ATID Co.,Ltd

XC1003-1 RFID API Reference Guide for Android Developers

Android Developer Guide

YH.PARK

2015-07-02

Document change record

Version	Date	Revision[1]	Description of change[2]	Writer
V1.0	2015-04-23	Draft	Draft Document	YH.PARK
V2.0	2015-07-02	ADD	Add Access method	YH.PARK

Revision : Define the contents are addition/modification/deletion

Description : Describe revised page number and contents

Table of Contents

1. Intro

This Demo Guide explains how to use XC1003-2 RFID SDK Library to develop Android application program. The development tool used Eclipse Juno and the supported development platform is Android 4.4.2

Library	Description
atid.dev.rfid	Android Library for controlling RFID device of XC1003
atid.system.comm	Android Library for controlling the communication between XC1003-1 and Device.
atid.system.ctrl	Android Library for control the XC1003 Device power
atid.system.device	Android Library for managing XC1003 Device
atid.util	Utility Library for using inside of XC1003-1 SDK Library

2. Reference Library Guide

2.1. ATRfidManager Class

ATRfidManager is a class which the developers can use the manage / control instance creation and resources between activities of XC1003-1 RFID

2.1.1. Methord

2.1.1.1. getInstance

- **Created RFID Reader Object and connect XC1003-1 device to RFID reader Object**

  Syntax

```
public static ATRfidReader getInstance()
```

- **Return value**

  Return RFID Reader's Instance

- **Remarks**
- getInstance method creates and returns the instance when the result is successful
- call the Main Activity's onCreate method

2.1.1.2. onDestroy

getInstance method disconnects XC1003 device and Release object

- **Syntax**

```
public static void onDestroy()
```

- **Remarks**

  onDestroy method makes free resource of RFID Reader Object and perform release.

  And call Main Activity's onDestroy method

2.1.1.3. wakeUp

wakeUp method is for calling XC1003 wakeup from sleep mode

- **Syntax**

```
public static void wakeUp()
```

- **Remarks**
- Call all App Activity's onStart method. After calling sleep method then if it is not calling wakeup method, XC1003 RFID Device will not work.

2.1.1.4. sleep

Right before call sleep method from XC1003 wakeup mode sleep.

.

- **Syntax**

```
public static void sleep()
```

- **Remarks**

Calling all the Application activity onStop method.

If it is not calling the sleep onSTOP, even it is going on the Sleep mode by XC1003 device power button, the RFID device is still working

2.2. ATRfidReader Class

ATRfidReader Class creates the Instance of RFID and sets the RFID reading & Configuration to Instance.

2.2.1. Methord

2.2.1.1. destroy

Destroy methord is for destruction of ATRfidReader's instance forcibly

- **Syntax**

```
public void destroy()
```

- **Remarks**

  It is not necessary to call separately, because it is called from ATRfidManager

2.2.1.2. connect

Connect method connects XC1003 Module

- **Syntax**

```
public boolean connect()
```

- **Return value**

  When the connection is successful, it returns "true' and when unsuccessful, it returns 'false'

- **Remarks**
- **It is running only once to connect RFID device**

2.2.1.3. disconnect

Disconnect method disconnects XC1003 module

- **Syntax**

```
public void disconnect()
```

- **Remarks**

  Disconnection occurs once when the device is disconnected

2.2.1.4. readEpc6cTag

readEpc6cTag method performs the reading ISO18000-6C in Single mode by Inventory

- **Syntax**

```
public ResultCode readEpc6cTag()
```

- **Return value**

  Return ResultCode enumertation type about the result of order word

- **Remarks**

  readEpc6ctag method performs inventory funtion by ISO18000-6C Tag with XC1003-1 module. If the tag is read normally, onReaderReadTag event is occurred

2.2.1.5. inventory6cTag

inventory6cTag method performs the reading of ISO18000-6C in Multiple mode by inventory function

- **Syntax**

```
public ResultCode inventory6cTag()
```

- **Return value**

  Return ResultCode enumertation type about result of order word

- **Remarks**
- onReaderDetectTag event is created when the reading ISO18000-6C Tag by inventory function in inventory6cTagmethod is successfully carried out.

2.2.1.6. readMemory6c

readMemory6C method performs Read memory function to ISO 18000-6C Tag

- **Syntax**

```
public ResultCode readMemory6c(BankType bank, int offset, int length)
public ResultCode readMemory6c(BankType bank, int offset, int length, String
password)
```

- **Parameters**

  bank : Specify Tag Memroy Bank

  offset : Specify start address of Tag Data in WORD unit

  length : Specify the Tag Data length in WORD unit

  password : Specify Tag's Access Password in 4Byte Hex strings

  Return value

  Return result of order word in ResultCode enumeration

  Remarks

  readMemory6c method performs Read memory funtion in ISO18000-6C Tag on XC1003-1 module, and if it is normal performance, the result will be transferred by onReaderResult event

2.2.1.7. writeMemory6c

writeMemory6C method performs Write memory function in ISO 18000-6C Tag

- **Syntax**

```
public ResultCode writeMemory6c(BankType bank, int offset, String data)
public ResultCode writeMemory6c(BankType bank, int offset, String data, String password)
```

- **Parameters**

  bank : Specify Tag Memroy Bank.

  offset : Specify start address of Tag Data in WORD unit

  data : Specify the Tag Data length in WORD unit

  password : Specify Tag's Access Password in 4Byte Hex strings

- **Return value**

  Return result of order word in ResultCode enumeration

- **Remarks**

- WriteMemory6c method performs write memory function in ISO18000-6C Tag on XC1003-1 module, and if it is normal performance, the result will be transferred by onReaderResult event

2.2.1.8. lock6c

Lock6C method performs Lock function in ISO18000-6C Tag

- **Syntax**

```
public ResultCode lock6c(LockParam param)
public ResultCode lock6c(LockParam param, String password)
```

- **Parameters**

  param: information about the memory bank area to perform LockParam class of instance

  password: Indicated the 4Byte Hex strings about tag's access password

- **Return value**

  Return the ResultCode enumeration of order performance result

  Remarks

  Lock6c method performs lock function in ISO18000-6C Tag on XC1003 module

  And if it is normal performance, the result will be transferred by onReaderResult event

2.2.1.9. kill6c

kill6C method performs Kill function in ISO18000-6C Tag

Syntax

```
public ResultCode kill6c(String password)
```

- **Parameters**

  password : Indicated the 4Byte Hex strings about tag's access password

- **Return value**
- Return the ResultCode enumeration of order performance result
- **Remarks**

  Kill6c method performs Kill function in ISO18000-6C Tag on XC1003 module

  And if it is normal performance, the result will be transferred by onReaderResult event

2.2.1.10. Stop

Stop method stops all Action type of functions (Inventory, Read / Write Memory, Lock, Kill)

- **Syntax**

```
public ResultCode stop()
```

- **Return value**

  Return result of order word in ResultCode enumeration

- **Remarks**

  To cancel all working operations and changed the state to stop state of XC1003's module

2.2.1.11. defaultProperties

defaultProperties method returns values for all properties to their initial values

- **Syntax**

```
public ResultCode defaultProperties()
```

- **Return value**
- **Return the command execution result by resultCode enumeration**
- **Remarks**
- **Defaultproperies method sets all property values are set in the XC1003 module to factory default values.**

2.2.1.12. getFirmewareVersion

Return the firmware version of XC1003 module

- **Syntax**

```
public String getFirmwareVersion() throws ATRfidReaderException
```

- **Return value**

  Returns Firmware version as string.

- **Remarks**

  getfirmwareVersion method should use after calling getInstance method

2.2.1.13. **getState**

getState method returns connection status between Reader Object and XC1003 module

- **Syntax**

```
public ConnectionState getState()
```

- **Return value**

  It returns connection condition, for more detailed in ConnectionState

- **Remarks**

  getfState method should use after calling getInstance method

2.2.1.14. **getAction**

getState method returns perform condition of XC1003 module.

- **Syntax**

```
public ActionState getAction()
```

- **Return value**

  It returns XC1003 module performance condition, for more detailed in ActionState

- **Remarks**

  getAction method should use after calling getInstance method

2.2.1.15. **getOperationTime**

getInventoryTime method returns operation time of XC1003 module

- **Syntax**

```
public int getOperationTime() throws ATRfidReaderException
```

- **Return value**

  Return module operation time as ms units

- **Remarks**

  Returns module's operation time as integer

  And the time is set up the ms units.

  If this values is set up the 0, it will be working until order "STOP" to module

2.2.1.16. **setOperationTime**

setOperationTime method is set up the operation time of XC1003 Module.

Syntax

```
public void setOperationTime(int time) throws ATRfidReaderException
```

- **Parameters**

  time : Module operation time which is ms unit

- **Remarks**

  Returns module's operation time as integer

  And the time is set up the ms units.

  If this values is set up the 0, it will be working until order "STOP" to module

2.2.1.17. getPowerRange

Returns XC1003 module's antenna output of power level as minimum, Maximum

- **Syntax**

```
public RangeValue getPowerRange() throws ATRfidReaderException
```

- **Return value**
- **Return RangeValue Class's instance which has Minimum and Maximum value of XC1003 module antenna output power**

- **Remarks**

  It changed by XC1003 module's antenna

2.2.1.18. getPower

Return XC1003 module's antenna output power level

- **Syntax**

```
public int getPower() throws ATRfidReaderException
```

- **Return value**

  This output level of Antenna multiply 10 integer values

- **Remarks**

  The set up value is Antenna output value * 10

  If you want set up the Antenna output value is 30dbm

  You need set up 30 * 10 = 300

  Value that can be set between maximum value and minimum value that is returned by the getPowerRange method.

2.2.1.19. **setPower**

setPower method is set up the XC1003 module's antenna output power level .

- **Syntax**

```
public void setPower(int power) throws ATRfidReaderException
```

- **Parameters**

Power: This output level of Antenna multiply 10 integer values

- **Remarks**

The set up value is Antenna output value * 10

If you want set up the Antenna output value is 30dbm

You need set up 30 * 10 = 300

Value that can be set between maximum value and minimum value that is returned by the getPowerRange method.

2.2.1.20. **getInventoryTime**

getInventoryTime method returns time of antenna activation during the time of XC1003 module's inventory round.

- **Syntax**

```
public int getInventoryTime() throws ATRfidReaderException
```

- **Return value**

Return antenna activation time (ms unit)

- **Remarks**

RFID module has the inventory time and idle time for one inventory Round time.

Inventory Round time is maximum 400ms.

Together with the inventory time and idle time cannot exceed the inventory Round time.

2.2.1.21. **setInventoryTime**

Setinventorytime method set up the time of antenna activation during the time of XC1003 module's inventory round.

- **Syntax**

```
public void setInventoryTime(int time) throws ATRfidReaderException
```

- **Parameters**

time : antenna activation time (ms unit)

- **Remarks**

For your information in getInventoryTime.

2.2.1.22. **getIdleTime**

getIdletime method returns the idle time from the inventory Round antenna hours of XC1003 moudle

- **Syntax**

```
public int getIdleTime() throws ATRfidReaderException
```

- **Return value**

  Return the idle time of the Antenna (ms unit)

- **Remarks**

  RFID module has the inventory time and idle time for one inventory Round time.

  Inventory Round time is maximum 400ms.

- Together with the inventory time and idle time cannot exceed the inventory Round time.

2.2.1.23. **setIdleTime**

SetIdletime method set up the time of antenna activation during the time of XC1003 module's inventory round.

- **Syntax**

```
public void setIdleTime(int time) throws ATRfidReaderException
```

- **Parameters**

  time : antenna activation time (ms unit)

- **Remarks**
- **For more detailed on getIdle Time**

2.2.1.24. **getReportRssi**

getReportRssi method returns whether report XC1003 module as an RSSI value with the EPC values while performing the inventory.

- **Syntax**

```
public boolean getReportRssi() throws ATRfidReaderException
```

- **Return value**

  Boolen type that determines whether the RSSI value reported.

- **Remarks**

  If the Return value is true, it returns ReaderReadTag event.

  RSSI value is reported in Data.

2.2.1.25. **setReportRssi**

SetReportRssi method returns whether report XC1003 module as an RSSI value with the EPC values while performing the inventory.

- **Syntax**

```
public void setReportRssi(boolean enabled) throws ATRfidReaderException
```

- **Parameters**

enabled : Boolen type that determines whether the RSSI value reported.

- **Remarks**

If the set up value is true, it returns ReaderReadTag event.

RSSI value is reported in Data.

2.2.1.26. getInventorySession

getinventroySeetion method returns Tag session which is using performed inventory of XC1003 module.

- **Syntax**

```
public InventorySession getInventorySession() throws ATRfidReaderException
```

- **Return value**

InventorySession enumeration indication the tag session

2.2.1.27. setInventorySession

SetomvemtprySession method sets the session of the tag to use, during performing the inventory of XC1003 module.

Syntax

```
public void setInventorySession(InventorySession session) throws
ATRfidReaderException
```

- **Parameters**

session : InventorySession enumeration indication the tag session

2.2.1.28. getInventoryTarget

getomvemtprySession method returns the session of the tag to use, during performing the inventory of XC1003 module.

- **Syntax**

```
public InventoryTarget getInventoryTarget() throws ATRfidReaderException
```

- **Return value**

InventorySession enumeration indication the tag session

2.2.1.29. **setInventoryTarget**

setinventoryTarget method set up the session of the tag to use, during performing the inventory of XC1003 module.

- **Syntax**

```
public void setInventoryTarget(InventoryTarget target) throws
ATRfidReaderException
```

- **Parameters**

target : InventorySession enumeration indication the tag session

2.2.1.30. **getUseSelectionMask**

getUseSelectionMask method returns XC1003 module whether to use the selection Mask while performing the inventory Access or other commands .

- **Syntax**

```
public boolean getUseSelectionMask() throws ATRfidReaderException
```

- **Return value**

Selection Mask boolen type to determine whether or not to use.

- **Remarks**

If the return value is true, selection Mask set to use the inventory feature performs command and Access.

2.2.1.31. **setUseSelectionMask**

setUseSelectionMask method set up the XC1003 module whether to use the selection Mask while performing the inventory Access or other commands .

- **Syntax**

```
public void setUseSelectionMask(boolean used) throws ATRfidReaderException
```

- **Parameters**

enabled : selection Mask boolen type to determine whether or not to use

- **Remarks**
- If set up value is true, selection Mask set to use the inventory feature performs command and Access.

2.2.1.32. **getSelectionMask6c**

getSelectionMask6c method returns selection mask value which set up on XC1003 module.

- **Syntax**

```
public SelectionMask6c getSelectionMask6c(int index) throws ATRfidReaderException
```

- **Parameters**

index : Index of Selection Mask arrangement which is for return (0 ~ 7)

- **Return value**

SelectionMask6C Class instance which has selection mask information to set on the specified index

- **Remarks**

XC1003 module can set up selection Mask to maximum 8

2.2.1.33. **setSelectionMask6c**

setSleectionMask6c method set up the Selection Mask on XC1003 module

- **Syntax**

```
public void setSelectionMask6c(int index, SelectionMask6c mask) throws
ATRfidReaderException
```

- **Parameters**

index : Index of Selection Mask arrangement which is for return (0 ~ 7)

mask : SelectionMask6C Class instance which has selection mask information to set on the specified index

Remarks

XC1003 module can set up selection Mask to maximum 8

2.2.1.34. **setEventListener**

Set up for using event in the application.

- **Syntax**

```
public void setEventListener(RfidReaderEventListener listener)
```

- **Parameters**

listener : The interface was created to handle specific events (RFID) in the application.

- **Remarks**
- **setEventListener method should use after calling getinstance method.**

2.2.1.35. **removeEventListener**

To disable an event in the application.

- **Syntax**

```
public void removeEventListener(RfidReaderEventListener listener)
public void removeEventListener()
```

- **Parameters**

listener :

The interface was created to handle specific events (RFID) in application

2.3. Evnet Handler

2.3.1.1. onReaderStateChanged

onReaderStateChange method returns the connection status of XC1003 module..

Returtn

- **Syntax**

```
void onReaderStateChanged(ATRfidReader reader, ConnectionState state);
```

- **Parameters**

  state : ConnectionState enumeration indicating the connection status of RFID.

- **Remarks**

  If the connection status is changed, it calls from Reader object which connect XC1003 module.

2.3.1.2. onReaderActionChanged

onReaderActionChange method returns the connection status of XC1003 module..

- **Syntax**

```
void onReaderActionChanged(ATRfidReader reader, ActionState action);
```

- **Parameters**

  state : ActionState enumeration is indicating the operating status of RFID

  Remarks

  If the connection status is changed, it calls from Reader object which connect XC1003 module.

2.3.1.3. onReaderReadTag.

onReaderReadTag method returns Tag EPC value which is reading by readEpc6cTag method or inventory6c method

- **Syntax**

```
void onReaderReadTag(ATRfidReader reader, String tag, float rssi);
```

- **Parameters**

  tag : Hex type of string which is indicating Tag EPC by reading inventory.

  rssi : Float Value that represents the RSSI values

- **Remarks**

  If XC1003 is reading EPC data of Tag by Inventory function,

  It calls from Reader object

2.3.1.4. onReaderResult

onReaderResult method returns result of Access performance command as Read memory , Write memory , Lock or Kill etc.

- **Syntax**

```
void onReaderResult(ATRfidReader reader, ResultCode code, ActionState action, String epc, String data);
```

- **Parameters**

code : Result Code enumeration which is Indicated about result of Access performance command

action : ActionState Enumeration which is indicating the access command performed.

epc : Hex type of string which is indicating EPC data of Access Tag

data : Hex type of string which is representing the tag ,if it perform access command is ReadMemory

- **Remarks**

If it performed which is related Access command, it calls from Reader Object which connect XC1003

2.4. Parameter Classes

2.4.1. RangeValue Class

2.4.1.1. Constructor

Initialized a new instance of RangeValue in the range

- **Syntax**

```
public RangeValue()
public RangeValue(int min, int max)
```

- **Parameters**

min : Integer representing the minimum value

max : integer representing the maximum value

Remarks

Value is used in the range that is returned from getPowerRange etc

2.4.1.2. Property Methods

2.4.1.2.1. getMin()

Returns minimum value.

- **Syntax**

```
public int getMin()
```

- **Return value**

Integer representing minimum set in the instance.

2.4.1.2.2. getMax()

Returns maximum value.

- **Syntax**

```
public int getMax()
```

- **Return value**

Integer represents the maximum values set in the instance.

2.4.2. LockParam Class

2.4.2.1. Constructor

Initializes new instance of LockParam in the range.

- **Syntax**

```
public LockParam()
public LockParam(LockType killPassword, LockType accessPassword,
                 LockType epc, LockType tid, LockType user)
```

- **Parameters**

killPassword :

Lock type enumeration indicating the operation of the Kill password Lock area.

accessPassword : Locktype enumeration indicating the operation of the Access password Lock area.

epc : LockType enumeration indicating the Lock the operation of EPC Memory Bank area.

tid : TID memory Bank lock Type enumeration indicating the Lock operation of the area.

user : LockType enumeration indicating the Lock user memory Bank operation in the region .

- **Remarks**

  To use Lock6c method parameter.

2.4.2.2. Property Methods

2.4.2.2.1. getKillPassword

Returns Lock perform of Kill Password area.

- **Syntax**

```
public LockType getKillPassword()
```

- **Return value**

  LockType enumeration indicating the operation of the Kill password Lock area.

2.4.2.2.2. setKillPassword

Set up the Lock perform of Kill Password area.

- **Syntax**

```
public void setKillPassword(LockType killPassword)
```

- **Parameters**

  killPassword :.LockType enumeration indicating the operation of the Kill password Lock area.

2.4.2.2.3. getAccessPassword

Returns Lock perform about Access password area .

- **Syntax**

```
public LockType getAccessPassword()
```

- **Return value**

  LockType enumeration indicating the Lock operation in the Access Password area.

2.4.2.2.4. setAccessPassword

Set up the Lock perform in Access password area.

- **Syntax**

```
public void setAccessPassword(LockType accessPassword)
```

- **Parameters**

accessPassword : Locktype enumeration indication the operation of the Access Password Lock area.

2.4.2.2.5. getEPC

Return Lock perform of EPC memory bank area.

- **Syntax**

```
public LockType getEPC()
```

- **Return value**

Locktype enumeration indicated Lock operation of the EPC memory bank area.

2.4.2.2.6. setEPC

Set up the Lock performed of EPC memory bank area.

- **Syntax**

```
public void setEPC(LockType epc)
```

- **Parameters**

epc : Locktype enumeration indicated Lock operation of EPC Memory Bank area.

getTID

Return Lock operation of TID Memory bank area.

- **Syntax**

```
public LockType getTID()
```

- **Return value**

LockType enumeration indicated Lock operation of TID memory bank area.

2.4.2.2.7. setTID

Set up the Lock operation of TID memory bank area.

- **Syntax**

```
public void setTID(LockType tid)
```

- **Parameters**

tid : LockType Enumeration indicated Lock operation of TID Memory bank area.

2.4.2.2.8. getUser

Returns Lock operation of User Memory bank area.

- **Syntax**

```
public LockType getUser()
```

- **Return value**

Locktype enumeration indicated Lock operation of user Memory bank area.

2.4.2.2.9. setUser

Set up the Lock operation of User Memory Bank area.

Syntax

```
public void setUser(LockType user)
```

- **Parameters**

user : Lock type enumeration indicated Lock operation of User Memory Bank area.

2.4.3. SelectionMask6c Class

2.4.3.1. Constructor

Initializes a new instance of SelectionMask6c Class represents the selection mask.

- **Syntax**

```
public SelectionMask6c()
public SelectionMask6c(MaskTargetType target, MaskActionType action,
                       BankType bank, int pointer, int length, String mask,
                       boolean truncate)
```

- **Parameters**

target : MaskTargetType enumeration indicating that the Session of the Tag mask destination.

action : MaskAtionType enumeration to determine the Ssession Settings for mask condition.

bank : BankType enumeration indicating the Memory bank which are subject of the Tag mask condition.

pointer : Interger representing the starting address to begin comparing the Mask value. (bit unit)

length : integer value specifying the length to compare the mask. (bit unit)

mask : Hex type string value that represents the Mask.

truncate : Boolean indicating whether or not cut to length Mask value

- **Remarks**

To use in the getSelectionMask6c or setSelectionMask6c method.

2.4.3.2. Property Methods

2.4.3.2.1. isUsed

Returns whether or not to use the currently set conditions Selection Mask.

- **Syntax**

```
public boolean isUsed()
```

- **Return value**

Boolean indicating whether to use the Selection Mask information.

2.4.3.2.2. setUsed

Whether or not to use the currently set conditions Selection Mask

- **Syntax**

```
public void setUsed(boolean used)
```

- **Parameters**

used : Boolean indication whether to use the selection mask information.

2.4.3.2.3. getTarget

Return of the session selection mask are subject tag.

- **Syntax**

```
public MaskTargetType getTarget()
```

- **Return value**

MaskTagettype enumeration indicating that the Session of the tag Mask destination.

2.4.3.2.4. setTarget

Set session of the Selection Mask tag that subject.

- **Syntax**

```
public void setTarget(MaskTargetType target)
```

- **Parameters**

target : MaskTargetType enumeration indicating that the Session of the Tag Mask destination

2.4.3.2.5. getAction

Returns way of session setting about the Selection mask.

- **Syntax**

```
public MaskActionType getAction()
```

- **Return value**

MaskAction type enumeration to determine the Session settings for Mask condition.

2.4.3.2.6. setAction

How to set the settings for Selection mask of Session condition.

- **Syntax**

```
public void setAction(MaskActionType action)
```

- **Parameters**

action : MaskAction type enumeration to determine the Session settings for Mask condition.

2.4.3.2.7. getBank

Return the Tag memory bank to be compared in the selection mask.

- **Syntax**

```
public BankType getBank()
```

- **Return value**

  Banktype enumeration indicating the memory bank which are the subject of the Tag mask condition.

2.4.3.2.8. setBank

Set up the Tag memory bank which is compare the Selection mask

- **Syntax**

```
public void setBank(BankType bank)
```

- **Parameters**
- **bank:** Bank type enumeration is indicating the memory bank which are the subject of the Tag mask condition.

2.4.3.2.9. getPointer

Return the Mask value of selection which begin to start compare address

- **Syntax**

```
public int getPointer()
```

- **Return value**

  Integer representing the starting address to begin comparing the mask value. (Bit unit)

2.4.3.2.10. setPointer

Set up the Mask value of selection which begin to start compare address.

- **Syntax**

```
public void setPointer(int pointer)
```

- **Parameters**
- **pointer: Integer representing the starting address to begin comparing the mask value (bit unit)**

2.4.3.2.11. getLength

Return the length compared to the value of the mask selection

- **Syntax**

```
public int getLength()
```

- **Return value**

  Integer value specify the length to compare the Mask (bit unit)

2.4.3.2.12. setLength

Set the length to compare the value of the mask selection Mask

- **Syntax**

```
public void setLength(int length)
```

- **Parameters**
- **length : Integer value specify the length to compare the Mask value (bit unit)**

2.4.3.2.13. getMask

Return mask value which is compare with Selection mask

- **Syntax**

```
public String getMask()
```

- **Return value**

  This is Hex type of string for shown mask value

2.4.3.2.14. setMask

Set up the Mask value which is compare with Selection mask

- **Syntax**

```
public void setMask(String mask)
```

- **Parameters**
- **mask : This is Hex type of string for shown mask value**

2.4.3.2.15. getTruncate

Return about cutting as Selection mask's Mask value length

- **Syntax**

```
public boolean getTruncate()
```

- **Return value**

  This is Boolean type to ask cut or not as Mask value's length

2.4.3.2.16. setTruncate

Set up for cutting or not about the selection mask' Mask value

- **Syntax**

```
public void setTruncate(boolean truncate)
```

- **Parameters**
- **truncate : This is Boolean type to ask cut or not as Mask value's length**

2.5. Enumerations

2.5.1. ActionState

Indicates the operation status of the current XC1003modul condition

Flag	Value	Description
Unknown	0x20	Unknown condition
Inventory6cMulti	0x66	Multi Inventory 6C Type
Inventory6cSingle	0x65	Single Inventory 6C Type
ReadMemory6c	0x72	Read Memory 6C Type
WriteMemory6c	0x77	Write Memory 6C Type
Lock	0x6C	Lock Tag
Kill	0x6B	Kill Tag
Stop	0x33	Stop

2.5.2. BankType

Indicated the Tag's Memory Bank.

Flag	Value	Description
Reserved	0	Readerved Bank
EPC	1	EPC Bank
TID	2	TID Bank
User	3	User Bank

2.5.3. ConnectionState

it shows XC1003's module and object reader connection status.

Flag	Value	Description
Disconnected	0	Disconnection
Connecting	2	Connecting
Connected	3	Connected

2.5.4. InventorySession

When it performs Inventory, Indicated the target Tag session

Flag	Value	Description
S0	0	Session 0
S1	1	Session 1
S2	2	Session 2

S3	3	Session 3

2.5.5. InventoryTarget

When it performs the Inventory, session which represents the state of the tag inventory destination. .

Flag	Value	Description
A	0	Session condition A
B	1	Session condition B
All	2	A or B

2.5.6. LockType

Indicated the Lock preformation in a specified area.

Flag	Value	Description
NoChange	0	Do not perform any operation.
Unlock	1	Perform Unlock
Lock	2	Perform Lock
PermaLock	3	Perform permanent Lock

2.5.7. MaskActionType

Flag	Value	Description
Assert_Deassert	0	**Matcing :** assert SL or inventoried → A **Not Matcing :** deassert SL or inventoried → B
Assert_DoNothing	1	**Matcing :** assert SL or inventoried → A **Not Matcing :** do nothing
DoNothing_Deassert	2	**Matcing :** do nothing **Not Matcing :** deassert SL or inventoried → B
Negate_DoNothing	3	**Matcing :** negate SL or (A → B, B → A) **Not Matcing :** do nothing
Deassert_Assert	4	**Matcing :** deassert SL or inventoried → B **Not Matcing :** assert SL or inventoried → A
Deassert_DoNothing	5	**Matcing :** deassert SL or inventoried → B **Not Matcing :** do nothing
DoNothing_Assert	6	**Matcing :** do nothing **Not Matcing :** assert SL or inventoried → A

DoNothing_Negate	7	**Matcing :** do nothing **Not Matcing :** negate SL or (A → B, B → A)

2.5.8. MaskTargetType

Flag	Value	Description
S0	0	Session 0
S1	1	Session 1
S2	2	Session 2
S3	3	Session 3
SL	4	Session Flag

2.5.9. ResultCode

Flag	Value	Description
NoError	0x0000	No error
OtherError	0x0001	Other error
Undefined	0x0002	Undefined
MemoryOverrun	0x0003	Memory overrun
MemoryLocked	0x0004	Memory locked
InsufficientPower	0x000B	Insufficient power
NonSpecificError	0x000F	Non-Specific error
InvalidResponse	0xE001	Invalid response
InOperation	0xE002	In operation
OutOfRange	0xE003	Out of range
NotConnected	0xE004	Disconnected
InvalidParameter	0xE010	Invalidate parameter
InvalidInstance	0xE100	Invalid instance
FailSendControlPacket	0xEE00	Failed to send control packet
FailReceivePacket	0xEE01	Failed to receive packet
InvalidControlResponse	0xEE02	Invalidate control response packet
UnknownControlResponse	0xEE0F	Unknown control response
InvalidRegisterParameter	0xEE10	Invalidate register parameter
InvalidRegisterResponse	0xEE11	Invalidate register response
UnknownRegisterResponse	0xEE12	Unknown register response

FailSendRegisterPacket	0xEE11	Failed to send register packet
NotSupported	0xEF00	Not Supported
Timeout	0xEFFF	Timeout
HandleMismatch	0xF001	Handle mismatch
CRCError	0xF002	CRC error on tag response
NoTagReply	0xF003	No tag reply
InvalidPassword	0xF004	Invalid password
ZeroKillPassword	0xF005	Zero kill password
TagLost	0xF006	Tag lost
CommandFormatError	0xF007	Command format error
ReadCountInvalid	0xF008	Read count invalid
OutOfRetries	0xF009	Out of retries
ParamError	0xFFFB	Parameter error
Busy	0xFFFC	Busy
InvalidCommand	0xFFFD	Invalid command
LowBattery	0xFFFE	Low battery
OperationFailed	0xFFFF	Operation failed

2.5.10. **ResultCode Enumerator**

Defines the result of the Action of XC1003 module

Flag	Value	Description
OtherError	0x00	Tag Error Code 0x00
Success	0x01	Indicates that the function successfully run
Undefined	0x02	Indicates that the function is not supported by the module
MemoryOverrun	0x03	Tag Error Code 0x03
MemoryLocked	0x04	Tag Error Code 0x04
Timeout	0x05	Indicates that the processing time for the function

		is over
InsufficientPower	0x0B	Tag Error Code 0x0B
NonSpecificError	0x0F	Tag Error Code 0x0F
AckHearteat	0xFF	