TN00024

LPC54608 TCP Socket example with emWin and FreeRTOS

Rev. 1.0 — 11 September 2017 Technical note

Document information

Info	Content
Keywords	LPC54608, TCP Socket, ADC, LED, Ethernet, emWin, FreeRTOS
Abstract	This technical note gives an overview of an example that shows TCP server and clients connecting on a custom port and exchanging ADC data and LED controls over the Ethernet network and displaying the information on the LCD display using emWin graphics and FreeRTOS

Revision history

Rev	Date	Description
1.0	20170911	Initial version.

Contact information

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

All information provided in this document is subject to legal disclaimers.

© NXP Semiconductors N.V. 2017. All rights reserved.

1. Introduction

The LPC5460x is a family of ARM Cortex-M4 based microcontrollers for embedded applications. LPCXpresso development board for LPC5460x MCUs is used in this technical note. Details of the board are found below:

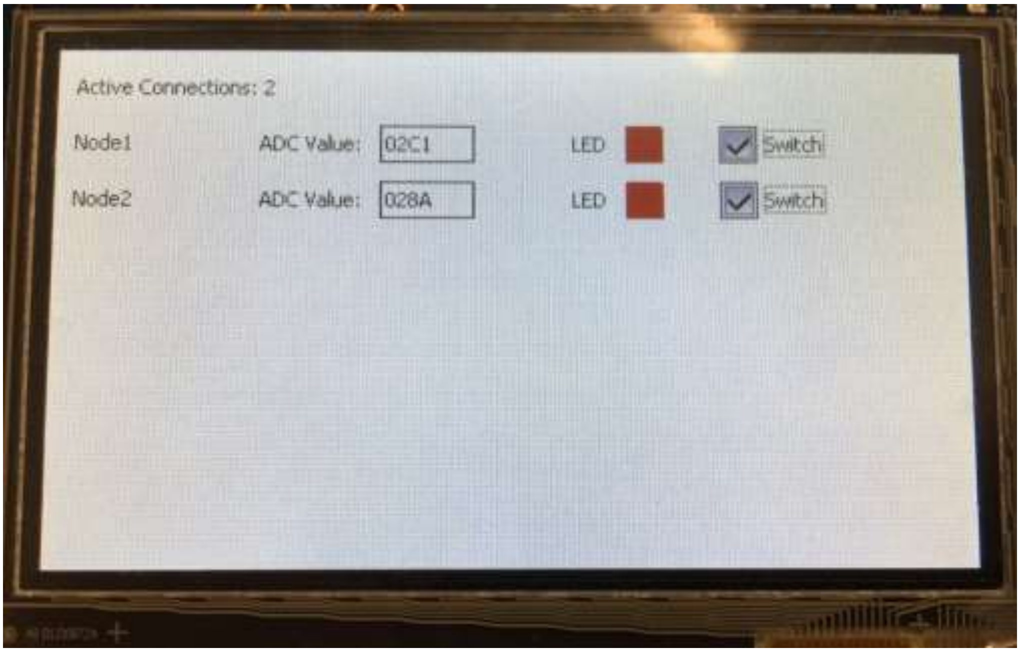
http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/lpc-cortex-m-mcus/lpc54000-series-cortex-m4-mcus/lpcxpresso-development-board-for-lpc5460x-mcus:OM13092

Fig 1. LPC54608 LPCXpresso Development Board

2. Description

The example demonstrates using sequential API's of the lwIP stack to implement a custom TCP server and client communicating on a user defined port and protocol running on FreeRTOS. The server receives ADC Channel 0 data (internal temperature sensor) from all the clients that it is connected, the server can control the LED's on all the connected clients. The client receives the ADC data from the server and can control the LED on the server, each client controls a different LED. The number of nodes connected to the ADC data and LED controls, and LED status of the connected node are displayed on the LCD display of the LPC54608 LPCXpresso development board using emWin graphics as shown in Fig 2 and Fig 3 below.

The TCP server and client implement a custom heart beat/Keep Alive messaging which enables the dynamic detection of node disconnection and connectivity. The example supports dynamic plug and play of the nodes which is reflected on the LCD display. In the example, the server and client are meant to be in the same network. Since, this is a user defined protocol and the client needs to know the IP address of the server.

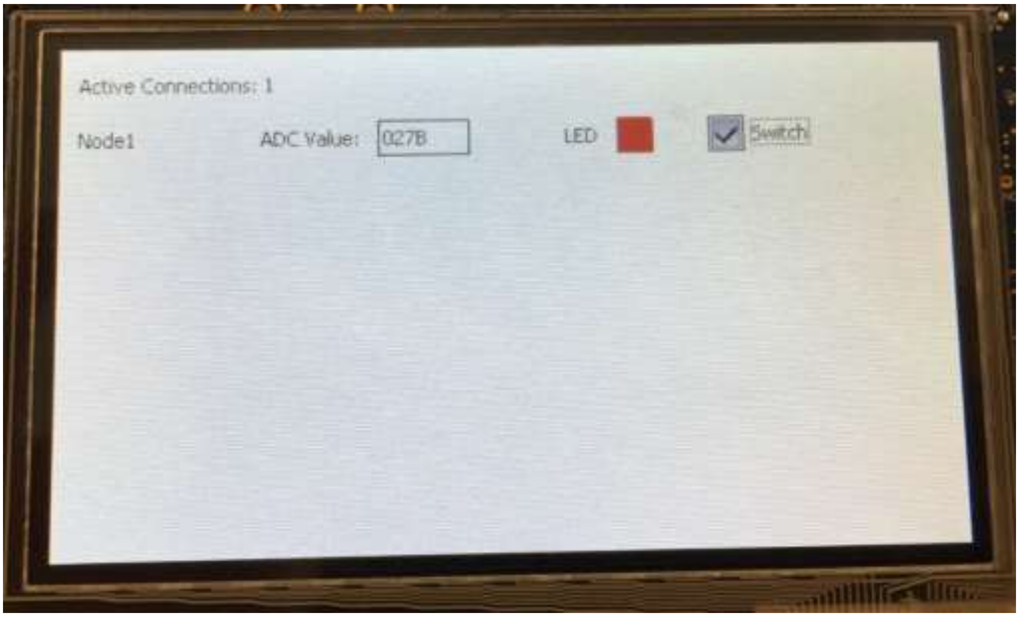


Active Connections: 2

Node1 ADC Value: 02C1 LED 🟥 ☑ Switch

Node2 ADC Value: 028A LED 🟥 ☑ Switch

Fig 2. GUI display on server connected to two clients



Active Connections: 1

Node1 ADC Value: 027B LED 🟥 ☑ Switch

Fig 3. GUI display on client connected to the server

All information provided in this document is subject to legal disclaimers.
© NXP Semiconductors N.V. 2017. All rights reserved.

The example is available in three tool chains:

- MCUXpresso IDE v10.0
- Keil MDK v5.23
- IAR Workbench v8.0

The Keil and IAR examples are found in:

lwip_tcpSocket_freertos\boards\lpcxpresso54608\demo_apps\lwip\lwip_tcpSocket\freertos

The MCUXpresso example can be found in the zip file:

lpc54608_tcpSocket_emWin_freertos_mcux.zip

3. Configuring the example

The header file app.h configures the node as either a server or one of the clients. The current implementation has the option of configuring the example as SERVER or CLIENT1 or CLIENT2. CLIENT1 and CLIENT2 option are the same except that the IP address and the MAC address are different. By choosing the right option you could compile the example as server or client and download it to the respective board.

The example has three FreeRTOS threads:

1. The TCP/IP core stack of lwIP this is as per lwIP design.
2. The application specific TCP Socket thread using sequential API’s of the lwIP.
3. The emWin GUI thread.

The application specific TCP Socket thread is implemented in tcpSocket.c/h. This implements the custom TCP server/client protocol. The TCP Socket thread communicates to the lwIP stack through sequential API’s which in turn uses messages queues. The TCP Socket thread and emWin GUI thread communicate through message queues as well. The TCP Socket thread also uses a FreeRTOS software timer to poll at 500 ms.

Since the TCP Socket thread can be blocked on multiple resource objects, QueueSet is being used for the TCP Socket thread and the queue set comprises of the message queues for the lwIP thread and emWin GUI thread and the semaphore used for the polling software timer.

In tcpSocket.c the function process_rx_data processes the received messages and takes actions like updating the GUI and controlling the LED’s. This function can be updated to change/enhance the protocol.

The emWin GUI thread is implemented in emWin_gui.c/h, it captures the touch events, calls the application GUI periodic function implemented in app_gui.c and runs the emWin execution routine. The emWin GUI thread is executed again after a 10 ms delay.

The file app_gui.c/h contains all the necessary implementation for the GUI on the LCD display and is customized as per user needs.

All information provided in this document is subject to legal disclaimers.

© NXP Semiconductors N.V. 2017. All rights reserved.

4. Legal information

4.1 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.

4.2 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.

4.3 Trademarks

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

5. List of figures

All information provided in this document is subject to legal disclaimers.

© NXP Semiconductors N.V. 2017. All rights reserved.

6. Contents

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

© NXP Semiconductors N.V. 2017. All rights reserved.

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

Date of release: 11 September 2017
Document identifier: TN00024