



C4 API Description

iDTRONIC GmbH
Donnersbergweg 1
67059 Ludwigshafen
Germany/Deutschland

Phone: +49 621 6690094-0
Fax: +49 621 6690094-9
E-Mail: info@idtronic.de
Web: idtronic.de

Issue 1.2
– 20. February 2018 –

Subject to alteration without prior notice.
© Copyright iDTRONIC GmbH 2017
Printed in Germany

Contents

1	1D/2D Code Reader.....	5
1.1	SetCodedFormat().....	5
1.2	Open().....	5
1.3	Scan().....	5
1.4	Close().....	5
2	HF-ISO14443A	6
2.1	SearchCard14443A().....	6
2.2	Auth14443A ().....	6
2.3	Read14443A ().....	6
2.4	Write14443A ().....	6
3	HF-ISO15693.....	7
3.1	FindCard15693().....	7
3.2	GetInformation15693().....	7
3.3	ReadSingleBlock15693 ().....	7
3.4	WriteSingleBlock ().....	7
4	UHF-ISO1800-6C.....	8
4.1	GetInstance().....	8
4.2	GetFirmware().....	8
4.3	SetOutputPower().....	8
4.4	InventoryRealTime().....	8
4.5	SelectEPC().....	8
4.6	ReadFrom6C().....	8
4.7	WriteTo6C().....	8
4.8	SetWorkArea().....	9
4.9	GetFrequency().....	9
4.10	SetFrequency().....	9
5	Fingerprint	10
5.1	Open().....	10
5.2	DetectFinger().....	10
5.3	GetFingerprintImage().....	10
5.4	GetChara().....	10
5.5	RegisterFingerprint ().....	10
5.6	SearchFingerprint().....	11
5.7	MatchFingerprint().....	11
5.8	GetFreeAddress().....	11
5.9	EmptyAllChara().....	11
5.10	EmptyChara().....	11
6	UHFLonger	12
6.1	GetInstance().....	12
6.2	SetOutputPower().....	12
6.3	InventoryRealTime().....	12
6.4	SelectEPC().....	12
6.5	ClearSelect().....	12

6.6	ReadData()	12
6.7	WriteData()	12
6.8	WriteEPC()	13
6.9	WriteAccess()	13
6.9.1	WriteKillPsd().....	13
6.10	LockTag()	13
7	LF125KManager	14
7.1	Open()	14
7.2	Start()	14
7.3	Pause().....	14
7.4	Close().....	14
8	Appendix	15
8.1	Key Codes	15
8.2	SerialPort:.....	15

1 1D/2D Code Reader

1.1 SetCodedFormat()

Function	void SetCodedFormat(String format)
Description	Set encoding format
Parameter	Format
Return	void

Note: Format of "UTF-8" by default.

1.2 Open()

Function	boolean Open(Handler handler)
Description	Load the 1D/2D Scanner and Start Receiving Thread
Parameter	Handler
Return	True: success False:failure

Note: Int the Handler: msg.what == Barcode1DManager.Barcode1D
String data = msg.getData().getString("data");

1.3 Scan()

Function	void Scan()
Description	Start the 1D/2D Scanner to Scan Barcode
Parameter	void
Return	void

1.4 Close()

Function	boolean Close()
Description	Unload the 1D/2D Scanner and Interrupt Receiving Thread
Parameter	void
Return	True:success False:failure

Note: Close the hardware module when the process is finished.

2 HF-ISO14443A

2.1 SearchCard14443A()

Function	byte[] SearchCard14443A(Error error)
Description	Used to search iso14443A card
Parameter	Error
Return	Byte[]: ISO14443A card uid Null: no card

Note: HfError error = new HfError();

The following error description is the same.

2.2 Auth14443A ()

Function	int auth14443A(int authType, byte[] access, byte[] uid, int sector, Error error)
Description	auth iso14443A card
Parameter	authType , password , uid , sector number , error
Return	0: auth success Other: error code

2.3 Read14443A ()

Function	byte[] read14443A(int block, Error error)
Description	Read ISO14443A card data
Parameter	Block number, error
Return	Byte[]: block data Null: read failure

Note: Block number ranges from block 0 to block 63;.

2.4 Write14443A ()

Function	int write14443A(byte[] writeData, int block, Error error)
Description	Write iso14443A card data
Parameter	Byte[] data, block number, error
Return	0: write success Other: write failure, error code

Note: Length of the data should be 16 bytes.

3 HF-ISO15693

3.1 FindCard15693()

Function	List<Iso15693InventoryInfo> findCard15693(Error error)
Description	Used to find ISO15693 card
Parameter	Error
Return	Iso15693InventoryInfo list, Iso15693InventoryInfo contain flag , dsfid and uids

3.2 GetInformation15693()

Function	Iso15693CardInformation getInformation15693(byte[] uid, int flag ,Error error)
Description	To get ISO15693 card system information
Parameter	Card uid , flag , Error
Return	Iso15693CardInformation Null: get failure

3.3 ReadSingleBlock15693 ()

Function	byte[] readSingleBlock15693(byte[] uid, int flag,int block, Error error)
Description	To read ISO15693 card single block data
Parameter	Card uid , flag , block number , Error
Return	Byte[] block data Null: read failure

3.4 WriteSingleBlock ()

Function	int writeSingleBlock(byte[] uid,int flag, int block, byte[] writeData, Error error)
Description	To write ISO15693 card single block data
Parameter	Card uid , flag , block number , data , Error
Return	0: write success Other: write failure

4 UHF-ISO1800-6C

4.1 GetInstance()

Function	UhfManager getInstance()
Description	Get UHF manager instance and open device
Parameter	void
Return	Uhf manager instance

4.2 GetFirmware()

Function	byte[] getFirmware()
Description	Get firmvare version
Parameter	void
Return	Version information

4.3 SetOutputPower()

Function	boolean setOutputPower(int value)
Description	Set antenna power gain
Parameter	Power value
Return	True: set success False: set failure

Note: Scope of value.

4.4 InventoryRealTime()

Function	List<byte[]> inventoryRealTime()
Description	Real-time get UHF EPC list
Parameter	void
Return	EPC list

4.5 SelectEPC()

Function	void selectEPC(byte[] epc)
Description	Select EPC
Parameter	EPC
Return	void

4.6 ReadFrom6C()

Function	byte[] readFrom6C(int memBank, int startAddr, int length, byte[] accessPassword)
Description	Read tag data
Parameter	Membank , start address , read length , password
Return	Tag data

4.7 WriteTo6C()

Function	boolean writeTo6C(byte[] password, int memBank, int startAddr, int dataLen, byte[] data)
----------	--

Description	Write data into ISO-1800-6C tag
Parameter	password , Membank , start address , data length , data
Return	True: write success False: write failure

4.8 SetWorkArea()

Function	int setWorkArea(int area)
Description	Set device work area
Parameter	Work area
Return	0: set success Other: set failure

Note: American standard by default. Generally does not need to be modified.

4.9 GetFrequency()

Function	int getFrequency()
Description	Get module work frequency
Parameter	void
Return	Frequency

4.10 SetFrequency()

Function	int setFrequency(int startFrequency, int freqSpace, int freqQuality)
Description	Set frequency
Parameter	Start Frequency, frequency Space, frequency Quality
Return	0: set success Other: set failure

Note: American standard by default. Generally do not need to be modified.

5 Fingerprint

5.1 Open()

Function	boolean Open()
Description	Open fingerprint device power
Parameter	boolean isRoot, Context context
Return	True: open success False: open failure

Note: If you have already get root permissions ,isRoot = true, else isRoot = false.

5.2 DetectFinger()

Function	boolean DetectFinger()
Description	Detect whether the finger on the sensor
Parameter	void
Return	True: finger exist False: finger inexistence

5.3 GetFingerprintImage()

Function	Bitmap GetFingerprintImage()
Description	Get fingerprint image bitmap
Parameter	void
Return	Bitmap: get fingerprint success Null: get fingerprint image failure

Note: Place the finger on the sensor properly during the process.

5.4 GetChara()

Function	byte[] GetChara()
Description	Get fingerprint characteristic value
Parameter	void
Return	Byte[]: Characteristic value Null: get fingerprint characteristic value failure

Note: Place the finger on the sensor properly during the process.

5.5 RegisterFingerprint ()

Function	int RegisterFingerprint(byte[] chara_1,byte[] chara_1_more,int min_score)
Description	Save fingerprint characteristic value in flash
Parameter	characteristic value_1, characteristic value_2, minimum score
Return	Int >=0: address ID of fingerprint characteriistic value in (register success) Int<0: error code (register failure)

Note: characteristic value_1 and characteristic value_2 must be from the same finger. Min_score should be greater than 50;

5.6 SearchFingerprint()

Function	int SearchFingerprint(byte[] chara)
Description	Search the fingerprint characteristic value from flash memory
Parameter	Finger print characteristic value
Return	Int >=0: address ID of the fingerprint characteristic value in the flash(search success) Int<0: error code (search failure)

Note: Parameters should be obtained dynamically from GetChara()

5.7 MatchFingerprint()

Function	int MatchFingerprint(byte[] chara_1,byte[] chara_2)
Description	Compare fingerprint characteristic value_1 and fingerprint characteristic value_2 to judge whether are they the same fingerprint ?
Parameter	characteristic value 1, characteristic value 2
Return	Match score

Note: Score value larger than 50 indicates the same finger

5.8 GetFreeAddress()

Function	int GetFreeAddress()
Description	Get free address
Parameter	void
Return	Free address ID

5.9 EmptyAllChara()

Function	boolean EmptyAllChara()
Description	Delete all fingerprint characteristic values than have registered in the flash
Parameter	void
Return	True : delete success False: delete failure

5.10 EmptyChara()

Function	boolean EmptyChara(int addressID)
Description	Delete the fingerprint characteristic value by address than has registered in the flash
Parameter	Address
Return	True : delete success False: delete failure

6 UHFLonger

6.1 GetInstance()

Function	UHFLongerManager getInstance()
Description	Get Uhf manager instance and open device
Parameter	void
Return	Uhf manager instance

6.2 SetOutputPower()

Function	boolean setOutputPower(int value)
Description	Set antenna power gain
Parameter	Power value
Return	True: set success False: set failure

Note: Scope of value 500~3000.

6.3 InventoryRealTime()

Function	List<byte[]> inventoryRealTime()
Description	Real-time get UHF EPC list
Parameter	void
Return	EPC list

6.4 SelectEPC()

Function	void selectEPC(byte[] epc)
Description	Select EPC
Parameter	Epc
Return	void

6.5 ClearSelect()

Function	boolean clearSelect()
Description	Clear select
Parameter	void
Return	false

6.6 ReadData()

Function	byte[] readData(int memBank, int start, int length, byte[] password)
Description	Read tag data
Parameter	Membank , start address , read length , password
Return	Tag data

6.7 WriteData()

Function	boolean writeData(int memBank, int start, byte[] password, byte[] wData)
----------	--

Description	Write data into ISO18000-6C tag
Parameter	Membank , start address , password , data
Return	True: write success False: write failure

6.8 WriteEPC()

Function	boolean writeEPC(byte[] newEPC, byte[] password)
Description	Write EPC
Parameter	New epc data, password
Return	True: write success False: write failure

6.9 WriteAccess()

Function	boolean writeAccess(byte[] newAccess, byte[] oldAccess)
Description	Rewrite access
Parameter	New access, old access.
Return	True: write success False: write failure

6.9.1 WriteKillPsd()

Function	boolean writeKillPsd(byte[] newKill, byte[] access)
Description	Rewrite kill password
Parameter	New kill, access
Return	True: write success False: write failure

6.10 LockTag()

Function	boolean lockTag(int lockType, int lockMem, byte[] access)
Description	Lock tag
Parameter	Lock type, Lock membank access
Return	True: write success False: write failure

7 LF125KManager

7.1 Open()

Function	boolean Open(Handler mHandler)
Description	Open device
Parameter	Handler
Return	void

Note: Int the Handler: msg.what == LfHdxManager.LF) {
Bundle bundle = msg.getData();
String data = bundle.getString("data");

7.2 Start()

Function	void start()
Description	Start read card or continue read card after pause
Parameter	void
Return	void

7.3 Pause()

Function	void Pause()
Description	Pause read card after start
Parameter	void
Return	void

7.4 Close()

Function	boolean Close()
Description	Close device
Parameter	void
Return	void

8 Appendix

8.1 Key Codes

1. F1: 131
2. F2: 132
3. Left button: 133
4. Right button: 135

8.2 SerialPort:

Open serial port

Function	SerialPort(int port, int baudrate, int flags)
Description	Open serial port
Parameter	Port number, Port baud rate Flags =0
Return	Serial port class instance

Note: Send and receive the data after opening serial port.

Close serial port:

Function	close(int port)
Description	Close serial port
Parameter	Port number
Return	void

Note: Power management of serial port is packaged in the SerialPort.class, please reference to SerialPortActivity.class for more details.