

Specification Of 1KBytes UHF Band RFID chip

DOCUMENT NUMBER:	REVISION NUMBER	RELEASE/REVISION DATE:
20130531-01F	Rev 001	May 31, 2013

FUJITSU LIMITED

1 FEATURES

1.1 General specification

- EPCglobal C1G2 1.2.0 compliant [Note1]

1.2 Mechanical specification

- Diameter: 200mm (8inch)
- Thickness: $150\mu\text{m} \pm 10\%$
- Number of pads: 4
- Die size without scribe: $1460\mu\text{m} * 1460\mu\text{m}$
- Bump material: $>99.9\%$ pure Au
- Map file distribution: IBIS Format (for Muehlbauer FCM10000)

Limiting values

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

Memory characteristics

- EEPROM data retention (T_{ret}):
 $30\text{years}@85^{\circ}\text{C}$
- EEPROM write endurance
 $100000@55^{\circ}\text{C}$

Interface characteristics

- Operating frequency: 840MHz 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 ($\text{Im}(Z_{\text{chip}})/\text{Re}(Z_{\text{chip}})$): <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

- Block write/erase function
- Blockpermalock function
- Block Password protect
- Custom defined command

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 Frontend module, baseband and EEPROM. There are 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.

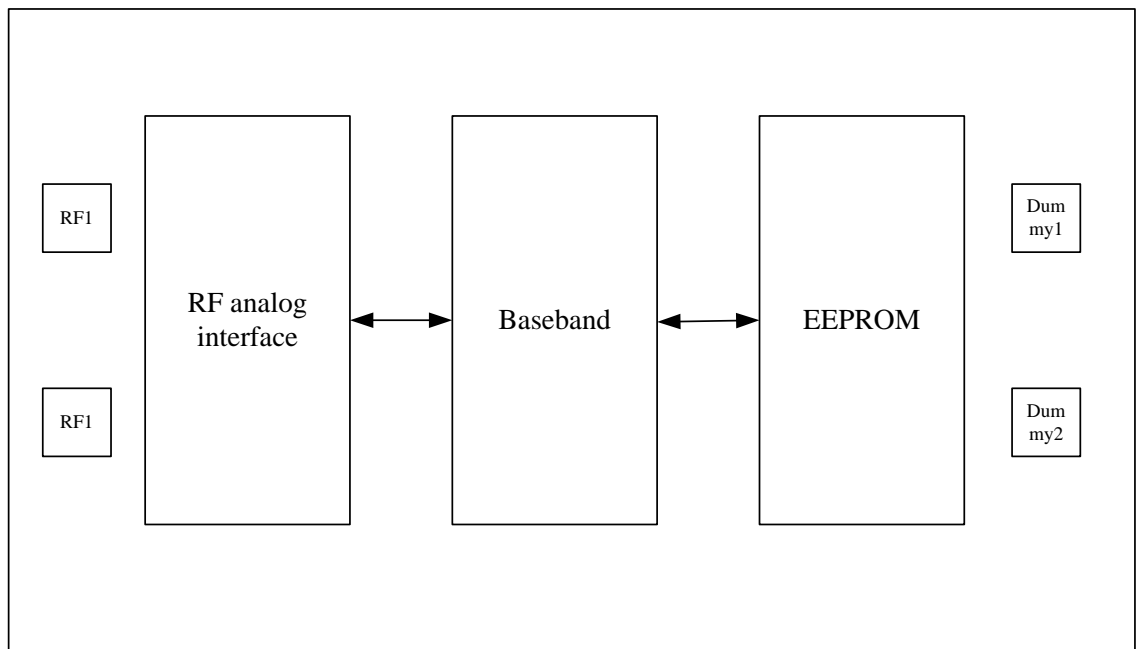


Fig.1 AIT-C01BCB chip block description

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_h to 1F_h; the access password shall be stored at memory addresses 20_h to 3F_h.
- **EPC memory** shall contain a CRC-16 at memory addresses 00_h to 0F_h, Protocol-Control (PC) bits at

memory addresses 10_h to 1F_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_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_h to 07_h. TID memory shall contain sufficient identifying information above 07_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_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:

00₂ Reserved

01₂ EPC

10₂ TID

11₂ 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.

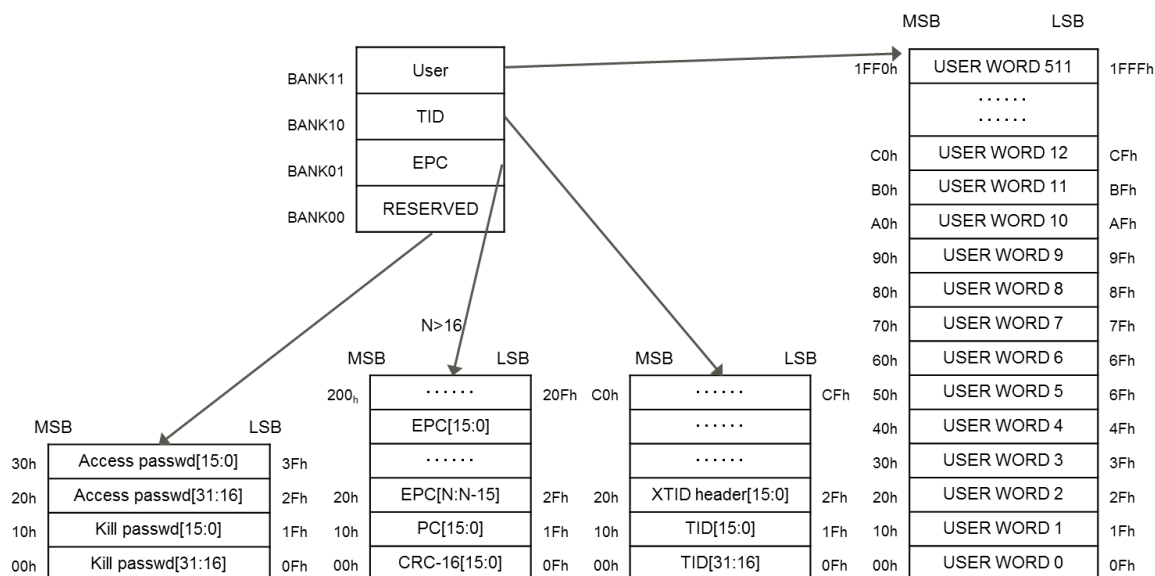


Fig 2. Logical Memory Map

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 00_h to 0F_h, PC bits at memory locations 10h to 1Fh, an EPC beginning at address 20_h, 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 00_h to 0F_h, 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 T_s or T_{hs} , 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:

00000_2 : Zero word.

00001_2 : One word (addresses 20_h to $2F_h$ in EPC memory).

00010_2 : Two words (addresses 20_h to $3F_h$ in EPC memory).

●
●
●

11111_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_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_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_h.

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

- Bits 17_h – 1F_h: A numbering system identifier (NSI) whose default value is 000000000₂. The MSB of the NSI is stored in memory location 17_h. If bit 17_h contains a logical 0, then a Tag is referred to as an EPCglobal™ Tag and PC bits 18_h – 1F_h shall be as defined in the EPC™ Tag Data Standards. If bit 17_h contains a logical 1, then a Tag is referred to as a non-EPCglobal™ Tag and PC bits 18_h – 1F_h shall contain the entire AFI defined in ISO/IEC 15961.

The default (unprogrammed) PC value shall be 3000_h.

During truncated replies a Tag substitutes 00000₂ 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.

Field	Memory Bank Bit Address	Value	Bit Number															
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Use Memory and BlockPermaLock	C0 _h - CF _h	0000 _h	Use Memory and BlockPermaLock Segment [15:0]															
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	B0 _h - BF _h	0000 _h	Use Memory and BlockPermaLock Segment [31:16]															
0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
BlockWrite and BlockErase	A0 _h - AF _h	0000 _h	BlockWrite and BlockErase Segment [15:0]															
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	90 _h - 9F _h	0000 _h	BlockWrite and BlockErase Segment [31:16]															
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	80 _h - 8F _h	0000 _h	BlockWrite and BlockErase Segment [47:32]															
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	70 _h - 7F _h	0000 _h	BlockWrite and BlockErase Segment [63:48]															
0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Optional Command Support	60 _h - 6F _h	0000 _h	Optional Command Support Segment [15:0]															
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Serial Number (48 bits)	50 _h - 5F _h	0000 _h	Serial Number [15:0]															
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	40 _h - 4F _h	xxxx	Serial Number [31:16]															
			X Location [7:0]								Y Location [7:0]							
	30 _h - 3F _h	xxxx	Serial Number [47:32]															
			Wafer Lot number [10:0]										Wafer number [4:0]					
XTID Header	20 _h - 2F _h	2000 _h	Extended TID Header															
			0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Extended Tag Identification	10 _h - 1F _h	0042 _h	MDID [3:0]				Tag Model Number											
			0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
	00 _h - 0F _h	E281 _h	Class Identification E2 _h								MDID [11:4]							
			1	1	1	0	0	0	1	0	1	0	0	0	0	0	0	1

■ Fujitsu MDID = 810_h

■ Tag Model Number = 042_h

■ Serialization = 48 + (1 - 1) * 16 = 48 bits = 3 words

Table 3.1. TID memory map

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 03_h to 07_h) shall have the default value 00000₂.

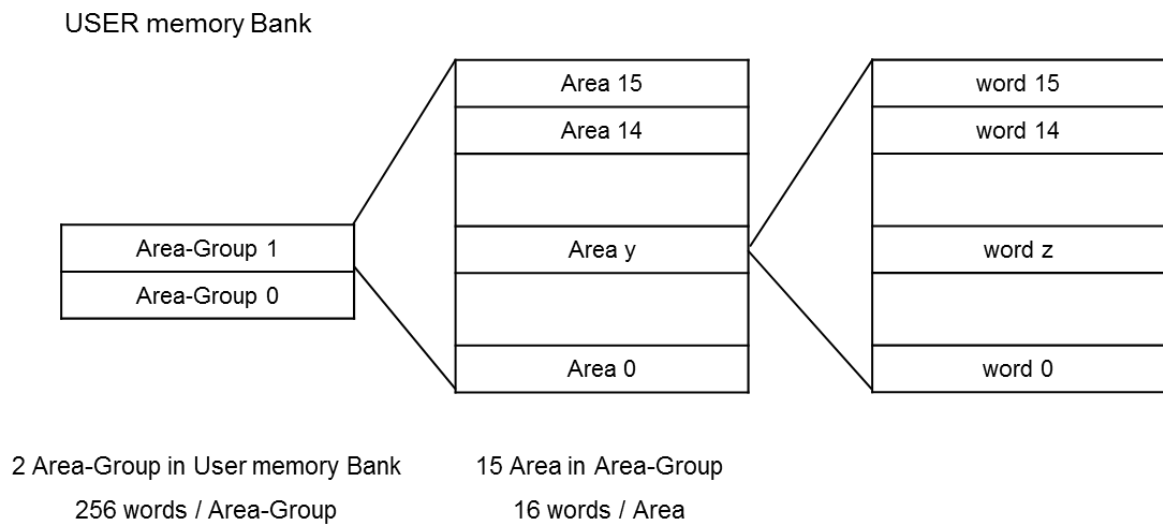


Fig.3 User memory architecture (8kbyte mode)

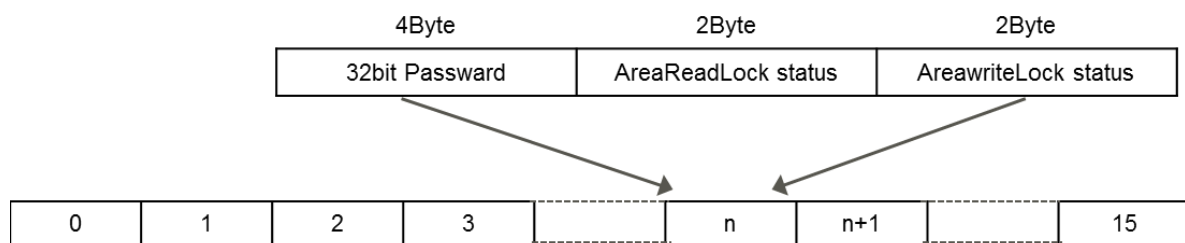


Fig.4 Control memory for Block n

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)

NAK 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 = 00_h specifies the first 16-bit memory word, WordPtr = 01_h 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 = 00_h the tag shall ignore the BlockWrite. If WordCount = 01_h 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 00_h to 0f_h 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 T_{RExt}=1 regardless of the T_{RExt} 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

	Command	MemBank	WordPtr	WordCount	Data	RN	CRC-16
# of bits	8	2	EBV	8	Variable	16	16
description	11000111	00: Reserved 01: EPC 10: TID 11: User	Starting address pointer	Number of words to write	Data to be written	<u>handle</u>	

Table 3.2. Blockwrite command

	Header	RN	CRC-16
# of bits	1	16	16
description	0	<u>handle</u>	

Table 3.3. Tag reply to a successful BlockWrite comand

3.13.14 BlockErase(Implement in AIT-C01BCB)

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

BlockErase allows an Interrogator to erase multiple words into 0_h 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 = 00_h specifies the first 16-bit memory word, WordPtr = 01_h 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 = 00_h the tag shall ignore the BlockErase. If WordCount = 01_h 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 00_h to 0f_h 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 T_{RExt}=1 regardless of the T_{RExt} 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.

	Command	MemBank	WordPtr	WordCount	RN	CRC-16
# of bits	8	2	EBV	8	16	16
description	11001000	00: Reserved 01: EPC 10: TID 11: User	Starting address pointer	Number of words to erase	<u>handle</u>	

Table 3.4 Blockerase command

	Header	RN	CRC-16
# of bits	1	16	16
description	0	<u>handle</u>	

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

First Command		Second Command		Tag Action and Response to 2 nd Command
<i>Lock</i>	pwd-write	permalock	<i>BlockPermalock</i> (<u>Read/Lock</u> = 1)	
	0	0		Permalock the blocks indicated by <u>Mask</u> ; respond as described in this section 6.3.2.11.3.9
	0	1		Reject the <i>BlockPermalock</i> ; respond with an error code
	1	0		Permalock the blocks indicated by <u>Mask</u> ; respond as described in this section 6.3.2.11.3.9
	1	1		Permalock the blocks indicated by <u>Mask</u> ; respond as described in this section 6.3.2.11.3.9
<i>BlockPermalock</i> (<u>Read/Lock</u> = 1)		pwd-write	permalock	
		0	0	Implement the <i>Lock</i> , but do not un-permalock any blocks that were previously permalocked; respond as described in 6.3.2.11.3.5
		0	1	Reject the <i>Lock</i> ; respond with an error code
		1	0	Implement the <i>Lock</i> , but do not un-permalock any blocks that were previously permalocked; respond as described in 6.3.2.11.3.5
		1	1	Implement the <i>Lock</i> ; respond as described in 6.3.2.11.3.5

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×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×BlockRange)-1.

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

- BlockRange specifies the range of Mask, starting at BlockPtr and ending (16×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 *BlocPermalock* command can permalock 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 wich memory blocks a Tag permalocks. Mask depends on the Read/Lock bit as follows:

- ◆ Read/Lock = 0: The Interrogator shall omit Mask from the *BlockPermalock* command.

BlockPermength 16x ck command.

- ◆ Read/Lock = 1 : The Interrogator shall include a Mask of length 16×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-permalocked.

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 Tags 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. The 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 manufacturer'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

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_{REPLY} or 20ms, where T_{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 T_1 , 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 T_{Rext} 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 T_1 , 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 T_{Rext} value in the *Query* that initiated the round.

Table 3.7 *BlockPermalock* command

	Command	RFU	MemBank	Read/Lock	BlockPtr	BlockRange	Mask	RN	CRC-16
# of bits	8	8	2	1	EBV	8	Variable	16	16
description	11001001	00 _h	00: RFU 01: EPC 10: TID 11: User	0: Read 1: Permalock	Mask start- ing address, specified in units of 16 blocks	Mask range, specified in units of 16 blocks	0: Retain current permalock set- ting 1: Assert perma- lock	<u>handle</u>	

Table 3.8 Tag reply to a successful BlockPermalock command with Read/Lock = 0

	Header	Data	RN	CRC-16
# of bits	1	Variable	16	16
description	0	Permalock bits	<u>handle</u>	

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

	Header	RN	CRC-16
# of bits	1	16	16
description	0	<u>handle</u>	

- **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 *Block-Permalock* completed successfully. The Tag's reply shall use the extended preamble, as appropriate (i.e. the Tag shall reply as if T_{Rext}=1 regardless of the T_{Rext} 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 T_{Rext}=1 regardless of the T_{Rext} 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 20ms 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 20ms 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 T_{Rext} = 1 regardless of the T_{Rext} value in the Query that initiated the round).

	Command	AreaGroupPtr	Data	Password	RN	CRC-16
# of bits	16	5	32	32	16	16
Description	1110 0000 0000 0100	AreaGroupPtr	New Passwd	Current Passwd	Handle	

Table 3.10. ChgAreaGroupPwd command

	Header	RN	CRC-16
# of bits	1	16	16
Description	0	Handle	

Table 3.11. Tag reply to ChgAreaGroupPwd command

3.13.17 AreaReadLock (Custom Command)

AIT-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: AIT-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

0 : Ignore the associated Action field and retain the current setting.

1 : Implement the associated Action field and overwrite the current AreaReadLock setting.

*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 RW transmit CW for the lesser of Treply or 20ms, where Treply is the time between the RW's AreaReadLock command and Tag's backscattered reply. The RW 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 R/W 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 R/W does not observe a reply within 20ms then the AreaReadLock 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 field, 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 T_{Rext} = 1 regardless of the T_{Rext} 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.

	Command	AreaGroupPtr	Payload	Password	RN	CRC-16
# of bits	16	5	32	32	16	16
Description	1110 0000 0000 0111	AreaGroupPtr	Mask/Action (see Table3.13)	32bit password	Handle	

Table 3.12. AreaReadLock command

Payload																									
MASK													Action												
0	1	2	3	4	5	6	7	8	9	10	11	12	0	1	2	3	4	5	6	7	8	9	10	11	12

Table3.13. AreaReadLock command Payload

	Header	RN	CRC-16
# of bits	1	16	16
Description	0	Handle	

Table 3.14. Tag reply to AreaReadLock command

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 a AreaWriteLock, depending on the success or failure of the Tag's memory lock operation.

- AreaWriteLock succeeds: After complete the AreaWriteLock, the Tag return 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 T_{Rext} = 1 regardless of the T_{Rext} 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

	Command	AreaGroupPtr	Payload	Password	RN	CRC-16
# of bits	16	5	32	32	16	16
Description	1110 0000 0000 1000	AreaGroupPtr	Mask/Action (see Table3.16)	32bit password	Handle	

Table 3.15. AreaWriteLock command

Payload																															
MASK																Action															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Table3.16. AreaWriteLock command Payload

	Header	RN	CRC-16
# of bits	1	16	16
Description	0	Handle	

Table3.17. Tag reply to AreaWriteLock command

Pwd write	Perma lock	Area Write Lock	Block Perma Lock	WRITE Operation	
				Secure state	Open state
0	0	0	0	Enable	Enable
		1	X	Disable	Disable
		X	1	Disable	Disable
0	1	0	0	Enable	Enable
		1	X	Disable	Disable
		X	1	Impossible to transit to this state	(Same as left)
1	0	0	0	Enable	Disable
		1	X	Disable	Disable
		X	1	Disable	Disable
1	1	X	X	Disable	Disable

Table 3.18 Write operation on User Memory

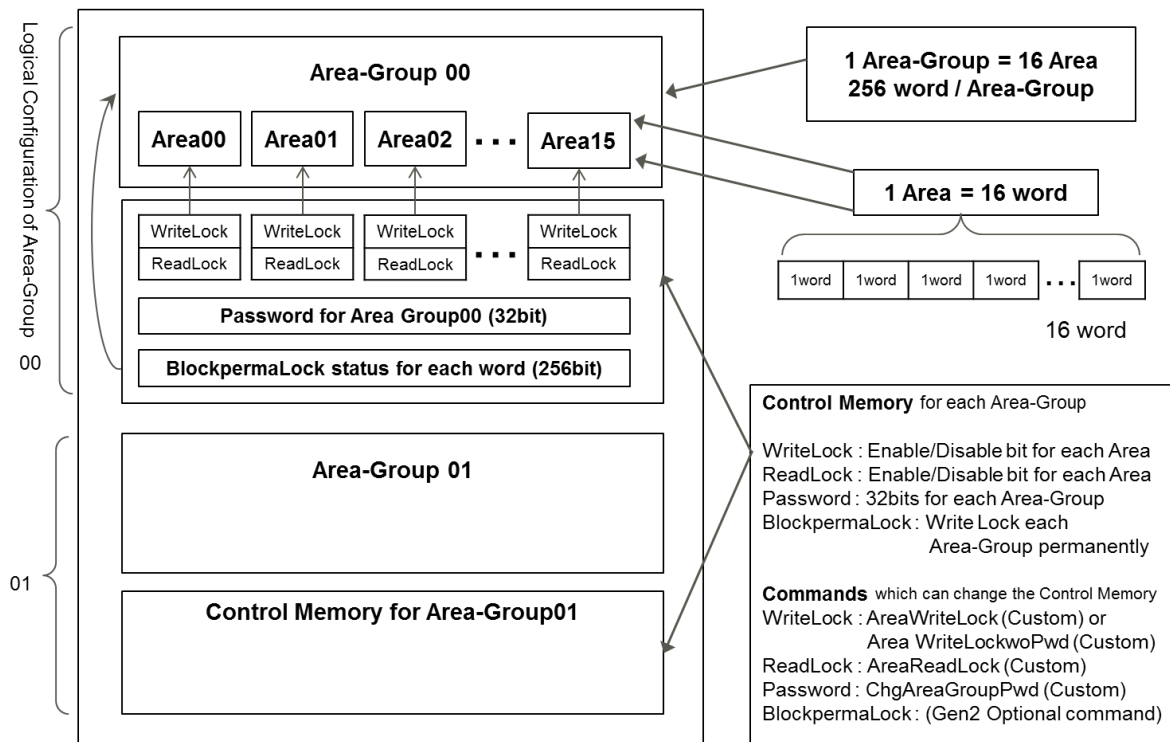


Fig.5 User memory control function and relationship

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 :

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 T_{Rext} = 1 regardless of the T_{Rext} 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

	Command	AreaGroupPtr	Payload	RN	CRC-16
# of bits	16	5	16	16	16
Description	1110 0000 0000 1001	AreaGroupPtr	Action (see Table3.20)	Handle	

Table 3.19. AreaWriteLockwoPwd command

Action															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Table3.20. AreaWriteLockwoPwd command Payload

	Header	RN	CRC-16
# of bits	1	16	16
Description	0	Handle	

Table3.21. Tag reply to AreaWriteLockwoPwd command

3.13.20 ChgCustomPwd (Custom Command)

AIT-C01BCB has the change Custom password function. Table 3.22 shows the format of the ChgCustomPwd command. The RW can use ChgCustomPwd to change the password of Custom. To change the password of Custom, the RW 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 T_{Rext} = 1 regardless of the T_{Rext}

value in the Query that initiated the round).

	Command	Data	Password	RN	CRC-16
# of bits	16	32	32	16	16
Description	1110 0000 0000 1100	New Passwd	Current Passwd	Handle	

Table 3.22 ChgCustomPwd command

	Header	RN	CRC-16
# of bits	1	16	16
Description	0	Handle	

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)