Atmel ATAES132

32K AES Serial EEPROM Specification

Preliminary Datasheet

Features

- 32Kbits of standard Serial EEPROM user memory
  - Compatible with the Atmel AT24C32D and the Atmel AT2530B
  - 16 user zones of 2Kbits each
- High security features
  - AES algorithm with 128-bit keys
  - AES-CCM for authentication
  - Message authentication code for cryptographic operations
  - Secure storage for sixteen 128 bit keys
  - Encrypted user memory read and write
  - FIPS random number generator
  - 16 non-reversible monotonic counters
- Flexible user configured security
  - User zone access rights are independently configured
  - Authentication prior to zone access
- Read/write, Encrypted, or Read only user zone options
- High speed serial interface options
  - 10MHz SPI (Mode 0 and 3)
  - 1MHz I^2C
- 2.5V to 5.5V supply, <250nA Sleep
- Packages: SOIC, TSSOP or UDFN
  - Serial EEPROM compatible pinout
- Operating temperature
  - -40° to +85°C

Description

The Atmel® ATAES132 is a high security Serial electrically-erasable and programmable read only memory (EEPROM) providing both authentication and confidential nonvolatile data storage capabilities. Access restrictions for the sixteen user zones are independently configured, which any key can be used with any zone. Keys can also be used for stand-alone authentication. This flexibility permits ATAES132 to be used in a wide range of applications.

The Atmel AES-128 cryptographic engine operates in the AES-CCM mode to provide authentication, stored data encryption/decryption, and message authentication codes. Both internally stored data and/or small external data can be protected by the ATAES132 device.

The ATAES132 pinout is compatible with the standard SPI and I^2C Serial EEPROM to allow placement on existing PC boards. The SPI and I^2C instruction sets are identical to the Atmel Serial EEPROM. The extended security functions are accessed by sending command packets to the ATAES132 using standard write instructions, and reading responses using standard read instructions. The ATAES132 Secure Serial EEPROM architecture allows it to be inserted into existing applications.

The ATAES132 chip incorporates multiple physical security mechanisms to prevent release of the internally stored secrets. Secure personalization features are provided to facilitate third-party product manufacturing.

Table of Contents

1. Introduction

The Atmel® ATAES132 is the first device in a family of high security Serial EEPROM using the advanced encryption standard (AES) cryptographic algorithm to add authentication capability to a standard Serial EEPROM. The ATAES132 provides 32Kbits of EEPROM user data memory, sixteen 128 bit key registers, sixteen non-reversible monotonic counters, factory unique die identification numbers, and a configuration memory. The configuration memory registers control access to the user memory, as well as the restrictions on key and counter functionality.

The user memory can be accessed directly with the standard SPI or I^2C commands if a user zone is configured for open or read-only access. If the user zone security is activated, then the extended ATAES132 command set is used to access the contents of a user zone. The extended ATAES132 commands are executed by writing the command packet to the virtual memory using standard SPI or I^2C write commands. The response packet is retrieved by reading it from the virtual memory using standard SPI or I^2C read commands.

The ATAES132 packages are compatible with the standard SPI and I^2C EEPROM footprints. This allows the ATAES132 to be inserted into many existing Serial EEPROM applications.

1.1. Scope

This *ATAES132 Specification* provides all specifications for its configuration and operation.

1.2. Conventions

The following nomenclature is used throughout this specification.

- **Host** (The SPI or I^2C master device)
  The host initiates all communications with slave devices on the serial interface bus
- **Client** (The ATAES132 Secure Serial EEPROM defined by this specification)
  Operates as a SPI or I^2C slave
- **Nnb** (Binary number)
  Denotes a binary number "nn" (Most significant bit on the left)
- **0xZZZZ** (Hexadecimal number)
  Denotes a hex number "ZZZZ" (Most significant bit on the left)
- **ZZZZ$_h$** (Hexadecimal number)
  Denotes a hex number "ZZZZ" (Most Significant Bit on the left)
- **RegName.FieldName** (Field name)
  Reference to bit field "FieldName" in register "RegName"
- **RegArray[xx].FieldName** (Field name)
  Reference to bit field "FieldName" in register "RegArray[xx]" where "xx" is the array index
- **UZ** (User zone)
  Reference to a user zone number
- **CntID** (Counter ID)
  Reference to a counter number
- **KeyID** (Key ID)
  References to a key register number

1.2.1. Byte Order

The ATAES132 device uses a “big-endian” coding scheme and utilizes the same bit and byte orders as the standard Serial EEPROM. The byte order is identical to the NIST AES specifications (see Appendix A):

- The most significant bit of each byte is transmitted first on the bus
- The most significant byte of multi-byte integers is transmitted prior to the least significant byte. This applies to the CRC, address and other 16 bit command parameters.
- All arrays are transmitted in index order, with byte index 0 first
- Configuration fields that are more than eight bits appear on the bus during a read or write in the index order in which they appear in this specification – the top byte in the input parameters table is byte[0] and appears first on the bus. These fields are arrays of bytes, not multi-byte integers.

1.3. Abbreviations

The following abbreviations are used throughout this specification.

- **AES** (Advanced encryption standard)
  Block cipher algorithm standardized by NIST, with 128 bit block size
- **AES-CCM**
  AES mode using the “Counter with Cipher Block Chaining-Message Authentication Code” algorithm
- **AES-ECB**
  AES mode using the “Electronic Code Book” algorithm
- **Ciphertext**
  Data communicated after it has been encrypted
- **Cleartext**
  Data communicated in a non-encrypted state
- **MAC** (Message authentication code)
  A 128 bit value used to validate the authenticity of ciphertext
- **Nonce** (Number used once)
  A value used in cryptographic operations
- **Plaintext**
  Data which is either the input to encryption or the output of a decryption operation
- **RFU** (Reserved for future use)
  Any feature, memory location, or bit that is held as reserved for future use by Atmel
- **RNG** (Random number generator)
  Produces high-quality pseudo-random numbers

1.4. Communication

The ATAES132 is designed to interface directly with SPI and I^2C microcontrollers. The read and write commands are identical to the standard Atmel Serial EEPROM memory commands for ease of use. Since the ATAES132 pinout is also similar to standard Atmel Serial EEPROM, in some cases, it is possible to use the ATAES132 on existing PC boards.

When read and/or write access to a user zone is unrestricted, then the memory is accessed using the standard I^2C or SPI read and write commands. Similarly, if *Authentication Only* is required and the Authentication requirement has been satisfied, then the memory is accessed directly by the host using standard I^2C or SPI read and write commands.

If the host begins a read operation in an open user zone, but continues reading until a prohibited section of memory is reached, the ATAES132 will continue to increment the address and will return 0xFF for each byte in the restricted user zone. If the host begins a read operation in an open user zone, but continues reading beyond the end of the user memory, the ATAES132 will return 0xFF for each byte requested but will stop incrementing the address.

All other operations, including execution of the extended commands, are performed by using the standard I^2C or SPI read and write commands to exchange data packets via the command and response memory buffers. The device status register reports the state of the device and is used for handshaking between the host and the ATAES132.

1.4.1. Sending Atmel ATAES132 Commands

The ATAES132 commands described in Section 7 are executed by writing the command block to virtual memory (Appendix D) using the standard SPI or I^2C write commands. The response block is retrieved by reading it from the virtual memory using the standard SPI or I^2C read commands.

1.4.1.1. Command Memory Buffer

The command memory buffer is a write-only memory buffer that is used by writing a command block to the buffer at the base address of 0xFE00. After the host completes its write operation to the buffer, the ATAES132 verifies the integrity of the block by checking the 16-bit checksum, and then executes the requested operation. See Section 6.1 for a description of the command packet. See Appendix D for additional command memory buffer information.

Table 1-1. The command memory buffer map

Base Address	Base + 1	Base + 2	Base + 3	......	......	......	......	Base + N-2	Base + N-1
Count	Opcode	Mode	Param1	Param1	Param2	....	DataX	CRC1	CRC2

1.4.1.2. Response Memory Buffer

The response memory buffer is a read-only memory buffer that is used by reading a response from the buffer at the base address of 0xFE00. The base address of the response memory buffer contains the first byte of the response packet after an ATAES132 command is processed. See Section 6.1 for a description of the response packet. See Appendix D for additional response memory buffer information.

Table 1-2. Response memory buffer map following a crypto command

Base Address	Base + 1	Base + 2	Base + 3	......	......	......	......	Base + N-2	Base + N-1
Count	ReturnCode	Data1	Data2	Data3	....	....	DataX	CRC1	CRC2

The response memory buffer is also used to report errors which occur during execution of standard I^2C or SPI write commands. When the I^2C or SPI command execution is complete (as indicated by the STATUS register), the response memory buffer contains a block containing an error code (ReturnCode) if an error occurred, otherwise it contains a block with ReturnCode = 0x00. See Section 6.3 for the error descriptions.

1.4.2. Device Status Register (STATUS)

The device status register is used for handshaking between the host microcontroller and the ATAES132. The host microcontroller is expected to read the STATUS Register before sending a command or reading a response.

The read-only device status register at address 0xFFF0 reports the current status of the ATAES132 device. This register can be read with the standard I^2C or SPI read memory commands. The SPI read status register command can also be used to read the STATUS register as described in Section K.3.6.

Reading the STATUS register does not increment the memory read address, so a host microcontroller can easily monitor the ATAES132 device status by repeatedly reading the STATUS register. See Appendix G for a detailed description of the STATUS register bits and status bit behavior.

Table 1-3. Device status register definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EERR	RRDY	Reserved	CRCE	Reserved	WAKEb	WEN	WIP

The device status register can always be read when the ATAES132 is configured for SPI interface mode even if the ATAES132 is processing a command or writing the EEPROM. When the ATAES132 is configured for I^2C interface mode, the host can read the STATUS register only when the I^2C device address is ACKed.

If the ATAES132 is in the sleep or standby power state, reading the STATUS register forces the ATAES132 to wakeup; the STATUS register is 0xFF until the wakeup process is complete.

2. Memory

The ATAES132 EEPROM is a nonvolatile memory which is divided into several sections, with each section having a different function. The user memory section contains 32Kbits for data storage. The configuration memory section contains the configuration information, security control registers, and counters. The key memory stores the sixteen secret keys used to perform cryptographic functions. The EEPROM page length is 32 bytes. The ATAES132 SRAM buffers and registers are located near the top of the memory address space, and are accessed using the standard EEPROM read/write commands.

The complete memory map is shown in Appendix B. Each portion of the memory is described briefly in the following sections.

2.1. User Memory

The 32Kbit user memory is organized as 16 user zones of 2Kbits each. Each user zone has an associated user zone configuration register in the configuration memory. A user zone can only be accessed when the security requirements specified in the associated user zone configuration register have been satisfied. All bytes within a user zone have the same access restrictions. Since the user zone access restrictions are independently configured, the security requirements for each user zone can be unique. Any key can be used with any user zone.

Each user zone can be configured to require authentication, read encryption, write encryption, a combination of these, or no security. The user memory can be accessed directly with standard SPI or I^2C commands if a user zone is configured for open or read-only access. If the user zone security is activated, then the extended ATAES132 command set is used to access the contents of a user zone.

2.1.1. Automatic Post-Write Data Verification

The write and EncWrite commands include an automatic data verification function. After the EEPROM write is complete, the data verification logic reads the new EEPROM contents and compares it to the data received from the host. If the data does not match the ATAES132 sets the EERR bit in the STATUS register and returns a DataMatch error code. If the data is correct, then the ReturnCode indicates success.

2.2. Key Memory

The Key Memory securely stores 16 keys which are each 128 bits long. Each key has an associated key configuration register in the configuration memory. Keys can only be used for the cryptographic functions enabled in the key configuration register. Individual keys can be configured to require a successful authentication prior to use. Key values can never be read from the ATAES132 under any circumstances. See Appendix F for the key memory map.

The key memory can be written prior to locking with either encrypted or cleartext data. Encrypted writes are performed using the EncWrite command (see Section 7.11). Cleartext writes are performed using the standard SPI or I^2C write commands (see Section 5.3). After locking, the key registers are managed with the KeyCompute, KeyExport, KeyImport, KeyLoad, and KeyTransfer commands. The KeyTransfer command allows the user memory to be used as an extended key memory; eight keys can be stored in each user zone (see Section 7.17).

2.3. Configuration Memory

The configuration memory contains all of the registers which control the user zone access requirements, the key usage restrictions, and the counter usage restrictions. The device level configuration option registers are also located in configuration memory.

The ATAES132 configuration memory includes a register programmed with unique read-only die identification data at the factory. The configuration memory also contains several registers for customer information. The Configuration memory registers can always be read using the BlockRead command (see Section 7.4). The lock command is used to permanently lock the contents of the configuration memory after personalization (see Section 7.19).

See Table 2-1 for a summary of the configuration memory registers sorted by register name. See Appendix E for the configuration memory map.

2.3.1. Non-reversible Monotonic Counters

The ATAES132 includes 16 monotonic nonvolatile (EEPROM) counters which can only be incremented. They can never be decremented or reset and are protected even if the power is interrupted during an increment operation. These monotonic counters can be used to track system usage or to store small values. Keys can also be configured to prevent exhaustive attacks by limiting key usage with a counter. Each counter has an associated counter configuration register in the configuration memory.

Each counter can increment up to a value of 2,097,134 using the count command; after which they can be no longer changed. Counters attached to keys are incremented each time the key is used – when the usage counter reaches its limit the key is disabled.

On shipment from Atmel, the EEPROM locations are initialized to their lowest value. The initial value of each counter may be written to a different value prior at personalization prior to locking the configuration. See Appendix H for additional information.

2.4. SRAM Memory

The ATAES132 SRAM is used to store volatile data and status information. The ATAES132 SRAM buffers and registers are mapped into the top of the memory address space, and are accessed using standard EEPROM read/write commands. The command memory buffer is used to send extended commands to the device. The response memory buffer is used to read responses to the extended commands from the device. An IO address reset register is used to reset the buffer address pointers. The STATUS register reports the state of the device.

The VolatileKey register and the authentication status register are stored in the SRAM and are managed by the internal logic. These registers can not be directly written or read by the user.

2.4.1. Nonce

The SRAM is used to store the nonce and the random number generator (RNG) seed. The RNG seed is generated automatically by the ATAES132 as described in Section 3.6. The nonce is generated using the nonce command or the NonceCompute command. The nonce and RNG seed register are erased when the device loses power, enters the sleep state, or is reset.

2.4.2. VolatileKey

The SRAM contains a session key register named VolatileKey. This key location can be written with the KeyCompute, KeyImport, KeyLoad, or KeyTransfer commands. The VolatileKey register is erased when the device loses power, enters the sleep state, or is reset. Restrictions on the VolatileKey are established when the register is created/loaded and persist until the power is lost or the key is reloaded.

The VolatileKey can never be used to read or write the user memory or to authenticate increments of the monotonic counters. VolatileKey can only be used to perform authentication operations and to encrypt or decrypt external data. See Section 4.3 for the VolatileKey usage restrictions.

2.4.3. Command Memory Buffer

The host executes extended the ATAES132 commands by writing a command block to the command memory buffer using a standard SPI or I^{2}C write command. After the host completes its write operation to the SRAM buffer, the ATAES132 verifies the integrity of the block by checking the 16-bit checksum, and then executes the requested operation.

2.4.4. Response Memory Buffer

The host receives responses to the extended ATAES132 commands by reading a response block from the response memory buffer using a standard SPI or I^{2}C read command. The base address of the response memory buffer contains the first byte of the response packet after an ATAES132 command is processed.

2.4.5. IO Address Reset Register

Writing the IO address reset register causes the address pointers in the command memory buffer and the response memory buffer to be reset to the base address of the buffers. Writing the IO address reset register does not alter the contents of the response memory buffer, or the value of the STATUS register.

2.4.6. Device Status Register (STATUS)

The device status register is used for handshaking between the host microcontroller and the ATAES132. The host is expected to read the STATUS register before sending a command or reading a response. Reading the STATUS register does not alter the contents of the command memory buffer, the response memory buffer, or the value of the STATUS register. See Appendix G for the definition and behavior of the STATUS register.

2.4.7. Authentication Status Register

The ATAES132 authentication status register stores the result of most recent authentication attempt. The authentication status register contains the authentication KeyID, the AuthComplete status flag, and the authentication usage restriction bits. Prior to executing the Auth command, the AuthComplete status flag is set to NoAuth. After successful inbound only or mutual authentication, the AuthComplete status flag is set to YesAuth.

Table 2-4. Summary of the configuration memory registers sorted by register name[1]

Name	Description	Write	Read	Bytes
Algorithm	Algorithm ID Code (0x0000)	Never	Always	2
ChipConfig	Device level cryptographic and power up configuration options	If LockConfig = unlocked	Always	1
Counters	16 monotonic counters, each capable of counting to 2M See Appendix H	If LockConfig = unlocked	Always	128
CounterConfig	Configuration information for each Counter See Section 4.4	If LockConfig = unlocked	Always	32
DeviceNum	Atmel device number code	Never	Always	1
EEPageSize	Length in bytes of physical EEPROM page, (32, 0x20)	Never	Always	1
EncReadSize	Maximum data length in bytes for EncRead (32, 0x20)	Never	Always	1
EncWriteSize	Maximum data length in bytes for EncWrite (32, 0x20)	Never	Always	1
FreeSpace	Free memory for customer data storage	If LockConfig = unlocked	Always	96
Jedec	Atmel JEDEC manufacturer code 0x001F	Never	Always	2
KeyConfig	Configuration information for each Key See Section 4.2	If LockConfig = unlocked	Always	64
LockConfig	Controls configuration memory write access, except SmallZone. Default is the 'unlocked' state. [2]	Via Lock command only	Always	1
LockKeys	Controls key memory write access Default is the 'unlocked' state [2]	Via Lock command only	Always	1
LockSmall	Controls SmallZone register write access Default is the 'unlocked' state [2]	Via Lock command only	Always	1
LotHistory	Atmel proprietary manufacturing information	Never	Always	8
ManufacturingID	Two byte manufacturing ID code	Never	Always	2
PermConfig	Atmel factory device configuration options	Never	Always	1
SerialNum	Guaranteed unique die serial number. SerialNum is optionally included in cryptographic calculations. See Section E.2.1	Never	Always	8
SmallZone	32 byte value. The first four bytes are optionally included in cryptographic calculations. See Section E.2.25	If LockSmall = unlocked	Always	32
TempCal	Indicates the source of the TempOffset value	If LockConfig = unlocked	Always	1
TempOffset	Temperature offset for calculating the die temperature using the values returned by the temp sensor	If LockConfig = unlocked	Always	8
I^2C Addr	Selects the serial interface mode and stores the I^2C device address	If LockConfig = unlocked	Always	1
ZoneConfig	Access and usage permissions for each user zone See Section 4.1	If LockConfig = unlocked	Always	64

Notes: 1. Changes to most of the configuration registers take effect immediately, which allows the functionality to be tested during the personalization process. Changes to the I^2C Addr register take effect at the next reset, power up, or wakeup from the Sleep State.

2. The LockConfig, LockKeys, and LockSmall bytes can only be changed with the Lock command (See Section 7.19). **Warning:** The Atmel ATAES132 must always be locked by the customer prior to shipment to the end user to protect the customer secrets.

3. Security Features

All ATAES132 security features are optional. Each feature is enabled or disabled by programming configuration bits in the EEPROM configuration memory. Each user zone, key, and counter is separately and independently configured.

This section describes the ATAES132 security features and cryptographic capabilities. The functionality associated with each portion of the memory is described in Section 2.

3.1. Architecture

The ATAES132 contains all circuitry for performing authentication, encryption and decryption using keys stored securely in the internal EEPROM. Since the secrets are stored securely in the ATAES132, they do not have to be exchanged prior to executing cryptographic operations.

The ATAES132 has fixed cryptographic functionality; it is not a microcontroller and cannot accept customer firmware. The ATAES132 contains a hardware AES cryptographic engine and has a fixed command set. Although the functionality is fixed, it is also flexible because each feature is enabled or disabled by the customer by programming registers in the EEPROM configuration memory. After personalization is complete, fuses lock the configuration so that it cannot be changed.

3.1.1. AES

The ATAES132 cryptographic functions are implemented with a hardware cryptographic engine using AES in CCM mode with a 128 bit key. AES-CCM mode provides both confidentiality and integrity checking with a single key. The integrity MAC includes both the encrypted data and additional authenticate-only data bytes as described in each command definition. Each MAC is unique due to inclusion of a nonce and an incrementing MacCount register in the MAC calculation.

See Appendix I for information about how the AES computations are performed. Hyperlinks to the AES standard are provided in Appendix A.

3.1.2. Hardware Security Features

The ATAES132 chip contains physical security features to prevent an attacker from determining the internal secrets. The ATAES132 includes tamper detectors for voltage, temperature, frequency, and light as well as an active metal shield over the circuitry, internal memory encryption, and other various features. The ATAES132 physical design and cryptographic protocol are designed to prevent or significantly complicate most algorithmic, timing, and side channel attacks.

3.2. Authentication

The authentication commands utilize AES-CCM to generate or validate a MAC value computed using an internally stored key. The command set supports both one way and mutual authentication. One ATAES132 device can generate packets for authentication of a second ATAES132 device containing the same key. The internal authentication status register remembers only the most recent authentication attempt. A user zone can be configured to require prior authentication of a designated key before access to the user zone is permitted.

3.2.1. Key Authentication

Individual keys can be configured to require a successful authentication prior to use. This requirement can be used to prevent some kinds of exhaustive attacks on the keys. The authentication requirement can be chained to require authentication of several keys prior to allowing a particular operation. The internal authentication status registers remember only the most recent authentication attempt.

3.3. Encrypted Memory Read/Write

A user zone can be configured to require an AES-CCM encryption for the EEPROM read or write operations. If encryption is required for write access, then the MAC is validated before the received (encrypted) data is written to the EEPROM. If encryption is required for read access, then the ATAES132 encrypts data when it is read from the internal EEPROM and generates an associated integrity MAC.

3.4. Data Encryption/Decryption

A key can be configured to allow encryption/decryption of small packets of data using AES-CCM with an internally stored key. The encrypt command encrypts 16 or 32 bytes of plaintext data provided by the host; the encrypted data and MAC are returned to the host. The decrypt command decrypts 16 or 32 bytes of encrypted data after verifying the MAC; the data is returned to the host only if the MAC is valid. When these commands are used, none of the data is stored in the internal EEPROM.

3.4.1. AES-ECB Encryption/Decryption

A key can be configured to allow AES-ECB mode operations using the legacy command. A single AES-ECB operation is performed using an internally stored key and the 16 byte input packet received with the AES-ECB command. The 16 byte result is returned to the host. No input or output formatting is performed by this command, and no data is stored in the internal EEPROM.

3.5. Keys

The ATAES132 securely stores sixteen 128 bit keys in the EEPROM. Keys can only be used for the cryptographic functions enabled in the ZoneConfig, CounterConfig or KeyConfig register bits in the configuration memory. Key values can never be read from the ATAES132 under any circumstances. Any key can be used with any user zone.

A seventeenth key register in the internal SRAM can be used for session keys.

See section 7.11 for the EncWrite command. See section 7.19 for the lock command.

3.5.1. Key Management

The key registers can be written with plaintext data or with encrypted data before the key memory is locked. After the key memory is locked, a key register can only be updated if the corresponding KeyConfig register allows updates.

Several key management commands are available for updating or generating the keys:

1. An encrypted key provided by the host can be written to an internal key register after validating the MAC. The KeyImport command and KeyLoad command perform this function.
2. Plaintext data provided by the host can be encrypted and returned to the host along with the MAC; this packet can be used as the encrypted key input to another ATAES132 device. The KeyExport command performs this function.
3. The internal random number generator can be used to create a key for use as a session key or for storage in an internal key register. The new key can also be encrypted and returned to the host for use as the encrypted key input to another ATAES132 device. The KeyCompute command and KeyExport command perform this function.
4. The contents of the session key register can be encrypted and returned to the host along with the MAC. The KeyExport command performs this function.
5. Keys stored in the user memory can be transferred to an internal key register or used as a session key. A user zone configured as extended key memory can be used to store eight keys. The KeyTransfer command performs this function.

3.5.2. Limited Use Keys

To prevent exhaustive attacks on the keys, the ATAES132 can be configured to limit the key usage with a monotonic counter. If a key is configured with a usage counter, then the following steps are performed for any command using that key:

1. Read the counter from memory to check if the count has reached the maximum count value
2. If the maximum count has been reached, then the command is not executed and an error code is returned
3. If the maximum count has not been reached, then the counter is incremented and the command is executed

By default, the counters are configured to allow two million counts, allowing two million operations using a key with the usage limits enabled. Atmel recommends that the customer configure key usage counters at personalization to a smaller number; the appropriate key usage limit is dependent on the application. See Appendix H for additional information.

3.5.3. Secure Personalization

The ATAES132 is designed to allow personalization of keys using encryption, so the secret key values cannot be determined by a third party. AES encryption of the keys prevents them from being determined by observation of data communicated to or from the ATAES132.

A transport key is programmed into the KeyID 00 register by Atmel during the chip manufacturing process. This transport key is securely exchanged between the customer and Atmel. During personalization, the secret keys are encrypted using the transport key before being written to the ATAES132.

Atmel also offers a secure personalization service at additional cost which uses a hardware security module (HSM) to store the customer secrets.

3.5.3.1. Key Diversification

Atmel recommends that each unit should contain one or more unique keys to minimize the potential impact of cloning. The keys stored in the ATAES132 should be a cryptographic combination of a root secret not stored in the chip along with the unique ATAES132 SerialNum register value. The host must have a secure place to store the root secret to protect the integrity of the diversified keys.

It may also be beneficial for the ATAES132 devices to contain secrets for validating the authenticity of the host. These secrets may need to be the same on all ATAES132 devices for a particular application to permit any client to validate any host.

3.6. Random Numbers

The ATAES132 includes a high quality random number generator (RNG) for nonce generation, child key creation, and for the general random number generation. The ATAES132 commands can generate random numbers for internal or external use. Sixteen byte random numbers for external use are generated using the internal RNG and the AES engine as described in NIST SP800-90.

The RNG can be used to generate the nonce for cryptographic operations. A mechanism is also provided to synchronize the nonces in two ATAES132 devices using random numbers generated by both devices. A key can be configured to require that cryptographic operations using the key use a nonce generated with the internal RNG.

3.6.1. Random Number Generation

The RNG architecture includes both a hardware random number generator and a stored random seed. On power up, the stored seed is read from the EEPROM, cryptographically combined with the hardware random number generator output, and then stored in SRAM. Whenever a random number is required, this SRAM seed is cryptographically combined with the hardware random number generator output and the optional input seed to create both a new SRAM seed and the random number.

For highest security, the EEPROM seed should be updated every power cycle in which the RNG is used. However the EEPROM seed register has a maximum life expectancy of 100,000 writes per unit. The host system is expected to manage the EEPROM seed by using the command mode option to suppress automatic EEPROM seed updates.

4. Security Configuration Registers

4.1. User Zone Configuration

Access permissions to each user zone are controlled by the ZoneConfig registers in the configuration memory. There is one ZoneConfig register for each user memory zone.

Table 4-5. Definition of the ZoneConfig Register bits(1)(2)

ZoneConfig Field	Byte	Bit	Description
AuthRead	0	0	If 1b, then authentication is required to read data If 0b, then authentication is not required to read data
AuthWrite	0	1	If 1b, then authentication is required to write data If 0b, then authentication is not required to write data
EncRead	0	2	If 1b, then encryption is required to read data If 0b, then encryption is not required to read data
EncWrite	0	3	If 1b, then encryption is required to write data If 0b, then encryption is not required to write data
WriteMode	0	4 to 5	If 00b, then this zone is permanently read/write If 01b, then this zone is permanently read-only If 10b, then the ReadOnly byte determines if writes are permitted If 11b, then the ReadOnly byte determines if writes are permitted and the Lock command must include an authenticating MAC calculated using the KeyID stored in ZoneConfig[UZ].WriteID.
UseSerial	0	6	If UseSerial = 1b and EncWrite = 1b, then the SerialNum must be included in EncWrite operations. If EncWrite = 0b, then this bit is ignored.
UseSmall	0	7	If UseSmall = 1b and EncWrite = 1b, the first 4 bytes of SmallZone must be included in EncWrite operations. If EncWrite = 0b, this bit is ignored.
ReadID	1	0 to 3	KeyID which is used to encrypt data read from this zone The same key is used to generate the MAC
AuthID	1	4 to 7	KeyID which is used for inbound authentication before access is permitted
Reserved	2	0 to 3	Reserved for future use. All bits must be 0b
WriteID	2	4 to7	KeyID which is used to decrypt data written to this zone The same key is used to verify the MAC
ReadOnly	3	0 to 7	The contents of this byte are ignored unless WriteMode contains 10b or 11b If 0x55, then the User Zone is read/write If any other value, then the User Zone is Read-Only This byte can be updated after the Configuration Memory is locked by using the Lock command (See Section 7.19.)

Notes: 1. Most changes to the ZoneConfig registers take effect immediately. Changes to the AuthRead and EncRead bits do not affect the SPI or I^{2}C Read command until the next reset or power up.

2. **Warning:** The Atmel ATAES132 must always be locked by the customer prior to shipment to the end user to protect the customer secrets. See Section 7.19 for the lock command.

4.2. Key Configuration

Restrictions on key usage are controlled by the KeyConfig registers in the configuration memory. There is one KeyConfig register for each key.

Table 4-6. Definition of the KeyConfig register bits(1)(2)(4)

KeyConfig Field	Byte	Bit	Description
ExternalCrypto	0	0	If 1b, then the key can be used with the encrypt and decrypt commands(3) If 0b, then the encrypt and decrypt commands are prohibited
InboundAuth	0	1	If 1b, then the key can only be used by the Auth command for Inbound Only or mutual authentication. The key can not be used by any other command, but KeyID can be the target of a key management command. If 0b, then key can be used for any purpose not prohibited by another KeyConfig bit, including Outbound Only authentication
RandomNonce	0	2	If 1b, then operations using this key require a random nonce (see Section 7.20) If 0b, then the nonce is not required to be random
LegacyOK	0	3	If 1b, then this key can be used with the legacy command If 0b, then the key cannot be used with the legacy command
AuthKey	0	4	If 1b, then this key requires prior authentication using the KeyID stored in LinkPointer If 0b, then prior authentication is not required
Child	0	5	If 1b, then key is permitted to be the target of a KeyCompute or KeyLoad command If 0b, then this use is prohibited
Parent	0	6	If 1b, then key may be used as the VolatileKey parent by the KeyCompute or KeyLoad commands. This key may also be used as the decrypt key by the KeyImport command when the target key is the VolatileKey. (see Section 4.3) If 0b, then this use is prohibited
ChangeKeys	0	7	If 1b, then key updates are permitted after locking. The new key is written using the EncWrite command with a MAC generated with the current value of key. (see Section 7.11) If 0b, then key updates with EncWrite command are prohibited
CounterLimit	1	0	If 1b, usage count limits are enabled for this key (see CounterNum) If 0b, then there are no usage limits
ChildMac	1	1	If 1b, then an input MAC is required to modify this key using the KeyCompute command If 0b, the KeyCompute command does not require an input MAC (it will be ignored if provided)
AuthOut	1	2	If 1b, then I^2C Auth signaling is enabled for this key (see Section J.5) If 0b, then I^2C Auth signaling is disabled for this key
AuthOutHold	1	3	If 1b, the I^2C AuthO output state is unchanged when an authentication reset is executed using this key If 0b, then the I^2C AuthO output is reset when an authentication reset is executed using this key (see Section J.5)
ImportOK	1	4	If 1b, then this key is permitted to be the target of a KeyImport command If 0b, then the KeyImport command is prohibited
ExportAuth	1	5	If 1b, then the KeyExport and KeyCompute commands require prior authentication using the KeyID stored in LinkPointer If 0b, then prior authentication is not required

KeyConfig Field	Byte	Bit	Description
TransferOK	1	6	If 1b, then this key is permitted to be the target of a KeyTransfer command (See Section 7.17) If 0b, then the KeyTransfer command is prohibited
AuthCompute	1	7	If 1b, then this key can be used with the AuthCompute command If 0b, then the key cannot be used with the AuthCompute command
LinkPointer	2	0 to 3	For child keys, stores the ParentKeyID For all other keys, the KeyID of the authorizing key (see AuthKey)
CounterNum	2	4 to 7	Stores the CntID of the monotonic counter attached to this key for usage limits or for MAC calculation. MAC calculations will include the counter if command mode bit 5 is 1b even if key usage limits are disabled.
Reserved	3	0 to 7	Reserved for future use. All bits must be 0b.

Notes:
1. Changes to the KeyConfig registers take effect immediately, which allows the functionality to be verified during the personalization process
2. **Warning:** The Atmel ATAES132 must always be locked by the customer prior to shipment to the end user to protect the customer secrets. See Section 7.19 for the lock command.
3. **Warning:** Since the encrypt command does not include an input MAC, the encrypt command can exhaustively be run with selected input data to attack the key. Requiring authentication prior to allowing encryption makes these attacks more difficult. To require prior authentication, the AuthKey and RandomNonce bits must be set to 1b.
4. A key can be disabled by setting KeyConfig[KeyN].AuthKey to 1b and KeyConfig[KeyN].LinkPointer to contain "KeyN", where KeyN = KeyID of the key being configured

4.3. VolatileKey Configuration

There is a seventeenth key register named VolatileKey that has a KeyID of 0xFF and is stored in the internal SRAM. This key location can be written only with the KeyCompute command (see Section 7.13) or KeyLoad command (see Section 7.16). The contents of the VolatileKey register are erased when the device is powered down, enters the sleep state or is reset.

When the VolatileKey register is loaded, restrictions are placed on its usage, which persist until the power is lost or the key is reloaded. The definition of the VolUsage field is shown in Table 4-3.

Table 4-7. Definition of the VolUsage field bits in the KeyCompute or KeyLoad command at VolatileKey creation

VolUsage Field Name	Byte	Bit	Description
AuthOK	0	0	If 1b, then the Auth command can be run using this key If 0b, then the Auth command is prohibited
EncryptOK	0	1 to 2	If 00b, then the encrypt command is prohibited If 01b, then the encrypt command can be run using this key without a prior authentication [(1)] If 10b or 11b, then encrypt command can be run using this key only with a prior authentication using this key [(1)]
DecryptOK	0	3	If 1b, then the decrypt command can be run using this key If 0b, then the decrypt command is prohibited
RandomNonce	0	4	If 1b, then operations using this key require a random nonce (See Section 7.20). If 0b, a fixed (input only) nonce is permitted
AuthCompute	0	5	If 1b, then the AuthCompute command can be run using this key If 0b, then the AuthCompute command is prohibited
LegacyOK	0	6	If 1b, then the legacy command can be run using this key If 0b, then the legacy command is prohibited
ExportOK	0	7	If 1b, then the VolatileKey can be encrypted and exported using the KeyExport command If 0b, then export of VolatileKey is prohibited
WriteCompute	1	0	If 1b, then the WriteCompute command can be run using this key If 0b, then the WriteCompute command is prohibited
DecRead	1	1	If 1b, then the DecRead command can be run using this key If 0b, then the DecRead command is prohibited
Reserved	1	2 to 7	Reserved for future use. All bits must be 0b.

Note: 1. **Warning:** Since the encrypt command does not include an input MAC, the encrypt command can be exhaustively run with selected input data to attack the VolatileKey. Requiring authentication prior to allowing encryption makes these attacks more difficult. To implement this, the Auth, and RandomNonce bits must be set to 1b, and the encrypt bits must be set to 10b or 11b when the VolatileKey is created.

4.4. Monontonic Counter Configuration

The CounterConfig register imposes restrictions on the usage of the counter command with a counter (See Section 7.5). There is one CounterConfig register for each counter. Each counter can increment up to a value of 2,097,134 using the count command; after which they can be no longer changed. See Appendix H for additional counter information.

The CounterConfig bits have no impact on the functionality of a Key Usage Counter. If a Counter is identified in a KeyConfig register (see Section 4.2) as a Key Usage Counter, then the Counter will increment each time the Key is used. The CounterConfig[CntID].IncrementOK bit is typically set to 0b to prohibit the Counter Command from incrementing a Key usage Counter.

Table 4-8. Definition of the CounterConfig register bits(1)(2)

CounterConfig Field	Byte	Bit	Description
IncrementOK	0	0	If 1b, then increments using the counter command are permitted If 0b, then increments using the counter command are prohibited
RequireMAC	0	1	If 1b, then the increment operation requires an input MAC If 0b, then an input MAC is prohibited
Reserved	0	2 to 7	Reserved for future use. All bits must be 0b.
IncrID	1	0 to 3	KeyID of the key used to generate the counter command input MAC for increment operations
MacID	1	4 to7	KeyID of the key used to generate the counter command output MAC for counter read operations

Note: 1. Changes to the CounterConfig registers take effect immediately, allowing the functionality to be verified during the personalization process

2. **Warning:** The Atmel ATAES132 must always be locked by the customer prior to shipment to the end user to protect the customer secrets. See Section 7.19 for the lock command.

5. Standard Serial EEPROM Read and Write Commands

This section provides a summary of the operations that can be performed using the standard Serial EEPROM read and write commands. For detailed information see the specification sections that are referenced below.

Table 5-9. Standard Serial EEPROM read and write commands

Name	Description
Read	The read command is used to read cleartext from the user zones, to retrieve a response by reading the response memory buffer, or to read the STATUS register
Write	The write command is used to write cleartext to unrestricted memory, or to send a command by writing the command packet to the command memory buffer. The write command is also used to write the IO address reset register.

5.2. Read

The ATAES132 supports the standard Serial EEPROM commands to read from the user memory. All bytes in the user memory address space may be read, however, only bytes in the user zones in which neither authentication nor encryption is required will return the actual data from the memory. All other locations will return the value 0xFF. See Appendix J for the I^2C read command and Appendix K for the SPI read command information.

When a read command is received, the device looks at the AuthRead and EncRead bits in the ZoneConfig register for the user zone to determine whether to return 0xFF or the EEPROM data. If the EncRead bit is 1b or the AuthRead bit is 1b, then 0xFF will always be returned.

If the ZoneConfig AuthRead bit is 1b and the EncRead is 0b, then the BlockRead command must be used to read the user zone (see Section 7.4). If the EncRead bit is 1b, then the EncRead command must be used to read the user zone (see Section 7.9).

The standard SPI and I^2C Read commands can be used to read any number of bytes in a single operation. Read operations can cross EEPROM page boundaries.

5.2.1. Read the Response Memory Buffer

The host sends ATAES132 commands to the device by writing the command packet to the command memory buffer using a standard SPI or I^2C write command. The ATAES132 processes the command packet and places the response in the response memory buffer. The host retrieves the response by reading the response packet using a standard SPI or I^2C read command. See Appendix D for additional information. See Appendix G for examples.

When any error occurs the EERR bit of the STATUS register is set to 1b to indicate an error. See section G.1 for more information.

5.2.2. Read the Key Memory or Configuration Memory

Reading the key memory is never allowed.

The read command can never be used to read data from the configuration memory. The BlockRead command is used to access the configuration memory (see Section 7.4).

If a standard SPI or I^2C read command is used within the configuration memory or key memory address space, then 0xFF will be returned for each byte. 0xFF is also returned for address locations which do not physically exist. The EERR bit of the STATUS register is set to 1b if 0xFF was substituted for any byte returned by a read command. See section G.1 for more information.

5.2.3. Read the STATUS Register

The host reads the STATUS register by reading address 0xFFF0. In SPI Interface mode the host can also read STATUS using the RDSR command. See Appendix G for detailed information and examples.

5.3. Write

The ATAES132 supports the standard Serial EEPROM commands to write to unrestricted user memory (AuthWrite and EncWrite are both 0b). See Appendix J for the I^{2}C write command and Appendix K for the SPI write command information. The ATAES132 is capable of writing 1 to 32 bytes on a single physical page with each write operation.

The write command can only write data to a single user zone; the data can not span multiple user zones. The write command can only write data to a single EEPROM page; the data can not cross page boundaries. The EERR bit of the STATUS register is set to 1b to indicate an error if a prohibited write is attempted. See section G.1 for more information.

5.3.1. Write the Command Memory Buffer

The host sends ATAES132 commands to the device by writing the command packet to the command memory buffer using a standard SPI or I^{2}C write command. The ATAES132 processes the command packet and places the response in the response memory buffer. The host retrieves the response by reading the response packet using a standard SPI or I^{2}C read command. See Appendix D for additional information. See Appendix G for examples.

When any error occurs, either the EERR bit or the CRCE bit of the STATUS register is set to 1b to indicate an error. See section G.1 for more information.

5.3.2. Write the IO Address Reset Register

The host can reset the pointer in the command memory buffer and the response memory buffer by writing to address 0xFFFE. See Section D.4 for additional information.

5.3.3. Write the Key Memory or Configuration Memory

The ATAES132 supports standard Serial EEPROM commands to write to the configuration memory or the key memory prior to locking. The ATAES132 is capable of writing 1 to 32 bytes on a single physical page with each write operation.

Note: Partial writes to key registers are prohibited

If LockKeys has a value of 0x55 (unlocked) and the address points to key memory, then the starting address must be the first byte of a key register and 16 bytes of cleartext data must be sent. If these conditions are not satisfied, then an error response will be generated and the EEPROM will remain unchanged.

If LockConfig has a value of 0x00 (locked) and the address points to the configuration memory, then a write command will generate an error and the EEPROM will be unchanged.

If LockConfig has a value of 0x55 (unlocked), then the user zone write restrictions imposed by ZoneConfig are enforced, but can be changed. Atmel does not recommend writing secret data into the user zones prior to locking of the configuration memory due to the fact that an attacker can change the ZoneConfig bits to allow read of the user zone if the configuration memory is unlocked.

When any error occurs, either the EERR bit or the CRCE bit of the STATUS register is set to 1b to indicate an error. See section G.1 for more information. See the lock command (Section 7.19) for additional information.

6. Atmel ATAES132 Commands

6.1. Command Block and Packet

The host sends ATAES132 extended commands to the device in a block of at least nine bytes. The ATAES132 responses are returned to the host in a block of at least four bytes.

Table 6-10. The command and response blocks are constructed in the following manner:

Byte #	Name	Meaning
0	Count	Number of bytes to be transferred to the device in the block, including count, packet, and checksum. This byte will always have a value of N.
1 to (N-3)	Packet	Command, parameters and data, or response. Data is transmitted in the byte order shown in the command definitions.
N-2, N-1	Checksum	Atmel CRC-16 verification of the count and packet bytes. See Appendix M for additional information and examples.

Table 6-11. The input command packet within the command block is constructed in the following manner:

Byte #	Name	Meaning
1	Opcode	The command code
2	Mode	Command modifier
3, 4	Param1	First command parameter
5, 6	Param2	Second command parameter
7+	Data	Optional input data

Table 6-12. The response packet within the response block is constructed in the following manner:

Byte #	Name	Meaning
1	ReturnCode	The command return code (see Section 6.3)
2+	Data	Optional output data

Table 6-13. When an error occurs the response packet contains only an error code:

Byte #	Name	Meaning
1	ReturnCode	The error code (see Section 6.3)

The host sends ATAES132 commands to the device by writing the command block to the Command Memory Buffer using a standard SPI or I^2C write command. ATAES132 processes the command packet and places the response block in the Response Memory Buffer. The host retrieves the response by reading the response block using a standard SPI or I^2C read command. If the host reads beyond the end of the block, then 0xFF is returned.

6.2. Command Summary

Table 6-6 shows the command set in alphabetical order by command name. Table 6-5 shows the command set sorted by the Opcode value. See Section 7 for the ATAES132 command definitions.

Table 6-14. Extended Atmel ATAES132 command set, sorted by Opcode value

Opcode[1]	Name	Description
0x00	Reset	Resets the device, clearing the cryptographic status
0x01	Nonce	Generates a 128 bit nonce using the internal random number generator for use by the cryptographic commands. This command can also be used to write a host nonce directly into the Nonce register.
0x02	Random	Returns a 128 bit random number from the internal random number generator
0x03	Auth	Performs one-way or mutual authentication using the specified key
0x04	EncRead	Encrypts 1 to 32 bytes of data from user memory and returns the encrypted data and integrity MAC
0x05	EncWrite	Writes 1 to 32 bytes of encrypted data into the user memory or key memory after verifying the integrity MAC
0x06	Encrypt	Encrypts 16 or 32 bytes of plaintext data provided by the host
0x07	Decrypt	Decrypts 16 or 32 bytes of data provided by the host after verifying the integrity MAC
0x08	KeyCompute	Generates a random number, stores it in key memory and returns the encrypted key to the host
0x09	KeyLoad	Writes an encrypted key to key memory after verifying the integrity MAC
0x0A	Counter	Increments a monotonic counter and/or returns the current counter value
0x0B	Crunch	Processes a seed value through the internal crunch engine. This function is used to detect clones.
0x0C	Info	Returns device information: the MacCount, authentication status, or the hardware revision code
0x0D	Lock	Permanently locks the configuration memory or key memory. Locked memory can never be unlocked.
0x0E	TempSense	Measures the die temperature
0x0F	Legacy	Performs a single AES-ECB mode operation on 16 bytes of data provided by the host
0x10	BlockRead	Reads 1 to 32 bytes of data from user memory or the configuration memory. Returns cleartext data.
0x11	Sleep	Places the device in the sleep state or standby state to reduce power consumption
0x13	NonceCompute	Generates a nonce in a manner which allows two ATAES132 devices to have identical nonce values
0x14	AuthCompute	Computes the input MAC required to execute the Auth command or to increment a counter using the counter command on a second ATAES132 device
0x15	AuthCheck	Checks the output MAC generated by the Auth command or by reading a counter using the counter command on a second ATAES132 device
0x16	WriteCompute	Encrypts data and generates the input MAC required to execute the EncWrite command
0x17	DecRead	Checks the output MAC and decrypts data which was encrypted by the EncRead command
0x18	KeyExport	Encrypts a key for export to an ATAES132 device. Optionally generates the key being exported.
0x19	KeyImport	Decrypts and writes a key which was output by the KeyExport command or KeyCompute command
0x1A	KeyTransfer	Transfers a key from user memory into the key memory or into the VolatileKey register

Note: 1. The most significant three bits of the command Opcode may contain any value; these three bits are ignored by the Atmel ATAES132 command decoder

Table 6-15. Extended ATAES132 command set, sorted by command name

Opcode[1]	Name	Description
0x03	Auth	Performs one-way or mutual authentication using the specified key
0x15	AuthCheck	Checks the output MAC generated by the Auth command or by reading a counter using the counter command on a second ATAES132 device
0x14	AuthCompute	Computes the input MAC required to execute the Auth command or to increment a counter using the counter command on a second ATAES132 device
0x10	BlockRead	Reads 1 to 32 bytes of data from user memory or the configuration memory. Returns cleartext data.
0x0A	Counter	Increments a monontonic counter and/or returns the counter value
0x0B	Crunch	Processes a seed value through the internal crunch engine. This function is used to detect clones.
0x17	DecRead	Checks the output MAC and decrypts data which was encrypted by the EncRead command
0x07	Decrypt	Decrypts 16 or 32 bytes of data provided by the host after verifying the integrity MAC
0x04	EncRead	Encrypts 1 to 32 bytes of data from user memory and returns the encrypted data and integrity MAC
0x06	Encrypt	Encrypts 16 or 32 bytes of plaintext data provided by the host
0x05	EncWrite	Writes 1 to 32 bytes of encrypted data into the user memory or key memory after verifying the integrity MAC
0x0C	Info	Returns device information: the MacCount, authentication status, or the hardware revision code
0x08	KeyCompute	Generates a random number, stores it in key memory and returns the encrypted key to the host
0x18	KeyExport	Encrypts a key for export to an ATAES132 device. Optionally generates the key being exported.
0x19	KeyImport	Decrypts and writes a key which was output by the KeyExport command or KeyCompute command
0x09	KeyLoad	Writes an encrypted key to key memory after verifying the integrity MAC
0x1A	KeyTransfer	Transfers a key from user memory into the key memory or into the VolatileKey register
0x0F	Legacy	Performs a single AES-ECB mode operation on 16 bytes of data provided by the host
0x0D	Lock	Permanently locks the configuration memory or key memory. Locked memory can never be unlocked.
0x01	Nonce	Generates a 128 bit nonce using the internal random number generator for use by the cryptographic commands. This command can also be used to write a host nonce directly into the Nonce register.
0x13	NonceCompute	Generates a nonce in a manner which allows two ATAES132 devices to have identical nonce values
0x02	Random	Returns a 128 bit random number from the internal random number generator
0x00	Reset	Resets the device, clearing the cryptographic status
0x11	Sleep	Places the device in the sleep state or standby state to reduce power consumption
0x0E	TempSense	Measures the die temperature
0x16	WriteCompute	Encrypts data and generates the input MAC required to execute the EncWrite command

Note: 1. The most significant three bits of the command Opcode may contain any value; these three bits are ignored by the Atmel ATAES132 command decoder

6.3. ReturnCode

The response packet for each ATAES132 command includes a ReturnCode to report success or failure to the host. The first four error codes (0x01 through 0x08) may occur concurrently with other codes.

The reset command and the sleep command do not generate a ReturnCode because they do not generate a response packet. All other ATAES132 commands generate a ReturnCode.

Table 6-16. ReturnCode field sorted by Value

Value	Name	Notes
0x00	Success	No errors
0x02	BoundaryError	Crossed a page boundary for a write, BlockRead or EncRead. Crossed a key register boundary for a write or EncWrite
0x04	RWConfig	Access to the specified user zone is not permitted due to the configuration or internal state
0x08	BadAddr	Attempted to write locked memory, or address is not implemented, or address is illegal for this command
0x10	CountErr	Counter limit reached, or count usage error, or restricted key error
0x20	NonceError	Nonce invalid or not available. Nonce not generated with internal RNG. MacCount limit has been reached.
0x40	MacError	Missing input MAC, or MAC compare failed
0x50	ParseError	Bad opcode, bad mode, bad param, invalid length, or other encoding failure
0x60	DataMatch	EEPROM post-write automatic data verification failed due to data mismatch
0x70	LockError	Lock command contained bad checksum or bad MAC
0x80	KeyErr	Key not permitted to be used for this operation, or wrong key was used for operation. Prior authentication has not been performed. Other authentication error or other key error.
0x90	TempSenseErr	Temperature sensor timeout error

If ReturnCode has any value other than 0x00, no additional data will be returned by ATAES132. If the ReturnCode is greater than zero for any command that performs cryptographic operations, then the nonce will be invalidated.

7. Command Definitions

The ATAES132 extended command definitions are described in this section. The commands are listed in alphabetical order by command name. The standard Serial EEPROM read and write commands are in Section 5 which are not included in this section. The cryptographic operations performed by the ATAES132 commands are described in Appendix I.

7.1. Auth Command

The Auth command performs a one-way or mutual authentication using AES-CCM. The Auth command options are shown in Table 7-1. The nonce register value is used as the CCM nonce for all Auth command MAC calculations.

- **Mutual authentication**
  The InMAC is verified, and upon success, an OutMAC is calculated and returned to the host. The AuthComplete status flag is set to YesAuth if the InMAC is verified.
- **Outbound only authentication**
  The OutMAC is calculated and output to the host. The AuthComplete status flag is set to NoAuth. The Outbound only authentication is also known as challenge-response authentication.
- **Inbound only authentication**
  The InMAC value is verified, and the success or failure is reported to the host. The AuthComplete status flag is set to YesAuth if the InMAC is verified.
- **Authentication reset**
  The AuthComplete status flag is set to NoAuth.

Table 7-17. Auth command options

Mode bit 1	Mode bit 0	Description	InMAC	OutMAC
1b	1b	Mutual authentication	Required	Generated
1b	0b	Outbound only authentication	Prohibited	Generated
0b	1b	Inbound only authentication	Required	No
0b	0b	Authentication reset	Prohibited	No

If a MAC is required or will be generated by the Auth command, then a valid nonce is required. If the KeyConfig[AKeyID].RandomNonce bit is 1b, then the nonce must be random.

The AuthCompute command can be used to generate the InMac required for inbound only authentication, or mutual authentication (see Section 7.3). The AuthCheck command can be used to validate the OutMac (see Section 7.2).

In the I^2C interface mode, the Auth command can also used for Auth signaling. See Section J.5 for the Auth signaling specifications.

7.1.2. Authentication Status Register

The authentication status register contains the AKeyID, the AuthComplete status flag, and the usage bits. Prior to executing the Auth command, the AuthComplete status flag is set to NoAuth. If the InMAC is successfully verified in the inbound only or mutual authentication mode, then the AuthComplete status flag is set to YesAuth.

The ATAES132 authentication status register only stores the result of the most recent authentication attempt. If there is a parsing or execution error then the prior authentication, status will be lost.

7.1.3. Authentication Usage

The usage field (Param2) controls which operations are permitted with a successful inbound only or mutual authentication (see Table 7-2). If Param2 is 0x0000 the AuthComplete flag is set to NoAuth but the authentication outputs are generated. Param2 is ignored if the outbound only authentication is performed.

Table 7-18. Definition of the Auth command usage field (Param2)

Byte #	Bit #	Name	Notes
0	0	ReadOK	If 1b, then the read and EncRead commands are enabled for the user zone reads after successful authentication. If 0b, then the read and EncRead commands are prohibited for the user zone reads if authentication is required in ZoneConfig[UZ]. (see Section 4.1)
0	1	WriteOK	If 1b, then the write and EncWrite commands are enabled for the user zone writes after successful authentication. If 0b, then the write and EncWrite commands are prohibited for user zone writes if authentication is required in ZoneConfig[UZ]. (see Section 4.1)
0	2	KeyUse	If 1b, then if a key requires authentication (KeyConfig[AKeyID].AuthKey is 1b), the EncRead, EncWrite, encrypt, decrypt, legacy, KeyCompute, and KeyLoad commands are enabled after successful authentication. If 0b, then the EncRead, EncWrite, encrypt, decrypt, legacy, KeyCompute, and KeyLoad commands using the authenticated key are prohibited after authentication. (see Section 4.2)
0	3-7	Zero	Reserved. Must be 0b
1	0:7	Zero	Reserved. Must be 0x00

If the AKeyID is VolatileKey, then VolUsage.AuthOK must be 1b when the key is loaded, or authentication will fail.

Table 7-19. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Auth	1	0x03
Mode	Mode	1	Bit 0 and 1: If 11b, then perform mutual authentication If 10b, then perform outbound only authentication If 01b, then perform inbound only authentication If 00b, then perform authentication reset Bits 2, 3, 4: Reserved. Must be 0b Bit 5: If 1b, include the associated usage counter in the authentication Bit 6: If 1b, include the SerialNum in the authentication Bit 7: If 1b, include the first four bytes of the SmallZone in the authentication
Param1	AKeyID	2	Upper byte is always 0x00. Lower byte is the pointer to the key. Legal values: 0x00 to 0x0F, 0xFF.
Param2	Usage	2	Authentication usage restrictions. Ignored if mode bits 0 and 1 are 00b or 10b.
Data	InMac	0 or 16	Input MAC to be verified (see Section I.3)

Table 7-20. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutMac	0 or 16	If an output MAC generation was required (and any optional input MAC verification succeeded), then a 16 byte MAC will be returned.

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.2. AuthCheck Command

The AuthCheck command is used to check the OutMAC generated by the Auth command or the counter command on a second ATAES132 device. This command can not check MACs created by other commands.

To use this command the nonce must be identical on both devices (see Section 7.21.1) and the MacCount must have the same value. Both devices must also contain identical key values, but it is not necessary for the KeyID on the origin device to match the KeyID on the destination device. In this section the device which generates the MAC is referred to as the origin device; the device checking the MAC is referred to as the destination device.

If mode bit 5, 6, or 7 is 1b, then the associated usage counter, SerialNum register value, or the first four bytes of the SmallZone register in the SecondBlock field must match the values on the origin device. The ManufacturingID register must be identical on both devices, since it is always included in the MAC calculation.

A valid nonce is required to run the AuthCheck command. If the KeyConfig[MacKeyID].RandomNonce bit is 1b, then the nonce must be random.

The AuthCheck command always sets the AuthComplete status flag to NoAuth.

Table 7-21. Input parameters

	Name	Size (Bytes)	Notes
Opcode	AuthCheck	1	0x15
Mode	Mode	1	The value of this field must match the mode field value used when executing the Auth command or the counter command which generated the OutMAC on the origin device
Param1	MacKeyID	2	Upper byte is always 0x00. Lower byte is the pointer to the key. Legal values: 0x00 to 0x0F, 0xFF
Param2	Zero	2	Always 0x0000
Data1	FirstBlock	11	The value of this field must match the first authenticate-only block used to calculate the MAC on the origin device
Data2	SecondBlock	16	The value of this field must match the second authenticate-only block used to calculate the MAC being checked on the origin device. If mode bits 5, 6, and 7 are 0b, then this field must be present, but is ignored.
Data3	InMac	16	MAC to be checked

Table 7-22. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.3. AuthCompute Command

The AuthCompute command is used to compute a MAC which will be used to execute the Auth command or the counter command on a second ATAES132 device.

To use this command, the nonce must be identical on both devices (see Section 7.21.1) and the MacCount must have the same value. Both devices must also contain identical key values, but it is not necessary for the KeyID on the origin device to match the KeyID on the destination device. In this section, the device which generates the MAC is referred to as the origin device, and the device checking the MAC is referred to as the destination device.

If mode bit 5, 6, or 7 is 1b, then the associated usage counter, SerialNum register value, or the first four bytes of the SmallZone register in the SecondBlock field must match the values on the destination device. The ManufacturingID register must be identical on both devices, since it is always included in the MAC calculation.

A valid nonce is required to run the AuthCompute command. If the KeyConfig[MacKeyID].RandomNonce bit is 1b, then the nonce must be random.

The AuthCompute command always sets the AuthComplete status flag to NoAuth.

Table 7-23. Input parameters

	Name	Size (Bytes)	Notes
Opcode	AuthCompute	1	0x14
Mode	Mode	1	The value of this field must match the mode field value to be used when executing the Auth command or the counter command on the destination device
Param1	MacKeyID	2	Upper byte is always 0x00. Lower byte is the pointer to the key. Legal values: 0x00 to 0x0F, 0xFF
Param2	Zero	2	Always 0x0000
Data1	FirstBlock	11	The value of this field must match the first authenticate-only block to be used when executing the Auth command or the counter command on the destination device
Data2	SecondBlock	16	The value of this field must match the second authenticate-only block to be used when executing the Auth command or the counter command on the destination device. If mode bits 5, 6, and 7 are 0b, then this field must be present, but is ignored.

Table 7-24. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutMac	16	The 16 byte MAC

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.4. BlockRead Command

The BlockRead command reads 1 to 32 bytes of plaintext data from a user zone or the configuration memory. This command differs from the standard Serial EEPROM read commands, since it can read the configuration memory. In addition, this command returns an error code if the read is unsuccessful. No encryption is performed by the BlockRead command; the EncRead command must be used for encrypted reads (see Section 7.9).

The BlockRead command can only read data from a single EEPROM page – the requested data can not cross page boundaries (see Section B.2). All bytes within the configuration memory can be read with the BlockRead command. If any part of the requested data lies in unimplemented or illegal memory, the command will generate an error code. The key memory can never be read under any circumstances – any attempt to read the key memory will generate an error code.

The user zone access is dependent upon the value of the EncRead and AuthRead bits of the ZoneConfig[UZ] register. If ZoneConfig[UZ].AuthRead is 0b, then BlockRead can read the user zone. If ZoneConfig[UZ].AuthRead is 1b, then BlockRead can only be used to access the user zone if the authentication requirement has been satisfied. If ZoneConfig[UZ].EncRead is 1b, then BlockRead can never be used to access the user zone. A single BlockRead command can only read data from a single user zone – the requested data can not span multiple user zones, or multiple EEPROM pages.

Table 7-25. Input parameters

	Name	Size (Bytes)	Notes
Opcode	BlockRead	1	0x10
Mode	Mode	1	Must be 0x00
Param1	Address	2	The address of data to read
Param2	Count	2	Upper byte is always 0x00. Lower byte is the number of bytes to read.

Table 7-26. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned Any command execution or validation failure generates a non-zero error code, per Section 6.3
OutData	0 - 32	Output data (cleartext)

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.5. Counter Command

The counter command reads or increments the internal non-reversible monotonic counters. Each counter can increment up to a value of 2,097,134 using the count command, after which they can be no longer changed. See Appendix H for additional counter information.

Table 7-27. Counter command options

Mode bit 1	Mode bit 0	Description	InMAC	OutMAC
1b	1b	Read Counter with MAC	Prohibited	Generated
0b	1b	Read Counter, no MAC	Prohibited	No
1b	0b	Increment Counter with MAC	Required	No
0b	0b	Increment Counter, no MAC	Prohibited	No

The CounterConfig[CntID].RequireMAC register bit determines if InMAC is required when incrementing the counter (see Section 4.4). If CounterConfig[CntID].RequireMAC = 1b, then InMAC is required, so mode bit 1 must be set to 1b when incrementing the counter. If CounterConfig[CntID].RequireMAC = 0b, then InMAC is prohibited, so mode bit 1 must be set to 0b.

If a MAC is required or generated, then a valid nonce is required to run the counter command. If the KeyConfig[KeyID].RandomNonce bit is set for the authorizing key, then the nonce must be random.

The AuthCompute command can be used to generate the InMac (see Section 7.3). The AuthCheck command can be used to validate the OutMac (see Section 7.2).

Table 7-28. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Counter	1	0x0A
Mode	Mode	1	Bit 0: If 1b, then read the counter If 0b, then Increment the counter Bit 1: If 1b, then InMAC is included in the input packet if bit 0 is 0b, or OutMAC is generated if bit 0 is 1b If 0b, then neither the input nor output packets include a MAC. Bits 2 to 4: Reserved. Must be 0b. Bit 5: If 1b, include the usage counter associated with the key(1) used to generate the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first four bytes of the SmallZone in the MAC
Param1	CountID	2	Upper byte is always 0x00. Upper nibble of lower byte is always 0x0. Lower nibble of lower byte is the counter to be queried.
Param2	Zero	2	Always 0x0000
Data	InMac	0 or 16	Integrity MAC for the counter increment operation

Notes: 1. The MAC is generated using the key identified by the KeyID in CounterConfig[CountID].IncrID for increment operations, or the KeyID in CounterConfig[CountID].MacID for counter read operations. The usage counter included in the MAC when mode bit five is 1b is identified by the CntID stored in KeyConfig[KeyID].CounterNum for the key used to generate the MAC.

Table 7-29. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned Any command execution or validation failure generates a non-zero error code, per Section 6.3
CountValue	4	The current value of the counter
OutMac	0 or 16	Integrity MAC for the counter read operation

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.6. Crunch Command

The crunch command processes a seed value and returns the result within a specified time. The command provides a 16 byte input seed which is combined with the ManufacturingID register and processed with the internal hardware crunch calculator. The calculation is performed within a specified time period.

The host system should read the response within a few milliseconds after the response is specified to be available and compare the returned value to the expected result to determine if authentic Atmel hardware is present. The crunch algorithm is proprietary and is available only in authentic Atmel hardware.

The crunch command does not use the AES engine or the nonce. Executing the crunch command does not change the authentication status or cryptographic state of the device.

7.6.1. Crunch Response Time

The response to the crunch command is available after a period a time that is dependent on the count field value. A large count value requires more time to process than a small count value. The expected response time for the crunch command is computed using the following equation:

$$((Count \times 256) + 600)) \times 1.25\ microseconds$$

Table 7-30. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Crunch	1	0x0B
Mode	Mode	1	Must be 0x00
Param1	Count	2	Upper byte is always 0x00 Lower byte is the iteration count for the crunch engine
Param2	Zero	2	Always 0x0000
Data	Seed	16	Input seed

Table 7-31. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned Any command execution or validation failure generates a non-zero error code, per Section 6.3
Result	16	Result out

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.7. DecRead Command

The DecRead command is used to check the OutMAC generated by an EncRead command on a second ATAES132 device. If the MAC matches, then the 1 to 16 bytes of data is returned to the host in the DecRead response.

To use this command, the nonce must be identical on both devices (see Section 7.21.1) and the MacCount must have the same value. Both devices must also contain identical key values, but it is not necessary for the KeyID on the origin device to match the KeyID on the destination device. In this section, the device which encrypts the data and generates the MAC is referred to as the origin device – the device checking the MAC is referred to as the destination device.

If mode bit 5, 6, or 7 is 1b, then the associated usage counter, SerialNum register value, or the first four bytes of the SmallZone register in the SecondBlock field, must match the values on the origin device. The ManufacturingID register must be identical on both devices, since it is always included in the MAC calculation.

A valid nonce is required to run the DecRead command. If the KeyConfig[DKeyID].RandomNonce bit is 1b, then the nonce must be random.

Table 7-32. Input parameters

	Name	Size (Bytes)	Notes
Opcode	DecRead	1	0x17
Mode	Mode	1	The value of this field must match the mode field value used when executing the EncRead command on the origin device.
Param1	DKeyID	2	Upper byte is always 0x00. Lower byte is the pointer to the decrypt key. Legal values: 0x00 to 0x0F, 0xFF.
Param2	Count	2	Upper byte is always 0x00. Lower byte is the number of data bytes to be decrypted.
Data1	FirstBlock	6	The value of this field must match the first authenticate-only block used when executing the EncRead command on the origin device.
Data2	SecondBlock	16	The value of this field must match the second authenticate-only block used when executing the EncRead command on the origin device. If mode bits 5, 6, and 7 are 0b, then this field must be present, but is ignored
Data3	InMac	16	Integrity MAC for the input data
Data4	InData	16	Input data (ciphertext) to be decrypted

Table 7-33. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutData	1 to 16	Decrypted (plaintext) output data

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.8. Decrypt Command

The decrypt command accepts 16 or 32 bytes of ciphertext, decrypts the data, verifies the MAC, and returns the decrypted data if the MAC matches. If the MAC does not match, then an error code is returned.

The decrypt command has two operating modes, normal decryption mode and the client decryption mode. The client decryption mode can decrypt packets which were encrypted by an ATAES132 device. The normal decryption mode decrypts packets generated by a cryptographic host. It cannot decrypt packets encrypted by the ATAES132.

If the DKeyID is VolatileKey: (See Section 4.3)
    The VolUsage.DecryptOK must be 1b when the VolatileKey was loaded

If the DKeyID is not the VolatileKey, then:
    The KeyConfig[DKeyID].ExternalCrypto bit must be 1b

If the KeyConfig[DKeyID].AuthKey bit is 1b, then:
    Prior authentication must be performed using the KeyID stored in KeyConfig[DKeyID].LinkPointer

A valid nonce is required to run the decrypt command. If KeyConfig[DKeyID].RandomNonce bit is 1b, then the nonce must be random.

7.8.1. Client Decryption Mode

In the client decryption mode, the decrypt command can be used to decrypt packets encrypted by the ATAES132 (either another device, or by the same device at a later time), using the encrypt command (see Section 7.10). All of the following requirements must be satisfied:

4. The device performing the encrypt operation (the encrypt device) and the device performing the decrypt operation (the decrypt device) must contain identical keys
5. The KeyID of the key used by the encrypt device (called EKeyID) must be known. EKeyID is passed to the decrypt device in the upper byte of decrypt Param1 for use in the MAC calculation.
6. The nonce used by the encrypt device must be known. The nonce is passed to the decrypt device using the nonce command with mode bit 0 = 0b (See Section 7.20), or is synchronized with the encrypt device using the procedure in Section 7.21.1.
7. The lower byte of the count (Encrypt Param2) used by the encrypt device must identical to the value used in the lower byte of decrypt Param2 by the decrypt device. [This is used in the MAC calculation].
8. The MacCount of the encrypt device (called EMacCount) must be known. EMacCount is passed to the decrypt device in the upper byte of decrypt Param2 for use in the data decryption operation.
9. The encrypt/decrypt command mode bits on both devices must be identical. Mode bit 5 must be 0b. Mode bit 6 must be 0b unless a single device is performing both the encrypt and the decrypt operations. Mode bit 7 can be 1b if the first four bytes of SmallZone are identical on both the encrypt and the encrypt devices.
10. The decrypt device KeyConfig[DKeyID] must have ExternalCrypto = 1b, and RandomNonce = 0b for the KeyID used for decryption if the nonce is passed using the nonce command with mode bit 0 = 0b.
11. The encrypt device KeyConfig[EKeyID] must have ExternalCrypto = 1b, and RandomNonce = 1b for the KeyID used for encryption (the EKeyID).

If these conditions are satisfied, then packets encrypted on the encrypt device can be decrypted on the decrypt device. If a single ATAES132 will be used to encrypt packets for later decryption, then the same key value must be stored in two appropriately configured key registers to allow all of the requirements above to be satisfied.

Table 7-34. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Decrypt	1	0x07
Mode	Mode	1	Bits 0 to 4: Reserved. Must be 0b. Bit 5: If 1b, include the usage counter associated with the encryption key in the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first 4 bytes of the SmallZone in the MAC
Param1	DKeyID	2	Normal decryption mode: The upper byte is always 0x00, and the lower byte is the KeyID of the decrypt key Client decryption mode: The upper byte is the EKeyID, and the lower byte is the KeyID of the decrypt key
Param2	Count	2	Normal decryption mode: The Upper byte is always 0x00. Lower byte is the number of bytes to be returned after decryption. Client decryption mode: The upper byte is the EMacCount. The lower byte is the number of bytes to be returned after decryption. (see Section 7.8.1)
Data1	InMac	16	Integrity MAC for the input data
Data2	InData	16 or 32	Input data (ciphertext) to be decrypted

Table 7-35. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutData	1 - 32	Decrypted (plaintext) output data

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.9. EncRead Command

EncRead reads 1 to 32 bytes of encrypted data from user memory, along with an integrity MAC. The EncRead command only performs encrypted reads; the BlockRead command is used for unencrypted reads (see Section 7.4).

The ZoneConfig[UZ].EncRead bit determines if a user zone can be accessed with the EncRead command. If the ZoneConfig[UZ].EncRead bit is 1b, then the EncRead command can read the user zone if the access requirements have been satisfied. A single EncRead command reads data from a single user zone – the requested data can not span multiple user zones. A single EncRead command reads data from a single EEPROM page – the requested data can not cross page boundaries (see Section B.2).

If ZoneConfig[UZ].Auth is 1b, then prior authentication is required with the following restrictions:

- The Auth command Usage.ReadOK bit must be 1b
- The authentication key AKeyID must match ZoneConfig[UZ].AuthID
- The Auth command must be run in Inbound Only Authentication or Mutual Authentication mode
- A valid Nonce is required to run the EncRead command. If KeyConfig[KeyID].RandomNonce for the Read key is 1b, then the Nonce must be random.

The DecRead command can be used to validate the OutMac and decrypt up to 16 bytes of data (see Section 7.7).

7.9.1. Configuration Memory Signature

The EncRead command cannot be used to read the configuration memory – the BlockRead command can be used to read the configuration memory. Any attempt to read any address in the configuration memory with the EncRead command will activate the configuration memory signature generation mode.

The configuration memory signature is an AES-CCM MAC generated over the entire configuration memory as described in Section I.17. A valid nonce is required to run the EncRead command in configuration memory signature generation mode. If KeyConfig[00].RandomNonce is 1b, then the nonce must be random. KeyID 00 is always used to generate the configuration memory signature.

The configuration memory signature generation mode is intended to be used during secure personalization of the ATAES132. The signature can be used to validate the contents of the configuration memory prior to programming secret data into other portions of the EEPROM.

7.9.2. Key Memory Signature

The EncRead command cannot be used to read the key memory. The key memory can never be read. Any attempt to read any address in the key memory with the EncRead command will activate the key memory signature generation mode; however, this signature can only be generated once per unit.

The key memory signature is an AES-CCM MAC generated over all 16 key registers as described in Section I.18. A valid nonce is required to run the EncRead command in key memory signature Generation mode. If KeyConfig[00].RandomNonce is 1b, then the nonce must be random. KeyID 00 is always used to generate the key memory signature.

The key memory signature generation mode is intended to be used during secure personalization of the ATAES132. The signature can be used to validate the contents of the key memory before locking the key memory.

Table 7-36. Input parameters

	Name	Size (Bytes)	Notes
Opcode	EncRead	1	0x04
Mode	Mode	1	Bits 0 to 4: Reserved. Must be 0b. Bit 5: If 1b, include the usage counter associated with the ZoneConfig[UZ].ReadID key in the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first 4 bytes of the SmallZone in the MAC
Param1	Address	2	The address of data to be read
Param2	Count	2	Upper byte is always 0x00. Lower byte is the number of bytes to read.
Data	-	0	

Table 7-37. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3
OutMac	16	Integrity MAC for the output data
OutData	16 or 32	Encrypted output data (ciphertext)

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.10. Encrypt Command

The encrypt command accepts 1 to 32 bytes of plaintext, encrypts the data and generates an integrity MAC. The encrypted data and OutMAC are returned to the system.

The encrypt command can be used to encrypt packets for decryption by the same or by another ATAES132 if the requirements described in Section 7.8.1 are satisfied.

If the EKeyID specifies a key in the key memory: the KeyConfig[EKeyID].ExternalCrypto bit must be 1b.

If KeyConfig[EKeyID].AuthKey bit is 1b, then prior authentication is required using the KeyID stored in KeyConfig[EKeyID].LinkPointer.

If the EKeyID specifies the VolatileKey: (See Section 4.3) The VolUsage.EncryptOK must be set to 01b, 10b, or 11b.

If the VolUsage.EncryptOK bits are set to 10b or 11b, then prior authentication is required using VolatileKey prior to execution of the encrypt command.

A valid Nonce command is required to run the Encrypt command. If the KeyConfig[EKeyID].RandomNonce bit is set for the encryption key, then the Nonce must be random.

Table 7-38. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Encrypt	1	0x06
Mode	Mode	1	Bits 0 to 4: Reserved. Must be 0b. Bit 5: If 1b, include the usage counter associated with the encryption key in the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first four bytes of the SmallZone in the MAC
Param1	EKeyID	2	Upper byte is always 0x00. Lower byte is the KeyID of the encrypt key
Param2	Count	2	Upper byte is always 0x00. Lower byte is the number of bytes to be encrypted
Data	InData	1 – 32	Input data to be encrypted (plaintext)

Table 7-39. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutMac	16	Integrity MAC for the output data
OutData	16 or 32	Encrypted data (ciphertext)

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.11. EncWrite Command

The EncWrite command decrypts the ciphertext input data, verifies the input MAC, and then writes 1 to 32 bytes to a user zone or 16 bytes to key memory.

The ZoneConfig[UZ].EncWrite bit determines if a user zone can be accessed with the EncWrite command. If the ZoneConfig[UZ].EncWrite bit is 1b, then the EncWrite command can write the user zone if the access requirements have been satisfied. A single EncWrite command writes data to a single user zone – the data can not span multiple user zones. A single EncWrite command writes data to a single EEPROM page – the data can not cross page boundaries (see Section B.2).

If ZoneConfig[UZ].Auth is 1b, then prior authentication is required with the following restrictions:

- The Auth command Usage.WriteOK bit must be 1b
- The authentication key AKeyID must match ZoneConfig[UZ].AuthID
- The Auth command must be run in Inbound Only Authentication or Mutual Authentication mode
- A valid Nonce is required to run the EncWrite command. If KeyConfig[KeyID].RandomNonce for the Write key is 1b, then the Nonce must be random.

7.11.1. Encrypted Key Writes

When EncWrite is used to write the key memory prior to locking, then the key data must be encrypted using KeyID 00. The input MAC is also calculated using KeyID 00. Writes to key memory must be 16 bytes in length, and begin at the starting address of the key.

If LockKeys has a value of 0x55 and the EncWrite address points to key memory, then key personalization mode is selected. In the key personalization mode, the following requirements are in effect:

- The Count field value must be 16
- The Address must match the starting address of the key register
- The input data must be encrypted with the current value in KeyID 00. If KeyConfig[WriteID].RandomNonce is 1b then the nonce must be random (See Section 7.20).
- The input MAC must be generated with the current value in KeyID 00. The input MAC will be verified.

If the key memory is locked, then the new key data is encrypted with the current value of the key being written. The key can only be updated if all of the following requirements are satisfied:

- The corresponding KeyConfig[KeyID].ChangeKeys bit is set to 1b (see Section 4.2)
- The count field value must be 16
- The address must match the starting address of the key register.
- The input data must be encrypted with the current value of the key. If KeyConfig[WriteID].RandomNonce is 1b then nonce be random (See Section 7.20).
- The input MAC must be generated with the current value of the key. The input MAC will be verified.

See section 7.19 for the lock command.

Table 7-40. Input parameters

	Name	Size (Bytes)	Notes
Opcode	EncWrite	1	0x05
Mode	Mode	1	Bits 0 to 4: Reserved. Must be 0b. Bit 5: If 1b, include the usage counter associated with the encryption key in the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first four bytes of the SmallZone in the MAC
Param1	Address	2	The starting address of memory to be written
Param2	Count	2	Upper byte is always 0x00. Lower byte is the number of bytes to be written.
Data1	InMac	16	Input MAC to be verified
Data2	InData	16 or 32	Encrypted Data (ciphertext)

Table 7-41. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.12. Info Command

The Info command reads various information about the device from internal registers. Param1 selects the information to read. Operation of this command does not require knowledge of any secrets.

Table 7-42. Coding of the selector field (Param1)

Selector	Name	Description
0x0000	MacCount	Read the MacCount register. The first byte is always 0x00; the second byte is the MacCount value.
0x0005	AuthStatus	Read the authentication status register. Returns 0xFFFF to indicate that the AuthComplete status flag = NoAuth. If the AuthComplete status flag = YesAuth, then the info returns the AKeyID as 0x00KK, where KK is the authentication KeyID.
0x0006	DeviceNum	Read the DeviceNum register. The first byte is the Atmel device code which is unique to this Atmel catalog number. The second byte provides the device revision number. See Table 7-29 for DeviceNum codes
0x000C	ChipState	Read the ChipState device state register: 0x0000 indicates the ChipState = Active 0xFFFF indicates the ChipState = Power Up 0x5555 indicates the ChipState = "Wakeup from Sleep" See Section L.3 for a detailed description of ChipState
All Other	Reserved	Reserved for future use

Table 7-43. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Info	1	0x0C
Mode	Mode	1	Must be 0x00
Param1	Selector	2	Selects the register to read
Param2	Zero	2	Always 0x0000
Data	-	0	

Table 7-44. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
Result	2	Current value of the register

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

Table 7-45. DeviceNum coding for INFO response and DeviceNum in the configuration memory register

Description	INFO DeviceNum	DeviceNum Register
Early pre-production samples	0x0A02	0x0A
Pre-production samples	0x0A04	0x0A

7.13. KeyCompute Command

The KeyCompute command generates a 16 byte random number and stores it in either the key memory or in the VolatileKey register. The newly generated key is then encrypted with the parent key and returned to the host along with a MAC.

If Mode bit 0 is 1b, then the target key is in the key memory:

- KeyConfig[ChildKeyID].Child must be 1b
- The KeyCompute command KeyID field contains the ChildKeyID
- KeyConfig[ChildKeyID].LinkPointer contains the ParentKeyID

If Mode bit 0 is 0b, then the target key is VolatileKey:

- KeyConfig[ParentKeyID].Parent must be 1b
- The KeyCompute command KeyID field contains the ParentKeyID
- The VolUsage field specifies VolatileKey usage restrictions as defined in Section 4.3

If KeyConfig[ParentKeyID].AuthKey bit is 1b or KeyConfig[EKeyID].ExportAuth bit is 1b, then prior authentication is required using the KeyID stored in KeyConfig[ParentKeyID].LinkPointer.

The InMAC and OutMAC are both calculated using the parent key (ParentKeyID). If KeyConfig[ChildKeyID].ChildMac is 1b, then an InMAC must be provided, otherwise the InMAC will be ignored.

A valid nonce is required to run the KeyCompute command. If the KeyConfig[ParentKeyID].RandomNonce bit is 1b, then the nonce must be random.

If the LockConfig register is unlocked (0x55), then the random number generator is latched in test mode and the KeyCompute command will generate non-random key values. If the LockConfig register is locked (0x00), then the RNG generates random numbers and the KeyCompute command functions normally.

The KeyImport command can be used to load a key generated by the KeyCompute command (see Section 7.15).

Warning: There is one random number generator (RNG) seed register in the EEPROM memory which is used by the KeyCompute, KeyExport, nonce, and random commands. The RNG seed register is subject to the same write endurance limitations as the other bytes in the EEPROM (see Section 9.2 for the EEPROM specifications) – the application developer must not exceed the write endurance limit.

Table 7-46. Input parameters

	Name	Size (Bytes)	Notes
Opcode	KeyCompute	1	0x08
Mode	Mode	1	Bit 0: If 1b, the key load target is key memory If 0b, then the target is VolatileKey (see Section 4.3) Bit 1: If 0b, update the EEPROM RNG seed register prior to key generation(1) If 1b, then generate Key using existing RNG seed Bits 2: If 1b, A key equivalent to what the KeyCompute InMac would be is generated. Including an InMac with the KeyCompute command is not required. Bits 3-4: Reserved. Must be 0. Bit 5: If 1b, include the usage counter associated with the ParentKeyID in the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first four bytes of the SmallZone in the MAC
Param1	KeyID	2	Upper byte is always 0x00. Lower byte is the ChildKeyID for key memory loads or the ParentKeyID for VolatileKey loads
Param2	VolUsage	2	Usage restrictions for VolatileKey if mode bit 0 is 0b (see Section 4.3)
Data	InMac	0 or 16	Optional input MAC (see above)

Notes: 1. The RNG seed register in the EEPROM will be updated automatically if mode bit 1 = 0b unless the seed register was previously updated after the most recent power on reset, wake from the sleep state, reset command, or tamper event. Updating the RNG seed register increases the randomness of the keys generated by the KeyCompute command, however, the EEPROM write endurance specification must be respected.

Table 7-47. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutMac	16	Output MAC for the encrypted key
OutData	16	Encrypted key value (ciphertext)

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.14. KeyExport Command

The KeyExport command is used to encrypt a key for export to a second ATAES132 device. The source of the key can be the internal random number generator, the VolatileKey register, or external data. The resulting encrypted key is used as the input to the KeyImport command or KeyLoad command. This command does not modify the stored keys.

If mode bits 2 and 3 are 00b, then a new key is generated for export:

- The internal random number generator is used to generate the key

If Mode bit 1 is 0b, then the EEPROM seed register will be updated prior to key generation.

If Mode bits 2 and 3 are 01b, then the key in the input packet will be exported:

- The KeyExport command InData field contains the key value
- Mode bit 1 is ignored

If Mode bits 2 and 3 are 10b or 11b, then the VolatileKey will be exported:

- Mode bit 1 is ignored

If KeyConfig[EKeyID].AuthKey bit is 1b or KeyConfig[EKeyID].ExportAuth bit is 1b, then prior authentication is required using the KeyID stored in KeyConfig[EKeyID].LinkPointer.

To use this command the nonce must be identical on both devices (see Section 7.21.1) and the MacCount must have the same value. Both devices must also contain identical key values, but it is not necessary for the encrypt KeyID on the origin device to match the decrypt KeyID on the destination device. In this section, the device which encrypts the key is referred to as the origin device – the device receiving the key is referred to as the destination device.

If mode bit 0 is 1b and mode bit 5, 6, or 7 is 1b, then the associated usage counter, SerialNum register value, or the first four bytes of the SmallZone register must be identical on both devices. If mode bit 0 is 0b and mode bit 5, 6, or 7 is 1b, then the value of SecondBlock must match the associated values on the destination device – the value of mode bits 5, 6, and 7 of the KeyExport command must also match the value in the FirstBlock field. The ManufacturingID register must be identical on both devices, since it is always included in the MAC calculation.

A valid nonce is required to run the KeyExport command. If the KeyConfig[EKeyID].RandomNonce bit is 1b, then the nonce must be random.

Warning: There is one random number generator (RNG) seed register in the EEPROM memory which is used by the KeyCompute, KeyExport, nonce, and random commands. The RNG seed register is subject to the same write endurance limitations as the other bytes in the EEPROM (see Section 9.2 for the EEPROM specifications) – the application developer must not exceed the write endurance limit.

Table 7-48. Input parameters

	Name	Size (Bytes)	Notes
Opcode	KeyExport	1	0x18
Mode	Mode	1	Bit 0: If 0b, then the key will be encrypted for use by the KeyImport command. FirstBlock and SecondBlock field values are not used. If 1b, then the key will be encrypted for use by the KeyLoad command using the FirstBlock and SecondBlock field values. Mode bits 5, 6, and 7 must match the values in the FirstBlock. Bit 1: If 0b, update the EEPROM RNG seed register prior to key generation[1] If 1b, then generate key using existing RNG seed Bits 2 and 3: If 00b, then a new key will be generated for export using the internal random number generator If 01b, then the InData field value will be exported If 10b or 11b,the VolatileKey will be exported Bit 5: If 1b, include the usage counter associated with the encrypt key in the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first four bytes of the SmallZone in the MAC
Param1	EKeyID	2	Upper byte is always 0x00. Lower byte is the pointer to the encrypt key. Legal values: 0x00 to 0x0F.
Param2	VolUsage	2	Usage restrictions for VolatileKey if Mode bit 0 is 0b and the TargetKeyID is intended to be the VolatileKey register. (See Section 4.3) For all other cases this field must be 0x0000
Data1	FirstBlock	6	If mode bit 0 is 1b, then all bytes must be 0x00 If mode bit 0 is 0b, the value of this field must match the first authenticate-only block to be used when executing the KeyLoad command on the destination device
Data2	SecondBlock	16	If mode bit 0 is 1b, then this field must be present, but is ignored If mode bit 0 is 0b, the value of this field must match the second authenticate-only block to be used when executing the KeyLoad command on the destination device. If mode bits 5, 6, and 7 are 0b, then this field must be present, but is ignored.
Data3	InData	0 or 16	If mode bits 2 and 3 are 01b, then this field contains the key (plaintext) to be encrypted. For all other cases this field is ignored.

Note: 1. The RNG seed register in the EEPROM will be updated automatically if mode bit 1 = 0b unless the seed register was previously updated after the most recent power on reset, wake from the sleep state, reset command, or tamper event. Updating the RNG seed register increases the randomness of the keys generated by the KeyCompute command, however, the EEPROM write endurance specification must be respected.

Table 7-49. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3
OutMac	16	Integrity MAC for the encrypted key
OutData	16	Encrypted key (ciphertext)

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.15. KeyImport Command

The KeyImport command accepts 16 bytes of ciphertext, decrypts the key, verifies the MAC, and stores the key in the key memory or in the VolatileKey register. The source of the encrypted key can be the KeyExport command or the KeyCompute command.

If TargetKeyID specifies that the target key is stored in the Key Memory:

- KeyConfig[TargetKeyID].ImportOK bit must be 1b
- KeyConfig[TargetKeyID].LinkPointer contains the decrypt KeyID
- The KeyImport command DKeyID field value is ignored

If KeyConfig[decrypt KeyID].AuthKey is 1b, then prior authentication is required using the KeyID stored in KeyConfig[decrypt KeyID].LinkPointer.

If TargetKeyID specifies that the target key is VolatileKey: (See Section 4.3)

- KeyConfig[DKeyID].Parent must be 1b
- The KeyImport command DKeyID field contains the decrypt KeyID

If KeyConfig[DKeyID].AuthKey is 1b, then prior authentication is required using the KeyID stored in KeyConfig[DKeyID].LinkPointer.

To use this command, the nonce must be identical on both devices (see Section 7.21.1) and the MacCount must have the same value. Both devices must also contain identical key values, but it is not necessary for the encrypt KeyID on the origin device to match the decrypt KeyID on the destination device. In this section, the device which encrypts the key and generates the MAC is referred to as the origin device – the device checking the MAC is referred to as the destination device.

If Mode bit 5, 6, or 7 is 1b, then the associated usage counter, SerialNum register value, or the first four bytes of the SmallZone register in the SecondBlock field must match the values on the origin device. The ManufacturingID register must be identical on both devices, since it is always included in the MAC calculation.

A valid nonce is required to run the KeyImport command. If the KeyConfig[KeyID].RandomNonce bit is 1b for the decrypt key, then the nonce must be random.

Table 7-50. Input parameters

	Name	Size (Bytes)	Notes
Opcode	KeyImport	1	0x19
Mode	Mode	1	Bit 0: If 1b, the key was encrypted with the KeyCompute command If 0b, the key was encrypted with the KeyExport command Bits 1 to 4: Reserved. Must be 0b. Bits 5 to 7: This value must match the mode bits 5, 6, and 7 value used when executing the KeyCompute command or KeyExport command on the origin device
Param1	TargetKeyID	2	Upper byte is always 0x00. Lower byte is the location where the decrypted key will be stored. Legal values: 0x00 to 0x0F (standard keys), 0xFF (volatile key).
Param2	DKeyID	2	Upper byte is always 0x00. If TargetKeyID = 0xFF, then lower byte is the pointer to the decrypt key. Legal values: 0x00 to 0x0F. If TargetKeyID = 0x00 to 0x0F, then this field must be present, but is ignored (see above).
Data1	FirstBlock	6	The value of this field must match the first authenticate-only block used when executing the KeyCompute command or KeyExport command on the origin device
Data2	SecondBlock	16	The value of this field must match the second authenticate-only block used when executing the KeyCompute command or KeyExport command on the origin device. If Mode bits 5, 6, and 7 are 0b, then this field must be present, but is ignored
Data3	InMac	16	MAC for the encrypted key
Data4	InData	16	Input key (ciphertext) to be decrypted

Table 7-51. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.16. KeyLoad Command

The KeyLoad command decrypts 16 bytes of ciphertext data, verifies the MAC, and then writes the key memory or the VolatileKey register.

If mode bit 0 specifies that the target key is stored in the key memory:

- KeyConfig[ChildKeyID].Child bit must be 1b
- The KeyLoad command KeyID field contains the ChildKeyID
- KeyConfig[ChildKeyID].LinkPointer contains the ParentKeyID

If KeyConfig[ParentKeyID].AuthKey is 1b, then prior authentication is required using the KeyID stored in KeyConfig[ParentKeyID].LinkPointer.

If mode bit 0 specifies that the target key is VolatileKey: (See Section 4.3)

- KeyConfig[ParentKeyID].Parent must be 1b
- The KeyLoad command KeyID field contains the ParentKeyID
- The VolUsage field specifies VolatileKey usage restrictions as defined in Section 4.3

If KeyConfig[ParentKeyID].AuthKey bit is 1b, then prior authentication is required using the KeyID stored in KeyConfig[ParentKeyID].LinkPointer.

A valid Nonce is required to run the KeyLoad command. If the appropriate KeyConfig[KeyID].RandomNonce bit is 1b, then the nonce must be random.

Table 7-52. Input parameters

	Name	Size (Bytes)	Notes
Opcode	KeyLoad	1	0x09
Mode	Mode	1	Bit 0: If 1b, the key load target is Key Memory. If 0b, target is VolatileKey. (See Section 4.3) Bits 1 to 4: Reserved. Must be 0b. Bit 5: If 1b, include the usage counter associated with ParentKeyID in the MAC Bit 6: If 1b, include the SerialNum in the MAC Bit 7: If 1b, include the first four bytes of the SmallZone in the MAC
Param1	KeyID	2	Upper byte is always 0x00. Lower byte is the ChildKeyID for the key memory loads or the ParentKeyID for VolatileKey loads.
Param2	VolUsage	2	Usage restrictions for VolatileKey if mode bit 0 is 0b (See Section 4.3)
Data1	InMac	16	Integrity MAC for the input data
Data2	InData	16	Encrypted key value (ciphertext)

Table 7-53. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.

The command and response packet is transmitted as a block beginning with Count and ending with a packet Checksum. This block format is described in section 6.1.

7.17. KeyTransfer Command

The KeyTransfer command copies key data from the user memory into the VolatileKey register or into a key register in the key memory. The KeyTransfer command allows a user zone to be utilized as an extended key memory.

Keys stored in the user memory cannot be utilized directly by the cryptographic commands – the keys must be transferred into either the VolatileKey register or into a key register in the key memory EEPROM prior to use. The usage restrictions for keys transferred into the VolatileKey register are transferred from the key data structure when the KeyTransfer command is executed. Usage restrictions for keys transferred into the key memory are stored in the KeyConfig[TargetKeyID] register – the KeyTransfer command does not alter the KeyConfig[TargetKeyID] register.

If KeyConfig[TargetKeyID].TransferOK is 0b then the key register cannot be updated with the KeyTransfer command.

If KeyConfig[TargetKeyID].TransferOK is 1b, then the KeyTransfer command can be used to update the Key register – the KeyConfig[TargetKeyID].LinkPointer contains the user zone number of the extended key memory.

If ZoneConfig[UZ].AuthRead is 1b for the user zone number containing the key data structure, then prior authentication is required using the KeyID stored in ZoneConfig[UZ].AuthID before a key can be transferred to either the VolatileKey register or into a key register in the key memory EEPROM.

7.17.1. Extended Key Memory Data Structure

When a user zone is utilized as the extended key memory, the keys are stored in the 32 byte key data structure as shown in Table 7-38. The first 16 bytes contain the key value, two bytes store the VolUsage restrictions, and the remaining bytes should contain all zeros. The starting address of each key data structure is required to be the first byte of a 32 byte physical page (see Section B.2).

Table 7-54. Key data structure in user memory

Address	0_h	1_h	2_h	3_h	4_h	5_h	6_h	7_h	8_h	9_h	A_h	B_h	C_h	D_h	E_h	F_h
$XX00_h$-$XX0F_h$	Key															
$XX10_h$-$XX1F_h$	VolUsage		Reserved (All bytes 0x00)													

Table 7-55. Input parameters

	Name	Size (Bytes)	Notes
Opcode	KeyTransfer	1	0x1A
Mode	Mode	1	Must be 0x00
Param1	TargetKeyID	2	Upper byte is always 0x00. Lower byte is the location where the key will be stored. Legal values: 0x00 to 0x0F (standard keys), 0xFF (volatile key).
Param2	Address	2	Starting address of the key data structure in user memory

Table 7-56. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.18. Legacy Command

The legacy command executes a single block of the AES engine in the “Electronic Code Book” mode, with no input or output formatting. This is known as AES-ECB mode and can be used to perform primitive AES encryption or decryption operations. This command does not use the nonce register value in the computation since the entire 16 byte AES input value comes from the input packet.

This command can only be executed if it is enabled for the device by setting ChipConfig.LegacyE to 1b and for the key by setting KeyConfig[LKeyID].LegacyOK is 1b.

Atmel recommends that any key with KeyConfig[LKeyID].LegacyOK = 1b should never be used with any other command – the legacy command can be used to exhaustively attack the key. If KeyConfig[LKeyID].AuthKey bit is 1b, then prior authentication is required using the KeyID stored in KeyConfig[LKeyID].LinkPointer.

Key usage limits are enforced if KeyConfig[LKeyID].CounterLimit is 1b, see Section 4.2. See Section E.2.16 for the ChipConfig register definition.

Table 7-57. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Legacy	1	0x0F
Mode	Mode	1	Must be 0x00
Param1	LKeyID	2	Upper byte is always 0x00 Lower byte is the KeyID for the AES key
Param2	Zero	2	Always 0x0000
Data	InData	16	Input to the AES block (plaintext)

Table 7-58. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutData	16	The output of the AES block (ciphertext)

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.19. Lock Command

The lock command permanently locks various segments of the EEPROM memory, including the configuration memory, the key memory, and the SmallZone register. Key, Counter, and User Memory access restrictions are locked when the configuration memory is locked. The SmallZone is locked independently of the other configuration memory registers.

Three registers in the configuration memory control the lock/unlock status of the memory segments:

1. The configuration memory is controlled by the LockConfig register (see Section E.2.11)
2. The key memory is controlled by the LockKeys register (see Section E.2.9)
3. The SmallZone register is controlled by the LockSmall register (see Section E.2.10)

If the lock control register contains 0x55, then the memory segment is unlocked. The lock command writes the specified lock register to 0x00 to lock the segment. The lock control registers can only be written with the lock command, but they can always be read with the BlockRead command. (See Section 7.4)

The lock command Param2 is an optional checksum (CRC-16) generated over the memory segment being locked. The value in the checksum field must match the CRC-16 calculated within the device for the lock operation to succeed. If the lock command returns a LockError ReturnCode, then the host system should re-write the memory segment and try the lock operation again.

The Atmel recommendation is the key memory be locked immediately after loading the keys. See Appendix P for personalization examples.

7.19.2. User Zone ReadOnly Activation

After the configuration memory is locked, the lock command can be used to activate the ReadOnly sser zone feature on appropriately configured user zones. The lock command changes the user zone from read/write to read-only if the following requirements are satisfied:

- ZoneConfig[Zone].WriteMode must be 10b or 11b
- Lock command mode bits 0 and 1 must be set to 11b
- The Lock command zone field contains the target user zone number (Zone)

If lock command mode bits 2 is 1b, then the checksum field contains the CRC-16 of the user zone contents.

If ZoneConfig[Zone].WriteMode is 11b, then the command must include an InMAC generated using the KeyID stored in ZoneConfig[Zone].WriteID, otherwise, the MAC is ignored.

The lock command changes the ZoneConfig[Zone].ReadOnly byte from 0x55 (read/write) to 0x00 when the ReadOnly feature is activated. It is not possible to change a read-only User Zone to read/write after Configuration Memory is locked.

Table 7-59. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Lock	1	0x0D
Mode	Mode	1	Bit 0-1: If 00b, lock the SmallZone register If 01b, lock the key memory If 10b, lock the configuration memory, excluding the SmallZone If 11b, then set the ZoneConfig[Zone].ReadOnly byte to ReadOnly Bit 2: If 1b, validate the memory checksum in Param2 If 0b, suppress the checksum validation (not recommended by Atmel) Bits 3-4: Reserved. Must be 0x00 Bit 5: If 1b, include the usage counter associated with the ZoneConfig[Zone].WriteID key in the MAC (Ignored unless Mode[0:1] is 11b) Bit 6: If 1b, include the SerialNum in the MAC (Ignored unless Mode[0:1] is 11b) Bit 7: If 1b, include the first four bytes of the SmallZone in the MAC (Ignored unless Mode[0:1] is 11b)
Param1	Zone	2	Upper byte is always 0x00. If Mode[0:1] is 11b the lower byte is the user zone to be locked. (see Section 7.19.2) For any other values of Mode[0:1], this field must be 0x0000.
Param2	Checksum	2	If mode bit 2 is 1b, contains the CRC-16 checksum generated of the memory segment being locked. If mode bit 2 is 0b, this parameter must be 0x0000.
Data	InMAC	0 or 16	If Mode[0:1] is 11b, contains the MAC authorizing update of ZoneConfig[Zone].ReadOnly as described in Section 7.19.2. For all other modes this field is ignored.

Table 7-60. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.20. Nonce Command

The nonce command generates and/or stores a 96 bit nonce in the SRAM nonce register for use by subsequent cryptographic commands. It is not necessary to generate a new nonce before each cryptographic operation because the ATAES132 includes the MacCount in the MAC calculations (see Section I.1) to guarantee uniqueness.

There are two nonce command options:

1. **Inbound nonce**

   The InSeed value is written directly to the nonce register. No random number generation or cryptographic nonce calculation is performed.

   Note: This option provides no defense against replay attacks or known plaintext attacks.

2. **Random nonce**

   The InSeed value is cryptographically combined with the new output of the random number generator and stored in the nonce register. The random number used for the nonce calculation is returned to the host in the response. See Section I.31 for the nonce algorithm.

If the LockConfig register is unlocked (0x55), then the random number generator is latched in the test mode and executing the nonce command with mode bit 0 = 1b will generate non-random values. If the LockConfig register is locked (0x00) then the RNG generates random numbers and the nonce command functions normally.

The nonce remains valid until one of the following events occurs:

- A MAC compare operation fails
- The MacCount reaches the maximum count (See Section I.1)
- The cryptographic state machine is reset due to: receipt of a reset command, power cycling (POR), or activation of the initialization sequence due to WakeUp from the sleep power state (see Section G.2.2)
- The execution of the nonce command resets the MacCount to zero (see Section I.1)

If a cryptographic operation involves two ATAES132 devices and a synchronized nonce is required, then the nonce synchronization procedure in Section 7.21.1 must be used. The nonce command cannot be used to generate a synchronized random nonce.

Warning: There is one random number generator (RNG) seed register in the EEPROM memory which is used by the KeyCompute, KeyExport, nonce, and random commands. The RNG seed register is subject to the same write endurance limitations as the other bytes in the EEPROM (see Section 9.2 for the EEPROM specifications) – the application developer must not exceed the write endurance limit.

Table 7-61. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Nonce	1	0x01
Mode	Mode	1	Bit 0: If 1b, generate a random nonce using the RNG If 0b, use the InSeed as the nonce (Inbound nonce mode), mode bit 1 is ignored Bit 1: If 0b, update the EEPROM RNG seed prior to nonce generation(1) If 1b, generate a random nonce using the existing RNG seed Bits 2-7: Reserved. Must be 0b
Param1	Zero	2	Always 0x0000
Param2	Zero	2	Always 0x0000
Data	InSeed	12	Input seed (required)

Note: 1. The RNG seed register in the EEPROM will be updated automatically if mode bit 1 = 0b unless the seed register was previously updated after the most recent power on reset, wake from the sleep state, reset command, or tamper event. Updating the RNG seed register increases the randomness of the nonce, however, the EEPROM write endurance specification must be respected.

Table 7-62. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution failure or validation failure generates a non-zero error code, per Section 6.3.
Random	0 or 16	In random nonce mode, the random number used to generate the nonce is returned. In inbound nonce mode, no data is returned.

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.21. NonceCompute Command

The NonceCompute command generates the nonce in a manner which allows two ATAES132 devices to have identical random nonces based on random numbers generated by both devices. The identical nonce values and identical MacCount values are required to encrypt data on one device for decryption by the other device.

The random command must be executed with mode bit 2 = 1b prior to execution of the NonceCompute command. The random command generates a random number which the NonceCompute command combines with the RandomSeed provided by the second ATAES132 to generate the random nonce.

The nonce remains valid until one of the following events occurs:

- A MAC compare operation fails
- The MacCount reaches the maximum count (see Section I.1)
- Due to the WakeUp from the sleep power state, the cryptographic state machine is reset due to either (see Section G.2.2):
  - Receipt of a reset command
  - Power cycling (POR), or
  - Activation of the initialization sequence

    This command resets the MacCount to zero only if the operation succeeds (see Section I.1). If an error occurs, the contents of the nonce register and the MacCount register remained unchanged – the NonceValid flag also remains unchanged.

7.21.1. Nonce Synchronization

The following procedure synchronizes the nonce and the MacCount on two ATAES132 devices. In this procedure, the device where the procedure begins is referred to as “A” and the device it is synchronized with is referred to as “B”.

1. The random command is executed on Device A with mode bit 2 set to 1b. The first 12 bytes of the random field value in the response are stored for use in step 2.
2. The nonce command is executed on Device B with mode bit 1 set to 1b. The 12 byte random number generated in step 1 is used as the nonce command InSeed field value. The 12 byte random field value in the response is stored for use in step 3.
3. The NonceCompute command is executed on Device A, using the 12 byte random number generated in step 2 as the RandomSeed field value.
4. Successful execution of this procedure sets the nonce status flags on both devices to NonceValid = YesNonce, NonceRandom = Random, and NonceCompute = No. The MacCount is zero on both devices.

Table 7-63. Input parameters

	Name	Size (Bytes)	Notes
Opcode	NonceCompute	1	0x13
Mode	Mode	1	The value of this field must match the mode field value used when executing the nonce command on the origin device
Param1	Zero	2	Always 0x0000
Param2	Zero	2	Always 0x0000
Data	RandomSeed	12	First 12 bytes output by the Nonce command on the origin device

Table 7-64. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution failure or validation failure generates a non-zero error code, per Section 6.3.

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.22. Random Command

The random command generates a random number using the internal high quality random number generator and the random number generation procedure recommended by NIST in SP800-90 (see Appendix A). The random command returns the generated random number to the host.

There are two random command options:

1. **Random number generation:**
   If mode bit 2 is 0b, the 16 byte random number is only returned to the host, it is not stored internally. This option does not affect the cryptographic state of the device.
2. **Nonce synchronization:**
   If mode bit 2 is 1b, then the first 12 bytes of the random number are stored in the nonce register for later use by the NonceCompute command. The 16 byte random number is returned to the host. The nonce status flags are changed to NonceValid = YesNonce, NonceRandom = Fixed, and NonceCompute = Yes. See Section 7.21 for the NonceCompute command and the nonce synchronization procedure.

If the LockConfig register is unlocked (0x55), then the random number generator is latched in the test mode, and the random command will always return 16 bytes of 0xA5. If the LockConfig register is locked (0x00), then the RNG generates random numbers.

Warning: There is one random number generator (RNG) seed register in the EEPROM memory, which is used by the KeyCompute, KeyExport, nonce, and random commands. The RNG seed register is subject to the same write endurance limitations as the other bytes in the EEPROM (see Section 9.2 for the EEPROM specifications) – the application developer must not exceed the write endurance limit.

Table 7-65. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Random	1	0x02
Mode	Mode	1	Bit 0: Reserved. Must be 0b Bit 1: If 0b, update the EEPROM RNG seed register prior to random number generation[1] If 1b, generate random number using the existing RNG seed Bit 2: If 0b, then return the random number. Do not change the nonce. If 1b, then store the first 12 bytes of the random number in the nonce register and return the 16 byte random number Bits 3 to 7: Reserved. Must be 0b
Param1	Zero	2	Always 0x0000
Param2	Zero	2	Always 0x0000
Data	-	0	

Note: 1. The RNG seed register in the EEPROM will be updated automatically if mode bit 1 = 0b unless the seed register was previously updated after the most recent power on reset, wake from the sleep state, reset command, or tamper event. Updating the RNG seed register increases the randomness of the random command output, however, the EEPROM write endurance specification must be respected.

Table 7-66. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution failure or validation failure generates a non-zero error code, per Section 6.3.
Random	16	The random number

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.23. Reset Command

The reset command forces the ATAES132 to reset the logic, including the AES engine, nonce, and authentication status flag. This command does not return a response.

When a Reset command is received, the ATAES132 performs the same power up reset sequence that occurs during wakeup from the sleep state. The reset is complete after the WakeUp Ready time $t_{WupSL.RDY.}$ (see Section 9.4.2)

7.23.1. SPI Reset

During the reset of an ATAES132 configured for SPI interface mode, the device will answer the SPI read status register command with 0xFF to indicate it is "busy". When reset is complete, the WIP status bit changes to 0b to indicate the device is in the active state. The ATAES132 will only accept the SPI read status register command while it is resetting – all other commands will be ignored. The SPI read status register command is described in Section K.3.6.

7.23.2. I^{2}C Reset

During the reset of an ATAES132 configured for I^{2}C interface mode, the host is required to perform ACK polling using the matching I^{2}C device address. The ATAES132 will answer the ACK poll with an I^{2}C NAK to indicate the device is "busy" during reset. The ACK poll reply will change to ACK when the device is in the active state. The ATAES132 will not accept any I^{2}C commands while it is "busy". ACK polling is described in Section J.3.8.

Table 7-67. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Reset	1	0x00
Mode	Mode	1	This byte can be any value
Param1	Zero	2	Always 0x0000
Param2	Zero	2	Always 0x0000
Data	-	0	

Table 7-68. Output parameters

Name	Size (Bytes)	Notes
		No response packet is returned by the reset command

The command packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.24. Sleep Command

The sleep command forces the ATAES132 into one of two low power states – sleep or standby. This command does not return a response.

The sleep state can be used to extend battery life in portable systems by powering down the ATAES132 internal circuitry when the device is sleeping. The standby state puts the internal circuitry in a low power state to reduce power consumption while preserving the volatile memory contents and the security state.

A device in the sleep state will not retain any volatile memory contents or security states. A device in the sleep state goes thru a full power up sequence upon wakeup.

A device in the standby state will retain all volatile memory contents. A device in the standby state does not go thru a power up sequence upon wakeup.

The ATAES132 exits the sleep or standby state if a wakeup event occurs on the IO pins. Wakeup is discussed in Section L.2.

See Appendix L for a detailed description of the ATAES132 sleep, standby, wakeup, and power management functions.

Table 7-69. Input parameters

	Name	Size (Bytes)	Notes
Opcode	Sleep	1	0x11
Mode	Mode	1	Bit 0 to 5: Reserved. Must be 0b Bit 6: If 0b, activate the sleep state If 1b, activate the standby state Bits7: Reserved. Must be 0b
Param1	Zero	2	Always 0x0000
Param2	Zero	2	Always 0x0000
Data	-	0	

Table 7-70. Output parameters

Name	Size (Bytes)	Notes
		No response packet is returned by the reset command

The command packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.25. TempSense Command

The TempSense command activates and reads the die temperature sensor. The temperature sensor is powered on while the command is being processed and then powered off to minimize power consumption. The temperature sensor specifications are in Table 9-8.

This command returns two 16 bit numbers, $T_{HIGH\ \&}$ T_{LOW}. The difference between the two returned values is the temperature code. The actual die temperature is calculated using a formula which is dependent upon the sensor calibration procedure. The TempCal register indicates the sensor calibration procedure used to generate the TempOffset register value (see Section E.2.17 and E.2.18). The default value of TempCal is 0x00, which indicates that the TempOffset is based on characterization.

When TempCal = 0x00 calculate the die temperature in degrees Celsius using the following formula:

$$T_{DIE}=1.65*(T_{HIGH}-T_{LOW})+67.2-T_{OFFSET}$$

The T_{OFFSET} value is stored in the first two bytes of the TempOffset register in the configuration memory. (see Section E.2.18)

The ATAES132 devices with a calibrated temperature sensor are available at additional cost. Contact Atmel for the die temperature formula for calibrated temperature sensors.

Table 7-71. Input parameters

	Name	Size (Bytes)	Notes
Opcode	TempSense	1	0x0E
Mode	Mode	1	Must be 0x00
Param1	Zero	2	Always 0x0000
Param2	Zero	2	Always 0x0000
Data	-	0	

Table 7-72. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code[(1)], per Section 6.3.
TempCodeH	2	Temperature code T_{HIGH}
TempCodeL	2	Temperature code T_{LOW}

Note: 1. In the event of a temperature sensor timeout error, a ReturnCode = 0x90 will be output along with the TempCodeH, and TempCodeL data fields. The TempCodeH and TempCodeL values are invalid when this error occurs. For any other error, only the ReturnCode value will be returned.

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

7.26. WriteCompute Command

The WriteCompute command encrypts data and computes the MAC required to execute the EncWrite command on a second ATAES132 device.

To use this command, the nonce must be identical on both devices (see Section 7.21.1) and the MacCount must have the same value. Both devices must also contain identical key values, but it is not necessary for the KeyID on the origin device to match the KeyID on the destination device. In this section, the device which encrypts data and generates the MAC is referred to as the origin device. The device checking the MAC is referred to as the destination device.

If mode bit 5, 6, or 7 is 1b, then the associated usage counter, SerialNum register value, or the first four bytes of the SmallZone register must be identical on both devices. The ManufacturingID register must be identical on both devices, since it is always included in the MAC calculation.

A valid nonce is required to run the WriteCompute command. If the KeyConfig[EKeyID].RandomNonce bit is 1b, then the nonce must be random.

The value of Param2 in the FirstBlock field must match the count field value.

Table 7-73. Input parameters

	Name	Size (Bytes)	Notes
Opcode	WriteCompute	1	0x16
Mode	Mode	1	The value of this field must match the mode field value to be used when executing the EncWrite command on the destination device
Param1	EKeyID	2	Upper byte is always 0x00. Lower byte is the pointer to the encrypt key. Legal values: 0x00 to 0x0F, 0xFF.
Param2	Count	2	Upper byte is always 0x00. Lower byte is the number of Data bytes to be encrypted.
Data1	FirstBlock	6	The value of this field must match the first authenticate-only block to be used when executing the EncWrite command on the destination device
Data2	SecondBlock	16	The value of this field must match the second authenticate-only block to be used when executing the EncWrite command on the destination device. If mode bits 5, 6, and 7 are 0b, then this field must be present, but is ignored.
Data3	InData	1 to 32	Input data to be encrypted (plaintext)

Table 7-74. Output parameters

Name	Size (Bytes)	Notes
ReturnCode	1	Upon success, 0x00 will be returned. Any command execution or validation failure generates a non-zero error code, per Section 6.3.
OutMac	16	The input MAC for the EncWrite command on the destination device
OutData	16 or 32	The encrypted data (ciphertext) to be written to the destination device using the EncWrite command

The command and response packet is transmitted as a block beginning with the count and ending with a packet checksum. This block format is described in Section 6.1.

8. Pin Lists

8.1. Package Pin List [SOIC, TSSOP, UDFN]

Table 8-75. Package pin list

Pin	Name	Description	Type
1	$\overline{CS}$	SPI mode $\overline{CS}$ / I^{2}C mode *not* used	Input
2	SO	SPI serial data out / I^{2}C mode *not* used or AuthO out	Output
3	N.C.	No connect	N.C.
4	Vss	Ground	Ground
5	SI / SDA	SPI serial data in / I^{2}C mode serial data I/O	Input / Output
6	SCK	Serial data clock	Input
7	N.C.	No connect	N.C.
8	Vcc	Power supply	Power

8.1.2. Pin Descriptions

8.1.2.1. $\overline{CS}$ [1]

SPI chip select bar input pin. In the SPI communication mode, this pin functions as the slave select input. In the I^{2}C communication mode, this pin is *not* used and should be tied to V_{CC} or V_{SS}.

8.1.2.2. SO [2]

Serial data out pin. In the SPI communication mode, this pin functions as the serial data output. In the I^{2}C communication mode, this pin is *not* used in the default configuration. It is always in the high impedance state. If Auth signaling is enabled, then this pin functions as the AuthO output. (See Section J.5)

8.1.2.3. N.C. [3]

No connect pin. This package pin is not used and can be left open by the user.

8.1.2.4. V_{SS} [4]

Ground

8.1.2.5. SI / SDA [5]

Serial data in pin. In SPI communication mode this pin functions as the serial data input. In I^{2}C communication mode this pin functions as the serial data I/O.

8.1.2.6. SCK [6]

Serial clock input pin. In both SPI and I^{2}C serial communication modes this pin is used as the serial interface clock.

8.1.2.7. N.C. [7]

No connect pin. This package pin is not used and can be left open by the user.

8.1.2.8. V_{CC} [8]

Supply Voltage

9. Electrical Characteristics

9.1. Absolute Maximum Ratings*

Operating temperature	−40°C to +85°C
Storage temperature	−65°C to + 150°C
Maximum operating voltage	6.0V
DC output current	5.0mA
Voltage on any pin	-0.7V to (V_{CC} + 0.7V)
HBM ESD	2000V minimum

Notice*: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only, and the functional operation of the device at these or any other condition beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

9.2. Reliability

The ATAES132 is fabricated with the Atmel high reliability CMOS EEPROM manufacturing technology. The reliability ratings in Table 9-1 apply to each byte of the EEPROM memory.

Table 9-76. EEPROM Reliability[1]

Parameter	Min	Typical	Max	Units
Write endurance (each byte)	100,000			Write cycles
Data retention (at 55°C)	10			Years
Data retention (at 35°C)	30	50		Years
Read endurance	Unlimited			Read cycles

Note: 1. These specifications apply to every byte of the user memory, configuration memory, and key memory. The write endurance specification also applies to the random number generator EEPROM seed register.

All values are preliminary and will be updated after characterization.

9.3. DC Characteristics

9.3.1. Supply Characteristics

Table 9-77. Supply voltage and current characteristics

Applicable over recommended operating range from T_A = −40°C to +85°C, V_{CC} = +2.5V to +5.5V (unless otherwise noted) (1)

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
V_{CC} (1)	Supply voltage		2.5		5.5	V
I_{CC1}	Supply current	V_{CC} = 3.3V at fmax (4), SO = Open, (3) Read, Write, or AES operation			6.0	mA
I_{CC2}	Supply current	V_{CC} = 5.5V at fmax (4), SO = Open, (3) Read, Write, or AES operation			10.0	mA
I_{CC3}	Idle current	V_{CC} = 3.3 V or 5.5V at fmax (4), SO = Open, (3) Waiting for a command		600	800	µA
I_{SL1}	Sleep current	V_{CC} = 3.3V, $\overline{CS}$ = V_{CC} (3) Sleep State (5)		0.1	0.25	µA
I_{SL2}	Sleep current	V_{CC} = 5.5V, $\overline{CS}$ = V_{CC} (3) Sleep State (5)		0.25	0.5	µA
I_{SB1}	Standby current	V_{CC} = 3.3V, $\overline{CS}$ = V_{CC} (3) Standby State (5)		15.0	30.0	µA
I_{SB2}	Standby current	V_{CC} = 5.5V, $\overline{CS}$ = V_{CC} (3) Standby State (5)		20.0	40.0	µA
I_{TEMP}	Temp sense supply current	Vcc = 5.5 V, Additional Icc current during temperature sense (3)		2.0		mA

Note:
1. Typical values are at 25˚ C and are for reference only. Typical values are not tested or guaranteed.
2. On power up, Vcc must rise continuously from Vss to the operating voltage with a rise time no faster than 1V/µS.
3. All input pins must be held at either Vss or Vcc during this measurement. In SPI interface mode, the $\overline{CS}$ pin must be at Vcc. In I^2C interface mode, the $\overline{CS}$ pin may be in either state.
4. Measurement is performed at the maximum serial clock frequency. In the I^2C interface mode, fmax is 1 MHz. In the SPI interface mode, fmax is 10 MHz.
5. See Appendix L for sleep and standby state information. The sleep command is described in Section 7.24.
6. The ATAES132 does not support hot swapping or hot plugging. Connecting or disconnecting this device to a system while power is energized can cause permanent damage to the ATAES132.

All values are preliminary and will be updated after characterization.

9.3.2. IO Characteristics

Table 9-78. DC characteristics

Applicable over recommended operating range from T_A = −40°C to +85°C, V_{CC} = +2.5V to +5.5V (unless otherwise noted)

Symbol	Parameter	Test conditions	Min	Max	Units
I_{LI}	Input current	V_{IN} = 0V or V_{CC}	−3.0	3.0	μA
I_{LO}	Output leakage	V_{OUT} = 0V or V_{CC}	−3.0	3.0	μA
V_{IL} (1)	Input low-voltage		−0.5	V_{CC} x 0.3	V
V_{IH} (1)	Input high-voltage		V_{CC} x 0.7	V_{CC} + 0.5	V
V_{OL1} (2)	Output low-voltage, except SI/SDA in I^2C mode	I_{OL} = 3.0mA	0	0.4	V
V_{OH1} (2)	Output high-voltage, except SI/SDA in I^2C mode	I_{OH} = −3.0mA	V_{CC} −0.8	Vcc	V
V_{OL2}	Output low-voltage, SI/SDA pin in the I^2C mode *only*	I_{OL} = 3.0mA	0	0.4	V

Note: 1. V_{IL} min and V_{IH} max are for reference only and are not tested

2. In the I^2C interface mode, if Auth signaling is enabled, the SO pin functions as the AuthO output. (See Section J.5) When AuthO is high, the V_{OH1} specification applies. When AuthO is not high, the pin is in the high impedance state – the V_{OL1} specification is not applicable.

All values are preliminary and will be updated after characterization.

9.4. AC Characteristics

Table 9-79. AC characteristics of the Atmel ATAES132

Applicable over recommended operating range from T_A = −40°C to + 85°C, V_{CC} = +2.5V to +5.5V

Symbol	Parameter	Min	Max	Units
t_{WC}	Write cycle time (1)	4.5	6.0	ms
t_{TEMP}	Temperature sensor read time	See Table 9-8		
	Command response time	See Appendix N		

Note: 1. The write cycle time includes the EEPROM erase, write, and automatic data write verification operations

All values are preliminary and will be updated after characterization.

9.4.2. Power Up, Sleep, Standby, and Wakeup Timing

Table 9-80. Power up, sleep, and wakeup timing characteristics [1]

Applicable over recommended operating range from T_A = −40°C to + 85°C, V_{CC} = +2.5V to +5.5V

Symbol	Parameter	Min	Typ	Max	Units
$t_{PU.STATUS}$	PowerUp time, status		500	600	µs
$t_{PU.RDY}$	PowerUp ready time		1200	1500	µs
t_{SB}	Sleep time, entering the standby state		65	100	µs
t_{SL}	Sleep time, entering the sleep state		55	90	µs
$t_{WupSB.STATUS}$	WakeUp status time, standby state		50	100	ns
$t_{WupSB.RDY}$	WakeUp ready time, standby state		500	600	µs
$t_{WupSL.STATUS}$	WakeUp status, sleep state		50	100	ns
$t_{WupSL.RDY}$	WakeUp ready time, sleep state		1000	1200	µs
$t_{CSS.Wup}$	$\overline{CS}$ setup time at WakeUp (see Figure 9-1)	100			ns

Notes: 1. All values are based on characterization and are not tested. Typical values are at 25° C and are for reference only.

2. See Appendix L for power up, sleep, standby, and wakeup specifications. The sleep command is described in Section 7.24.

All values are preliminary and will be updated after characterization.

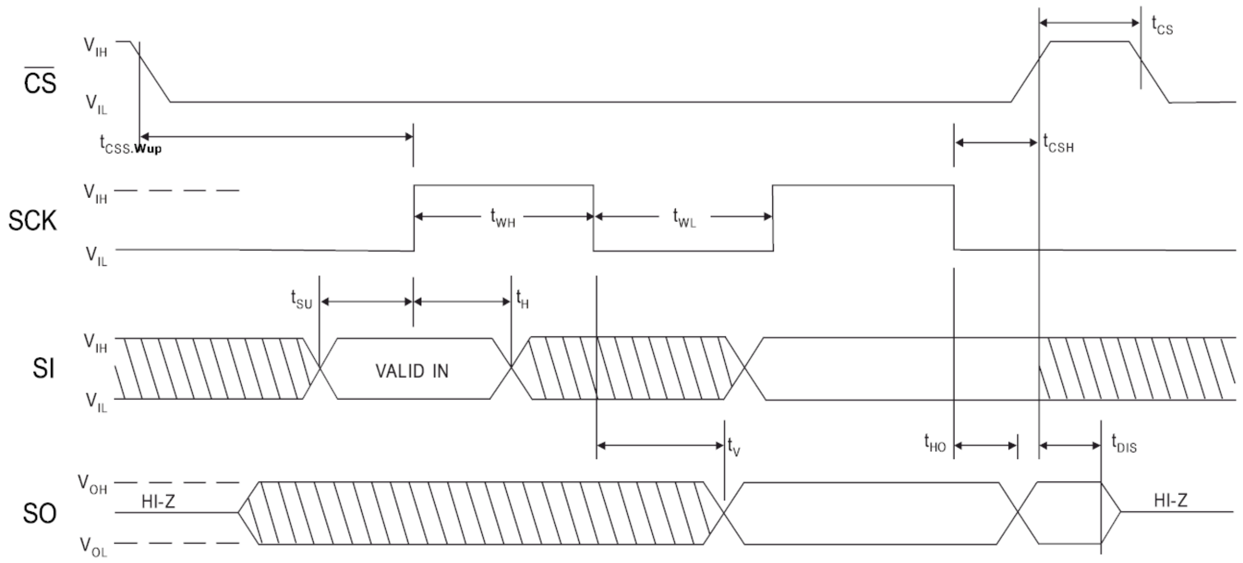


Figure 9-1. SPI interface timing, $\overline{CS}$ setup time at wakeup

9.4.3. I^{2}C Interface Timing

Table 9-81. AC characteristics of I^{2}C interface

Applicable over recommended operating range from T_A = −40°C to + 85°C, V_{CC} = +2.5V to +5.5V, CL = 1 TTL Gate and 100 pF *(unless otherwise noted)*

Symbol	Parameter	Min	Max	Units
f_{SCK}	SCK clock frequency	0	1	MHz
	SCK clock duty cycle	30	70	percent
t_{HIGH}	SCK high time	400		ns
t_{LOW}	SCK low time	400		ns
$t_{SU.STA}$	Start setup time	250		ns
$t_{HD.STA}$	Start hold time	250		ns
$t_{SU.STO}$	Stop setup time	250		ns
$t_{SU.DAT}$	Data in setup time	100		ns
$t_{HD.DAT}$	Data in hold time	0		ns
t_R	Input rise time (1)		300	ns
t_F	Input fall time (1)		100	ns
t_{AA}	Clock low to data out valid	50	550	ns
t_{DH}	Data out hold time	50		ns
t_{BUF}	Time bus must be free before a new transmission can start (1)	500		ns

Notes: 1. Values are based on characterization and are not tested

2. AC measurement conditions:

RL (connects between SDA and Vcc): 2.0 kΩ (for Vcc +2.5V to +5.0V)

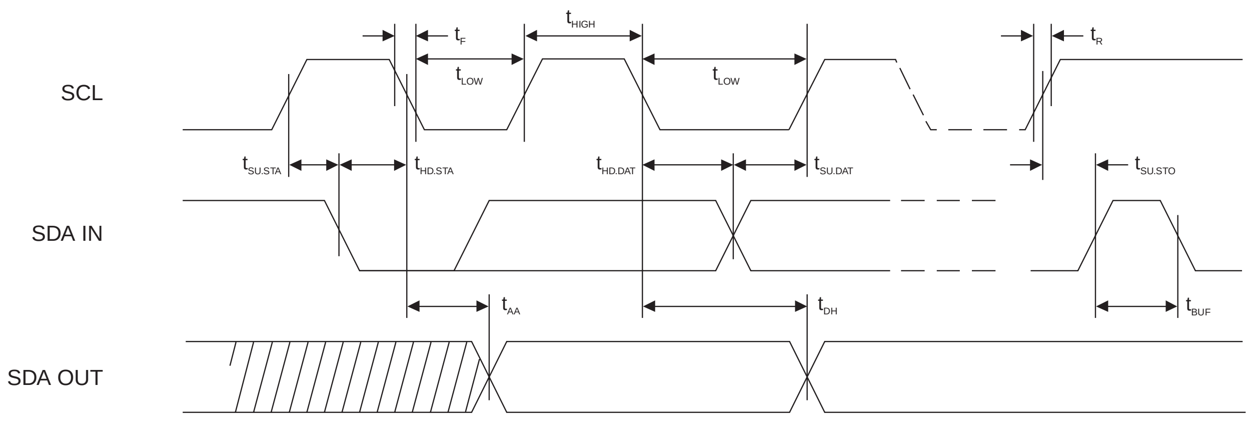
Input pulse voltages: 0.3 Vcc to 0.7 Vcc

Input rise and fall times: ≤ 50ns

Input and output timing reference voltage: 0.5 Vcc

All values are preliminary and will be updated after characterization.

Figure 9-2. I^{2}C synchronous data timing



9.4.4. SPI Interface Timing

Table 9-82. AC characteristics of SPI interface

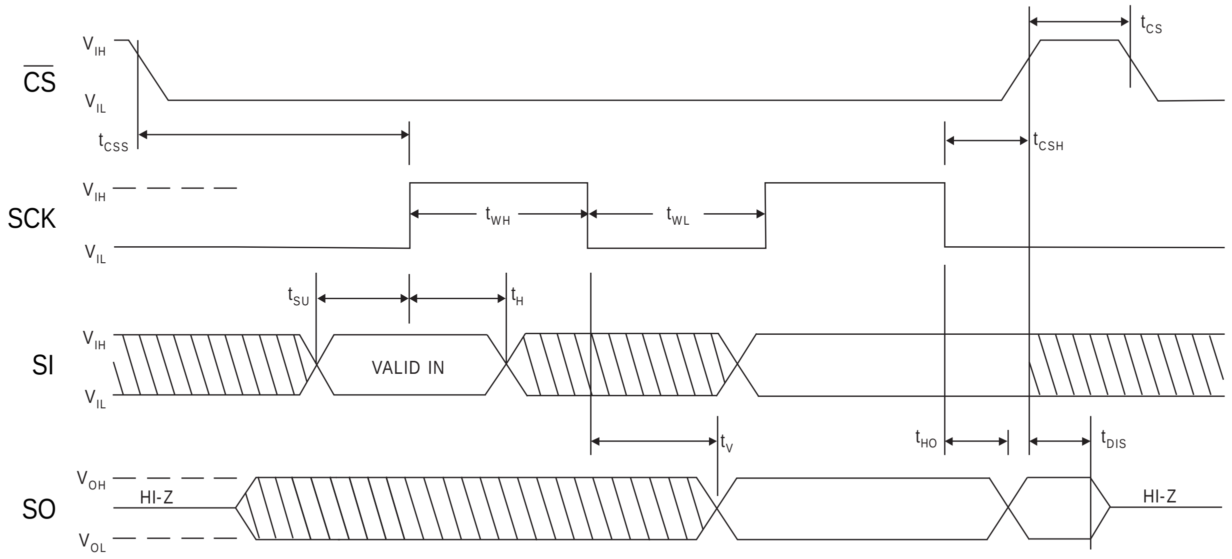
Applicable over recommended operating range from T_A = −40°C to + 85°C, V_{CC} = +2.5V to +5.5V, CL = 1 TTL Gate and 30 pF *(unless otherwise noted)*

Symbol	Parameter	Min	Max	Units
f_{SCK}	SCK clock frequency	0	10	MHz
	SCK clock duty cycle	30	70	percent
t_{WH}	SCK high time	40		ns
t_{WL}	SCK low time	40		ns
t_{CS}	$\overline{CS}$ high time	50		ns
t_{CSS}	$\overline{CS}$ setup time	50		ns
t_{CSH}	$\overline{CS}$ hold time	50		ns
t_{SU}	Data in setup time	10		ns
t_H	Data in hold time	10		ns
t_{RI}	Input rise time (1)		2	µs
t_{FI}	Input fall time (1)		2	µs
t_V	Output valid	0	40	ns
t_{HO}	Output hold time	0		ns
t_{DIS}	Output disable time		50	ns

Note: 1. Values are based on characterization and are not tested

All values are preliminary and will be updated after characterization.

Figure 9-3. SPI synchronous data timing



9.5. Temperature Sensor Characteristics

Table 9-83. Temperature sensor characteristics of the Atmel ATAES132

Applicable over recommended operating range from T_A = −40°C to + 85°C, V_{CC} = +2.5V to +5.5V (1)

Symbol	Parameter	Min	Max	Units
T_{DIE}	Die temperature	-40	85	degrees C
T_{ACCY}	Uncalibrated temperature sensor accuracy(2)		5	degrees C
$T_{ACCY.CAL1}$	Calibrated temperature sensor accuracy(2)		2	degrees C
t_{TEMP}	Temperature sensor read time		145	ms

Notes: 1. Values are based on characterization and are not tested

2. T_{ACCY} is the accuracy of the temperature sensor over the T_{DIE} temperature range when the temperature is calculated using the characterization value of the TempOffset register which is programmed at the factory. The accuracy can be improved to $T_{ACCY.CAL1}$ by performing a calibration procedure on each unit. See Section E.2.17 for additional information.

All values are preliminary and will be updated after characterization.

Appendix A. Standards and Reference Documents

A.1. National and International Standards

The Atmel ATAES132 is designed to comply with the requirements of the AES standard.

FIPS-197	Specification for the Advanced Encryption Standard (AES). 26 November 2001 Available at: http://csrc.nist.gov/groups/ST/toolkit/block_ciphers.html

A.2. References

SP800-38A	NIST Special Publication 800-38A. Recommendation for Block Cipher Modes of Operation: Methods and Techniques. December 2001 Available at: http://csrc.nist.gov/groups/ST/toolkit/BCM/current_modes.html
SP800-38C	NIST Special Publication 800-38C. Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality. May 2004 Available at: http://csrc.nist.gov/groups/ST/toolkit/BCM/current_modes.html
SP800-90	NIST Special Publication 800-90. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. (Revised) March 2007 Available at: http://csrc.nist.gov/groups/ST/toolkit/random_number.html
JEP106xx	*JEDEC Standard. Standard Manufacturer's Identification Code.* JEDEC Solid State Technology Association. Updated periodically. (JEP106AA is dated April 2009) Available at http://www.jedec.org
ISO/IEC7816-1:1998	Identification Cards - Integrated Circuit(s) Cards with Contacts - Part 1: Physical Characteristics October 1998 Available at: http://www.iso.org or http://www.ansi.org or from your National Standards Body.
ISO/IEC7816-2:2007	Identification Cards - Integrated Circuit(s) Cards with Contacts - Part 2: Dimension and Location of the Contacts October 2007 Available at: http://www.iso.org or http://www.ansi.org or from your National Standards Body.

Appendix B. Memory Map

B.1. The Atmel ATAES132 Memory Map

Reserved memory cannot be written or read.

Table B-1. The Atmel ATAES132 memory map

Byte Address	Description
0000_h-$0FFF_h$	User memory
1000_h-$EFFF_h$	*Reserved*
$F000_h$-$F05F_h$	Configuration memory – Device config
$F060_h$-$F07F_h$	Configuration memory – CounterConfig
$F080_h$-$F0BF_h$	Configuration memory – KeyConfig
$F0C0_h$-$F0FF_h$	Configuration memory – ZoneConfig
$F100_h$-$F17F_h$	Configuration memory - Counters
$F180_h$-$F1DF_h$	Configuration memory – FreeSpace
$F1E0_h$-$F1FF_h$	Configuration memory – SmallZone
$F200_h$-$F2FF_h$	Key memory
$F300_h$-$FDFF_h$	*Reserved*
$FE00_h$	Command / response memory buffer
$FE01_h$-$FFDF_h$	*Reserved*
$FFE0_h$	IO address reset
$FFE1_h$-$FFEF_h$	*Reserved*
$FFF0_h$	STATUS register
$FFF1_h$-$FFFF_h$	*Reserved*

The user memory is described in Appendix C. The configuration memory is described in Appendix E. The key memory is described in Appendix F. The virtual command memory is described in Appendix D.

B.2. EEPROM Page Boundary

The ATAES132 EEPROM has 32 byte physical pages. An EEPROM write can never cross the boundary between two physical pages. BlockRead and EncRead operations cannot cross the boundary between two physical pages. Table B-2 illustrates the page boundary locations for ATAES132.

Table B-2. EEPROM page boundary locations for the Atmel ATAES132

Address	0_h	1_h	2_h	3_h	4_h	5_h	6_h	7_h	8_h	9_h	A_h	B_h	C_h	D_h	E_h	F_h
$XX00_h$-$XX0F_h$	32 Byte EEPROM Page															
$XX10_h$-$XX1F_h$																
$XX20_h$-$XX2F_h$	32 Byte EEPROM Page															
$XX30_h$-$XX3F_h$																
$XX40_h$-$XX4F_h$	32 Byte EEPROM Page															
$XX50_h$-$XX5F_h$																
$XX60_h$-$XX6F_h$	32 Byte EEPROM Page															
$XX70_h$-$XX7F_h$																
$XX80_h$-$XX8F_h$	32 Byte EEPROM Page															
$XX90_h$-$XX9F_h$																
$XXA0_h$-$XXAF_h$	32 Byte EEPROM Page															
$XXB0_h$-$XXBF_h$																
$XXC0_h$-$XXCF_h$	32 Byte EEPROM Page															
$XXD0_h$-$XXDF_h$																
$XXE0_h$-$XXEF_h$	32 Byte EEPROM Page															
$XXF0_h$-$XXFF_h$																

Appendix C. User Memory Map

The 32Kbit user memory consists of 16 user zones, each containing 2Kbits (256 bytes). The physical page size is 32 bytes – write operations cannot cross page boundaries.

Every memory zone has an independent set of access restrictions, and all bytes within a zone have the same access restrictions. The configuration memory (Appendix E) contains an access register for each memory zone which defines the access requirements for the user zone.

Table C-3. User memory map

Byte address	Description
0000_h-$00FF_h$	User Zone 0
0100_h-$01FF_h$	User Zone 1
0200_h-$02FF_h$	User Zone 2
0300_h-$03FF_h$	User Zone 3
0400_h-$04FF_h$	User Zone 4
0500_h-$05FF_h$	User Zone 5
0600_h-$06FF_h$	User Zone 6
0700_h-$07FF_h$	User Zone 7
0800_h-$08FF_h$	User Zone 8
0900_h-$09FF_h$	User Zone 9
$0A00_h$-$0AFF_h$	User Zone A
$0B00_h$-$0BFF_h$	User Zone B
$0C00_h$-$0CFF_h$	User Zone C
$0D00_h$-$0DFF_h$	User Zone D
$0E00_h$-$0EFF_h$	User Zone E
$0F00_h$-$0FFF_h$	User Zone F

Appendix D. Command Memory Map

The ATAES132 commands are executed by writing the command packet to the virtual memory using the standard SPI or I^{2}C write commands. The response packet is retrieved by reading it from the virtual memory using the standard SPI or I^{2}C read commands. The command/response memory buffer is 64 bytes.

The ATAES132 commands are executed by writing the command packet to virtual memory at starting address 0xFE00 using the standard write commands (see Appendix J and Appendix K). The response packet is retrieved by reading from the virtual memory at starting address 0xFE00 using the standard read commands. The device status register (STATUS) is located at 0xFFF0 (see Appendix G).

To reset the address pointer in the command/response memory buffer to the base address of the buffer, the host writes one or more bytes to the IO address reset register at address 0xFFE0 using the standard write command. Any value can be written to the IO address reset register to reset the buffer address pointer.

Table D-4. Command/response virtual memory map

Byte address	Description
FE00$_h$	Command/response memory buffer
FE01$_h$-FFDF$_h$	*Reserved*
FFE0$_h$	IO address reset
FFE1$_h$-FFEF$_h$	*Reserved*
FFF0$_h$	STATUS register
FFF1$_h$-FFFF$_h$	*Reserved*

D.2. Command Memory Buffer

The command memory buffer is a write-only buffer memory that is used by writing a command block to the buffer at the base address of 0xFE00. After the host completes its write operation to the buffer, the ATAES132 verifies the integrity of the block by checking the 16-bit checksum, and then executes the requested operation. See Section 6.1 for a description of the crypto command block.

Write operations which begin at any other location within the buffer are invalid and will not be processed by the ATAES132.

Table D-5. Command memory buffer map

Base address	Base + 1	Base + 2	Base + 3	......	......	......	......	Base + N-2	Base + N-1
Count	Opcode	Mode	Param1	Param1	Param2	....	DataX	CRC1	CRC2

D.2.2. Using the Command Memory Buffer

The host should write a single byte to the IO address reset register before writing a new command block to the command memory buffer – this resets the buffer address pointer to the base address. The host then writes the ATAES132 command block to the buffer using one or more standard SPI or I^2C write commands. After the entire command block is written by the host microcontroller, the ATAES132 checks the 16-bit checksum and executes the command. The host should read the STATUS register to determine if an error occurred or if the response is ready to be read.

If a checksum error occurs, then the buffer address pointer must be reset by the host before the command block is retransmitted. If no errors occur, then the response can be read from the response memory buffer as described in Section D.3.2 (See Appendix G for examples).

The command memory buffer size is 64 bytes. If the host writes more than 64 bytes to the buffer, it will cause a buffer overflow error. If the host hardware must send more bytes to the ATAES132 than are required to transmit a command block (due to host hardware limitations), then all bytes transmitted after the block checksum must contain 0xFF.

D.3. Response Memory Buffer

The response memory buffer is a read-only memory buffer that is used by reading a response from the buffer at the base address of 0xFE00. The base address of the response memory buffer contains the first byte of the response packet after a crypto command is processed. See Section 6.1 for a description of the crypto response packet.

Read operations which begin at any location above the base address are invalid and will either be NAKed (in I^2C mode) or be ignored (output will tri-state in SPI mode).

Table D-6. Response memory buffer map following a crypto command

Base Address	Base + 1	Base + 2	Base + 3	......	......	......	......	Base + N-2	Base + N-1
Count	ReturnCode	Data1	Data2	Data3	....	....	DataX	CRC1	CRC2

The response memory buffer is also used to report errors that occur during execution of standard I^2C or SPI write commands. When the I^2C or SPI command execution is complete (as indicated by the STATUS register), the response memory buffer contains a block containing an error code (ReturnCode) if an error occurred, otherwise it contains a block containing ReturnCode = 0x00. Reading the response memory buffer does not alter the contents of the response memory buffer or the STATUS register (see Appendix G). See Section 6.3 for the error descriptions.

Table D-7. Response memory buffer map following a standard I^2C or SPI write operation

Base Address	Base + 1	Base + N-2	Base + N-1	......	......	......	......	......	......
Count	ReturnCode	CRC1	CRC2	FF_h	FF_h	FF_h	FF_h	FF_h	FF_h

D.3.2. Using the Response Memory Buffer

After an ATAES132 command is executed, the RRDY bit of the STATUS register is set to 1b to indicate that a new response is available in the response memory buffer. The host reads the response block from the buffer using one or more standard SPI or I^2C read commands. After the entire response block is read, the host microcontroller checks the 16-bit checksum.

If a checksum error occurs, then the buffer address pointer must be reset by the host before the response block is re-read. If the host reads more bytes from the response buffer than necessary to retrieve the block, then all bytes after the block checksum will contain 0xFF (See Appendix G for examples). The response memory buffer size is 64 bytes.

D.4. IO Address Reset Register

Writing the IO address reset register (address 0xFFE0) with any value causes the address pointers in the command memory buffer and the response memory buffer to be reset to the base address of the buffer. The IO address reset register can be written with 1 to 32 bytes of data without generating an error; the data bytes will be ignored.

Writing the IO address reset register does not alter the contents of the response memory buffer, or the value of the STATUS register. Writing the IO address reset register clears the command memory buffer (See Appendix G for examples).

D.5. Device Status Register (STATUS)

The device status register is used for handshaking between the host microcontroller and the ATAES132. The host is expected to read the STATUS register before sending a command or reading a response. See Appendix G for the definition and behavior of the STATUS register. If the ATAES132 is configured in SPI Interface mode, the STATUS register can also be read using the SPI RDSR command as described in Section K.3.6.

Reading the STATUS register does not alter the contents of the command memory buffer, the response memory buffer, or the value of the STATUS register.

Appendix E. Configuration Memory Map

The ATAES132 configuration memory is located from address 0xF000 to address 0xF1FF. The configuration memory can always be read using the BlockRead command (see Section 7.4). See Section E.2 for descriptions of each configuration register. A memory map showing the default register values is in Appendix O.

E.1. Configuration Memory Map

Table E-8. The configuration memory map for the Atmel ATAES132

Address	0_h / 8_h	1_h / 9_h	2_h / A_h	3_h / B_h	4_h / C_h	5_h / D_h	6_h / E_h	7_h / F_h
F000$_h$-F007$_h$	SerialNum							
F008$_h$-F00F$_h$	LotHistory							
F010$_h$-F017$_h$	Jedec		Reserved			Algorithm		EEPageSize
F018$_h$-F01F$_h$	EncReadSize	EncWrtSize	DeviceNum	Reserved				
F020$_h$-F027$_h$	LockKeys	LockSmall	LockConfig	Reserved				
F028$_h$-F02F$_h$	Reserved			ManufacturingID		PermConfig	Reserved	
F030$_h$-F037$_h$	Reserved							
F038$_h$-F03F$_h$								
F040$_h$-F047$_h$	I^{2}CAddr	ChipConfig	TempCal	TempOffset				
F048$_h$-F04F$_h$	TempOffset			RFU				
F050$_h$-F057$_h$	RFU							
F058$_h$-F05F$_h$								
F060$_h$-F067$_h$	CounterConfig 00		CounterConfig 01		CounterConfig 02		CounterConfig 03	
F068$_h$-F06F$_h$	CounterConfig 04		CounterConfig 05		CounterConfig 06		CounterConfig 07	
F070$_h$-F077$_h$	CounterConfig 08		CounterConfig 09		CounterConfig 0A		CounterConfig 0B	
F078$_h$-F07F$_h$	CounterConfig 0C		CounterConfig 0D		CounterConfig 0E		CounterConfig 0F	
F080$_h$-F087$_h$	KeyConfig 00				KeyConfig 01			
F088$_h$-F08F$_h$	KeyConfig 02				KeyConfig 03			
F090$_h$-F097$_h$	KeyConfig 04				KeyConfig 05			
F098$_h$-F09F$_h$	KeyConfig 06				KeyConfig 07			
F0A0$_h$-F0A7$_h$	KeyConfig 08				KeyConfig 09			
F0A8$_h$-F0AF$_h$	KeyConfig 0A				KeyConfig 0B			
F0B0$_h$-F0B7$_h$	KeyConfig 0C				KeyConfig 0D			
F0B8$_h$-F0BF$_h$	KeyConfig 0E				KeyConfig 0F			
F0C0$_h$-F0C7$_h$	ZoneConfig 00				ZoneConfig 01			
F0C8$_h$-F0CF$_h$	ZoneConfig 02				ZoneConfig 03			
F0D0$_h$-F0D7$_h$	ZoneConfig 04				ZoneConfig 05			
F0D8$_h$-F0DF$_h$	ZoneConfig 06				ZoneConfig 07			
F0E0$_h$-F0E7$_h$	ZoneConfig 08				ZoneConfig 09			
F0E8$_h$-F0EF$_h$	ZoneConfig 0A				ZoneConfig 0B			
F0F0$_h$-F0F7$_h$	ZoneConfig 0C				ZoneConfig 0D			
F0F8$_h$-F0FF$_h$	ZoneConfig 0E				ZoneConfig 0F			

Address	0$_h$ / 8$_h$	1$_h$ / 9$_h$	2$_h$ / A$_h$	3$_h$ / B$_h$	4$_h$ / C$_h$	5$_h$ / D$_h$	6$_h$ / E$_h$	7$_h$ / F$_h$
F100$_h$-F107$_h$	Counter 00							
F108$_h$-F10F$_h$	Counter 01							
F110$_h$-F117$_h$	Counter 02							
F118$_h$-F11F$_h$	Counter 03							
F120$_h$-F127$_h$	Counter 04							
F128$_h$-F12F$_h$	Counter 05							
F130$_h$-F137$_h$	Counter 06							
F138$_h$-F13F$_h$	Counter 07							
F140$_h$-F147$_h$	Counter 08							
F148$_h$-F14F$_h$	Counter 09							
F150$_h$-F157$_h$	Counter 0A							
F158$_h$-F15F$_h$	Counter 0B							
F160$_h$-F167$_h$	Counter 0C							
F168$_h$-F16F$_h$	Counter 0D							
F170$_h$-F177$_h$	Counter 0E							
F178$_h$-F17F$_h$	Counter 0F							
F180$_h$-F187$_h$	FreeSpace							
F188$_h$-F18F$_h$								
F190$_h$-F197$_h$								
F198$_h$-F19F$_h$								
F1A0$_h$-F1A7$_h$								
F1A8$_h$-F1AF$_h$								
F1B0$_h$-F1B7$_h$								
F1B8$_h$-F1BF$_h$								
F1C0$_h$-F1C7$_h$								
F1C8$_h$-F1CF$_h$								
F1D0$_h$-F1D7$_h$								
F1D8$_h$-F1DF$_h$								
F1E0$_h$-F1E7$_h$	SmallZone							
F1E8$_h$-F1EF$_h$								
F1F0$_h$-F1F7$_h$								
F1F8$_h$-F1FF$_h$								

The configuration memory map in Table E-1 is color coded. The registers shown in orange are locked at the factory and cannot be changed by the customer. The contents of the lock registers (shown in blue) can only be changed by using the lock command (see Section 7.19).

Configuration registers shaded with green can be written by the customer prior to locking (by setting LockConfig to 0x00 using the lock command). The SmallZone (shown in yellow) can be written by the customer prior to locking (by setting LockSmall to 0x00 using the lock command) – SmallZone is locked separately from the remainder of the configuration memory.

E.2. Configuration Register Descriptions

Each register in the configuration memory is briefly described in this section. References are provided to detail information in other sections of this specification. The registers are described in the same order that they occur in the memory map in Section E.1.

E.2.1. SerialNum Register

SerialNum is an eight byte read-only register that is programmed by Atmel at the factory. The contents of this register are guaranteed to be unique on each unit over the production life of the ATAES132 product family. The contents of this register can optionally be included in the cryptographic calculations by setting mode bit 6 to 1b as described in the command definitions in Section 7. This register cannot be changed by the customer.

It is recommended that the SerialNum register value be used to perform key diversification.

E.2.2. LotHistory Register

LotHistory is an eight byte read-only register that is programmed by Atmel at the factory. This register contains proprietary data which is not intended for customer use. This register cannot be changed by the customer.

E.2.3. Jedec Register

Jedec is a two byte read-only register that is programmed by Atmel at the factory. The Jedec register always contains 0x001F, which is the Jedec manufacturing identification code assigned to Atmel. This register cannot be changed by the customer.

E.2.4. Algorithm Register

Algorithm is a two byte read-only register that is programmed by Atmel at the factory. The default value 0x0000 indicates 128 bit AES-CCM. This register cannot be changed by the customer.

E.2.5. EEPageSize Register

EEPageSize is a one byte read-only register that is programmed by Atmel at the factory. The default value 0x20 indicates a 32 byte physical EEPROM page size. This register cannot be changed by the customer.

E.2.6. EncReadSize Register

EncReadSize is a one byte read-only register that is programmed by Atmel at the factory. The default value 0x20 indicates that 32 bytes is the maximum data length which can be returned by the EncRead command. This register cannot be changed by the customer.

E.2.7. EncWrtSize Register

EncWrtSize is a one byte read-only register that is programmed by Atmel at the factory. The default value 0x20 indicates that 32 bytes is the maximum data length which can be written using the EncWrite command. This register cannot be changed by the customer.

E.2.8. DeviceNum Register

DeviceNum is a one byte read-only register that is programmed by Atmel at the factory. This byte indicates the device type (32K bit, ATAES1xx family). The INFO command returns this byte, along with a hardware revision byte as shown in Table E-2. This register cannot be changed by the customer.

Table E-9. DeviceNum coding for INFO response and DeviceNum in configuration memory register

Description	INFO DeviceNum	DeviceNum register
Early Pre-Production Samples	0x0A02	0x0A
Pre-Production Samples	0x0A04	0x0A

See Section 7.12 for the INFO command description.

E.2.9. LockKeys Register

LockKeys is a one byte register that controls write access to key memory. The default value of LockKeys is the unlocked state (0x55). The LockKeys register can only be changed by using the lock command (see Section 7.19). After the lock command is ran, this register will contain 0x00 and the key memory will be locked. It is impossible to unlock memory which has been locked.

E.2.10. LockSmall Register

LockSmall is a one byte register that controls write access to the SmallZone register. The default value of LockSmall is the unlocked state (0x55). The LockSmall register can only be changed by using the lock command (see Section 7.19). After the lock command is ran, this register will contain 0x00 and the SmallZone will be locked. It is impossible to unlock memory which has been locked.

E.2.11. LockConfig Register

LockConfig is a one byte register that controls write access to configuration memory, except the SmallZone register. The default value of LockConfig is the unlocked state (0x55). The LockConfig register can only be changed by using the lock command (see Section 7.19). After the lock command is ran, this register will contain 0x00 and the configuration memory will be locked, except for the SmallZone register, which is controlled by the LockSmall register. It is impossible to unlock memory which has been locked.

If the LockConfig register is unlocked (0x55) then the random number generator is latched in test mode and the random command will always return 16 bytes of 0xA5. The KeyCompute and nonce commands will create non-random results when the RNG is in test mode. If the LockConfig register is locked (0x00), then the RNG generates random numbers and the random, KeyCompute, and nonce commands function normally.

E.2.12. Reserved Registers

Any configuration memory locations which are identified as reserved in Table E-1, memory map, are reserved by Atmel for future use. All reserved registers are read-only registers that are programmed by Atmel at the factory. These memory locations are programmed with Atmel proprietary data. The contents of the reserved registers will vary and are not intended for any customer use. These registers cannot be changed by the customer.

E.2.13. ManufacturingID Register

ManufacturingID is a two byte read-only register that is programmed by Atmel at the factory. This register contains a customer spicfic value. The default ManufacturingID register contains 0x0000. This register cannot be changed by the customer.

E.2.14. PermConfig Register

PermConfig is a one byte read-only register that is programmed by Atmel at the factory. This register cannot be changed by the customer. The default value 0x01 enables all cryptographic commands.

Table E-10. PermConfig register definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved for future use							EncryptE

If the EncryptE bit is 1b, then the encrypt, decrypt and legacy command availability is determined by the ChipConfig.EncDecrE and ChipConfig.LegacyE bits. If the EncryptE bit is 0b, then the encrypt, decrypt, and legacy commands are disabled. See the ChipConfig register definition in Section E.2.16 for additional information.

E.2.15. I^2CAddr Register

I^2CAddr is a one byte register that controls the ATAES132 serial interface. The customer can write the I^2CAddr register with the standard I^2C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11).

Table E-11. I^2CAddr register definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I^2C device address							SPI/I^2C

Bit 0 selects the serial interface mode; 0b selects SPI interface mode, while 1b selects I^2C interface mode. If bit 0 is 0b, then the contents of bits 1 to 7 are ignored.

The default value of the I^2CAddr register depends on the ordering code (see Appendix Q); I^2CAddr is 0xA1 (I^2C Device Address is 0xA0) for catalog numbers with an I^2C interface configuration, I^2CAddr is 0x00 for catalog numbers with a SPI interface configuration. See Appendix J for the I^2C interface specifications. See Appendix K for the SPI interface specifications.

E.2.16. ChipConfig Register

ChipConfig is a one byte register that controls device level functionality of the ATAES132. The customer can write the ChipConfig register with the standard I^2C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11).

Table E-12. ChipConfig register definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PowerUpState		Reserved for future use				EncDecrE	LegacyE

If the ChipConfig.LegacyE bit is 1b, then the legacy command (Section 7.18) is enabled. If ChipConfig.LegacyE is 0b, then a parse error ReturnCode will be returned in response to a legacy command. If the ChipConfig.EncDecrE bit is 1b, then the encrypt command (Section 7.10) and decrypt command (Section 7.8) are enabled. If ChipConfig.EncDecrE is 0b, then a parse error ReturnCode will be returned in response to an encrypt command or decrypt command.

The default configuration of the PermConfig register allows the customer to control the availability of the encrypt, decrypt, and legacy commands using the ChipConfig register. However, the ChipConfig.EncDecrE bit and ChipConfig.LegacyE bit will be ignored if the ATAES132 is configured at the factory to disable external encryption (see the PermConfig Register definition in Section E.2.14).

Table E-13. Coding of the PowerUpState bits in the ChipConfig register

Bit 7	Bit 6	Description
1	1	Device goes to the Active State at Power Up
1	0	
0	1	Device goes to the Standby State at Power Up
0	0	Device goes to the Sleep State at Power Up

The ChipConfig.PowerUpState bits are used to configure the behavior of the ATAES132 at initial power up. Table E-6 shows the definition of the ChipConfig.PowerUpState bits. See Appendix L for detailed information regarding the ATAES132 power management functions.

The default value of the ChipConfig register is 0xC3. In this configuration, the ATAES132 goes to the active state at power up, the encrypt, decrypt and legacy commands are enabled.

E.2.17. TempCal Register

The TempCal register contains a value indicating the calibration procedure used to determine the TempOffset register value at the factory. The temperature sensor calibration procedure determines the accuracy of the die temperature measurement. The default value of TempCal is 0x00.

Table E-14. TempCal register definition

TempCal Value	Description	Accuracy(1)	TempOffset
0x00	Uncalibrated temperature sensor	TACCY	2 bytes
0x01	Calibrated temperature sensor, procedure 1	TACCY.CAL1	
0xFF	Temperature sensor Offset unknown	N/A	N/A

Note: 1. The die temperature accuracy specifications are listed in Table 9-8

The customer can write the TempCal register with the standard I^{2}C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11).

E.2.18. TempOffset Register

TempOffset is an eight byte register that contains a temperature offset value which is used to calculate the die temperature measured by the high-accuracy internal sensor. The customer can write the TempOffset register with the standard I^{2}C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11).

The default TempOffset register value programmed by Atmel at the factory is determined by characterization, as indicated by a TempCal register value of 0x00. Only the first two bytes of the TempOffset register are used to calculate the die temperature when the TempCal register value is 0x00. See Section 7.25 for the temperature calculation.

It is possible to substantially improve the accuracy of the temperature sensor by measuring the temperature offset for each unit, and overwriting the TempOffset register with the new offset value during personalization. Contact Atmel for calibrated temperature sensor information.

E.2.19. RFU Registers

Any configuration memory locations which are identified as RFU in Table E-1, memory map, are registers in customer writable memory that are reserved by Atmel for future use (in a future ATAES family product or a major product revision). The default value of the RFU registers is 0xFF.

The customer can write the RFU registers with the standard I^{2}C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11). The RFU registers should only be programmed to 0xFF; all other values are prohibited.

E.2.20. CounterConfig Registers

The 16 CounterConfig registers are used to individually configure the 16 Counters. Each CounterConfig register controls one counter. CounterConfig 00 controls Counter 00, CounterConfig 01 controls Counter 01, etc.

Each CounterConfig register is a two byte array which is stored as shown in Table E-8. The customer can write the CounterConfig registers with the standard I^{2}C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11). See Appendix H for additional counter information.

Table E-15. Partial configuration memory map showing CounterConfig register byte locations for four registers

Address	0_h	1_h	2_h	3_h	4_h	5_h	6_h	7_h
$F060_h$-$F067_h$	CounterConfig 0		CounterConfig 1		CounterConfig 2		CounterConfig 3	
	Byte 0	Byte 1	Byte 0	Byte 1	Byte 0	Byte 1	Byte 0	Byte 1

The CounterConfig register imposes restrictions on the usage of the counter command (see Section 7.5) with a counter. The CounterConfig bits have no impact on the functionality of a key usage counter. If a counter is identified in a KeyConfig register (see Section E.2.21) as a Key Usage Counter, then the counter will increment each time the key is used. The CounterConfig[CntID].IncrementOK is typically set to 0b to prohibit the counter command from incrementing a key usage counter.

Table E-16. Definition of the CounterConfig register bits(1)

CounterConfig Field	Byte	Bit	Description
IncrementOK	0	0	If 1b, then increments using the Counter command are permitted If 0b, then increments using the Counter command are prohibited
RequireMAC	0	1	If 1b, then the increment operation requires an input MAC If 0b, then an input MAC is prohibited
Reserved	0	2 to 7	Reserved for future use. All bits must be 0b
IncrID	1	0 to 3	KeyID of the key used to generate the Counter command input MAC for increment operations
MacID	1	4 to7	KeyID of the key used to generate the Counter command output MAC for counter read operations

Note: 1. Changes to the CounterConfig registers take effect immediately, which allow the functionality to be verified during the personalization process

E.2.21. KeyConfig Registers

The 16 KeyConfig registers are used to individually configure the 16 keys. Each KeyConfig register controls one key. KeyConfig 00 controls Key 00, KeyConfig 01 controls Key 01, etc.

Each KeyConfig register is a four byte array which is stored as shown in Table E-10. The customer can write the KeyConfig registers with the standard I^{2}C or SPI write commands unless the Configuration Memory has been locked (see the LockConfig register definition in Section E.2.11).

Table E-17. Partial configuration memory map showing KeyConfig register byte locations for two registers

Address	0_h	1_h	2_h	3_h	4_h	5_h	6_h	7_h
$F080_h$-$F087_h$	KeyConfig 0				KeyConfig 1			
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 0	Byte 1	Byte 2	Byte 3

A key can be disabled by setting KeyConfig[KeyN].AuthKey to 1b and KeyConfig[KeyN].LinkPointer to contain "KeyN", where the KeyN = KeyID of the key being configured.

Table E-18. Definition of the KeyConfig register bits (1)(3)

CounterConfig Field	Byte	Bit	Description
ExternalCrypto	0	0	If 1b, then the key can be used with the encrypt and decrypt commands (2) If 0b, then the encrypt and decrypt commands are prohibited
InboundAuth	0	1	If 1b, then the key can only be used by the Auth command for inbound only or mutual authentication. Key can not be used by any other command, but KeyID can be the target of a key management command. If 0b, then key can be used for any purpose not prohibited by another KeyConfig bit, including Outbound Only authentication
RandomNonce	0	2	If 1b, then operations using this key require a Random Nonce. (see Section 7.20) If 0b, then the nonce is not required to be random
LegacyOK	0	3	If 1b, then this key can be used with the legacy command If 0b, then the key cannot be used with the legacy command
AuthKey	0	4	If 1b, then this key requires prior authentication using the KeyID stored in LinkPointer If 0b, then prior authentication is not required
Child	0	5	If 1b, then key is permitted to be the target of a KeyCompute and/or KeyLoad command If 0b, then this use is prohibited
Parent	0	6	If 1b, then key may be used as the VolatileKey parent by the KeyCompute or KeyLoad commands. This key may also be used as the decrypt key by the KeyImport command when the target key is the VolatileKey. (see Section 4.3) If 0b, then this use is prohibited
ChangeKeys	0	7	If 1b, then key updates are permitted after locking. The new key is written using the EncWrite command with a MAC generated with the current value of key. (see Section 7.11) If 0b, then key updates with EncWrite command are prohibited
CounterLimit	1	0	If 1b, usage count limits are enabled for this key (see CounterNum) If 0b, then there are no usage limits
ChildMac	1	1	If 1b, then an input MAC is required to modify this key using the KeyCompute command If 0b, the KeyCompute command does not require an input MAC (it will be ignored if provided)
AuthOut	1	2	If 1b, then I^2C Auth signaling is enabled for this key (see Section J.5) If 0b, then I^2C Auth signaling is disabled for this key
AuthOutHold	1	3	If 1b, the I^2C AuthO output state is unchanged when an Authentication Reset is executed using this key If 0b, then the I^2C AuthO output is reset when an Authentication Reset is executed using this key (see Section J.5)
ImportOK	1	4	If 1b, then this key is permitted to be the target of a KeyImport command. If 0b, then the KeyImport command is prohibited
ExportAuth	1	5	If 1b, then the KeyExport and KeyCompute commands require prior authentication using the KeyID stored in LinkPointer If 0b, then prior authentication is not required

CounterConfig Field	Byte	Bit	Description
TransferOK	1	6	If 1b, then this key is permitted to be the target of a KeyTransfer command. (See Section 7.17) If 0b, then the KeyTransfer command is prohibited
AuthCompute	1	7	If 1b, then this key can be used with the AuthCompute command If 0b, then the key cannot be used with the AuthCompute command
LinkPointer	2	0 to 3	For child keys, stores the ParentKeyID For all other keys, the KeyID of the authorizing key (see AuthKey)
CounterNum	2	4 to 7	Stores the CntID of the counter attached to this key for usage limits and/or for MAC calculation. MAC calculations will include the counter if command mode bit five is 1b even if key usage limits are disabled
Reserved	3	0 to 7	Reserved for future use. All bits must be 0b

Notes:

1. Changes to the KeyConfig registers take effect immediately, which allows the functionality to be verified during the personalization process.
2. **Warning:** Since the encrypt command does not include an input MAC, the encrypt command can be exhaustively ran with selected input data to attack the key. Requiring authentication prior to allowing encryption makes these attacks more difficult. To require prior authentication, the AuthKey, and RandomNonce bits must be set to 1b.
3. A key can be disabled by setting KeyConfig[KeyN].AuthKey to 1b and KeyConfig[KeyN].LinkPointer to contain "KeyN", where KeyN = KeyID of the key being configured

E.2.22. ZoneConfig Registers

The 16 ZoneConfig registers are used to individually configure the 16 user zones. Each ZoneConfig register controls one user zone. ZoneConfig 00 controls user zone 00, ZoneConfig 01 controls User Zone 01, etc.

Each ZoneConfig register is a four byte array which is stored as shown in Table E-12. The customer can write the ZoneConfig registers with the standard I^{2}C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11).

Table E-19. Partial configuration memory map showing ZoneConfig register byte locations for the two registers

Address	0_h	1_h	2_h	3_h	4_h	5_h	6_h	7_h
$F0C0_h$-	ZoneConfig 0				ZoneConfig 1			
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 0	Byte 1	Byte 2	Byte 3

Table E-20. Definition of the ZoneConfig register bits [(1)]

CounterConfig Field	Byte	Bit	Description
AuthRead	0	0	If 1b, then authentication is required to read data If 0b, then authentication is not required to read data
AuthWrite	0	1	If 1b, then authentication is required to write data If 0b, then authentication is not required to write data
EncRead	0	2	If 1b, then encryption is required to read data. If 0b, then encryption is not required to read data
EncWrite	0	3	If 1b, then encryption is required to write data If 0b, then encryption is not required to write data
WriteMode	0	4 to 5	If 00b, then this zone is permanently read/write If 01b, then this zone is permanently read-only If 10b, then the ReadOnly byte determines if writes are permitted If 11b, then the ReadOnly byte determines if writes are permitted and the lock command must include an authenticating MAC calculated using the KeyID stored in ZoneConfig[UZ].WriteID.
UseSerial	0	6	If UseSerial = 1b and EncWrite = 1b, then the SerialNum must be included in EncWrite operations. If EncWrite = 0b, then this bit is ignored.
UseSmall	0	7	If UseSmall = 1b and EncWrite = 1b, the first 4 bytes of SmallZone must be included in EncWrite operations. If EncWrite = 0b, then this bit is ignored.
ReadID	1	0 to 3	KeyID which is used to encrypt data read from this zone The same key is used to generate the MAC
AuthID	1	4 to 7	KeyID which is used for inbound authentication before access is permitted
Reserved	2	0 to 3	Reserved for future use. All bits must be 0b
WriteID	2	4 to 7	KeyID which is used to decrypt data written to this zone The same key is used to verify the MAC
ReadOnly	3	0 to 7	The contents of this byte are ignored unless WriteMode contains 10b or 11b If 0x55, then the user zone is read/write If any other value, then the user zone is read-only This byte can be updated after the configuration memory is locked by using the lock command (See Section 7.19.)

Note: 1. Most changes to the ZoneConfig registers take effect immediately. Changes to the AuthRead and EncRead bits do not affect the SPI or I^2C read command until the next reset or power up.

E.2.23. Counter Registers

The 16 Counter registers are used to store the counter values. The default value of the counters is equivalent to a count value of zero. See Appendix H for Counter information.

The customer can write the counter registers with the standard I^{2}C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11).

E.2.24. FreeSpace Register

The FreeSpace register is 96 bytes of memory for storage of customer data. The customer can write the FreeSpace register with the standard I^{2}C or SPI write commands unless the configuration memory has been locked (see the LockConfig register definition in Section E.2.11).

The default value of the FreeSpace register is 0xFF in all bytes. The FreeSpace register can be programmed with any value – the contents will not change the behavior of the ATAES132.

E.2.25. SmallZone Register

The SmallZone register is 32 bytes of memory for storage of customer data. Optionally, the first four bytes of the SmallZone may be included in cryptographic calculations by setting mode bit 7 to 1b as described in the command definitions in Section 7. The customer can write the SmallZone register with the standard I^{2}C or SPI write commands unless the SmallZone register has been locked (see the LockSmall register definition in Section E.2.10).

The default value of the SmallZone register is 0xFF in all bytes. The SmallZone register can be programmed with any value – the contents will not change the behavior of the ATAES132.

Appendix F. Key Memory Map

Table F-21. The key memory map.The ATAES132 key memory is located at address 0xF200.

Address	0_h / 8_h	1_h / 9_h	2_h / A_h	3_h / B_h	4_h / C_h	5_h / D_h	6_h / E_h	7_h / F_h
$F200_h$-$F207_h$	Key 00							
$F208_h$-$F20F_h$								
$F210_h$-$F217_h$	Key 01							
$F218_h$-$F21F_h$								
$F220_h$-$F227_h$	Key 02							
$F228_h$-$F22F_h$								
$F230_h$-$F237_h$	Key 03							
$F238_h$-$F23F_h$								
$F240_h$-$F247_h$	Key 04							
$F248_h$-$F24F_h$								
$F250_h$-$F257_h$	Key 05							
$F258_h$-$F25F_h$								
$F260_h$-$F267_h$	Key 06							
$F268_h$-$F26F_h$								
$F270_h$-$F277_h$	Key 07							
$F278_h$-$F27F_h$								
$F280_h$-$F287_h$	Key 08							
$F288_h$-$F28F_h$								
$F290_h$-$F297_h$	Key 09							
$F298_h$-$F29F_h$								
$F2A0_h$-$F2A7_h$	Key 0A							
$F2A8_h$-$F2AF_h$								
$F2B0_h$-$F2B7_h$	Key 0B							
$F2B8_h$-$F2BF_h$								
$F2C0_h$-$F2C7_h$	Key 0C							
$F2C8_h$-$F2CF_h$								
$F2D0_h$-$F2D7_h$	Key 0D							
$F2D8_h$-$F2DF_h$								
$F2E0_h$-$F2E7_h$	Key 0E							
$F2E8_h$-$F2EF_h$								
$F2F0_h$-$F2F7_h$	Key 0F							
$F2F8_h$-$F2FF_h$								

The VolatileKey (KeyID = 0xFF) does not exist in EEPROM. It is a temporary key that resides in the internal SRAM memory. The internal SRAM cannot be accessed directly. See section 4.3 for VolatileKey information.

Prior to locking the key memory, it can be written with either encrypted or cleartext data. Encrypted writes are performed using the EncWrite command (see Section 7.11). Cleartext writes are performed using the standard SPI or I^2C write commands (see Section 5.3). The key memory can never be read with the BlockRead command, or the EncRead command, or with standard I^2C or SPI read commands.

Appendix G. Understanding the STATUS Register

The device status register is used for handshaking between the host microcontroller and the ATAES132. The host microcontroller is expected to read the STATUS register before sending a command or reading a response.

G.1. Device Status Register (STATUS) Definition

Address 0xFFF0 contains the read-only device status register which indicates the current status of the ATAES132 device. The SPI read status register command can be used to read the STATUS register as described in Section K.3.6.

This register can also be read with the standard I^2C or SPI read memory commands. Reading the STATUS register does not increment the memory read address, so a host microcontroller can easily monitor the ATAES132 device status by repeatedly reading the STATUS register.

Table G-22. Device status register definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EERR	RRDY	Reserved	CRCE	Reserved	WAKEb	WEN	WIP

Table G-23. Device status register bit definitions(1)(2)

Bit	Definition
Bit 0 (WIP)	“0b” indicates the device is ready, waiting for a command “1b” indicates a write cycle or a cryptographic operation is in progress
Bit 1 (WEN)	"0b" indicates the device is not SPI write enabled, or is in I^2C interface mode “1b” indicates the device is SPI write enabled
Bit 2 (WAKEb)	"0b" indicates the device is not in the Sleep or Standby power state “1b” indicates the device is in the Sleep or Standby power state
Bit 3 (Reserved)	Always "0b". This bit is reserved for future use. (1)
Bit 4 (CRCE)	"0b" indicates the most recent command block contained a correct checksum (CRC) “1b” indicates the most recent command block contained an error
Bit 5 (Reserved)	Always "0b". This bit is reserved for future use. (1)
Bit 6 (RRDY)	"0b" indicates the response memory buffer is empty “1b” indicates the response memory buffer is ready to read
Bit 7 (EERR)	"0b" indicates the most recent command did not generate an error during execution “1b” indicates the most recent command generated an execution error

Notes: 1. When the SPI RDSR command is used to read the STATUS register during an EEPROM write or during execution of any ATAES132 command, then status bits 0 - 7 are "1b"s (See Section K.3.6). When the STATUS register is read from address 0xFFF0 under the same circumstances, the reserved bits will read as 0b.

2. STATUS register bits 0 - 7 are "1b"s during wakeup. During the first phase of wakeup ($t_{PU.STATUS}$), the SO pin is tri-stated and any attempt to read it will be system-dependent. See Appendix L for additional information.

The device status register can always be read when the ATAES132 is configured for SPI interface mode, even when the ATAES132 is processing a command or writing the EEPROM. When the ATAES132 is configured for I^2C interface mode, the random read command can only be used to read the STATUS register when the device address is ACKed.

If the ATAES132 is in the sleep or standby power state, reading the STATUS register forces the ATAES132 to wakeup – the STATUS register is 0xFF until the wakeup process is complete.

G.1.2. WIP Status Bit [0]

The WIP status bit is used to indicate the device is busy or a "Write is In Progress". If WIP = 0b, then the ATAES132 is in the active state and is waiting to receive a command. If WIP = 1b, then ATAES132 is in the active state and is performing an EEPROM write, or is processing an ATAES132 command.

G.1.3. WEN Status Bit [1]

If the ATAES132 is configured in I^2C interface mode, then the WEN status bit is always 0b (See Appendix J for I^2C information).

If the ATAES132 is configured in SPI interface mode, then the WEN status bit is 0b after the device initially powers up or exits the sleep state (See Appendix K for SPI interface information). When WEN = 0b the user memory is write protected, and any attempt to write the user memory using the SPI WRITE command will fail. The host must send an SPI WREN command to the device to set WEN = 1b prior to each SPI WRITE command.

If the ATAES132 is configured in SPI interface mode, then the WEN status bit will return to 0b when any write instruction is received. The WEN status bit can be forced to 0b by sending a SPI WRDI command (See Section K.3.3), or by sending a RESET command (See Section 7.23), or by putting the device in the sleep state. Powering the device off will reset the WEN bit to 0b. The SPI READ command and SPI RDSR command do not affect the state of the WEN bit.

It is not necessary to set WEN = 1b prior to writing to the command memory buffer or the IO address reset register (See Appendix D). Writing the command memory buffer or the IO address reset register forces WEN to 0b.

G.1.4. WAKEb Status Bit [2]

The WAKEb status bit is 0b when the ATAES132 has completed a power up sequence and is in the ACTIVE state. WAKEb is 1b when the ATAES132 is in the sleep or standby state, or is in the process of waking up.

Note: Reading the STATUS register will cause a device in the sleep state or standby state to wakeup. (See Appendix L for power state and power management information.)

G.1.5. CRCE Status Bit [4]

The CRCE status bit is set to 1b if a block is received with a short count, bad checksum, or if the block causes a buffer overrun. If only the checksum (CRC) was incorrect, then the block may be resent without change. If the command memory buffer contains a partial command block, then the CRCE status bit is 1b and all other status bits are 0b. This indicates that the correct checksum has not yet been received.

The EERR bit will remain 0b when a checksum error occurs and the response memory buffer will remain empty because these errors do not result in a ReturnCode being generated. If a buffer overrun occurs, then the CRCE and EERR bits will be set to 1b.

G.1.6. RRDY Status Bit [6]

The RRDY status bit is 0b when the response memory buffer is empty. If RRDY = 1b, then the response memory buffer contains a response block or a ReturnCode resulting from the most recent command or command block received (See Section D.3 for response memory buffer information).

G.1.7. EERR Status Bit [7]

If the command is processed without error, the EERR bit is set to 0b. When any error other than a checksum error occurs, the EERR status bit is set to 1b to indicate an error. The host can read the error code (ReturnCode) from the response memory buffer (address 0xFE00) using the READ command if the RRDY status bit is 1b.

Reading the STATUS register does not reset the status register bits or alter the response memory buffer contents. Reading the response memory buffer does not alter the contents of the response memory buffer or the STATUS register. Reading beyond the end of the response memory buffer will not cause the STATUS register bits to change.

The EERR status bit will be set to 1b if an SPI or I^2C read is attempted using an invalid address, or an address pointing to a protected portion of the memory. EERR will also be set to 1b if an SPI or I^2C read begins at an authorized address, but continues into protected memory. In both of these cases, the RRDY status bit is 0b and the response memory buffer will remain empty because these errors do not generate a ReturnCode. Reading beyond the end of user zone F will not cause the EERR bit to be set to 1b.

Note: If an SPI or I^2C read begins at an authorized address and continues into protected memory, the EERR bit will be set to 1b.

G.1.8. Reserved Status Bits [3, 5]

The reserved status bits are always 0b when the ATAES132 is capable of accepting a command. The reserved status bits are 1b during power up and during wakeup from the sleep state or the standby state.

G.2. STATUS Register Behavior in the I^2C Interface Mode

The following sections describe the device behavior and expected STATUS register values during commonly performed operations. In the I^2C Interface mode, the ATAES132 will always NAK instructions containing a non-matching I^2C device address. The ATAES132 will ACK instructions with a matching I^2C device address if the device is capable of accepting an instruction. See Appendix J for the I^2C interface specifications.

When the ATAES132 is busy or unable to respond for any reason, it will NAK a matching I^2C device address. The ACK/NAK response to the I^2C device address operates similar to the way the WIP Status bit changes value in SPI Interface mode.

G.2.1. Power Up

The ATAES132 will NAK all instructions received during power up to indicate that it is not ready to accept a command from the host. When the power up process is complete (after time $t_{PU.RDY}$), then the ATAES132 will enter the state specified by ChipConfig Register bits 6 and 7 – the active state, the standby state, or the sleep state (see Section L.2.1). In I^2C Interface mode, it is impossible to read the STATUS register until the completion of power up.

Upon completion of power up, the command memory buffer is empty, the response memory buffer are all 0xFF's, and the ChipState = 0xFFFF. The default EEPROM address is set to 0x0000, and the command and response memory buffer pointers are set to the base address of the buffers. If the device is configured to enter the active state at power up, then the STATUS will be 0x00 as shown in Table G-3.

Table G-24. After power up to the active state, the STATUS register contains:

Bit	Definition
Bit 0 (WIP)	“0b” indicates the device is ready, waiting for a command
Bit 1 (WEN)	"0b" indicates the device is in I^2C interface mode
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state
Bit 3 (Reserved)	Always "0b"
Bit 4 (CRCE)	"0b" indicates no checksum error
Bit 5 (Reserved)	Always "0b"
Bit 6 (RRDY)	"0b" indicates the response memory buffer is empty
Bit 7 (EERR)	"0b" indicates no errors during execution

If the device is configured to enter the sleep state, then the ATAES132 will NAK any attempt to read the STATUS at the completion of power up as described in Section G.2.2. If the device is configured to enter the standby state, then the ATAES132 will NAK any attempt to read the STATUS at the completion of power up as described in Section G.2.3 – ChipState will remain 0xFFFF in the standby state.

Note: ACK polling or attempting to read the STATUS register after power up is completed will cause the device to WakeUp.

G.2.2. WakeUp from Sleep

The ATAES132 will NAK all instructions received during WakeUp from the sleep power state to indicate that it is not ready to accept a command from the host. When the WakeUp process is complete (after time $t_{WupSL.RDY}$), then the ATAES132 will enter the active state. In I^2C interface mode, it is impossible to read the STATUS register until the completion of WakeUp.

Upon completion of WakeUp from sleep, the command memory buffer is empty, the response memory buffer are all 0xFF's, and the ChipState = 0x5555. The default EEPROM address is set to 0x0000, and the command and response memory buffer pointers are set to the base address of the buffers. Upon completion of WakeUp, the STATUS will be 0x00 as shown in Table G-3.

G.2.3. WakeUp from Standby

The ATAES132 will NAK all instructions received during WakeUp from the standby power state to indicate that it is not ready to accept a command from the host. When the WakeUp process is complete (after time $t_{WupSB.RDY}$), then ATAES132 will enter the active state. In I^2C Interface mode, it is impossible to read the STATUS register until the completion of WakeUp.

Upon completion of WakeUp from standby the command memory buffer is empty, and the response memory buffer are all 0xFF's. The ChipState will be the value it had prior to entering the standby state. Upon completion of WakeUp the STATUS will be 0x00 as shown in Table G-3.

G.2.4. Read STATUS Register

To read the STATUS register the host sends a Random Read memory instruction (RREAD) with a starting address of 0xFFF0 when ATAES132 ACKs the I^2C device address. Reading the STATUS register does not increment the read address, so the host can poll the STATUS by reading any number of bytes beginning with address 0xFFF0.

Reading the STATUS register does not change the command memory buffer contents or the response memory buffer contents. Reading the STATUS register does not change the command memory buffer pointer or the response memory buffer pointer. Reading the STATUS register does not change the STATUS register.

G.2.5. Read User Memory

The ATAES132 instructions for directly reading the user memory are identical to the standard Atmel Serial EEPROM instructions. The host can send a read memory instruction (READ, RREAD, SREAD) whenever the ATAES132 ACKs the I^2C device address. If the address being read is valid and access is not prohibited, then the contents of that byte will be returned to the host. If the address is invalid, or access is prohibited for any reason, then 0xFF will be returned to the host in place of the prohibited byte.

A read operation begins with an I^2C start condition and ends with an I^2C NAK by the host. If one or more bytes are accessed during the read operation at an invalid or protected address, then the EERR bit will be set to 1b (see Table G-4). If all bytes accessed by the read operation are valid and the host satisfied the required access conditions, then the EERR bit will be set to 0b. The contents of the command memory buffer and the response memory buffer will remain unchanged.

Note: If an I^2C read begins at an authorized address and continues into protected memory, the EERR bit will be set to 1b.

Table G-25. After an I^2C read memory operation, the STATUS register contains:

Bit	Definition
Bit 0 (WIP)	“0b” indicates the device is ready, waiting for a command
Bit 1 (WEN)	"0b" indicates the device is in I^2C interface mode
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state
Bit 3 (Reserved)	Always "0b"
Bit 4 (CRCE)	"0b" indicates no checksum error
Bit 5 (Reserved)	Always "0b"
Bit 6 (RRDY)	"0b" indicates the response memory buffer is unchanged(1)
Bit 7 (EERR)	"0b" indicates no errors during execution of the read operation "1b" indicates 0xFF was returned in place of one or more invalid or prohibited bytes read

Note: 1. A read memory operation does not change the contents of the response memory buffer. The EERR status bit is used to indicate success, or to indicate an error. No ReturnCode is generated by a memory read error.

G.2.6. Write User Memory

The ATAES132 instructions for directly writing the user memory are identical to the standard Atmel Serial EEPROM. The host can send a write memory instruction (BWRITE, PWRITE) whenever the ATAES132 ACKs the I^2C device address. If the address being written is valid, access requirements have been satisfied, and no page boundaries are crossed, then the data provided by the host will be written after the host generates an I^2C stop condition. If the address is invalid, or access is prohibited for any reason, then the ATAES132 will discard the data and no EEPROM write will occur.

A memory write operation begins with an I^2C start condition and ends with a I^2C stop condition by the host. If the host does not provide an I^2C stop condition, then no write will occur, no ReturnCode will be generated, and the STATUS is 0x00 to indicate the ATAES132 is waiting for a command.

If the host provides the required I^2C stop condition, then the ATAES132 will NAK the I^2C device address during the EEPROM write operation. When the write operation is complete, then ATAES132 will ACK the I^2C device address.

Upon completion of a memory write operation, the command memory buffer is empty, and the response memory buffer contains a ReturnCode. The command and the response memory buffer pointers are set to the base address of the buffers. The STATUS will be as shown in Table G-5.

Table G-26. After an I^2C write memory operation, the STATUS register contains:

Bit	Definition
Bit 0 (WIP)	“0b” indicates the device is ready, waiting for a command
Bit 1 (WEN)	"0b" indicates the device is in I^2C interface mode
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state
Bit 3 (Reserved)	Always "0b"
Bit 4 (CRCE)	"0b" indicates no checksum error
Bit 5 (Reserved)	Always "0b"
Bit 6 (RRDY)	"1b" indicates the response memory buffer contains a response block
Bit 7 (EERR)	"0b" indicates no errors during execution of the write operation "1b" indicates the write operation generated an error; see the ReturnCode for the cause

G.2.7. Write Command Memory Buffer

To write the command memory buffer, the host sends a write memory instruction (BWRITE, PWRITE) with a starting address of 0xFE00 when the ATAES132 ACKs the I^2C device address. As each byte is written, the command memory buffer pointer increments by one.

A command block begins with the COUNT byte and ends with the two byte Checksum (see Section 6.1). If the entire command block is not received, then the device will not attempt to process the command and will not generate a response block. The STATUS register will have the CRCE bit = 1b until the entire command block is received (as shown in Table G-6).

Table G-27. If the command memory buffer contains a partial command block, the STATUS Register contains:

Bit	Definition
Bit 0 (WIP)	"0b" indicates the device is ready, waiting for a command
Bit 1 (WEN)	"0b" indicates the device is in I^2C interface mode
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state
Bit 3 (Reserved)	Always "0b"
Bit 4 (CRCE)	"1b" indicates a checksum error (The checksum has not yet been received)
Bit 5 (Reserved)	Always "0b"
Bit 6 (RRDY)	"0b" indicates the response memory buffer is unchanged
Bit 7 (EERR)	"0b" indicates no errors during execution of the command block (It was not executed yet)

If the host provides a complete command block, then the ATAES132 will NAK the I^2C device address during command processing. When command processing is complete, then the ATAES132 will ACK the I^2C device address.

If the command block contains a bad checksum, a short COUNT, or the block causes a buffer overrun, then the CRCE bit of the STATUS register will be set to 1b as shown in Table G-7. The response memory buffer will be unchanged because no ReturnCode is generated by these error conditions. The EERR Status bit is 1b if a buffer overrun error occurs. The EERR bit is 0b if a bad checksum or short count error occurs.

If the Command Block contains a good checksum, then ATAES132 will process the command and load the response in the Response Memory Buffer. Upon completion of command processing the RRDY bit of the STATUS register is set to 1b as shown in 0.

Table G-28. After an I^2C write command memory buffer resulting in CRCE = 1b, the STATUS register contains:

Bit	Definition
Bit 0 (WIP)	"0b" indicates the device is ready, waiting for a command
Bit 1 (WEN)	"0b" indicates the device is in I^2C interface mode
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state
Bit 3 (Reserved)	Always "0b"
Bit 4 (CRCE)	"1b" indicates a checksum error, short count, or command buffer overrun error
Bit 5 (Reserved)	Always "0b"
Bit 6 (RRDY)	"0b" indicates the response memory buffer is unchanged
Bit 7 (EERR)	"0b" indicates no errors during execution of the command block (It was not executed) "1b" indicates a command buffer overrun error

Table G-29. After an I^2C write command memory buffer resulting in CRCE = 0b , the STATUS register contains:

Bit	Definition
Bit 0 (WIP)	“0b” indicates the device is ready, waiting for a command
Bit 1 (WEN)	"0b" indicates the device is in I^2C interface mode
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state
Bit 3 (Reserved)	Always "0b"
Bit 4 (CRCE)	"0b" indicates no checksum error
Bit 5 (Reserved)	Always "0b"
Bit 6 (RRDY)	"1b" indicates the response memory buffer contains a response block
Bit 7 (EERR)	"0b" indicates no errors during execution of the command block "1b" indicates the command block generated an error; see the ReturnCode for the cause

Writing the command memory buffer resets the response memory buffer pointer to the base address. Writing the command memory buffer does not change the response memory buffer contents until the entire command block is received and processed.

The host can re-write the contents of the command memory buffer by resetting the buffer pointer (by writing the IO address reset register) and sending a write memory instruction (BWRITE, PWRITE) with a starting address of 0xFE00.

Note: If the host must write the command memory buffer with more bytes than is required to send the command block due to hardware limitations, then the host should transmit 0xFF bytes after the checksum. The extra bytes will be discarded by the ATAES132 and will not result in a buffer overrun, or any other error.

G.2.8. Read Response Memory Buffer

To read the Response Memory Buffer the host sends a Random Read memory instruction (RREAD) with a starting address of 0xFE00 when ATAES132 ACKs the I^2C device address. The host can read any number of bytes from the Response Memory Buffer without causing an error. As each byte is read, the Response Memory Buffer pointer increments by 1. If the host reads beyond the end of the Response Block, then 0xFF will be returned for any byte after the Checksum.

Reading the Response Memory Buffer does not change the Command Memory Buffer contents or the Response Memory Buffer contents. Reading the Response Memory Buffer resets the Command Memory Buffer pointer to the base address. Reading the Response Memory Buffer does not change the STATUS register.

The host can re-read the contents of the Response Memory Buffer by resetting the buffer pointer (by writing the IO Address Reset register) and sending a Random Read memory instruction (RREAD) with a starting address of 0xFE00.

G.2.9. Write IO Address Reset Register

To reset the pointer for the command memory buffer and the pointer for the response memory buffer, the host sends a write memory instruction (BWRITE, or PWRITE) with a starting address of 0xFFE0. The IO address reset register can be written with 1 to 32 bytes of data without generating an error – the data bytes will be ignored. The command and the response memory buffer pointers are set to the base address of the buffers. The command memory buffer is empty, and the response memory buffer contents are unchanged. Writing the IO address reset register changes the CRCE status bit to 0b – all of the other STATUS bits are unchanged.

G.3. STATUS Register Behavior in the SPI Interface Mode

The following sections describe the device behavior and expected STATUS register values during commonly performed operations. See Appendix K for the SPI interface specifications. In SPI Interface mode, there are two ways to read the STATUS register:

- Using the SPI RDSR command, or
- Reading STATUS from address 0xFFF0

When the ATAES132 is busy or unable to respond for any reason, the WIP status bit is 1b......

G.3.1. Power Up

The ATAES132 will .... during power up to indicate that it is not ready to accept a command from the host. When the power up process is complete (after time $t_{PU.RDY}$), then the ATAES132 will enter the state specified by ChipConfig Register bits 6 and 7 (see Section L.2.1): the active state, the standby state, or the sleep state.

Upon completion of power up, the command memory buffer is empty, the response memory buffer are all 0xFF's, and the ChipState = 0xFFFF. The default EEPROM address is set to 0x0000, and the command and response memory buffer pointers are set to the base address of the buffers. If the device is configured to enter the active state, then the STATUS will be 0x00 as shown in Table G-9.

Table G-30. After power up to the active state, the STATUS register contains:

Bit	Definition
Bit 0 (WIP)	“0b” indicates the device is ready, waiting for a command
Bit 1 (WEN)	"0b" indicates the device is not write enabled
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state
Bit 3 (Reserved)	Always "0b"
Bit 4 (CRCE)	"0b" indicates no checksum error
Bit 5 (Reserved)	Always "0b"
Bit 6 (RRDY)	"0b" indicates the response memory buffer is empty
Bit 7 (EERR)	"0b" indicates no errors during execution

If the device is configured to enter the sleep state, then the STATUS will be 0xFF at the completion of power. If the device is configured to enter the standby state, then the STATUS will be 0xFF?? at the completion of power up – ChipState will remain 0xFFFF in the standby state.

Note: Reading the STATUS register after power up is completed will cause the device to WakeUp

Appendix H. Understanding the Non-Reversible Monotonic Counters

Each monotonic counter can increment up to a value of 2,097,134 using the count command, after which, can be no longer changed. Counters attached to keys are also incremented each time the key is used – when the counter reaches its limit the key is disabled. The value in the counter can never be reset or lowered. The counters include a power interruption protection feature to prevent corruption of the count value if power is removed during the increment operation.

On shipment from Atmel, the counter registers are initialized to their lowest value. The initial value of each counter may be written to a different value at personalization, prior to locking the configuration.

H.1.1. Monotonic Counter Registers

Each monotonic counter register contains two count values to prevent the count value from being corrupted if power is interrupted during a counter increment operation. Each count value is stored as a combination of two count fields:

1. Counter A is stored in LinCountA and BinCountA
3. Counter B is stored in LinCountB and BinCountB

Table H-1 shows the location of the count fields within the counter register in configuration memory.

Table H-31. Partial configuration memory map showing counter register field locations

Address	0_h	1_h	2_h	3_h	4_h	5_h	6_h	7_h
$F100_h$-$F107_h$	Counter 00							
	LinCountA		LinCountB		BinCountB		BinCountA	

The counter registers can always be read from the configuration memory using the BlockRead command. However, the count command is the preferred method of reading the counters. When the counter is read using the count command, the ATAES132 automatically selects the appropriate counter register fields and returns them to the host in the response packet. See Section 7.5 for the counter command.

H.1.2. Reading the Monotonic Counters

The counter command is the recommended method for reading a counter. The counter command returns a 4 byte CountValue field which is formatted as shown in Table H-2. Optionally, the counter command can also return a MAC for cryptographic authentication of the CountValue. The definition of the CountValue field is shown in Table H-3. See Section 7.5 for the counter command.

Table H-32. CountValue field

Byte 0	Byte 1	Byte 2	Byte 3
LinCount	CountFlag	BinCount	

The CountValue contains a linear counter field (LinCount), a binary counter field (BinCount), and the CountFlag field. The CountFlag field indicates if the counter value was read from the Counter A or Counter B EEPROM location. CountFlag also indicates if the eight bit LinCount field is the Most Significant Byte (MSB) or Least Significant Byte (LSB) of the 16 bit LinCount field in EEPROM. If the LSB of LinCount has been returned, then the LinCount field value is one to eight; if the MSB of LinCount has been returned, then the LinCount field value is 9 to 16.

Table H-33. Definition of the CountValue field in the response to the counter command

Byte	Name	Description
0	LinCount	Contains the eight bit linear counter value identified in the CountFlag field
1	CountFlag	If 0x00 then, LinCount contains the LSB of LinCountA BinCount contains the BinCountA value If 0x02 then, LinCount contains the MSB of LinCountA BinCount contains the BinCountA value If 0x04 then, LinCount contains the LSB of LinCountB BinCount contains the BinCountB value If 0x06 then, LinCount contains the MSB of LinCountB BinCount contains the BinCountB value All other values are reserved for future use
2	BinCount (LSB)	Contains the Least Significant Byte of the binary counter identified in the CountFlag field
3	BinCount (MSB)	Contains the Most Significant Byte of the binary counter identified in the CountFlag field

The equivalent decimal value of the counter can be determined using the counter calculator in the Atmel Crypto Evaluation Studio, (ACES) configuration environment. Sample code is also available for decoding the counter value. Both are available as free downloads at www.atmel.com.

H.2. Personalizing the Monontonic Counters

The counter registers are personalized with initial values prior to locking the configuration memory. The standard Serial EEPROM write commands are used to write the configuration memory (see Section 5.3.3). The lock command is used to lock the configuration memory (see Section 7.19).

The initial values can be determined using the counter calculator in the ACES configuration environment. ACES is available for free download at www.atmel.com.

Appendix I. Cryptographic Computations

The ATAES132 implements all its cryptographic commands using AES in CCM mode, with a 128 bit key length per NIST SP800-38C. CCM mode provides both confidentiality and integrity checking with a single key. The integrity MAC includes both the encrypted data and additional authenticate-only data bytes. The particular information authenticated with each command is described within the command descriptions in Section 7.

The device construction ensures that the nonce will be unique for each MAC calculated.

I.1. MacCount

The one byte MacCount is stored in an internal register and is used in the AES-CCM computations. Since MacCount changes, it speeds up computation by eliminating the need to generate a new random nonce for every crypto computation. This register is incremented prior to performing each MAC calculation.

The MacCount register is set to zero when the nonce command is executed and is subsequently incremented prior to any MAC computation being performed. Because of this, the value that will be used for calculating the first MAC of the first command after the nonce command is MAC = 1.

There are two commands (Auth and KeyCompute) which can be configured to both verify an input MAC and calculate an output MAC. When either of these two commands is run in dual authentication mode, MacCount will be incremented twice.

The value of MacCount for a particular MAC calculation is always one greater than that used for the previous MAC calculation. After 255 MAC calculations, the device will invalidate the internal nonce and commands which require a valid nonce will fail. At this point, a new nonce command must be run to generate a new nonce.

The MacCount is set to zero if any of the following events occurs:

- The Nonce command is executed
- A MAC compare operation fails
- The MacCount reaches the maximum count
- A reset event occurs: power up (see Section L.3.2), WakeUp from sleep (see Section L.3.3), the reset command (see Section 7.23), or a security tamper is activated, causing the hardware to reset

If there is a CRC error on the incoming command packet, then MacCount will not be incremented. If the device receives any command that does not involve MAC computation the MacCount will not be incremented.

If a cryptographic commands is received that involves MAC computation, then the MacCount will be incremented regardless of whether or not there is a subsequent success or failure of the command. The MacCount will also be incremented regardless of whether or not the particular instance of the command uses the cryptographic engine. If a command fails due to MAC comparison failure then the nonce is invalidated and the MacCount register is set to zero.

The current value of this register should be known by the system; however, it may also be read out of the device using the Info command at any time (See Section 7.12).

I.2. MacFlag

To prevent spoofing of the MAC value, a flag byte is included in each MAC calculation. MacFlag provides information about the state of the device during the MAC calculation.

Table I-34. Definition of the MacFlag bits

Bit #	Name	Notes
0	Random	If 1b, then the nonce command was run with the RNG enabled and the nonce is guaranteed to be unique If 0b, the nonce value has been sent to the device by the system and may not be unique
1	Input	This bit is 1b for MAC values that are sent to the device as inputs This bit is 0b for MAC values output by the ATAES132
3-7	Zero	All bits must be 0b

I.3. MAC Generation

The following example shows how the integrity MAC is calculated for an authentication operation requiring up to 14 bytes of authenticate-only data. This operation involves three passes through the AES crypto engine, all three using the same key. If there are more than 14 bytes of authenticate-only data, then another pass through the AES crypto engine is required.

There are two passes through the AES crypto engine in CBC mode to create the cleartext MAC. The inputs to the crypto engine for those blocks are labeled B0 and B1, and the outputs are B'0 and B'1 respectively.

B0 is composed of the following 128 bits:

1 byte flag, fixed value of b0111 1001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte length field – always 0x00 00 for authentication only

B1 is the XOR of B'0 with the following 128 bits:

2 byte length field, size of authenticate-only data
14 byte data to be authenticated only

B'1 is the cleartext MAC, which must be encrypted before being sent to the system

There is one additional pass through the AES crypto engine in CTR mode to create the key block that is used to encrypt the MAC. The input to the crypto engine for this block is labeled A0 and the output is A'0. A'0 is the MAC sent to the system as the output parameter of the Auth command.

A0 is composed of the following 128 bits:

1 byte flag, fixed value of b0000 0001
12 byte nonce, as generated by ATAES132 during nonce command
1 byte MacCount, 1 for first MAC generation
2 byte counter field – always 0x00 00 for A0

A'0 is XOR'd with the cleartext MAC (B'1) and sent to the system

Input integrity MACs for Auth, counter, KeyCompute, and lock are also calculated using this procedure. If the input MAC does not match A'0, then the command returns an AUTH error.

I.4. Data Encryption

The following example shows how the encrypted data and integrity MAC are calculated for a 128 bit data read from the device with up to 14 bytes of authenticate-only data. This operation involves five passes through the AES crypto engine, all five using the same key. If there are more than 14 bytes of authenticate-only data and/or more than 128 bits of data being read, then one, two or three more passes through the AES crypto engine are required.

There are three passes through the AES crypto engine in CBC mode to create the cleartext MAC. The inputs to the crypto engine for those blocks are labeled B0, B1 and B2, and the outputs are B’0, B’1 and B’2 respectively.

B0 is composed of the following 128 bits:

1 byte flag, fixed value of b0111 1001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte length field – max 0x00 20 if 256 bits of encrypted data, min 0x00 01 for one byte

B1 is the XOR of B’0 with the following 128 bits:

2 byte length field, size of authenticate-only data
14 byte data to be authenticated only

B2 is the XOR of B’1 with the following 128 bits:

16 bytes cleartext data

B’2 is the cleartext MAC, which must be encrypted before being sent to the system

There are two passes through the AES crypto engine in CTR mode to create the key block that is used to encrypt the data and the MAC. The inputs to the crypto engine for those blocks are labeled A0 and A1, and the outputs are A’0 and A’1 respectively. A’0 and A’1 are the blocks sent to the system as the output parameters of the EncRead and decrypt commands.

A0 is composed of the following 128 bits:

1 byte flag, fixed value of b0000 0001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte counter field – always 0x00 00 for A0

A’0 is XOR’d with the cleartext MAC and sent to the system

A1 is composed of the following 128 bits:

1 byte flag, fixed value of b0000 0001
12 byte nonce, as generated by ATAES132 during nonce command
1 byte MacCount, 1 for first MAC generation
2 byte counter field – always 0x00 01 for A1

A’1 is XOR’d with the cleartext data and sent to the system

This sequence is also used for the Encrypt command, in addition to EncRead.

I.5. Data Decryption

The following example shows how the encrypted data and integrity MAC are calculated for a 128 bit data block write to the device with up to 14 bytes of authenticate-only data. This operation involves five passes through the AES crypto engine, all five using the same key. If there are more than 14 bytes of authenticate-only data and/or more than 128 bits of data being written, then 1, 2 or 3 more passes through the AES crypto engine are required.

There are two passes through the AES crypto engine in CTR mode to create the key block that is used to decrypt the data and the MAC. The inputs to the crypto engine for those blocks are labeled A0 and A1, and the outputs are A’0 and A’1 respectively. A’0 & A’1 are the blocks sent to the system as the output parameters of the EncRead and decrypt commands.

A0 is composed of the following 128 bits:

1 byte flag, fixed value of b0000 0001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte counter field – always 0x00 00 for A0

A’0 is XOR’d with the encrypted input MAC and stored in the internal SRAM as the MAC T

A1 is composed of the following 128 bits:

1 byte flag, fixed value of b0000 0001
12 byte nonce, as generated by ATAES132 during nonce command
1 byte MacCount, 1 for first MAC generation
2 byte counter field – always 0x00 01 for A1

A’1 is XOR’d with the encrypted input data and stored in the internal SRAM as the message M

There are three passes through the AES crypto engine in CBC mode to create the expected MAC value. The inputs to the crypto engine for those blocks are labeled B0, B1 and B2, and the outputs are B’0, B’1 and B’2 respectively.

B0 is composed of the following 128 bits:

1 byte flag, fixed value of b0111 1001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte length field – max 0x00 20 if 256 bits of encrypted data, min 0x00 01 for one byte

B1 is the XOR of B’0 with the following 128 bits:

2 byte length field, size of authenticate-only data
14 byte data to be authenticated only

B2 is the XOR of B’1 with the following 128 bits:

16 bytes of cleartext message M

B’2 is the cleartext MAC. If this matches the stored T value, then the write to memory proceeds. If there is no match, the device returns an error flag and does not modify memory.

This sequence is also used for the Decrypt and KeyLoad commands, in addition to EncWrite.

I.6. Auth Command MAC

The MACs are calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
11 bytes	FirstBlock field containing:	
	1 byte	Auth Opcode (0x03)
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
	4 bytes	0x00
1 byte	Padding of value 0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculations.

4 bytes	Usage counter value, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.7. AuthCheck Command – Auth MAC

The Auth command MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
11 bytes	FirstBlock field containing:	
	1 byte	Auth Opcode (0x03)
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
	4 bytes	0x00
1 byte	Padding of value 0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculations.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.8. AuthCheck Command – Counter MAC

The counter command MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
11 bytes	FirstBlock field containing:	
	1 byte	Counter Opcode (0x0A)
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
	4 bytes	CountValue, the output parameter
1 byte	Padding of value 0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculations.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.9. AuthCompute Command – Auth MAC

The Auth command MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
11 bytes	FirstBlock field containing:	
	1 byte	Auth Opcode (0x03)
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
	4 bytes	0x00
1 byte	Padding of value 0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculations.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.10. AuthCompute Command – Counter MAC

The counter command MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
11 bytes	FirstBlock field containing:	
	1 byte	Counter Opcode (0x0A)
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
	4 bytes	0x00
1 byte	Padding of value 0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculations.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.11. BlockRead Command

The BlockRead command does not perform a cryptographic operation and does not use or generate a MAC.

I.12. Counter Command MAC

The InMAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	Counter Opcode (0x0A)	
11 bytes	FirstBlock field containing	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
	4 bytes	CountValue
	1 byte	0x00

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the InMAC calculation.

4 bytes	Usage counter value for MAC generation key, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

The OutMAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID
1 byte	Counter Opcode (0x0A)
1 byte	Mode
2 bytes	Param1
2 bytes	Param2
1 byte	MacFlag
4 bytes	CountValue, the output parameter
1 byte	0x00

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the OutMAC calculation.

4 bytes	Usage counter value for MAC generation key, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.13. Crunch Command

The Crunch command does not perform a cryptographic operation and does not use or generate a MAC.

I.14. DecRead Command

The MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	EncRead Opcode (0x04)	
6 bytes	FirstBlock field containing:	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.15. Decrypt Command MAC

In normal decryption mode, the InMAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID
1 byte	Decrypt Opcode (0x07)
1 byte	Mode
2 bytes	Param1
2 bytes	Param2
1 byte	MacFlag
5 bytes	0x00

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	Usage counter value, or 0x00 if not selected or if KeyID is VolatileKey
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.15.1. Client Decrypt MAC

In Client Decryption mode, the InMAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID
1 byte	Encrypt Opcode (0x06)
1 byte	Mode
2 bytes	Upper Byte = 0x00, Lower Byte = EKeyID
2 bytes	Upper Byte = 0x00, Lower Byte = Lower Byte of Param2
1 byte	MacFlag = 0x01
5 bytes	0x00

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	0x00 if Usage counter value is not selected, or 0x00 if KeyID is VolatileKey
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

The device MacCount will be changed to the EMacCount value when a decrypt command is received with the client decryption mode is selected. The EMacCount will be used when decrypting the data and the MacCount will be incremented by the decrypt operation. (After processing the command, the device MacCount will equal EMacCount plus one)

I.16. EncRead Command MAC

The OutMAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	EncRead Opcode (0x05)	
6 bytes	FirstBlock field containing	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	Usage counter value, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.17. EncRead Command Configuration Memory Signature MAC

The following example shows how the integrity MAC is calculated for a 512 byte (32 block) certification of the data from the configuration memory. This operation involves multiple passes through the AES crypto engine, all using the same key, KeyID 00. If the mode parameter indicates that there is an additional block of authenticate-only data, then another pass through the AES crypto engine is required.

There are 35 passes through the AES crypto engine in CBC mode to create the clear text MAC. The inputs to the crypto engine for those blocks are labeled B0, B1 and B2 …, and the outputs are B'0, B'1 and B'2 … respectively.

B0 is composed of the following 128 bits:

1 byte flag, fixed value of b0111 1001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte length field – always 0x00 00

B1 is the XOR of B'0 with the following 128 bits:

2 byte length field, value of 528 or 544
14 byte ManufacturingID, Opcode, etc.

B2 is the XOR of B'1 with the following 128 bits:

16 bytes counter+serial+small, if mode indicates, otherwise this block does not exist

B3 is the XOR of B'2 with the following 128 bits:

First 16 bytes of config – in the clear

B4 is the XOR of B'3 with the following 128 bits:

Second 16 bytes of config – in the clear
… and so on …

B'34 is the clear text MAC, which must be encrypted before being sent to the system

There is one pass through the AES crypto engine in CTR mode to encrypt the MAC.

A0 is composed of the following 128 bits:

1 byte flag, fixed value of b0000 0001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte counter field – always 0x00 00

A'0 is XOR'd with the clear text MAC and sent to the system

I.18. EncRead Command Key Memory Signature MAC

The following example shows how the integrity MAC is calculated for a 256 byte (16 block) certification of the data from the key memory. This operation involves multiple passes through the AES crypto engine, all using the same key, KeyID 00. If the mode parameter indicates that there is an additional block of authenticate-only data, then another pass through the AES crypto engine is required.

There are 19 passes through the AES crypto engine in CBC mode to create the clear text MAC. The inputs to the crypto engine for those blocks are labeled B0, B1 and B2 …, and the outputs are B’0, B’1 and B’2 … respectively.

B0 is composed of the following 128 bits:

1 byte flag, fixed value of b0111 1001
12 byte nonce, as generated by the Nonce command
1 byte MacCount, 1 for first MAC generation
2 byte length field – always 0x00 00

B1 is the XOR of B’0 with the following 128 bits:

2 byte length field, value of 272 or 288
14 byte ManufacturingID, Opcode, etc.

B2 is the XOR of B’1 with the following 128 bits:

16 bytes counter+serial+small, if mode indicates, otherwise this block does not exist

B3 is the XOR of B’2 with the following 128 bits:

First 16 bytes of config – in the clear

B4 is the XOR of B’3 with the following 128 bits:

Second 16 bytes of config – in the clear
… and so on …

B’18 is the clear text MAC, which must be encrypted before being sent to the system

There is one pass through the AES crypto engine in CTR mode to encrypt the MAC.

A0 is composed of the following 128 bits:

1 byte flag, fixed value of b0000 0001
12 byte nonce, as generated by the nonce command
1 byte MacCount, 1 for first MAC generation
2 byte counter field – always 0x00 00

A’0 is XOR’d with the clear text MAC and sent to the system

I.19. Encrypt Command MAC

The OutMAC is calculated using the following 14 bytes in the default authenticate-only block

2 bytes	ManufacturingID
1 byte	Encrypt Opcode (0x06)
1 byte	Mode
2 bytes	Param1
2 bytes	Param2
1 byte	MacFlag
5 bytes	0x00

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	Usage counter value, or 0x00 if not selected or if KeyID is VolatileKey
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.20. EncWrite Command MAC

The InMAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	EncWrite Opcode (0x05)	
6 bytes	FirstBlock field containing:	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	Usage counter value, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.21. Info Command

The Info command does not perform a cryptographic operation and does not use or generate a MAC.

I.22. KeyCompute Command MAC

The input and output MACs are both calculated using the parent key.

Both MACs are calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	KeyCompute Opcode (0x08)	
6 bytes	FirstBlock field containing	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculations.

4 bytes	Usage counter value, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.23. KeyExport Command – KeyImport MAC

The MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID
1 byte	KeyExport Opcode (0x18)
1 byte	Mode
2 bytes	Param1
2 bytes	Param2
1 byte	MacFlag
5 bytes	0x00

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	Usage counter value, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.24. KeyExport Command – KeyLoad MAC

The MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	KeyLoad Opcode (0x09)	
6 bytes	FirstBlock field containing:	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.25. KeyImport Command – KeyCompute MAC

The MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	KeyCompute Opcode (0x08)	
6 bytes	FirstBlock field containing:	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculations.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.26. KeyImport Command – KeyExport MAC

The MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	KeyExport Opcode (0x18)	
6 bytes	FirstBlock field containing:	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

I.27. KeyLoad Command MAC

The InMAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	KeyLoad Opcode (0x09)	
6 bytes	FirstBlock field containing:	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	Usage counter value, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

I.28. KeyTransfer Command

The KeyTransfer command does not perform a cryptographic operation and does not use or generate a MAC.

I.29. Legacy Command

The legacy command executes a single block of the AES engine with no input or output formatting. This is known as ECB mode, and can be used to perform various AES encryption and/or authentication operations. This command does not use the nonce register value in the computation since the entire 16 byte AES engine input value comes from the input packet.

I.30. Lock Command MAC

If required due to the value of the mode parameter and ZoneConfig[UZ].WriteMode, the MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID
1 byte	Lock Opcode (0x0D)
1 byte	Mode
2 bytes	Param1
2 bytes	Param2
1 byte	MacFlag
5 bytes	0x00

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

4 bytes	Usage counter value, or 0x00 if not selected
8 bytes	SerialNum[0:7], or 0x00 if not selected
4 bytes	SmallZone[0:3], or 0x00 if not selected

The AES key used for the MAC calculation is that specified in ZoneConfig[Zone].WriteID

I.31. Nonce Command

If the random nonce option is selected, then the internal random nonce is generated using the following function:

Block A is:

1 byte	Nonce Opcode (0x01)
1 byte	Mode
2 bytes	0x00
12 bytes	Input Seed

Block B is:

2 bytes	ManufacturingID
2 bytes	0x00
12 bytes	Internally generated random number

AES is executed in ECB mode with an input value of A and a key of B. The output of the AES crypto engine is XOR'd with Block A and the first 12 bytes of the result are stored in the internal nonce register.

If the LockConfig register is unlocked (0x55), then the random number generator is latched in the test mode and the nonce command will generate non-random values. If the LockConfig register is locked (0x00), then the random number generator generates random numbers and the Nonce command functions normally.

I.32. NonceCompute Command

The random nonce is generated using the following function:

Block A is:

1 byte	Nonce Opcode (0x01)
1 byte	Mode
2 bytes	0x00
12 bytes	Nonce Register

Block B is:

2 bytes	ManufacturingID
2 bytes	0x00
12 bytes	Random Seed

AES is executed in ECB mode with an input value of A and a key of B. The output of the AES crypto engine is XOR'd with Block A and the first 12 bytes of the result are stored in the internal nonce register.

I.33. Random Command

Generates a random number using the internal high quality random number generator and the random number generation procedure recommended by NIST in SP800-90 (see Appendix A).

I.34. Reset Command

The reset command does not perform a cryptographic operation and does not use or generate a MAC.

I.35. Sleep Command

The sleep command does not perform a cryptographic operation and does not use or generate a MAC.

I.36. TempSense Command

The TempSense command does not perform a cryptographic operation and does not use or generate a MAC.

I.37. WriteCompute Command

The MAC is calculated using the following 14 bytes in the default authenticate-only block:

2 bytes	ManufacturingID	
1 byte	EncWrite Opcode (0x05)	
6 bytes	FirstBlock field containing:	
	1 byte	Mode
	2 bytes	Param1
	2 bytes	Param2
	1 byte	MacFlag
5 bytes	0x00	

If *any* of the optional authenticate fields are selected in the mode parameter, then a second authenticate-only block is included in the MAC calculation.

16 bytes	SecondBlock field containing:	
	4 bytes	Usage counter value, or 0x00 if not selected
	8 bytes	SerialNum[0:7], or 0x00 if not selected
	4 bytes	SmallZone[0:3], or 0x00 if not selected

Appendix J. I^{2}C Interface

The ATAES132 two-wire serial interface is designed to interface directly to microcontrollers with I^{2}C interface ports. The serial interface and cleartext read/write operations operate similar to the Atmel I^{2}C Serial EEPROM.

The host sends ATAES132 extended commands to the device by writing the command packet to the command memory buffer at address 0xFE00. The ATAES132 processes the command packet and places the response in the response memory buffer. The host retrieves the response by reading the response packet from address 0xFE00.

See Section G.2 for additional information regarding the ATAES132 behavior in I^{2}C interface mode. See Section J.6 for I^{2}C compatibility information.

J.1. I^{2}C Serial Interface Description

When the ATAES132 is configured in I^{2}C serial communication mode, the serial interface operates as an I^{2}C compatible standard-mode I^{2}C slave device as described in this appendix. I^{2}C is a synchronous serial interface protocol that is a defacto industry standard which is not formally documented or controlled. Multiple I^{2}C devices can share the data bus; however, each I^{2}C slave must have a unique I^{2}C device address to prevent bus contention. SCK clock frequencies up to 1MHz are supported by the ATAES132.

The serial interface communication mode is selected by programming the I^{2}CAddr register in the configuration memory as described in Section E.2.15. The I^{2}C device address is also located in the I^{2}CAddr register. The ATAES132 will only respond to I^{2}C instructions which have a matching I^{2}C device address.

J.1.1. I^{2}C Master

The I^{2}C master device generates the serial clock and sends instructions to the I^{2}C slave devices. In this specification, the I^{2}C master is usually referred to as the Host or the Host microcontroller.

J.1.2. I^{2}C Slave

I^{2}C slave devices receive the serial clock as an input, and receive instructions from the I^{2}C master. I^{2}C slaves can never generate traffic on the I^{2}C interface, slaves can only respond to instructions provided by the I^{2}C master. The ATAES132 always operates as a slave. In this specification the slave is usually referred to as the client or the device.

J.1.3. I^{2}C Device Address

Each ATAES132 has a 7 bit I^{2}C device address (stored in the I^{2}CAddr register, as described in Section E.2.15) which is used by the Host to direct commands to a specific device on the I^{2}C interface. I^{2}C devices will only respond to instructions with a matching I^{2}C device address. When the ATAES132 is in the standby state or sleep state, a matching I^{2}C device address will cause the device to wakeup (See Appendix L for power management specifications).

The LSB of the I^{2}C device address byte is the read/write operation select bit – a "read" operation is initiated if the R/W bit is high and a "write" operation is initiated if the R/W bit is low.

J.1.4. Relationship of Clock to Data

Data on the SDA pin may change only during SCK low time periods. Data changes during SCK high periods indicate an I^{2}C START or I^{2}C STOP condition. The SDA pin is pulled high by an external resistor when no devices are driving the I^{2}C data bus. The timing requirements for the clock and data signals are illustrated in Section J.7.

J.1.5. I^{2}C START Condition

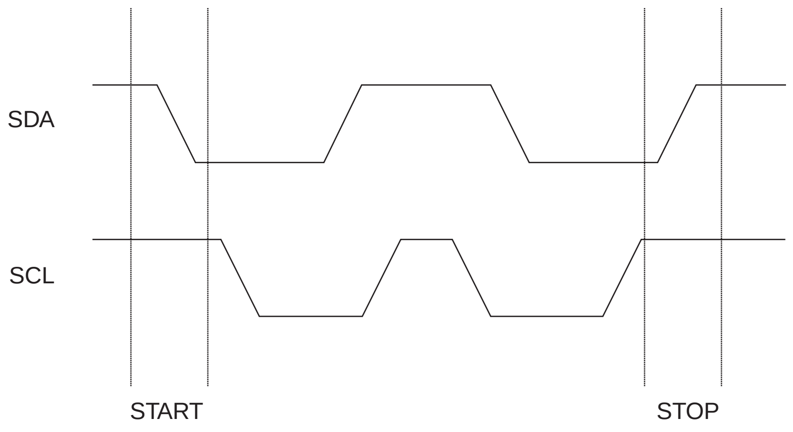
A high-to-low transition of SDA with SCK high is an I^{2}C START condition. An I^{2}C START condition must precede the I^{2}C device address for any instruction. I^{2}C START conditions are only generated when the Host is driving the bus – slaves are not allowed to generate an I^{2}C START condition.

The slave will reset its serial interface immediately when an I^{2}C START condition is received. An I^{2}C START condition cannot be followed immediately with an I^{2}C STOP condition. Figure J-1 illustrates an I^{2}C START condition.

J.1.6. I^{2}C STOP Condition

A low-to-high transition of SDA with SCK high is an I^{2}C STOP condition. I^{2}C STOP conditions are only generated when the Host is driving the bus – slaves are not allowed to generate an I^{2}C STOP condition. Figure J-1 illustrates an I^{2}C STOP condition.

Figure J-1. I^{2}C START condition and I^{2}C STOP condition definitions



J.1.7. I^{2}C ACK

All addresses and data words are serially transmitted to and from the ATAES132 in 8-bit words. The receiving I^{2}C device sends a zero (ACK) during the ninth clock cycle to acknowledge receipt of each byte.

An I^{2}C Host can use acknowledge polling to monitor the progress of an EEPROM write and to determine if the slave is ready to accept a new instruction. See Section J.3.8 for a discussion of ACK polling.

J.1.8. I^{2}C NAK

When the receiving I^{2}C device fails to send a zero during the ninth clock cycle to acknowledge that it has received a byte, then SDA remains high due to the external pull-up resistor. This generates a NO ACK (NAK) signal to the device sending the byte.

J.1.9. Data Format

All instructions and data on the I^{2}C bus must be formatted as eight bit bytes, followed by a ninth bit (ACK or NAK) generated by the receiving device. The MSB is the first bit of each byte transmitted and received.

J.2. Pin Descriptions

When the ATAES132 is configured in the I^2C interface communication mode, the package pins are assigned the functionality described in this section.

Note: The pin numbers listed here are the SOIC, TSSOP, and UDFN package pin numbers.

J.2.1. $\overline{\text{CS}}$ [Pin 1]

In the I^2C communication mode, this pin is not used and should be tied to V_{CC} or V_{SS}. The state of this pin does not affect the functionality or active state power consumption of the ATAES132 when I^2C communication mode is selected.

J.2.2. SO [Pin 2]

In the I^2C communication mode, this pin is not used in the default configuration. It is always in the high impedance state. In this configuration, the pin can be tied to V_{CC} or V_{SS}. The state of this pin does not affect the functionality or active state power consumption of the ATAES132 when I^2C communication mode is selected.

If Auth signaling is enabled, then the SO pin functions as the AuthO signal output. In this configuration the AuthO signal is high after a specified key is authenticated. The AuthO output is in the high impedance state when the device has not authenticated. See Section J.5 for the Auth signaling specifications.

J.2.3. N.C. [Pin 3]

No connect pin. This package pin is not used and can be left open by the user. The state of this pin does not affect the functionality or power consumption of the ATAES132.

J.2.4. V_{SS} [Pin 4]

Ground

J.2.5. SI / SDA [Pin 5]

Bidirectional serial sata I/O pin. In the I^2C communication mode, this pin functions as the serial data I/O (SDA). This pin is an open drain buffer and may be wire ORed with any number of other open drain or open collector devices. The SDA pin must be pulled high with an external resistor for the I^2C bus to operate correctly.

Data on the SDA pin may change only during the SCK low time periods. Data changes during SCK high periods indicate a I^2C START or I^2C STOP condition. Data transfer on the SDA line is half-duplex as described by the I^2C command definitions in Section J.3; the host and client cannot simultaneously drive the SDA line.

J.2.6. SCK [Pin 6]

Serial Clock input pin. In the I^2C communication mode, this pin is used as the serial interface clock (SCK). The SCK input is used to transfer data in to the ATAES132 on the rising edge of clock and to transfer data out on the falling edge of clock. The ATAES132 never drives SCK because it is a standard-mode I^2C slave device – slave device clock stretching is not supported. The SCK line is high when the bus is idle.

If the I^2C master uses a normal totem pole output to drive SCK, then no pull-up resistor is required on the SCK line. If the I^2C master uses an open drain or open collector output to drive SCK, then an external pull-up resistor is required.

J.2.7. N.C. [Pin 7]

No connect pin. This package pin is not used and can be left open by the user. The state of this pin does not affect the functionality or power consumption of the ATAES132.

J.2.8. Vcc [Pin 8]

Supply voltage. Power cannot be removed from the ATAES132 when the I^2C interface is active. The device may be permanently damaged if the requirements in Section 9.1 and Section 9.3 are exceeded.

J.3. I^{2}C Instruction Set

The ATAES132 utilizes the Atmel AT24C32C Serial EEPROM instruction set. The ATAES132 I^{2}C Instruction Set is shown in Table J-1.

Table J-35. I^{2}C Instruction Set for the Atmel ATAES132

Instruction Name	Operation
BWRITE	Byte write. Writes one byte to memory
PWRITE	Page write. Writes 2 to 32 bytes to memory
READ	Read. Reads data from memory starting at the current address
RREAD	Random read. Reads data from memory starting at the specified address
SREAD	Sequential read. Reads additional data from memory
SRESET	Software reset. Resets the internal memory address counter to 0000h

If the ATAES132 receives an invalid or undefined instruction code it will be ignored and the associated data bytes will be discarded. When any error occurs, the EERR bit of the STATUS register is set to 1b to indicate an error. The host can read the error code from the response memory buffer at address 0xFE00 using the READ command.

J.3.2. Byte Write (BWRITE)

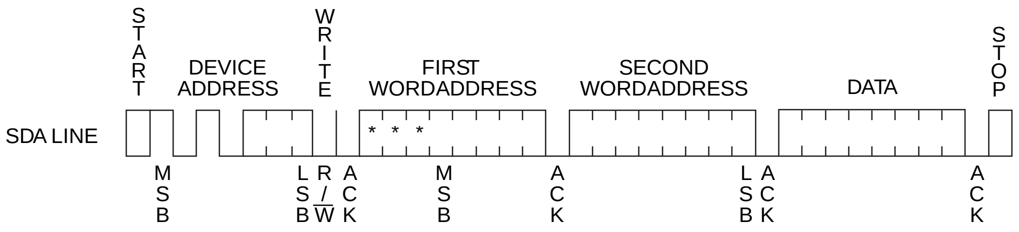
A byte write operation requires two 8-bit data word addresses following the I^{2}C device address byte. Upon receipt of the START condition and device address, the ATAES132 will respond with I^{2}C ACK and then clock in the two address bytes (ACKing each byte). The ATAES132 will ACK the receipt of the data byte from the Host. The host microcontroller must terminate the write sequence with a STOP condition to initiate the write operation.

At this time, the EEPROM enters an internally-timed write cycle to the nonvolatile memory. All inputs are disabled during this write cycle and the EEPROM will NAK the device address until the write is complete.

If the Host transmits an invalid address, the EEPROM will NAK the second address byte and any data bytes.

When any error occurs, the RRDY and EERR bits of the STATUS register are set to 1b to indicate an error. The Host can read the error code from the response memory buffer (address 0xFE00) using the RREAD command. If the command is processed without error, the EERR bit is set to 0b. Reading the response memory buffer does not reset the error code or the STATUS register .

Figure J-2. Byte write



J.3.3. Page Write (PWRITE)

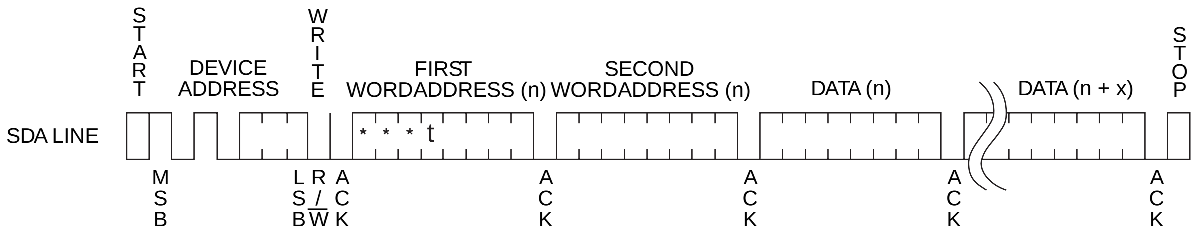
The ATAES132 is capable of 32-byte page writes. A page write is initiated the same way as a byte write, but the host microcontroller does not send a STOP condition after the first data byte is clocked in. Instead, after the device ACKs receipt of the first data byte, the host microcontroller can transmit up to 31 more data bytes (each byte will be ACKed by the ATAES132). The EEPROM will respond with an I^{2}C ACK after each data byte is received. The Host must terminate the page write sequence with a STOP condition. The data address is internally incremented following the receipt of each data byte.

If more than 32 bytes of data are transmitted or the page boundary is crossed, then no data will be written.

If the host transmits an invalid word address the EEPROM will NAK the second address byte and all data bytes.

When any error occurs, the RRDY and EERR bits of the STATUS register are set to 1b to indicate an error. The host can read the error code from the response memory buffer (address 0xFE00) using the RREAD command. If the command is processed without error, the EERR bit is set to 0b. Reading the response memory buffer does not reset the error code or the STATUS register.

Figure J-3. Page write



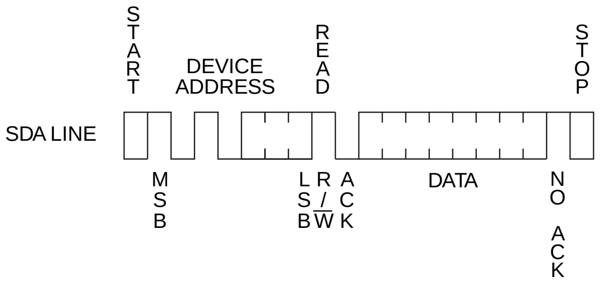
J.3.4. Current Address Read (READ)

The internal data byte address counter maintains the last address accessed during the last read or write operation, incremented by one. This address stays valid between operations as long as the device power is maintained.

To perform a current address read, the host sends the device address with the read/write select bit set to one (READ), this byte is ACKed by the EEPROM. Then the host clocks out the data byte located at the current address. After the byte is received, the host responds with an I^{2}C NAK and a following STOP condition to terminate the read operation.

When any error occurs, the EERR bit of the STATUS register is set to 1b to indicate an error. If the command is processed without error the EERR bit is set to 0b.

Figure J-4. Current address read of one data byte



J.3.5. Random Read (RREAD)

A random read requires a “dummy” byte write sequence to load in the data byte address. Once the device address and data byte address are clocked in and acknowledged by the ATAES132, the host microcontroller must generate another start condition. The microcontroller then initiates a current address read by sending the device address with the read/write select bit high (READ). The ATAES132 I^{2}C ACKs the device address and serially clocks out the data byte. After the byte is received, the host responds with an I^{2}C NAK and a following STOP condition to terminate the read operation.

If the host transmits an invalid word address the EEPROM will NAK the second address byte.

When any error occurs the EERR bit of the STATUS register is set to 1b to indicate an error. If the command is processed without error the EERR bit is set to 0b.

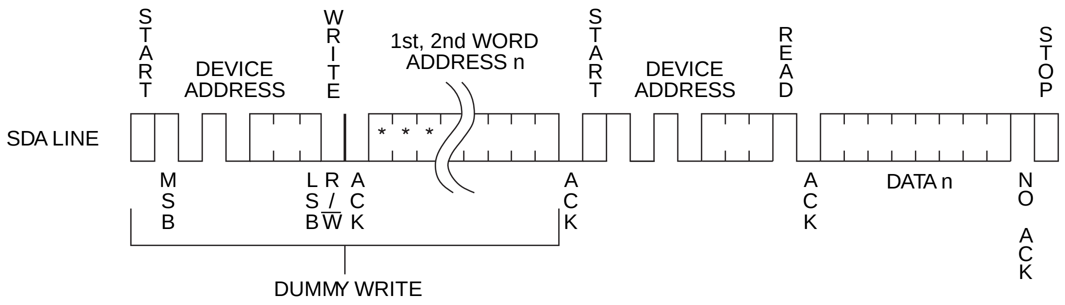


Figure J-5. Random read

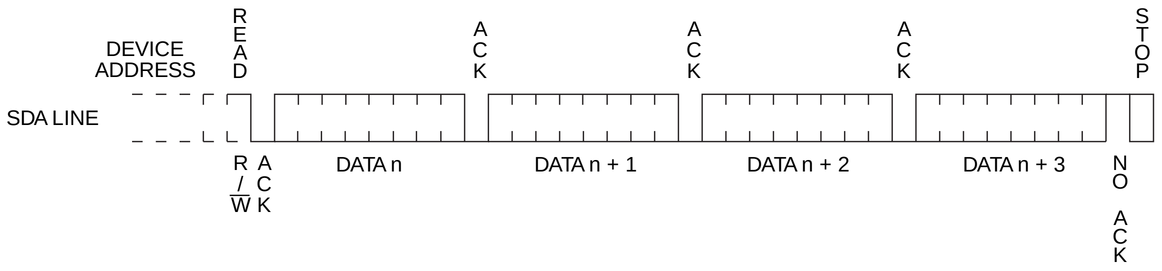
J.3.6. Sequential Read (SREAD)

Sequential reads are initiated by either a current address read or a random read. After the host microcontroller receives a data byte, it responds with an I^2C ACK. As long as the EEPROM receives an acknowledge, it will continue to increment the data byte address and serially clock out sequential data bytes. The sequential read operation is terminated when the microcontroller responds with an I^2C NAK and a following STOP condition.

When any error occurs, the EERR bit of the STATUS register is set to 1b to indicate an error. If the command is processed without error the EERR bit is set to 0b.

Note: If an I^2C read begins at an authorized address and continues into protected memory, the EERR bit will be set to 1b. Attempting to read protected memory will result in 0xFF data returned to the host for each protected byte address.

Figure J-6. Sequential read



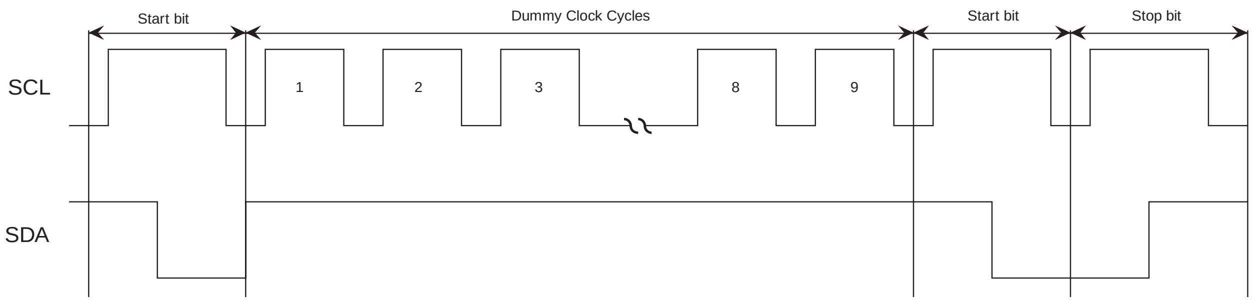
J.3.7. Software Reset (SRESET)

After an interruption in protocol, power loss or system reset, the ATAES132 in I^2C interface mode can be protocol reset by following these steps:

- Send a START condition
- Clock 9 cycles
- Send another START condition followed by STOP condition as shown below

The device is ready for next communication after these steps have been completed. The internal data address is also reset to 0000h by this procedure.

Figure J-7. Software reset



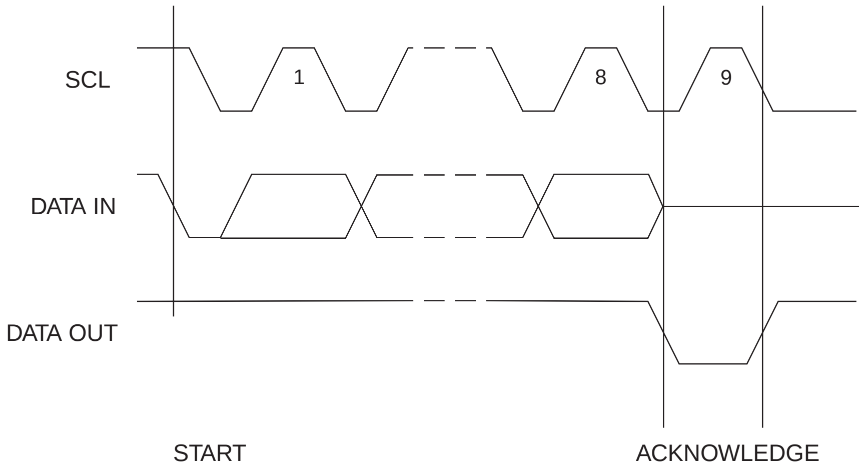
The ATAES132 requires that the clock be pulled low between the START condition and the STOP condition at the end of the sequence as illustrated in Figure J-7, it will not reset if this clock transition is omitted. See Section J.4 for detailed I^{2}C interface resynchronization instructions.

J.3.8. Acknowledge Polling

The host can initiate an acknowledge (ACK) polling immediately after a write command or the ATAES132 extended Crypto command is transmitted. Acknowledge polling involves sending a START condition followed by the I^{2}C device address. The read/write bit of the I^{2}C device address is representative of the operation desired by the host.

During an EEPROM write operation, the ATAES132 will NAK the I^{2}C device address, indicating the device is "busy". When the internal write cycle has completed, then the ATAES132 will ACK the I^{2}C device address, allowing the read or write sequence to continue. The ATAES132 also NAKs during the processing of Crypto commands, so Acknowledge polling can also be used to determine when processing of the ATAES132 extended commands is complete.

Figure J-8. Output acknowledge (I^{2}C ACK)



J.4. I^{2}C Interface Synchronization Procedure

If the host and client I^{2}C interfaces lose synchronization for any reason, the host should send clocks until SDA goes high, followed by the SRESET command to reset the ATAES132 interface. See Section J.3.7 for a description of the SRESET command.

J.5. I^{2}C Auth Signaling

The Auth signaling option allows an authentication signal (AuthO) to be output by the ATAES132. Auth signaling is available only in the I^{2}C Interface mode in standard plastic packages.

The Auth signaling option is controlled by two bits in the KeyConfig registers – the KeyConfig[KeyID].AuthOut bit and the KeyConfig[KeyID].AuthOutHold bit (see Table J-2). By default KeyConfig[KeyID].AuthOut bit is 0b for all keys, disabling the Auth signaling option.

Table J-36. Auth signaling KeyConfig bit functions

AuthOut Bit	AuthOutHold Bit	Operation
1b	X	First successful Auth command forces AuthO high. Additional Auth commands do not change AuthO, AuthO output remains latched high.
0b	X	Successful or unsuccessful Auth commands cause no AuthO change
X	1b	Authentication reset does not change the AuthO output state
X	0b	Authentication reset forces AuthO to the high impedance state

If the KeyConfig[AKeyID].AuthOut bit is 1b for the authentication key (AKeyID) then Auth signaling is enabled for that key, the AuthO signal is output on the SO pin. AuthO is latched high after a successful inbound only authentication, or mutual authentication using the Auth command (see Section 7.1). AuthO will remain high until the device is powered off unless an authentication reset is received.

If the KeyConfig[AKeyID].AuthOutHold bit is 0b for the key (AKeyID) used to execute an authentication reset, then the AuthO signal latch will be latched in the high impedance state when the command is received (with a correct checksum). If KeyConfig[AKeyID].AuthOutHold bit is 1b then AuthO will be unchanged by execution of an authentication reset sequence.

An authentication reset is an Auth command with mode bits 0 and 1 set to 00b. Knowledge of the key value is not required to execute an authentication reset (see Section 7.1). The ATAES132 does not memorize the KeyID used to activate Auth signaling. Each Auth command is processed using the KeyConfig[AKeyID] bits of the AKeyID in the command packet.

Auth signaling is not a security feature. The AuthO signal does not reflect the real-time state of the AuthComplete status flag. The reset command, the sleep command, and the tamper detectors will not change the state of AuthO. The state of the AuthO latch is determined only by success or failure of the Auth command and the configuration of the KeyConfig bits. The Info command should be used to determine the authentication status of the device (see Section 7.12).

The KeyConfig[AKeyID].AuthOut bit and the KeyConfig[AKeyID].AuthOutHold bit are ignored when the ATAES132 is configured in SPI Interface mode.

J.5.2. Using the AuthO Output

When Auth signaling is enabled, the AuthO signal output is either a logic high or in the high impedance state. AuthO can be used to drive an LED, or as a control signal to other circuitry. When AuthO is used as a control signal a pull down resistor should be used to transform the high impedance state into a logic low.

J.6. I^{2}C Compatibility

The ATAES132 is design to operate on a bus with other I^{2}C compatible devices. The ATAES132 is a standard-mode client device capable of operating at clock speeds up to 1MHz (with bus timing scaled accordingly). The ATAES132 is not a fast-mode or high-speed mode device.

This section lists the I^{2}C options or features which are not supported by the ATAES132. Any feature which differs from the I^{2}C specification is also listed.

- The ATAES132 does not perform client clock stretching
- The ATAES132 will not respond to an I^{2}C “general call” command
- The ATAES132 may be damaged if the clock or data signal levels are above V_{CC}. The power supply to the ATAES132 cannot be switched off while the bus is active. All of the voltage limits in Section 9.1 must be respected.

- The ATAES132 inputs include Schmitt triggers and spike suppression, however, the outputs do not include falling edge slope control.
- On I^{2}C devices a START condition followed immediately by a STOP condition is never permitted. On ATAES132 this sequence is permitted only as part of the SRESET command sequence (see Section J.3.7).

J.7. Timing Diagrams

Figure J-9. I^{2}C synchronous data timing (see Section 9.4.3 for I^{2}C timing specifications)

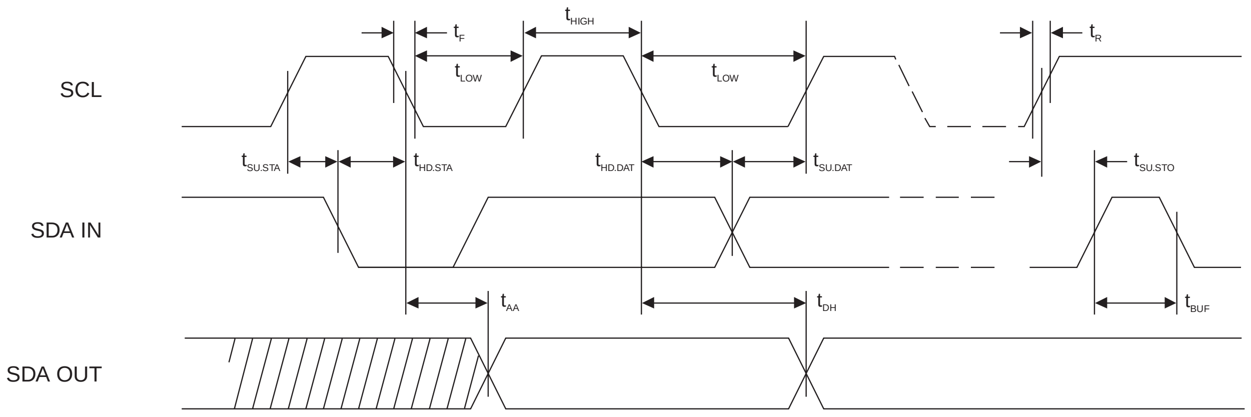
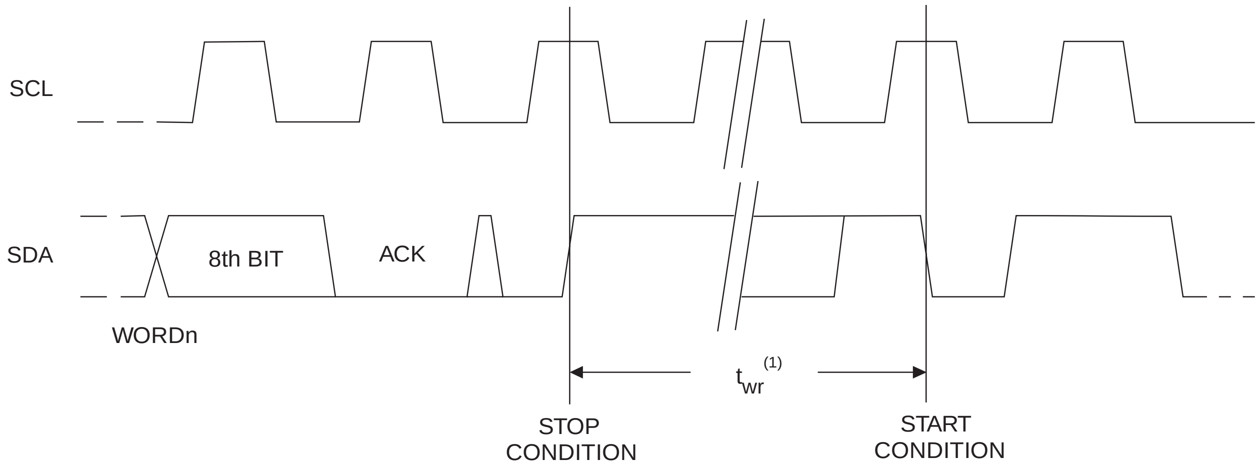


Figure J-10. I^{2}C write cycle timing



Appendix K. SPI Interface

The ATAES132 serial peripheral interface (SPI) is designed to interface directly to the microcontrollers using SPI Mode 0 or Mode 3. IO and clear-text read/write operations operate similar to the Atmel SPI serial EEPROM.

The host sends ATAES132 commands to the device by writing the command packet to the command memory buffer at address 0xFE00. The ATAES132 processes the command packet and places the response in the response memory buffer. The host retrieves the response by reading the response packet from address 0xFE00.

See Section G.3 for additional information on the ATAES132 behavior in SPI interface mode.

K.1. SPI Serial Interface Description

When the ATAES132 is configured in the SPI communication mode, the serial interface operates as a Mode 0 and Mode 3 slave device as described in this appendix. Serial peripheral interface (SPI) is a synchronous serial interface protocol that is a defacto industry standard which is not formally documented or controlled. Multiple SPI devices can share the data bus, however, each SPI slave must have a separate $\overline{CS}$ control line to prevent bus contention.

The serial interface communication mode is selected by programming the I^{2}CAddr register in the configuration memory as described in Section E.2.15.

K.1.1. SPI Master

The SPI bus master device generates the serial clock and sends instructions to the SPI slave devices. In this specification, the bus master is usually referred to as the host or the host microcontroller.

K.1.2. SPI Slave

SPI slave devices receive the serial clock as an input and receive instructions from the bus master. SPI slaves can never generate traffic on the SPI bus, slaves can only respond to instructions provided by the bus master. The ATAES132 always operates as a slave. In this specification the slave is usually referred to as the client.

K.1.3. Relationship of Clock to Data

The ATAES132 supports two of the four standard SPI interface modes, Mode 0 and Mode 3.

In Mode 0, the default state of SCK is low, and the data is clocked in (SI) on the rising edge of the clock. Data out (SO) changes on the falling edge of the clock.

In Mode 3, the default state of SCK is high, and data is clocked in (SI) on the rising edge of the clock. Data out (SO) changes on the falling edge of the clock.

K.1.4. SPI Instruction Code

Each SPI command begins with the SPI master bring the $\overline{CS}$ input low to select the device, followed by transmission of an eight bit SPI instruction code to the SI input of the SPI slave. Following the instruction code, additional bytes may be clocked into SI or out of SO as required by the SPI command (see Section K.3 for SPI command definitions). When the exchange of data bytes related to the SPI instruction code is complete, then the $\overline{CS}$ input is brought high to deactivate the SPI slave interface.

If an invalid instruction code is received, then the ATAES132 will ignore any data received on the data input pin (SI), and the data output pin (SO) will remain in a high impedance state.

K.1.5. Data Format

All instructions and data on the SPI bus must be formatted as eight bit bytes. The MSB is the first bit of each byte transmitted and received.

K.2. Pin Descriptions

When the ATAES132 is configured in SPI communication mode, the package pins are assigned the functionality described in this section.

K.2.1. $\overline{\text{CS}}$ [Pin 1]

SPI chip select bar input pin. In SPI communication mode, this pin functions as the slave select input. ATAES132 is selected when the $\overline{\text{CS}}$ pin is low, allowing instructions and data to be accepted on the serial data input pin (SI), and allowing data to be transmitted on the serial data output pin (SO). When the device is not selected, data will not be accepted via the SI pin, and the serial output pin (SO) will remain in a high impedance state.

When the ATAES132 is in the standby state or sleep state, a high-to-low transition on the $\overline{\text{CS}}$ pin will cause the device to wakeup (See Appendix L for power management specifications). It is recommended that the $\overline{\text{CS}}$ pin be connected to V_{CC} with a pull-up resistor so the $\overline{\text{CS}}$ pin follows V_{CC} during power up and power down.

K.2.2. SO [Pin 2]

Serial data out pin. In the SPI communication mode, this pin functions as the serial data output. When the $\overline{\text{CS}}$ pin is high, the SO pin will always be in a high impedance state because the SPI interface is disabled.

K.2.3. N.C. [Pin 3]

No connect pin. This package pin is not used and can be left open by the user. The state of this pin does not affect the functionality or power consumption of the ATAES132.

K.2.4. V_{SS} [Pin 4]

Ground

K.2.5. SI / SDA [Pin 5]

Serial data in pin. In the SPI communication mode, this pin functions as the serial data input. When the $\overline{\text{CS}}$ pin is high, the SI pin will not accepted data because the SPI interface is disabled.

K.2.6. SCK [Pin 6]

Serial clock input pin. In the SPI communication mode, this pin is used as the serial interface clock. All data on the SI and SO pins is synchronized by SCK as described in Section K.1.3.

K.2.7. N.C. [Pin 7]

No connect pin. This package pin is not used and can be left open by the user. The state of this pin does not affect the functionality or power consumption of the ATAES132.

K.2.8. Vcc [Pin 8]

Supply voltage. Power cannot be removed from the ATAES132 when the SPI bus is active. The device may be permanently damaged if the requirements in Section 9.1 and Section 9.3 are exceeded.

K.3. SPI Instruction Set

The ATAES132 utilizes an 8-bit SPI instruction register. The SPI instruction set is listed in Table K-1.

Table K-37. SPI instruction set for the Atmel ATAES132

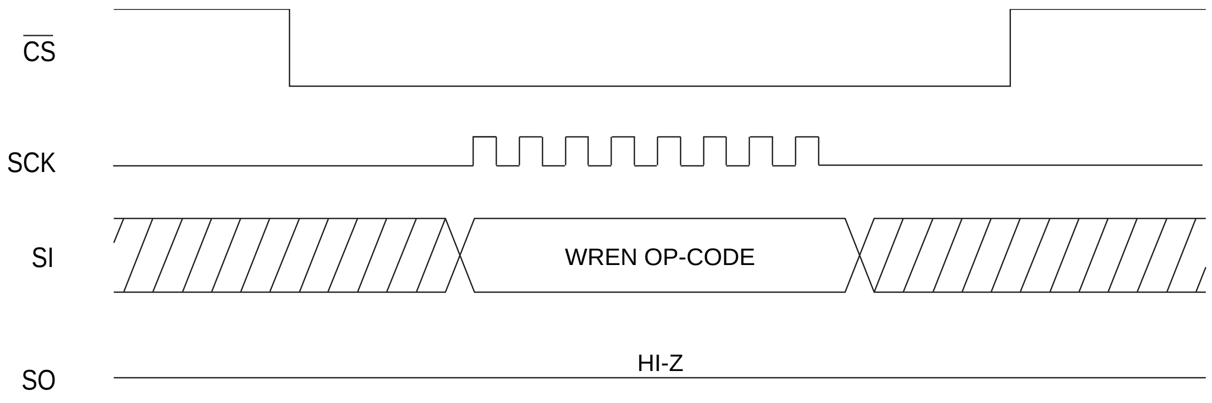
Instruction name	Instruction code	Operation
WRITE	0000 0010 b	Write data to memory
READ	0000 0011 b	Read data from memory
WRDI	0000 0100 b	Reset write enable register
RDSR	0000 0101 b	Read status register
WREN	0000 0110 b	Set write enable latch

If the ATAES132 receives an invalid instruction code or an invalid memory address, then no response will be sent – the SO output will remain in the high impedance state. When any error occurs, the EERR bit of the STATUS register is set to 1b to indicate an error. The host can read the error code from the response memory buffer at address 0xFE00 using the READ command. Reading the response memory buffer does not reset the error code or change the STATUS.

K.3.2. Write Enable Command (WREN):

The device will power up in the write disable state when V_{CC} is applied. All EEPROM write instructions must therefore be preceded by a write enable instruction. It is not necessary to send the write enable instruction prior to sending command packets to the command memory buffer.

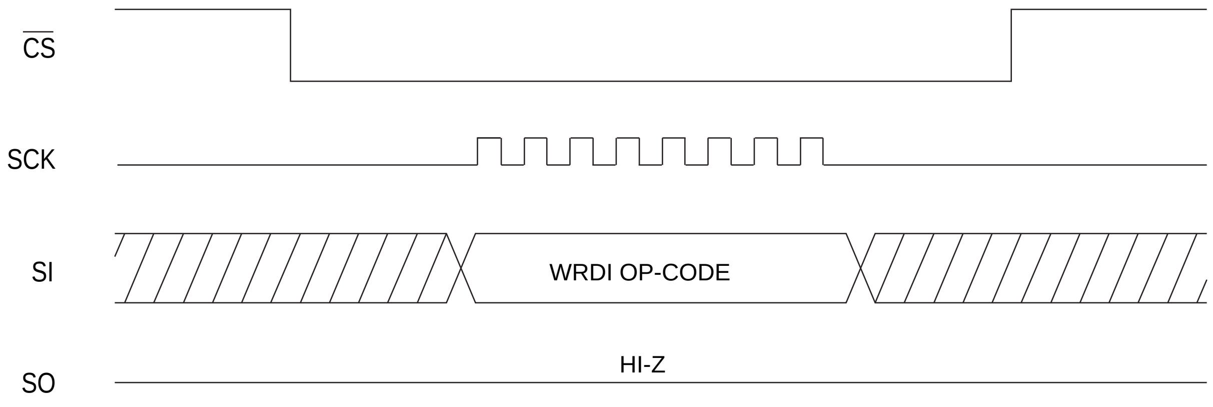
Figure K-11. SPI write enable (WREN) timing



K.3.3. Write Disable Command (WRDI):

The write enable flag can be disabled by sending the write disable instruction.

Figure K-12. SPI write disable (WRDI) timing

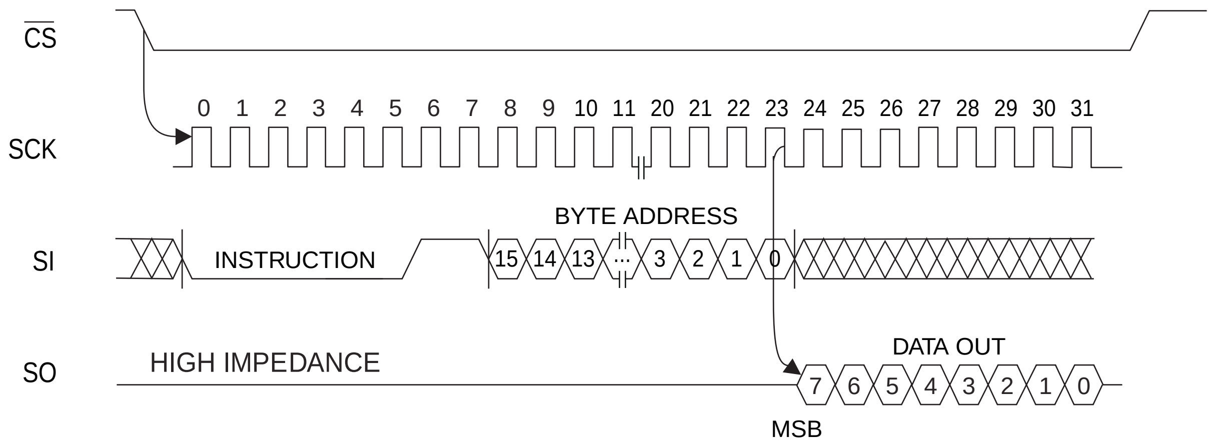


K.3.4. Read Memory Command (READ):

Reading data from the ATAES132 requires the following sequence. The host drives the $\overline{\text{CS}}$ line low to select a device and then transmits the read instruction code on the SI line followed by the address of the byte to be read. The client ignores any data on the SI line which follows a read memory instruction.

The client shifts out the data at the specified address on the SO line. If only one byte is to be read, the $\overline{\text{CS}}$ line must be driven high after the data byte comes out. If multiple bytes are to be read, the host can sequentially clock the data out of the ATAES132 since the byte address is automatically incremented. The $\overline{\text{CS}}$ line must be driven high by the host after the last data byte is read. If the highest address is reached, the address counter will *not* roll over.

Figure K-13. SPI READ memory timing



When any error occurs, the EERR bit of the STATUS register is set to 1b to indicate an error. If the command is processed without error the EERR bit is set to 0b.

Note: If an SPI read begins at an authorized address, but continues into protected memory the EERR bit will be set to 1b

K.3.5. Write Memory Command (WRITE):

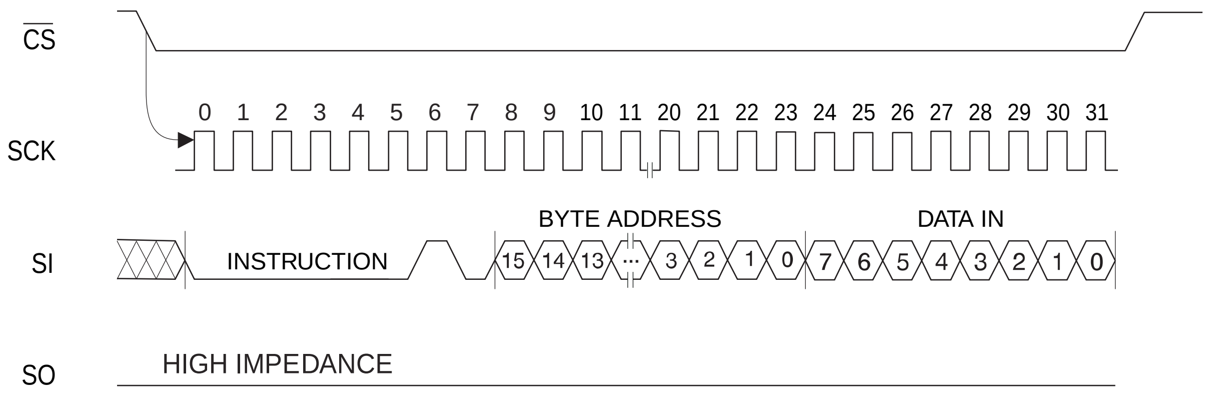
In order to write to the ATAES132, two separate instructions must be executed. First, the device must be write enabled via the write enable (WREN) instruction. Then a write memory instruction may be executed. All commands received while a write cycle is in progress will be ignored except the read status register (RDSR) instruction.

A write memory command requires the following sequence. The host drives the $\overline{CS}$ line low to select a device and then transmits the write instruction code on the SI line followed by the address of the byte to write and the 1 to 32 data bytes to be written. The byte address is automatically incremented as each byte is clocked in. The $\overline{CS}$ line must be driven high by the host during the SCK low time immediately after clocking in the last data bit. The low-to-high transition of the $\overline{CS}$ pin initiated the EEPROM write process. The SO pin remains in the high impedance state during the entire write sequence.

The ready/busy status of the device can be determined by initiating a read status register (RDSR) instruction. If the WIP status bit is 1b, the write cycle is still in progress. If the WIP status bit is 0b, the write cycle has ended and the ATAES132 is ready to accept a new command. Only the read status register (RDSR) instruction is enabled during the EEPROM write cycle.

The ATAES132 is capable of a 32-byte page write operation. After each byte of data is received, the data address is internally incremented by one. If more than 32 bytes of data are transmitted or if the page boundary is crossed, then no data will be written. The ATAES132 is automatically returned to the write disable state at the completion of a write cycle.

Figure K-14. SPI WRITE memory timing



When any error occurs, the RRDY and EERR bits of the STATUS register are set to 1b to indicate an error. The host can read the error code from the response memory buffer (address 0xFE00) using the READ command. If the command is processed without error, the EERR bit is set to 0b. Reading the response memory buffer does not reset the error code or the STATUS register .

If the device is not write enabled (WREN), the device will ignore the write instruction and will return to the waiting for a command. A new $\overline{CS}$ falling edge is required prior to the new instruction code.

K.3.6. Read Status Register Command (RDSR):

The read status register instruction provides access to the STATUS register. The ready/busy status of the device can be determined using the RDSR instruction. Alternately, the STATUS register can be read directly from memory as described in Section G.2.4.

If the ATAES132 is performing an EEPROM memory write or is processing a command when the STATUS read is performed, then all eight bits are ones if the RDSR command is used to read the STATUS register, emulating the behavior of Atmel Serial EEPROM. See Appendix G for a detailed description of the STATUS register bits and status bit behavior.

Table K-38. Device status register definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EERR	RRDY	Reserved	CRCE	Reserved	WAKEb	WEN	WIP

The device status register can always be read, even if the the ATAES132 is processing a command or writing the EEPROM. The SPI RDSR command is the preferred method for reading the STATUS in SPI interface mode.

If the ATAES132 is in the sleep or standby power state, reading the STATUS register forces the ATAES132 to wakeup – the STATUS register is 0xFF until the wakeup process is complete.

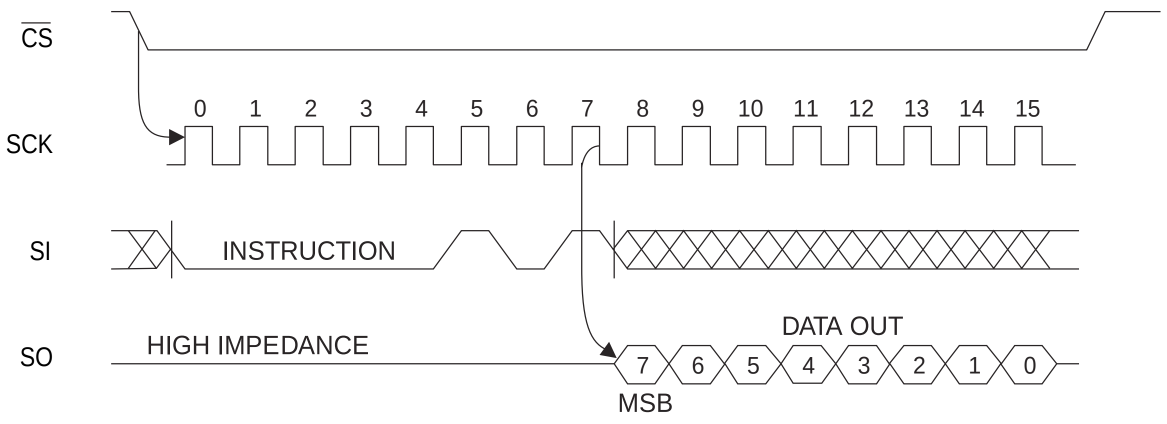
Table K-39. Read status register bit definition using SPI RDSR command(1)(2)

Bit	Definition
Bit 0 (WIP)	“0b” indicates the device is ready, waiting for a command “1b” indicates a write cycle or a cryptographic operation is in progress
Bit 1 (WEN)	"0b" indicates the device is not SPI write enabled “1b” indicates the device is SPI write enabled
Bit 2 (WAKEb)	"0b" indicates the device is not in the sleep or standby power state “1b” indicates the device is in the sleep or standby power state
Bit 3 (Reserved)	Always "0b". This bit is reserved for future use.(1)
Bit 4 (CRCE)	"0b" indicates the most recent command block contained a correct checksum (CRC) “1b” indicates the most recent command block contained an error
Bit 5 (Reserved)	Always "0b". This bit is reserved for future use.(1)
Bit 6 (RRDY)	"0b" indicates the response memory buffer is empty “1b” indicates the response memory buffer is ready to read
Bit 7 (EERR)	"0b" indicates the most recent command did not generate an error during execution “1b” indicates the most recent command generated an execution error

Notes: 1. When the SPI RDSR command is used to read the STATUS register during an EEPROM write or during execution of any ATAES132 command, then status bits 0 - 7 are "1b"s. The reserved bits will read as 0b if the STATUS register is read directly from memory during an EEPROM write or during execution of an ATAES132 command.

2. STATUS register bits 0 - 7 are "1b"s during wakeup. During the first phase of wakeup ($t_{PU.STATUS}$), the SO pin is tri-stated and any attempt to read it will be system-dependent. See for Appendix L additional information.

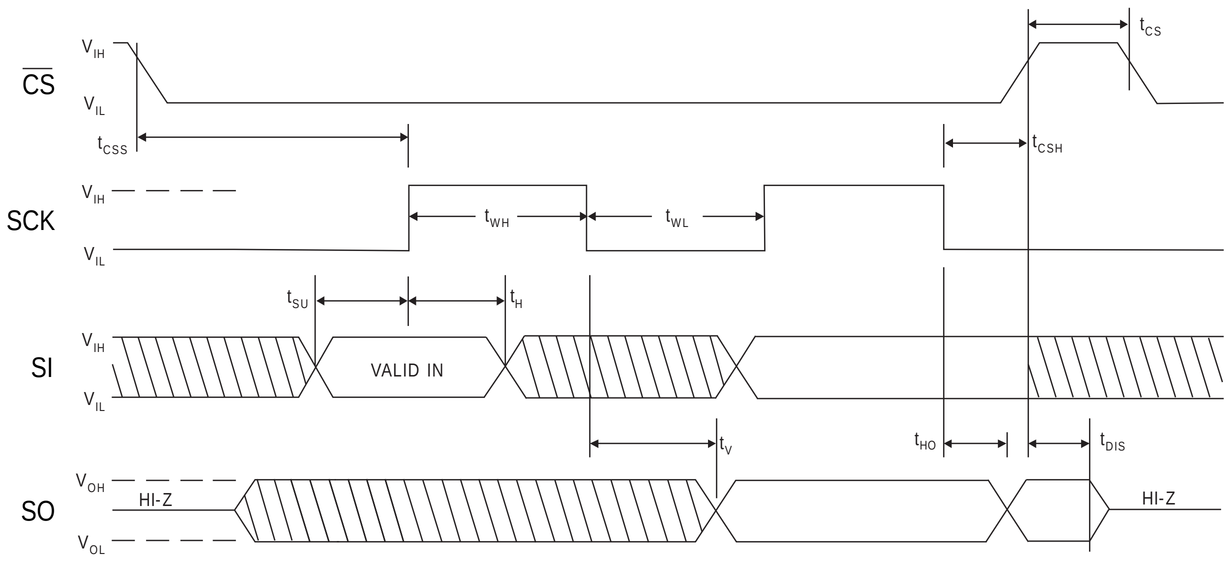
Figure K-15. SPI read status register (RDSR) timing



Reading the STATUS register does not change the contents STATUS register or the contents of the response memory buffer.

K.4. Timing Diagram

Figure K-16. SPI synchronous data timing (see Section 9.4.4 for SPI Timing Specifications)



Appendix L. Power Management

The ATAES132 contains several features which facilitate power management. This appendix describes the various power states and features.

L.1. Power State Descriptions

The ATAES132 has three powered states, and the off state. Two low power states are available to reduce power consumption when the system is not using the ATAES132.

L.1.1. Active State

The ATAES132 is in the active state after it has completed the power up process and is fully powered. The WIP status bit is 0b when the ATAES132 is in the active state and waiting for a command. The WIP status bit is 1b when the ATAES132 is in the active state and processing a command or performing an EEPROM write. (See Section G.1.2 for WIP Status Bit information)

The supply current of the ATAES132 in the active state is several milliamps. (See Section 9.3.1 for Icc specifications)

An ATAES132 in the Active State is capable of accepting a command immediately if the WIP Status Bit is 0b. The I^2C Timing Specifications for the Active State are in Section 9.4.3. The SPI Timing Specifications for the Active State are in Section 9.4.4.

L.1.2. Standby State

ATAES132 can enter the Standby State in two ways: 1. The host can send a Sleep command to place ATAES132 into Standby. 2. ATAES132 will automatically enter the Standby State at power up if configured to do so. (See Section L.2.1) The Standby State preserves the ATAES132 volatile memory contents and the security state.

All eight Status Bits are 1b when ATAES132 is in the Standby State and during the wakeup process. (See Appendix G for Status Bit information)

The supply current of ATAES132 in the Standby State is several microamps (See Section 9.3.1 for I_{SB} specifications)

An ATAES132 in the Standby State is capable of reporting the device STATUS immediately, but cannot accept a command until the Wakeup process is complete. The Timing Specifications for exiting the Standby State are in Section 9.4.2.

L.1.3. Sleep State

The ATAES132 can enter the sleep state in two ways:

1. The host can send a sleep command to place the ATAES132 into standby
4. The ATAES132 will automatically enter the sleep state at power up if configured to do so. (See Section L.2.1)

The sleep state clears the ATAES132 volatile memory contents and the security state.

All eight status bits are 1b when the ATAES132 is in the sleep state and during the wakeup process. (See Appendix G for Status Bit information)

The supply current of the ATAES132 in the standby state is less than one microamp. (See Section 9.3.1 for I_{SB} specifications)

An ATAES132 in the sleep state is capable of reporting the device STATUS immediately, but cannot accept a command until the wakeup process is complete. The timing specifications for exiting the sleep state are in Section 9.4.2.

L.1.4. Off State

When the ATAES132 device is unpowered or when V_{CC} is significantly below the minimum V_{CC} voltage, then the device is in the off state. A device in the off state cannot respond to any commands.

L.2. Power State Transitions

Power Up is a transition from the Off State to one of the three powered states. Power Down is the transition from a powered state to the Off State. Wakeup is the transition from one of the two low power states to the Active State.

L.2.1. Power Up

Power up begins when the power supply is turned on, causing the V_{CC} voltage to rise continuously from V_{SS} to the operating voltage. Power up occurs in three stages.

1. **First stage**
   The voltage regulator and other analog circuitry are activated
5. **Second stage**
   The serial interface logic is activated so that the ATAES132 can report the device status to the host
6. **Third stage**
   The ATAES132 enters the power state specified by the ChipConfig register

During the power up process, the device is unable to accept commands. In the SPI interface mode, the device is ready to receive a read status register command after the power up time $t_{PU.STATUS}$. The power up ready time of $t_{PU.RDY}$ specifies the time required to complete the power up process. In the I^2C interface mode, the device will NAK all instructions prior to the completion of power up (time $t_{PU.RDY}$).

The last stage of the power up procedure is to enter the active, standby, or sleep state specified by bit 6 and 7 of the ChipConfig register. The ChipState register is set to 0xFFFF at power up. (see Section L.3)

Table L-40. Coding of the ChipConfig.PowerUpState bits in the ChipConfig register

Bit 7	Bit 6	Description
1	1	Device goes to the Active State at Power Up
1	0	
0	1	Device goes to the Standby State at Power Up
0	0	Device goes to the Sleep State at Power Up

During power up, the SPI chip select should follow the V_{CC} voltage. It is recommended that the $\overline{CS}$ pin be connected to V_{CC} with a pull-up resistor if the ATAES132 is configured in the SPI interface mode. The ATAES132 does not support hot swapping or hot plugging. Connecting or disconnecting this device to a system while power is energized can cause permanent damage to the ATAES132.

L.2.2. Power Down

Before power down, the device must be deselected (if configured for SPI) and placed in the active, standby, or sleep state. During power down, the SPI chip select should be allowed to follow the V_{CC} voltage if the ATAES132 is configured in SPI interface mode.

The ATAES132 should not be powered down when the WIP status bit indicates that an EEPROM write or cryptographic operation is in progress. If the WIP status bit is 0b, then it is safe to power down the device.

L.2.3. Entering the Standby State

If the ATAES132 is in the active state, the host can send a sleep command to place the ATAES132 in the standby state. (See Section 7.24) It is not possible to transition the device directly from the sleep state to the standby state. The host must wakeup the device and then must send a sleep command to place the device in standby.

The device can also be configured to enter the standby state at power up as described in Section L.2.1.

The ATAES132 exits standby state only if a wakeup event occurs on the IO pins. Wakeup is discussed in Sections L.2.5 and L.2.6. The ChipState register does not change when the ATAES132 enters or leaves the standby state. (see Section L.3)

L.2.4. Entering the Sleep State

If the ATAES132 is in the active state, the host can send a sleep command to place the ATAES132 in the sleep state. (See Section 7.24) It is not possible to transition the device directly from the standby state to the sleep state. The host must wakeup the device and then must send a sleep command to place the device in sleep.

The device can also be configured to enter the sleep state at power up as described in Section L.2.1.

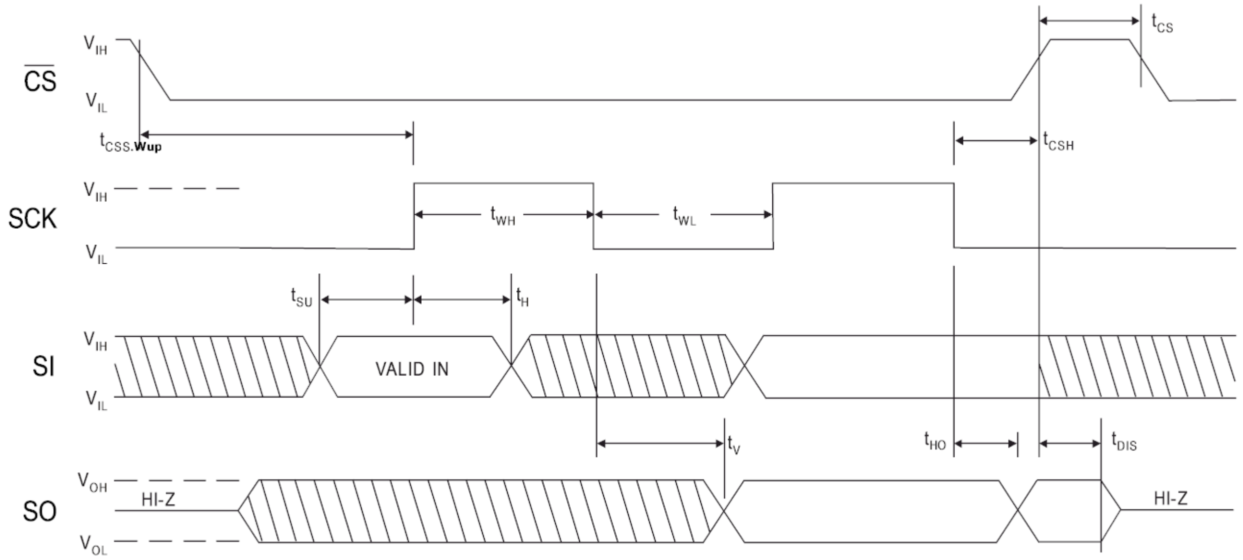
The ATAES132 exits sleep mode only if a wakeup event occurs on the IO pins. Wakeup is discussed in Sections L.2.5 and L.2.6. The ChipState register changes to 0x5555 when the ATAES132 leaves the sleep state. (see Section L.3)

L.2.5. SPI WakeUp

To wakeup the ATAES132 configured for SPI interface mode, the host is required to read the status register using the SPI read status register command. The ATAES132 will answer the SPI read status register command with the device "status" if the host has not violated the 100nS minimum $t_{CSS.Wup}$ setup time requirement. The ATAES132 "status" will indicate the device is "busy" (status = 0xFF) during wakeup. When wakeup is complete, the ATAES132 "status" changes to indicate the device is in the active state.

The ATAES132 will only accept the SPI read status register command while it is "busy". All other commands will be ignored. The SPI read status register command is described in Section K.3.6.

Figure L-17. SPI interface timing, $\overline{CS}$ setup time at wakeup



The wakeup process begins when a device in the standby or sleep state experiences a high-to-low transition of the $\overline{CS}$ pin. The device is ready to receive a read status register command from the host after wakeup time $t_{WupSB.STATUS}$ for the standby state, or $t_{WupSL.STATUS}$ for the sleep state. The wakeup is complete after the wakeup ready time of $t_{WupSB.RDY}$ for the standby state, or $t_{WupSL.RDY}$ for the sleep state – $t_{WupSB.RDY}$ / $t_{WupSL.RDY}$ begin when the $\overline{CS}$ pin high-to-low transition occurs and end when the device enters the active state. The wakeup timing specifications are in Table 9-5.

L.2.6. I^{2}C WakeUp

To wakeup an ATAES132 configured for I^{2}C interface mode, the host is required to perform ACK polling using the matching I^{2}C device address. The ATAES132 will answer the ACK poll with an I^{2}C NAK to indicate the device is "busy" during wakeup. The ACK poll reply will change to ACK when the device is in the active state.

The ATAES132 will not accept any commands while it is "busy". The ATAES132 will NAK the I^{2}C device address if it does not match the internal I^{2}C device address, and will not wakeup if a non-matching I^{2}C device address is received.

The wakeup process begins when a device in the standby or sleep state receives an I^{2}C start signal followed immediately by a I^{2}C device address that matches the ATAES132 I^{2}CAddr register. The device is ready to receive an ACK poll from the host after wakeup time $t_{WupSB.STATUS}$ for the standby state, or $t_{WupSL.STATUS}$ for the sleep state. The wakeup is complete after the wakeup ready time of $t_{WupSB.RDY}$ for the standby state, or $t_{WupSL.RDY}$ for the sleep state – $t_{WupSB.RDY}$ / $t_{WupSL.RDY}$ begin when a matching I^{2}C address is received, and end when the device enters the active state. The wakeup timing specifications are in Table 9-5.

L.3. Understanding the ChipState Register

The info command (see Section 7.12) provides access to the ChipState register. The ChipState register value indicates if the device has recently experienced a power up event or wakeup from the sleep power state. This information can be useful for determining how to recover from an unexpected transaction error.

Table L-41. Description of the ChipState register value returned by the info command

ChipState	Description
0x0000	ChipState = Active. Device has remained Active since the previous Crypto command was processed(1)
0x5555	ChipState = "Wakeup from sleep". Device has experienced a wakeup from the sleep power state since the previous Crypto command was processed(1)
0xFFFF	ChipState = Power up. Device has experienced a power up event since the previous Crypto command was processed(1)

Notes: 1. The following subsections describe the events which cause ChipState to change values, and events that do not change ChipState

L.3.2. ChipState = Power Up

The following events cause the ChipState register to be set to the power up state (0xFFFF). The events in this table cause the device to be initialized and placed in the power state specified in the ChipConfig register. (see Section L.2.1)

Table L-42. Description of events causing the ChipState register to be set to 0xFFFF

Event	Event description
Power up	Power up of the device (Section L.2.1)
Power interruption	Power interruption or brownout resulting in device reset

L.3.3. ChipState = "WakeUp from Sleep"

The following events cause the ChipState register to be set to the "WakeUp from Sleep" state (0x5555). The events in this table cause the security registers to be cleared, the logic reinitialized, and the device returned to the active power state (ready to receive a command).

Table L-43. Description of events causing the ChipState register to be set to 0x5555

Event	Event description
WakeUp from Sleep	WakeUp from the Sleep Power State (Section L.1.3)
Reset Command	Device receives a valid Reset command block. (Section 7.23)
Tamper	Device reset initiated by the tamper sensors. (Section 3.1.2)

L.3.4. Events that Do Not Change ChipState

The following events cause *NO CHANGE* in the ChipState register value. These events do not modify the security state of the ATAES132 and therefore do not cause the ChipState to change.

Table L-44. Description of events causing *NO CHANGE* in the ChipState register

Event	Event description
WakeUp from Standby	WakeUp from the Standby Power State (Section L.1.2)
Reading STATUS	Reading the STATUS register with SPI RDSR or standard read commands (Appendix G)
Writing IO Address Reset	Writing the IO address reset register (Section D.4)
Reading a Response	Reading the response memory buffer (Section D.3)
Command CRC Error	Device receives ANY command block which results in a CRCE Error(1) (Section G.1.5)
Command Invalid	Device receives a command block containing an undefined/invalid Opcode (Section 6.2)
ACK Polling	I^2C acknowledge polling (Section J.3.8)
I^2C Read	I^2C standard read [READ, RREAD, SREAD instructions] (Section J.3)
Invalid I^2C Write	I^2C standard write beginning at any address from 0x1000 to 0xEFFF or above 0xF300, except address 0xFE00 [BWRITE, PWRITE instructions] (2) (Section J.3)
I^2C SRESET	I^2C SRESET instruction (Section J.3.7)
SPI Read	SPI standard read [READ instruction] (Section K.3.4)
Invalid SPI Write	SPI standard write beginning at any address from 0x1000 to 0xEFFF or above 0xF300, except address 0xFE00 [WREN, WRITE, WRDI instructions] (2) (Section K.3)
Info Command	Device receives a valid info command block (Section 7.12)

Notes: 1. A CRCE error results from a command block with a short count, bad checksum, or a buffer overrun

2. Writing the command memory buffer (address 0xFE00) may or may not change the ChipState, depending on which command is written to the buffer

L.3.5. ChipState = Active

The following events cause the ChipState register to be set to the active state (0x0000). The events in this table may result in a change in the security state of the device.

Table L-45. Description of events causing the ChipState register to be set to 0x0000

Event	Event description
Auth Command	Device receives a valid Auth command block (Section 7.1)
AuthCheck Command	Device receives a valid AuthCheck command block (Section 7.2)
AuthCompute Command	Device receives a valid AuthCompute command block (Section 7.3)
BlockRead Command	Device receives a valid BlockRead command block (Section 7.4)
Counter Command	Device receives a valid Counter command block (Section 7.5)
Crunch Command	Device receives a valid Crunch command block (Section 7.6)
DecRead Command	Device receives a valid DecRead command block (Section 7.7)
Decrypt Command	Device receives a valid Decrypt command block (Section 7.8)
EncRead Command	Device receives a valid EncRead command block (Section 7.9)
Encrypt Command	Device receives a valid Encrypt command block (Section 7.10)
EncWrite Command	Device receives a valid EncWrite command block (Section 7.11)
KeyCompute Command	Device receives a valid KeyCompute command block (Section 7.13)
KeyExport Command	Device receives a valid KeyExport command block (Section 7.14)
KeyImport Command	Device receives a valid KeyImport command block (Section 7.15)
KeyLoad Command	Device receives a valid KeyLoad command block (Section 7.16)
KeyTransfer Command	Device receives a valid KeyTransfer command block (Section 7.17)
Legacy Command	Device receives a valid Legacy command block (Section 7.18)
Lock Command	Device receives a valid Lock command block (Section 7.19)
Nonce Command	Device receives a valid Nonce command block (Section 7.20)
NonceCompute Command	Device receives a valid NonceCompute command block (Section 7.21)
Random Command	Device receives a valid Random command block (Section 7.22)
Sleep Command	Device receives a valid Sleep command block (Section 7.24)
TempSense Command	Device receives a valid TempSense command block (Section 7.25)
WriteCompute Command	Device receives a valid WriteCompute command block (Section 7.26)
I^{2}C Write	I^{2}C standard write beginning at any user zone address, any configuration memory address, or any key memory address [BWRITE, PWRITE instructions] (Section J.3)
SPI Write	SPI standard write beginning at any user zone address, any configuration memory address, or any key memory address [WREN, WRITE, WRDI instructions] (Section K.3)

Appendix M. Block Checksum

An Atmel CRC-16 checksum is used to verify the integrity of blocks communicated to and from the ATAES132.

The host sends ATAES132 extended commands to the device in a block of at least four bytes. The ATAES132 responses are returned to the host in a block of at least four bytes. The command and response blocks are constructed in the following manner:

Byte #	Name	Meaning
0	Count	Number of bytes to be transferred to the device in the block, including count, packet and checksum. This byte will always have a value of N.
1 to (N-3)	Packet	Command, parameters and data, or response. Data is transmitted in the byte order shown in command definitions in Section 7.
N-2, N-1	Checksum	Atmel CRC-16 verification of the count and packet bytes.

The Atmel CRC-16 polynomial is 0x8005. The initial register value should be 0x0000. After the last bit of the count and packet has been transmitted, the internal CRC register should have a value that matches that in the block. The first checksum byte transmitted (N-2) is the most significant byte of the CRC value and last byte of the block is the least significant byte of the CRC.

M.1. Checksum Function

```
/** \This function calculates a 16-bit CRC.
 * \param[in] count number of bytes in data buffer
 * \param[in] data pointer to data
 * \param[out] crc pointer to calculated CRC (high byte at crc[0])
 */
void CalculateCrc(uint8_t length, uint8_t *data, uint8_t *crc)
{
   uint8_t counter;
   uint8_t crcLow = 0, crcHigh = 0, crcCarry;
   uint8_t polyLow = 0x05, polyHigh = 0x80;
   uint8_t shiftRegister;
   uint8_t dataBit, crcBit;

   for (counter = 0; counter < length; counter++) {
      for (shiftRegister = 0x80; shiftRegister > 0x00; shiftRegister >>= 1) {
         dataBit = (data[counter] & shiftRegister) ? 1 : 0;
         crcBit = crcHigh >> 7;

         // Shift CRC to the left by 1.
         crcCarry = crcLow >> 7;
         crcLow <<= 1;
         crcHigh <<= 1;
         crcHigh |= crcCarry;

         if ((dataBit ^ crcBit) != 0) {
            crcLow ^= polyLow;
            crcHigh ^= polyHigh;
         }
      }
   }
   crc[0] = crcHigh;
   crc[1] = crcLow;
}
```

M.2. Checksum Examples

```
DATA = 09 02 02 00 00 00 00  CRC = 0xF960
```

Appendix N. Atmel ATAES132 Command Response Time

The typical and maximum time required for the ATAES132 to process an extended command is shown in Table N-1. The response time is the time from sending the last bit of the last byte of the command block to the command memory buffer until the STATUS register (or I^{2}C ACK) indicates the response block is available. The "Typical" response time is the average time required for an error free command to be processed on a typical device at room temperature. The "Maximum" response time is the worst case time for the command to be processed over the specified temperature range. (With or without an error condition, whichever results in the worst response time)

Table N-46. Typical and maximum response times for the Atmel ATAES132 extended commands [(1)]

Command description	Typical time[(2)]	Maximum time[(3)]
Auth, Reset (Mode [0:1] = 00b)	0.4 milliseconds	
Auth, Inbound Only (Mode [5:7] = 000b)	1.5 milliseconds	
Auth, Inbound Only (Mode [5:7] not 000b)		
Auth, Inbound Only (Mode [5:7] not 000b), with Key Usage[(5)]		
Auth, Outbound Only (Mode [5:7] = 000b)		
Auth, Outbound Only (Mode [5:7] not 000b)		
Auth, Outbound Only (Mode [5:7] not 000b), with Key Usage[(5)]		
Auth, Mutual (Mode [5:7] = 000b)	2.4 milliseconds	
Auth, Mutual (Mode [5:7] not 000b)		
Auth, Mutual (Mode [5:7] not 000b) , with Key Usage[(5)]		14.4 milliseconds
AuthCheck (Mode [5:7] = 000b)		
AuthCheck (Mode [5:7] not 000b)		
AuthCheck (Mode [5:7] not 000b), with Key Usage[(5)]		
AuthCompute (Mode [5:7] = 000b)		
AuthCompute (Mode [5:7] not 000b)		
AuthCompute (Mode [5:7] not 000b), with Key Usage[(5)]		
BlockRead, 32 bytes	0.8 milliseconds	1.1 milliseconds
Counter, Read, without MAC	0.5 milliseconds	
Counter, Read, with OutMAC (Mode [5:7] = 000b)	0.7 milliseconds	
Counter, Read, with OutMAC (Mode [5:7] not 000b)		10 milliseconds
Counter, Read, with OutMAC (Mode [5:7] not 000b), with Key Usage[(5)]		
Counter, Increment, without MAC	1.5 milliseconds	
Counter, Increment, with InMAC (Mode [5:7] = 000b)	1.7 milliseconds	
Counter, Increment, with InMAC (Mode [5:7] not 000b)		10 milliseconds
Counter, Increment, with InMAC (Mode [5:7] not 000b), with Key Usage[(5)]		
Crunch, with Count 0x0001	0.85 milliseconds	1.14 milliseconds
DecRead (Mode [5:7] = 000b)		
DecRead (Mode [5:7] not 000b)		
DecRead (Mode [5:7] not 000b), with Key Usage[(5)]		
Decrypt, 1 to 16 bytes (Mode [5:7] = 000b)	2.2 milliseconds	
Decrypt, 1 to 16 bytes (Mode [5:7] not 000b)		
Decrypt, 1 to 16 bytes (Mode [5:7] not 000b), with Key Usage[(5)]		10 milliseconds

Command description	Typical time[2]	Maximum time[3]
Decrypt, 17 to 32 bytes (Mode [5:7] = 000b)	2.9 milliseconds	
Decrypt, 17 to 32 bytes (Mode [5:7] not 000b)		
Decrypt, 17 to 32 bytes (Mode [5:7] not 000b), with Key Usage[5]		14.2 milliseconds
EncRead, 1 to 16 bytes (Mode [5:7] = 000b)	2.3 milliseconds	
EncRead, 1 to 16 bytes (Mode [5:7] not 000b)		
EncRead, 1 to 16 bytes (Mode [5:7] not 000b), with Key Usage[5]		
EncRead, 17 to 32 bytes (Mode [5:7] = 000b)		
EncRead, 17 to 32 bytes (Mode [5:7] not 000b)		
EncRead, 17 to 32 bytes (Mode [5:7] not 000b), with Key Usage[5]		14.4 milliseconds
EncRead, Configuration Memory Signature Generation Mode		
EncRead, Key Memory Signature Generation Mode		
Encrypt, 1 to 16 bytes (Mode [5:7] = 000b)	2.3 milliseconds	
Encrypt, 1 to 16 bytes (Mode [5:7] not 000b)		
Encrypt, 1 to 16 bytes (Mode [5:7] not 000b), with Key Usage[5]		
Encrypt, 17 to 32 bytes (Mode [5:7] = 000b)	2.8 milliseconds	
Encrypt, 17 to 32 bytes (Mode [5:7] not 000b)		
Encrypt, 17 to 32 bytes (Mode [5:7] not 000b), with Key Usage[5]		
EncWrite, 1 to 16 bytes (Mode [5:7] = 000b)	4.1 milliseconds	
EncWrite, 1 to 16 bytes (Mode [5:7] not 000b)		
EncWrite, 1 to 16 bytes (Mode [5:7] not 000b), with Key Usage[5]		
EncWrite, 17 to 32 bytes (Mode [5:7] = 000b)		
EncWrite, 17 to 32 bytes (Mode [5:7] not 000b)		
EncWrite, 17 to 32 bytes (Mode [5:7] not 000b), with Key Usage[5]		
Info		
KeyCompute, without RNG Seed Update. (Mode [5:7] = 000b)		
KeyCompute, without RNG Seed Update. (Mode [5:7] not 000b)		
KeyCompute, without RNG Seed Update. (Mode [5:7] not 000b), with Key Usage[5]		
KeyCompute, with RNG Seed Update. (Mode [5:7] = 000b)		
KeyCompute, with RNG Seed Update. (Mode [5:7] not 000b)		26 milliseconds
KeyCompute, with RNG Seed Update. (Mode [5:7] not 000b), with Key Usage[5]		
KeyExport, without RNG Seed Update. (Mode [5:7] = 000b)		
KeyExport, without RNG Seed Update. (Mode [5:7] not 000b)		
KeyExport, without RNG Seed Update. (Mode [5:7] not 000b), with Key Usage[5]		
KeyExport, with RNG Seed Update. (Mode [5:7] = 000b)		
KeyExport, with RNG Seed Update. (Mode [5:7] not 000b)		
KeyExport, with RNG Seed Update. (Mode [5:7] not 000b), with Key Usage[5]		
KeyImport (Mode [5:7] = 000b)		
KeyImport (Mode [5:7] not 000b)		
KeyImport (Mode [5:7] not 000b), with Key Usage[5]		

Command description	Typical time[2]	Maximum time[3]
KeyLoad (Mode [5:7] = 000b)		
KeyLoad (Mode [5:7] not 000b)		
KeyLoad (Mode [5:7] not 000b), with Key Usage[5]		
KeyTransfer		
Legacy		
Legacy, with Key Usage[5]		
Lock, without MAC (Mode [5:7] = 000b)		
Lock, without MAC (Mode [5:7] not 000b)		
Lock, without MAC (Mode [5:7] not 000b), with Key Usage[5]		
Lock, with MAC (Mode [5:7] = 000b)		
Lock, with MAC (Mode [5:7] not 000b)		
Lock, with MAC (Mode [5:7] not 000b), with Key Usage[5]		
Nonce, Inbound		
Nonce, Random, without RNG Seed Update	0.5 milliseconds	
Nonce, Random, with RNG Seed Update	7.5 milliseconds	11 milliseconds
NonceCompute		
Random, without RNG Seed Update	0.5 milliseconds	
Random, without RNG Seed Update	7.0 milliseconds	11 milliseconds
Reset [4]		
Sleep, enter Standby State [4]		
Sleep, enter Sleep State [4]		
TempSense	80 milliseconds	145 milliseconds
WriteCompute, 1 to 16 Bytes (Mode [5:7] not 000b)		
WriteCompute, 1 to 16 Bytes (Mode [5:7] not 000b)		
WriteCompute, 1 to 16 Bytes (Mode [5:7] not 000b), with Key Usage[5]		
WriteCompute, 17 to 32 Bytes (Mode [5:7] = 000b)		
WriteCompute, 17 to 32 Bytes (Mode [5:7] not 000b)		
WriteCompute, 17 to 32 Bytes (Mode [5:7] not 000b), with Key Usage[5]		

Notes:
1. The values in this table are based on characterization and/or simulation. These parameters are not tested.
2. The typical response time is the time required for 60% of devices to place a packet in the response memory buffer and change the WIP STATUS bit to 0b after successful execution of the command at room temperature. If an error occurs, the response will be available in a shorter amount of time.
3. The maximum response time is the time required for 95% of devices to place a packet in the response memory buffer and change the WIP STATUS bit to 0b after successful execution of the command at the worst case temperature.

   Note: 5 % of the devices may be slower than this number. The Host is expected to read the STATUS register to determine when a response is available (see Appendix G).
2. The reset command and the sleep command do not generate a response. The response times are the time required for the operation to be completed.
3. These times are with the key usage limits enabled in the KeyConfig register. All other times are with the key usage limits disabled in the KeyConfig register.

All values are preliminary and will be updated after characterization.

Appendix O. Default Configuration

The ATAES132 memory map is shown in Table O-1 with the default memory values. Reserved memory cannot be written or read.

Table O-47. The Atmel ATAES132 memory map showing the default memory contents

Byte Address	Description
0000_h-$0FFF_h$	User memory [Default = All Bytes FF_h]
1000_h-$EFFF_h$	Reserved
$F000_h$-$F1FF_h$	Configuration memory [See Section O.2 for Default Values]
$F200_h$-$F2FF_h$	Key memory [See Section O.3 for Default Values]
$F300_h$-$FDFF_h$	Reserved
$FE00_h$	Command / response memory buffer
$FE01_h$-$FFFD_h$	Reserved
$FFE0_h$	IO address reset
$FFE1_h$-$FFEF_h$	Reserved
$FFF0_h$	STATUS register
$FFF1_h$-$FFFF_h$	Reserved

O.2. Configuration Memory Contents

The default contents of the configuration memory after completion of production test are shown in Table O-2. This configuration enables most functions and is expected to be changed by the customer during personalization. See Appendix E for the configuration memory map.

Table O-48. Default configuration memory contents. All register values shown are hexadecimal numbers.

Address	0_h / 8_h	1_h / 9_h	2_h / A_h	3_h / B_h	4_h / C_h	5_h / D_h	6_h / E_h	7_h / F_h
$F000_h$-$F007_h$	Unique Die Serial Number							
$F008_h$-$F00F_h$	Atmel proprietary data							
$F010_h$-$F017_h$	00	1F	Atmel proprietary data			00	00	20
$F018_h$-$F01F_h$	20	20	0A	Atmel proprietary data				
$F020_h$-$F027_h$	55	55	55	Atmel proprietary data				
$F028_h$-$F02F_h$	Atmel proprietary data			EE	00	03	Atmel data	
$F030_h$-$F037_h$	Atmel proprietary data							
$F038_h$-$F03F_h$								
$F040_h$-$F047_h$	I^2CAddr	C3	TempCal	TempOffset value				
$F048_h$-$F04F_h$	TempOffset value			FF	FF	FF	FF	FF
$F050_h$-$F057_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F058_h$-$F05F_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F060_h$-$F067_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F068_h$-$F06F_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F070_h$-$F077_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F078_h$-$F07F_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F080_h$-$F087_h$	00	00	00	00	FF	FF	FF	FF
$F088_h$-$F08F_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F090_h$-$F097_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F098_h$-$F09F_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F0A0_h$-$F0A7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F0A8_h$-$F0AF_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F0B0_h$-$F0B7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F0B8_h$-$F0BF_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F0C0_h$-$F0C7_h$	00	FF	FF	FF	00	FF	FF	FF
$F0C8_h$-$F0CF_h$	00	FF	FF	FF	00	FF	FF	FF
$F0D0_h$-$F0D7_h$	00	FF	FF	FF	00	FF	FF	FF
$F0D8_h$-$F0DF_h$	00	FF	FF	FF	00	FF	FF	FF
$F0E0_h$-$F0E7_h$	00	FF	FF	FF	00	FF	FF	FF
$F0E8_h$-$F0EF_h$	00	FF	FF	FF	00	FF	FF	FF
$F0F0_h$-$F0F7_h$	00	FF	FF	FF	00	FF	FF	FF
$F0F8_h$-$F0FF_h$	00	FF	FF	FF	00	FF	FF	FF
$F100_h$-$F107_h$	FF	FF	00	00	00	00	00	00
$F108_h$-$F10F_h$	FF	FF	00	00	00	00	00	00
$F110_h$-$F117_h$	FF	FF	00	00	00	00	00	00

Address	0_h / 8_h	1_h / 9_h	2_h / A_h	3_h / B_h	4_h / C_h	5_h / D_h	6_h / E_h	7_h / F_h
$F118_h$-$F11F_h$	FF	FF	00	00	00	00	00	00
$F120_h$-$F127_h$	FF	FF	00	00	00	00	00	00
$F128_h$-$F12F_h$	FF	FF	00	00	00	00	00	00
$F130_h$-$F137_h$	FF	FF	00	00	00	00	00	00
$F138_h$-$F13F_h$	FF	FF	00	00	00	00	00	00
$F140_h$-$F147_h$	FF	FF	00	00	00	00	00	00
$F148_h$-$F14F_h$	FF	FF	00	00	00	00	00	00
$F150_h$-$F157_h$	FF	FF	00	00	00	00	00	00
$F158_h$-$F15F_h$	FF	FF	00	00	00	00	00	00
$F160_h$-$F167_h$	FF	FF	00	00	00	00	00	00
$F168_h$-$F16F_h$	FF	FF	00	00	00	00	00	00
$F170_h$-$F177_h$	FF	FF	00	00	00	00	00	00
$F178_h$-$F17F_h$	FF	FF	00	00	00	00	00	00
$F180_h$-$F187_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F188_h$-$F18F_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F190_h$-$F197_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F198_h$-$F19F_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1A0_h$-$F1A7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1A8_h$-$F1AF_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1B0_h$-$F1B7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1B8_h$-$F1BF_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1C0_h$-$F1C7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1C8_h$-$F1CF_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1D0_h$-$F1D7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1D8_h$-$F1DF_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1E0_h$-$F1E7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1E8_h$-$F1EF_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1F0_h$-$F1F7_h$	FF	FF	FF	FF	FF	FF	FF	FF
$F1F8_h$-$F1FF_h$	FF	FF	FF	FF	FF	FF	FF	FF

The configuration memory map in Table O-2 is color coded. The registers shown in orange are locked at the factory and cannot be changed by the customer. The contents of the Lock registers (shown in blue) can only be changed by using the Lock command (see Section 7.19).

Configuration registers shaded with green can be written by the customer prior to locking (by setting LockConfig to 0x00 using the Lock command). The SmallZone (shown in yellow) can be written by the customer prior to locking (by setting LockSmall to 0x00 using the Lock command); SmallZone is locked separately from the remainder of the configuration memory.

The default value of the I^{2}CAddr register is 0x01 for devices configured for I^{2}C Interface mode. The default value of I^{2}CAddr is 0x00 for devices configured for SPI Interface mode. See Appendix Q for ordering codes.

The default value of the TempCal register and TempOffset register depend on the temperature sensor procedure performed by Atmel at product test. See Sections E.2.17 and E.2.18 for additional information.

O.3. Key Memory Contents

The key memory contains pseudorandom values after completion of production test, except for key 00 which contains the transport key. Device personalization can be performed without knowledge of the Transport Key, however, secure personalization can only be performed if the transport key value has been obtained from Atmel.

Appendix P. Serial Memory Backward Compatibility

The ATAES132 Secure Serial EEPROM architecture was developed to allow security to be retrofitted into systems using standard Atmel Serial EEPROM. The ATAES132 package pinouts, the interface protocol, and the command set are all compatible with standard I^{2}C and SPI EEPROM, but are not identical.

This section describes the differences which must be considered when the ATAES132 is inserted into systems using I^{2}C or SPI Serial EEPROM.

P.1. I^{2}C Serial EEPROM Compatibility

This section describes differences between the Atmel AT24C32C standard Atmel 32K bit I^{2}C Serial EEPROM and the ATAES132 Secure Serial EEPROM configured for I^{2}C communication mode.

P.1.1. Package Pins

On AT24C32C pins 1, 2, and 3 are used to set I^{2}C device address bits A_0, A_1, and A_2. AT24C32C pin 7 is the write protect (WP) input.

On ATAES132 pins 1, 2, 3, and 7 are not used in I^{2}C communication mode. These pins should be tied to V_{CC} or V_{SS}. The state of these four pins has no impact on the functionality of the ATAES132 in the I^{2}C communication mode. See Section J.2 for the pin descriptions.

P.1.2. I^{2}C Device Address

The AT24C32C I^{2}C device address is $1010A_2A_1A_0$b, with A_0, A_1, and A_2 determined by the state of pins one, two, and three. A maximum of eight AT24C32C devices are permitted on the I^{2}C interface.

On the ATAES132, the I^{2}C device address is determined by the contents of the I^{2}CAddr register (see Section J.1.3). The ATAES132 I^{2}C device address can be any set to any value, allowing up to 127 devices on the I^{2}C interface.

P.1.3. Write Protect

The AT24C32C write protect (WP) input pin inhibits all EEPROM write operations when the WP pin is high. If WP is low, then EEPROM write operations are allowed.

On the ATAES132, the user memory write permissions are controlled by the ZoneConfig Registers (see Section E.2.22). The user memory is divided into 16 user zones which are independently controlled by 16 ZoneConfig Registers – different write permissions can be assigned to different sections of the memory. By default all user memory has open write access.

P.1.4. Page Write Operations

If the host attempts to write data across the physical (32 byte) EEPROM page boundary, the AT24C32C wraps to the beginning of the EEPROM page where the page write operation begins and performs the EEPROM write after receiving a STOP condition. If the host attempts to write more than 32 bytes in a page write operation, then the AT24C32C wraps the data at the page boundary and performs the EEPROM write after receiving a STOP condition. Partial page writes are supported by the AT24C32C.

The ATAES132 does not allow write operations to cross physical (32 byte) EEPROM page boundaries (see Section B.2), and does not allow a write operation if more than 32 data bytes are received from the host. In both cases the EEPROM contents remain unchanged, the data is discarded, and an error bit is set in the STATUS register (see Section J.3.3). Partial page writes are supported by the ATAES132.

P.1.5. Read Operations

Reading beyond the end of physical memory on the AT24C32C causes the internal data address register to rollover to address zero. The read operation continues from address zero.

If an ATAES132 read operation begins at a valid user memory address but continues past the end of user memory, the read operation will not wrap to the beginning of user memory. Reading beyond the end of user memory causes 0xFF to be returned to the host in reply to the read, the internal data address register stops incrementing, and an error bit is set in the STATUS register (see Section G.2.5).

P.1.6. Read Protect

The AT24C32C and other standard I^2C EEPROM do not have a read inhibit function.

On the ATAES132, the user memory read permissions are controlled by the ZoneConfig Registers (see Section E.2.22). The user memory is divided into 16 user zones which are independently controlled by 16 ZoneConfig Registers – different read permissions can be assigned to different sections of the memory. If read access is prohibited, then 0xFF will be returned to the host in reply to a read command (see Section 5.2). By default all user memory has open read access.

P.1.7. Standby Mode

Standard I^2C EEPROM automatically enter low power standby mode upon completion of any internal operation.

The ATAES132 has three powered states: the active state and two low power states, the standby state and the sleep state. The ATAES132 will remain in the active state between operations unless the host sends a sleep command to activate the standby state or the sleep state. The ATAES132 can also be configured to automatically enter a low power state at power up. See Appendix L for details on the power management features.

P.1.8. Operating Voltage

The AT24C32C operating voltage range is 1.8V minimum to 5.5 Vmaximum.

The ATAES132 operating voltage range is 2.5V minimum to 5.5 Vmaximum. See Section 9.3 for DC specifications.

P.2. SPI Serial EEPROM Compatibility

This section describes differences between the AT25320B standard Atmel 32K bit SPI Serial EEPROM and the ATAES132 Secure Serial EEPROM configured for SPI communication mode.

P.2.1. Package Pins

On AT25320B pin three is the $\overline{\text{WP}}\overline{\textbf{WP}}$ input, and pin seven is the $\overline{\text{HOLD}}\overline{\textbf{HOLD}}$ input.

On ATAES132 pins three and seven are not used in SPI communication mode – these pins can be tied to V_{CC} or V_{SS}. The state of these two pins with no impact on the functionality of the ATAES132 in the SPI communication mode. See Section K.2 for the pin descriptions.

P.2.2. Write Protect ($\overline{\textbf{WP}}$)

The AT25320B $\overline{\textbf{WP}}$ input pin inhibits all EEPROM write operations when the WP pin is low. If WP is high, then EEPROM write operations are allowed. The write protect pin can be disabled by writing the WPEN bit in the STATUS register to 0b.

On the ATAES132, the user memory write permissions are controlled by the ZoneConfig Registers (see Section E.2.22). The user memory is divided into 16 user zones which are independently controlled by 16 ZoneConfig Registers – different write permissions can be assigned to different sections of the memory. By default, all user memory has open write access.

P.2.3. HOLD

The AT25320B $\overline{\text{HOLD}}$ input pin allows the host to pause communication with the memory temporarily (by bringing $\overline{\text{HOLD}}$ low), and then resume the communication sequence (by bringing $\overline{\text{HOLD}}$ high). The sequence continues exactly from the point where it was paused, as if there was no interruption.

The ATAES132 does not have a HOLD function. If communications are interrupted, the sequence must be restarted, beginning with a high to low transition on the $\overline{\text{CS}}\overline{\text{CS}}$ input.

P.2.4. Page Write Operations

If the host attempts to write data across the physical (32 byte) EEPROM page boundary, the AT25320B wraps to the beginning of the EEPROM page where the page write operation begin and performs the EEPROM write after receiving a low to high transition on the $\overline{\text{CS}}$ input. If the host attempts to write more than 32 bytes in a page write operation, then the AT25320B wraps the data at the page boundary and performs the EEPROM write after receiving a STOP condition. Partial page writes are supported by the AT25320B.

The ATAES132 does not allow write operations to cross physical (32 byte) EEPROM page boundaries (see Section B.2), and does not allow a write operation if more than 32 data bytes are received from the host. In both cases, the EEPROM contents remain unchanged, the data is discarded, and an error bit is set in the STATUS register (see Section J.3.3). Partial page writes are supported by the ATAES132.

P.2.5. Read Operations

Reading beyond the end of physical memory on AT25320B causes the internal data address register to rollover to address zero. The read operation continues from address zero.

If an ATAES132 read operation begins at a valid user memory address but continues past the end of user memory, the read operation will not wrap to the beginning of user memory. Reading beyond the end of user memory causes 0xFF to be returned to the host in reply to the read, the internal data address register stops incrementing, and an error bit is set in the STATUS register.

P.2.6. Read Protect

The Atmel AT25320B and other standard SPI EEPROM do not have a read inhibit function.

On the ATAES132, the user memory read permissions are controlled by the ZoneConfig Registers (see Section E.2.22). The user memory is divided into 16 user zones which are independently controlled by 16 ZoneConfig Registers – different read permissions can be assigned to different sections of the memory. If read access is prohibited, then 0xFF will be returned to the host in reply to a read command (see Section 5.2). By default all user memory has open read access.

P.2.7. STATUS Register

The AT25320B STATUS register definition is shown in Table P-1. The default state of all STATUS bits is 0b. The WPEN bit controls the write protect pin. Block write protection is controlled by the BP0 and BP1 bits. If WEN = 1b, then the device is write enabled. If WIP = 0b, the device is ready to accept a command – WIP = 1b indicates a write cycle is in progress. The Reserved bits are 0b except when an internal write cycle is in progress. All bits of the STATUS register are 1b when an internal write cycle is in progress.

Table P-49. Atmel AT25320B STATUS register fefinition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPEN	Reserved	Reserved	Reserved	BP1	BP0	WEN	WIP

The ATAES132 STATUS register definition is shown in Table P-2 and described in Appendix G. The default state of all STATUS bits is 0b. The WEN, WIP, and reserved bits are similar to standard SPI Serial EEPROM: If WEN = 1b, then the device is write enabled. If WIP = 0b, the device is ready to accept a command; WIP = 1b indicates a write cycle or a cryptographic operation is in progress. The reserved bits are 0b except when an internal write cycle or a cryptographic operation is in progress. All bits of the STATUS register are 1b when an internal write cycle or a cryptographic operation is in progress.

Table P-50.   Atmel ATAES132 STATUS register definition

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EERR	RRDY	Reserved	CRCE	Reserved	WAKEb	WEN	WIP

ATAES132 reports errors to the host using the EERR and CRCE bits. The RRDY bit indicates if the Response Memory Buffer is empty (0b), or ready to read (1b). The WAKEb bit indicates if the device is in the Sleep or Standby Power State. See Section G.1 for detailed descriptions of each STATUS bit.

P.2.8. Write Status Register Command (WRSR)

The AT25320B STATUS register contains three bits which control the block write protect function, and the write protect pin. These bits can be changed by sending a write status register (WRSR) command to the memory.

The ATAES132 does not support the write status register (WRSR) command. The WRSR command will be ignored if it is received.

P.2.9. Block Write Protect

The AT25320B STATUS register contains two block protect bits (BP0 and BP1) which control the block write protect function. By writing the STATUS register the user can set the block protect bits to inhibit writes in ¼, ½, or the full memory array.

On the ATAES132, the user memory write permissions are controlled by the ZoneConfig Registers (see Section E.2.22). The user memory is divided into 16 user zones which are independently controlled by 16 ZoneConfig Registers – different write permissions can be assigned to different sections of the memory. By default all user memory has open write access.

P.2.10. Standby Mode

Standard SPI EEPROM automatically enter low power standby mode upon completion of any internal operation.

The ATAES132 has three powered states: the active state and two low power states, the standby state and the sleep state. The ATAES132 will remain in the active state between operations unless the host sends a sleep command to activate the standby state or the sleep state. The ATAES132 can also be configured to automatically enter a low power state at power up. See Appendix L for details on the power management features.

P.2.11. Operating Voltage

The AT25320B operating voltage range is 1.8V minimum to 5.5V maximum.

The ATAES132 operating voltage range is 2.5V minimum to 5.5V maximum. See Section 9.3 for DC specifications.

P.2.12. Maximum Operating Frequency

The AT25320B maximum SCK frequency is 10MHz when V_{CC} is 2.7 V to 5.5 V. The maximum SCK frequency is 20MHz when V_{CC} is 4.5 V to 5.5 V.

The ATAES132 maximum SCK frequency is 10MHz when V_{CC} is 2.5 V to 5.5 V. See Section 9.4 for AC specifications.

Appendix Q. Ordering Information

The ATAES132 production ordering codes are listed in Section R.1. To increase security ATAES132 packages are not marked with the ordering code, the ATAES132 standard packages are marked with a trace code which is unique for each manufacturing lot. Contact Atmel for additional information.

Atmel Ordering Codes

Atmel ordering code	Package type	Interface configuration	Temperature range
ATAES132-SH-EQ	SOIC, Bulk	SPI	-40 °C to 85 °C
ATAES132-SH-ER	SOIC, Bulk	I^2C	-40 °C to 85 °C
ATAES132-SH-FB	SOIC, Bulk	I^2C, AuthO	-40 °C to 85 °C
ATAES132-SH-EQ-T	SOIC, Tape & Reel	SPI	-40 °C to 85 °C
ATAES132-SH-ER-T	SOIC, Tape & Reel	I^2C	-40 °C to 85 °C
ATAES132-SH-FB-T	SOIC, Tape & Reel	I^2C, AuthO	-40 °C to 85 °C
ATAES132-TH-EQ	TSSOP, Bulk	SPI	-40 °C to 85 °C
ATAES132-TH-ER	TSSOP, Bulk	I^2C	-40 °C to 85 °C
ATAES132-TH-FB	TSSOP, Bulk	I^2C, AuthO	-40 °C to 85 °C
ATAES132-TH-EQ-T	TSSOP, Tape & Reel	SPI	-40 °C to 85 °C
ATAES132-TH-ER-T	TSSOP, Tape & Reel	I^2C	-40 °C to 85 °C
ATAES132-TH-FB-T	TSSOP, Tape & Reel	I^2C, AuthO	-40 °C to 85 °C
ATAES132-MA3H-EQ-T	UDFN, Tape & Reel	SPI	-40 °C to 85 °C
ATAES132-MA3H-ER-T	UDFN, Tape & Reel	I^2C	-40 °C to 85 °C
ATAES132-MA3H-FB-T	UDFN, Tape & Reel	I^2C, AuthO	-40 °C to 85 °C

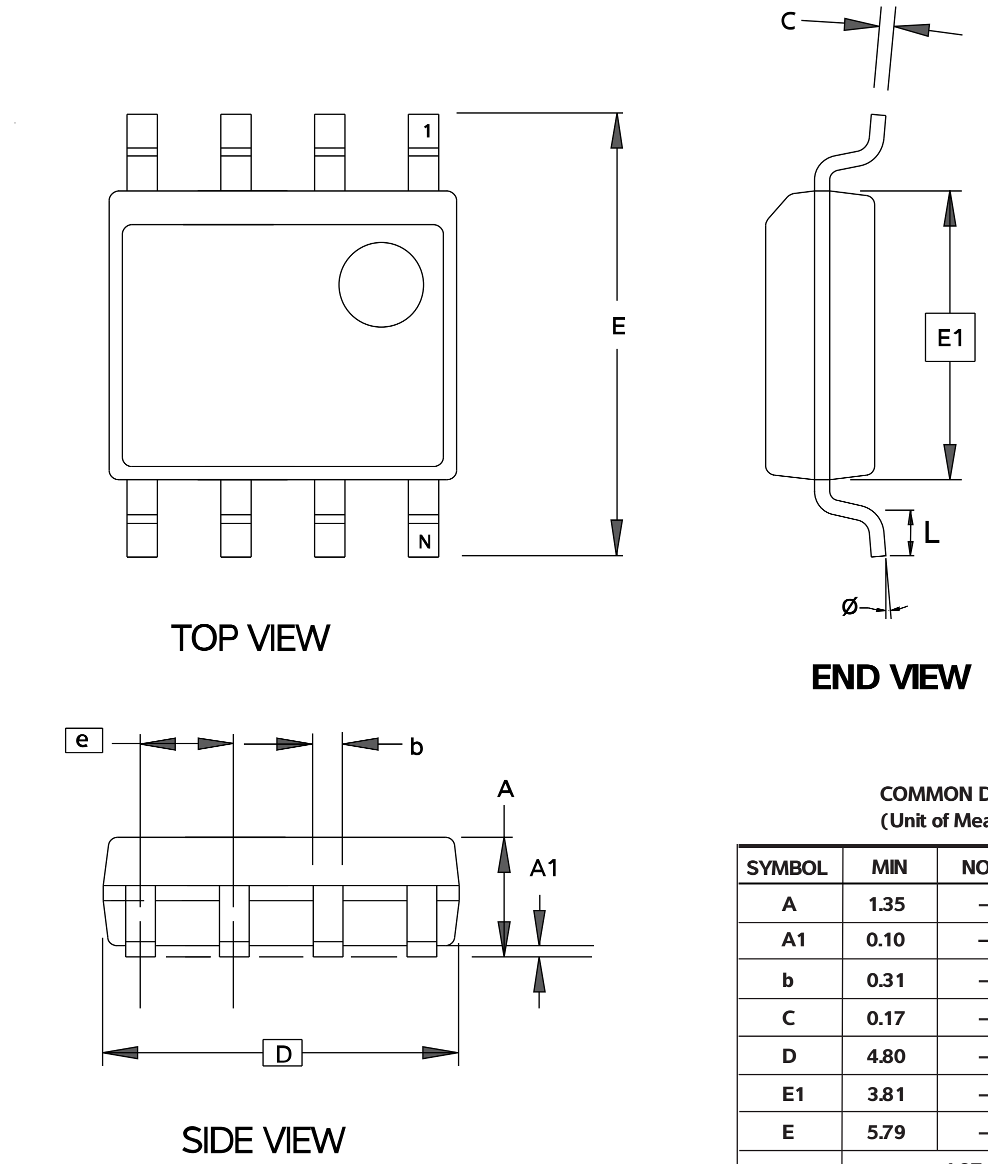
Note: 1. AuthO indicates device supports Auth Signaling. See Section J.5

Package type	Description
SOIC	8 pin SOIC, NiPdAu lead finish, Green(1)
TSSOP	8 pin TSSOP, NiPdAu lead finish, Green(1)
UDFN	8 pin UDFN/USON 2 x 3mm, NiPdAu lead finish, Green(1)

Note: 1. Lead-free, halogen-free package. Exceeds RoHS requirements

Q.1. Mechanical Information

8S1 – JEDEC SOIC



Notes: This drawing is for general information only. Refer to JEDEC Drawing MS-012, Variation AA for proper dimensions, tolerances, datums, etc.

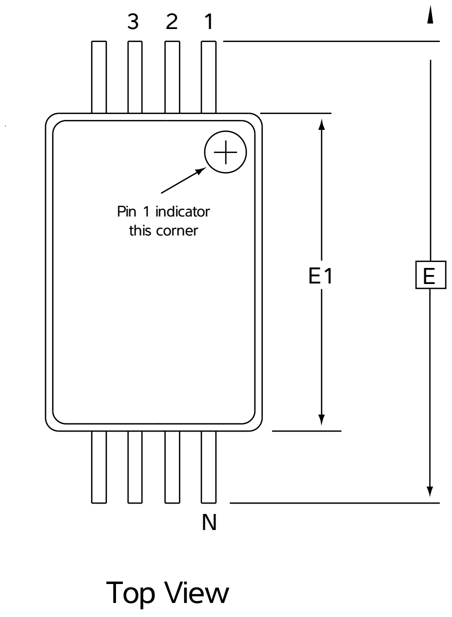
COMMON DIMENSIONS
(Unit of Measure = mm)

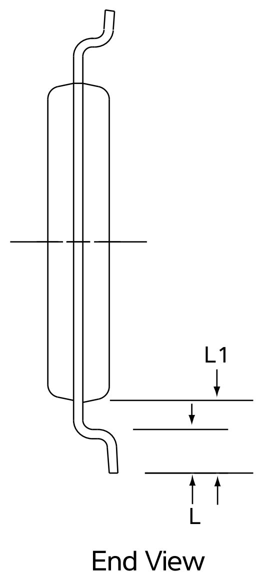
SYMBOL	MIN	NOM	MAX	NOTE
A	1.35	–	1.75	
A1	0.10	–	0.25	
b	0.31	–	0.51	
C	0.17	–	0.25	
D	4.80	–	5.05	
E1	3.81	–	3.99	
E	5.79	–	6.20	
e	1.27 BSC			
L	0.40	–	1.27	
Ø	0°	–	8°	

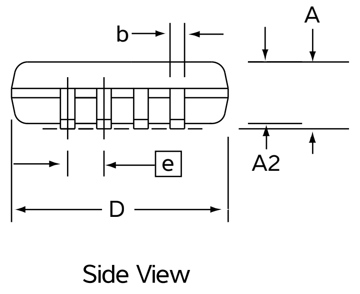
5/19/10

	TITLE	GPC	DRAWING NO.	REV.
Package Drawing Contact: packagedrawings@atmel.com	8S1, 8-lead (0.150" Wide Body), Plastic Gull Wing Small Outline (JEDEC SOIC)	SWB	8S1	F

8A2 – TSSOP







COMMON DIMENSIONS
(Unit of Measure = mm)

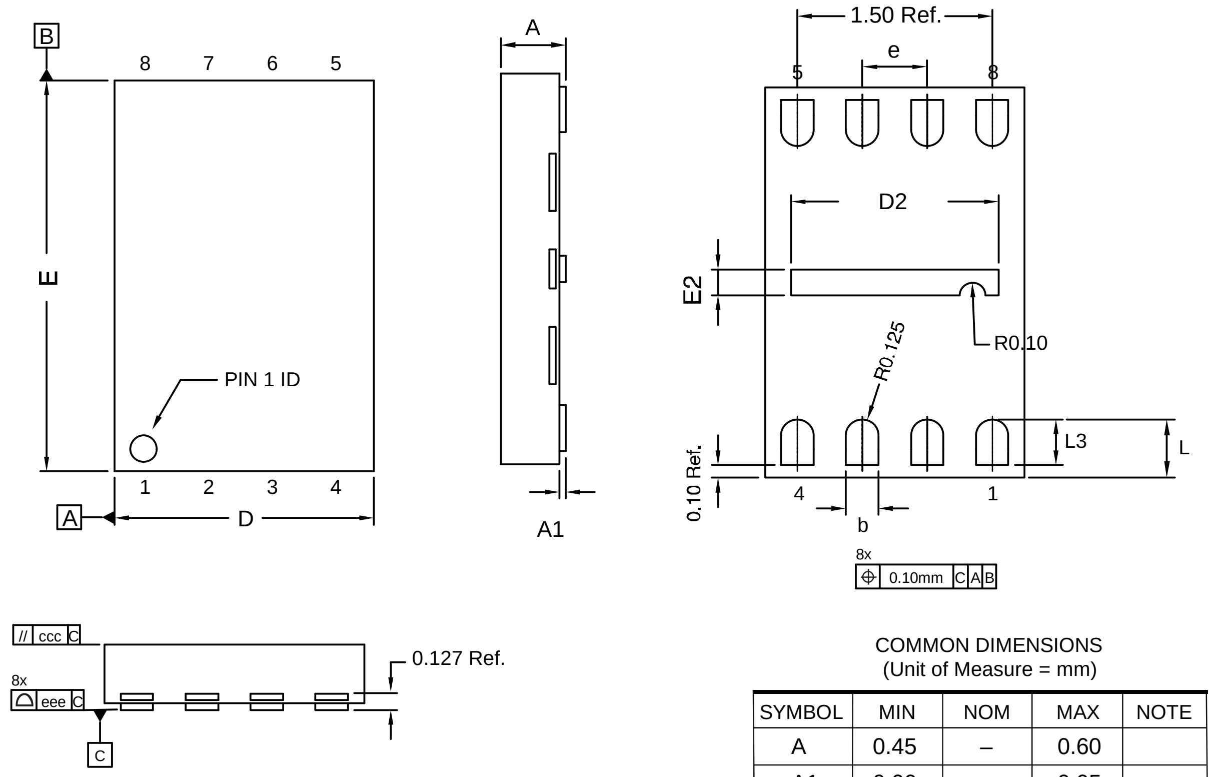
SYMBOL	MIN	NOM	MAX	NOTE
D	2.90	3.00	3.10	2, 5
E	6.40 BSC			
E1	4.30	4.40	4.50	3, 5
A	–	–	1.20	
A2	0.80	1.00	1.05	
b	0.19	–	0.30	4
e		0.65 BSC		
L	0.45	0.60	0.75	
L1		1.00 REF		

Notes:
1. This drawing is for general information only. Refer to JEDEC Drawing MO-153, Variation AA, for proper dimensions, tolerances, datums, etc.
2. Dimension D does not include mold Flash, protrusions or gate burrs. Mold Flash, protrusions and gate burrs shall not exceed 0.15mm (0.006in) per side.
3. Dimension E1 does not include inter-lead Flash or protrusions. Inter-lead Flash and protrusions shall not exceed 0.25mm (0.010in) per side.
4. Dimension b does not include Dambar protrusion. Allowable Dambar protrusion shall be 0.08mm total in excess of the b dimension at maximum material condition. Dambar cannot be located on the lower radius of the foot. Minimum space between protrusion and adjacent lead is 0.07mm.
5. Dimension D and E1 to be determined at Datum Plane H.

5/19/10

Package Drawing Contact: packagedrawings@atmel.com	TITLE 8A2, 8-lead 4.4mm Body, Plastic Thin Shrink Small Outline Package (TSSOP)	GPC TNR	DRAWING NO. 8A2	REV. E

8MA3 – UDFN/USON Ultra Thin No Lead Package



COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.45	–	0.60	
A1	0.00	–	0.05	
B	0.20	–	0.30	
D	1.95	2.00	2.05	
D2	1.50	1.60	1.70	
E	2.95	3.00	3.05	
E2	0.10	0.20	0.30	
E	–	0.50	–	
L	0.40	0.45	0.50	
L3	0.30	–	0.40	
ccc	–	–	0.05	
eee	–	–	0.05	

Notes:
1. All dimensions are in mm. Angles in degrees.
2. Coplanarity applies to the exposed pad as well as the terminals. Coplanarity shall not exceed 0.05 mm.
3. Warpage shall not exceed 0.05 mm.
4. Package length/package width are considered as special characteristic.
5. Refer to Jede MO-236/MO-252

8/8/08

	TITLE	GPC	DRAWING NO.	REV.
Package Drawing Contact: packagedrawings@atmel.com	8MA3, 8-pad, 2 x 3 x 0.6 mm Body, 0.5 mm Pitch, 1.6 x 0.2 mm Exposed Pad, Saw Singulated Thermally Enhanced Plastic Ultra Thin Dual Flat No Lead Package (UDFN/USON)	YCQ	8MA3	A

Appendix R. Errata

R.1. First Silicon Errata (INFO DeviceNum = 0x0A01 or 0x0A02)

Pre-production version is not suitable for qualification. First silicon is not fully tested and may not meet all of the parametric specifications listed in Section 9.

R.1.1. Memory Contents

EEPROM contents are not identical to the production configuration described in Appendix O. This is intentional. The first silicon can be easily distinguished from later revisions.

R.1.2. Configuration Memory Map Change

The ManufacturingID register and PermConfig register location in first silicon are different than in later revisions. In later revisions the PermConfig register is at address 0xF02D, and the ManufacturingID register is at address 0xF02B to 0xF02C.

The TempCal register did not exist in first silicon. The TempOffset register was two bytes in first silicon, but was expanded to accommodate improved temperature sensor calibration procedures.

Table R-51. Partial configuration memory map for the Atmel ATAES132 first silicon

Address	0$_h$ / 8$_h$	1$_h$ / 9$_h$	2$_h$ / A$_h$	3$_h$ / B$_h$	4$_h$ / C$_h$	5$_h$ / D$_h$	6$_h$ / E$_h$	7$_h$ / F$_h$
F000$_h$-F007$_h$	SerialNum							
F008$_h$-F00F$_h$	LotHistory							
F010$_h$-F017$_h$	Jedec		ManufacturingID		PermConfi	Algorithm		EEPage
F018$_h$-F01F$_h$	EncRead	EncWrtSiz	DeviceNum	Reserved				
F020$_h$-F027$_h$	LockKeys	LockSmall	LockConfig	Reserved				
F028$_h$-F02F$_h$	Reserved							
F030$_h$-F037$_h$								
F038$_h$-F03F$_h$								
F040$_h$-F047$_h$	I^{2}CAddr	ChipConfi	TempOffset		RFU			
F048$_h$-F04F$_h$	RFU							

R.1.3. SPI Write Output State

The SO output pin should always be in the high impedance state during an SPI Write operation. Actual behavior is the SO pin is in the high impedance state if no errors are detected in the command packet; however, if the packet contains an error, then SO may be forced high when the error is detected. SO will remain high until the entire write command packet is clocked in and the $\overline{\text{CS}}$ input goes high.

This problem will be fixed in future revisions of ATAES132.

R.1.4. SPI Configuration Memory Write Error not Flagged

If an SPI write to the configuration memory is attempted while the device is in the write disable state then the EEPROM write will fail (as expected). The EERR bit of the STATUS register should be set to 1b to indicate an error. Actual behavior is the EEPROM does not write and the EERR status bit is 0b (erroneously indicating that no error occurred).

This problem will be fixed in future revisions of ATAES132.

R.1.5. SPI Write Enable Changed by Illegal Read

If an SPI Read is executed after the write enable flag is enabled, the write enable flag should not change. Actual behavior is an SPI read of an illegal address causes the write enable flag to change to the disabled state. The flag is not changed by SPI reads of legal addresses.

This problem will be fixed in future revisions of ATAES132.

R.1.6. I^2C Current Address Read

If a standard I^2C current address read operation is performed after a standard I^2C byte write or I^2C page write operation, the first byte returned should be the byte following the last byte clocked in by the write command. Actual behavior is the internal address register used by the standard I^2C read operations is not updated when a standard I^2C byte write or I^2C page write operation is performed, therefore, the I^2C current address read does not return the expected bytes.

This problem will be fixed in future revisions of ATAES132.

R.1.7. I^2C Device Address

In the I^2C interface mode, the device should only ACK the device address if all seven bits match the value stored in the I^2CAddr register. Actual behavior is the ATAES132 ACKs an I^2C address in which only the upper six bits match the value in the I^2CAddr register.

This problem will be fixed in future revisions of ATAES132.

R.1.8. I^2C Data In Hold Time

The minimum I^2C data in Hold time specification is 0ns. This revision of the ATAES132 requires 10ns minimum I^2C dataIn Hold time.

This problem will be fixed in future revisions of ATAES132.

R.1.9. EncWrite to Key Memory

The EncWrite command should permit key memory to be written using encrypted data as described in Section 7.11. Actual behavior is that the EncWrite command appears to function correctly when writing to key memory, however, the new contents of the key register will be incorrect. Since the new key value is unknown, all operations with the key will generate incorrect cryptographic results.

It is recommended that the key memory be written with cleartext using the standard write commands as described in Section 5.3.3 prior to locking key memory. The key memory can be updated after locking using the KeyCompute command or the KeyLoad command if the key is configured as a child key in the KeyConfig register.

This problem will be fixed in future revisions of ATAES132.

R.1.10. EERR Status bit Reset by Memory Read

If a SPI or I^2C read begins at an authorized address and continues into protected memory, the EERR bit will be set to 1b. Once set, the EERR status bit should retain the 1bs state during the read operation, regardless of how much data is read. Actual behavior is that the EERR bit will be set to 0b on the 129th byte read, causing the error information to be lost.

This problem will be fixed in future revisions of ATAES132.

R.1.11. EERR Status bit Incorrect for Memory Read Beyond End of User Memory

If a SPI or I^2C read begins at an authorized user memory address and continues beyond address 0x0FFF, the EERR bit should be set to 1b and 0xFF should be returned for each data byte. Actual behavior is 0xFF is returned for each data byte above address 0x0FFF, however, the EERR bit is 0b.

This problem will be fixed in future revisions of ATAES132.

R.1.12. RRDY Status bit Not Reset by Memory Read in I^{2}C Interface Mode

If a memory read is attempted using an invalid address, then the EERR bit will be set to 1b and the RRDY bit should be set to 0b. Actual behavior in I^{2}C Interface mode is that the RRDY bit is not reset when an invalid address read is attempted – if the RRDY bit is 1b before the read, then it remains 1b. The ATAES132 operates correctly in SPI Interface mode.

This problem will be fixed in future revisions of ATAES132.

R.1.13. I^{2}C Auth Signaling Not Supported

I^{2}C Auth signaling is not supported in this revision of the device. The SO pin is always in the high impedance state in I^{2}C interface mode.

R.1.14. Random Command Mode Not Supported

The random command does not support Mode Bit 2. This revision of the device can only generate random numbers for external use using the random command. Nonce values can only be generated with the nonce command. If the host attempts to use this mode with the ATAES132 first silicon, an error code will be generated.

R.1.15. EncRead Signature Generation Modes Not Supported

The EncRead command does not support the key memory signature generation mode or the configuration memory signature generation mode. Any attempt to read the configuration memory or key memory using the EncRead command generates an error code

R.1.16. Vcc Voltage Limitation

The absolute maximum operating voltage is 4.0V. The temperature sensor will be permanently damaged if the V_{CC} is in excess of 4.0V is applied to the device. Overvoltage damage to the temperature sensor will result in a permanent increase in the power consumption of the device.

This problem will be fixed in future revisions of ATAES132.

R.1.17. Slow V_{CC} Power Up

If the rise time of V_{CC} during power up is 100 milliseconds or more, then the ATAES132 may be configured incorrectly. The device may not load the ChipConfig register correctly, causing the device to enter the wrong power state at power up. The device may not load the I^{2}CAddr register correctly resulting in selection of the wrong interface mode or loading a random I^{2}C device address; as a result it may be impossible for the host microcontroller to communicate with the ATAES132. Selection of the wrong interface mode at power up could result in permanent damage to the ATAES132.

This problem will be fixed in future revisions of the ATAES132.

R.2. Pre-Production Errata (INFO DeviceNum = 0x0A04)

Pre-production version, not fully qualified.

R.2.1. SPI Mode 3 Not Supported

SPI Mode 3 communication is not supported in this revision of the device. This problem will be fixed in future revisions of the ATAES132.

R.2.2. Temperature Sensor is non-functional

The temperature sensor is non-functional and the TempSense command should not be executed. Execution of the TempSense command will result in the part reseting.

R.2.3. I^{2}C Maximum Command Block Length is 63 bytes

If the command block is 64 bytes or greater the device will not accept another command until the power is cycled. No command requires 64 bytes unless extra bytes are added to the end for padding. Because of the length of the DecRead, WriteCompute and KeyImport commands, they need to be sent as more than one block.

R.2.4. RRDY Status bit Not Reset by Memory Read in I^{2}C Interface Mode

If a memory read is attempted using an invalid address, then the EERR bit will be set to 1b and the RRDY bit should be set to 0b. Actual behavior in the I^{2}C interface mode is that the RRDY bit is not reset when an invalid address read is attempted – if the RRDY bit is 1b before the read, then it remains 1b. ATAES132 operates correctly in SPI Interface mode.

This problem will be fixed in future revisions of ATAES132

R.2.5. EERR and RRDY Status bits Reset by not polling during command execution

In the I^{2}C mode, if the master does not poll the slave during a command execution, including an EEPROM write, the EERR staus bit will be reset and always appear to be zero. If the last byte of the command is sent in a sepperate packet the RRDY bit is also cleared.

R.2.6. EERR and RRDY Status bits Reset by not polling during EEPROM Writes

In I^{2}C mode, if the master does not poll the slave during an EEPROM write, the EERR and RRDY staus bits will be reset and always appear to be zero.

R.2.7. KeyCompute , mode bit 2 =1b

The KeyCompute command with mode bit = 1b is not supported in this revision of the device. This option will be supported in future revisions of the ATAES132

Appendix S. Revision History

Doc. Rev.	Date	Comments
8760A	05/2011	Initial document release

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
JAPAN
Tel: (+81)(3) 3523-3551
Fax: (+81)(3) 3523-7581