Specification
Of
1KBytes UHF Band RFID chip

<table>
<thead>
<tr>
<th>DOCUMENT NUMBER:</th>
<th>REVISION NUMBER</th>
<th>RELEASE/REVISION DATE:</th>
</tr>
</thead>
<tbody>
<tr>
<td>20130531-01F</td>
<td>Rev 001</td>
<td>May 31, 2013</td>
</tr>
</tbody>
</table>
1 FEATURES

1.1 General specification

- EPCglobal C1G2 1.2.0 compliant [Note1]

1.2 Mechanical specification

- Diameter: 200mm (8inch)
- Thickness: 150μm ±10%
- Number of pads: 4
- Die size without scribe: 1460μm * 1460μm
- Bump material: > 99.9% pure Au
- Map file distribution: IBIS Format (for Muehlbauer FCM10000)

Limiting values

- Storage temperature range (T_{stg}): -55°C to +125°C
- Operating temperature (T_{oper}): -40°C to +85°C
- Electrostatic discharge voltage HBM (V_{ESD}): ±2KV

Memory characteristics

- EEPROM data retention (T_{ret}): 30 years @ 85°C
- EEPROM write endurance: 100000 @ 55°C

Interface characteristics

- Operating frequency: 860MHz to 960MHz
- Minimum operating power supply at read (P_{minr}) [Note2][Note5]: -15dbm
- Minimum operating power supply at write (P_{minw}) [Note3][Note5]: -10dbm
- Maximum working RF field strength [Note4][Note5]: 15dbm
- Absolute Limit RF field strength [Note4][Note5]: 25dbm
- Quality factor (Im(Z_{chip})/Re(Z_{chip})): < 10
- Parallel input capacitance (C_{cp}) [Note6]: 0.975pF
- Parallel input resistance (R_{cp}) [Note6]: 2.25Kohm

Memory

- EPC Bank: 496bit (Except CRC, PC)
- TID Bank: 208bit
- Reserved Bank: 64bit
- User: 1Kbyte
Substance Control Requirements

- Meet the requirement of QP7513 General Specification On Substance Control Requirements V1.00

[Note1]: EPC™ Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz – 960 MHz Version 1.2.0 Gen2
Reference doc: uhfc1g2_1_2_0-standard-20080511.pdf

[Note2]: Power to process a Query Command, Measured with a 50ohm source impedance.

[Note3]: Minimum -10dbm for Write/Kill/Lock/Blockwrite/Blockerase/Blockpermalock/ChgAreaGroupPwd/AreaReadLoc/AreaWriteLock/AreaWriteLockwoPwd commands power and -7dbm continues wave for at least 4ms after these commands send. Measured with a 50ohm source impedance.

[Note4]: Power to process all Command, Measured with a 50ohm source impedance.

[Note5]: At normal temperature range -20°C to 40°C variation less than ±0.5dbm
At extended temperature range -40°C to 65°C variation less than ±0.75dbm
At extreme temperature range -40°C to 85°C variation less than ±1dbm

[Note6]: At minimum operating power. Tested result tested with strip under flip chip.

2 TERMS AND DEFINITIONS

2.1 Terms and definitions

- The terms and definitions refer to section 4.1, Ref.1.
- Extreme temperature range -40°C to 85°C.

2.2 Symbols

- The symbols refer to section 5.1, Ref.1.

2.3 Abbreviated terms

- The abbreviated terms refer to section 5.2, Ref.1.
3 FUNCTIONAL DESCRIPTION

3.1 Block Description

The AIT-C01BCB chip consists of RF module, Analog Fronted module, baseband and EEPROM. There four PADs in the chip, Two RF pads and two dummy pads. Energy and data are transferred via an antenna. No further external components are necessary. Fig.1 shows the diagram of AIT-C01BCB.

![AIT-C01BCB chip block description](image)

3.2 Communication

The signaling interface between an Interrogator and a Tag may be viewed as the physical layer in a layered network communication system. The signaling interface defines frequencies, modulation, data coding, RF envelope, data rates, and other parameters for RF communications.
3.2.1 Operation Frequency

Tags shall receive power from and communicate with Interrogators within the frequency range from 840 MHz to 960 MHz, inclusive.

3.2.2 Interrogator-to-Tag (R=>T) communications

Interrogator-to-Tag (R=>T) communications refer to section 6.3.1.2, Ref.1.

3.2.3 Tag-to-Interrogator (T=>R) communications

Tag-to-Interrogator (T=>R) communications refer to section 6.3.1.3, Ref.1.

3.2.4 Transmission order

Transmission order refers to section 6.3.1.4, Ref.1.

3.2.5 Cyclic-redundancy check (CRC)

Cyclic-redundancy check (CRC) refers to section 6.3.1.5, Ref.1.

3.2.6 Link timing

Link timing refers to section 6.3.1.6, Ref.1.

3.3 Tag memory

Tag memory shall be logically separated into four distinct banks. Each of which may comprise zero or more memory words. A logical memory map is shown in Fig.2. The four memory banks are:

- **Reserved memory** shall contain the kill, access passwords. The kill password shall be stored at memory addresses 00\textsubscript{h} to 1F\textsubscript{h}; the access password shall be stored at memory addresses 20\textsubscript{h} to 3F\textsubscript{h}.

- **EPC memory** shall contain a CRC-16 at memory addresses 00\textsubscript{h} to 0F\textsubscript{h}, Protocol-Control (PC) bits at
memory addresses 10\textsubscript{h} to 1F\textsubscript{h} and a code (such as an EPC, and hereafter referred to as an EPC) that identifies the object to which the Tag is or will be attached beginning at address 20\textsubscript{h}. XPC option is not supported in AIT-C01BCB.

- **TID memory** shall contain an 8-bit ISO/IEC 15963 allocation class identifier at memory locations 00\textsubscript{h} to 07\textsubscript{h}. TID memory shall contain sufficient identifying information above 07\textsubscript{h} for an Interrogator to uniquely identify the custom commands and/or optional features that a Tag supports.

  **User memory** allows user-specific data storage.

  The logical addressing of all memory banks shall begin at zero (00\textsubscript{h}). Commands that access memory have a MemBank parameter that selects the bank, and an address parameter, specified using the EBV format described in reference 1 Annex A, to select a particular memory location within that bank. When Tags backscatter memory contents, this backscatter shall fall on word boundaries (except in the case of a truncated reply). MemBank is defined as follows:

  00\textsubscript{2} Reserved
  01\textsubscript{2} EPC
  10\textsubscript{2} TID
  11\textsubscript{2} User

  Operations in one logical memory bank shall not access memory locations in another bank.

  Memory writes, detailed in section 6.3.2.9 ref.1 involve the transfer of 16-bit words from Interrogator to Tag. A Write command writes 16 bits (i.e. one word) at a time, using link cover-coding to obscure the data during R=>T transmission.
Interrogators may lock, permanently lock, unlock, or permanently unlock memory, thereby preventing or allowing subsequent changes (as appropriate). The kill and access passwords are individually lockable, as are EPC, TID, and User memory. If the kill and/or access passwords are locked they are usable by only the Kill and Access commands, respectively, and are rendered both unwriteable and unreadable by any other command. The EPC, TID, and User memory banks are always readable regardless of their lock status.

### 3.3.1 Reserved Memory

Reserved memory contains the kill and access passwords.

#### 3.3.1.1 Kill password

The kill password is a 32-bit value stored in reserved memory 00h to 1Fh, MSB first. The default (unprogrammed) value shall be zero. An Interrogator shall use a kill password once, to kill the Tag and render it no responsive there after. A Tag shall not execute a kill operation if its kill password is zero. A
Tag that does not implement a kill password operates as if it has a zero-valued kill password that is permanently read/write locked.

### 3.3.1.2 Access password

The access password is a 32-bit value stored in reserved memory 20h to 3Fh, MSB first. The default (unprogrammed) value shall be zero. A Tag with a nonzero access password shall require an Interrogator to issue this password before transitioning to the secured state. A Tag that does not implement an access password operates as if it has a zero-valued access password that is permanently read/write locked.

### 3.3.2 EPC Memory

EPC memory contains a CRC-16 at memory addresses 00h to 0Fh, PC bits at memory locations 10h to 1Fh, an EPC beginning at address 20h, and EPC total up to 496 bits (exclude CRC and PC).

#### 3.3.2.1 CRC-16

The PC and EPC are protected by the CRC-16 that a Tag backscatters during an inventory operation. Because Interrogators may issue a *Select* command that includes all or part of this CRC-16 in the mask, and may issue a *Read* command to cause the Tag to backscatter this CRC-16, this CRC-16 is logically mapped into EPC memory. At power-up a Tag shall compute this CRC-16 over EPC memory location 10h to the end of the EPC (not necessarily to the end of EPC memory, but to the end of the EPC specified by the length field in the PC) and map the computed CRC-16 into EPC memory 00h to 0Fh, MSB first. Because the (PC+EPC) is stored in EPC memory on word boundaries, this CRC-16 shall be computed on word boundaries. Tags shall finish this CRC-16 computation and memory mapping by the end of
interval Ts or Ths, respectively. Tags shall not recalculate this CRC-16 for a truncated reply.

3.3.2.2 Protocol-control (PC) word

The PC word contains physical-layer information that a Tag backscatters with its EPC during an inventory operation. There are 16 PC bits, stored in EPC memory at addresses $10_{h}$ to $1F_{h}$, with bit values defined as follows:

- Bits $10_{h} - 14_{h}$: The length of the EPC that a Tag backscatters, in words:
  - $0000_2$: Zero word.
  - $0001_2$: One word (addresses $20_{h}$ to $2F_{h}$ in EPC memory).
  - $0010_2$: Two words (addresses $20_{h}$ to $3F_{h}$ in EPC memory).
  - $1111_2$: 31 words (addresses $20_{h}$ to $20F_{h}$ in EPC memory).

- Bit $15_{h}$: A User-memory indicator (UMI). If bit $15_{h}$ is deasserted then the Tag either does not implement User memory or User memory contains no information. If bit $15_{h}$ is asserted then User memory contains information. AIT-C01BCB implements the UMI using Method 1 described below.

  **Method 1**: The Tag computes the UMI. At power up, and prior to computing the CRC-16, the Tag shall compute the logical OR of bits $03_{h}$ to $07_{h}$ of User memory and map the computed value into bit $15_{h}$. The Tag shall include the computed UMI value in the CRC-16 calculated at power up. If an Interrogator modifies any of bits $03_{h}$ to $07_{h}$ of User memory then both the UMI and the CRC-16 may be incorrect until the Interrogator power cycles the Tag. The UMI shall not be directly writeable by an Interrogator --- if an Interrogator writes the PC word, the tag shall ignore the data value that the Interrogator provides for bit $15_{h}$.

- Bits $16_{h}$: An XPC_W1 indicator (XI). If bit $16_{h}$ is deasserted then the Tag either does not implement
an XPC_W1 word or the XPC_W1 word is zero-valued, in which case the Tag shall backscatter only
the PC (and not the XPC_W1) word during inventory. If bit 16\textsubscript{h} is asserted then the Tag implements
an XPC_W1 word and one or more of the XPC_W1 bits have nonzero values, indicating that the Tag
has been previously recommissioned. In this latter case the Tag shall backscatter the XPC_W1 word
immediately after the PC word, and before the EPC, during inventory.

If a Tag implements the XPC_W1 word then at power-up, and prior to computing the CRC-16, the
Tag shall compute the bitwise logical OR of the XPC_W1 word and map the computed value into bit
16\textsubscript{h} (i.e. into the XI). The Tag shall include the computed XI value in the CRC-16 calculated at
power-up. If an Interrogator recommissions the Tag then both the XI bit and the CRC-16 may be
incorrect until the Interrogator power cycles the Tag. The XI bit shall not be directly writeable by an
Interrogator — if an Interrogator writes the PC word, the Tag shall ignore the data value that the
Interrogator provides for bit 16\textsubscript{h}.

In AIT-C01BCB XI should always be de-asserted.

- Bits 17\textsubscript{h} – 1F\textsubscript{h}: A numbering system identifier (NSI) whose default value is 0000000000\textsubscript{2}. The MSB of
  the NSI is stored in memory location 17\textsubscript{h}. If bit 17\textsubscript{h} contains a logical 0, then a Tag is referred to as
  an EPCglobal™ Tag and PC bits 18\textsubscript{h} – 1F\textsubscript{h} shall be as defined in the EPC™ Tag Data Standards. If bit
  17\textsubscript{h} contains a logical 1, then a Tag is referred to as a non-EPCglobal™ Tag and PC bits 18\textsubscript{h} – 1F\textsubscript{h}
  shall contain the entire AFI defined in ISO/IEC 15961.

The default (unprogrammed) PC value shall be 3000\textsubscript{h}.

During truncated replies a Tag substitutes 00000\textsubscript{2} for the PC bits. If an Interrogator modifies the EPC
length during a memory write, and it wishes the Tag to subsequently backscatter the modified EPC,
then it must write the length of the new or updated EPC into the first 5 bits of the Tag’s PC. A Tag shall
backscatter an error code, if an Interrogator attempts to write a EPC length that is not supported by the
Tag to the first 5 bits of the Tag’s PC. At power-up a Tag shall compute its CRC-16 over the number of EPC words designated by the first 5 bits of the PC rather than over the length of the entire EPC memory.

### 3.3.3 TID Memory

TID memory locations $00_{h}$ to $07_{h}$ shall contain one of two ISO/IEC 15963 class-identifier values — either $E0_{h}$ or $E2_{h}$. TID memory locations above $07_{h}$ shall be defined according to the registration authority defined by this class identifier value and shall contain, at a minimum, sufficient identifying information for an Interrogator to uniquely identify the custom commands and/or optional features that a Tag supports. TID memory may also contain Tag and vendor-specific data (for example, a Tag serial number).

Note: The Tag manufacturer assigns the class-identifier value (i.e. $E0_{h}$ or $E2_{h}$), for which ISO/IEC 15963 defines the registration authorities. The class-identifier does not specify the Application. If the class identifier is $E0_{h}$, TID memory locations $08_{h}$ to $0F_{h}$ contain an 8-bit manufacturer identifier, TID memory locations $10_{h}$ to $3F_{h}$ contain a 48-bit Tag serial number (assigned by the Tag manufacturer), the composite 64-bit Tag ID (i.e. TID memory $00_{h}$ to $3F_{h}$) is unique among all classes of Tags defined in ISO/IEC 15693, and TID memory is permalocked at the time of manufacture. If the class identifier is $E2_{h}$, TID memory locations $08_{h}$ to $13_{h}$ contain a 12-bit Tag mask-designer identifier (obtainable from the registration authority), TID memory locations $14_{h}$ to $1F_{h}$ contain a vendor-defined 12-bit Tag model number, and the usage of Tag memory locations above $1F_{h}$ is defined in version 1.3 and above of the EPCglobal™ Tag Data Standards.

AIT-C01BCB supports extended TID (XTID), and the TID memory map is defined at Table 3.1.
### Table 3.1. TID memory map

<table>
<thead>
<tr>
<th>Field</th>
<th>Memory Bank</th>
<th>Value</th>
<th>Bit Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Memory and BlockPermaLock</td>
<td>C0h - CFh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td></td>
<td>B0h - BFh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td>BlockWrite and BlockErase</td>
<td>A0h - AFh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td></td>
<td>90h - 9Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td></td>
<td>80h - 8Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td></td>
<td>70h - 7Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td>Optional Command Support</td>
<td>60h - 6Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td>Serial Number (48 bits)</td>
<td>50h - 5Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td></td>
<td>40h - 4Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td></td>
<td>30h - 3Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td>XTIID Header</td>
<td>20h - 2Fh</td>
<td>0000h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td>Extended Tag Identification</td>
<td>10h - 1Fh</td>
<td>0042h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
<tr>
<td></td>
<td>00h - 0Fh</td>
<td>E281h</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
</tbody>
</table>

- Fujitsu MDID = 810h
- Tag Model Number = 042h
- Serialization = 48 + (1 – 1) * 16 = 48 bits = 3 words

#### 3.3.4 User Memory

AIT-C01BCB contains 1KByte User memory. User memory has two Area-Group, each Area-Group contains 16 Areas, and each Area contains 16 words. Show in Fig.3.

Each Area-Group should have memory for Read / Write (32bit), Password, and Other control memory. Show in Fig.4. Block control memory can not visit use normal EPC C1G2 command. Custom should use custom command 3.13.16-3.13.18 to modify control memory.

If a Tag’s User memory has not yet been programmed then the 5 LSBs of the first word of User memory (i.e. memory addresses 03h to 07h) shall have the default value 000002.
### 3.4 Sessions, inventoried and selected flags

Sessions and inventoried flags refer to section 6.3.2.2, Ref.1.

### 3.5 Selected flags

Selected flags refer to section 6.3.2.3, Ref.1.

### 3.6 Tag states and slot counter

Tag states and slot counter refer to section 6.3.2.4, Ref.1.

### 3.7 Tag random or pseudo-random number generator

Tag random or pseudo-random number generator refers to section 6.3.2.5, Ref.1.
3.8 Managing Tag populations

Managing Tag populations refer to section 6.3.2.6, Ref.1

3.9 Selecting Tag populations

Selecting Tag populations refer to section 6.3.2.7, Ref.1

3.10 Inventorying Tag populations

Inventorying Tag populations refer to section 6.3.2.8, Ref.1

3.11 Accessing individual Tags

Inventorying Tag populations refer to section 6.3.2.9, Ref.1

3.12 Killing or recommissioning a Tag

Killing or recommissioning a Tag refer to section 6.3.2.10, Ref.1

3.13 Commands and Tag reply

3.13.1 Select (mandatory)

Select command refer to section 6.3.2.11.1.1, Ref.1

Note:
If part of the bits to be selected in the user memory is read locked by the AreaReadLock(3.13.17), the Tag shall be set to be non-matching and transition to Ready state except in Kill state.

3.13.2 Query (mandatory)

Query command refer to section 6.3.2.11.2.1, Ref.1

3.13.3 QueryAdjust (mandatory)

QueryAdjust command refer to section 6.3.2.11.2.2, Ref.1
3.13.4 QueryRep (mandatory)

QueryAdjust command refer to section 6.3.2.11.2.3, Ref.1

3.13.5 ACK (mandatory)

ACK command refer to section 6.3.2.11.2.4, Ref.1

3.13.6 NAK (mandatory)

NACK command refer to section 6.3.2.11.2.5, Ref.1

3.13.7 Req_RN (mandatory)

Req_RN command refer to section 6.3.2.11.3.1, Ref.1

3.13.8 Read (mandatory)

Read command refer to section 6.3.2.11.3.2, Ref.1
Note: If part of the words to be read in the user memory is locked by the AreaReadLock(3.13.17), the Tag reply an error code (Memory locked). The error code is compliant with Annex I. Ref.1.

3.13.9 Write (mandatory)

Write command refer to section 6.3.2.11.3.3, Ref.1
Note: If part of the words to be written in the user memory is locked by the AreaWriteLock(3.13.18), the Tag reply an error code (Memory locked). The error code is compliant with Annex I. Ref.1

3.13.10 Kill (mandatory)

Kill command refer to section 6.3.2.11.3.4, Ref.1
Note: The "recommission" is not supported in AIT-C01BCB. If AIT-C01BCB received kill command, it ignores the recommissioning bits and treats them as though they were zero, meaning that, if AIT-C01BCB receives a properly formatted kill command sequence with the correct kill password it kills itself dead regardless of the values of the recommissioning bits.

3.13.11 Lock (mandatory)

Lock command refer to section 6.3.2.11.3.5, Ref.1
3.13.12 Access (Implement in AIT-C01BCB)

Access command refer to section 6.3.2.11.3.6, Ref.1

3.13.13 BlockWrite (Implement in AIT-C01BCB)

AIT-C01BCB implement a BlockWrite command, as shown in Table 3.2. BlockWrite allows an Interrogator to write multiple words in a Tag’s Reserved, EPC, TID, or User memory using a single command.

BlockWrite has the following fields:

- MemBank specifies whether the BlockWrite occurs in Reserved, EPC, TID, or User memory.
  
  BlockWrite commands shall apply to a single memory bank. Successive BlockWrites may apply to different banks.

- WordPtr specifies the starting word address for the memory write, where words are 16 bits in length. For example, WordPtr = 00h specifies the first 16-bit memory word, WordPtr = 01h specifies the second 16-bit memory word, etc. WordPtr uses EBV formatting (see reference 1. Annex A).

- WordCount specifies the number of 16-bit words to be written. If WordCount = 00h, the tag shall ignore the BlockWrite. If WordCount = 01h, the tag shall write a single data word. AIT-C01BCB supports 1~8 words to be written in a single BlockWrite command. If WordCount > 8 the tag shall backscatter an error code (Error-specific memory overrun), see reference 1. Annex I.

- Data contains the 16-bit words to be written, and shall be 16×WordCount bits in length. Unlike a Write, the data in a BlockWrite are not cover-coded, and an Interrogator need not issue a Req_RN before issuing a BlockWrite.

The BlockWrite command also includes the Tag’s handle and a CRC-16. The CRC-16 is calculated over the first command-code bit to the last handle bit.
If a Tag receives a BlockWrite with a valid CRC-16 but an invalid handle it shall ignore the BlockWrite and remain in its current state (open or secured, as appropriate).

A BlockWrite shall be prepended with a frame-sync.

After issuing a BlockWrite an Interrogator shall transmit CW for the lesser of TREPLY or 4ms, where TREPLY is the time between the Interrogator’s BlockWrite command and the Tag’s backscattered reply. An Interrogator may observe several possible outcomes from a BlockWrite, depending on the success or failure of the Tag’s memory write operation:

- **The BlockWrite succeeds:** After completing the BlockWrite a Tag shall backscatter the reply shown in Table 3.3 and comprising a header (a 0-bit), the Tag’s handle, and a CRC-16 calculated over the 0-bit and handle. If the Interrogator observes this reply within 4 ms then the BlockWrite completed successfully.

- **The Tag encounters an error:** The Tag shall backscatter an error code during the CW period rather than the reply shown in Table 3.3 (see reference 1. Annex I for error-code definitions and for the reply format, write to the CRC-16 stored in 00h to 0fh in EPC bank is also an error condition).

- **The BlockWrite does not succeed:** If the Interrogator does not observe a reply within 4ms then the BlockWrite did not complete successfully. The Interrogator may issue a Req_RN command (containing the Tag’s handle) to verify that the Tag is still in the Interrogator’s field, and may reissue the BlockWrite.

Upon receiving a valid BlockWrite command a Tag shall write the commanded Data into memory. The Tag’s reply to a BlockWrite shall use the extended preamble, as appropriate (i.e. a Tag shall reply as if TRext=1 regardless of the TRext value in the Query that initiated the round).

Note: If part of the words to be written in the user memory is locked by the AreaWriteLock(3.13.18), the Tag reply an error code (Memory locked). The error code is compliant with Annex I,Ref.1
3.13.14 BlockErase (Implement in AIT-C01BCB)

AIT-C01BCB implements a BlockErase command, as shown in Table 3.4.

BlockErase allows an Interrogator to erase multiple words into 0h in a Tag’s Reserved, EPC, TID, or User memory using a single command.

BlockErase has the following fields:

- MemBank specifies whether the BlockErase occurs in Reserved, EPC, TID, or User memory.
  - BlockErase commands shall apply to a single memory bank. Successive BlockErases may apply to different banks.

- WordPtr specifies the starting word address for the memory erase, where words are 16 bits in length. For example, WordPtr = 00h specifies the first 16-bit memory word, WordPtr = 01h specifies the second 16-bit memory word, etc. WordPtr uses EBV formatting (see reference 1. Annex A).

- WordCount specifies the number of 16-bit words to be erased. If WordCount = 00h the tag shall ignore the BlockErase. If WordCount = 01h the tag shall erase a single data word. AIT-C01BCB supports 1-8 words to be erased in a single BlockErase command. If WordCount > 8 the tag shall
backscatter an error code (Error-specific memory overrun), see reference 1. Annex I.

- The BlockErase command also includes the Tag’s handle and a CRC-16. The CRC-16 is calculated over the first command-code bit to the last handle bit.

  If a Tag receives a BlockErase with a valid CRC-16 but an invalid handle it shall ignore the BlockErase and remain in its current state (open or secured, as appropriate).

  A BlockErase shall be prepended with a frame-sync.

  After issuing a BlockErase an Interrogator shall transmit CW for the lesser of TRePLY or 4ms, where TRePLY is the time between the Interrogator’s BlockErase command and the Tag’s backscattered reply.

  An Interrogator may observe several possible outcomes from a BlockErase, depending on the success or failure of the Tag’s memory erase operation:

  - **The BlockErase succeeds:** After completing the BlockErase a Tag shall backscatter the reply shown in Table 3.5 and comprising a header (a 0-bit), the Tag’s handle, and a CRC-16 calculated over the 0-bit and handle. If the Interrogator observes this reply within 4 ms then the BlockErase completed successfully.

  - **The Tag encounters an error:** The Tag shall backscatter an error code during the CW period rather than the reply shown in Table 3.5 (see reference 1. Annex I for error-code definitions and for the reply format, erase the CRC-16 stored in 00h to 0fh in EPC bank is also an error condition).

  - **The BlockErase does not succeed:** If the Interrogator does not observe a reply within 4ms then the BlockErase did not complete successfully. The Interrogator may issue a Req_RN command (containing the Tag’s handle) to verify that the Tag is still in the Interrogator’s field, and may reissue the BlockErase.

    Upon receiving a valid BlockErase command a Tag shall erase the commanded memory words.

The Tag’s reply to a BlockErase shall use the extended preamble, as appropriate (i.e. a Tag shall reply as
if TRext=1 regardless of the TRext value in the Query that initiated the round).

Note: If part of the words to be written in the user memory is locked by the AreaWriteLock(3.13.18), the Tag reply an error code (Memory locked). The error code is compliant with Annex I.Ref.1.

<table>
<thead>
<tr>
<th># of bits</th>
<th>Command</th>
<th>MemBank</th>
<th>WordPtr</th>
<th>WordCount</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>EBV</td>
<td>8</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>description</td>
<td>11001000</td>
<td>00: Reserved</td>
<td>Starting address pointer</td>
<td>Number of words to erase</td>
<td>handle</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.4 Blockerase command

<table>
<thead>
<tr>
<th># of bits</th>
<th>Header</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>description</td>
<td>0</td>
<td>handle</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.5. Tag reply to a successful Blockerase command

3.13.15 BlockPermalock (Implement in AIT-C01BCB)

AIT-C01BCB implement a BlockPermalock command, as shown in Table 3.7. BlockPermalock allows an Interrogator to:

- Permalock one or more blocks (individual sub-portions) in a Tag’s User memory, or
- Read the permalock status of the memory blocks in a Tag’s User memory.

In BlockPermalock command operation the block size define as Word (16 bits). There are 512 blocks in AIT-C01BCB.

Only Tags in the secured state shall execute a BlockPermalock command.

The BlockPermalock command differs from the Lock command in that BlockPermalock permanently locks blocks of User memory in an unwriteable state, whereas Lock reversibly or permanently locks a password or an entire memory bank in a writeable or unwriteable state. Table 3.6 specifies how a Tag shall react to a BlockPermalock command (with Read/Lock = 1) that follows a prior
Lock command, or vice versa.

Table 3.6 Precedence for Lock and BlockPermalock commands

<table>
<thead>
<tr>
<th>First Command</th>
<th>Second Command</th>
<th>Tag Action and Response to 2nd Command</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>pwr-write</td>
<td>permalock</td>
</tr>
<tr>
<td>Lock</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>BlockPermalock (Read/Lock = 1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pwr-write</td>
<td>permalock</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The BlockPermalock command has the following fields:

- **MemBank** specifies whether the BlockPermalock applies to EPC, TID, or User memory. BlockPermalock commands shall apply to a single memory bank. Successive BlockPermalocks may apply to different memory banks. Class-1 Tags shall only execute a BlockPermalock command if MemBank = 11 (User memory); if a Class-1 Tag receives a BlockPermalock with MemBank<>11 it shall ignore the command and instead backscatter an error code (see reference 1. Annex G), remaining in the secured state. Higher-functionality Tags may use the other MemBank values to expand the functionality of the BlockPermalock command.

- **Read/Lock** specifies whether a Tag backscatters the permalock status of, or permalocks, one or more blocks within the memory bank specified by MemBank. A Tag shall interpret the Read/Lock bit as follows:

  Read/Lock = 0: A Tag shall backscatter the permalock status of blocks in the specified memory bank,
starting from the memory block located at BlockPtr and ending at the memory block located at BlockPtr+\((16\times\text{BlockRange})-1\). A Tag shall backscatter a “0” if the memory block corresponding to that bit is not permalocked and a “1” if the block is permalocked. An Interrogator omits Mask from the BlockPermalock command when Read/Lock = 0.

Read/Lock = 1: A Tag shall permlock those blocks in the specified memory bank that are specified by Mask, starting at BlockPtr and ending at BlockPtr+\((16\times\text{BlockRange})-1\).

- **BlockPtr** specifies the starting address for Mask, in units of 16 blocks. For example, BlockPtr = 00h indicates block 0, BlockPtr = 01h indicates block 16, BlockPtr = 02h indicates block 32. BlockPtr uses EBV formatting (see reference 1. Annex A).

- **BlockRange** specifies the range of Mask, starting at BlockPtr and ending \((16\times\text{BlockRange})-1\) blocks later. If BlockRange = 00h then a Tag shall ignore the BlockPermalock command and instead backscatter an error code (see reference 1. Annex I.), remaining in the **secured** state. A single AIT-C01BCB BlockPermalock command can permlock between 0 and 128 blocks of User memory. That means the valid number of BlockRange parameter is 1~8, if BlockRange > 8 then the tag backscatter an error code (Error-specific memory overrun), see reference 1. Annex I.

- **Mask** specifies which memory blocks a Tag permlocks. Mask depends on the Read/Lock bit as follows:
  - **Read/Lock = 0**: The Interrogator shall omit Mask from the BlockPermalock command. *BlockPerm*length 16x ck command.
  - **Read/Lock = 1**: The Interrogator shall include a Mask of length \(16\times\text{BlockRange}\) bits in the BlockPermalock command. The Tag shall interpret each bit of Mask as follows:
    - **Mask** bit = 0: Retain the current permalock setting for the corresponding memory block.
    - **Mask** bit = 1: Permalock the corresponding memory block. If a block is already permalocked
then the Tag shall retain the current permalock setting. A memory block, once permalocked, cannot be un-permaloced.

The following examples illustrate the usage of Read/Lock, BlockPtr, BlockRange, and Mask:

- If BlockRange=00 the Tag ignores the BlockPermalock command.
- If Read/Lock=1, BlockPtr=01, and BlockRange=01 the Tag operates on sixteen blocks starting at block 16 and ending at block 31, permalocking those blocks whose corresponding bits are asserted in Mask.

The BlockPermalock command contains 8 RF Class-1 Tag shall ignore these bits. Interrogators shall set these bits to 00 when communicating with Class-1 Tags. Higher-functionality Tags may use these bits to expand the functionality of the BlockPermalock command.

The BlockPermalock command also includes the Tag’s handle and a CRC-16. Th CRC-16 is calculated over the first command-code bit to the last handle bit. If a Tag receives a BlockPermalock with a valid CRC-16 but an invalid handle it shall ignore the BlockPermalock and remain in the secured state.

If a Tag receives a BlockPermalock command that it cannot execute because User memory does not exist, or in which Read/Lock = 0 and Mask <> 0, or in which Mask has a length that is not equal to 16×BlockRange bits, or in which the LSB and/or the 2SB of a Tag’s XPC_W1 word is/are asserted, or in which one of the asserted Mask bits references a non-existent block, then the Tag shall ignore the BlockPermalock command and backscatter an error code (see reference 1. Annex G), remaining in the secured state.

Certain Tags, depending on the Tag manufacture’s implementation, may be unable to execute a BlockPermalock command with certain BlockPtr and BlockRange values, in which case the Tag shall ignore the blockPermalock command and instead backscatter an error code (see reference 1. Annex G), remaining in the secured state. Because a Tag contains information in its TID memory that an Interrogator can use to uniquely identify the optional features that supports, this specification

©FUJITSU LIMITED 2013
recommends that Interrogators read a Tag's TID memory prior to issuing a BlockPermalock command.

If an Interrogator issues a BlockPermalock command in which BlockPtr and BlockRange specify one or more nonexistent blocks, but Mask only asserts permalocking on existent blocks, then the Tag shall execute the command.

A BlockPermalock shall be prepended with a frame-sync.

After issuing a BlockPermalock command an Interrogator shall transmit CW for the lesser of $T_{\text{REPLY}}$ or 20ms, where $T_{\text{REPLY}}$ is the time between the Interrogator's BlockPermalock and the Tag’s backscattered reply. An Interrogator may observe several possible outcomes from a BlockPermalock command, depending on the value of the Read/Lock bit in the command and, if Read/Lock = 1, the success or failure of the Tag’s memory-lock operation:

- **Read/Lock = 0 and the Tag is able to execute the command:** The Tag shall backscatter the reply shown in Table 3.8, within time T1, comprising a header (a 0-bit), the requested permalock bits, the Tag’s handle, and a CRC-16 calculated over the 0-bit, permalock bits, and handle. The Tag’s reply shall use the preamble specified by the TRext value in the Query that initiated the round.
- **Read/Lock = 0 and the Tag is unable to execute the command:** the Tag shall backscatter an error code, within time T1, rather than the reply shown in Table 3.8 (see reference 1. Annex G for error-code definitions and for the reply format). The Tag’s reply shall use the preamble specified by the TRext value in the Query that initiated the round.

Table 3.7 BlockPermalock command

<table>
<thead>
<tr>
<th># of bits</th>
<th>Command</th>
<th>RFU</th>
<th>MemBank</th>
<th>Read/Lock</th>
<th>BlockPtr</th>
<th>BlockRange</th>
<th>Mask</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>11001001</td>
<td>00</td>
<td>02</td>
<td>1</td>
<td>EBV</td>
<td>8</td>
<td>Variable</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td><strong>description</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00</td>
<td>00: RFU 01: EPC 10: TID 11: User</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Read 1: Permalock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Mask starting address, specified in units of 16 blocks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Mask range, specified in units of 16 blocks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Retain current permalock setting</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Assert permalock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>handle</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.8 Tag reply to a successful BlockPermalock command with Read/Lock = 0

<table>
<thead>
<tr>
<th># of bits</th>
<th>Header</th>
<th>Data</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Variable</td>
<td>16</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.9 Tag reply to a successful BlockPermalock command with Read/Lock = 1

<table>
<thead>
<tr>
<th># of bits</th>
<th>Header</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

- **Read/Lock = 1 and The BlockPermalock succeeds:** After completing the BlockPermalock, the Tag shall backscatter the reply shown in Table 3.9 comprising a header (a 0-bit), the Tag’s handle, and a CRC-16 calculated over the 0-bit and handle. If the Interrogator observes this reply within 20 ms then the BlockPermalock completed successfully. The Tag’s reply shall use the extended preamble, as appropriate (i.e. the Tag shall reply as if TRext=1 regardless of the TRext value in the Query that initiated the round).

- **Read/Lock = 1 and the Tag encounters an error:** The Tag shall backscatter an error code during the CW period rather than the reply shown in Table 3.9 (see reference 1. Annex G for error-code definitions and for the reply format). The Tag’s reply shall use the extended preamble, as appropriate (i.e. the Tag shall reply as if TRext=1 regardless of the TRext value in the Query that initiated the round).

- **Read/Lock = 1 and the BlockPermalock does not succeed:** If the Interrogator does not observe a reply within 20 ms then the BlockPermalock did not complete successfully. The Interrogator may issue a Req_RN command (containing the Tag’s handle) to verify that the Tag is still in the Interrogator’s field, and may reissue the BlockPermalock.

  Upon receiving a valid BlockPermalock command a Tag shall perform the commanded operation,
unless the Tag does not support block permalocking, in which case it shall ignore the command.

3.13.16 ChgAreaGroupPwd (Custom Command)

AIT-C08BCB has the change Area group password function. Table 3.10 shows the format of the ChgAreaGroupPwd command. The R/W can use ChgAreaGroupPwd to change the password of each AreaGroup. To change the password of the AreaGroup, the R/W shall send the existing and new passwords. The initial password set as “0” at shipment stage from factory.

Only the tag in the secured or open state can execute the ChgAreaGroupPwd command.

ChgAreaGroupPwd has the following fields:

- **AreaGroupPtr**: AIT-C01BCB has 16 AreaGroup(00~15), which is specified with 4 bit value. MSB bit shall be padded with Zero “0”.

- **Data**: specifies the new password.

- **Password**: specifies the current password.

The ChgAreaGroupPwd command also includes the Tag handle and CRC-16. The CRC-16 is calculated over the first command code bit to the last handle bit.

If a Tag in the open or secured state receives ChgAreaGroupPwd with a valid CRC-16 but an invalid handle, it shall ignore the ChgAreaGroupPwd and remain in its current state. If the password set in the Passwd field does not correspond to the stored value, the new password is not written. The Tag will not reply and return to the arbitrate state.

A ChgAreaGroupPwd shall be pretended with a frame-sync.

After issuing ChgAreaGroupPwd, the R/W transmit CW for the lesser of Treply or 20ms, where Treply is the time between the R/W's ChgAreaGroupPwd command and Tag's backscattered reply. The R/W may observe several possible incomes from a ChgAreaGroupPwd, depending on the success or failure of the
Tag’s password change operation.

- **ChgAreaGroupPwd succeeds:** After complete the ChgAreaGroupPwd, the Tag return the response shown in Table 3.11. The reply includes a header ("0" bit), Tag handle, and CRC-16 calculated over the "0" bit and handle. If the R/W observes this reply within 20 ms then the ChgAreaGroupPwd is completed successfully.

- **The tag encounters an error:** Tag returns an error code during the CW period rather than the reply shown in Table 3.11.

- **Failure:** If the R/W does not observe a reply within 20 ms then the ChgAreaGroupPwd does not complete successfully. The R/W may issue a Req_RN command (containing the Tag’s handle) to verify that the tag is still in the R/W filed, and may reissue the ChgAreaGroupPwd.

  Upon receiving a valid ChgAreaGroupPwd command, the Tag rewrites the AreaGroup password with the specified data. The new password is valid immediately after rewriting. The tag’s reply to a ChgAreaGroupPwd uses the extended preamble. (i.e., the Tag reply as if TRext = 1 regardless of the TRext value in the Query that initiated the round).

<table>
<thead>
<tr>
<th># of bits</th>
<th>Command</th>
<th>AreaGroupPtr</th>
<th>Data</th>
<th>Password</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>1110 0000 0000 0100</td>
<td>16</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>16</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.10. ChgAreaGroupPwd command

<table>
<thead>
<tr>
<th># of bits</th>
<th>Header</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>0</td>
<td>16</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 3.11. Tag reply to ChgAreaGroupPwd command
3.13.17 AreaReadLock (Custom Command)

AIL-C01BCB has the AreaReadLock function. The AreaReadLock command specifies the AreaReadLock status in the control memory. The format of AreaReadLock command is shown in Table 3.12.

Only the tag in the secured or open state can execute the AreaReadLock command.

The AreaReadLock command has the following fields:

- **AreaGroupPtr**: AIL-C01BCB has 16 AreaGroup (00~15), which is specified with 4 bit value. MSB bit shall be padded with Zero “0”.

- **AreaReadLock**: contains a 32-bit payload defined as follows.
  - **MASK0-15**: Set the AreaReadLock status (1: Assert AreaReadLock, 0: Deassert AreaReadLock)
  - **Action0-15**: Set the AreaReadLock status (1: Assert AreaReadLock, 0: Deassert AreaReadLock)

- **Password**: Set the corresponding password to the AreaGroup specified by the AreaGroupPtr. If the password is wrong, the AreaReadLock status can not be changed.

The AreaReadLock command also includes the Tag handle and CRC-16. The CRC-16 is calculated over the first command code bit to the last handle bit.

If a Tag in the open or secured state receives AreaReadLock with a valid CRC-16 but an invalid handle, it shall ignore the AreaReadLock and remain in its current state.

An AreaReadLock shall be prepended with a frame-sync.

After issuing AreaReadLock, the R/W transmit CW for the lesser of Treply or 20ms, where Treply is the time between the R/W’s AreaReadLock command and Tag’s backscattered reply. The R/W may observe several possible incomes from a AreaReadLock, depending on the success or failure of the
Tag's memory lock operation.

- **AreaReadLock succeeds:** After complete the AreaReadLock, the Tag return the response shown in Table 3.14. The reply includes a header ("0" bit), Tag handle, and CRC-16 calculated over the "0" bit and handle. If the RW observes this reply within 20 ms then the AreaReadLock is completed successfully.

- **The tag encounters an error:** Tag returns an error code during the CW period rather than the reply shown in Table 3.14.

- **Failure:** If the RW does not observe a reply within 20ms then the AreaReadLock does not complete successfully.

The RW may issue a Req_RN command (containing the Tag's handle) to verify that the tag is still in the RW filed, and may reissue the AreaReadLock.

Upon receiving a valid AreaReadLock command, the Tag performs the lock operation.

The tag’s reply to a AreaReadLock use the extended preamble. (i.e., the Tag reply as if TRext = 1 regardless of the TRext value in the Query that initiated the round).

The Read operation will not be affected by EPC Gen2 Lock status; Read operation will be enabled/disabled only by the AreaReadLock status.

<table>
<thead>
<tr>
<th></th>
<th>Command</th>
<th>AreaGroupPtr</th>
<th>Payload</th>
<th>Password</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td># of bits</td>
<td>16</td>
<td>5</td>
<td>32</td>
<td>32</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Description</td>
<td>1110 0000 0000 0111</td>
<td>AreaGroupPtr</td>
<td>Mask/Action (see Table3.13)</td>
<td>32bit password</td>
<td>Handle</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.12. AreaReadLock command

<table>
<thead>
<tr>
<th>MASK</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1</td>
<td>0 1 2 3 4 5 6 7 8 9 1 1 1 1 1 1 1 1</td>
</tr>
</tbody>
</table>
### 3.13.18 AreaWriteLock (Custom Command)

AIT-C01BCB has the AreaWriteLock function. The AreaWriteLock command specifies the AreaWriteLock status in the control memory.

The format of AreaWriteLock command is shown in Table 3.15.

Only the tag in the secured or open state can execute the AreaWriteLock command.

The AreaWriteLock command has the following fields:

- **AreaGroupPtr**: AIT-C01BCB has 16 AreaGroup(00-15), which is specified with 4 bit value. MSB bit shall be padded with Zero 0.

- **AreaWriteLock** contains a 32-bit payload defined as follows.

  * **MASK0-15**
    
    0: Ignore the associated Action field and retain the current setting.
    
    1: Implement the associated Action field and overwrite the current AreaWriteLock setting.

  * **Action0-15**: Set the AreaWriteLock status
    
    (1: Assert AreaWriteLock, 0: Deassert AreaWriteLock)

- **Password**: Set the corresponding password to the AreaGroup specified by the AreaGroupPtr. If the password is wrong, the AreaWriteLock status can not be changed.

The AreaWriteLock command also includes the Tag handle and CRC-16. The CRC-16 is calculated
over the first command code bit to the last handle bit.

If a Tag in the open or secured state receives AreaWriteLock with a valid CRC-16 but an invalid handle, it shall ignore the AreaWriteLock and remain in its current state.

An AreaWriteLock shall be prepended with a frame-sync.

After issuing AreaWriteLock, the R/W transmit CW for the lesser of Treply or 20ms, where Treply is the time between the R/W's AreaWriteLock command and Tag's backscattered reply. The R/W may observe several possible incomes from an AreaWriteLock, depending on the success or failure of the Tag's memory lock operation.

- AreaWriteLock succeeds: After complete the AreaWriteLock, the Tag returns the response shown in Table 3.17. The reply includes a header ("0" bit), Tag handle, and CRC-16 calculated over the "0" bit and handle. If the R/W observes this reply within 20 ms then the AreaWriteLock is completed successfully.
- The tag encounters an error: Tag returns an error code during the CW period rather than the reply shown in Table 3.17.
- Failure: If the R/W does not observe a reply within 20ms then the AreaWriteLock does not complete successfully.

The R/W may issue a Req_RN command (containing the Tag's handle) to verify that the tag is still in the R/W filed, and may reissue the AreaWriteLock.

Upon receiving a valid AreaWriteLock command, the Tag performs the lock operation. The tag’s reply to an AreaWriteLock use the extended preamble. (i.e., the Tag reply as if TRext = 1 regardless of the TRext value in the Query that initiated the round).

Write operation on the User memory must be controlled as Table 3.18

The user memory control function is shown in Fig.5
<table>
<thead>
<tr>
<th>Command</th>
<th>AreaGroupPtr</th>
<th>Payload</th>
<th>Password</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td># of bits</td>
<td>16</td>
<td>5</td>
<td>32</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>Description</td>
<td>1110 0000 0000 1000</td>
<td>AreaGroupPtr</td>
<td>Mask/Action (see Table 3.16)</td>
<td>32bit password</td>
<td>Handle</td>
</tr>
</tbody>
</table>

Table 3.15. AreaWriteLock command

<table>
<thead>
<tr>
<th>MASK</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>1 0 1 2 3 4 5 0 1 2</td>
<td>1 0 1 2 3 4 5 1 0 1</td>
</tr>
</tbody>
</table>

Table 3.16. AreaWriteLock command Payload

<table>
<thead>
<tr>
<th>Header</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td># of bits</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>Description</td>
<td>0</td>
<td>Handle</td>
</tr>
</tbody>
</table>

Table 3.17. Tag reply to AreaWriteLock command

<table>
<thead>
<tr>
<th>Pwd write</th>
<th>Perma lock</th>
<th>Area Write Lock</th>
<th>Block Perma Lock</th>
<th>WRITE Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Secure state</td>
</tr>
<tr>
<td>0 0</td>
<td>0</td>
<td>Enable</td>
<td>Enable</td>
<td></td>
</tr>
<tr>
<td>1 X</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td></td>
</tr>
<tr>
<td>X 1</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>0</td>
<td>Enable</td>
<td>Enable</td>
<td></td>
</tr>
<tr>
<td>1 X</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td></td>
</tr>
<tr>
<td>X 1</td>
<td>Impossible to transit to this state</td>
<td>(Same as left)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>0</td>
<td>Enable</td>
<td>Disable</td>
<td></td>
</tr>
<tr>
<td>1 X</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td></td>
</tr>
<tr>
<td>X 1</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>X</td>
<td>Disable</td>
<td>Disable</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.18 Write operation on User Memory
3.13.19  AreaWriteLockwoPwd (Custom Command)

AIT-C01BCB has the AreaWriteLockwoPwd function. The AreaWriteLockwoPwd command can set the AreaWriteLock status in the control memory **without password**. This command **can not reset** the AreaWriteLock status.

The format of AreaWriteLockwoPwd command is shown in Table 3.19.

Only the tag in the secured or open state can execute the AreaWriteLockwoPwd command.

The AreaWriteLockwoPwd command has the following fields:

- **AreaGroupPtr**: AIT-C01BCB has 16 AreaGroup(00~15), which is specified with 4 bit value. MSB bit shall be padded with Zero“0”

- **AreaWriteLockwoPwd** contains a 16-bit payload defined as follows.

  *Action0-15* : 

©FUJITSU LIMITED 2013
Set the AreaWriteLock status (1: Assert AreaWriteLock, 0: No Action)

The AreaWriteLockwoPwd command also includes the Tag handle and CRC-16. The CRC-16 is calculated over the first command code bit to the last handle bit.

If a Tag in the open or secured state receives AreaWriteLockwoPwd with a valid CRC-16 but an invalid handle, it shall ignore the AreaWriteLockwoPwd and remain in its current state.

An AreaWriteLockwoPwd shall be prepended with a frame-sync.

After issuing AreaWriteLockwoPwd, the R/W transmit CW for the lesser of Treply or 20ms, where Treply is the time between the R/W's AreaWriteLockwoPwd command and Tag's backscattered reply. The R/W may observe several possible incomes from an AreaWriteLockwoPwd, depending on the success or failure of the Tag's memory lock operation.

- AreaWriteLockwoPwd succeeds: After complete the AreaWriteLockwoPwd, the Tag return the response shown in Table 3.21. The reply includes a header ("0" bit), Tag handle, and CRC-16 calculated over the "0" bit and handle. If the R/W observes this reply within 20 ms then the AreaWriteLockwoPwd is completed successfully.

- The tag encounters an error: Tag returns an error code during the CW period rather than the reply shown in Table 3.21.

- Failure: If the R/W does not observe a reply within 20ms then the AreaWriteLockwoPwd does not complete successfully.

The R/W may issue a Req_RN command (containing the Tag's handle) to verify that the tag is still in the R/W filed, and may reissue the AreaWriteLockwoPwd.

Upon receiving a valid AreaWriteLockwoPwd command, the Tag perform the lock operation.

The tag’s reply to a AreaWriteLockwoPwd use the extended preamble. (i.e., the Tag reply as if TRext = 1 regardless of the TRext value in the Query that initiated the round).
Write operation on the User memory must be controlled as Table 3.18.

The user memory control function is shown in Fig.5

<table>
<thead>
<tr>
<th>Command</th>
<th>AreaGroupPtr</th>
<th>Payload</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td># of bits</td>
<td>16</td>
<td>5</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Description</td>
<td>1110 0000 0000 1001</td>
<td>AreaGroupPtr</td>
<td>Action (see Table 3.20)</td>
<td>Handle</td>
</tr>
</tbody>
</table>

Table 3.19. AreaWriteLockwoPwd command

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</td>
</tr>
</tbody>
</table>

Table 3.20. AreaWriteLockwoPwd command Payload

<table>
<thead>
<tr>
<th>Header</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td># of bits</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>Description</td>
<td>0</td>
<td>Handle</td>
</tr>
</tbody>
</table>

Table 3.21. Tag reply to AreaWriteLockwoPwd command

3.13.20 ChgCustomPwd (Custom Command)

A1T-C01BCB has the change Custom password function. Table 3.22 shows the format of the ChgCustomPwd command. The R/W can use ChgCustomPwd to change the password of Custom. To change the password of Custom, the R/W shall send the existing and new passwords. The initial password set as “0” at shipment stage from factory.

Only the tag in the secured or open state can execute the ChgCustomPwd command.

ChgCustomPwd has the following fields:

- Data: specifies the new custom password.
- Password: specifies the current custom password.
The ChgCustomPwd command also includes the Tag handle and CRC-16. The CRC-16 is calculated over the first command code bit to the last handle bit.

If a Tag in the open or secured state receives ChgCustomPwd with a valid CRC-16 but an invalid handle, it shall ignore the ChgCustomPwd and remain in its current state. If the password set in the Passwd field does not correspond to the stored value, the new password is not written. The Tag will not reply and return to the arbitrate state.

A ChgCustomPwd shall be pretended with a frame-sync.

After issuing ChgCustomPwd, the R/W transmit CW for the lesser of Treply or 20ms, where Treply is the time between the R/W's ChgCustomPwd command and Tag's backscattered reply. The R/W may observe several possible incomes from a ChgCustomPwd, depending on the success or failure of the Tag's password change operation.

- **ChgCustomPwd succeeds**: After complete the ChgCustomPwd, the Tag return the response Shown in Table 3.23. The reply includes a header ("0" bit), Tag handle, and CRC-16 calculated over the "0" bit and handle. If the R/W observes this reply within 20 ms then the ChgCustomPwd is completed successfully.

- **The tag encounters an error**: Tag returns an error code during the CW period rather than the reply shown in Table 3.23.

- **Failure**: If the R/W does not observe a reply within 20ms then the ChgCustomPwd does not complete successfully. The R/W may issue a Req_RN command (containing the Tag's handle) to verify that the tag is still in the R/W filed, and may reissue the ChgCustomPwd.

Upon receiving a valid ChgCustomPwd command, the Tag rewrites the Custom password with the specified data. The new password is valid immediately after rewriting. The tag’s reply to a ChgCustomPwd uses the extended preamble. (i.e., the Tag reply as if TExt = 1 regardless of the TText.
value in the Query that initiated the round).

<table>
<thead>
<tr>
<th>Command</th>
<th>Data</th>
<th>Password</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td># of bits</td>
<td>16</td>
<td>32</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>Description</td>
<td>1110 0000</td>
<td>New</td>
<td>Current</td>
<td>Handle</td>
</tr>
<tr>
<td></td>
<td>0000 1100</td>
<td>Passwd</td>
<td>Passwd</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.22 ChgCustomPwd command

<table>
<thead>
<tr>
<th>Header</th>
<th>RN</th>
<th>CRC-16</th>
</tr>
</thead>
<tbody>
<tr>
<td># of bits</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>Description</td>
<td>0</td>
<td>Handle</td>
</tr>
</tbody>
</table>

Table 3.23 Tag reply to ChgCustomPwd command

4 Reference

[1]: uhfc1g2_1_2_0-standard-20080511.pdf
EPC™ Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz – 960 MHz Version 1.2.0 Gen2)