Audio Reader
PROGRAMMING REFERENCE MANUAL

PART NUMBER 99875582-1

SEPTEMBER 2012

Confidential

This document contains the proprietary information of MagTek. Its receipt or possession does not convey any rights to reproduce or disclose its contents or to manufacture, use or sell anything it may describe. Reproduction, disclosure or use without specific written authorization of MagTek is strictly forbidden.

MAGTEK®
REGISTERED TO ISO 9001:2008
1710 Apollo Court
Seal Beach, CA 90740
Phone: (562) 546-6400
FAX: (562) 546-6301
Technical Support: (651) 415-6800
www.magtek.com

REVISIONS

Rev Number	Date	Notes
1.01	2012 Apr 30	Initial Release
1.02	2012 Jun 20	Adds setConfigurationParams
1.03	2012 Sept 24	Added more detailed descriptions to getMaskedTracks, getTrack1-3, getTrack1Masked-3, getMagneprint, getMagnePrintStatus, getDeviceSerial, getSessionID, getKSN, getEncryptionStatus

SOFTWARE LICENSE AGREEMENT

IMPORTANT: YOU SHOULD CAREFULLY READ ALL THE TERMS, CONDITIONS AND RESTRICTIONS OF THIS LICENSE AGREEMENT BEFORE INSTALLING THE SOFTWARE PACKAGE. YOUR INSTALLATION OF THE SOFTWARE PACKAGE PRESUMES YOUR ACCEPTANCE OF THE TERMS, CONDITIONS, AND RESTRICTIONS CONTAINED IN THIS AGREEMENT. IF YOU DO NOT AGREE WITH THESE TERMS, CONDITIONS, AND RESTRICTIONS, PROMPTLY RETURN THE SOFTWARE PACKAGE AND ASSOCIATED DOCUMENTATION TO THE ABOVE ADDRESS, ATTENTION: CUSTOMER SUPPORT.

TERMS, CONDITIONS, AND RESTRICTIONS

MagTek, Incorporated (the "Licensor") owns and has the right to distribute the described software and documentation, collectively referred to as the "Software".

LICENSE: Licensor grants you (the "Licensee") the right to use the Software in conjunction with MagTek products. LICENSEE MAY NOT COPY, MODIFY, OR TRANSFER THE SOFTWARE IN WHOLE OR IN PART EXCEPT AS EXPRESSLY PROVIDED IN THIS AGREEMENT. Licensee may not decompile, disassemble, or in any other manner attempt to reverse engineer the Software. Licensee shall not tamper with, bypass, or alter any security features of the software or attempt to do so.

TRANSFER: Licensee may not transfer the Software or license to the Software to another party without the prior written authorization of the Licensor. If Licensee transfers the Software without authorization, all rights granted under this Agreement are automatically terminated.

COPYRIGHT: The Software is copyrighted. Licensee may not copy the Software except for archival purposes or to load for execution purposes. All other copies of the Software are in violation of this Agreement.

TERM: This Agreement is in effect as long as Licensee continues the use of the Software. The Licensor also reserves the right to terminate this Agreement if Licensee fails to comply with any of the terms, conditions, or restrictions contained herein. Should Licensor terminate this Agreement due to Licensee's failure to comply, Licensee agrees to return the Software to Licensor. Receipt of returned Software by the Licensor shall mark the termination.

LIMITED WARRANTY: Licensor warrants to the Licensee that the disk(s) or other media on which the Software is recorded are free from defects in material or workmanship under normal use.

THE SOFTWARE IS PROVIDED AS IS. LICENSOR MAKES NO OTHER WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Because of the diversity of conditions and PC hardware under which the Software may be used, Licensor does not warrant that the Software will meet Licensee specifications or that the operation of the Software will be uninterrupted or free of errors.

IN NO EVENT WILL LICENSOR BE LIABLE FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE, OR INABILITY TO USE, THE SOFTWARE. Licensee's sole remedy in the event of a defect in material or workmanship is expressly limited to replacement of the Software disk(s) if applicable.

GOVERNING LAW: If any provision of this Agreement is found to be unlawful, void, or unenforceable, that provision shall be removed from consideration under this Agreement and will not affect the enforceability of any of the remaining provisions. This Agreement shall be governed by the laws of the State of California and shall inure to the benefit of MagTek, Incorporated, its successors or assigns.

ACKNOWLEDGMENT: LICENSEE ACKNOWLEDGES THAT HE HAS READ THIS AGREEMENT, UNDERSTANDS ALL OF ITS TERMS, CONDITIONS, AND RESTRICTIONS, AND AGREES TO BE BOUND BY THEM. LICENSEE ALSO AGREES THAT THIS AGREEMENT SUPERSEDES ANY AND ALL VERBAL AND WRITTEN COMMUNICATIONS BETWEEN LICENSOR AND LICENSEE OR THEIR ASSIGNS RELATING TO THE SUBJECT MATTER OF THIS AGREEMENT.

QUESTIONS REGARDING THIS AGREEMENT SHOULD BE ADDRESSED IN WRITING TO MAGTEK, INCORPORATED, ATTENTION: CUSTOMER SUPPORT, AT THE ABOVE ADDRESS, OR E-MAILED TO support@magtek.com.

Table of Contents

SECTION 1. MagTekSCRA CLASS

Classes	Description
MagTekSCRA	This class allows you to perform reader functions.

Methods:

openDevice:	Open device
closeDevice:	Close device
isDeviceConnected	Check the connection status of reader
clearBuffers	Clear card data that is stored through the setCardData function
getMaskedTracks	Retrieves the existing stored masked track data
getTrack1	Retrieves encrypted track1
getTrack2	Retrieves encrypted track2
getTrack3	Retrieves encrypted track3
getTrack1Masked	Retrieves masked track1
getTrack2Masked	Retrieves masked track2
getTrack3Masked	Retrieves masked track3
getMagnePrint	Retrieves encrypted MagnePrint
getMagnePrintStatus	Retrieves encrypted MagnePrintStatus
getDeviceSerial	Retrieves device serial number
getSessionID	Retrieves session ID
getKSN	Retrieves key serial number
getTagValue	Retrieves tag value in tracks data
getMagTekDeviceSerial	Retrieves device serial number created by MagTek
getDeviceName	Retrieves device model name
getDeviceCaps	Retrieves device capabilities
getDeviceStatus	Retrieves device status
getTLVVersion	Retrieves TLV version
getDevicePartNumber	Retrieves device part number
sendCommandToDevice	Send command to device
getCapMSR	Retrieve MSR Capability
getCapTracks	Retrieve Tracks Capability
getCapMagStripeEncryption	Retrieve MagStripe Encryption Capability
setDeviceProtocolString	Sets the protocol String for iDynamo
listenForEvents	Setup the events to listen for
getDeviceType	Retrieves the Device Type
getCardPANLength	Retrieves the Length of teh PAN
getResponseData	Retrieved the whole Response from the reader
getCardName	Retrieves the Name in the Card
getCardIIN	Retrieves the IIN in the Card
getCardLast4	Retrieves the Last 4 of the PAN
getCardExpDate	Retrieves the Expiration Date
getCardServiceCode	Retrieves the Service Code
getCardStatus	Retrieves the Card Status
getTrackDecodeStatus	Retrieves the Track Decode Status
getResponseType	Retrieve Response Type
setDeviceType	Sets the type of device to Open
isDeviceOpened	Retrieves device opened status

Notifications:

trackDataReadyNotification	Receieves notification when track data is available on reader
devConnectionNotification	Receieves notification when reader’s connection status is changed
getSDKVersion	Retrieves SDK version
getOperationStatus	Retrieves operation status

Enum:

MTSCRADeviceType	MAGTEKAUDIOREADER
MTSCRATransactionStatus	TRANS_STATUS_OK TRANS_STATUS_START TRANS_STATUS_ERROR
MTSCRATransactionData	TLV_OPSTS TLV_CARDSTS TLV_TRACKSTS TLV_CARDNAME TLV_CARDIIN TLV_CARDLAST4 TLV_CARDEXPDATE TLV_CARDSVCCODE TLV_CARDPANLEN TLV_ENCTK1 TLV_ENCTK2 TLV_ENCTK3 TLV_DEVSN TLV_DEVSNMAGTEK TLV_DEVFW TLV_DEVNAME TLV_DEVCAPS TLV_TLVVERSION TLV_DEVPARTNUMBER TLV_KSN TLV_CMAC

Methods

openDevice:

This function opens the reader.

```
- (BOOL) openDevice
```

Parameters

Return Value

YES if the device is opened successfully. Otherwise, return NO.

closeDevice:

This function close the reader.

```
- (BOOL) closeDevice
```

Parameters

Return Value

YES if the device is closed successfully. Otherwise, return NO.

isDeviceConnected

This function retrieves the connection status of the reader.

```
- (BOOL) isDeviceConnected
```

Parameters

Return Value

YES if the device is connected. Otherwise, return NO.

clearBuffers

Clears all the buffer that is stored during card swipe or command response.

```
- (void) clearBuffers
```

Parameters

getMaskedTracks

Retrieves existing stored Masked data, only supported for iDynamo, it will return a empty string in audio reader. If decodable track data exists for a given track, it is located in the Masked Track Data field that corresponds to the track number. The length of each Masked Track Data field is fixed at 112 bytes, but the length of valid data in each field is determined by the Masked Track Data Length field that corresponds to the track number. Masked Track Data located in positions greater than indicated in the Masked Track Data Length field are undefined and should be ignored. The HID specification requires that reports be fixed in size but the number of bytes encoded on a card may vary. Therefore, the Input Report always contains the maximum amount of bytes that can be encoded on the card and the number of valid bytes in each track is indicated by the Masked Track Data Length field.

The Masked Track Data is decoded and converted to ASCII and then it is "masked." The Masked Track Data includes all data starting with the start sentinel and ending with the end sentinel. Much of the data is "masked;" a specified mask character is sent instead of the actual character read from the track. Which characters are masked depends on the format of the card. Only ISO/ABA (Financial Cards with Format Code B) and AAMVA cards are selectively masked; all other card types are either entirely masked or sent totally in the clear. There is a separate masking property for ISO/ABA cards and AAMVA cards. See the ISO Track Masking property and the AAMVA Track Masking property for more information. (Refer Appendix EError! Reference source not found. in Reference Manual 99875475 for a description on how ISO/ABA and AAMVA cards are identified.)

Each of these properties allows the application to specify masking details for the Primary Account Number and Driver's License / ID Number (DL/ID#), the masking character to be used, and whether a correction should be applied to make the Mod 10 9 (Luhn algorithm) digit at the end of the number be correct.

```
- (NSString *) getMaskedTracks
```

Parameters

Return Value

```
     Return stored masked tracks data string.
```

getTrack1

Retrieve Encrypted Track1 if any. This field contains the encrypted track data for track 1.

```
- (NSString *) getTrack1
```

Parameters

Return Value

```
     Return stored encrypted track1 data string.
```

getTrack2

Retrieve Encrypted Track2 if any. This field contains the encrypted track data for track 2.

```
- (NSString *) getTrack2
```

Parameters

Return Value

```
     Return stored encrypted track2 data string.
```

getTrack3

Retrieve Encrypted Track3 if any. This field contains the encrypted track data for track 3.

```
- (NSString *) getTrack3
```

Parameters

Return Value

```
     Return stored encrypted track3 data string.
```

getTrack1Masked

Retrieve Masked Track1 if any

```
- (NSString *) getTrack1
```

Parameters

Return Value

```
     Return stored masked track1 data string.
```

For an ISO/ABA card, the PAN is masked as follows:

- The specified number of initial characters is sent unmasked. The specified number of trailing characters is sent unmasked. If Mod 10 correction is specified, all but one of the intermediate characters of the PAN are set to zero; one of them will be set such that last digit of the PAN calculates an accurate Mod 10 check of the rest of the PAN *as transmitted*. If the Mod 10 correction is not specified, all of the intermediate characters of the PAN are set to the specified mask character.
- The Card Holder's name and the Expiration Date are transmitted unmasked.
- All Field Separators are sent unmasked.
- All other characters are set to the specified mask character.

For an AAMVA card, the specified mask character is substituted for each of the characters read from the card.

getTrack2Masked

Retrieve Masked Track2 if any

```
- (NSString *) getTrack2
```

Parameters

Return Value

```
     Return stored masked track2 data string.
```

For an ISO/ABA card, the PAN is masked as follows:

- The specified number of initial characters are sent unmasked. The specified number of trailing characters are sent unmasked. If Mod 10 correction is specified, all but one of the intermediate characters of the PAN are set to zero; one of them will be set such that last digit of the PAN calculates an accurate Mod 10 check of the rest of the PAN as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the PAN are set to the specified mask character.
- The Expiration Date is transmitted unmasked.
- All Field Separators are sent unmasked.
- All other characters are set to the specified mask character.

For an AAMVA card, the DL/ID# is masked as follows:

- The specified number of initial characters are sent unmasked. The specified number of trailing characters are sent unmasked. If Mod 10 correction is specified, all but one of the intermediate characters of the DL/ID#PAN are set to zero; one of them will be set such that last digit of the DL/ID# calculates an accurate Mod 10 check of the rest of the DL/ID# as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the DL/ID# are set to the specified mask character.
- The Expiration Date and Birth Date are transmitted unmasked.
- All other characters are set to the specified mask character.

getTrack3Masked

Retrieve Masked Track3 if any

```
- (NSString *) getTrack3
```

Parameters

Return Value
    Return stored masked track3 data string.

For an ISO/ABA card, the PAN is masked as follows:

- The specified number of initial characters are sent unmasked. The specified number of trailing characters are sent unmasked. If Mod 10 correction is specified, all but one of the intermediate characters of the PAN are set to zero; one of them will be set such that last digit of the PAN calculates an accurate Mod 10 check of the rest of the PAN as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the PAN are set to the specified mask character.
- All Field Separators are sent unmasked.
- All other characters are set to the specified mask character.

For an AAMVA card, the specified mask character is substituted for each of the characters read from the card.

getMagnePrint

Supported on uDynamo only. This 128 byte Binary field contains the MagnePrint data. Only the number of bytes specified in the MagnePrint data length field are valid. The least significant bit of the first byte of data in this field corresponds to the first bit of MagnePrint data. If the Enable/Disable MagnePrint property is set to disable MagnePrint, this field will not be sent.

```
- (NSString *) getMagnePrint
```

Parameters

Return Value
    Empty String.

getMagnePrintStatus

Supported on uDynamo only

```
- (NSString *) getMagnePrintStatus
```

Parameters

Return Value

Empty String.

This Binary field represents 32 bits of MagnePrint status information. Each character represents 4 bits (hexadecimal notation). For example, suppose the characters are: "A1050000":

Nibble	1				2				3				4				5				6				7				8			
Value	A				1				0				5				0				0				0				0			
Bit	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	23	22	21	20	19	18	17	16	31	30	29	28	27	26	25	24
Value	1	0	1	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Usage*	R	R	R	R	R	R	R	M	R	R	R	R	R	R	R	R	0	0	D	0	F	L	N	S	0	0	0	0	0	0	0	0

* Usage Legend:

- D = Direction
- F = Too Fast
- L = Too Slow
- M = MagnePrint capable
- N = Too Noisy
- R = Revision

This four-byte field contains the MagnePrint status. The MagnePrint status is in little endian byte order. Byte 1 is the least significant byte. Byte 1 LSB is status bit 0. Byte 4 MSB is status bit 31. MagnePrint status is defined as follows:

Bit 0	=	This is a MagnePrint-capable product (usage M)
Bits 1-15	=	Product revision & mode (usage R)
Bit 16	=	STATUS-only state (usage S)
Bit 17	=	Noise too high or "move me" away from the noise source (used only in STATUS) (usage N)
Bit 18	=	Swipe too slow (usage L)
Bit 19	=	Swipe too fast (usage F)
Bit 20	=	Unassigned (always set to Zero)
Bit 21	=	Actual Card Swipe Direction (0 = Forward, 1 = Reverse) (usage D)
Bits 22-31	=	Unassigned (always set to Zero)

If the Enable/Disable MagnePrint property is set to disable MagnePrint, this field will not be sent.

getDeviceSerial

Retrieve device serial number. This 16-byte ASCII field contains the device serial number. The device serial number is a NUL (zero) terminated string. So the maximum length of the device serial number, not including the null terminator, is 15 bytes. This device serial number can also be retrieved and set with the device serial number property explained in the property section of this document. This field is stored in non-volatile memory, so it will persist when the unit is power cycled.

```
- (NSString *) getDeviceSerial
```

Parameters

Return Value

```
     Return stored device serial number.
```

getSessionID

Not Supported on Audio Reader. This 8-byte Binary field contains the encrypted version of the current Session ID. Its primary purpose is to prevent replays. After a card is read, this property will be encrypted, along with the card data, and supplied as part of the transaction message. The clear text version of this will never be transmitted. To avoid replay, the application sets the Session ID property before a transaction and verifies that the Encrypted Session ID returned with card data decrypts to the value set.

```
- (NSString *) getSessionID
```

Parameters

Return Value

```
     Empty String
```

getKSN

Retrieve key serial number. This 10-byte Binary field contains the DUKPT Key Serial Number used to encrypt the encrypted fields in this message. This 80-bit field includes the Initial Key Serial Number in the leftmost 59 bits and a value for the Encryption Counter in the rightmost 21 bits. If no keys are loaded, all bytes will have the value 0x00.

```
- (NSString *) getKSN
```

Parameters

Return Value

```
     Return stored key serial number.
```

getTagValue

Retrieve individual tag value, only supported in audio reader

```
- (NSString *) getTagValue: (UInt32)tag
```

```
Parameters
tag
     A MTSCRATransactionData Enum type.

Return Value
     Return tag value.
```

getMagTekDeviceSerial

Retrieve Device Serial Number created by MagTek

```
- (NSString *) getMagTekDeviceSerial
```

```
Parameters

Return Value
     Return stored key serial number created by MagTek.
```

getFirmware

Retrieve firmware version number.

```
- (NSString *) getFirmware
```

```
Parameters

Return Value
     Return firmware version.
```

getDeviceName

Get device model name.

```
- (NSString *) getDeviceName
```

```
Parameters

Return Value
     Return device model name.
```

getDeviceCaps

Get device capabilities. For future use.

```
- (NSString *) getDeviceCaps
```

Parameters

Return Value

```
     Return device capabilities.
```

getDeviceStatus

Get device status. For future use.

```
- (NSString *) getDeviceStatus
```

Parameters

Return Value

```
     Return device status.
```

getTLVVersion

Get TLV Version of firmware.

```
- (NSString *) getTLVVersion
```

Parameters

Return Value

```
     Return TLV version of firmware as a Two-byte hex string.
```

getDevicePartNumber

Not Supported on Audio Reader.

```
- (NSString *) getDevicePartNumber
```

Parameters

Return Value

```
     Empty String.
```

sendCommandToDevice

Send command to device.

```
- (Void *) sendCommandToDevice: (NSString *)pData
```

```
Parameters
     Command string.
Return Value
     Send command to device. Please refer to Section 2 for command
list.
```

getCapMSR

Retrieve MSR Capability. Returned when Discovery Command is sent.

```
- (NSString *) getCapMSR
```

```
Parameters

Return Value
     Return MSR Capability
     0 = No MSR,
     1 = MSR
```

getCapTracks

Retrieve Tracks Capability. Returned when Discovery Command is sent.

```
- (NSString *) getCapTracks
```

```
Parameters

Return Value
     Return Tracks Capability
     Bit 0 = 1 / Track 1 supported,
     Bit 1 = 1 / Track 2 supported,
     Bit 2 = 1 / Track 3 supported,
     all other bits 0.
```

getCapMagStripeEncryption

Retrieve MagStripe Encryption Capability. Returned when Discovery Command is sent.

```
- (NSString *) getCapMagStripeEncryption
```

```
Parameters

Return Value
     Return MagStripe Encryption Capability
     0 = No Encryption,
     1 = TDES DUKPT / PIN Variant,
     other values TBD
```

setDeviceProtocolString

Sets the protocol String for iDynamo

```
- (void) setDeviceProtocolString: (NSString *)pData
```

```
Parameters
     Protocol String
```

listenForEvents

Setup the events to listen for

```
- (void) setDeviceProtocolString: (UInt32 *)event
```

```
Parameters
     Event
```

getDeviceType

Retrieves the Device Type. For future use.

```
- (int) getDeviceType
```

```
Parameters

Return Value
     Device Type
```

getCardPANLength

Retrieves the Length of teh PAN

```
- (int) getCardPANLength
```

```
Parameters

Return Value
     Card PAN Length
```

getResponseData

Retrieves the whole response from the reader

```
- (NSString *) getResponseData
```

```
Parameters

Return Value
     Response Data
```

getCardName

Retrieves the Name in the Card

```
- (NSString *) getCardName
```

Parameters

Return Value

```
      Card Name
```

getCardIIN

Retrieves the IIN in the Card

```
- (NSString *) getCardIIN
```

Parameters

Return Value

```
      IIN
```

getCardLast4

Retrieves the Last 4 of the PAN

```
- (NSString *) getCardLast4
```

Parameters

Return Value

```
      Last 4 of the PAN
```

getCardExpDate

Retrieves the Expiration Date

```
- (NSString *) getCardExpDate
```

Parameters

Return Value

```
      Expiration Date
```

getCardServiceCode

Retrieves the Service Code

```
- (NSString *) getCardServiceCode
```

Parameters

Return Value

```
      Service Code
```

getCardStatus

Retrieves the Card Status

```
- (NSString *) getCardStatus
```

Parameters

Return Value

Card Status

getTrackDecodeStatus

Retrieves the Track Decode Status. This is a one-byte value, which indicates the status of decoding track 1. Bit position zero indicates if there was an error decoding track 1 if the bit is set to one. If it is zero, then no error occurred. If a track has data on it that is not noise, and it is not decodable, then a decode error is indicated. If a decode error is indicated, the corresponding track data length value for the track that has the error will be set to zero and no valid track data will be supplied.

```
- (NSString *) getTrackDecodeStatus
```

Parameters

Return Value

Track Decode Status. Consists of three 2-byte hex values representing the decode status for tracks 1, 2, and 3 (respectively from left to right).

Values are:
00 = Track OK
01 = Track read Error
02 = Track is Blank

getResponseType

Retrieves the Response Type.

```
- (NSString *) getResponseType
```

Parameters

Return Value

For Audio Reader, always “C101”.

setDeviceType

Sets the type of device to Open

```
- (void) setDeviceType: (UInt32 *)deviceType
```

Parameters

Device Type. Options:
MTSCRADeviceType.MAGTEKAUDIOREADER
MTSCRADeviceType. MAGTEKIDYNAMO

```
     MTSCRADeviceType. MAGTEKNONE
```

isDeviceOpened

Retrieves device opened status

```
- (BOOL) isDeviceOpened
```

Parameters

Return Value

```
     Boolean whether device is opened or not
```

getBatteryLevel

Retrieves device battery level

```
- (long) getBatteryLevel
```

Parameters

Return Value

```
     Battery Level (0 to 100)
```

getSDKVersion

Retrieves the SDK Version

```
- (NSString *) getSDKVersion
```

Parameters

Return Value

```
     SDK Version
```

getOperationStatus

Retrieves the operation status.

```
 - (NSString *) getOperationStatus
```

Parameters

Return Value

```
     2-byte bit string in hex. The bits correspond to:
     Bit 0 = DUKPT Keys exhausted (1=exhausted, 0=keys available)
     Bit 1 = Initial DUKPT key Injected, always set to One (Primary
     DUKPT Key)
     Bit 2 = Encryption Enabled
     Bit 3 = Reserved (always set to zero)
     Bit 4 = Reserved (always set to zero)
     Bit 5 = Reserved (always set to zero)
     Bit 6 = Reserved (always set to zero)
     Bit 7 = Reserved (always set to zero)
     Bit 8 = Reserved (always set to zero)
     Bit 9 = Initial DUKPT key injected (Secondary DUKPT Key)
     Bit 10 = DUKPT Key used for encryption,
          0=Primary,
          1=Secondary
     Bit 11 = DUKPT Key Variant used to encrypt data,
          0=PIN Variant,
          1=Data Variant/Bidirectional
     Bits 12-15 = Unassigned (always set to Zero)
```

setConfigurationParams

Sets configuration parameters

```
- (void) setConfigurationParams:(NSString *)pData
```

Parameters

```
     pData
```

PAN_MOD10_CHECKDIGIT = TRUE/FALSE (default is TRUE)

Notification

trackDataReadyNotification

Notification received when tracks data is available on reader.

devConnectionNotification

Notification received when the connection status of the reader is changed.

Enum

MTSCRADeviceType

MAGTEKAUDIOREADER is used to open/close audio reader.

MTSCRATransactionStatus

TRANS_STATUS_OK is triggered when transaction succeed.

TRANS_STATUS_START is triggered when reader starts sending data.

TRANS_STATUS_ERROR is triggered when reader fails sending data.

MTSCRATransactionData

TLV_OPSTS
Operation Status

TLV_CARDSTS
Card Information

TLV_TRACKSTS
Card tracks status

TLV_CARDNAME
Card holder name

TLV_CARDIIN
Card issuer identification number

TLV_CARDLAST4
Last four digits of PAN number

TLV_CARDEXPDATE
Card Expiration date

TLV_CARDSVCCODE
Card service code

TLV_CARDPANLEN
The length of PAN number

TLV_ENCTK1
Encrypted track 1

TLV_ENCTK2

Encrypted track 2

TLV_ENCTK3
Encrypted track 3

TLV_DEVSN
Device serial number

TLV_DEVSNMAGTEK
Device serial number created by Magek

TLV_DEVFW
Device firmware version

TLV_DEVNAME
Device model name

TLV_DEVCAPS
Device capabilities

TLV_DEVSTATUS
Device status

TLV_TLVVERSION
Firmware TLV version

TLV_DEVPARTNUMBER
Device part number

TLV_KSN
KSN

TLV_CMAC
CMAC

SECTION 2. COMMANDS

Discovery

Send discovery command to device.

```
- (Void *) sendCommandToDevice: (NSString *)command
```

Parameters

Command string: use “C10206C20503840900” as command string.

Return Value

Following device information can be retrieved.

Device SN, internal

Device serial number created by chip manufacturer.

use getDeviceSerial method to retrieve data.

Device SN, MagTek

Device serial number created by MagTek.

use getDeviceSerialMagTek method to retrieve data.

Device Firmware Part Number

Device firmware part number.

use getFirmware method to retrieve data.

Device Model Name

Device model name.

use getDeviceName method to retrieve data.

Device TLV Version

Device TLV version.

use getTLVVersion method to retrieve data.

Device Part Number

Device part number.

use getDevicePartNumber method to retrieve data.

Capability - MSR

0 = No MSR, 1 = MSR

use getCapMSR method to retrieve data.

Capability - TRACKS

0 = Supported tracks: None.
1 = Supported tracks: Track1.
2 = Supported tracks: Track2.
3 = Supported tracks: Track1, Track2.

4 = Supported tracks: Track3.
5 = Supported tracks: Track1, Track3.
6 = Supported tracks: Track2, Track3.
7 = Supported tracks: Track1, Track2, Track3.

use getCapTracks method to retrieve data.

Capability – MagStripe Encryption

0 = No Encryption, 1 = TripDES DUKPT

use getCapMagStripeEncryption method to retrieve data.

SECTION 3. CODE EXAMPLES

Open Device:

```
    self.mtSCRALib = [[MTSCRA alloc] init];
    [self.mtSCRALib
listenForEvents:(TRANS_EVENT_OK|TRANS_EVENT_START|TRANS_EVENT_ERROR)];

   //iDynamo
    [self.mtSCRALib setDeviceType:(MAGTEKIDYNAMO)];
    [self.mtSCRALib setDeviceProtocolString:(“com.magtek.idynamo”)];
    [self.mtSCRALib setDeviceType:(MAGTEKIDYNAMO)];
   //Audio
    //[self.mtSCRALib setDeviceType:(MAGTEKAUDIOREADER)];
    [self.mtSCRALib openDevice];
```

Close Device:

```
    [self.mtSCRALib closeDevice];
```

Get Tracks Data From Reader:

```
  [[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(trackDataReady:) name:@"trackDataReadyNotification"
object:nil];


- (void)trackDataReady:(NSNotification *)notification
{
    NSNumber *status = [[notification userInfo]
valueForKey:@"status"];

    [self performSelectorOnMainThread:@selector(onDataEvent:)
withObject:status waitUntilDone:YES];
}

- (void)onDataEvent:(id)status
{
      //[self clearLabels];

      switch ([status intValue]) {
        case TRANS_STATUS_OK:

            NSLog(@"TRANS_STATUS_OK");
            break;

        case TRANS_STATUS_ERROR:
            NSLog(@"TRANS_STATUS_ERROR");

            break;

        default:
            break;
    }


}
```

Get Connection Status Of Reader:

```
[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(devConnStatusChange)
name:@"devConnectionNotification" object:nil];


- (void)devConnStatusChange
{
    BOOL isDeviceConnected = [self.mtSCRALib isDeviceConnected];

    if (isDeviceConnected)
    {
        self.deviceStatus.text = @"Device Connected";
    }
    else
    {
        self.deviceStatus.text = @"Device Disconnected";
    }
}
```