ANALOG DEVICES

AN-946
APPLICATION NOTE

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

Flash/EE Memory Programming via LIN—Protocol 6

by Aude Richard

INTRODUCTION

A key feature of the ADuC7032, ADuC7033, ADuC7039, ADuCM300, ADuCM330, ADuCM331, ADuCM330WFS, and ADuCM331WFS devices is the ability to download code to on-chip Flash/EE memory while in-circuit. This in-circuit code download is performed via the local interconnect network (LIN) communications bus.

This application note describes the download protocol implemented on the ADuC7032, ADuC7033, ADuC7039, ADuCM300, ADuCM330, ADuCM331, ADuCM330WFS, and ADuCM331WFS devices using Protocol 6 to enable users to develop their own LIN programming tools either for series production programming or for application update.

In this application note, host refers to the host machine (microcontroller, DSP, or other machine) attempting to download data to the ADuC7032, ADuC7033, ADuC7039, ADuCM300, ADuCM330, ADuCM331, ADuCM330WFS, and ADuCM331WFS. Loader refers specifically to serial download firmware resident on these devices.

Note that this application note describes Protocol 6 only. Protocol 6 follows the general procedure defined by UDS (ISO/DIS 14229-1.2, Road Vehicles Unified Diagnostic Services). However, due to the limited code space that is available, the services are limited to the minimum actually required.

Protocol 4 is described in the AN-881 Application Note, *Flash/EE Memory Programming via LIN—Protocol 4*.

The protocol is shown on the device branding, on Line 3. A60 refers to a released version of Protocol 6 whereas A40 refers to Protocol 4.

Table 1. Branding Example for the ADuC7033

Line	LFCSP
Line 1	ADuC7033
Line 2	BCPZ 8L
Line 3	A60 date code
Line 4	Assembly lot number

The programming sequence can be initiated and controlled by a diagnosis tester, which is connected to the LIN master. The LIN master can also be a controller area network (CAN). To facilitate the routing of the diagnosis messages from CAN to LIN, the LIN protocol for programming the module must comply with the *LIN Specification Package* (Revision 2.2A, December 31, 2010).

RUNNING THE LIN LOADER

Use only the LIN and power pins to download code. Table 2 shows the parameters needed to enter download. To enable a download via LIN, the ADuC7032, ADuC7033, ADuC7039, ADuCM300, ADuCM330, ADuCM331, ADuCM330WFS, and ADuCM331WFS device enters loader mode only if the download pin is at the level shown during reset and the Page 0 check code at the address shown in Table 2 is neither the key nor the Page 0 check code of the required type. See Figure 1 for applicable Arm7™ devices and Figure 2 for Cortex-M3 devices.

Typically, the download pin is kept as indicated and entry to download mode is determined by the check key. The user code must have a built-in mechanism for corrupting the Page 0 check key and for resetting the device. This mechanism allows entry to download mode to reprogram the device. Program the check code last to enable reentry to download mode, even in the event that the power fails or another error occurs during programming of the bulk of the program.

The checksum (CS) of Page 0 is simply the sum of all the half words in Page 0, excluding the two half words of the word at the address in Table 2. The checksum must also be stored at this address. The cyclic redundancy check (CRC) is calculated as described in the data sheet of the relevant devices and is the same as will be calculated by the device using the flash sign command for Page 0.

Table 2. Parameters to Enter Download

	Download Pin		Page 0 Check Code		
Part	Name	Level	Key	Address	Type
ADuC7032	NTRST	L	0x27011970	0x80014	CS
ADuC7033	NTRST	L	0x27011970	0x80014	CS
ADuC7039	NTRST	L	0x16400000	0x801FC	CRC
ADuCM300	P0.5	H	0x16400000	0x7FC	CRC
ADuCM330 and ADuCM330WFS	P0.5	H	0x16400000	0x7FC	CRC
ADuCM331 and ADuCM331WFS	P0.5	H	0x16400000	0x7FC	CRC

TABLE OF CONTENTS

REVISION HISTORY

11/2019—Rev. C to Rev. D

Added ADuCM300, ADuCM330WFS, and ADuCM331WFS ..... Universal
Changes to Running the LIN Loader Section 1
Changes to Figure 2 Caption 4
Added Figure 3; Renumbered Sequentially 5

7/2015—Rev. B to Rev. C

Changed ADuC703x to ADuC7032, ADuC7033, and ADuC7039 ..... Throughout
Changed ADuCM33x to ADuCM330 and ADuCM331 ..... Throughout
Changed Running the ADuC703x Loader Section to Running the LIN Loader Section 1
Changes to Introduction Section, Table 1, and Running the LIN Loader Section 1
Added Table 2; Renumbered Sequentially 1
Changes to Figure 1 3
Added Figure 2 4
Changed Packet Structure Section to Frame Structure Section 5
Changes to Frame Structure Section, Table 3, and Table 4 5
Changed Commands Implemented in the On-Chip Loader Section to Frames Implemented in the On-Chip Loader Section 6
Changes to Frames Implemented in the On-Chip Loader Section, Assign NAD Section, Table 5, Table 6, Read-by-Identifier Section, Table 7, and Table 8 6
Changed Identifier 0x32, Identifier 0x33, and Identifier 0x34 Section to Identifier 0x30 to Identifier 0x34 7
Changes to Identifier 0x30 to Identifier 0x34 Section 7
Changes to Table 9 and Table 10 7
Changes to Erase Routine Section, Table 11, Request Download Section, Table 12, Transfer Data Section, and Table 13 8
Deleted Sample LIN Programming Utilities Section 8
Added Cortex-M3 Devices Only Section 9
Changes to Table 14, Request (Check Routine) Section, Table 15, ECU Reset Section, and Table 16 9

4/2013—Rev. A to Rev. B

Added ADuCM330/ADuCM331 ..... Universal

4/2011—Rev. 0 to Rev. A

Changes to Introduction Section 1
Changes to Request Section 5
Added Table 4; Renumbered Sequentially 5

5/2008—Revision 0: Initial Version

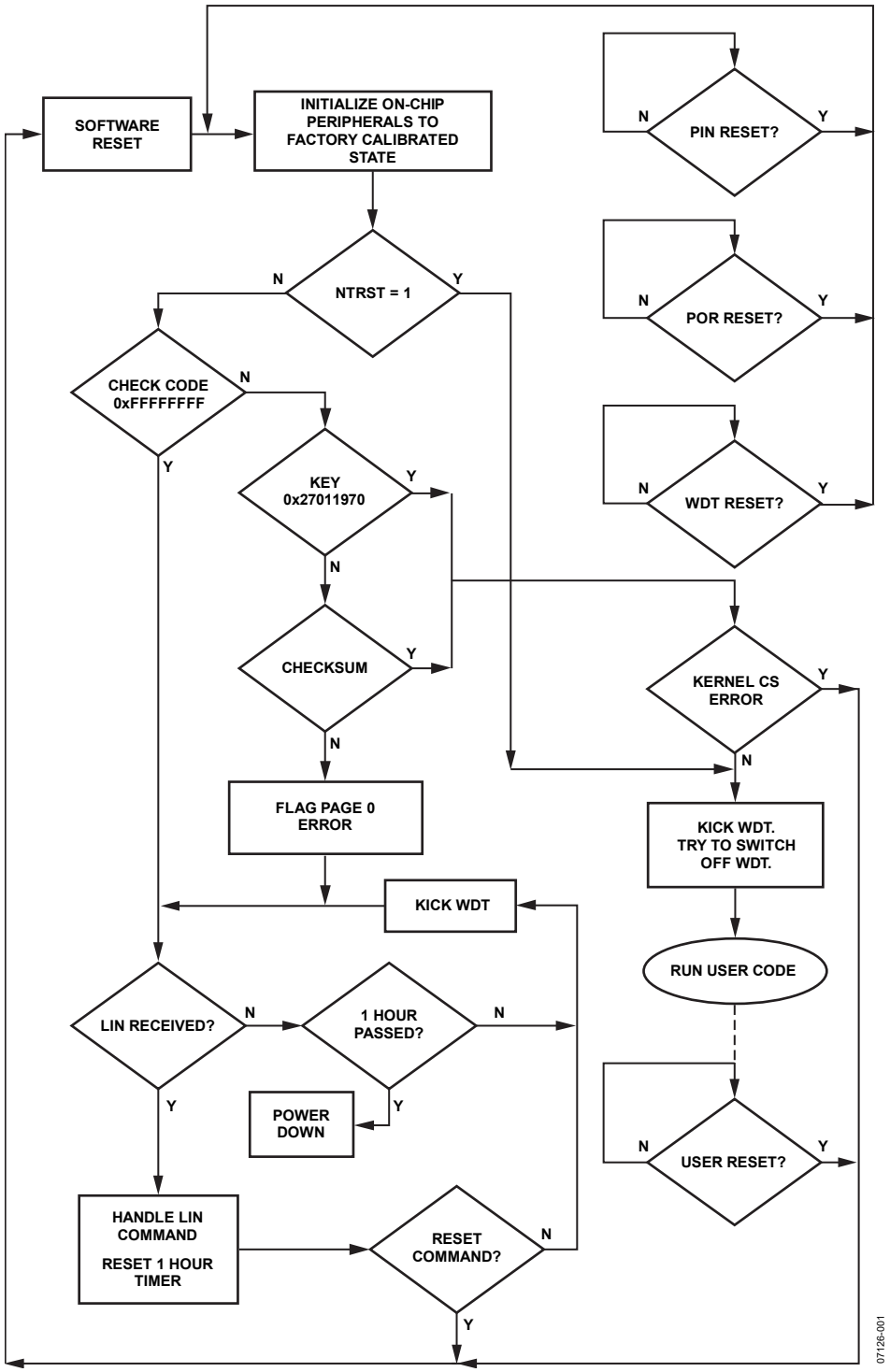

Figure 1. Kernel Flowchart—Arm7 Devices

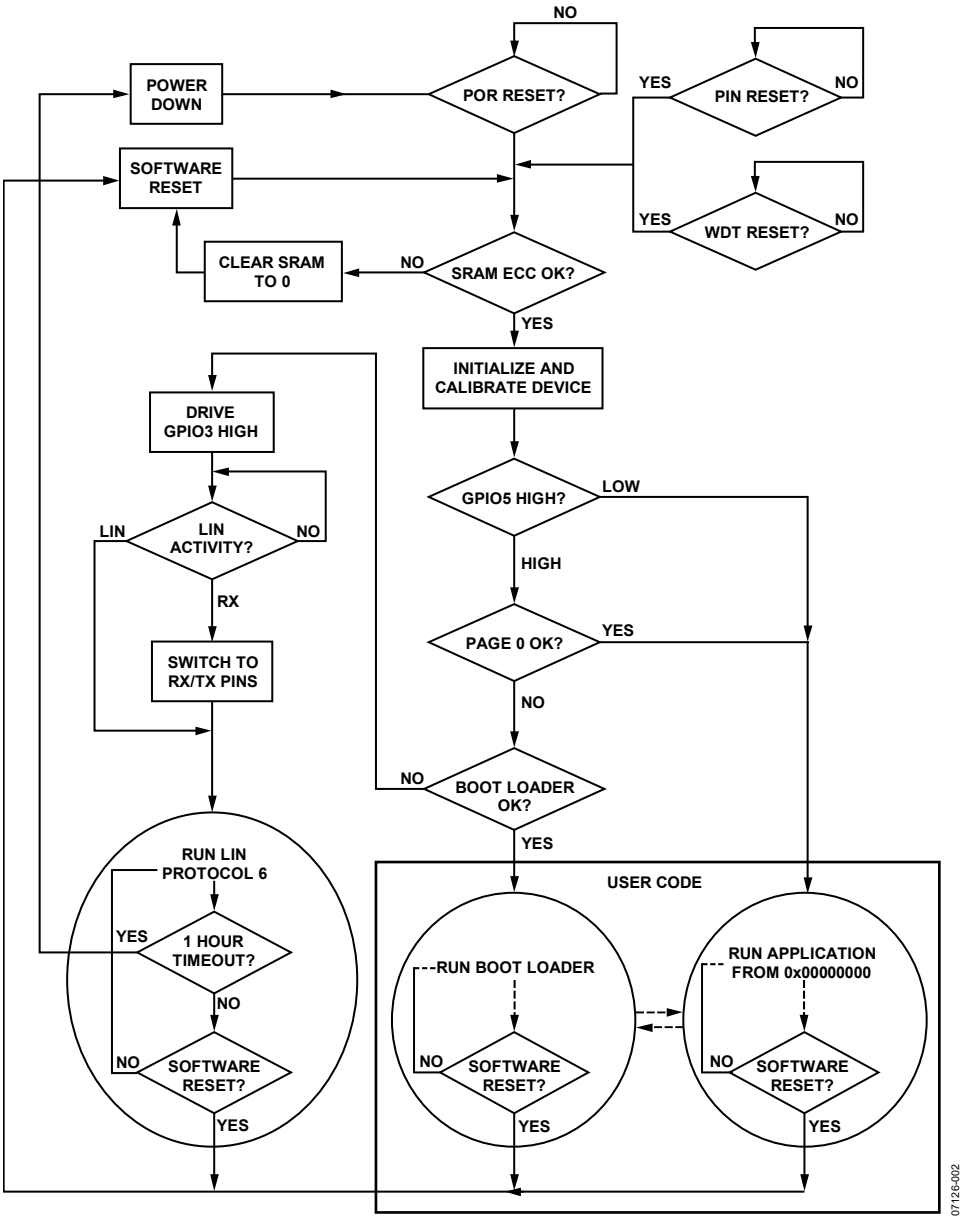

Figure 2. Kernel Flowchart—Arm® Cortex®-M3 ADuCM3xx Devices

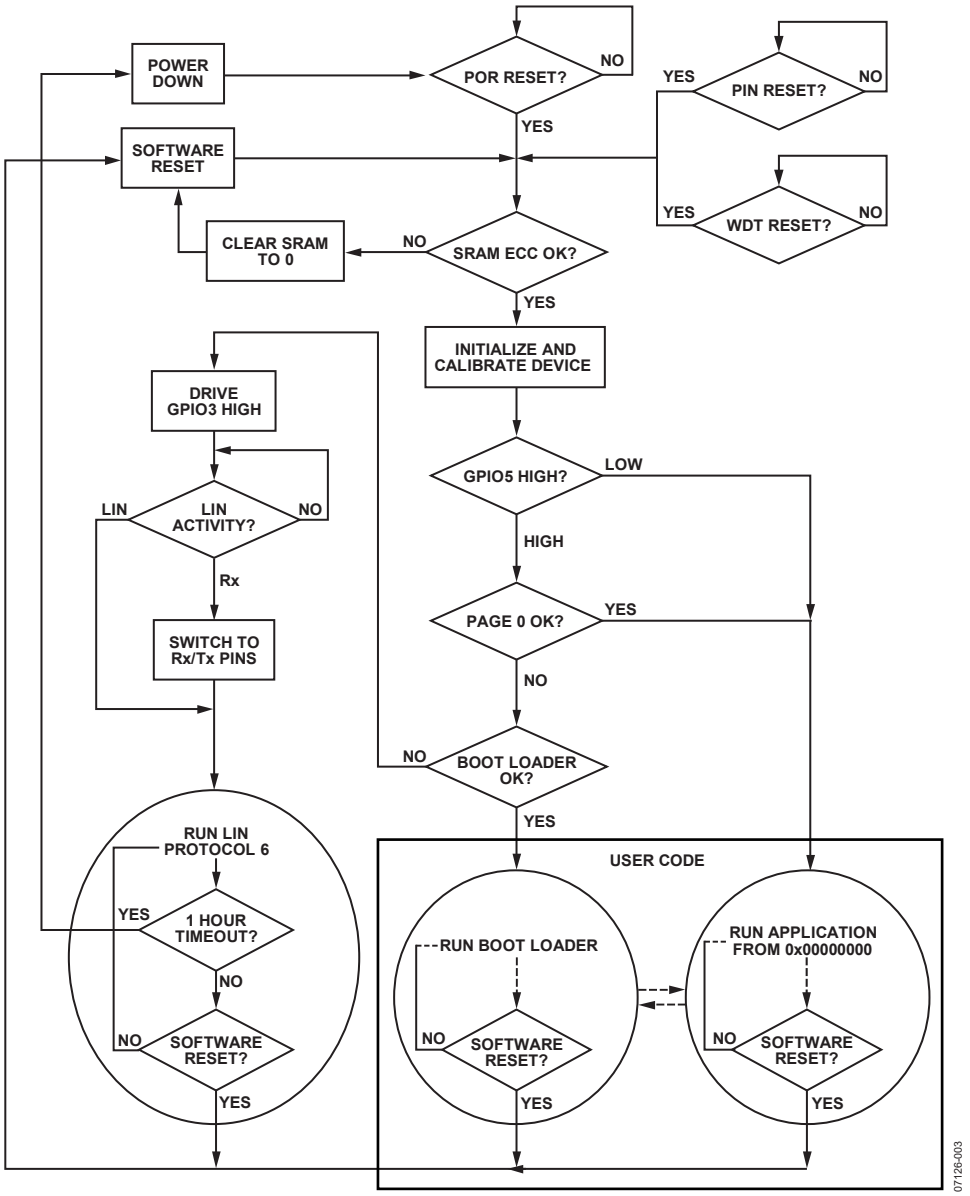

Figure 3. Kernel Flowchart—Arm Cortex-M3 ADuCM33xWFS Devices

FRAME STRUCTURE

The LIN communication with the loader must comply with the following general requirements as per the *LIN Diagnostic and Configuration Specification*, Revision 2.0, September 23, 2003:

- The kernel must implement a slot for each of the two LIN diagnostic frames: master request frame and slave response frame.
- The request from the LIN master must comply with the packet data unit (PDU) format shown in Table 3.
- The responses must comply with the PDU format shown in Table 4.
- Only the PCI type, single frame (SF) can be used. First frame (FF) and consecutive frames (CF) are not supported.
- All frames use the classic checksum.
- Unrecognized frames are ignored.
- Any frame with an error, such as a communication error, is ignored and as a result, a faulty erase routine frame is ignored. A faulty request download frame is ignored and, therefore, subsequent transfer data frames are not recognized and no programming occurs. Any faulty transfer data frame terminates recognition of transfer data frames. In fact, any frame with correct NAD and PCI ≠ 0x05 or SID ≠ 0x36 or a wrong checksum terminates recognition of transfer data frames.
- The addresses shown in the Values column in Table 6 through Table 8 and Table 10 through Table 16 in the Frames Implemented in the On-Chip Loader section are hard coded values and are not examples.

Table 3. Frame Identifier 0x3C

Byte	Description
Byte 1	Node address (NAD)
Byte 2	Protocol control information (PCI)
Byte 3	Service identifier (SID)
Byte 4	Data 1
Byte 5	Data 2
Byte 6	Data 3
Byte 7	Data 4
Byte 8	Data 5

Table 4. Frame Identifier 0x3D

Byte	Description
Byte 1	Node address
Byte 2	Protocol control information
Byte 3	Response identifier (RSID = SID + 0x40)
Byte 4	Data 1
Byte 5	Data 2
Byte 6	Data 3
Byte 7	Data 4
Byte 8	Data 5

FRAMES IMPLEMENTED IN THE ON-CHIP LOADER

The following seven request frames, described in this section, are implemented in Protocol 6:

- Assign NAD
- Read-by-identifier
- Erase routine
- Request download
- Transfer data
- Check routine
- ECU reset

Some of the request frames have corresponding response frames as detailed after each request frame.

ASSIGN NAD

This request assigns a new NAD to the slave so that subsequent diagnostic requests are directed to this specific slave.

Request (Assign NAD)

The LIN Consortium assigns the supplier ID, 0x003A, to Analog Devices, Inc. The function IDs for Protocol 6 are listed in Table 5.

Table 5. Function IDs

Device	MSB	LSB
ADuC7032	0x00	0x32
ADuC7033	0x00	0x32
ADuC7039	0x00	0x39
ADuCM330 and ADuCM331WFS	0x4D	0x30
ADuCM300, ADuCM331, and ADuCM331WFS	0x4D	0x31

To guard against the loss of a slave as a result of network corruption, the slave always recognizes an assign NAD frame with the broadcast NAD 0x7F. This recognition occurs regardless of what the actual NAD of the slave is when this assign NAD frame is transmitted. The kernel then makes the decision whether or not the frame is intended for this slave by checking the supplier ID and function ID. Function ID 0x32 is used as an example within this document.

Table 6. Assign NAD Request

Byte	Description	Values
Byte 1	Initial NAD	0x7F
Byte 2	PCI	0x06
Byte 3	SID	0xB0
Byte 4	Analog Devices supplier ID LSB	0x3A
Byte 5	Analog Devices supplier ID MSB	0x00
Byte 6	Function ID LSB	See Table 5
Byte 7	Function ID MSB	See Table 5
Byte 8	New NAD	User value

Response (Assign NAD)

The slave does not respond to this request.

READ-BY-IDENTIFIER

Before starting the programming sequence, the diagnosis tester requests an identification of the LIN slave using the read by identifier request to ensure the correct slave type is connected.

Request (Read-by-Identifier)

Four identifiers (Identifier 0x0, Identifier 0x32, Identifier 0x33, and Identifier 0x34) are supported.

Table 7. Read-By-Identifier Request

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x06
Byte 3	SID	0xB2
Byte 4	Identifier	0x0/0x32/0x33/0x34
Byte 5	AD supplier ID LSB	0x3A
Byte 6	AD supplier ID MSB	0x00
Byte 7	Function ID LSB	See Table 5
Byte 8	Function ID MSB	See Table 5

Identifier 0x0

The read-by-identifier request with Identifier 0x0 returns the LIN product identification information. This information consists of the 8-byte data frame response shown in Table 8.

Table 8. Identifier 0x0 Data Frame Response

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x06
Byte 3	RSID	0xF2
Byte 4	AD Supplier ID LSB	0x3A
Byte 5	AD Supplier ID MSB	0x00
Byte 6	Function ID LSB	See Table 5
Byte 7	Function ID MSB	See Table 5
Byte 8	Variant	0x00

Identifier 0x30 to Identifier 0x34

The response to these identifiers return user configured data from device memory. The loader expects the contents of the data bytes to be located in the last page of the Flash/EE memory as shown in Table 9.

Table 9. Flash Locations Allocated to Other Identifiers

Address	ID	Byte	Contents
Base + 0xE3	0x30	Data 1	User value
Base + 0xE4	0x30	Data 2	User value
Base + 0xE5	0x30	Data 3	User value
Base + 0xE6	0x30	Data 4	User value
Base + 0xE7	0x30	Data 5	User value
Base + 0xE8	0x31	Data 1	User value
Base + 0xE9	0x31	Data 2	User value
Base + 0xEA	0x31	Data 3	User value
Base + 0xFB	0x31	Data 4	User value
Base + 0xFC	0x31	Data 5	User value

Address	ID	Byte	Contents
Base + 0xED	0x32	Data 1	User value
Base + 0xEE	0x32	Data 2	User value
Base + 0xEF	0x32	Data 3	User value
Base + 0xF0	0x32	Data 4	User value
Base + 0xF1	0x32	Data 5	User value
Base + 0xF2	0x33	Data 1	User value
Base + 0xF3	0x33	Data 2	User value
Base + 0xF4	0x33	Data 3	User value
Base + 0xF5	0x33	Data 4	User value
Base + 0xF6	0x33	Data 5	User value
Base + 0xF7	0x34	Data 1	User value
Base + 0xF8	0x34	Data 2	User value
Base + 0xF9	0x34	Data 3	User value
Base + 0xFA	0x34	Data 4	User value
Base + 0xFB	0x34	Data 5	User value

Note the following:

- The last four bytes of the last page of the Flash/EE memory are reserved for the checksum.
- The ADuC7032, ADuC7033, and ADuC7039 have the base Address 0x97700 and only Identifier 0x32 to Identifier 0x34 can be used.
- The ADuCM330 and ADuCM331WFS have the base Address 0x17F00 and only Identifier 0x30 to Identifier 0x32 can be used.
- The ADuCM300, ADuCM331, and ADuCM331WFS have the base Address 0x1FF00 and only Identifier 0x30 to Identifier 0x32 can be used.

Response (Read-by-Identifier)

Positive response of the LIN slave is shown in Table 10.

Table 10. Read-by-Identifier Slave Response

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x06
Byte 3	RSID	0xF2
Byte 4	Data 1	User value
Byte 5	Data 2	User value
Byte 6	Data 3	User value
Byte 7	Data 4	User value
Byte 8	Data 5	User value

The slave does not give a negative response.

ERASE ROUTINE

Overview

It is possible to erase several pages at once and to request the download and transfer of update data for several subsequent pages. Which update strategy is chosen depends only on the diagnosis tester. However, because 1 in 1000 LIN frames can be expected to show a transmission error, repeating the erase, programming, and verification cycle independently for each single page is recommended. The following three constraints must be taken into account:

- It is not possible to program a memory area smaller than one Flash page of the slave.
- Special consideration must be given to Page 0 programming. It must initially be left with Check Code 0xFFFFFFFF.
- In addition to the verification of the single Flash/EE memory pages, a verification of the checksum of the entire user Flash/EE memory area is recommended before assuming the code is correctly downloaded.
- After the last page is verified, Page 0 must be reprogrammed (without erasing), except this time the desired check code must be included.

Request (Erase Routine)

The erase routine erases the contents of N number of Flash pages, starting with Page P. The value N = 0 is reserved for future use.

Table 11. Erase Routine Request

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x06
Byte 3	SID	0x31
Byte 4	Subfunction ID, first byte	0xFF
Byte 5	Subfunction ID, second byte	0x00
Byte 6	Index of start page, LSB P	User value
Byte 7	Index of start page, MSB P	User value
Byte 8	Number of pages to be erased, N	User value

Index, in Byte 6 and Byte 7, is the page start address right shifted to allow the page size. For example, for the ADuC7032, the start address of Page 2 in Flash/EE memory is 0x80400. This number right shifted by nine bits (for 0x200 bytes per page) is 0x0402. This is represented as Byte 6 = 0x02 and Byte 7 = 0x04.

Response (Erase Routine)

The slave does not respond to this request.

REQUEST DOWNLOAD

Refer to the Overview section.

Request (Request Download)

Table 12. Request Download Request

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x04
Byte 3	SID	0x34
Byte 4	Index of start page, LSB P	User value
Byte 5	Index of start page, MSB P	User value
Byte 6	Number of pages to be programmed, N	User value
Byte 7	Unused	0xFF
Byte 8	Unused	0xFF

The request download request defines the memory area to be programmed. The subsequent data, transmitted via the transfer data frames, is written to N number of pages, starting with Page P. Note that programming always starts at a page boundary and partial programming of pages is not advised.

Response (Request Download)

The slave does not respond to this request.

TRANSFER DATA

These requests are acted on only when following a request download request.

Request (Transfer Data)

The transfer data frames transmit Flash data. The slave expects N × bytes per page of data, where N is the number of pages as defined by the request download request. Only full 4-byte words are allowed. With LIN scheduled at 10 ms per frame, it takes approximately 512/4 × 10 ms = 1.28 sec to flash a single 512-byte page.

Table 13. Data Transfer Request

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x05
Byte 3	SID	0x36
Byte 4	Data 1	User value
Byte 5	Data 2	User value
Byte 6	Data 3	User value
Byte 7	Data 4	User value
Byte 8	Unused	0xFF

Response (Transfer Data)

The slave does not respond to this request.

CHECK ROUTINE

Request (Check Routine)

The check routine calculates the checksum for the memory area starting with Page P and ending with Page P + N − 1. The response, with N = 0, is undefined. Run this check for each single page, and also after all programming is done because errors in the erase or download could affect pages other than the intended pages.

The diagnosis tester compares the checksum received from the LIN slave with a reference checksum provided in the Flash data container. If the checksums differ, the programming procedure is repeated. The checksum is the sum over all 16-bit values from the first 16-bit half word of Page P to the last 16-bit half word of Page P + N − 1. Checksum = (∑ 16-bit words) Modulo 32.

After the check routine request is received, the slave requires about 1 µs per byte to calculate the checksum. Do not initiate the response frame before sufficient time to complete the check has elapsed. This is only relevant to multipage checks because single page checks are always fast enough.

Analog Devices assumes an error model in which not all the half words or bits in the checked area are programmed as required. Such a page always shows fewer zeros and gives a higher checksum. Alternatively, programming an unerased page consistently gives more zeros and a lower checksum. The third possibility is single incorrect half words or bits. The probability for detecting such errors is the same whether one uses a cyclic redundancy check (CRC) or the simple checksum.

Cortex-M3 Devices Only

If, in addition to the index of Start Page P, the MSB (Bit 7 of Byte 7) is set, the chip calculates the CRC of the requested block instead of the CS. This allows two methods of checking the downloaded code for even more confidence that the download is correct.

Table 14. Check Routine Request

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x06
Byte 3	SID	0x31
Byte 4	Subfunction ID, first byte	0xFF
Byte 5	Subfunction ID, second byte	0x01
Byte 6	Index of start page, LSB P	User value
Byte 7	Index of start page, MSB P	User value
Byte 8	Number of pages, N	User value

Response (Check Routine)

Table 15. Check Routine Response

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x05
Byte 3	RSID	0x71
Byte 4	Checksum LSB	User value
Byte 5	Checksum, second byte	User value
Byte 6	Checksum, third byte	User value
Byte 7	Checksum, MSB	User value
Byte 8	Unused	0xFF

ECU RESET

Request (ECU Reset)

Table 16. ECU Reset Request

Byte	Description	Values
Byte 1	NAD	User value
Byte 2	PCI	0x02
Byte 3	SID	0x11
Byte 4	Subfunction ID	0x01
Byte 5	Unused	0xFF
Byte 6	Unused	0xFF
Byte 7	Unused	0xFF
Byte 8	Unused	0xFF

The ECU reset performs a reset of the slave. The ADuC7032, ADuC7033, ADuC7039, ADuCM300, ADuCM330, ADuCM331, ADuCM330WFS, and ADuCM331WFS device restarts as shown in Figure 1. If the value check code is valid, this results in the execution of the application software.

Response (ECU Reset)

The slave does not respond to this request.

NOTES

AN07126-0-11/19(D)