Advanced Card Systems Ltd.
Card & Reader Technologies

AIR60U

ePassport Reader

Application Programming Interface

V1.06

info@acs.com.hk
www.acs.com.hk

Table of Contents

List of Figures

List of Tables

1.0. Introduction

The **AIR60U ePassport Reader** is an ID document reader built for fast and reliable identity verification in space-limited environments. Its slim, modular design integrates **Optical Character Recognition (OCR), RFID/NFC, and contact smart card** reading into a single device, enabling rapid capture of identity data from **ICAO DOC 9303**-compliant e-passports and ID cards.

An optional **QR code scanner** expands support for digital identity documents, making the AIR60U a versatile choice for a wide range of applications. With its compact size, and cost-efficient architecture, the AIR60U is well-suited for deployment in hotels, car rental counters, travel agencies, transportation terminals, and other high-throughput service environments.

By automating identity data capture, the AIR60U improves processing speed, reduces manual errors, and enhances customer experience — all while fitting seamlessly into kiosks, e-gates, or desktop setups.

This document is to introduce the API of AIR60U ePassport and QR Code Scanner related features.

2.0. Standard Workflow

This section illustrates the standard API workflow of the ACS AIR60U operating in both Auto Mode and Manual Mode.

2.1. Auto Mode

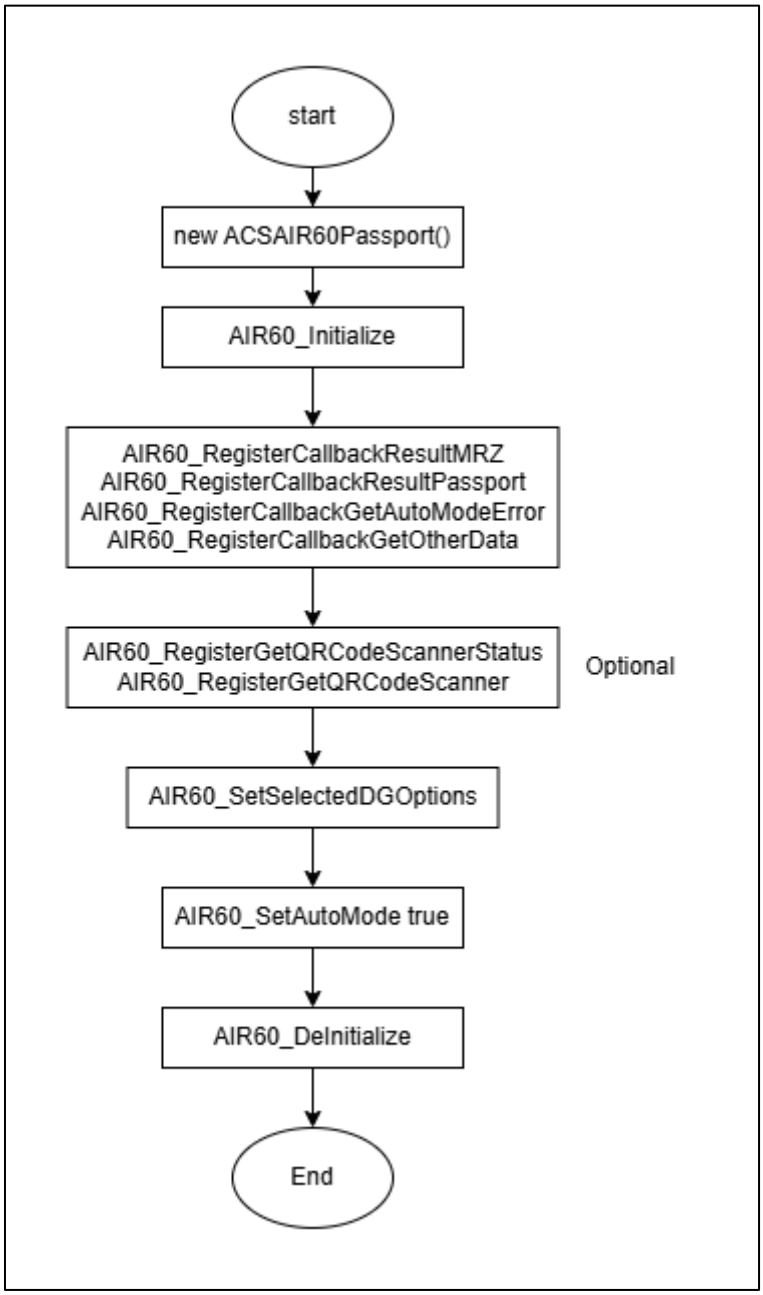

Figure 1: AIR60U Auto Mode Workflow

2.2. Manual Mode

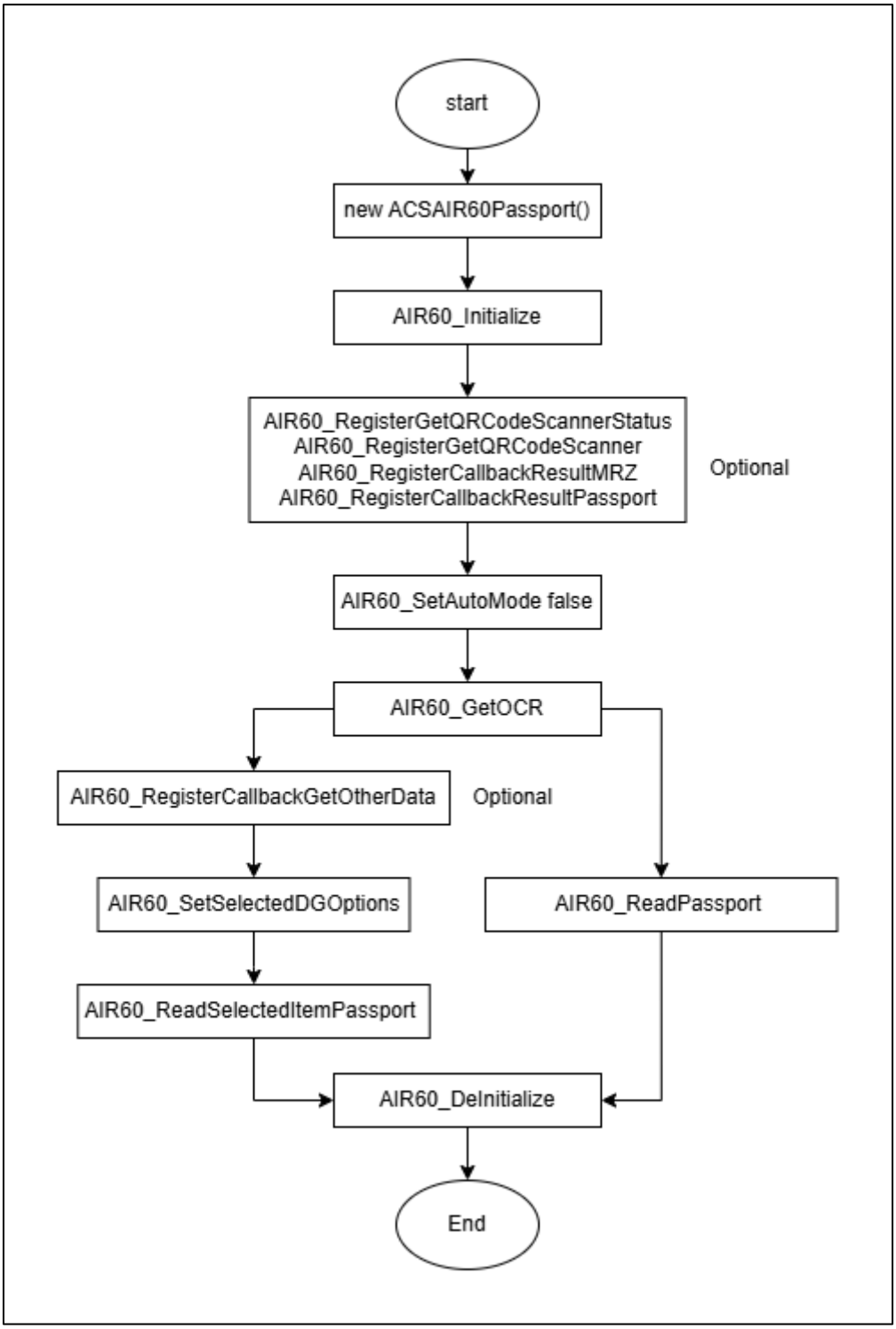

Figure 2: AIR60U Manual Mode Workflow

3.0. AIR60U API

3.1. Using AIR60U API

The AIR60U API provides interfaces for accessing the document reader and retrieving QR Code Scanner related information. These interfaces are implemented in C# and provided in the form of DLL.

To use the AIR60 API, please include the following code into your project:

using ACS.AIR60.PassportAPI

Please import the following DLL files into your project:

AIR60PassportAPI.dll

AIR60ManageCardDll.dll

3.1.1. Opening Device

Use AIR60_Initialize to initialize the device.

3.1.2. Get Chip Data

Auto mode:

Get data using the function registered with AIR60_RegisterCallbackResultPassport.

Manual mode:

Use AIR60_GetOCR to read MRZ. Optionally invoke AIR60_ValidateMRZInfo to verify the correctness of the MRZ information, then call AIR60_ReadSelectedItemPassport to retrieve the chip data.

3.1.3. Closing Device

Use AIR60_DeInitialized to close the device releases all resources; then the device must be reinitialized to reuse it.

3.1.4. Get Device Info

Use AIR60_GetFirmwareVersion to retrieve the device's firmware info

Use AIR60_GetOCRVersion to retrieve the OCR's version info

Use AIR60_GetSerialNumber to retrieve the device's serial number

Use AIR60_GetProductInfo to retrieve the device's product info

3.2. Device

3.2.1. Function Documentation

3.2.1.1. AIR60_Initialize

Format:

bool AIR60_Initialize()

Function Description: Initialize AIR60U.

This function initializes AIR60U.

Parameters		Description
None		-
Return Value	*True*	The operation completed successfully.
	False	An AIR60APIException is thrown.

Table 1: AIR60_Initialize Function Description

Source Code Example:

```
private ACSAIR60Passport acsAIR60Passport;
acsAIR60Passport = new ACSAIR60Passport();
...
bool bAIR60Init = false;
...
 try
 {
     ...
     bAIR60Init = acsAIR60Passport.AIR60_Initialize();
     ...
 }
 catch (AIR60APIException ex)
{
     ...
```

}

3.2.1.2. AIR60_DeInitialize

Format:

bool AIR60_DeInitialize()

Function Description: Deinitialize AIR60U.

This function deinitializes AIR60U and releases its resource.

Parameters		Description
None		-
Return Value	*True*	The operation completed successfully.
	False	The operation failed.

Table 2: AIR60_DeInitialize Function Description

Source Code Example:

```
if (acsAIR60Passport != null)
{
    acsAIR60Passport.AIR60_DeInitialize();
    acsAIR60Passport = null;
}
```

3.2.1.3. AIR60_SetAutoMode

Format:

void AIR60_SetAutoMode(bool isAutoMode)

Function Description: Set Operation Mode.

This function sets the AIR60U's operating mode. In Auto mode, the AIR60 automatically checks for card insertion. Upon detection, it reads the passport's MRZ information automatically and only retrieves selected Data Group data, with each card insertion triggering a single read operation. The selected Data Groups are configured via SetSelectedDGOptions, and the selected Data Group data is retrieved through the function registered by AIR60_RegisterCallbackResultPassport.

Parameters		Description
isAutoMode[in]		True: Auto Mode False: Manual Mode (Default)
Return Value	*None*	-

Table 3: AIR60_SetAutoMode Function Description

Source Code Example:

```
acsAIR60Passport.AIR60_SetAutoMode(true);
```

3.2.1.4. AIR60_IsAutoMode

Format:

bool AIR60_IsAutoMode()

Function Description: Get Operation Mode.

This function reads the AIR60U's operating mode.

Parameters		Description
None		-
Return Value	*True*	Auto Mode
	False	Manual Mode

Table 4: AIR60_IsAutoMode Function Description

Source Code Example:

```
if (acsAIR60Passport.AIR60_IsAutoMode())
{
    ...
}
```

3.2.1.5. AIR60_IsConnected

Format:

bool AIR60_IsConnected()

Function Description: Get Connection Status.

This function reads the AIR60U's connection status.

Parameters		Description
None		-
Return Value	*True*	AIR60 successfully connected.
	False	Failed to connect.

Table 5: AIR60_IsConnected Function Description

Source Code Example:

```
if (acsAIR60Passport.AIR60_IsConnected())
```

3.2.1.6. AIR60_Disconnect

Format:

bool AIR60_Disconnect()

Function Description: Disconnect AIR60U.

This function disconnects the AIR60U.

Parameters		Description
None		-
Return Value	*True*	The operation completed successfully.

Parameters		Description
	False	The operation failed.

Table 6: AIR60_ Disconnect Function Description

Source Code Example:

```
acsAIR60Passport.AIR60_Disconnect();
```

3.2.1.7. AIR60_GetFirmwareVersion

Format:

string AIR60_GetFirmwareVersion()

Function Description: Get AIR60U Firmware Version.

This function reads AIR60U firmware version.

Parameters		Description
none		-
Return Value	*String*	AIR60U Firmware Version

Table 7: AIR60_GetFirmwareVersion Function Description

Source Code Example:

```
string AIR60FWVer = string.Empty;
AIR60FWVer = acsAIR60Passport.AIR60_GetFirmwareVersion();
```

3.2.1.8. AIR60_GetOCRVersion

Format:

string AIR60_GetOCRVersion()

Function Description: Get AIR60U OCR Version.

This function reads AIR60U OCR version.

Parameters		Description
none		-
Return Value	*String*	AIR60U OCR Version

Table 8: AIR60_GetOCRVersion Function Description

Source Code Example:

```
string AIR60OCRVer = string.Empty;
AIR60OCRVer = acsAIR60Passport.AIR60_GetOCRVersion();
```

3.2.1.9. AIR60_GetSerialNumber

Format:

string AIR60_GetSerialNumber()

Function Description: Get AIR60U Serial Number.

This function reads AIR60U serial number.

Parameters		Description
none		-
Return Value	*String*	AIR60U Serial Number

Table 9: AIR60_GetSerialNumber Function Description

Source Code Example:

```
string AIR60SerNO = string.Empty;
AIR60SerNO = acsAIR60Passport.AIR60_GetSerialNumber();
```

3.2.1.10. AIR60_GetProductInfo

Format:

string AIR60_GetProductInfo()

Function Description: Get AIR60U Product Information.

This function reads AIR60U product information.

Parameters		Description
none		-
Return Value	*String*	AIR60U Product Information

Table 10: AIR60_GetProductInfo Function Description

Source Code Example:

```
String AIR60PInfo = string.Empty;

AIR60PInfo = acsAIR60Passport.AIR60_GetProductInfo();
```

3.3. Passport Data

3.3.1. Function Documentation

3.3.1.1. AIR60_GetOCR

Format:

string AIR60_GetOCR()

Function Description: Get MRZ Information.

This function reads the Passport's MRZ information by OCR.

Parameters		Description
None		-
Return Value	*String*	Passport MRZ Information.
	Exception	An AIR60APIException is thrown.

Table 11: AIR60_GetOCR Function Description

Source Code Example:

```
string strPassMRZ = string.Empty;
...
try
{
...
strPassMRZ = acsAIR60Passport.AIR60_GetOCR();
...
}
catch (AIR60APIException ex)
{
...
}
```

3.3.1.2. AIR60_ReadPassport

Format:

PassportReadResult AIR60_ReadPassport(string MRZInfo)

Function Description: Read All Data Groups of Passport.

This function reads all data groups within the Passport.

Parameters		Description
MRZInfo[in]		MRZ information
Return Value	PassportReadResult	Data Structure of the Data Groups
	Exception	An AIR60APIException is thrown.

Table 12: AIR60_ReadPassport Function Description

Source Code Example:

```
string strMRZ = this.txtMRZInfo.Text;
...
try
{
    ...
    passPortData = acsAIR60Passport.AIR60_ReadPassport(strMRZ);
    ...
}
catch (AIR60APIException ex)
{
    ...
}
```

3.3.1.3. AIR60_SetSelectedDGOptions

Format:

void AIR60_SetSelectedDGOptions(List<DatagroupType> selectedDGTypeList)

Function Description: Set the Selected Data Groups of Passport.

This function configures the selected data groups within the Passport.

Parameters		Description
selectedDGTypeList		List of Data Groups to be read.
Return Value	*None*	-

Table 13: AIR60_SetSelectedDGOptions Function Description

Note:

1. All DGs are read if selectedDGTypeList is null.

2. No DGs are read if selectedDGTypeList is not null but contains no elements.

3. If selectedDGTypeList is not null and contains elements, only DGs specified by the element list are read.

Source Code Example:

```
List<DatagroupType> selectedDGTypeList = null;
try
{
    ...
    if (selectedDGTypeList == null)
    {
        selectedDGTypeList = new List<DatagroupType>();
        selectedDGTypeList.Add(DatagroupType.DataCOM);
        selectedDGTypeList.Add(DatagroupType.DataGroup1);
        selectedDGTypeList.Add(DatagroupType.DataGroup2);
    }
    acsAIR60Passport.AIR60_SetSelectedDGOptions(selectedDGTypeList);
    passPortData = acsAIR60Passport.AIR60_ReadSelectedItemPassport(strMRZ);
}
catch (Exception ex)
{
  ...
}
```

3.3.1.4. AIR60_ReadSelectedItemPassport

Format:

PassportReadResult ReadSelectedItemPassport(string MRZInfo)

Function Description: Read the Selected Data Groups of Passport.

This function reads the selected data groups within the Passport.

Parameters		Description
MRZInfo[in]		MRZ information
Return Value	PassportReadResult	Data Structure of the Selected Data Groups which configured via AIR60_SetSelectedDGOptions
	Exception	An AIR60APIException is thrown.

Table 14: AIR60_ReadSelectedItemPassport Function Description

Source Code Example:

```
PassportReadResult passPortData = null;
string strMRZ = this.txtMRZInfo.Text;
Try
{
    ...

    passPortData = acsAIR60Passport.AIR60_ReadSelectedItemPassport(strMRZ);
    ...
}
catch (AIR60APIException ex)
{
    ...
}
```

3.3.1.5. AIR60_ValidateMRZInfo

Format:

bool AIR60_ValidateMRZInfo(string sMRZInfo)

Function Description: Validate the MRZ of Passport.

This function validates the MRZ information of Passport.

Parameters		Description
MRZInfo[in]		MRZ information
Return Value	*True*	The MRZ is valid.
	False	The MRZ is invalid.

Table 15: AIR60_ValidateMRZInfo Function Description

Source Code Example:

```
string strMRZ = this.txtMRZInfo.Text;
if (acsAIR60Passport.AIR60_ValidateMRZInfo(strMRZ))
{
    ...
}
```

3.3.1.6. AIR60_RegisterCallbackResultMRZ

Format:

void AIR60_RegisterCallbackResultMRZ(AIR60DelegateReadMRZ readMRZ)

Function Description: Register the MRZ Result Function.

This function register callback of the ResultMRZ function.

Parameters		Description
readMRZ		A function that retrieves MRZ information, see below.
Return Value	*None*	-

Table 16: AIR60_RegisterCallbackResultMRZ Function Description

3.3.1.6.1. AIR60DelegateReadMRZ

Format:

public delegate void AIR60DelegateReadMRZ(string MRZ, bool bMRZIsRight)

Function Description: MRZ Result Function.

This function retrieves MRZ information.

Parameters		Description
MRZ[in]		MRZ information of Passport.
bMRZIsRight[in]		If the MRZ is valid.
Return Value	*None*	-

Table 17: AIR60DelegateReadMRZ Function Description

Source Code Example:

```
AIR60DelegateReadMRZ m_delegateReadMRZ;
m_delegateReadMRZ = new AIR60DelegateReadMRZ(readMRZ);
acsAIR60Passport.AIR60_RegisterCallbackResultMRZ(m_delegateReadMRZ);
private void readMRZ(string mrz, bool bMRZIsRight)
{
    if (this.InvokeRequired)
    {
        this.BeginInvoke(new Action(() =>
        {
            readMRZ(mrz, bMRZIsRight);
        }));
            return;
    }
    if (mrz != null)
    {
        ...
        if (mrz != string.Empty)
        {
        if (bMRZIsRight)
```

```
            ...
            }
            ...
        }
        ...
    }
```

3.3.1.7. AIR60_RegisterCallbackResultPassport

Format:

void AIR60_RegisterCallbackResultPassport(AIR60DelegateGetPassportDataread getPassportData)

Function Description: Register the Get Passport Data Function.

This function register callback of the getPassportData function.

Parameters	Description	
getPassportData	A function that read Passport data, see below.	
Return Value	*None*	-

Table 18: AIR60_RegisterCallbackResultPassport Function Description

3.3.1.7.1. AIR60DelegateGetPassportDataread

Format:

public delegate void AIR60DelegateGetPassportData(PassportReadResult passportResult)

Function Description: Read Passport Information Function.

This function retrieves passport chip information.

Parameters	Description	
passportResult[in]	The Passport chip information retrieved.	
Return Value	*None*	-

Table 19: AIR60DelegateGetPassportDataread Function Description

Source Code Example:

```
AIR60DelegateGetPassportData m_deleteGetPassportDate;
m_deleteGetPassportDate = new AIR60DelegateGetPassportData(getPassportDate);
acsAIR60Passport.AIR60_RegisterCallbackResultPassport(m_deleteGetPassport
Date);
private void getPassportDate(PassportReadResult passportResult)
{
    if (this.InvokeRequired)
    {
        this.BeginInvoke(new Action(() =>
        {
            getPassportDate(passportResult);
        }));
        return;
    }
    toolStripProgressBar1.Visible = false;
    passPortData = passportResult;
    updateStatus(passPortData);
    PrintPassportData(passPortData);
}
```

3.3.1.8. AIR60_RegisterCallbackGetAutoModeError

Format:

void AIR60_RegisterCallbackGetAutoModeError(AIR60DelegateGetAutoModeEror GetAutomodeErrorFunc)

Function Description: Register the Get Auto Mode Error Function.

This function register callback of the GetAutomodeErrorFunc function.

Parameters	Description
GetAutomodeErrorFunc	A function that reads error messages in Auto mode, see below.

Parameters		Description
Return Value	*None*	-

Table 20: AIR60_RegisterCallbackGetAutoModeError Function Description

3.3.1.8.1. AIR60DelegateGetAutoModeEror

Format:

public delegate void AIR60DelegateGetAutoModeEror(string ErrorMessage)

Function Description: Read Auto Mode Error Function.

This function retrieves error message of auto mode.

Parameters		Description
ErrorMessage[in]		The error message in Auto mode.
Return Value	*None*	-

Table 21: AIR60DelegateGetAutoModeEror Function Description

Source Code Example:

```
AIR60DelegateGetAutoModeEror m_deleteGetAutoModeError;
m_deleteGetAutoModeError = new AIR60DelegateGetAutoModeEror(getAutoModeErrFunc);
acsAIR60Passport.AIR60_RegisterCallbackGetAutoModeError(m_deleteGetAuto
ModeError);
private void getAutoModeErrFunc(string ErrMessage)
{
    if (this.InvokeRequired)
    {
        this.BeginInvoke(new Action(() =>
        {
            getAutoModeErrFunc(ErrMessage);
        }));
        return;
    }
    else
```

```
        {
            MessageBox.Show(ErrMessage + " \nPlease put passport again!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
```

3.3.1.9. AIR60_RegisterCallbackGetOtherData

Format:

void AIR60_RegisterCallbackGetOtherData(AIR60DeletgateGetOtherData GetOtherDataFunc)

Function Description: Register the Get Other Data Function.

This function register callback of the GetOtherDataFunc function.

Parameters	Description
GetOtherDataFunc	A function that retrieves the time consumption when reading selected Data Groups and obtains type identifiers of Data Groups that are not selected but are present in the chip, see below.
Return Value *None*	-

Table 22: AIR60_RegisterCallbackGetOtherData Function Description

Note: The function registered with AIR60_RegisterCallbackGetOtherData will be called by the API and it contains the following parameters when being invoked:

- Parameters 1. Read selected DG file numbers and the processing time
- Parameter 2. Unselected but present DG file numbers

3.3.1.9.1. AIR60DeletgateGetOtherData

Format:

public delegate void AIR60DeletgateGetOtherData(Dictionary<DatagroupType, long> tagDictory, List<DatagroupType> remnantDataGroupList)

Function Description: Read Other Data Function.

This function retrieves error message of auto mode.

Parameters	Description
tagDictory[in]	The list of selected Data Groups and its corresponding time consumption.
remnantDataGroupList[in]	List of the unselected but present DG file numbers
Return Value *None*	-

Table 23: AIR60DelegateGetOtherData Function Description

Source Code Example:

```
Dictionary<DatagroupType, long> lastTagDictory;
List<DatagroupType> lastRemnantDataGroupList;
AIR60DeletgateGetOtherData m_deleteGetOtherDate;
m_deleteGetOtherDate = new AIR60DeletgateGetOtherData(getOtherDataFunc);
acsAIR60Passport.AIR60_RegisterCallbackGetOtherData(m_deleteGetOtherDate);
 private void getOtherDataFunc(Dictionary<DatagroupType, long> tagDictory,
List<DatagroupType> remnantDataGroupList)
{
    lastTagDictory = tagDictory;
    lastRemnantDataGroupList = remnantDataGroupList;
}
```

3.3.1.10. AIR60_RegisterGetQRCodeScannerStatus

Format:

Void
AIR60_RegisterGetQRCodeScannerStatus(AIR60DelegateGetQRCodeScannerStatus QRCodeScannerStatusFunc)

Function Description: Register the Get QR Code Scanner Status Function.

This function register callback of the QRCodeScannerStatusFunc function.

Parameters	Description
QRCodeScannerStatusFunc	A function that gets QR Code Scanner status, see below.
Return Value	*None* -

Table 24: AIR60_RegisterCallbackGetOtherData Function Description

3.3.1.10.1. AIR60DelegateGetQRCodeScannerStatus

Format:

public delegate void AIR60DelegateGetQRCodeScannerStatus(bool blQRcodeScannerStatus)

Function Description: Read QR Code Scanner Status.

This function retrieves the status of the QR code scanner.

Parameters	Description
blQRcodeScannerStatus[in]	True: QR Code Scanner is present False: QR Code Scanner is absent
Return Value	*None* -

Table 25: AIR60DelegateGetQRCodeScannerStatus Function Description

Source Code Example:

```
AIR60DelegateGetQRCodeScannerStatus m_deleteGetQRCodeScannerStatus;

m_deleteGetQRCodeScannerStatus=new
IR60DelegateGetQRCodeScannerStatus(getQRCodeScannerStat   usFunc);

    acsAIR60Passport.AIR60_RegisterGetQRCodeScannerStatus(m_deleteGetQRC
odeScannerStatus);

     private void getQRCodeScannerStatusFunc(bool blScannerStatus)

    {

        if (blScannerStatus)

            DisplayLogsMessage("QRCodeScanner is present!");

        else

            DisplayLogsMessage("Failed to find QRCodeSCanner!");

    }
```

3.3.1.11. AIR60_RegisterGetQRCodeScanner

Format:

void AIR60_RegisterGetQRCodeScanner(AIR60DelegateGetQRCodeScannerInfo GetQRCodeScannerInfoFunc)

Function Description: Register the Get QR Code Information Function.

This function register callback of the GetQRCodeScannerInfoFunc function.

Parameters		Description
GetQRCodeScannerInfoFunc		A function that retrieves QR Code information, see below.
Return Value	*None*	-

Table 26: AIR60_RegisterCallbackGetOtherData Function Description

3.3.1.11.1. AIR60DelegateGetQRCodeScannerInfo

Format:

public delegate void AIR60DelegateGetQRCodeScannerInfo(string QRCodeScannerInfo)

Function Description: Read QR Code Information.

This function retrieves the information of the QR code.

Parameters		Description
QRCodeScannerInfo[in]		QR Code Information
Return Value	*None*	-

Table 27: AIR60DelegateGetQRCodeScannerInfo Function Description

Source Code Example:

```
AIR60DelegateGetQRCodeScannerInfo m_deleteGetQRCodeScannerInfo;
m_deleteGetQRCodeScannerInfo = new AIR60DelegateGetQRCodeScannerInfo
(getQRCodeScannerInfoFunc);
acsAIR60Passport.AIR60_RegisterGetQRCodeScanner(m_deleteGetQRCodeScannerInfo);
  private void getQRCodeScannerInfoFunc(string QRCodeScannerInfo)
```

```
{
    if (this.InvokeRequired)
    {
        this.BeginInvoke(new Action(() =>
        {
            getQRCodeScannerInfoFunc(QRCodeScannerInfo);
        }));
        return;
    }
    DisplayBarCodeScannerMessage(QRCodeScannerInfo);
}
```

3.3.1.12. AIR60_ParseMRZ

Format:

Public MRZDataResult AIR60_ParseMRZ(string MRZInfo)

Function Description: Parse the MRZ Information.

This function parses the Passport's MRZ information from OCR. Please call AIR60_ValidateMRZInfo to validate the MRZ information before calling AIR60_ParseMRZ.

Parameters		Description
MRZInfro[in]		**MRZ Information**
Return Value	MRZDataResult	Data Structure of the MRZ information.
	Exception	An AIR60APIException is thrown.

Table 28: AIR60_ParseMRZ Function Description

Source Code Example:

```
string strMRZInfo= string.Empty;
strMRZInfo=....
```

```
...
Try
{
    if (acsAIR60Passport.AIR60_ValidateMRZInfo(strMRZInfo))
    {
   MRZDataResult DG1Data=acsAIR60Passport.AIR60_ParseMRZ(strMRZInfo);
        ...
    }
}
Catch(AIR60APIException  ex)
{
    ...
}
```

4.0. Main Structures

4.1. ICAO Data Interpretation

“DG” stands for Data Group.

All documents that comply with the ICAO9303 standard contain files such as DG1-DG16, COM, and SOD.

The specific meanings of these files are as follows:

DG_COM:Header and data group presence information

DG1: Machine readable zone information

DG2: Encoded identification features - face

DG3: Additional identification features - finger(s)

DG4: Additional identification features - iris(es)

DG5: Displayed portrait

DG6: Reserved for future use

DG7: Displayed signature or usual mark

DG8: Data feature(s) (this data group has yet to be defined)

DG9: Structure feature(s) (this data group has yet to be defined)

DG10: Substance feature(s) (this data group has yet to be defined)

DG11: Additional personal detail(s)

DG12: Additional document detail(s)

DG13: Optional detail(s)

DG14: Security options

DG15: Active authentication public key info

DG16: Person(s) to notify

SOD: Document security object

4.2. PassportReadResult Structure

```
public class PassportReadResult
{
```

```
    public CommonDataResult DG_COM;
    public MRZDataResult DG1;
    public FaceDataResult DG2;
    public FingerDataResult DG3;
    public IrisDataResult DG4;
    public PortraitDataResult DG5;
    public BasicDataResult DG6;
    public BasicDataResult DG7;
    public BasicDataResult DG8;
    public BasicDataResult DG9;
    public BasicDataResult DG10;
    public DG11DataResult DG11;
    public BasicDataResult DG12;
    public BasicDataResult DG13;
    public SecurityDataResult DG14;
        public SecurityDataResult DG15;
        public SecurityDataResult DG16;
        public SODDataResult SOD;
}
```

4.3. ICAOFileInfo Structure

```
public struct ICAOFileInfo
{
   public string DataGroup;//Data Group
   public string EFName;//EF Name
   public string ShortEFIdentifier;//Short EF Identifier
   public string EFIdentifier;//EF IDentifler
   public string Tag;//Tag
}
```

4.4. BasicDataResult Structure

```
public class BasicDataResult
{
    public ICAOFileInfo iCAOFileInfo;
    public bool IsRead ;
    public byte[] Data ;
}
```

4.5. CommonDataResult Structure

```
public class CommonDataResult : BasicDataResult
{
    public string LdsVersionNumber;// Version number
    public string UnicodeVersionNumber;// Unicode version number
    public byte[] PresentDataGroups ;// present Data Groups
}
```

4.6. MRZDataResult Structure

```
public class MRZDataResult : BasicDataResult
{
    public MrzFormat Format;
    public string Type;
    public string CountryCode;
    public string Number;
      public string OptionalData1;
      public string OptionalData2;
    public DateTime? BirthDate;
    public Gender Gender;
    public DateTime? ExpirationDate;
```

```
    public string Nationality;
    public string Surname;
    public string Name;
}
```

4.7. FaceDataResult Structure

```
public class FaceDataResult : BasicDataResult
{
    public int NumberOfInstances;//Number of instances
    public byte[] FaceImageData;//Image data of face
}
```

4.8. FingerDataResult Structure

```
public class FingerDataResult : BasicDataResult
{
    public int NumberOfFingerprints;//Numbers of fingerprint
    public List<byte[]> FingerprintData;//Data of figerprint
}
```

4.9. IrisDataResult Structure

```
public class IrisDataResult : BasicDataResult
{
    public int NumberOfIrisScans;//Number of iris scan
    public List<byte[]> IrisData;// Data of iris
}
```

4.10. PortraitDataResult Structure

```
public class PortraitDataResult : BasicDataResult
```

```
{
    public byte[] PortraitImageData; //Image data of portrait
}
```

4.11. DG11DataResult Structure

```
public class DG11DataResult : BasicDataResult
{
    public string NameOfHolder;
    public List<string> OtherNames;
    public string PersonalNumber;
    public string FullDateOfBirth;
    public List<string> PlaceOfBirth;
    public List<string> PermanentAddress;
    public string Telephone;
    public string Profession;
    public string Title;
    public string PersonalSummary;
    public byte[] ProofOfCitizenship;
    public List<string> OtherValidTDNumbers;
    public string CustodyInformation;
}
```

4.12. DG12DataResult Structure

```
public class DG12DataResult : BasicDataResult
{
public string IssuingAuthority;
public string DateOfIssue;
public List<string> NamesOfOtherPersons;
```

```
public string EndorseMentsAndObservations;
public string TaxOrExitRequirements;
public byte[] ImageOfFront;
public byte[] ImageOfRear;
public string DateAndTimeOfPersonalization;
public string PersonalizationSystemSerialNumber;
}
```

4.13. SecurityDataResult Structure

```
public class SecurityDataResult : BasicDataResult
{
    public Dictionary<string, object> SecurityInfo;
}
```

4.14. SODDataResult Structure

```
public class SODDataResult : BasicDataResult
{
    public Dictionary<string, byte[]> DataGroupHashes;
}
```

4.15. DatagroupType Structure

```
public enum DatagroupType
{
    DataCOM = 96, //DG_COM
    DataGroup1 = 97, //DG1
    DataGroup2 = 117, //DG2
    DataGroup3 = 99, //DG3
    DataGroup4 = 118, //DG4
```

```
    DataGroup5 = 101, //DG5
    DataGroup6 = 102, //DG6
    DataGroup7 = 103, //DG7
    DataGroup8 = 104, //DG8
    DataGroup9 = 105, //DG9
    DataGroup10 = 106, //DG10
    DataGroup11 = 107, //DG11
    DataGroup12 = 108, //DG12
    DataGroup13 = 109,//DG13
    DataGroup14 = 110,//DG14
    DataGroup15 = 111,//DG15
    DataGroup16 = 112,//DG16
    DataSOD = 119 //SOD
}
```