AN11241

AES encryption and decryption software on LPC microcontrollers

Rev. 1.1 — 14 March 2014 Application note

Document information

Info	Content
Keywords	AES, encryption, decryption, FIPS-197, Cortex-M0, Cortex-M3
Abstract	This application note discusses an implementation of an AES encryption algorithm according to the FIPS197 standard.

Revision history

Rev	Date	Description
1.1	20140314	Updated library files (cipher.c and decrypt_aes128.c).
1	20120725	Initial 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 software library provides an implementation of an AES encryption algorithm according to the FIPS197 standard, available at http://csrc.nist.gov/csor.

The library has been developed and tested with the following tool chain:

Tool chain	MDK-ARM Professional	V 4.53.0.0
C Compiler	Armcc.Exe	V 4.1.0.894
Assembler	Armasm.Exe	V 4.1.0.894
Linker/Locator	ArmLink.Exe	V 4.1.0.894
Librarian	ArmAr.Exe	V 4.1.0.894

1.1 Build options for Cortex M0 build

```
-c --cpu Cortex-M0 --li -O3 --apcs=interwork --split_sections -I.\inc -I
C:\dev\Keil_4_50\ARM\RV31\Inc  -I  C:\dev\Keil_4_50\ARM\CMSIS\Include  -I
C:\dev\Keil_4_50\ARM\Inc\NXP\LPC11Axx -DTARGET_LPC11A -o ".\libs\*.o" --
depend ".\libs\*.d"
```

1.2 Build options for Cortex M3 build

```
-c --cpu Cortex-M3 --li -O3 --apcs=interwork --split_sections -I.\inc -
IC:\dev\Keil_4_50\ARM\INC\NXP\LPC17xx -I C:\dev\Keil_4_50\ARM\RV31\Inc -I
C:\dev\Keil_4_50\ARM\CMSIS\Include   -DTARGET_LPC1769 -o ".\libs\*.o" --
depend ".\libs\*.d"
```

The library supports all defined AES key sizes (128, 192 or 256 bits wide) and can be compiled to include or exclude support for any of them. To add or remove support for a specific AES Key format, uncomment or comment the following definitions:

```
#define AES128_SUPPORT
#define AES192_SUPPORT
#define AES256_SUPPORT
```

within the "cypher.h" header file, and rebuild the project called crypt_lib.uvpjt.

For runtime operation, the user can choose between the builds CORTEX_M0 and CORTEX_M3_M4, depending on the target type.

1.3 Notes

The library also supports a special build mode called TEST_MODE, available by selecting the appropriate build rule (either CORTEX_M0_TEST_MODE or CORTEX_M3_M4_TEST_MODE).

Building with this option includes additional checks within the library after the expansion of the encryption key, and after the encryption or decryption of the data. These checks allow the user to include a special test function called runFips197Test(), defined in the fips197_test_keys.c module.

This function runs encryption and decryption tests using as a reference the keys and test vectors defined in the Fips197 standard in appendix A, B and C. This might be useful to verify a correct implementation when porting the library to another tool chain, or when using different optimizations.

In case of errors, the utility function ERROR_CODE checkErrors(void) can be called to verify the library status. The following values are returned as status:

OK	
AES128_DECRYPTKEY_FAILURE	AES128_CRYPTKEY_FAILURE
AES192_DECRYPTKEY_FAILURE	AES192_CRYPTKEY_FAILURE
AES256_DECRYPTKEY_FAILURE	AES256_CRYPTKEY_FAILURE
AES_DECRYPT_FAILURE	AES_CRYPT_FAILURE

The library has been tested on the following targets: LPC11Axx (Cortex-M0 based), LPC1769 (Cortex-M3 based). For porting the library to other targets, the user may extend or modify the target definition in the cipher.h file

```
#ifdef TARGET_LPC11A
 #include "lpc11axx.h"
#endif
#ifdef TARGET_LPC1769
 #include "lpc1769.h"
#endif
```

and include other targets accordingly. Although there is really nothing target specific within the library, other than the CPU core, this is required to include the appropriate headers from the CMSIS library.

2. API Implementation

The following APIs are available to process the data:

- void initCrypt(const CypherConfig configuration)
- void initDeCrypt(const CypherConfig configuration)
- void crypt(const plainData_t* plainText, const uint32_t size, cryptData_t* cryptedText)
- void deCrypt(const cryptData_t* cryptedText, const uint32_t size, plainData_t* plainText)

The pointer of type CypherConfig used to initialize the library needs to point to a configuration structure of the form:

```
typedef struct CypherConfigStr {
       CRYPT_MODE    cryptMode;
       key_t*     keyLocation;
       #ifdef TEST_MODE
       const expKey_t*   referenceKey;
       const cryptData_t*  referenceCryptData;
       #endif
} CypherConfigStr;
```

The user application has to provide a configuration with the following mandatory items defined:

- cryptMode defines the configuration, which can be chosen among the values AES_128, AES_192 or AES_256
- keyLocation is a pointer to the memory location where the AES Key is stored

2.1 Notes

‘referenceKey’ and ‘referenceCryptData’ are only meaningful for the TEST_MODE build. However, the user does not have to specify anything when building for this test mode, since the reference configurations used are already provided within the fips197_test_keys.c module.

The ‘crypt’ and ‘decrypt’ functions require pointers to the data item where the input data is located, the size of the data input buffer, and a pointer to the location where the result operation will be stored.

3. API usage notes

1. initCrypt and initDeCrypt need to be called before any crypt or decrypt action. They are needed to initialize the internal expanded cypher keys which will be used to process the data during the encryption or decryption process. They need to be called only once per key. For efficiency reasons, there is only one buffer available to hold the expanded crypt or decrypt key, so if the key is changed, the initCrypt / initDecrypt APIs need to be called to re-initialize the system for the new expanded key.
2. The library does not dynamically allocate data buffers, which means that the pointers passed within the crypt and decrypt functions to store the results need to point to valid memory (which needs to be as large as the size as the input buffer).
3. The buffers used to hold the data need to be a multiple of 16 bytes, which is the minimum data block handled by the AES algorithm.
4. The buffers used to hold the data need to be aligned to a 4 byte address in memory.

4. Application requirement

The following data items need to be allocated within the system RAM:

- For holding the expanded crypt key: 176 bytes (AES128), 208 bytes (AES192), 240 bytes (AES256). This is not computed every time for performance reasons, but kept in a dedicated buffer, under the assumption that the key does not change often.
- 5 bytes per configuration structure (crypt or decrypt).

Regarding the total library size, for reference the following numbers were obtained

```
Code   (inc. data)   RO Data  RW Data  ZI Data   Library Name
2076       70         552       18      480      crypt_lib_CM0.lib
Code   (inc. data)   RO Data  RW Data  ZI Data   Library Name
2094       66         552       18      480      crypt_lib_CM3_CM4.lib
```

Note that these figures include both encryption and decryption support for each AES Key type. Removing support for some key types, or using only encryption or decryption, will lead to much smaller figures.

There are no other dependencies on additional external libraries or modules.

5. Legal information

5.1 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.

5.2 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. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

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 and its suppliers accept 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 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.

5.3 Trademarks

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

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'.

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

Date of release: 14 March 2014
Document identifier: AN11241