SAGEM FPS USB FUNCTIONALITIES

Prepared by
Evolute Systems Pvt. Ltd.
Srirama, 15/6, 3rd Floor, Cambridge Road,
Ulsoor, Bangalore - 560 008
Ph: +9180-42074082/42078198

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Data Structure Documentation

T_BIODB_DATA_TRANSPORT Struct Reference

Bio data structure.

`#include <libMSO_Struct.h>`

Data Fields

- PUC **m_puc_Data**
  Data buffer.
- US **m_us_SizeData**
  Size of Buffer.
- UC **m_uc_IdData**
  Data ID: ID_USER_ID or ID_USER_DATA.
- UC **m_uc_RFU**
  Reserved.

Detailed Description

Bio data structure.

Field Documentation

PUC T_BIODB_DATA_TRANSPORT::m_puc_Data

Data buffer.

US T_BIODB_DATA_TRANSPORT::m_us_SizeData

Size of Buffer.

UC T_BIODB_DATA_TRANSPORT::m_uc_IdData

Data ID: ID_USER_ID or ID_USER_DATA.

UC T_BIODB_DATA_TRANSPORT::m_uc_RFU

Reserved.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_BUF Struct Reference

Data Buffer structure.

`#include <libMSO_Struct.h>`

Data Fields

- UL **m_ul_Size**
  Buffer size.
- PUC **m_puc_Buf**
  point to allocated buffer of size m_ul_Size

Detailed Description

Data Buffer structure.

Field Documentation

UL T_BUF::m_ul_Size

Buffer size.

PUC T_BUF::m_puc_Buf

point to allocated buffer of size m_ul_Size

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_BUF_PK Struct Reference

Public list field structure.

`#include <libMSO_Struct.h>`

Data Fields

- **T_BUF m_x_Buf**
  Pk Buffer.
- UC **m_uc_IdPk**
  PK type (ID_PKCOM Recommanded).
- UC **m_uc_Index**
- UC **m_uc_Pad** [2]

Detailed Description

Public list field structure.

See also:
MSO_Bio_Verify

Field Documentation

T_BUF T_BUF_PK::m_x_Buf

Pk Buffer.

UC T_BUF_PK::m_uc_IdPk

PK type (ID_PKCOM Recommanded).

UC T_BUF_PK::m_uc_Index

UC T_BUF_PK::m_uc_Pad[2]

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_EXPORT_IMAGE Struct Reference

Export Image Structure.

`#include <libMSO_Struct.h>`

Data Fields

- **PT_BUF io_px_BufImage**
  Image Buffer.
- UC **i_uc_ExportImageType**
  Default: 0.
- UC **i_uc_CompressionType**
  Compression Type (ID_COMPRESSION_).
- UC **i_uc_CompressionParam**
- UC **pad** [1]

Detailed Description

Export Image Structure.

See also:
MSO_Bio_Enroll

Field Documentation

PT_BUF T_EXPORT_IMAGE::io_px_BufImage

Image Buffer.

UC T_EXPORT_IMAGE::i_uc_ExportImageType

Default: 0.

UC T_EXPORT_IMAGE::i_uc_CompressionType

Compression Type (ID_COMPRESSION_).

UC T_EXPORT_IMAGE::i_uc_CompressionParam

UC T_EXPORT_IMAGE::pad[1]

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_EXPORT_PK Struct Reference

Export Pk Structure.

`#include <libMSO_Struct.h>`

Data Fields

- **PT_BUF io_px_BiometricData**
  Pk Buffer.
- UC **i_uc_ExportMinutiae**
  Export type: Set to 1 to export minutiae with default size.
- UC **pad** [3]

Detailed Description

Export Pk Structure.

See also:
MSO_Bio_Enroll

Field Documentation

PT_BUF T_EXPORT_PK::io_px_BiometricData

Pk Buffer.

UC T_EXPORT_PK::i_uc_ExportMinutiae

Export type: Set to 1 to export minutiae with default size.

UC T_EXPORT_PK::pad[3]

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_FIELD Struct Reference

Field structure.

`#include <libMSO_Struct.h>`

Data Fields

- US **m_us_FieldMaxSize**
  Field Max size.
- UC **m_auc_FieldName** [MAX_FIELD_NAME_LEN+1]
  Field Name.
- UC **m_uc_Right**
  Field type: PUBLIC or PRIVATE.

Detailed Description

Field structure.

Field Documentation

US T_FIELD::m_us_FieldMaxSize

Field Max size.

UC T_FIELD::m_auc_FieldName[MAX_FIELD_NAME_LEN+1]

Field Name.

UC T_FIELD::m_uc_Right

Field type: PUBLIC or PRIVATE.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_ILV_ADD_FIELD Struct Reference

Additionnal field structure.

`#include <libMSO_Struct.h>`

Data Fields

- US **m_us_FieldMaxSize**
  Field Max size.
- UC **m_auc_FieldName** [MAX_FIELD_NAME_LEN]
  field name

Detailed Description

Additionnal field structure.

Use in MSO_BioDB_GetBaseConfig.c

Field Documentation

US T_ILV_ADD_FIELD::m_us_FieldMaxSize

Field Max size.

UC T_ILV_ADD_FIELD::m_auc_FieldName[MAX_FIELD_NAME_LEN]

field name

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_ILV_BASE_CONFIG Struct Reference

Base configuration structure.

`#include <libMSO_Struct.h>`

Data Fields

- UC **m_uc_RequestStatus**
  Request status.
- UC **m_uc_Finger**
  Number of finger per user.
- UL **m_ul_MaxRecord**
  Max number of database record.
- UL **m_ul_CurrentRecord**
  Current number of databse record.
- UL **m_ul_FreeRecord**
  Number of free database record.
- UL **m_ul_AddField**
  Number of additionnal field.

Detailed Description

Base configuration structure.

Use in MSO_BioDB_GetBaseConfig.c

Field Documentation

UC T_ILV_BASE_CONFIG::m_uc_RequestStatus

Request status.

UC T_ILV_BASE_CONFIG::m_uc_Finger

Number of finger per user.

UL T_ILV_BASE_CONFIG::m_ul_MaxRecord

Max number of database record.

UL T_ILV_BASE_CONFIG::m_ul_CurrentRecord

Current number of databse record.

UL T_ILV_BASE_CONFIG::m_ul_FreeRecord

Number of free database record.

UL T_ILV_BASE_CONFIG::m_ul_AddField

Number of additionnal field.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_ILV_DB Struct Reference

ILV database structure.

`#include <libMSO_Struct.h>`

Data Fields

- UC **m_uc_IndexDB**
  Database Index.
- UC **m_uc_FlashType**
  Not used: set to 0.
- US **m_us_UserMax**
  Max number of user.
- UC **m_uc_PkMax**
  Max number of PK.

Detailed Description

ILV database structure.

Use in MSO_BioDB_CreateDb.c

Field Documentation

UC T_ILV_DB::m_uc_IndexDB

Database Index.

UC T_ILV_DB::m_uc_FlashType

Not used: set to 0.

US T_ILV_DB::m_us_UserMax

Max number of user.

UC T_ILV_DB::m_uc_PkMax

Max number of PK.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_MORPHO_CALLBACK_ENROLLMENT_STATUS Struct Reference

Enrollment command status.

`#include <libMSO_Struct.h>`

Data Fields

- UC **m_uc_nbFinger**
  Current number of finger acquisition.
- UC **m_uc_nbFingerTotal**
  Total number of finger needed for one finger.
- UC **m_uc_nbCapture**
  Current number of finger captured.
- UC **m_uc_nbCaptureTotal**
  Total number of finger to capture.

Detailed Description

Enrollment command status.

Status returned when MORPHO_CALLBACK_ENROLLMENT_CMD event occurs

---

Field Documentation

UC T_MORPHO_CALLBACK_ENROLLMENT_STATUS::m_uc_nbFinger

Current number of finger acquisition.

UC T_MORPHO_CALLBACK_ENROLLMENT_STATUS::m_uc_nbFingerTotal

Total number of finger needed for one finger.

UC T_MORPHO_CALLBACK_ENROLLMENT_STATUS::m_uc_nbCapture

Current number of finger captured.

UC T_MORPHO_CALLBACK_ENROLLMENT_STATUS::m_uc_nbCaptureTotal

Total number of finger to capture.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_MORPHO_CALLBACK_IMAGE_STATUS Struct Reference

Image status.

`#include <libMSO_Struct.h>`

Data Fields

- **T_MORPHO_IMAGE_HEADER m_x_ImageHeader**
  Image header.
- PUC **m_puc_Image**
  Image buffer.

Detailed Description

Image status.

Image returned when MORPHO_CALLBACK_IMAGE_CMD event occurs

Field Documentation

T_MORPHO_IMAGE_HEADER T_MORPHO_CALLBACK_IMAGE_STATUS::m_x_ImageHeader

Image header.

PUC T_MORPHO_CALLBACK_IMAGE_STATUS::m_puc_Image

Image buffer.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_MORPHO_IMAGE_HEADER Struct Reference

Image header structure.

`#include <libMSO_Struct.h>`

Data Fields

- UC **m_uc_HeaderRevision**
  Header Revision number.
- UC **m_uc_HeaderSize**
  Header size.
- US **m_us_NbRow**
  Row number.
- US **m_us_NbCol**
  Column number.
- US **m_us_ResY**
  Y axis resolution.
- US **m_us_ResX**
  X axis resolution.
- UC **m_uc_CompressionType**
  Compression type.
- UC **m_uc_NbBitsPerPixel**
  Bits per pixel (8).

Detailed Description

Image header structure.

Field Documentation

UC T_MORPHO_IMAGE_HEADER::m_uc_HeaderRevision

Header Revision number.

UC T_MORPHO_IMAGE_HEADER::m_uc_HeaderSize

Header size.

US T_MORPHO_IMAGE_HEADER::m_us_NbRow

Row number.

US T_MORPHO_IMAGE_HEADER::m_us_NbCol

Column number.

US T_MORPHO_IMAGE_HEADER::m_us_ResY

Y axis resolution.

US T_MORPHO_IMAGE_HEADER::m_us_ResX

X axis resolution.

UC T_MORPHO_IMAGE_HEADER::m_uc_CompressionType

Compression type.

UC T_MORPHO_IMAGE_HEADER::m_uc_NbBitsPerPixel

Bits per pixel (8).

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_MSO_USB_DEVICE_PROPERTIES Struct Reference

Device properties structure.

`#include <libMSO_Struct.h>`

Data Fields

- struct usb_device * **m_px_device**
  usb_device structure
- PUC **m_puc_SerialNumber**
  Device Serial number.
- PUC **m_puc_FriendlyName**
  Device Name (MSO300, CBM ...).
- UC **m_uc_Index**
  Index Number in device list.
- PUC **m_puc_DevicePath**
  Bus and device number: /proc/bus/usb/BBB/DDD.

Detailed Description

Device properties structure.

For each enumerate device, a **T_MSO_USB_DEVICE_PROPERTIES** structure is filled

---

Field Documentation

struct usb_device* T_MSO_USB_DEVICE_PROPERTIES::m_px_device `[read]`

usb_device structure

PUC T_MSO_USB_DEVICE_PROPERTIES::m_puc_SerialNumber

Device Serial number.

PUC T_MSO_USB_DEVICE_PROPERTIES::m_puc_FriendlyName

Device Name (MSO300, CBM ...).

UC T_MSO_USB_DEVICE_PROPERTIES::m_uc_Index

Index Number in device list.

PUC T_MSO_USB_DEVICE_PROPERTIES::m_puc_DevicePath

Bus and device number: /proc/bus/usb/BBB/DDD.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

T_TRANSPORT_PUBLIC_LIST_FIELD Struct Reference

Public list field structure.

`#include <libMSO_Struct.h>`

Data Fields

- UL **m_ul_UserIndex**
  User index in database.
- UL **m_ul_DataLenght**
  Data length.
- PUC **m_puc_Data**
  Buffer Data.

Detailed Description

Public list field structure.

Field Documentation

UL T_TRANSPORT_PUBLIC_LIST_FIELD::m_ul_UserIndex

User index in database.

UL T_TRANSPORT_PUBLIC_LIST_FIELD::m_ul_DataLenght

Data length.

PUC T_TRANSPORT_PUBLIC_LIST_FIELD::m_puc_Data

Buffer Data.

The documentation for this struct was generated from the following file:

- **libMSO_Struct.h**

File Documentation

libMSO.h File Reference

```
#include "libMSO_Def.h"
#include "libMSO_Struct.h"
```

Defines

- #define **MSO_SERIAL_NUMBER_LEN** 24
  Serial number max length.
- #define **COM_USB** "USB"
  USB Connection: See MSO_InitCom.
- #define **COM_RS232** "RS232"
  RS232 Connection: See MSO_InitCom.
- #define **DEFAULT_COM_INTERFACE** COM_USB
  Default Com interface: USB.
- #define **DEFAULT_BAUD_RATE** 115200
  RS232 default baudrate: 115200.
- #define **ID_FORMAT_TEXT** 47
  Text format for Get_Descriptor function.
- #define **ID_FORMAT_BIN** 48
  Binary format for Get_Descriptor function.
- #define **SECU_TUNNELING** 0x01
  < Secure protocols are not supported on Linux
- #define **SECU_OFFERED_SECURITY** 0x02
  If flag is set, the MorphoSmart uses the offered security protocol.
- #define **SECU_PK_ONLY_SIGNED** 0x04
  The MorphoSmart uses only templates with an X9.84 envelop and a signature.
- #define **SECU_NEVER_EXPORT_SCORE** 0x10
  If flag is set, the MorphoSmart never export its matching score.

Typedefs

- typedef I(* **T_pFuncILV_Buffer**)(PVOID, I, PVOID)
  Callback fonction: MSO_RegisterAsyncILV function.

Functions

- I **MSO_InitCom** (**MORPHO_HANDLE** *i_ph_Mso100Handle, PC i_str_Interface, PC i_str_comName, I i_i_baudRate)
  Initialize the connnection with the MorphoSmart.
- I **MSO_CloseCom** (**MORPHO_HANDLE** *io_ph_Mso100Handle)
  Close the communication.
- I **MSO_Usb_EnumDevices** (**PT_MSO_USB_DEVICE_PROPERTIES** *o_ppx_DeviceProperties, PUL o_pul_NbreDevices)
  Enumerate devices connected on USB bus.

- I **MSO_Usb_ReleaseEnumDevices** (**PT_MSO_USB_DEVICE_PROPERTIES** *o_ppx_DeviceProperties, UL i_ul_NbreDevices)
  Release Device properties structure.
- I **MSO_GetDescriptor** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_DescFormat, PUC o_puc_ILV_Status, VOID *o_pv_DescProduct, UL i_ul_SizeOfDescProduct, VOID *o_pv_DescSensor, UL i_ul_SizeOfDescSensor, VOID *o_pv_DescSoftware, UL i_ul_SizeOfDescSoftware)
  Get information on device.
- I **MSO_RegisterAsyncILV** (**MORPHO_HANDLE** i_h_Mso100Handle, I i_us_I, **T_pFuncILV_Buffer** i_p_Callback, PVOID i_pv_context)
  Register Callback function for asynchronous event.
- I **MSO_UnregisterAsyncILV** (**MORPHO_HANDLE** i_h_Mso100Handle, I i_us_I)
  UnRegister Callback function for asynchronous events.
- I **MSO_UnregisterAllAsyncILV** (**MORPHO_HANDLE** i_h_Mso100Handle)
  UnRegister Callback function for all asynchronous events.
- I **MSO_BioDB_CreateDb** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, US i_us_UserMax, UC i_uc_NbFinger, UC i_uc_NormalizedPK_Type, UC i_uc_NbAddField, **PT_FIELD** i_px_AddField, PUC o_puc_ILV_Status)
  Create a Database with various input paramater.
- I **MSO_BioDB_DestroyDb** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, PUC o_puc_ILV_Status, PUL o_pul_EmbeddedError)
  Destroy database.
- I **MSO_BioDB_DeleteUser** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, UL i_ul_IndexUser, PUC o_puc_ILV_Status, PUL o_pul_EmbeddedError)
  Delete User of index i_ul_IndexUser in the database.
- I **MSO_BioDB_GetBaseConfig** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, PUC o_puc_FingerNb, PUL o_pul_MaxRecord, PUL o_pul_CurrentRecord, PUL io_pul_AddFieldNb, **PT_FIELD** o_px_AddField, PUC o_puc_NormalizedPK_Type, PUC o_puc_ILV_Status, PUL o_pul_EmbeddedError)
  Get base configuration.
- I **MSO_BioDB_EraseDb** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, PUC o_puc_ILV_Status, PUL o_pul_EmbeddedError)
  Erase Database.
- I **MSO_BioDB_GetPublicListData** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, UL i_ul_UidData, PUL io_pul_NbTransport, **PT_TRANSPORT_PUBLIC_LIST_FIELD** io_ax_TransportPublicField, PUC o_puc_ILV_Status)
  Get Data list in Database.
- I **MSO_BioDBAddBaseRecord** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, UC i_uc_NbPk, **PT_BUF_PK** i_px_Pk, **PT_BUF** i_px_UserId, UC i_uc_NbAddField, **PT_BUF** i_px_AddField, PUL o_pul_IndexUser, PUC o_puc_ILV_Status, PUC o_puc_Base_Status, BOOL i_b_NoCheckOnTemplate)
  Add record into Database.
- I **MSO_Bio_Enroll** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_IndexDB, US i_us_Timeout, UC i_uc_EnrollmentType, UC i_uc_NbFinger, UC i_uc_SaveRecord, UC i_uc_NormalizedPK_Type, UC i_uc_NbAddField, **PT_BUF** i_px_AddField, UL i_ul_AsynchronousEvent, **PT_EXPORT_PK** io_px_ExportPk, **PT_EXPORT_IMAGE** io_px_ExportImage, PUC o_puc_EnrollStatus, PUL o_pul_UserDBIndex, PUC o_puc_ILV_Status)
  Enrollment function.
- I **MSO_Bio_Identify** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_UidDB, US i_us_Timeout, US i_us_MatchingTreshold, UL i_ul_AsynchronousEvent, PUC o_puc_MatchingResult, PUL o_pul_UserDBIndex, **PT_BUF** o_px_UserID, PUL io_pul_AddFieldNb, **PT_BUF** o_px_AddFieldValue, PUL o_pul_score, PUC o_puc_ILV_Status)

Identify function.

- I **MSO_Bio_Verify** (**MORPHO_HANDLE** i_h_Mso100Handle, US i_us_Timeout, US i_us_MatchingTreshold, UC i_uc_NbFinger, **PT_BUF_PK** i_px_Pk, UL i_ul_AsynchronousEvent, PUC o_puc_MatchingResult, PUL o_pul_score, PUC o_puc_ILV_Status)
  Verify function.
- I **MSO_Bio_IdentifyMatch** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_UidDB, US i_us_MatchingTreshold, UC i_uc_NbPk, **PT_BUF_PK** i_px_Pk, PUC o_puc_MatchingResult, PUL o_pul_UserDBIndex, **PT_BUF** o_px_UserID, PUL o_pul_score, PUC o_puc_ILV_Status)
  Identify match function.
- I **MSO_Bio_VerifyMatch** (**MORPHO_HANDLE** i_h_Mso100Handle, UC i_uc_NbPkSrc, **PT_BUF_PK** i_px_PkSrc, UC i_uc_NbPkRef, **PT_BUF_PK** i_px_PkRef, US i_us_MatchingTreshold, PUC o_puc_MatchingResult, PUC o_puc_ListRefIndex, PUL o_pul_score, PUC o_puc_ILV_Status)
  Verify Match function.
- I **MSO_Cancel** (**MORPHO_HANDLE** i_h_Mso100Handle)
  Cancel a live acquisition.
- I **MSO_SECU_GetSerialNumber** (**MORPHO_HANDLE** i_h_Mso100Handle, UC o_auc_SerialNumber[MSO_SERIAL_NUMBER_LEN], PUC o_puc_SecuConfig, PUS o_pus_SecuMaxFAR, PUC o_puc_ILV_Status)
  Get serial number and Security config.

Define Documentation

#define COM_RS232 "RS232"

RS232 Connection: See MSO_InitCom.

#define COM_USB "USB"

USB Connection: See MSO_InitCom.

#define DEFAULT_BAUD_RATE 115200

RS232 default baudrate: 115200.

#define DEFAULT_COM_INTERFACE COM_USB

Default Com interface: USB.

#define ID_FORMAT_BIN 48

Binary format for Get_Descriptor function.

#define ID_FORMAT_TEXT 47

Text format for Get_Descriptor function.

#define MSO_SERIAL_NUMBER_LEN 24

Serial number max length.

#define SECU_NEVER_EXPORT_SCORE 0x10

If flag is set, the MorphoSmart never export its matching score.

#define SECU_OFFERED_SECURITY 0x02

If flag is set, the MorphoSmart uses the offered security protocol.

#define SECU_PK_ONLY_SIGNED 0x04

The MorphoSmart uses only templates with an X9.84 envelop and a signature.

#define SECU_TUNNELING 0x01

< Secure protocols are not supported on Linux

If flag is set, the MorphoSmart uses the tunneling protocol

Typedef Documentation

typedef I(* T_pFuncILV_Buffer)(PVOID, I, PVOID)

Callback fonction: MSO_RegisterAsyncILV function.

Function Documentation

I MSO_Bio_Enroll (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, US *i_us_Timeout*, UC *i_uc_EnrollmentType*, UC *i_uc_NbFinger*, UC *i_uc_SaveRecord*, UC *i_uc_NormalizedPK_Type*, UC *i_uc_NbAddField*, PT_BUF *i_px_AddField*, UL *i_ul_AsynchronousEvent*, PT_EXPORT_PK *io_px_ExportPk*, PT_EXPORT_IMAGE *io_px_ExportImage*, PUC *o_puc_EnrollStatus*, PUL *o_pul_UserDBIndex*, PUC *o_puc_ILV_Status*)

Enrollment function.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: DataBase index
i_us_Timeout,: Timeout for live finger acquisition, set to 0 for infinite timeout
i_uc_EnrollmentType,: Set to 0 for 2 fingerprints acquisition per finger
i_uc_NbFinger,: Number of finger
i_uc_SaveRecord,: Set to 1 to store record in database
i_uc_NormalizedPK_Type,: Set to 0 for standard database
i_uc_NbAddField,: Number of additionnal field
i_px_AddField,: Add field tab
i_ul_AsynchronousEvent,: Mask of Asynchronous events
io_px_ExportPk,: allocate memory to export PK or set to NULL
io_px_ExportImage,: allocate memory to export Image or Set to NULL
o_puc_EnrollStatus,: Return Enrollment Status
o_pul_UserDBIndex,: Return User database index
o_puc_ILV_Status,: Return Status

Returns:

0 upon success, < 0 if an error occurs

See also:

T_BUF, **T_EXPORT_PK**, **T_EXPORT_IMAGE**

I MSO_Bio_Identify (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_UidDB*, US *i_us_Timeout*, US *i_us_MatchingTreshold*, UL *i_ul_AsynchronousEvent*, PUC *o_puc_MatchingResult*, PUL *o_pul_UserDBIndex*, PT_BUF *o_px_UserID*, PUL *io_pul_AddFieldNb*, PT_BUF *o_px_AddFieldValue*, PUL *o_pul_score*, PUC *o_puc_ILV_Status*)

Identify function.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_UidDB,: DataBase index
i_us_Timeout,: Timeout for live finger acquisition, set to 0 for infinite timeout
i_us_MatchingTreshold,: Set value to 0 to 10, recommended 5
i_ul_AsynchronousEvent,: Mask of Asynchronous events
o_puc_MatchingResult,: Return matching result
o_pul_UserDBIndex,: Return user base index
o_px_UserID,: Return User ID
io_pul_AddFieldNb,: Return number of additionnal field
o_px_AddFieldValue,: Return add field tab
o_pul_score,: Return score, Set o_pul_score to NULL if you don't want score
o_puc_ILV_Status,: Return Status

Returns:

0 upon success, < 0 if an error occurs

I MSO_Bio_IdentifyMatch (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_UidDB*, US *i_us_MatchingTreshold*, UC *i_uc_NbPk*, PT_BUF_PK *i_px_Pk*, PUC *o_puc_MatchingResult*, PUL *o_pul_UserDBIndex*, PT_BUF *o_px_UserID*, PUL *o_pul_score*, PUC *o_puc_ILV_Status*)

Identify match function.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_UidDB,: DataBase index
i_us_MatchingTreshold,: Set value to 0 to 10, recommended 5
i_uc_NbPk,: Number of PK
i_px_Pk,: Buf_Pk tab
o_puc_MatchingResult,: Return matching result
o_pul_UserDBIndex,: Return User database index
o_px_UserID,: Return User ID
o_pul_score,: Return score, Set o_pul_score to NULL if you don't want score
o_puc_ILV_Status,: Return Status

Returns:

0 upon success, < 0 if an error occurs

See also:

T_BUF, **T_BUF_PK**

I MSO_Bio_Verify (MORPHO_HANDLE *i_h_Mso100Handle*, US *i_us_Timeout*, US *i_us_MatchingTreshold*, UC *i_uc_NbFinger*, PT_BUF_PK *i_px_Pk*, UL *i_ul_AsynchronousEvent*, PUC *o_puc_MatchingResult*, PUL *o_pul_score*, PUC *o_puc_ILV_Status*)

Verify function.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_us_Timeout,: Timeout for live finger acquisition, set to 0 for infinite timeout
i_us_MatchingTreshold,: Set value to 0 to 10, recommended 5
i_uc_NbFinger,: number of finger
i_px_Pk,: BUF_PK tab: idPk: ID_PKCOMP, ID_PKCOMP_NORM, ID_PKMAT, ID_PKMAT_NORM, ID_PKMOC, ID_PKBASE_INDEX
i_ul_AsynchronousEvent,: Mask of Asynchronous event
o_puc_MatchingResult,: Return matching result
o_pul_score,: Return score, Set o_pul_score to NULL if you don't want score
o_puc_ILV_Status,: Return Status

Returns:

0 upon success, < 0 if an error occurs

See also:

T_BUF_PK

I MSO_Bio_VerifyMatch (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_NbPkSrc*, PT_BUF_PK *i_px_PkSrc*, UC *i_uc_NbPkRef*, PT_BUF_PK *i_px_PkRef*, US *i_us_MatchingTreshold*, PUC *o_puc_MatchingResult*, PUC *o_puc_ListRefIndex*, PUL *o_pul_score*, PUC *o_puc_ILV_Status*)

Verify Match function.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_NbPkSrc,: Number of source Pk (Must be set to 1)
i_px_PkSrc,: Buf_Pk tab (type: ID_PKCOMP,ID_PKCOMP_NORM,ID_X984_BIO_TOKEN)
i_uc_NbPkRef,: Number of Reference Pk
i_px_PkRef,: Reference Pk tab (type: ID_PKCOMP,ID_PKCOMP_NORM,ID_X984_BIO_TOKEN)
i_us_MatchingTreshold,: Set value to 0 to 10, recommended 5
o_puc_MatchingResult,: Return matching result
o_puc_ListRefIndex,: Return Index in i_px_PkRef tab
o_pul_score,: Return score, Set o_pul_score to NULL if you don't want score
o_puc_ILV_Status,: Return Status

Returns:

0 upon success, < 0 if an error occurs

See also:

T_BUF_PK

I MSO_BioDB_CreateDb (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, US *i_us_UserMax*, UC *i_uc_NbFinger*, UC *i_uc_NormalizedPK_Type*, UC *i_uc_NbAddField*, PT_FIELD *i_px_AddField*, PUC *o_puc_ILV_Status*)

Create a Database with various input paramater.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: Base index, must be set to 0
i_us_UserMax,: Max user in the DataBase
i_uc_NbFinger,: Number of finger per User
i_uc_NormalizedPK_Type,: If the value is different from 0, it means that the database is normalized, otherwise templates are not normalized. Normalization is reserved for specific usage.

i_uc_NbAddField,: Number of additionnal field.
i_px_AddField,: Field Structure: Name, length, right
o_puc_ILV_Status,: Return status

Returns:

0 upon success, < 0 if an error occurs

See also:

PT_FIELD

I MSO_BioDB_DeleteUser (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, UL *i_ul_IndexUser*, PUC *o_puc_ILV_Status*, PUL *o_pul_EmbeddedError*)

Delete User of index i_ul_IndexUser in the database.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: Base index
i_ul_IndexUser,: Index of User to delete
o_puc_ILV_Status,: Return Status
o_pul_EmbeddedError,: Embedded status (0 if no error)

Returns:

0 upon success, < 0 if an error occurs

I MSO_BioDB_DestroyDb (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, PUC *o_puc_ILV_Status*, PUL *o_pul_EmbeddedError*)

Destroy database.

The Database is totaly destroy. We must use the MSO_BioDB_CreateDb function to create a new base with new parameters

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: Base index
o_puc_ILV_Status,: Return status
o_pul_EmbeddedError,: Return Embedded status (0 if no error)

Returns:

0 upon success, < 0 if an error occurs

I MSO_BioDB_EraseDb (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, PUC *o_puc_ILV_Status*, PUL *o_pul_EmbeddedError*)

Erase Database.

The function erase all record in database but doesn't destroy the database structure

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: DataBase index
o_puc_ILV_Status,: Return Status
o_pul_EmbeddedError,: Return Embedded Status

Returns:

0 upon success, < 0 if an error occurs

I MSO_BioDB_GetBaseConfig (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, PUC *o_puc_FingerNb*, PUL *o_pul_MaxRecord*, PUL *o_pul_CurrentRecord*, PUL

***io_pul_AddFieldNb*, PT_FIELD *o_px_AddField*, PUC *o_puc_NormalizedPK_Type*, PUC *o_puc_ILV_Status*, PUL *o_pul_EmbeddedError*)**

Get base configuration.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: DataBase index
o_puc_FingerNb,: Number of finger per User
o_pul_MaxRecord,: Max Number of user in the DataBase
o_pul_CurrentRecord,: Number of User recorded in the Database
io_pul_AddFieldNb,: Number of additionnal field
o_px_AddField,: Add Field Structure
o_puc_NormalizedPK_Type,: Set to 0 for standard database
o_puc_ILV_Status,: Return Status
o_pul_EmbeddedError,: Embedded status (0 if no error)

Returns:

0 upon success, < 0 if an error occurs

See also:

PT_FIELD

I MSO_BioDB_GetPublicListData (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, UL *i_ul_UidData*, PUL *io_pul_NbTransport*, PT_TRANSPORT_PUBLIC_LIST_FIELD *io_ax_TransportPublicField*, PUC *o_puc_ILV_Status*)

Get Data list in Database.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: DataBase index
i_ul_UidData,: Data type (0: UserID, 1: Add Field1, 2: Add Field2 ...)
io_pul_NbTransport,: Number of Data
io_ax_TransportPublicField,: Array of Data Structure
o_puc_ILV_Status,: Return Status

Returns:

0 upon success, < 0 if an error occurs io_pul_NbTransport must be set to number of structure io_ax_TransportPublicField allocated Menmber m_puc_Data in io_ax_TransportPublicField must be also allocated

See also:

T_TRANSPORT_PUBLIC_LIST_FIELD

I MSO_BioDBAddBaseRecord (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_IndexDB*, UC *i_uc_NbPk*, PT_BUF_PK *i_px_Pk*, PT_BUF *i_px_UserId*, UC *i_uc_NbAddField*, PT_BUF *i_px_AddField*, PUL *o_pul_IndexUser*, PUC *o_puc_ILV_Status*, PUC *o_puc_Base_Status*, BOOL *i_b_NoCheckOnTemplate*)

Add record into Database.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_uc_IndexDB,: DataBase index

i_uc_NbPk,: Number of reference template
i_px_Pk,: BUF_PK tab: idPk: ID_PKCOMP, ID_PKCOMP_NORM, ID_PKMAT, ID_PKMAT_NORM, ID_PKMOC, ID_PKBASE_INDEX
i_px_UserId,: User ID
i_uc_NbAddField,: Number of additionnal field
i_px_AddField,: AddField Tab
o_pul_IndexUser,: Return index of user added in database
o_puc_ILV_Status,: Return Status
o_puc_Base_Status,: Return Base Status
i_b_NoCheckOnTemplate,: Check on template flag

Returns:

0 upon success, < 0 if an error occurs

See also:

PT_BUF_PK, **PT_BUF**

I MSO_Cancel (MORPHO_HANDLE *i_h_Mso100Handle*)

Cancel a live acquisition.

Parameters:

i_h_Mso100Handle,: Handle to the device

Returns:

0 upon success, < 0 if an error occurs

I MSO_CloseCom (MORPHO_HANDLE * *io_ph_Mso100Handle*)

Close the communication.

Parameters:

io_ph_Mso100Handle,: Handle to the device. Set to NULL after closing the communication

Returns:

0 upon success, < 0 if an error occurs

I MSO_GetDescriptor (MORPHO_HANDLE *i_h_Mso100Handle*, UC *i_uc_DescFormat*, PUC *o_puc_ILV_Status*, VOID * *o_pv_DescProduct*, UL *i_ul_SizeOfDescProduct*, VOID * *o_pv_DescSensor*, UL *i_ul_SizeOfDescSensor*, VOID * *o_pv_DescSoftware*, UL *i_ul_SizeOfDescSoftware*)

Get information on device.

Parameters:

i_h_Mso100Handle,: Handle to the device.
i_uc_DescFormat,: Format output: ID_FORMAT_TEXT or ID_FORMAT_BIN
o_puc_ILV_Status,: Return status
o_pv_DescProduct,: Product description
i_ul_SizeOfDescProduct,: size of memory allocated for i_pv_DescProduct
o_pv_DescSensor,: Sensor description
i_ul_SizeOfDescSensor,: size of memory allocated for i_pv_DescSensor
o_pv_DescSoftware,: Sofware description
i_ul_SizeOfDescSoftware,: size of memory allocated for i_pv_DescSoftware

Returns:

0 upon success, < 0 if an error occurs

I MSO_InitCom (MORPHO_HANDLE * *i_ph_Mso100Handle*, PC *i_str_Interface*, PC *i_str_comName*, I *i_i_baudRate*)

Initialize the connnection with the MorphoSmart.

Parameters:

i_ph_Mso100Handle,: Handle to the device.
i_str_Interface,: Interface type: COM_USB or COM_RS232.
i_str_comName,: Serial device name: ex /dev/ttyS0. Don't forget to change rigth if needed
i_i_baudRate,: RS232 connection baud rate: max 115200.

Returns:

0 upon success, < 0 if an error occurs

I MSO_RegisterAsyncILV (MORPHO_HANDLE *i_h_Mso100Handle*, I *i_us_I*, T_pFuncILV_Buffer *i_p_Callback*, PVOID *i_pv_context*)

Register Callback function for asynchronous event.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_us_I,: Mask of T_MORPHO_CALLBACK_COMMAND
i_p_Callback,: Callback function. called when async events occurs
i_pv_context,: Context to send to the Callback function

Returns:

0 upon success, < 0 if an error occurs

See also:

T_MORPHO_CALLBACK_COMMAND

I MSO_SECU_GetSerialNumber (MORPHO_HANDLE *i_h_Mso100Handle*, UC *o_auc_SerialNumber*[MSO_SERIAL_NUMBER_LEN], PUC *o_puc_SecuConfig*, PUS *o_pus_SecuMaxFAR*, PUC *o_puc_ILV_Status*)

Get serial number and Security config.

Parameters:

i_h_Mso100Handle,: handle to the device
o_auc_SerialNumber,: Return serial number
o_puc_SecuConfig,: Return Secu config Mask
o_pus_SecuMaxFAR,: Return Max Far, Command with MathingTreshold<SecuMaxFAR are rejected
o_puc_ILV_Status,: Return Status

Returns:

0 upon success, < 0 if an error occurs

I MSO_UnregisterAllAsyncILV (MORPHO_HANDLE *i_h_Mso100Handle*)

UnRegister Callback function for all asynchronous events.

Parameters:

i_h_Mso100Handle,: Handle to the device

Returns:

0 upon success, < 0 if an error occurs

I MSO_UnregisterAsyncILV (MORPHO_HANDLE *i_h_Mso100Handle*, I *i_us_I*)

UnRegister Callback function for asynchronous events.

Parameters:

i_h_Mso100Handle,: Handle to the device
i_us_I,: Mask of T_MORPHO_CALLBACK_COMMAND

Returns:

0 upon success, < 0 if an error occurs

See also:

T_MORPHO_CALLBACK_COMMAND

I MSO_Usb_EnumDevices (PT_MSO_USB_DEVICE_PROPERTIES * *o_ppx_DeviceProperties*, PUL *o_pul_NbreDevices*)

Enumerate devices connected on USB bus.

This function allocate memory and fill the Device properties structure

Parameters:

o_ppx_DeviceProperties,: point to a tab of Device properties structure
o_pul_NbreDevices,: point to the number of device connected

Returns:

0 upon success, < 0 if an error occurs

I MSO_Usb_ReleaseEnumDevices (PT_MSO_USB_DEVICE_PROPERTIES * *o_ppx_DeviceProperties*, UL *i_ul_NbreDevices*)

Release Device properties structure.

this function release memory and and set o_ppx_DeviceProperties to NULL

Parameters:

o_ppx_DeviceProperties,: point to a tab of Device properties structure
i_ul_NbreDevices,: number of device structure to release

Returns:

0 upon success, < 0 if an error occurs

See also:

T_MSO_USB_DEVICE_PROPERTIES

libMSO_Def.h File Reference

Defines

- #define **VOID** void
- #define **UC** unsigned char
- #define **C** char
- #define **US** unsigned short
- #define **S** short
- #define **UL** unsigned long
- #define **L** long
- #define **I** int
- #define **UI** unsigned int
- #define **PUI** unsigned int*
- #define **PVOID** void*
- #define **PUC** unsigned char*
- #define **PC** char*
- #define **PUS** unsigned short*
- #define **PS** short*
- #define **PUL** unsigned long*
- #define **PL** long*
- #define **PI** int*
- #define **BOOLEAN** UC
- #define **BOOL** UC
- #define **B** UC
- #define **DWORD** unsigned long
- #define **HANDLE** void *
- #define **RETURN_NO_ERROR** 0
  No error.
- #define **ILV_OK** 0x00
  Successful result.
- #define **ILVERR_ERROR** 0xFF
  An error occurred.
- #define **ILVERR_BADPARAMETER** 0xFE
  Input parameters are not valid.
- #define **ILVERR_INVALID_MINUTIAE** 0xFD
  The minutiae is not valid.
- #define **ILVERR_INVALID_USER_ID** 0xFC
  The record identifier does not exist in the database.
- #define **ILVERR_INVALID_USER_DATA** 0xFB
  The user data are not valid.
- #define **ILVERR_TIMEOUT** 0xFA
  No response after defined time.
- #define **ILVERR_INVALID_ID_PROTOCOL** 0xF9
  The protocole used is not valid.
- #define **ILVERR_ALREADY_ENROLLED** 0xF8
  The person is already in the base.
- #define **ILVERR_BASE_NOT_FOUND** 0xF7

The specified base does not exist.

- #define **ILVERR_BASE_ALREADY_EXISTS** 0xF6
  The specified base already exist.
- #define **ILVERR_BIO_IN_PROGRESS** 0xF5
  Command received during biometric processing.
- #define **ILVERR_CMD_INPROGRESS** 0xF4
  Command received while another command is running.
- #define **ILVERR_FLASH_INVALID** 0xF3
  Flash type invalid.
- #define **ILVERR_NO_SPACE_LEFT** 0xF2
  Not Enough memory for the creation of a database.
- #define **ILVERR_FIELD_NOT_FOUND** 0xE9
  Field does not exist.
- #define **ILVERR_FIELD_INVALID** 0xE8
  Field size or field name is invalid.
- #define **ILVERR_SECURITY_MODE** 0xE7
  The request is not compatible with security mode.
- #define **ILVERR_USER_NOT_FOUND** 0xE6
  The searched user is not found.
- #define **ILVERR_CMDE_ABORTED** 0xE5
  Commanded has been aborted by the user.
- #define **ILVERR_SAME_FINGER** 0xE4
  There are two templates of the same finger.
- #define **ILVERR_NO_HIT** 0xE3
  Presented finger does not match.
- #define **ILVERR_FFD** 0xDB
  False finger detected.
- #define **ILVERR_MOIST_FINGER** 0xDA
  Too moist finger detected.
- #define **ILVERR_NOT_IMPLEMENTED** 0x9D
  The request is not yet implemented.
- #define **ILVSTS_OK** 0
  Successful.
- #define **ILVSTS_HIT** 1
  Authentication or Identification succeed.
- #define **ILVSTS_NO_HIT** 2
  Authentication or Identification failed.
- #define **ILVSTS_LATENT** 3
  Security Protection Triggered.
- #define **ILVSTS_DB_FULL** 4
  The database is full.
- #define **ILVSTS_DB_EMPTY** 5
  The database is empty.
- #define **ILVSTS_BAD_QUALITY** 6

Bad finger and/or enroll quality.

- #define **ILVSTS_DB_OK** 7
  The database is right.
- #define **ILVSTS_ACTIVATED** 8
  The MorphoModule is activated.
- #define **ILVSTS_NOTACTIVATED** 9
  The MorphoModule is not activated.
- #define **ILVSTS_DB_KO** 10
  The flash can not be accessed.
- #define **ILVSTS_FFD** 0x22
  False finger detected.
- #define **ILVSTS_MOIST_FINGER** 0x23
  Too moist finger detected.

Typedefs

- typedef void * **MORPHO_HANDLE**
  Handle to a device.

Define Documentation

#define B UC

#define BOOL UC

#define BOOLEAN UC

#define C char

#define DWORD unsigned long

#define HANDLE void *

#define I int

#define ILV_OK 0x00

Successful result.

Status code return by function: o_puc_ILV_Status.

#define ILVERR_ALREADY_ENROLLED 0xF8

The person is already in the base.

#define ILVERR_BADPARAMETER 0xFE

Input parameters are not valid.

#define ILVERR_BASE_ALREADY_EXISTS 0xF6

The specified base already exist.

#define ILVERR_BASE_NOT_FOUND 0xF7

The specified base does not exist.

#define ILVERR_BIO_IN_PROGRESS 0xF5

Command received during biometric processing.

#define ILVERR_CMD_INPROGRESS 0xF4

Command received while another command is running.

#define ILVERR_CMDE_ABORTED 0xE5

Commanded has been aborted by the user.

#define ILVERR_ERROR 0xFF

An error occurred.

#define ILVERR_FFD 0xDB

False finger detected.

#define ILVERR_FIELD_INVALID 0xE8

Field size or field name is invalid.

#define ILVERR_FIELD_NOT_FOUND 0xE9

Field does not exist.

#define ILVERR_FLASH_INVALID 0xF3

Flash type invalid.

#define ILVERR_INVALID_ID_PROTOCOL 0xF9

The protocole used is not valid.

#define ILVERR_INVALID_MINUTIAE 0xFD

The minutiae is not valid.

#define ILVERR_INVALID_USER_DATA 0xFB

The user data are not valid.

#define ILVERR_INVALID_USER_ID 0xFC

The record identifier does not exist in the database.

#define ILVERR_MOIST_FINGER 0xDA

Too moist finger detected.

#define ILVERR_NO_HIT 0xE3

Presented finger does not match.

#define ILVERR_NO_SPACE_LEFT 0xF2

Not Enough memory for the creation of a database.

#define ILVERR_NOT_IMPLEMENTED 0x9D

The request is not yet implemented.

#define ILVERR_SAME_FINGER 0xE4

There are two templates of the same finger.

#define ILVERR_SECURITY_MODE 0xE7

The request is not compatible with security mode.

#define ILVERR_TIMEOUT 0xFA

No response after defined time.

#define ILVERR_USER_NOT_FOUND 0xE6

The searched user is not found.

#define ILVSTS_ACTIVATED 8

The MorphoModule is activated.

#define ILVSTS_BAD_QUALITY 6

Bad finger and/or enroll quality.

#define ILVSTS_DB_EMPTY 5

The database is empty.

#define ILVSTS_DB_FULL 4

The database is full.

#define ILVSTS_DB_KO 10

The flash can not be accessed.

#define ILVSTS_DB_OK 7

The database is right.

#define ILVSTS_FFD 0x22

False finger detected.

#define ILVSTS_HIT 1

Authentication or Identification succeed.

#define ILVSTS_LATENT 3

Security Protection Triggered.

#define ILVSTS_MOIST_FINGER 0x23

Too moist finger detected.

#define ILVSTS_NO_HIT 2

Authentication or Identification failed.

#define ILVSTS_NOTACTIVATED 9

The MorphoModule is not activated.

#define ILVSTS_OK 0

Successful.

Status Codes definition :

#define L long

#define PC char*

#define PI int*

#define PL long*

#define PS short*

#define PUC unsigned char*

#define PUI unsigned int*

#define PUL unsigned long*

#define PUS unsigned short*

#define PVOID void*

#define RETURN_NO_ERROR 0

No error.

#define S short

#define UC unsigned char

#define UI unsigned int

#define UL unsigned long

#define US unsigned short

#define VOID void

Simple type

Typedef Documentation

typedef void* MORPHO_HANDLE

Handle to a device.

libMSO_Struct.h File Reference

Data Structures

- struct **T_MSO_USB_DEVICE_PROPERTIES**
  Device properties structure.
- struct **T_BUF**
  Data Buffer structure.
- struct **T_BIODB_DATA_TRANSPORT**
  Bio data structure.
- struct **T_FIELD**
  Field structure.
- struct **T_TRANSPORT_PUBLIC_LIST_FIELD**
  Public list field structure.
- struct **T_ILV_DB**
  ILV database structure.
- struct **T_ILV_ADD_FIELD**
  Additionnal field structure.
- struct **T_ILV_BASE_CONFIG**
  Base configuration structure.
- struct **T_BUF_PK**
  Public list field structure.
- struct **T_EXPORT_PK**
  Export Pk Structure.
- struct **T_EXPORT_IMAGE**
  Export Image Structure.
- struct **T_MORPHO_IMAGE_HEADER**
  Image header structure.
- struct **T_MORPHO_CALLBACK_ENROLLMENT_STATUS**
  Enrollment command status.
- struct **T_MORPHO_CALLBACK_IMAGE_STATUS**
  Image status.

Defines

- #define **MAX_FIELD_NAME_LEN** 6
  Maximun length of field name.
- #define **PUBLIC** 0
  Public field ID.
- #define **PRIVATE** 1
  Private field ID.
- #define **ID_PKCOMP** 2
- #define **ID_PKMAT** 3
- #define **ID_PKMAT_NORM** 53
- #define **ID_PKCOMP_NORM** 55

- #define **ID_PKBASE** 58
- #define **ID_PKMOC** 59

Typedefs

- typedef struct **T_MSO_USB_DEVICE_PROPERTIES** * **PT_MSO_USB_DEVICE_PROPERTIES**
- typedef struct **T_BUF** * **PT_BUF**
- typedef struct **T_BIODB_DATA_TRANSPORT** * **PT_BIODB_DATA_TRANSPORT**
- typedef struct **T_FIELD** * **PT_FIELD**
- typedef struct **T_TRANSPORT_PUBLIC_LIST_FIELD** * **PT_TRANSPORT_PUBLIC_LIST_FIELD**
- typedef struct **T_ILV_DB** * **PT_ILV_DB**
- typedef struct **T_ILV_ADD_FIELD** * **PT_ILV_ADD_FIELD**
- typedef struct **T_ILV_BASE_CONFIG** * **PT_ILV_BASE_CONFIG**
- typedef struct **T_BUF_PK** * **PT_BUF_PK**
- typedef struct **T_EXPORT_PK** * **PT_EXPORT_PK**
- typedef struct **T_EXPORT_IMAGE** * **PT_EXPORT_IMAGE**
- typedef enum **T_MORPHO_FAR** * **PT_MORPHO_FAR**
- typedef struct **T_MORPHO_CALLBACK_IMAGE_STATUS T_MORPHO_IMAGE**
- typedef I(* **T_MORPHO_CALLBACK_FUNCTION**)(PVOID i_pv_context, **T_MORPHO_CALLBACK_COMMAND** i_i_command, PVOID i_pv_param)
  Callback function prototype.