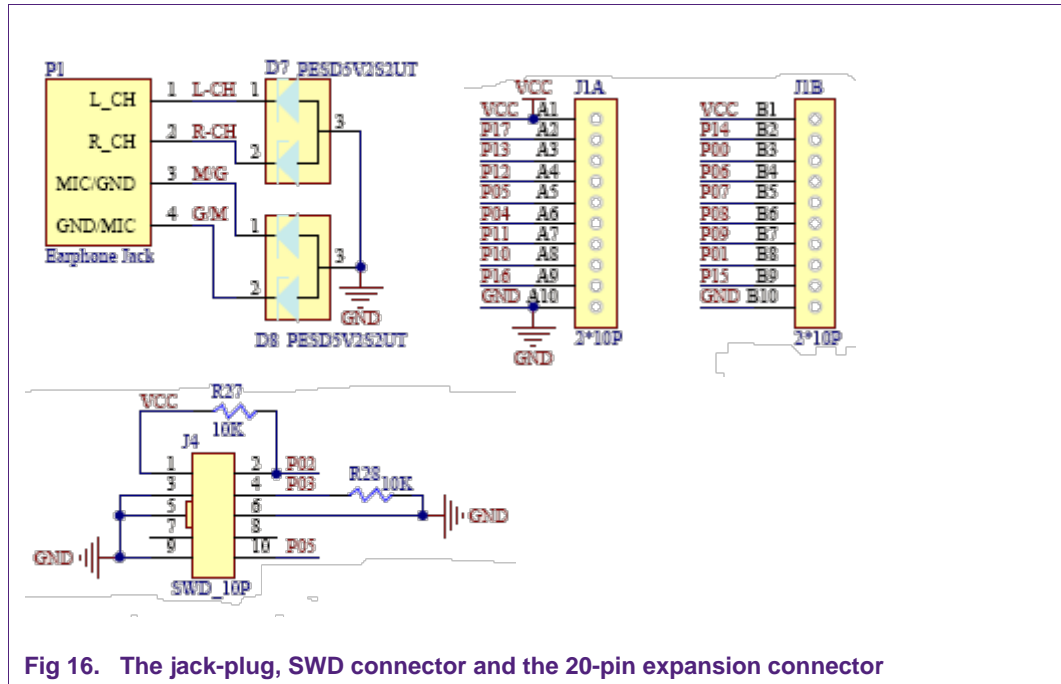


Fig 16. The jack-plug, SWD connector and the 20-pin expansion connector

4. Quick-Jack software description

The Smartphone Quick-Jack Solution requires two software programs:

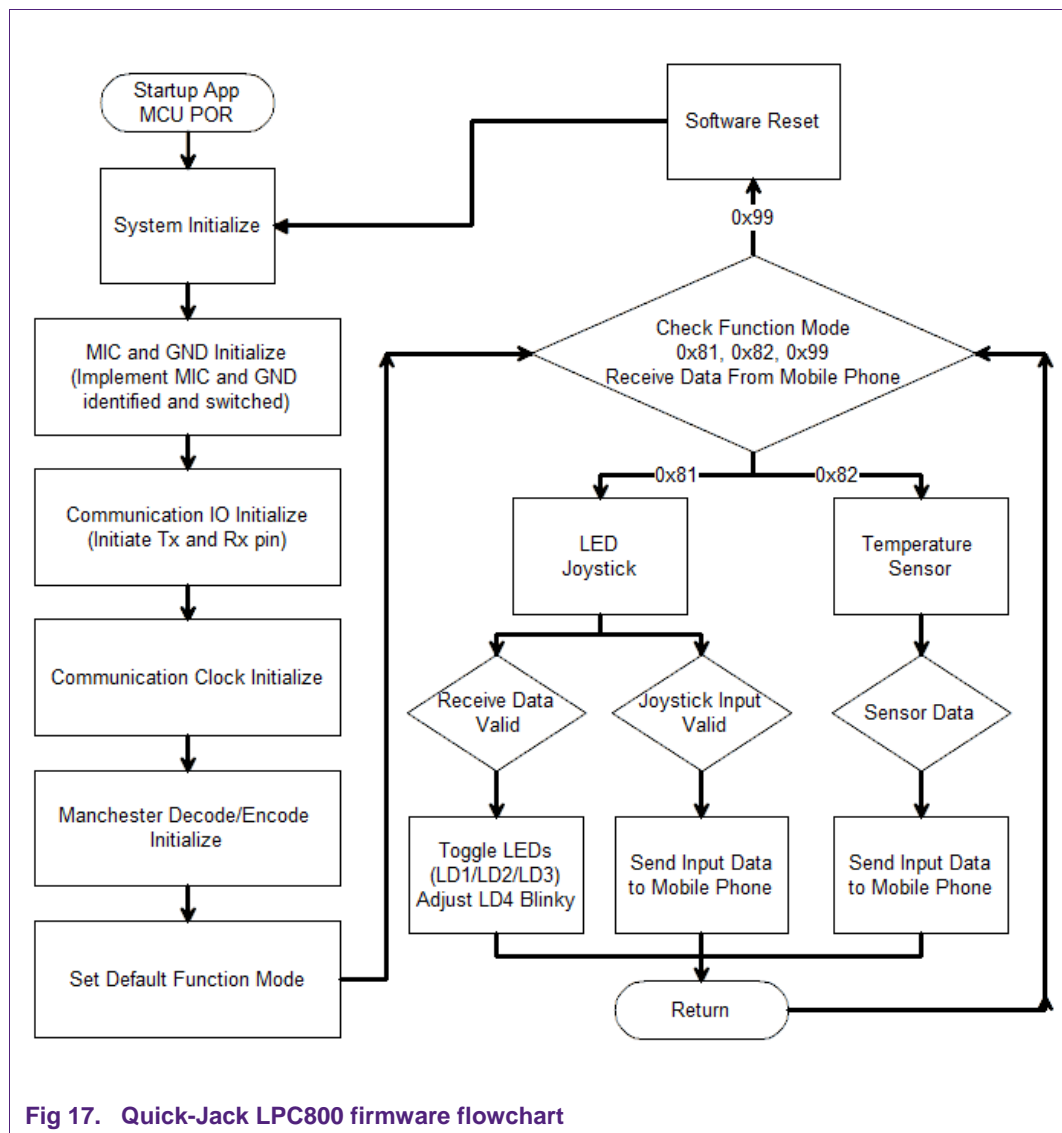
- The app running on the smartphone
- The embedded firmware running on the LPC800 microcontroller

The source code for LPC800 and Android is freely available for download on <http://www.lpcware.com/quick-jack>.

A high-level description of the software will be given in this chapter. For more details, please refer to the actual source code.

4.1 LPC800 firmware flowchart

The high-level flowchart of the LPC800 firmware is shown in Fig 17.



4.2 Smartphone app flowchart

The high-level flowchart of the smartphone app is shown in [Fig 18](#).

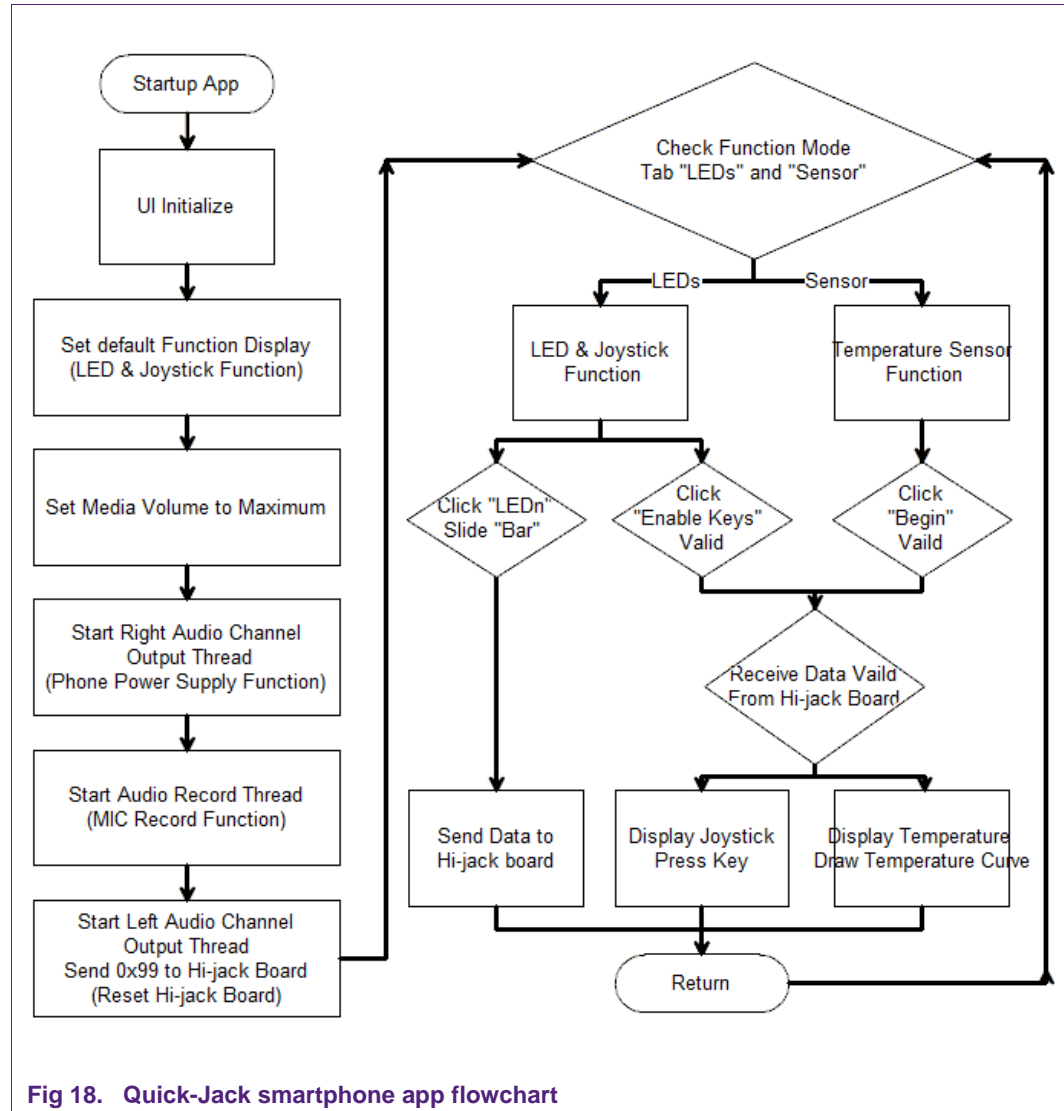


Fig 18. Quick-Jack smartphone app flowchart

5. Measurements and waveforms

To get a better understanding on how Quick-Jack works, this chapter shows a number of waveforms measured on the Quick-Jack board during operation.

Following waveforms will be shown and explained:

- [Quick-Jack power transfer](#)
- [Data communication \(Phone → Quick-Jack board\)](#)
- [Data communication \(Quick-Jack board → phone\)](#)

5.1 Quick-Jack power transfer

As mentioned in [chapter 2.2. Power transfer](#), the Quick-Jack board is powered by the smartphone.

The phone's right audio-channel is used to transfer the actual power. A plain 21 kHz sine wave is used as carrier. The energy-harvester circuit boosts the amplitude of the sine wave and rectifies it to a DC voltage.

[Fig 20](#) shows the waveform of the right audio-channel (oscilloscope Ch1 in yellow) and the boosted DC voltage (oscilloscope Ch2 in blue).

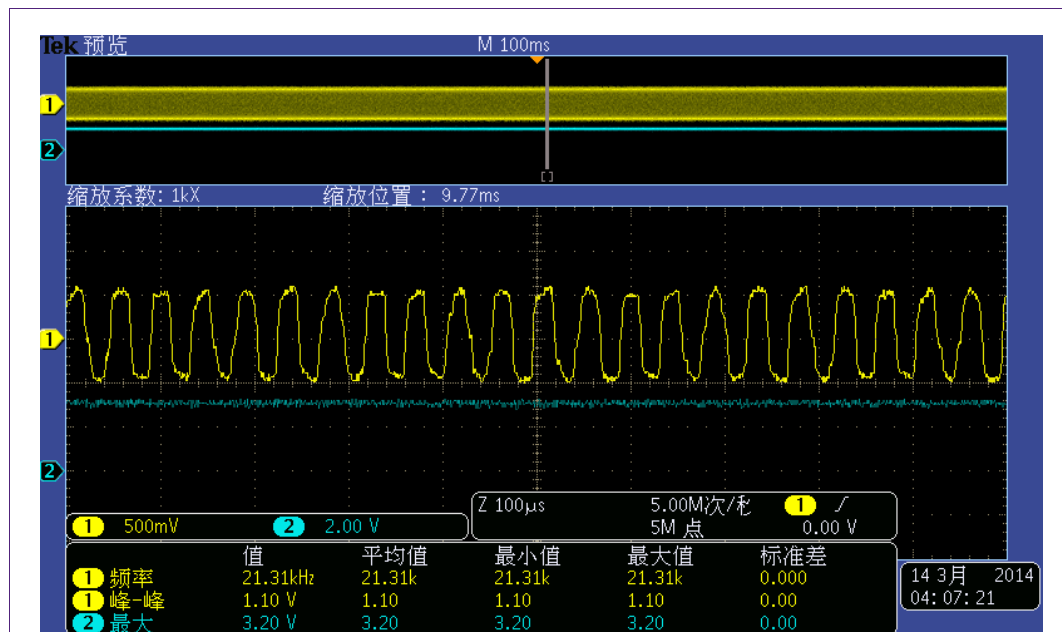


Fig 19. Power transfer to Quick-Jack board using the right audio-channel of the smartphone.

5.2 Data communication (Phone → Quick-Jack board)

The left audio-channel is used to transfer data from the phone to the Quick-Jack board. The data-payload is encapsulated to form a small packet, and Manchester line coding is used to encode the bits.

On the Quick-Jack board, a comparator is used to convert the analog data into digital data. The LPC800 then decodes the Manchester-encoded bitstream and continues processing the data-payload.

Fig 20 shows the waveform of the left audio-channel (oscilloscope Ch1 in yellow) and how that signal looks like after the comparator (oscilloscope Ch2 in blue). The waveform includes 3 idle-bits ('1'), 1 start-bit ('0'), 8 data-bits, 1 Parity-bit, 1 stop-bit ('1') and 3 idle-bits ('1'). The data-payload send to the Quick-Jack board is the byte 0x72.

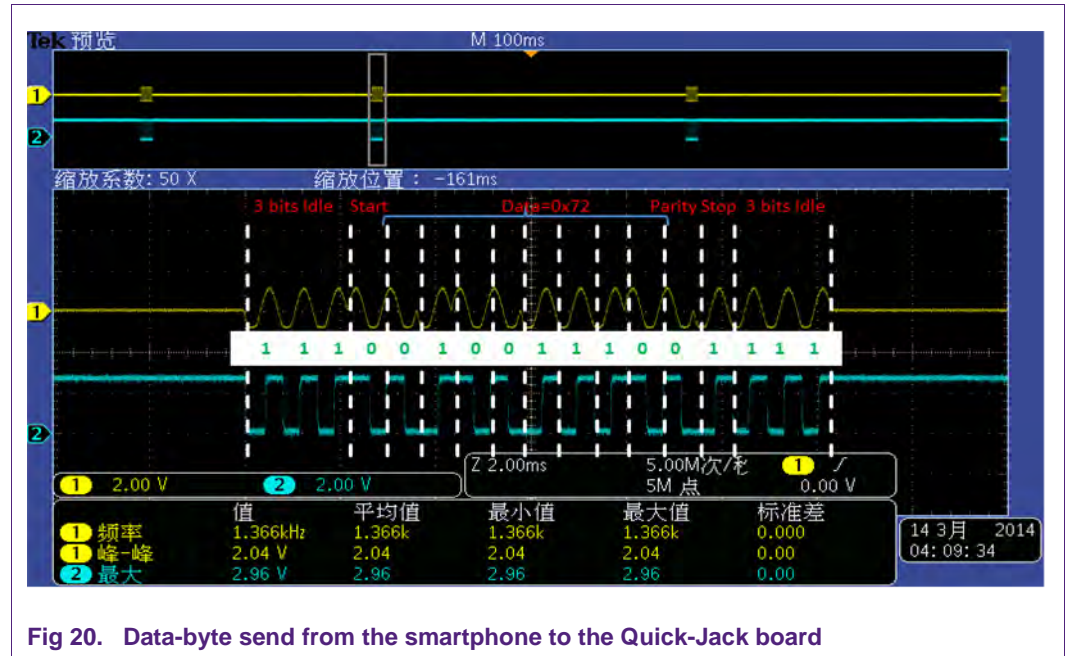


Fig 20. Data-byte send from the smartphone to the Quick-Jack board

5.3 Data communication (Quick-Jack board → phone)

Data communication from the Quick-Jack board to the phone is very similar to the other way around as described above.

First, the Quick-Jack board encapsulated the data-payload and converts it to a Manchester encoded bitstream. This bitstream is shifted out on one of the LPC800's GPIO pins. A simple RC filter removes any DC-offset. The smartphone's MIC signal is connected to the output of this filter.

The app running on the phone performs edge detection, then decodes the Manchester-encoded bitstream and continues processing the data-payload.

Fig 21 shows the waveform of the LPC800's GPIO pin shifting out the bitstream (oscilloscope Ch2 in blue) and how that signal looks like after the RC filter (oscilloscope Ch1 in yellow). The waveform includes three idle-bits ('1'), one start-bit ('0'), eight data-bits, one parity-bit, one stop-bit ('1') and three idle-bits ('1'). The data-payload sent to the phone is the byte 0x18.

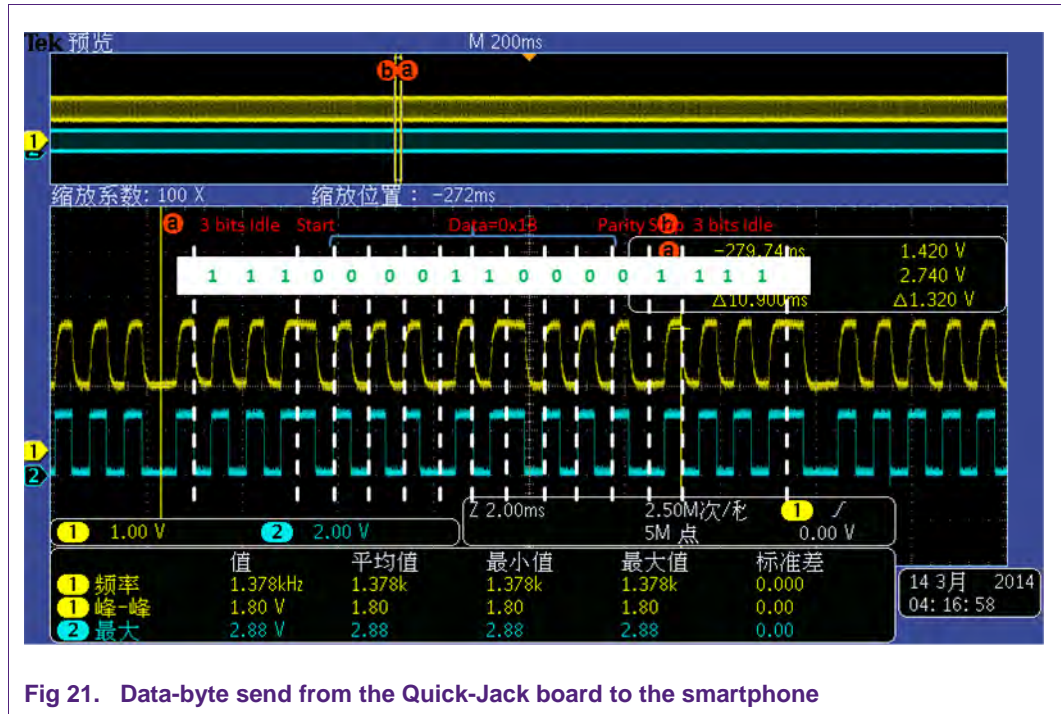


Fig 21. Data-byte send from the Quick-Jack board to the smartphone

6. NXP Quick-Jack Quick Start Guide

Using the NXP Smartphone Quick-Jack Solution is easy and straightforward.

This chapter explains how to get up and running with the Smartphone Quick-Jack Solution.

6.1 Hardware requirements

The Smartphone Quick-Jack Solution requires the NXP Quick-Jack board (OM13069) and a compatible smartphone.

The Quick-Jack board can be ordered through a variety of distributors using order number OM13069.

Quick-Jack has been tested and verified to work on the following smartphones:

- iOS: iPhone 4, iPhone 4S, iPhone 5, iPhone 5S (iOS 6.1.6 and iOS 7.0.4).
- Android: Samsung Galaxy S3.

Read more about Quick-Jack smartphone compatibility in chapter 8. [Quick-Jack smartphone compatibility](#)

6.2 Software requirements

The Quick-Jack board ships pre-programmed. The latest version of the firmware, including the source code, can be found on <http://www.lpcware.com/quick-jack>. Refer to chapter 7. [Building and flashing the LPC812 firmware](#) for details on how to build and flash firmware.

The smartphone example app can be downloaded from the *App Store* (iOS). The Android App (APK) and source code can be downloaded from <http://www.lpcware.com/quick-jack>.

6.3 Getting up and running

To get up and running with Quick-Jack, follow these steps:

1. Download and install the app. The app can be downloaded from the *App Store* (iOS). The app can be found in the app-store by searching for 'NXP Quick-Jack'.
2. Make sure the coin cell battery (size CR1220) is inserted and that the jumpers on the Quick-Jack board are set to default ([Fig 22](#)).

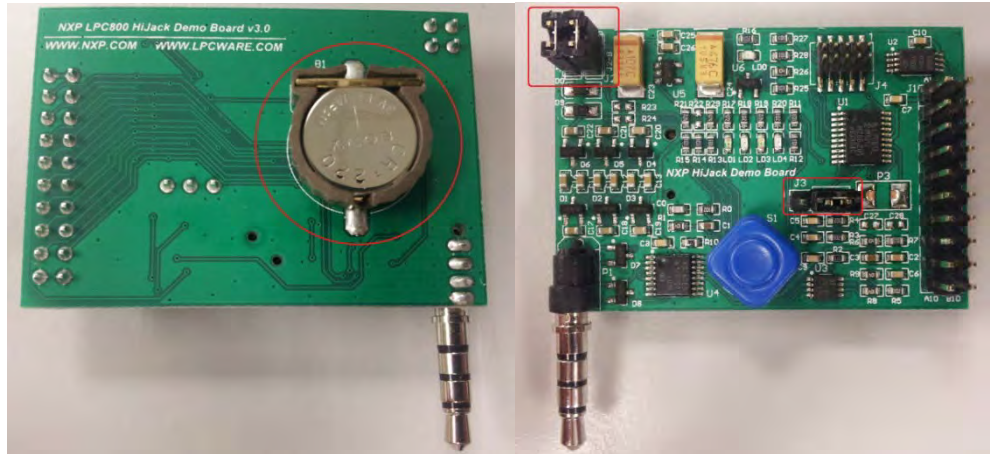


Fig 22. Coin cell battery inserted, jumpers set to default configuration (JP2 on both positions, JP3 placed on the right position)

3. Insert the Quick-Jack board in the phone's audio jack. Then, configure the media-volume/headset-volume on the phone to maximum volume (Fig 23).

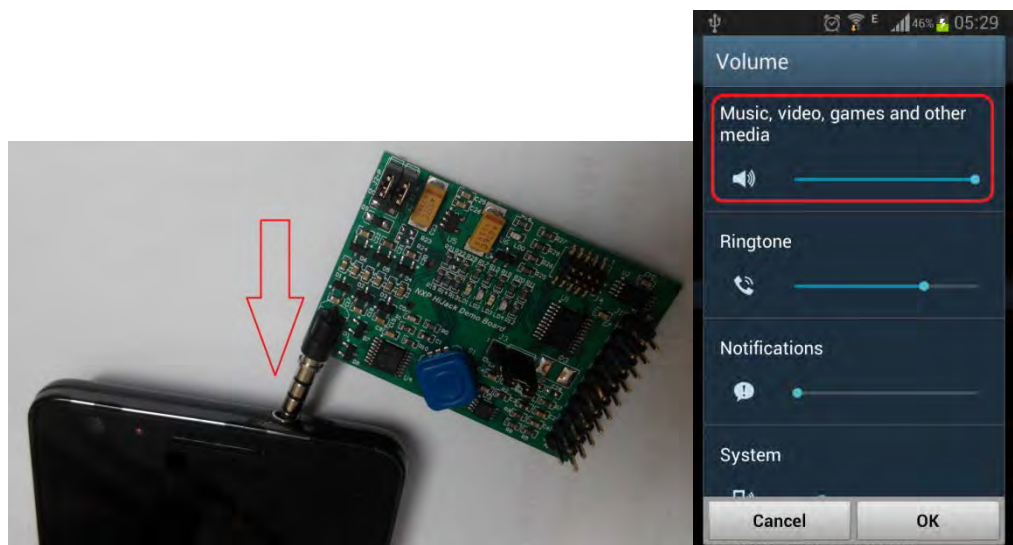


Fig 23. Phone's media-volume/headset-volume set to maximum

4. Start the app. As soon as the app is running, LED LD0 should be lit and LED LD4 should be blinking (Fig 24).



Fig 24. After starting the Quick-Jack app, LD0 will be lit and LD4 will be blinking

After establishing initial connection (LD4 blinking), the Quick-Jack board can be controlled from the app:

- By adjusting the slider, the blink rate of LD4 can be changed (faster/slower).
- By clicking the *LED1/LED2/LED3* buttons, on-board LEDs LD1 to LD3 can be toggled on and off ([Fig 25](#)).

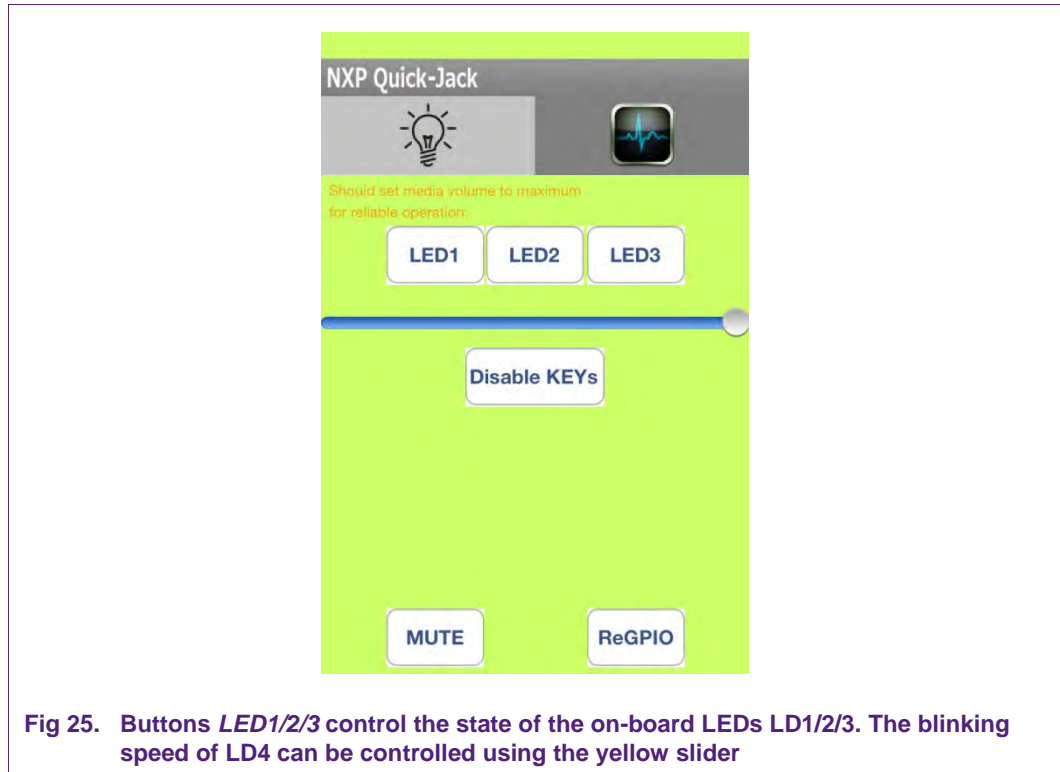


Fig 25. Buttons *LED1/2/3* control the state of the on-board LEDs LD1/2/3. The blinking speed of LD4 can be controlled using the yellow slider

- The state of the on-board joystick is reflected on the circle in the app ([Fig 26](#)).

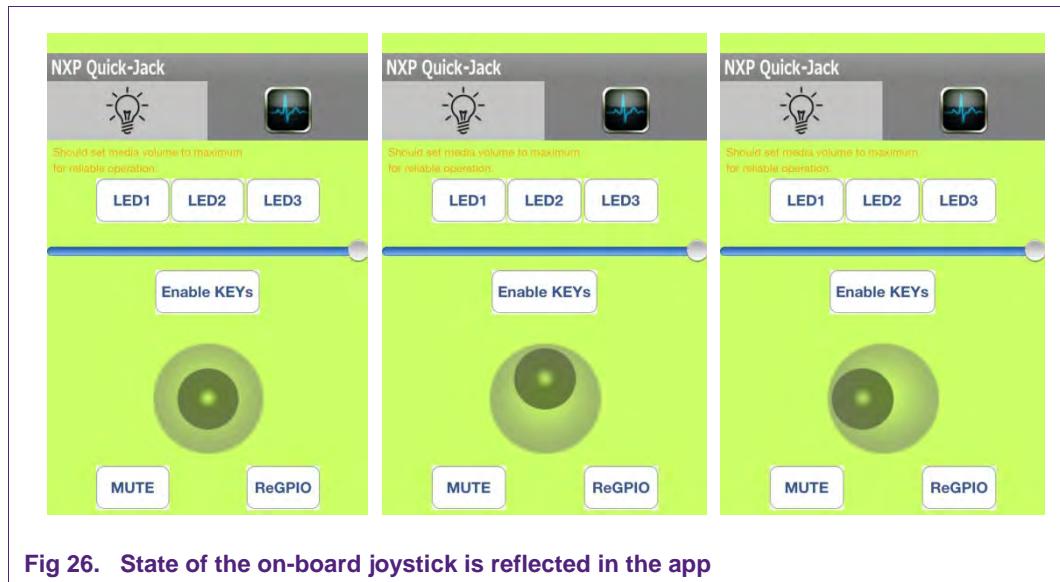


Fig 26. State of the on-board joystick is reflected in the app

- Clicking on the sensor tab (top right in the app) shows the temperature sensor interface. After clicking the *Begin* button, the current temperature as measured by the on-board SE98 sensor will be displayed and a temperature vs. time graph will be plotted ([Fig 27](#)). Note that when the sensor tab is opened, LD4 stops blinking, when returning to the LEDs tab, LD4 resumes blinking.

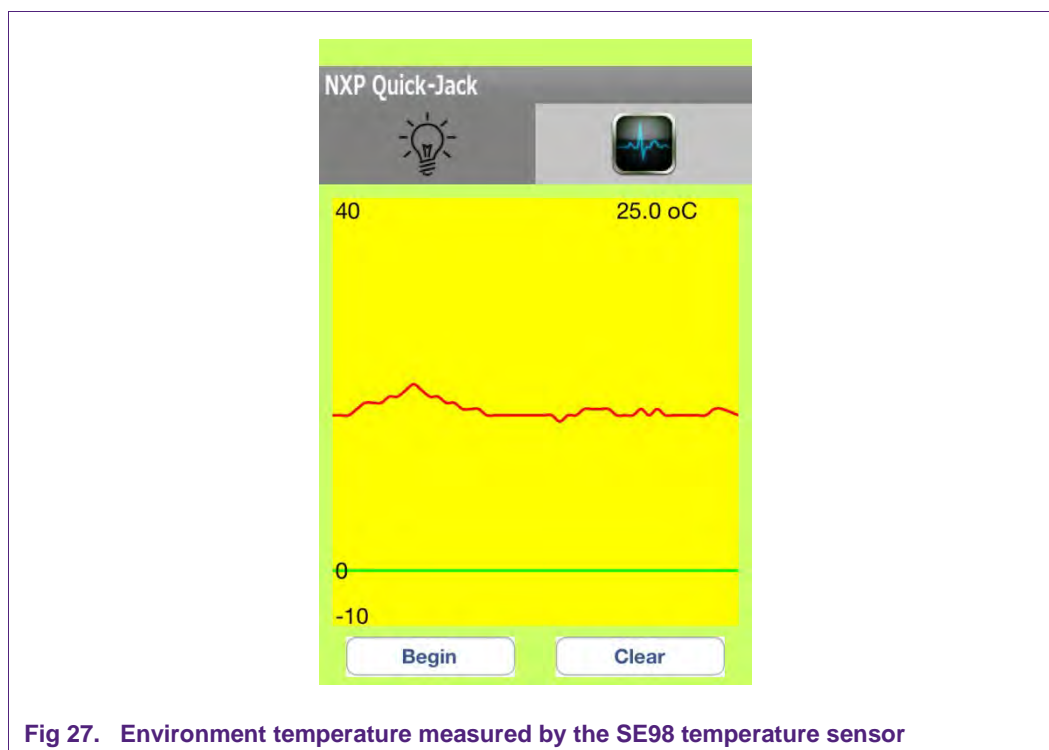


Fig 27. Environment temperature measured by the SE98 temperature sensor

- Press the phone's Return button (Android) or the phone's Home button (iOS) to exit the app.

6.4 Jumper settings

Two jumper headers are available on the Quick-Jack board, jumpers JP2 and JP3 (Fig 28):

- **JP2 (Power) - Power supply select.** Four-pin header JP2 allows placing two jumpers. The left (outer) jumper connects the energy-harvester to the input of the LDO; the right (inner) jumper connects the coin cell battery to the input of the LDO. Diodes prevent current flowing from one source into the other, allowing Quick-Jack to be powered by both the battery and the energy-harvester at the same time. This operation is advised when using Quick-Jack and is the default setting.
- **JP3 (Communication) – Left-channel audio signal path select.** This jumper can be positioned in either the left- or right position. When placed on the right two pins, the phone's left-channel audio signal – the signal with the Manchester encoded data – is routed through the on-board comparator (default mode).
- When placed on the left two pins, the on-board comparator is bypassed and the analog signal is fed straight into the LPC812. In this case, the LPC812's internal comparator must be enabled and configured in order to convert the phone's analog data to digital data. Note that this mode is not supported in the default firmware.

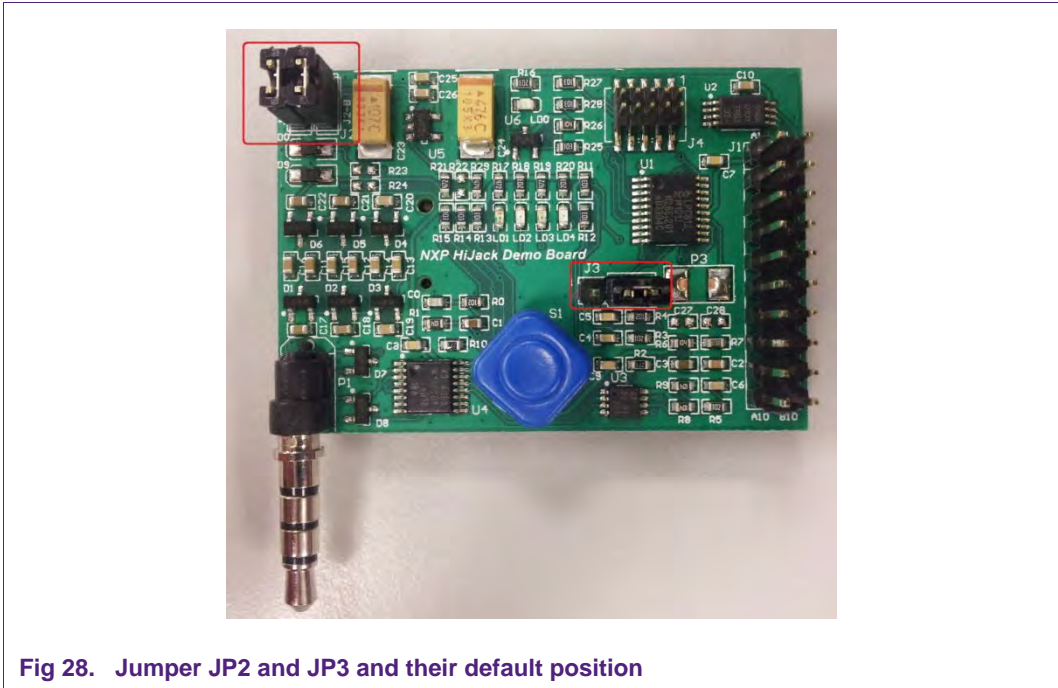


Fig 28. Jumper JP2 and JP3 and their default position

7. Building and flashing the LPC812 firmware

This chapter provides details on how to compile the LPC812 source code and how to flash the resulting binary into the LPC812.

Note: The Quick-Jack board comes pre-programmed. Building and flashing the firmware is only necessary when you plan on changing the firmware.

7.1 Building the firmware

The LPC812 Quick-Jack firmware (based on the LPC800 LPCOpen) can be downloaded from <http://www.lpcware.com/quick-jack>. The source-code and projects for LPCXpresso, Keil μ Vision and IAR EWARM are included in the download.

Refer to <http://www.lpcware.com/lpcopen> for details on how to build the software.

7.2 Flashing the firmware

After building the firmware, the resulting binary can be flashed by connecting a SWD probe (e.g. LPC-Link2, ULINK, J-Link) to the SWD connector J4 ([Fig 16](#)).

8. Quick-Jack smartphone compatibility

Quick-Jack has been developed for both the iOS and the Android mobile operating systems.

Quick-Jack has been tested and verified to work on the following devices:

- iOS: iPhone 4, iPhone 4S, iPhone 5 and iPhone 5S (iOS 6.1.6 and iOS 7.0.4).
- Android: Samsung Galaxy S3.

Compatibility with other iOS/Android devices is not guaranteed. This chapter gives a short overview why there may be compatibility issues, and gives hints how these may be resolved.

8.1 iOS

The NXP Smartphone Quick-Jack Solution has been tested on several generations of iPhone devices, with no issues found. Since hardware among *iDevices* and software among iOS versions are typically compatible, it is expected that Quick-Jack works on the majority of *iDevices*.

8.2 Android

Many mobile-phone manufacturers use Android as a mobile operating system. The large number of manufacturers causes a huge spread among the hardware platforms running on Android; often even at a single manufacturer different hardware platforms are used for different Android phones.

Since the hardware platforms differ from phone to phone, many different software drivers exist to support all these hardware platforms. Besides software differences related to hardware, several different Android versions now exist and manufacturers usually modify the base android platform to their needs.

Due to these significant variations founds among Android phones, it is difficult for Android app-developers to ensure compatibility with all available Android phones. This is especially true for apps using low-level I/O and re-purposing the phone's hardware, like the Quick-Jack application does.

The Smartphone Quick-Jack Solution is a proof-of-concept. It is not intended as final product, but instead intended to demonstrate the low-power capabilities of the LPC800-series, and to showcase a possible application which benefits from this feature. Therefore, the Quick-Jack Android compatibility is limited to the Samsung Galaxy S3. Other Android phones may work, but certain properties of the current hardware/software are known to limit the number of supported Android phones. The Quick-Jack hardware/software may be enhanced to support more or other type of phones.

A number of limitations were identified which limits the number of Android phones Quick-Jack is compatible with. These limitations all apply to board-to-phone communication; phone-to-board communication is expected to work on a much large number of Android devices.

8.2.1 MIC signal impedance

As shown in the schematic ([Fig 13](#)), the MIC signal is emulated by one of the LPC800 GPIO pins combined with a simple circuit. The circuit has a pull-down resistor connected to the jack-plug's MIC signal.

For iOS devices, this resistor needs to be fairly small (~2 k Ω) for the phone to recognize Quick-Jack's emulated MIC as valid MIC. Unfortunately, this is known to cause problems on some Android phones which may require a higher impedance (e.g. >10 k Ω).

A possible fix for this issue is to either choose to only support one phone (low impedance) or the other (high impedance), or choose to be able to automatically change the impedance by e.g. automatically connecting or disconnecting an impedance from the MIC signal to GND.

8.2.2 Temporarily short-circuit on MIC signal

As explained in chapter [2.1. Physical interface](#), the pinout of the headset may differ depending on the type of phone. Quick-Jack has an on-board circuit which detects which type of headset is connected ([Fig 10](#)), which then configures an analog switch to connect the signals accordingly ([Fig 11](#)).

The detection circuit ([Fig 10](#)) uses a comparator to identify the GND and MIC pins by connecting these pins of the audio jack to the IN+ and IN- pins of the comparator. At normal conditions, the sensing circuit is non-invasive (i.e. it only measures and does not influence impedance/voltage/current on the MIC/GND jack pins significantly). However, when the comparator is not powered, the IN+/IN- pins of the comparator behave as if they are internally connected to diodes to the comparator's GND and VDD pins. Since Quick-Jack is powered by phone, Quick-Jack will always be inserted into the headset while it is not powered. This causes Quick-Jack to influence the electrical properties of the jackplug's MIC signal until the energy harvester has harvested enough energy to enable the LDO ([Fig 9](#)). Some phones (e.g. Samsung Galaxy S4) detect this upon inserting Quick-Jack, and instead of configuring the jack's MIC as default MIC, they switch back to the phone's internal MIC instead. This results in the Quick-Jack app not receiving the data from the Quick-Jack board, but the environment sound of the phone instead. This leads to failure of board-to-phone communication.

There are several ways to fix the issue on hardware level. Two proposed methods:

- Disconnect the comparator's IN+ and IN- pins from the jackplug and choose another way to perform the MIC/GND detection, or even choose to not auto-detect the type of headset and only support one of the standards.
- Use a series resistor (10 k Ω to 100 k Ω) to connect the jackplug's MIC/GND pins to the comparator's IN+/IN- pins.

9. Conclusion

The NXP Smartphone Quick-Jack Solution repurposes the standard 3.5 mm stereo audio jack found on most smartphones into a self-powered data channel that makes communication with these smartphones as easy as plugging a headset jack into the audio port. This allows for quick development of low-cost electronics which can read sensor data and use the phone's mobile connectivity to transfer this data to the cloud.

The NXP LPC800-series microcontroller is an excellent fit for this application due to its low power consumption.

Quick-Jack can be ordered from many of the NXP distributors. The Quick-Jack schematic, LPC800 firmware, and Android app source code are available free of cost on <http://www.lpcware.com/quick-jack>.

10. Legal information

10.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

10.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP

Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out of the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US\$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

10.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

11. List of figures

Fig 1.	NXP Smartphone Quick-Jack Solution	4	Fig 17.	Quick-Jack LPC800 firmware flowchart.....	17
Fig 2.	Two major headset standards.....	5	Fig 18.	Quick-Jack smartphone app flowchart	18
Fig 3.	Power transfer from phone to Quick-Jack board	6	Fig 19.	Power transfer to Quick-Jack board using the right audio-channel of the smartphone.....	19
Fig 4.	Manchester coding.....	7	Fig 20.	Data-byte send from the smartphone to the Quick-Jack board	20
Fig 5.	Quick-Jack hardware block diagram	8	Fig 21.	Data-byte send from the Quick-Jack board to the smartphone	21
Fig 6.	Diode voltage multiplier, primary power source	9	Fig 22.	Coin cell battery inserted, jumpers set to default configuration (JP2 on both positions, JP3 placed on the right position).....	23
Fig 7.	Battery, secondary power source.....	10	Fig 23.	Phone's media-volume/headset-volume set to maximum.....	23
Fig 8.	Jumpers for selecting the enabled power source(s)	10	Fig 24.	After starting the Quick-Jack app, LD0 will be lit and LD4 will be blinking.....	24
Fig 9.	LDO and LDO-enable circuit	10	Fig 25.	Buttons <i>LED1/2/3</i> control the state of the on-board LEDs LD1/2/3. The blinking speed of LD4 can be controlled using the yellow slider	25
Fig 10.	A comparator is used to identify the type of headset port Quick-Jack is connected to	11	Fig 26.	State of the on-board joystick is reflected in the app	25
Fig 11.	Analog switches connect the audio jack pins to the correct signals on the circuit board	11	Fig 27.	Environment temperature measured by the SE98 temperature sensor	26
Fig 12.	Circuit diagram for phone-to-board communication	12	Fig 28.	Jumper JP2 and JP3 and their default position	27
Fig 13.	Circuit diagram for board-to-phone communication	13			
Fig 14.	LPC812 low-power ARM Cortex-M0 microcontroller	14			
Fig 15.	Quick-Jack on-board I/O devices: LEDs, joystick and the SE98 temperature sensor	15			
Fig 16.	The jack-plug, SWD connector and the 20-pin expansion connector	16			

12. Contents

1.	Introduction	3	10.3	Trademarks	32
1.1	NXP Smartphone Quick-Jack Solution.....	3	11.	List of figures.....	33
1.2	LPC800	4	12.	Contents	34
2.	Quick-Jack fundamentals	5			
2.1	Physical interface	5			
2.2	Power transfer	5			
2.3	Manchester line code	6			
3.	Quick-Jack hardware description	8			
3.1	Hardware block diagram	8			
3.2	Quick-Jack schematic description	8			
3.2.1	Power supply.....	9			
3.2.2	Audio jack MIC/GND auto-connect circuit	10			
3.2.3	Audio communication circuit.....	11			
3.2.4	LPC812 microcontroller.....	13			
3.2.5	I/O devices (LEDs, joystick and temperature sensor)	14			
3.2.6	Connectors (jack-plug, SWD, expansion header)	15			
4.	Quick-Jack software description	17			
4.1	LPC800 firmware flowchart	17			
4.2	Smartphone app flowchart	18			
5.	Measurements and waveforms	19			
5.1	Quick-Jack power transfer.....	19			
5.2	Data communication (Phone → Quick-Jack board).....	19			
5.3	Data communication (Quick-Jack board → phone)	20			
6.	NXP Quick-Jack Quick Start Guide.....	22			
6.1	Hardware requirements.....	22			
6.2	Software requirements	22			
6.3	Getting up and running.....	22			
6.4	Jumper settings.....	26			
7.	Building and flashing the LPC812 firmware....	28			
7.1	Building the firmware.....	28			
7.2	Flashing the firmware.....	28			
8.	Quick-Jack smartphone compatibility.....	29			
8.1	iOS	29			
8.2	Android.....	29			
8.2.1	MIC signal impedance.....	29			
8.2.2	Temporarily short-circuit on MIC signal	30			
9.	Conclusion.....	31			
10.	Legal information	32			
10.1	Definitions	32			
10.2	Disclaimers.....	32			

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.