AN10744

Ethernet secondary ISP bootloader

Rev. 01 — 3 September 2008 Application note

Document information

Info	Content
Keywords	Network, Ethernet, Bootloader
Abstract	Developing a secondary bootloader using Ethernet.

Revision history

Rev	Date	Description
01	20080903	Initial version.

Contact information

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

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

1. Introduction

The NXP LPC2000 series flash microcontrollers provide the user a convenient way to update the flash contents in the field for bug fixes or product updates. This can be achieved using the following two methods:

• In-System Programming: In-System programming (ISP) is programming or reprogramming the on-chip flash memory, using the boot loader software and UART0 serial port. This can be done when the part resides in the end-user board.

• In Application Programming: In-Application (IAP) programming is performing erase and write operation on the on-chip flash memory, as directed by the end-user application code.

A secondary bootloader allows the user's application code to be downloaded using alternative channels other than the standard UART0 used by the primary bootloader (on-chip). Possible secondary bootloaders can be written for USB, Ethernet, SPI, SSP, CAN, and even I/Os. The secondary bootloader utilizes IAP as a method to update the user's application code.

This application note will use Ethernet as an example for developing the secondary bootloader. The code was tested using the LPC2378 and the Keil's MCB2300 evaluation board. This code can be easily modified to work with LPC23xx and LPC24xx series microcontrollers that have an enabled Ethernet block.

The following sections will present a guideline for development and implementation of the Ethernet secondary bootloader design, configuration, and test.

2. Bootloader design

As with any bootloader, the user needs to select an appropriate bootloader entry mechanism for the secondary bootloader. It can be through a dedicated hardware pin or through software handshake.

Upon power-up, the secondary ISP bootloader needs to check the entry mechanism. If valid, the secondary ISP bootloader will be executed. If the entry mechanism is not valid, then the code will jump to the user code address.

Within the secondary ISP bootloader, an agreed communication protocol with the outside world is required.

It is worth noting that even when the secondary bootloader is installed on the chip, the primary on-chip bootloader will always execute after reset or power up. However, the primary bootloader entry mechanism can be deactivated using the Code Read Protection feature found in the LPC2000 devices. In other words, by enabling CRP3, the default ISP entry mechanism (determined by checking P0.14 for the LPC214x or P2.10 for the LPC23xx/24xx devices) will be bypassed, allowing user code, in this case the secondary bootloader, will always be executed after reset or power up.

2.1 Entry mechanism

2.1.1 Dedicated hardware pin

The secondary ISP bootloader checks the status of a pin to determine if the entry is valid. This is the easiest way since no post processing is needed.

2.1.2 Activity indicator LED

While the microcontroller has entered into the Ethernet bootloader, it will activate the Activity Indicator LED. This LED shows that the bootloader is currently running.

2.1.3 Software handshake

In some applications, the system is required to share existing pins or limited pins, the Ethernet bootloader will have to perform some form of handshaking to ensure necessary conditions are met before entering.

2.1.4 Channel selection

In many applications, serial communication is preferred because it saves on I/O pins.

In this application note, the focus will be on the Ethernet communications channel.

All bootloaders have some form of Host-Slave relationship. Usually, the host device (in this case the windows workstation), will be initiating the communication. The host can be a PC or another embedded system. With this relationship, the bootloader developer will need to consider the effort needed for developing and testing the Slave software interface.

In the LPC2000 family supports an on chip UART bootloader. Although this utility has the ability to use the UART channel it is NOT support. FlashMagic provides support for the UART channel. The Ethernet host software will be discussed further in section 4.

Remark: FlashMagic can be downloaded for free at:

http://www.nxp.com/redirect/flashmagictool.com/

2.2 Bootloader exit

Upon completion or timeout, the exit strategy is usually a reset or power cycle with the application removing the Entry condition (before next power on). It is also possible to issue the ISP “GO” command, allowing the controller to be redirected to the user application.

3. Target design

3.1 Prerequisites

It is assumed that the microcontroller has an enabled Ethernet block found in the LPC23xx/24xx series.

The assigned Media Access Controller (MAC) Address on the controller must be a unique address that **must** not be duplicated on the same Layer 2 broadcast collision domain.

Remark: When connecting directly from a workstation to the microcontroller a crossover Ethernet cable may be needed.

Only one node may establish communications with the microcontroller at any given time.

3.2 Communications design

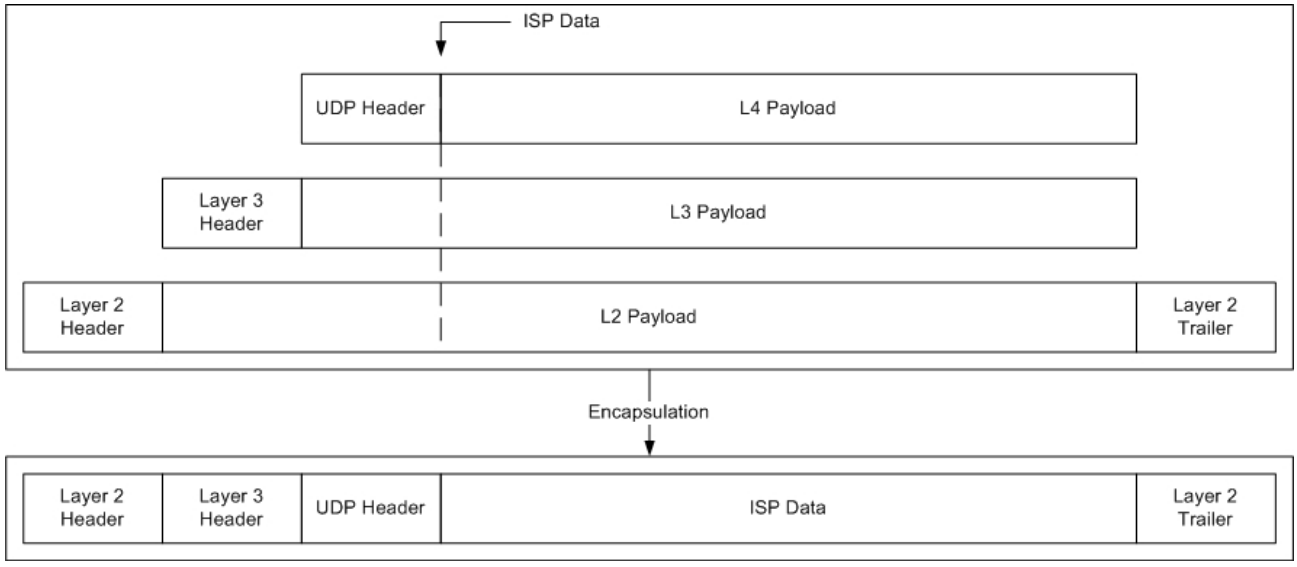
The ISP/IAP communications protocol was originally designed to be used with the UART serial interface and already handles the “handshaking" procedure. To keep the overhead on the embedded system to a minimum the Ethernet will function as simple means of setting packet filters and transporting data without any synchronization.

To mimic an initial synchronization process with the microcontroller, the Host will send a "**?**<CR><LF>" to the microcontroller and wait for a "**Synchronized**<CR><LF>" to be sent back. As the Host receives the "**Synchronized**<CR><LF>" string, it will also reply with "**Synchronized**<CR><LF>" and wait for an "**OK**<CR><LF>". After the synchronization has been established, the host will send the unlock code "**U 23130**<CR><LF>".

Remark: In case of a communication error, the microcontroller has to be reset and the synchronization and unlock procedure have to be repeated.

3.2.1 User Datagram Protocol (UDP)

The UDP packet itself is a "connectionless" oriented protocol in which the transmitting node doesn't care if the packet arrives at its destination. UDP has the advantage that it doesn't need to maintain an active "handshaking" session to transfer data, making it more suitable on systems with limited resources.



(1) Depiction of the combined Layers 2, 3, and 4

Fig 1. Data encapsulation

The Ethernet bootloader accepts data based upon Layer 2 frames by checking the destination address of each frame the Ethernet controller receives. To simply the development process on the windows platform the .NET framework is used to transmit UDP packets. UDP packets however are encapsulated within the Layer 2 frame, meaning that the actual data needed for the Ethernet bootloader within each frame is offset.

For the microcontroller to send data back to the host, it needs to swap the destination and source fields from the Layer 2, 3, and 4 headers. Additionally, it needs to perform a checksum calculation with the data returned from the ISP handler.

3.2.2 Filters

After the microcontroller received the synchronization packet ("**?**"), it will only respond to packets that have the identical source MAC address, UDP source port and UDP

destination port. This feature is implemented for additional protection to prevent stray packets from interfering in the communication session.

3.2.3 EMAC descriptors

A typical windows workstation will generally be able to transmit data at a much higher rate than an embedded system. Since the Ethernet packets are “connectionless”, the Ethernet controller doesn’t care if it lost a packet or not. However, the ISP/IAP protocol requires the host send 20 UUEncoded data packets without receiving any acknowledgment until after the 20th packet. In order to prevent the loss of data packets due to the limited buffer, the EMAC descriptors have been modified.

- The descriptor size has been modified to only accept up to 120 bytes of Layer 2 Ethernet data.
- The number of Rx descriptors has been increased from 4 to 25 to buffer the 20 UUEncoded data packets.
- The number of Tx descriptors has been decreased from 4 to 3 to accommodate the additional Rx descriptors.

In typical applications the Layer 2 payload is around 1500 bytes, thus the user must ensure that no other node is sending anything to that particular microcontroller.

Remark: LPC23xx/24xx Rev - will not work properly using the given EMAC descriptor settings. See the “Ethernet.3” in the LPC23xx/24xx errata sheet.

3.2.4 Checking for incoming data

In order for the bootloader to check if a packet has arrived, it polls a circular receive buffer and checks to see if the number of packets “consumed” matches the number of packets “produced”. The Ethernet hardware increments the MAC_RXPRODUCEINDEX and the Ethernet driver will increment the MAC_RXCONSUMEINDEX. While the two values match, the bootloader waits for a packet.

Remark: When the “MAC_RXCONSUMEINDEX” and “MAC_RXPRODUCEINDEX” indexes match, the receive buffer is empty.

3.3 Modifying the bootloader

3.3.1 Setup File (sbl_config.h)

This file configures the secondary bootloader. The user should change this according to their application.

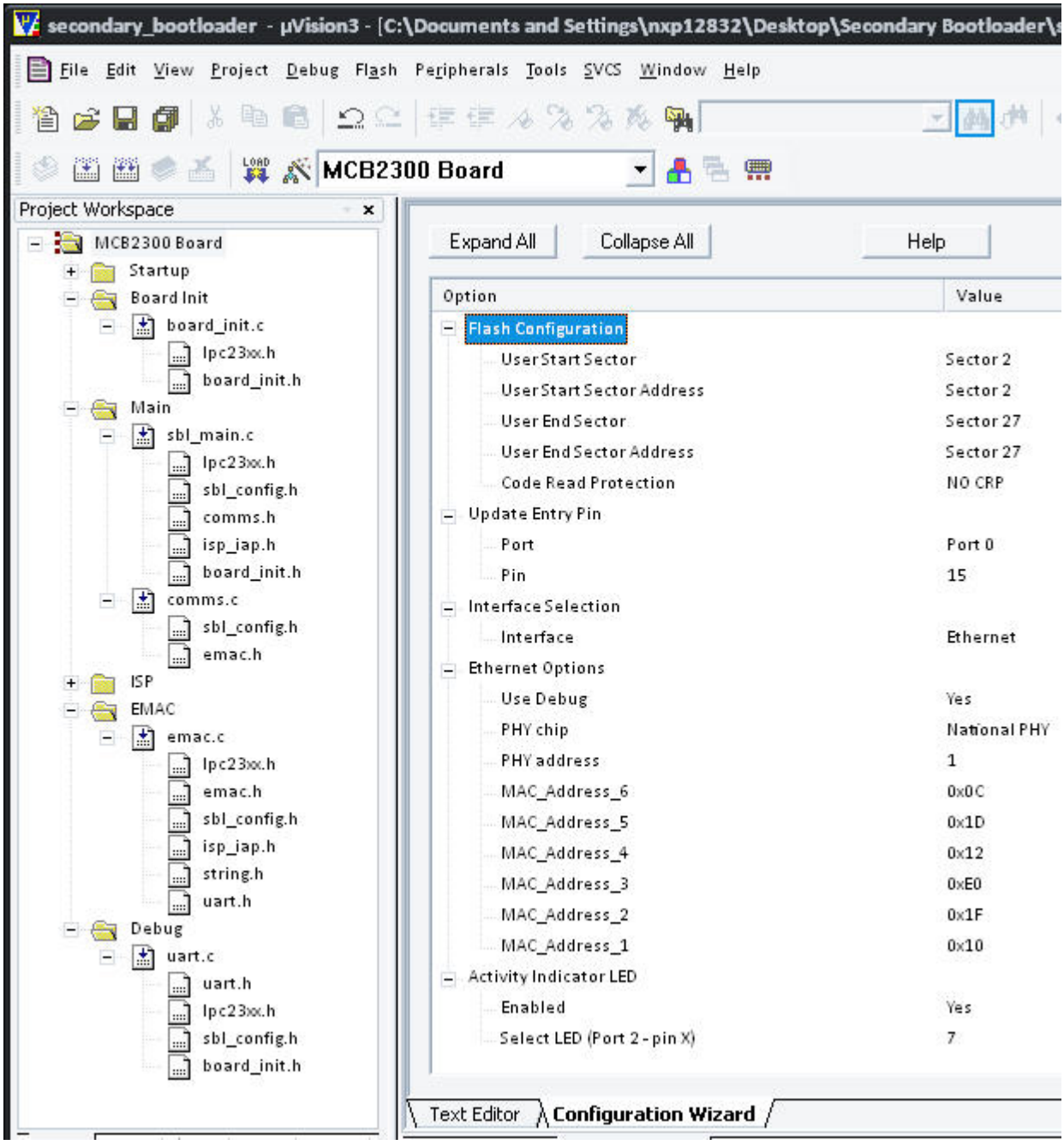


Fig 2. Keil's uVision Configuration Wizard for the sbl_config.h header

Table 1. Configuration parameters for sbl_config.h

Configuration Parameter Definitions	
Flash Configuration	
USER_SECTOR_START	Flash sector that contains the user code entry point
USER_START_SECTOR_ADDRESS	Address of USER_SECTOR_START
USER_END_SECTOR	Last flash sector that contain user code
USER_END_SECTOR_ADDRESS	Address of USER_END_SECTOR
CRP (Code Read Protection)	Sets the Code protection value (CRP1,2,3)
Update Entry Pin	
ISP_ENTRY_GPIO_REG (Port)	Port address used by the Entry Pin
ISP_ENTRY_PIN (Pin)	Pin number used by the Entry Pin
Interface Selection	
USE_ETHERNET (Interface)	Selects interface method, “Other” disables Ethernet
Ethernet Options	
ETHERNET_DEBUG (Use Debug)	Enables or Disables UART0 debug output
MYMAC_6 (MAC_6)	1st Byte of the MAC Address
MYMAC_5 (MAC_5)	2nd Byte of the MAC Address
MYMAC_4 (MAC_4)	3rd Byte of the MAC Address
MYMAC_3 (MAC_3)	4th Byte of the MAC Address
MYMAC_2 (MAC_2)	5th Byte of the MAC Address
MYMAC_1 (MAC_1)	6th Byte of the MAC Address
Activity Indictor LED	
LED_ENABLED (Enabled)	Enabled or Disables Status LED
PORT2_PIN (Select LED)	Select the LED pin of Port 2 (0-7)

Remark: The MAC Address follows the following format:

“MAC_6 - MAC_5 - MAC_4 - MAC_3 - MAC_2 - MAC_1”

3.3.2 Entry pin

This specifies which pin is used to detect whether or not the Ethernet bootloader should start after reset.

3.3.3 Activity indicator LED

When the Activity Indicator LED is blinking it indicates that the Ethernet bootloader is currently running in ISP mode.

3.3.4 Ethernet debug

The Ethernet Debug option allows the user to enable or disable the UART0 debug information output. See UART Debugging for configuration details.

3.3.5 Code Read Protection (CRP)

The secondary Ethernet bootloader can be configured with the LPC23xx/24xx series CRP. Specifics on how CRP works can be found in the LPC23xx/24xx user manual.

Table 2. CRP

CRP explanation	CRP Status
The user flash can be read or written.	No CRP
The user flash content cannot be read but can be updated. The flash memory sectors are updated depending on the new firmware image	CRP1
The user flash content cannot be read but can be updated. The entire user flash memory should be erased before writing the new firmware image.	CRP2
The user flash content cannot be read or updated. The Ethernet bootloader ignores the Entry Mechanism and always executes the user application if present. If user application is not present then the bootloader is executed.	CRP3

3.4 Modifying the user application

Since the secondary Ethernet bootloader also resides in the user flash space, the user code application needs to be modified. The bootloader resides in the first two sectors (0, 1) in flash memory. The application entry point needs to be modified such that it doesn't overlap with the bootloader's code.

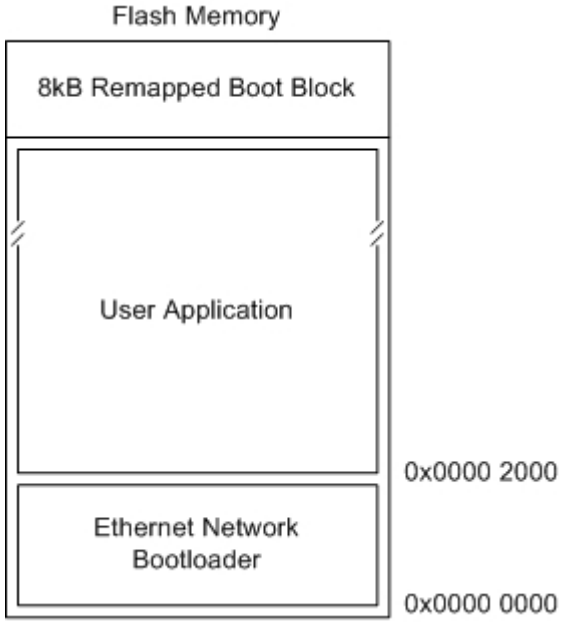
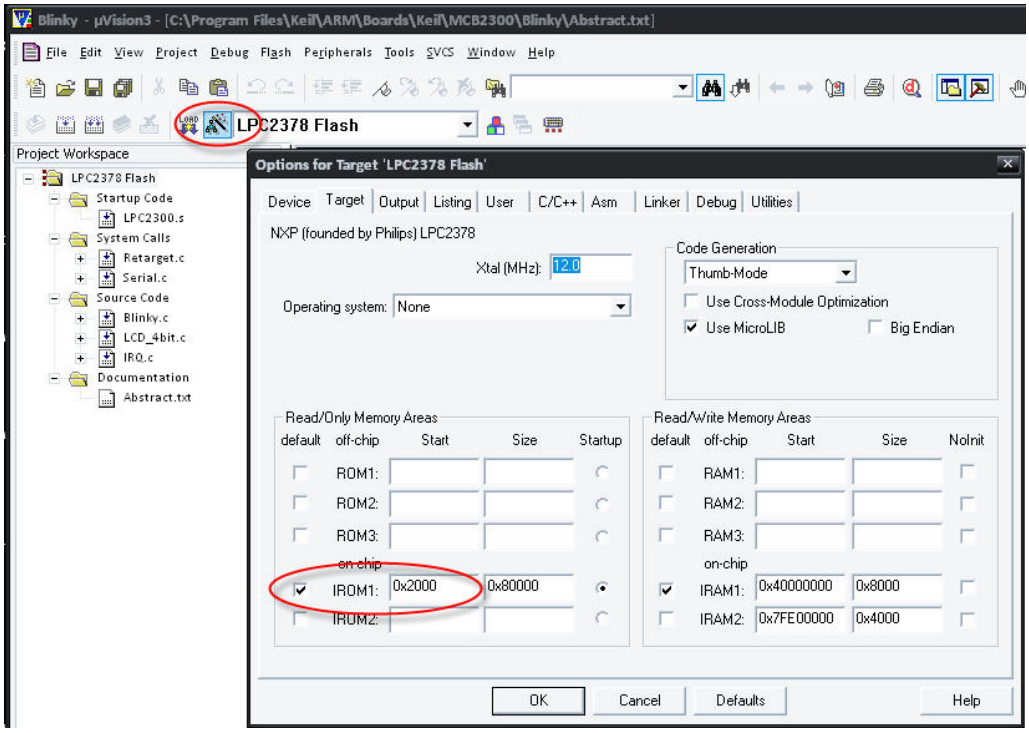


Fig 3. Flash memory layout

To maximize the flash memory, the user application should start immediately after the bootloader at 0x0000 2000 (Start of sector 2).



(1) Modify the Application Code's entry point

Fig 4. Sample blinky code

In Keil's uVision, the application entry point can be changed in "**Debug**" >> "**Options for Target**" >> "**Target**". Change the "**IROM1**" address to "0x2000".

4. Ethernet flash utility

4.1 Prerequisites

The Ethernet Flash utility requires:

- .NET 2.0 Framework
- Installed and configured windows compatible Ethernet card

Remark: Before attempting to execute any of the Ethernet Utility's functions in Network mode, the Ethernet card **must** have already have established an active link. See Connectivity Link.

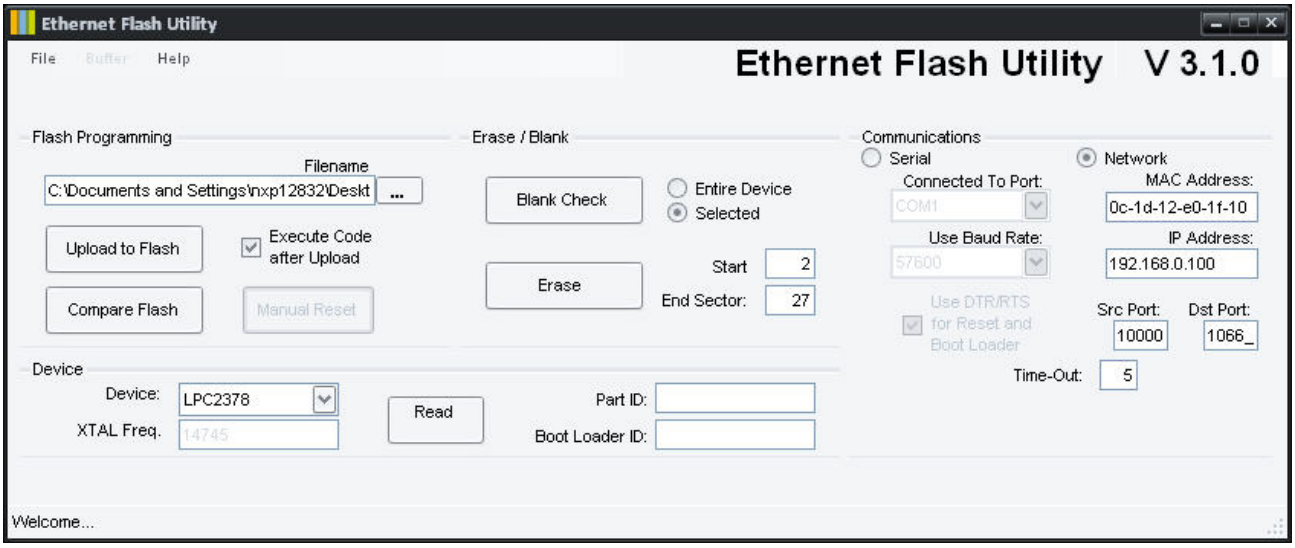


Fig 5. Ethernet flash utility

4.2 Ethernet channel

Upon opening the Ethernet Flash utility, the application will open in Network mode. Generic default settings pertaining to the network configuration are applied.

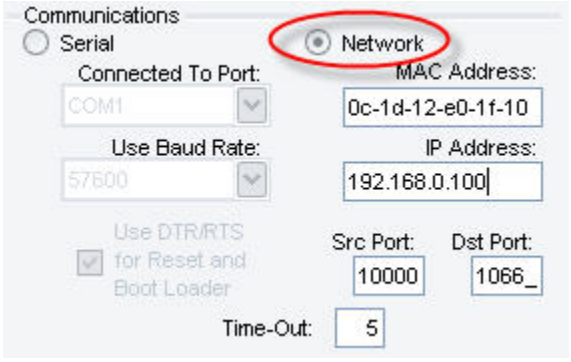


Fig 6. Network mode selection

A network connection with the target board will be established upon the first attempt to execute a Program Function, described in section 4.3. After a session has been established, the user cannot change any of the network settings. If a communication failure occurs or if the application goes into Serial mode, the network session will be terminated. After the network connection is terminated, the user must reset the target board before other communication attempts can be made.

Remark: The “Buffer” menu option is not supported while the application is in Network mode.

4.2.1.1 Application configuration

To configure the network communications the user must supply the following parameters:

Table 3. Ethernet network configuration
Description of the Ethernet parameters

Name	Purpose
MAC Address	Specify the MAC Address of the target board to communicate with.
IP Address	Specify an unused IP address to be assigned to the target board during the communications session.
Src (Source) Port	Specify the Host's local port number (0 - 65535).
Dst (Destination) Port	Specify the port number the target board is listening to (0 - 65535).
Time Out	Specify the time duration the application will wait for a reply from the target board.

Remark: All except of the IP Address can be kept in their default values. The IP Address should be modified such that it can communicate on the workstation's Ethernet adapter.

MAC address

Important: This value **must** match the MAC Address assigned to the target board.

The Ethernet bootloader does not have the capability of responding to ARP (Address Resolution Protocol) requests. To ensure windows will generate the proper UDP packets for the destined target board, the programmed MAC address needs to be supplied.

The application will make a system call to the "ARP" command and create a static ARP entry using the supplied MAC Address. This entry is deleted after the network session is terminated or if the windows system is rebooted.

The MAC Address field will turn red if doesn't follow the correct format.

Remark: The MAC Address must follow the following format using hexadecimal digits:

Format: xx-xx-xx-xx-xx-xx

Example: 0c-1d-12-e0-1f-10

IP address

The microcontroller doesn't need to have an IP Address assigned; however for the host to communicate with the controller it requires one. The required IP Address must be unique address that isn't used on the network. Windows will use that IP Address to select the correct Ethernet card to transmit the UDP packet. If the IP Address is already used by another node, that particular station will start replying, causing interference with the network communications.

The IP Address field will turn red if doesn't follow the correct format.

Remark: The IP Address must follow the following format using decimal digits (0-255):

Format: xxx.xxx.xxx.xxx

Example: 192.168.0.100

Source port

The source port is a logical port number that is opened when communications are initiated. This value can be from 0 – 65535. 0 will allow the windows workstation to automatically assign a port to the UDP connection.

Destination port

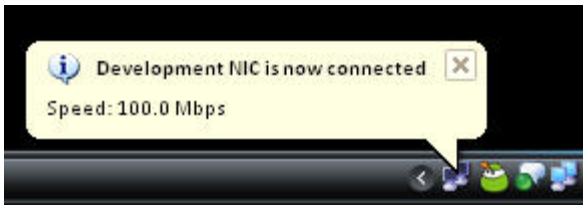
The destination port is a logical port number that target board is listening to. This is useful for debugging purposes while using a network monitoring software. This value can be from 0 – 65535. 0 will allow the windows workstation to automatically assign a port to the UDP connection.

Time out

The Time Out value sets a time during on how long the application will wait for a reply back from the target after a packet has been sent. In Network mode this value roughly corresponds to units in seconds.

Connectivity link

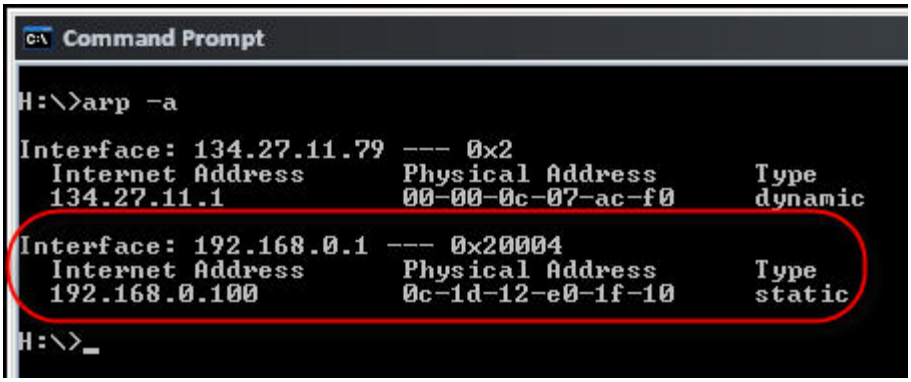
Before attempting to execute any of the Ethernet Utility's functions in Network mode, the Ethernet card **must** have already have established an active link.



(1) Network Link is Active

Fig 7. Windows network link indicator

Caution: If the link was not active on the first attempt, the static ARP entry will not resolve to the correct Ethernet adapter. You may however manually delete the static ARP entry using the "ARP" command and then try again.



(1) The controller's MAC Address has been mapped to the Ethernet adapter on the 192.168.0.1 interface

Fig 8. ARP command

4.2.1.2 Windows configuration

Direct connection

A direct connection with the target board is possible using a Cat5/e cross-over Ethernet patch cable. In order to create a consistent interface, a static IP Address assignment to the windows Ethernet adapter is recommended. This can be done by going to the

"**Network Connections**" window, selecting the desired Ethernet adapter, and then selecting "File" >> "**Properties**".

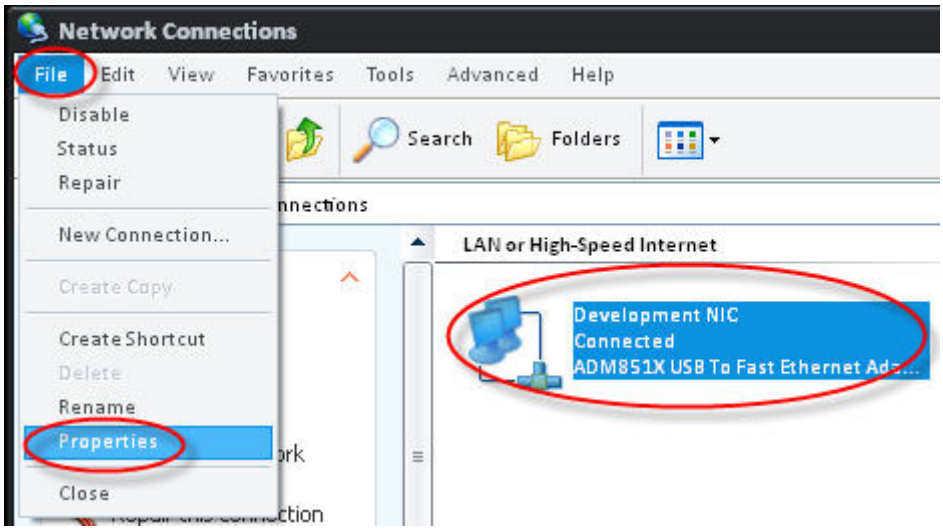


Fig 9. Adapter properties

After having opened the adapter properties window, scroll down the text box to select the "**Internet Protocol (TCP/IP)**" item and click on "**Properties**".

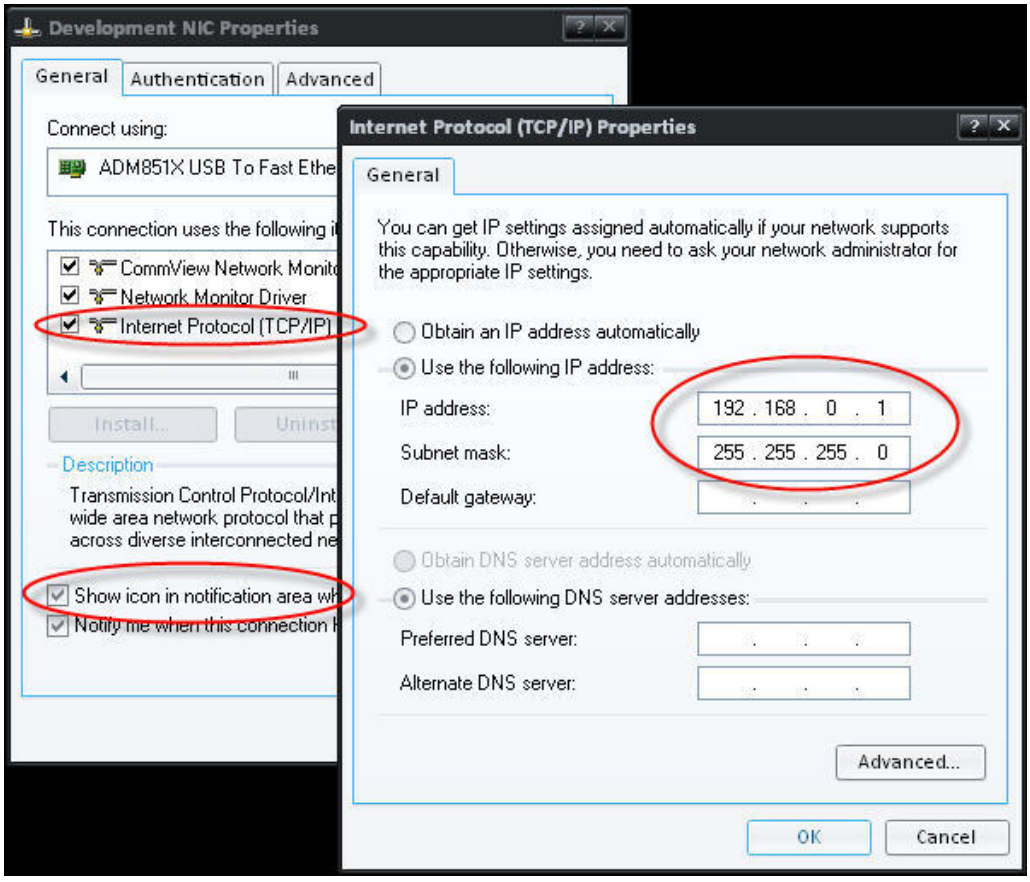


Fig 10. Static adapter configuration

Specify a desired unique IP Address and the subnet mask (in this case IP: 192.168.0.1 with Mask: 255.255.255.0) and then click "**OK**" on both windows.

Local Area Network (LAN) connection

There are many ways of configuring a LAN which are out of the scope of this documentation; hence contact your local administrator to provide the necessary assistance. The following are key requirements:

- If the target board connects directly to a Layer 2 appliance (Switch), use a regular Cat5/e patch cable.
- The target board and the windows workstation must be within the same Layer 2 (switched) collision domain.
- The windows workstation should be configured to communicate on the network.
- The IP Address specified in the application must not already exist on the network.
- The destination and source ports specified in the application must not be filtered by any network appliances.

4.3 Program functions

The following functions are available for use with either the Ethernet or serial bootloaders, unless noted otherwise.

Before using any of the following functions, specify the necessary network / serial communication parameters.

The results of each program function are displayed in the status bar and/or as a message box.

4.3.1 Uploading code

Specify the hex file of the user application and click on the “Upload to Flash” button.

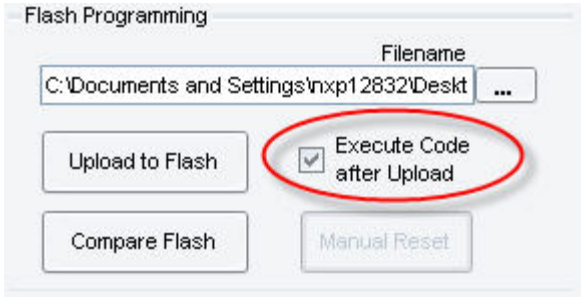


Fig 11. Upload and compare flash program functions

If the “Execute Code after Upload” is checked, it will automatically execute the flashed code after upload.

4.3.2 Comparing code

Using the same hex file as with the “Upload to Flash” function, you may perform a code comparison by clicking on “Compare Flash”. This function will compare the specified file with the contents of the flash. It can be used to verify prior “Upload” function executions.

If the file contents matches with the contents in flash, a message box will appear stating: “File Compare Passed!”

4.3.3 Erasing code

When erasing in Network mode, you must **not** use the “Entire Device” radio button selection. Sectors 0 and 1 are occupied by the Ethernet bootloader. An attempt to erase these sectors will cause an error in network mode.

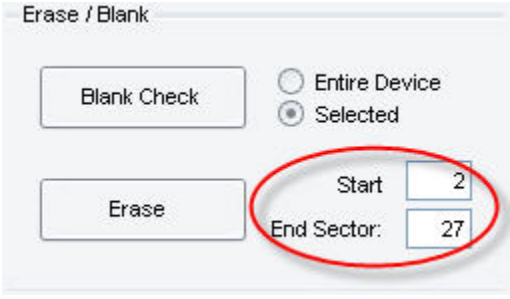


Fig 12. Specifying flash sectors

Select the “Selected” radio button and specify the sectors to be erased. The sectors are specified in the “Start” and “End Sector” fields. The maximum value in the “End Sector” is dictated by the device selected from the “Device” drop-box.

No restrictions in Serial Mode.

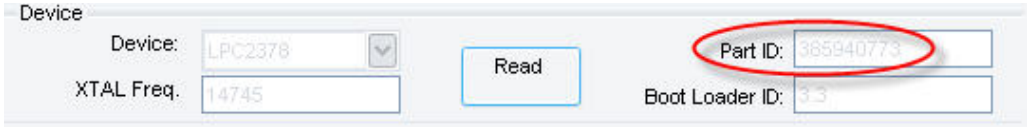
4.3.4 Blank check

In Network mode, “Blank Check” has the same constraints as the erase function. If the “Entire Device” radio button is selected, the blank check function will always fail because the network bootloader resides in the first two sectors in flash. For this reason, the remaining sectors should be specified using the “Start” and “End Sector” fields.

The results of the “Blank Check” are displayed in a message box.

4.3.5 Read part ID

The “Read” function will read the Part ID and the Firmware version of the LPC2xxx; however, only the LPC23xx/24xx microcontrollers are supported in the Network Flash Utility. If the device is recognized then the “Device” drop-box selects that particular device and disables any changes. If the device is not recognized then a “Type not supported” message box will appear and the user has the option of manually selecting a device from a family of similar microcontrollers.



(1) The values have been populated by the “Read” Program function

Fig 13. Device information

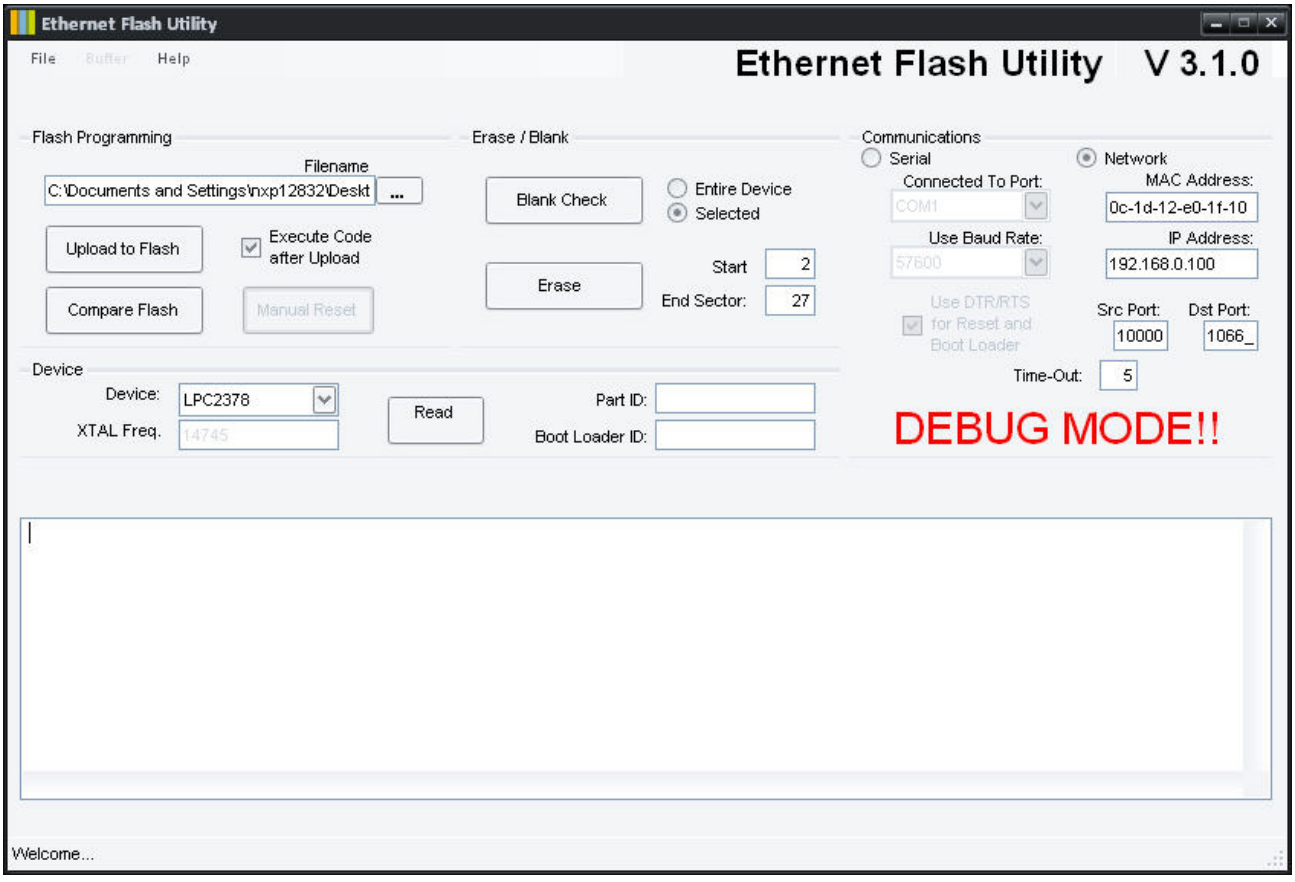
4.3.6 Manual reset (serial mode only)

In Network mode, the “Manual Reset” function is not available. In Serial mode, this function will trigger a reset if the development board supports it.

4.4 Command line arguments

Automated command line features are not available in network mode.

However, by supplying “log” as an argument, the application will open in DEBUG mode. This mode will allow the developer to see the incoming and outgoing ISP commands. The Ethernet Utility Installer creates a “DEBUG MODE” link for the application in the Start Menu.



(1) Incoming and Outgoing ISP command are displayed in the bottom text field.

Fig 14. Ethernet utility in DEBUG MODE

5. Testing

The secondary Ethernet bootloader has been tested and verified to work on the LPC2368, LPC2378, and LPC2478.

5.1 Network connectivity

Before attempting to execute one of the Program Functions the user must ensure that the network card has proper connectivity. Some of the items to check are:

- Does the Ethernet Card have a proper configuration?
- If the target board is connected directly to the windows workstation, is a cross-over cable used instead of a regular Cat5/e patch cable?
- Is there a duplicate IP Address for the microcontroller already on the network?
- Has the development board been reset and do the network link and indicator LEDs light up?

5.2 ISP responses

The microcontroller responses to ISP commands that are documented in the LPC23xx/LPC24xx user manual.

Synchronization must be established before any Program Functions are executed. Details on synchronization are described in Section 3.2. Afterwards, any of the given ISP command codes can be sent to the microcontroller. In turn, it will send back the matching response codes that are documented in the LPC23xx/LPC24xx user manual.

5.2.1 Debugging mode

The Ethernet Flash Utility is started in debugging mode by passing “log” as command line argument, see Section 4.4.

Remark: Flash upload and compare speeds are greatly reduced by having the debugging information enabled.

5.2.2 Network traffic monitoring

A network traffic monitor can be used to capture incoming and outgoing data packets on a particular Ethernet interface. The Ethernet bootloader uses UDP formatted data packets.

5.2.3 UART debugging

If the bootloader is built with the ETHERNET_DEBUG enabled, then it will output debugging information onto the UART channel. This information can be easily viewed using Window’s HyperTerminal.

Bits per second:	57600
Data bits:	8
Parity:	None
Stop Bits:	1

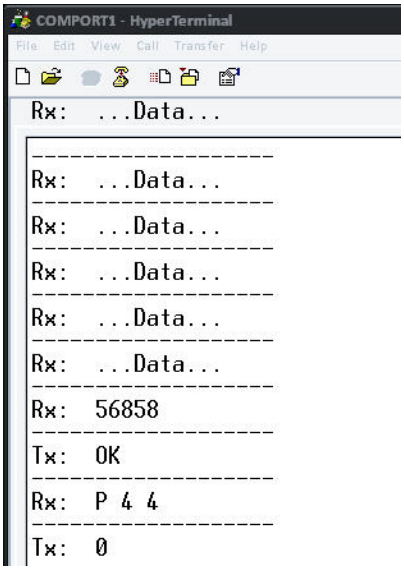


Fig 15. Sample UART debugging output

Remark: Flash upload and compare speeds are greatly reduced by having the debugging information enabled.

6. Conclusion

Using the sample Ethernet bootloader source code and the Ethernet Flash utility, the user can develop different bootloaders that can utilize the Ethernet channel. The Flash Utility allows for a quick way to upload user applications onto flash without special hardware.

7. Legal information

7.1 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.

7.2 Disclaimers

General — 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.

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 medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of a 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 for 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.

8. Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, email to: salesaddresses@nxp.com

Date of release: 3 September 2008
Document identifier: AN10744_1