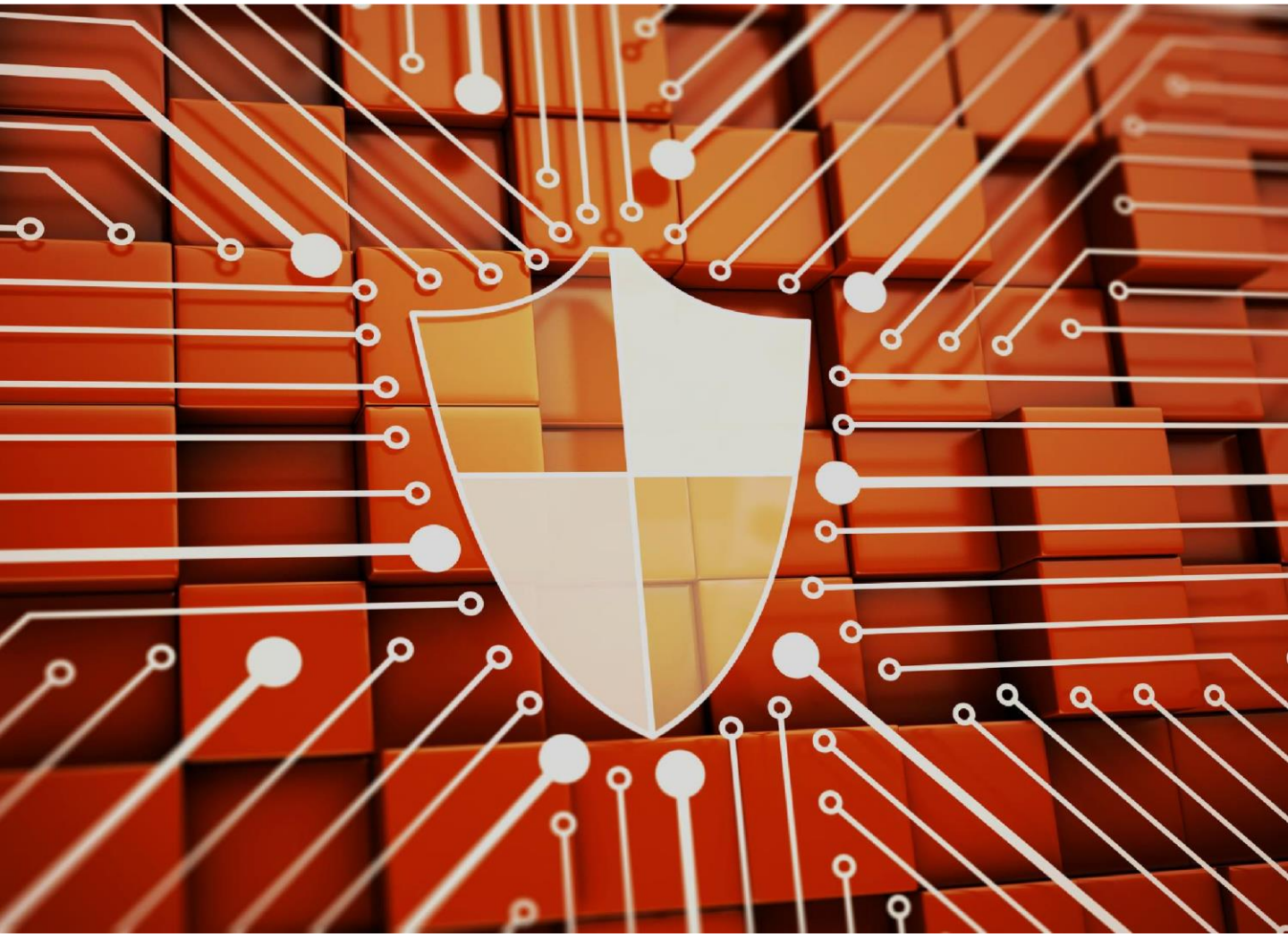
August 2021

Document Version 3.3

IRISHIELD™–USB AND IRISHIELD™–UART SOFTWARE DEVELOPER'S MANUAL



Iritech, Inc.

Document
Change Record

This page records any updates and revisions made to the IriShield™-USB and IriShield™-UART Software Developer's Manual.

Doc ver.	Date	Change Description	SDK ver
3.0	19th Jun 2013	The first release of IDDK 2000 library which supports both IriShield binocular and monocular devices	3.2.6
3.1	09th Jan 2014	Update power management and usbfs	3.3.0
3.2	14th May 2014	Minor update	3.3.0
3.3	23rd Jan 2015	Support Mac OS X	3.3.1

Note

The information contained herein is provided solely for the purpose of assisting customers to understand how to operate and integrate IriTech's hardware and software and is not to be released, reproduced, or used for any other purpose without written permission of IriTech, Inc.

IriTech reserves the right to make changes on this hardware and software with the intent to improve its functionalities. Information and specifications contained in this document are subject to change without prior notice and do not represent a commitment on part of IriTech.

Table of Contents

1. Introduction

1.1 Manual Overview

The purpose of this manual is to provide useful information on IriTech's family of embedded camera systems. It includes an overview of the software development kit, its functionality, component terminology, and useful tips on how to begin programming your own application. For details of application programming interfaces, please see the corresponding API Reference Manual. Information about hardware specifications can be found in the Hardware Developer's Manuals for a specific device.

1.2 IriShield Overview

IriShield is an iris biometrics device that has its own computation power, memory, storage, and peripherals to perform iris image acquisition, template generation, and matching. Furthermore, it includes an embedded security infrastructure that utilizes the most popular and strongest cryptography algorithms to enable data security, integrity, and authenticity during transmission.

1.2.1 Irishield Models

IriShield device comes in three models with three different communication protocols including USB, UART and IP. IriShield™-USB, where both data communication and power supply are done via a single USB cable, is supposed to work with a host PC with any OS. IriShield™-UART is intended to work with embedded systems with/without any OS. For IriShield™-IP, no host device is needed; data will be sent directly from the camera to a server using IP protocol (Ethernet interface). As a matter of fact, since UART affords a much slower speed than USB does, IriShield™-UART is not recommended for applications where smooth streaming is critical. However, it would be an excellent option when the embedded host system does not support a USB port or is affected in a hazardous way by supporting a USB port, or when displaying streaming iris images is not required during capturing.

Noticeably, IriShield employs IriTech's top-performing iris image quality assessment and iris recognition algorithms, which were proven in extensive NIST's IREX tests. It supports a new compact iris encoding scheme (iris N-template) and efficient matching algorithms, which not only optimize template management but also produce very high matching accuracy. This makes IriShield™ devices completely compatible with IriTech's enterprise solutions, such as IriCore or IriMaster.

1.2.2 Device Features

Every IriShield™ device has embedded automated iris capture and security infrastructure functionality by default. Other features—such as template comparison and template generation—are dynamically integrated into the device depending on each customer's requirement. For example, a customer may have a server to do the template matching, so template comparison is not desired on the device. IriShield can be customized to include template generation only in addition to the featured defaults.

1.2.3 Onboard Iris Image Acquisition

The IriShield device contains a compact iris camera that consists of one (monocular device) or two image sensors (binocular device) for image acquisition. During operation, an independent infrared LED flexibly illuminates the details inside the iris. Images from the sensors are streamed through a Quality Measurement module (QM) inside the device for real-time evaluation to filter out non-eye or unqualified eye images. Many quality factors are taken into account in QM to make sure that the selected qualified images contain detailed iris textures with sufficient sharpness. This quality analyzing mechanism is so effective and fast that the best qualified iris image is detected and captured immediately right after a person's eye enters the focus range of the camera. The whole process above is called the capturing process, and can be customized with different operation modes, capturing modes, and quality alternatives.

Every IriShield device supports an automated capturing process in which the device controls its peripherals to get images, evaluate every image, and make every qualifying decision by itself without any external support. Users, however, should know how to operate the device to make this automation produce desirable quality images.

Capturing process in IriShield can be customized as a callback function which is invoked only when there is a new frame or new capturing status or errors so that developers can easily handle what to do next.

1.2.4 On-Board Iris Image Quality Assessment

IriShield is equipped with IriTech's winning iris image quality assessment algorithms (NIST IREX II) to aggressively evaluate the captured image at the end of each capture to produce its total quality score and occlusion information. These quality metrics make it easier for developers to set different quality standards for enrollment and matching.

1.2.5 On-Board Iris Recognition

1.2.5.1 Template Generation

The captured image is a VGA image which is 640x480 pixels and includes non-iris parts. Due to the large size of raw images, subsequent image processing, transmission, and storage can be burdensome. IriTech addresses this matter by utilizing much smaller data structures to store iris features, called templates.

An iris template is a specialized binary data structure that contains the valuable features extracted from the raw eye image. These features are compressed and encoded into a compact size which accelerates the matching process and makes it efficient in transmission, storage, and processing. Template is the very informative material to iris recognition and helps to produce highly accurate results during matching process. All of iris recognition functionalities of the device are performed based on templates.

IriShield has employed a new uniform template structure, called N-template for both storing and matching. Each N-template is compact and able to encode multiple irises (recommended from the same person) from different eyes. It eliminates the trouble of managing different kinds of templates of different eyes from different persons at a customer repository, and facilitates

template exchange in biometric networks. Moreover, N-template is generated by the most up-to-date algorithms which make it more informative in producing higher matching accuracy.

1.2.5.2 Iris Recognition

The IriShield device has sufficient memory to afford an in-memory template gallery of thousands of iris slots to serve for 1:1 and 1:N matching. Moreover, the non-volatile flash memory assures that this database maintains persistent even after the device is powered off. The gallery is structured as a simple database where customers can easily manage many templates by their own IDs. Each ID can have at least one iris (one iris slot) and at most 8 irises (8 iris slots) of both left and right eyes.

Matching is executed using templates instead of the original images. Matching mainly occurs between the template(s) from the captured image(s) and those from the gallery. IriShield device also supports the comparison between the template(s) form the captured image(s) and user-input template(s).

1.2.6 Iritech's Onboard Security Infrastructure

To support users with secure communication between IriShield and a host/server, IriTech provides a Security Infrastructure within the device. This Public Key Infrastructure (PKI)-based security infrastructure takes advantage of the prominence of both symmetric and asymmetric cryptography algorithms. In particular, all functions related to image, template, and sensitive information transmission between the host and the device through USB bus are provided with state-of-the-art cryptographic standards employing very high security level 256-bit AES and 2048-bit RSA.

1.2.6.1 PKI Key Distribution

In IriTech's Security Infrastructure, the user possesses his/her own 2048-bit RSA private/public key pair to be used throughout the system. The private key must be kept confidential at his/her own site (e.g., at a central server) while its paired public key is freely distributed to every device in the most common form of a public key certificate - an X509 certificate.

To provide more robust security ability for IriShield, IriTech proposed a PKI-based Security Infrastructure. Every IriShield can generate a unique 2048-bit public/private key pair by itself on customer's demand and exports only public key to outside. A private key is generated on demand and never leaves the device; therefore, it significantly decreases the chance of exposing the key to intruders. Optionally, users can choose to create the key pair by themselves and import them into the device in the form of a PKCS

1.2.6.2 End-To-End Security

The PKI-based Security Infrastructure provides authentication and ensures integrity in the communication between user and device. Images captured by a device are digitally signed with the device's private key, and the user can verify this signature using the corresponding public key to assure that the received data is from the trusted device and is the original. Thereby, message integrity and user authentication are established during the transmission.

Security in transmission is also taken care of by the device. At the beginning of each transmission, an AES-256 key (called session key) is generated randomly to encrypt the signed image symmetrically. Then, the user's public key is employed to protect the session key that has just been used to encrypt the data. Finally, the encrypted images are transmitted along with the encrypted symmetric keys. Therefore, it prevents anyone except the intended recipient from reading the message.

An impressive feature of IriShield is being secured right from the manufacturing process, where each device will be provided with a random secret AES 256-bit key, which is stored very securely. Thanks to the proprietary secure boot functionality of the Texas Instruments chip. This key will be used to encrypt all data, such as PKI key pairs, template data and other sensitive information, before they are stored onto the NAND memory of the device to make sure they cannot be used for any unauthorized or malicious purpose.

1.2.6.3 Administrative Audits

The IriShield device has the capability of securely recording information on the capturing operation, such as timestamp, capturing station, capturing operator ID, and other user-specific data. With this audit information, the usage and performance of each capturing station and each operator therein can be easily monitored and traced. Especially with this customizable data, customers can embed their OTP, PIN, or other factors into iris data to have a single secured multi-factor authentication packet.

1.2.7 System Configuration and Roles

IriShield is designed to allow customers to customize it as a friendly and easy-to-use device or a stringent self-protecting system in which critical functionalities can only be accessed with appropriate privileges.

1.2.7.1 System Configuration

The following configuration parameters can be used to customize IriShield.

- **Encryption Mode**: By default, a host can retrieve captured images and templates from the device both in plain BLOB (Binary Large Object) and in encrypted signed BLOB. However, IriShield can be configured to allow only encrypted signed BLOB.

- **Deduplication**: By default, the device does not check duplication before enrollment. This configuration requests the device to perform basic duplication check which reports error when the query template matches with at least one other from the enrolled gallery. The threshold for deduplication is configurable.

- **Supervised Enrollment**: By default, any user of the device can perform enrollment and unenrollment. With this Supervised Enrollment configuration, only Superusers are allowed to perform those critical tasks.

- **Iris Data Closure**: When enabled, this configuration prevents captured images and templates from being exported to outside.

- **Streaming max scale**: This configuration prevents original iris data exposure through stream images.

- **Baud-rate**: By default, IriShield™-UART communicates at the speed of 115200 kbps. Users can change this baud rate to match their speed requirement. The device needs to be reset before the new baud rate goes into effect.

- **Flow Control**: By default, IriShield™-UART communicates without flow control. Users can enable flow control anytime. However, it needs to be reset before the new mode goes into effect.

- **Image format and compression quality**: Image format and compression quality of streaming image can be configurable to match their speed requirement via UART communication.

- **Power management**: Device can put itself into a specific power-saving mode when IDDK_SleepDevice function was called. Or device automatically can be in a power-saving mode after some specific amount of time without receiving any command from host. Or it can be configured to automatically enter DEEPSLEEP when detecting a falling edge on DS_input pin.

1.2.7.2 System Roles

To prevent unauthorized accesses and configuration, the device can be customized to allow only the authenticated users to perform their granted tasks. There are three groups of users: Administrator, Superuser, and User. Privileges for each group are listed in the following table.

Privilege	Administrator	Superuser	User
Capture	✓	✓	✓
Change device configuration	✓	×	×
Lock/Unlock device	✓	×	×
Reset RSA keys	✓	×	×
Identify and Verify	✓	✓	✓
Login and Logout	✓	✓	×
Enroll and Unenroll	×	✓	×

Administrators manage the device, but they may not be the end-users. For example, IriShield devices have been purchased for door access in a building. Before the devices actually do their job, Administrators will be in charge of configuring them to prevent unauthorized accesses. Administrators can lock and unlock device or reset cryptography keys, but they may not be allowed to access the doors where those devices are installed.

Superuser and User are the actual end-users who actually undergo iris recognition to open the doors. The only difference between Superuser and User is that the former can enroll and unenroll other users (except Administrator) whereas the latter cannot.

Therefore, in the device, Administrators' enrollment data are maintained in an internal database separated from the main gallery of actual users. If an Administrator is also an end-user, he/she needs to be enrolled twice.

When manufactured, the device does not have any Administrator or Superuser, and all functionalities are open. However, as soon as the first Administrator or Superuser is enrolled, accesses to corresponding critical functions are limited to only authenticated users.

A device can have maximum of five Administrators (each can be enrolled with eight irises at most) and as many Superusers as the gallery can afford. An Administrator can enroll/unenroll other Administrators. A Superuser can change the role of other Superusers or Users.

Please be warned that Administrator and Superuser are authenticated by their own irises. If Administrators or Superusers are enrolled, it is recommended to have more than one enrollee for each group, because there will be no way to regain full access to the device in case the only Administrator or Superuser cannot be authenticated due to reasons such as losing his/her irises or death.

1.2.8 Power Management

There have been two power schemes released along with IriShield's firmware: First-released Power Scheme and Second-released Power Scheme.

1.2.8.1 First-released Power Scheme

This power scheme only supports USB-enabled device and includes two different power-saving modes: STANDBY and SLEEP.

Power Modes	Description	Wakeup By	Cautions
STANDBY	Device's main processor enter idle mode. Its peripherals (flash, uart, usb, GPIO, etc.) are still full clocked and powered. Device goes into STANDBY when + Iddk_SleepDevice is called	Any activity on USB/UART communication link (except for KEEP ALIVE frames from USB).	
SLEEP	Operating clock frequency is lowered down. Peripherals and memory enter minimum power consumption states. Main processors enter idle mode. Device goes into SLEEP when + Iddk_SleepDevice is called	Re-enumeration on USB port	Device needs some amount of time (several hundred miliseconds) to put itself into SLEEP. During this time, it puts its peripherals including USB and itself into low power state. If there is an incomming command during its sleeping procedure, communication is corrupted and host may receive unexpected data or error.

1.2.8.2 Second-released Power Scheme

IriShield device offers different power-saving schemes including STANDBY, SLEEP and DEEPSLEEP during device's idling period. Not only is the power consumption of SLEEP and DEEPSLEEP different from the previous scheme but also the ways to wake device up from those saving modes are also different. STANDBY stays the same for both schemes.

Power modes	Description	Wakeup by	Cautions
STANDBY	Device's main processor enter idle mode. Its peripherals (flash, uart, usb, GPIO, etc.) are still full clocked and powered. Device goes into STANDBY when + Iddk_SleepDevice is called + Device is configured to automatically go into STANDBY after some idling time.	Any activity on USB/UART communicati on link (except for KEEP ALIVE frames from USB).	
SLEEP	Operating clock frequency is lowered down. Peripherals and memory enter minimum power consumption states. Main processors enter idle mode. Device goes into SLEEP when + Iddk_SleepDevice is called + Device is configured to automatically go into SLEEP after some idling time. + There is no activity on USB bus for more than 3 milliseconds.	Access to any API in the correspondin g SDK (Please refer to the API reference manual for more detail).	Device needs some amount of time (several hundred miliseconds) to put itself into SLEEP. During this time, it puts its peripherals including USB and UART as well as itself into low power state. If there is an incomming command during its sleeping procedure, communication is corrupted and host may receive unexpected data or error.
DEEPSLEEP	Device's peripherals and memory are either powered off or put into deep sleep state. All communication components including USB and UART are powered off. Device goes into DEEPSLEEP when + Iddk_SleepDevice is called + Device is configured to automatically go into DEEPSLEEP after some idling time. + Device is configured to automatically go into DEEPSLEEP when a falling edge is detected on DS_input PIN.	An external interrupt generated by a level shift from logical low to logical high of DS_input pin (a rising edge).	Device is totally in deep sleep mode only when DS_input pin is logical low. When an USB-enabled device is in deep sleep, it is detached from host system as if it was unplugged When a USB-enabled device is fully waken up, it needs reenumerating by USB host.

1.2.9 Compatibility Between Irishield Monocular and Binocular Devices

IriTech devices are designed to operate compatibly with one another in a way that templates generated from a monocular device are capable to be used in a binocular device, and vice versa.

The monocular device captures image from a single camera sensor. Therefore, a generated template contains no eye-subtype information. In other words, it is an unknown eye template. On the contrary, the binocular device is equipped with two camera sensors which capture images from the left eye and the right eye virtually simultaneously. Therefore, the generated template contains the eye-subtype information.

However, the adopted N-template can contain many irises of any eye-side and matching between those N-templates is also supported. Each iris in one N-template will be respectively compared with all irises in the other to produce an array of "sub-distances." If an iris from the left eye is compared with an iris from the right eye, a high distance of 3.0 will be returned. A distance of 4.0 implies a matching error. Finally, the result distance between the two N-templates will be the smallest value among those "subdistances

1.2.10 Supported Development Platform

IriShield currently supports USB drivers and development libraries (IDDK) for the following operating systems.

- Windows XP and Windows 7 32-bit/64-bit
- Windows CE 6.0
- Linux 32-bit/64-bit
- Mac OS X 64-bit
- Embedded Linux: kernel version 2.6.21
- Android

NOTE:

- Current model of IriShield only supports USB and UART communication.
- Different from those for non-embedded environment, driver binaries and device SDKs for embedded systems (e.g., Windows CE and Embedded Linux) are built based on the CPU model that the host system is using. To obtain the correct binaries for your embedded system, please contact IriTech, Inc., and give us more specific information about your CPU model. We may need your specific BSP, kernel, or tool chain to have the proper binaries produced for you.

1.2.11 Host With No OS or Unsupported OS

It is easy to integrate IriShield into existing biometric systems although they may not have an operating system or have operating systems not listed in 1.2.10. Developers should contact IriTech, Inc. to obtain IriShield Packet Protocol Manual, which details the ways in how to send/receive commands and data to/from IriShield and how to process exchanged data at communication level.

1.2.12 Special Note for Android Platforms

IriTech provides two development alternatives for Android platform:

- Android on native code: Driver and SDK are built on native code (written in C/C++). To install driver, developer needs to have root privilege of the Android host target. Native SDK easily interacts with managed code (written in Java) by Java Native Interface (JNI). Device manufacturer can use these native driver and SDK to enable IriShield directly in the host firmware. It can be built on almost all kernel and Android versions. Developers need to contact with IriTech, Inc. to provide their kernel version, configuration and tool-chain to get the compatible driver and SDK.
- Android USB Host Mode: Driver and SDK are developed on managed code (written in Java) based on Android USB Host APIs in android.hardware.usb. They are easily installed and require no root privilege. However, the Android-powered device has to enable USB host mode and Android version has to be 3.1 and higher. This alternative is good for commercial Android devices on which consumers have no detail knowledge about the kernel and root privilege.

1.2.13 Endianness

Byte order of host system significantly affects the communication with the IriShield device. The IDDK 2000 library currently supports system with little endian byte order only. We are testing and going to support big endian byte order soon.

1.3 Term and Abbreviations

This section lists and defines terms and abbreviations used throughout this document.

Advanced Encryption Standard	A symmetric encryption standard adopted by the U.S. government. It is a block cipher design that works on a block size of 128 bits and a key size of 128, 192, or 256 bits.
AES	See Advanced Encryption Standard.
Binary Large Object	A collection of binary data.
BLOB	See Binary Large Object. In this document context, BLOBs are used to store data structures that are serialized into a sequence of bytes.
Block	In symmetric cryptography, block is a fixed-length group of bits.
Block cipher	A symmetric key cipher operating on fixed-length blocks. A block cipher algorithm encrypts a block of plaintext into a block of cipher text of the same size.
Byte	A data element that is eight bits in size.
CA	See Certificate Authority.
Capturing process	A process occurs inside device to capture qualified iris images.

CBC	See Cipher-Block Chaining.
Certificate Authority	A trusted third party in PKI that issues digital certificates for use by other parties.
Cipher-Block Chaining	A block cipher mode of operation. The message is divided into blocks of fixed-length. Each block of plaintext is XORed with the previous cipher text block before being encrypted. The first block is XORed with a random Initialization Vector.
CRCam	Device's private key. This key is used by a device to decrypt incoming encrypted data and sign outgoing data.
CRCust	Customer's private key. This key is used by a customer to decrypt encrypted data coming from device and to sign data that is going to device.
Cryptography module	A software component in device that is in charge of signing and encrypting outgoing templates and images.
CUCam	Device's public key. This key is used by a customer to encrypt data that is going to be sent to device, and verify signed data that comes from device.
CUCust	Customer's public key. This key is used in a device to encrypt outgoing data and verify signed incoming data.
DER	See Distinguished Encoding Rules.
Matching distance	A value that represents the difference between two templates. The smaller is the distance, the greater is the possibility that those templates belong to the same person.
Distinguished Encoding Rules	A standard message transfer syntax. It is intended for situations when a unique encoding is needed, such as in cryptography, and ensures that a data structure produces a unique serialized representation.
Enumeration	A procedure to detect and identify an IriShield device.
Gallery	See template gallery.
Host	The computer system where an IriShield is installed. This includes the hardware platform (CPU, bus, etc.), operating system and IDDK 2000 in use.
Identification	The procedure of matching a template against a gallery to find out to which enrolled person's ID the query template belongs.
Initialization Vector	A block that is required to allow a block cipher to be executed in several modes of operation, such as CBC. It must be known by the recipient of the encrypted data to be able to decrypt that data. Its size depends on each encryption algorithm, e.g., 128 bits in AES.

IriTech image	A standard image format of IriTech's proprietary products, and is a Binary data representing a qualified iris image of which is produced by a capturing process. In addition to the raw image data, IriTech image contains other information that is understandable only to IriTech devices (including IriShield Mono/Bino, IriCAMMES Bino/Mono, IriMagic, and IriHerald Device).
Iris camera	A camera specially designed to capture iris images. It has infrared LEDs to provide illumination to highlight the texture inside irises. Each IriShield device consists of one (monocular device) or two image sensors (binocular device).
Iris Recognition module	A software component in device that is in charge of generating templates from iris images and matching templates. It also manages a template gallery.
Library	A collection of data structures, types, subroutines and resources that are provided by IDDK 2000.
PFX	A predecessor to PKCS#12.
PKCS#12	Personal Information Exchange Syntax Standard that defines a format commonly used to store private keys with accompanying public key certificates, protected with a password-based symmetric key.
PKI	Public Key Infrastructure.
Quality Measurement module	A software component in device that is in charge of evaluating images stream from iris camera to select the best qualified iris images.
RSA	An algorithm for public-key cryptography. It is suitable for signing as well as encryption.
SHA1	A cryptographic hash function.
Template	A specialized binary data structure that encodes compressed iris features from one or multiple iris images. These features are the very material to iris recognition. Templates are only created and processed by Iris Recognition module.
Template gallery	An in-memory database in device that is used to store enrolled templates.
Verification	The procedure of comparing a query template with a specific template to find out whether these two come from the same person.
X509 Certificate	A standard format for public key exchange in PKI.
N-template	An IriTech's proprietary template.

2. Software Installation

The following software packages must be installed properly to permit interfacing between the hardware and the host.

- **IriShield Device Driver Package:** Each IriTech device has its own driver package depending on the host Operating System. The driver must be installed correctly so the host system can recognize the device and setup data communication with it. With an improperly installed driver, IDDK 2000 reports communication errors such as device not being found or device's I/O failure, or behaves unexpectedly during operation.
- **IDDK 2000 for Developers:** This package contains necessary developing resources including the library and demonstration codes to help developers program and control IriShield, both binocular and monocular device in their customized applications.

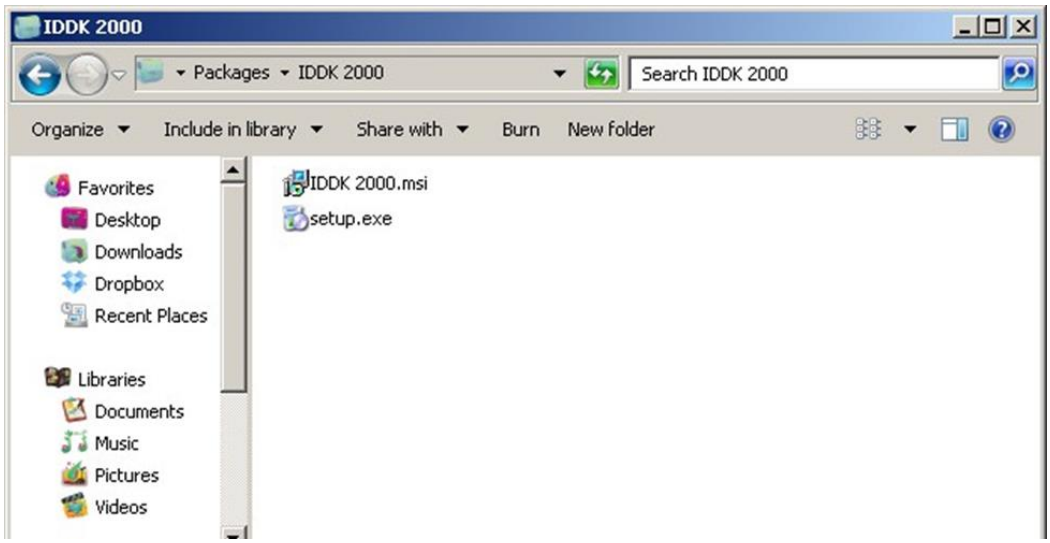
2.1 IDDK 2000 Installtion

- *Note: The SDK version and name in the example screenshots may differ from the users'. The SDK version is also subject to change without notice.*

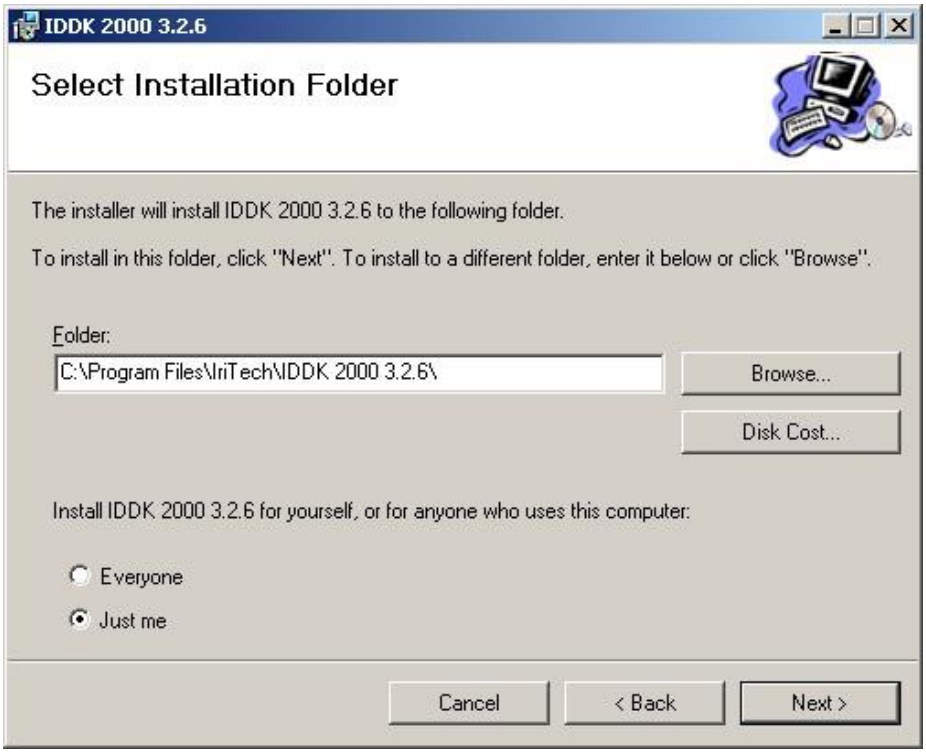
2.1.1 C/C++

2.1.1.1 MS Windows XP and MS Windows 7

1. Download or copy the software package go to the "IDDK 2000" folder. Double click on the "setup.exe" file.

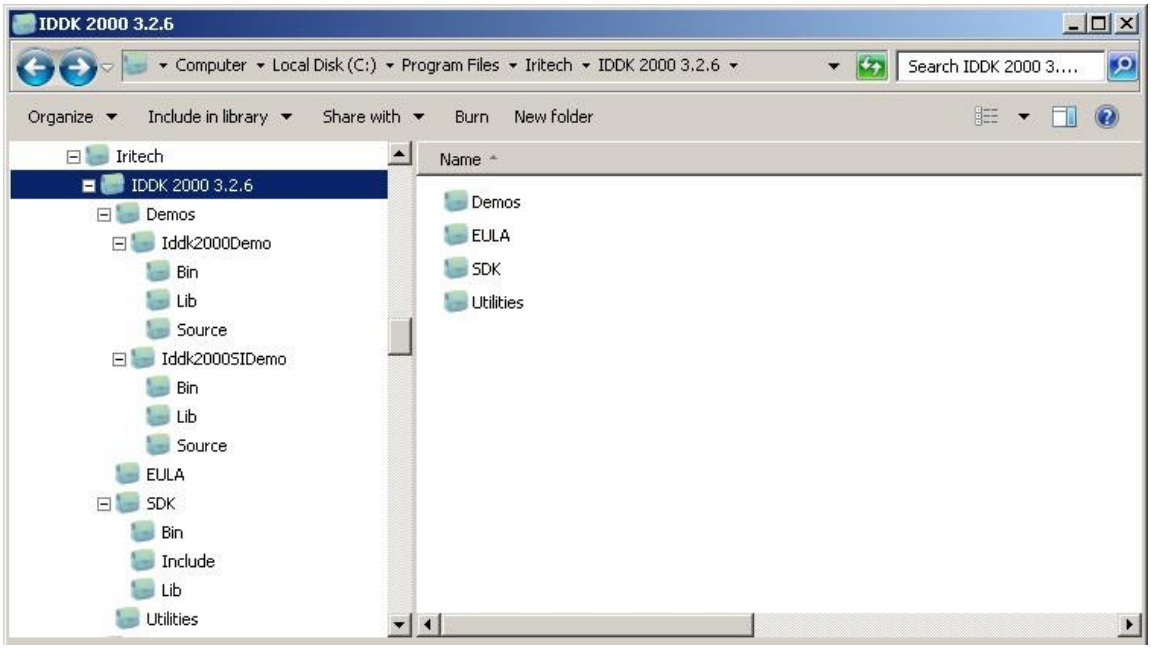


   The installer will guide you through all the installation process. Select the appropriate options and click "**Next**" to move to the next steps.
2. If you want to install the SDK in a different folder instead of the one set as default, click "**Browse**" and specify the desired folder. If not, click "**Next**" to continue.



3. Follow the instructions from the installer to finish the setup process. When the following wizard appears, click "**Close**" to successfully complete the IDDK 2000 Mono installation.

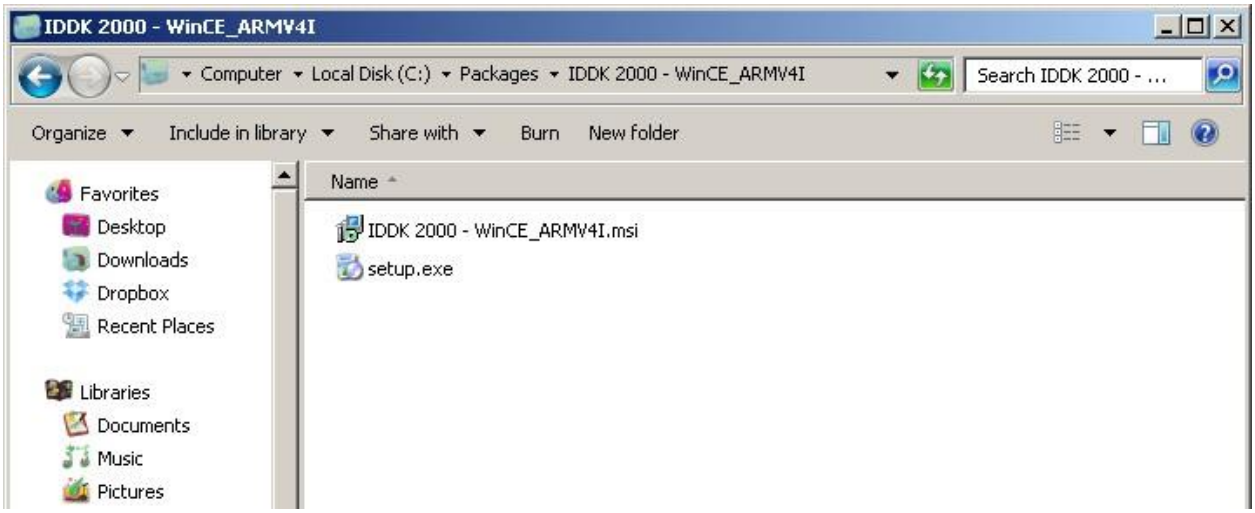
4. When finished, you can verify if the program has been properly installed by checking the installation folder.



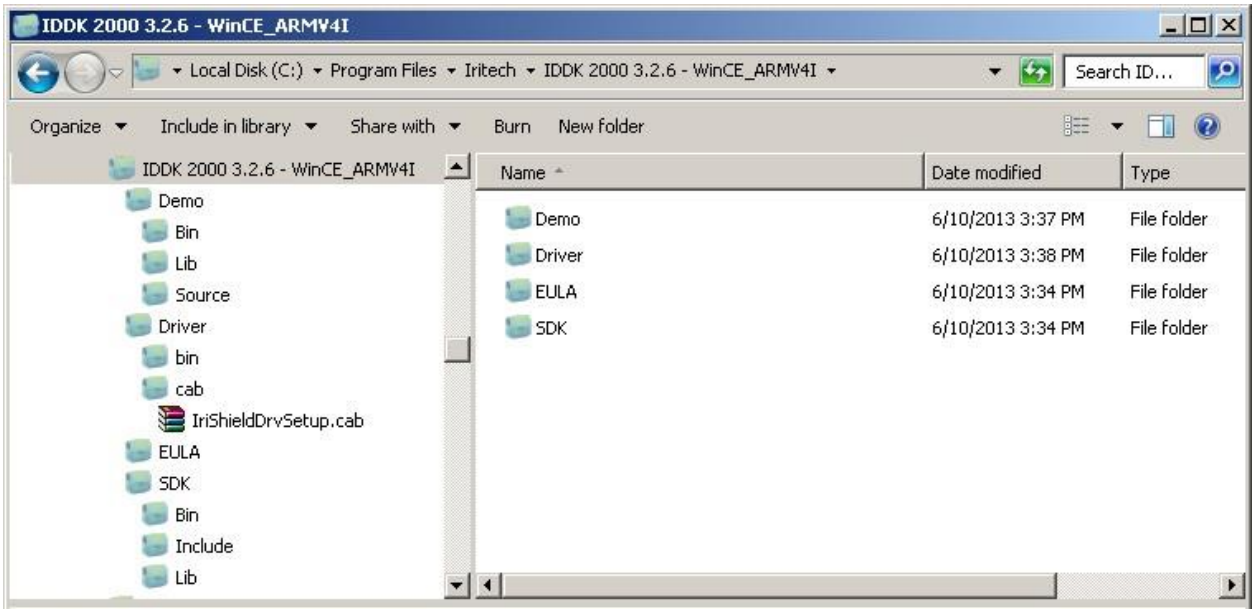
2.1.1.2 MS Windows CE

The IDDK 2000 installation for Windows CE will be conducted on the development PC in similar ways as in installation for Windows XP/Windows 7 (see 2.1.1.1). Please refer to the following steps:

1. Insert and open the software CD and go to the "**IDDK 2000 – WinCE_ARMV4I**" folder. Double click on the "**setup.exe**" file.



   The installer will guide you through all the installation process. Follow the instructions as in 2.1.1.1 to complete the setup process successfully.
2. When finished, you can verify if the program has been properly installed on the development PC computer by checking the installation folder.



The SDK package consists of the following folders essential for developing IriShield applications on Windows CE:

- **Driver**: Includes both cabinet installation package (*IriShieldDrvSetup.cab*) for end-users and binary driver file (*irishielddrv.dll*) for developers.
- **SDK**: Includes headers, library and dynamic-link files necessary for developing IriShield application in CE platforms.
- **Demos**: Includes a console demonstration project as a simple guidance on how to use the library. Due to license constraint of cryptography algorithms, sample codes for IriTech security infrastructure are not provided in embedded environment. Developers can refer to these sample codes in the SDK package for desktop.

NOTE: *Although IDDK 2000 for Windows CE installed in a development computer which is a desktop PC, it works in Windows CE environment only. Any attempt to execute the above Driver, SDK and Samples in desktop environment will cause failure with platform incompatibility exception.*

2.1.1.3 Linux

The following demonstration for IDDK 2000 installation is performed on Ubuntu 12.04 as an example. The setup process on other Linux distributions can be executed exactly in the same way.

1. Open a Terminal. **Change to root privilege.**
2. Go to the directory that contains the installation package (e.g.,*/home/iritech/Downloads/IDDK2000*):

```
iritech@Mozart-computer:~$ cd ~/Downloads/IDDK2000
iritech@Mozart-computer:~/Downloads/IDDK2000$ ls
IDDK-2000-3.2.6.tar.gz  install.sh  uninstall.sh
iritech@Mozart-computer:~/Downloads/IDDK2000$
```


The directory includes 3 items:

- install.sh
- uninstall.sh
- IDDK-2000 -<version>.tar.gz

3. To install IDDK 2000, execute "**install.sh**":

```
iritech@Mozart-computer:~/Downloads/IDDK2000$ sudo su
[sudo] password for iritech:
root@Mozart-computer:/home/iritech/Downloads/IDDK2000# chmod 777 install.sh
root@Mozart-computer:/home/iritech/Downloads/IDDK2000# ./install.sh
```

4. Wait for the installation to complete.

```
root@Mozart-computer:/home/iritech/Downloads/IDDK2000# ./install.sh
################ Extracting ################
Iddk/demo/
Iddk/demo/source/
Iddk/demo/source/Iddk2000_utils.h
Iddk/demo/source/Makefile
Iddk/demo/source/Iddk2000Defs.h
Iddk/demo/source/Iddk2000Apis.h
Iddk/demo/source/Iddk2000_features.cpp
Iddk/demo/source/Iddk2000_main.cpp
Iddk/demo/source/Iddk2000_features.h
Iddk/demo/source/Iddk2000_main.h
Iddk/drivers/
Iddk/drivers/icamm7d0/
Iddk/drivers/icamm7d0/install.sh
Iddk/drivers/icamm7d0/iritech.run.64
Iddk/drivers/icamm7d0/root.sh
Iddk/drivers/icamm7d0/built-in/
Iddk/drivers/icamm7d0/installdriver.sh
Iddk/drivers/icamm7d0/iritech.db
Iddk/drivers/icamm7d0/readme
Iddk/drivers/icamm7d0/iritech.rules
Iddk/drivers/icamm7d0/iritech.run
Iddk/drivers/icamm7d0/driver.conf
Iddk/sdk/
Iddk/sdk/Iddk2000Defs.h
Iddk/sdk/Iddk2000Apis.h
Iddk/sdk/libIddk2000.so.3.2.6
################ Checking ################
Searching old version ...
/opt
/opt/IriTech
/opt/IriTech/IDDK-2000-3.2.6
################ Installing ################
Install path: /opt/IriTech/IDDK-2000-3.2.6
Moving files from temporary directory
Making symbolic links ...
Deleting all temporary files
IDDK 2000  SDK 3.2.6 was installed successfully
Press ENTER to exit
```

5. Press any key to successfully complete the IDDK 2000 installation.
6. When finished, you can verify if the program has been properly installed by checking the installation folder.

```
root@Mozart-computer:/home/iritech/Downloads/IDDK2000#
root@Mozart-computer:/home/iritech/Downloads/IDDK2000# cd /opt/IriTech/IDDK-2000-3.2.6/
root@Mozart-computer:/opt/IriTech/IDDK-2000-3.2.6# ls
demo  drivers  sdk
root@Mozart-computer:/opt/IriTech/IDDK-2000-3.2.6#
```

2.1.1.4 MAC OS X

The following demonstration for IDDK 2000 installation is performed on OS X Yosemite 10.10.1 as an example. The setup process on other OS X version can be executed exactly in the same way.

1. Extract installation package from compress file
2. Go to the directory that contains the installation package (e.g.,/Users/IriTech/Downloads/IDDK2000_MACOS)

```
Hieps-Mac:IDDK2000_MACOS iritech$ ls
IDDK-2000-3.3.0-OSX.tar.gz      uninstall.sh
install.sh
```

   The directory includes 3 items:

   - install.sh
   - uninstall.sh
   - IDDK-2000 -<version>.tar.gz

3. To install IDDK 2000, execute "**install.sh**":

```
Hieps-Mac:IDDK2000_MACOS iritech$ ./install.sh
```

4. Wait for the installation to complete

```
################ Extracting ################
x Iddk/
x Iddk/demo/
x Iddk/sdk/
x Iddk/sdk/Iddk2000Apis.h
x Iddk/sdk/Iddk2000Defs.h
x Iddk/sdk/IddkBase.h
x Iddk/sdk/libIddk2000.dylib.3.3.0
x Iddk/demo/source/
x Iddk/demo/source/Iddk2000_features.cpp
x Iddk/demo/source/Iddk2000_features.h
x Iddk/demo/source/Iddk2000_main.cpp
x Iddk/demo/source/Iddk2000_main.h
x Iddk/demo/source/Iddk2000_utils.h
x Iddk/demo/source/Iddk2000Apis.h
x Iddk/demo/source/Iddk2000Defs.h
x Iddk/demo/source/IddkBase.h
x Iddk/demo/source/Makefile
################ Checking ################
Searching old version ...
/opt
/opt/IriTech
/opt/IriTech/IDDK-2000-3.3.0-OSX
################ Installing ################
Install path: /opt/IriTech/IDDK-2000-3.3.0-OSX
Moving files from temporary directory
Making symbolic links ...
Deleting all temporary files
IDDK 2000  SDK 3.3.0 was installed successfully
Press ENTER to exit
```

5. Press any key to successfully complete the IDDK 2000 installation.
6. When finished, you can verify if the program has been properly installed by checking the installation folder.

```
Hieps-Mac:IDDK-2000-3.3.0-OSX iritech$ ls
demo    sdk
```

2.1.1.5 Embedded Linux

The IDDK 2000 installation for Embedded Linux will be conducted on the development PC computer and is similar to the installation for Linux (see 2.1.1.3). Please refer to the following steps:

1. Open a Terminal. Go to the directory that contains the package (e.g.,*/home/iritech/Downloads/IDDK2000*):

```
iritech@iritech-desktop:~/Downloads/IDDK2000$ ls
Iddk2000_3.3.0_ERelease_IriTech.tar.gz
iritech@iritech-desktop:~/Downloads/IDDK2000$
```

   - The directory includes 1 items: IDDK-2000-<version>-ERelease_<Partner's name>.tar.gz for target devices enabling usbfs
   - Or IDDK-2000-<version>-ERelease_<Partner's name>_IriShieldDriver.tar.gz for target devices using IriTech Driver

2. Extract:

```
iritech@iritech-desktop:~/Downloads/IDDK2000$ tar -xf Iddk2000_3.3.0_ERelease_IriTech.tar.gz
```

3. When finished, you can verify if the program has been properly extracted: Using usbfs

```
iritech@iritech-desktop:~/Downloads/IDDK2000$ cd Iddk2000_3.3.0_ERelease_IriTech/
iritech@iritech-desktop:~/Downloads/IDDK2000/Iddk2000_3.3.0_ERelease_IriTech$ ls
demo  sdk
```

   - Using IriTech Driver

```
iritech@iritech-desktop:~/Downloads/IDDK2000/Iddk2000_3.3.0_ERelease_IriTech_IriShieldDriver$ ls
demo  drivers  sdk
```

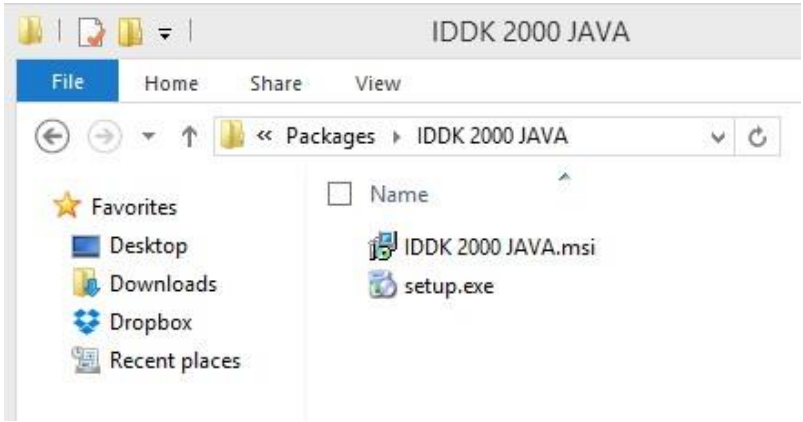
2.1.1.6 Other Operating Systems and Non-OS Platforms

See 1.2.11

2.1.2 Java

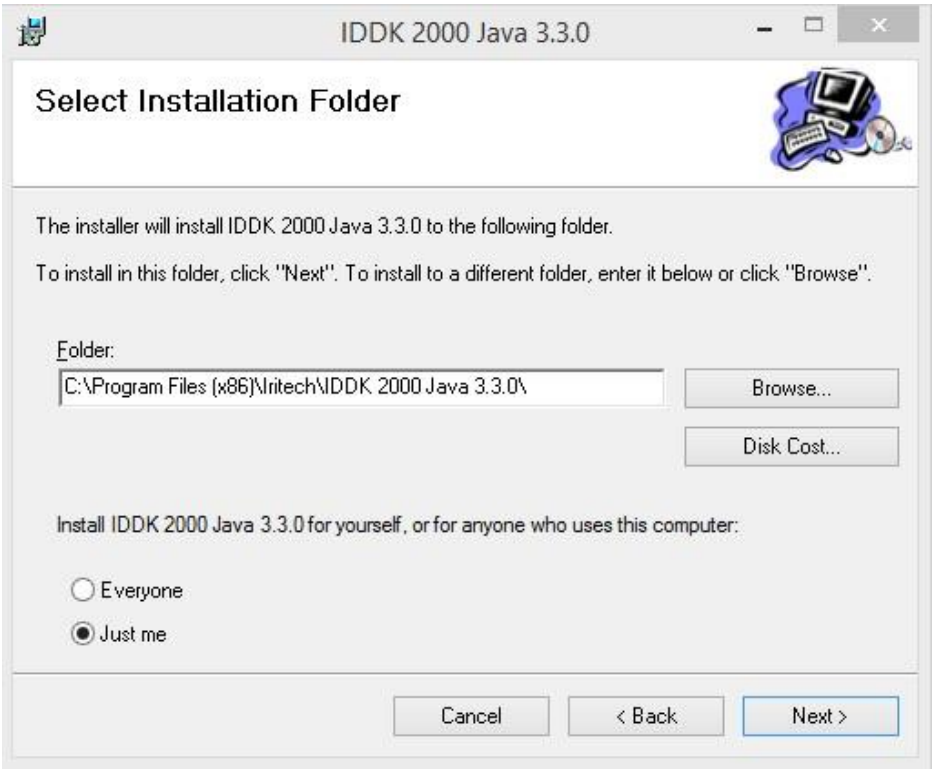
2.1.2.1 MS Windows XP and MS Windows 7

1. Download or copy the software package, then go to the "IDDK 2000 JAVA" folder. Double click on the "setup.exe" file.



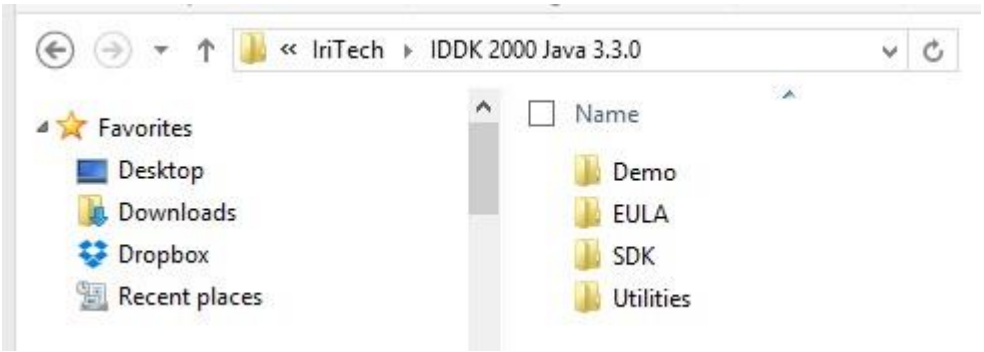
The installer will guide you through all the installation process. Select the appropriate options and click "**Next**" to move to the next steps.

2. If you want to install the SDK in a different folder instead of the one set as default, click "**Browse**" and specify the desired folder. If not, click "**Next**" to continue.



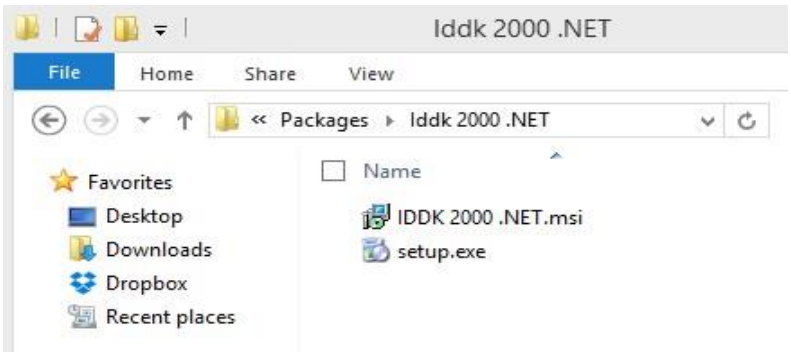
3. Follow the instructions from the installer to finish the setup process. When the following wizard appears, click "**Close**" to successfully complete the IDDK 2000 JAVA installation.

4. When finished, you can verify if the program has been properly installed by checking the installation folder.



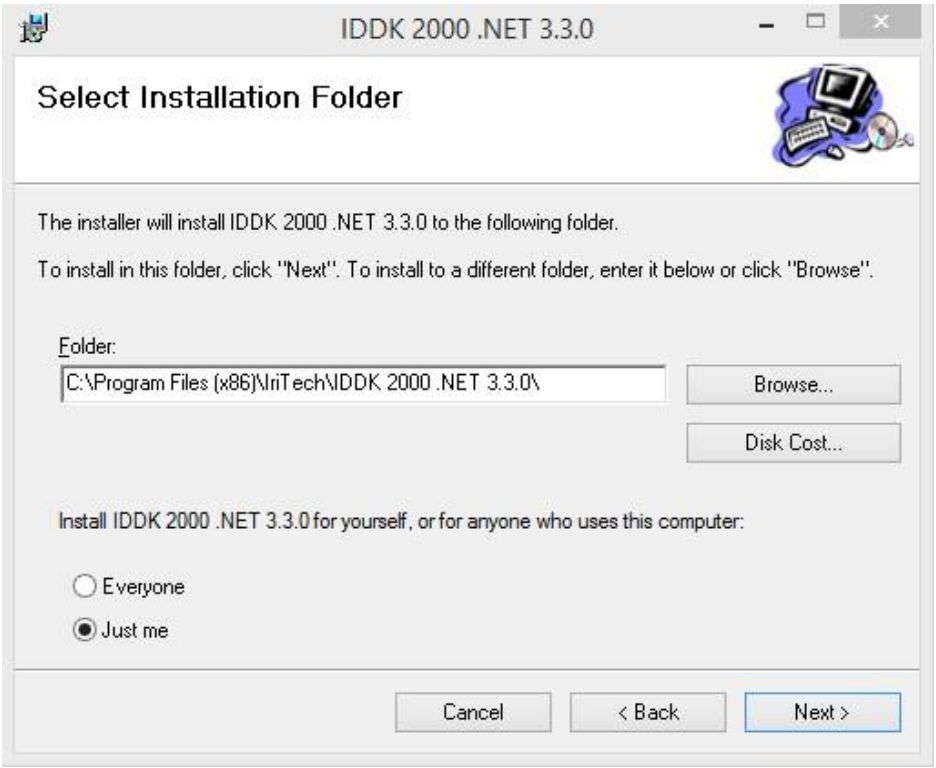
2.1.3 .NET

1. Download or copy the software package go to the “IDDK 2000 .NET” folder. Double click on the “setup.exe” file.

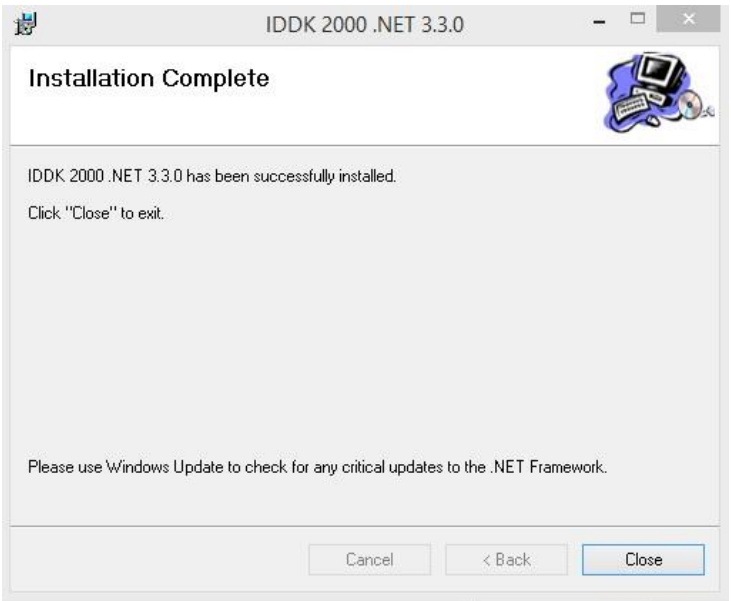


The installer will guide you through all the installation process. Select the appropriate options and click "**Next**" to move to the next steps.

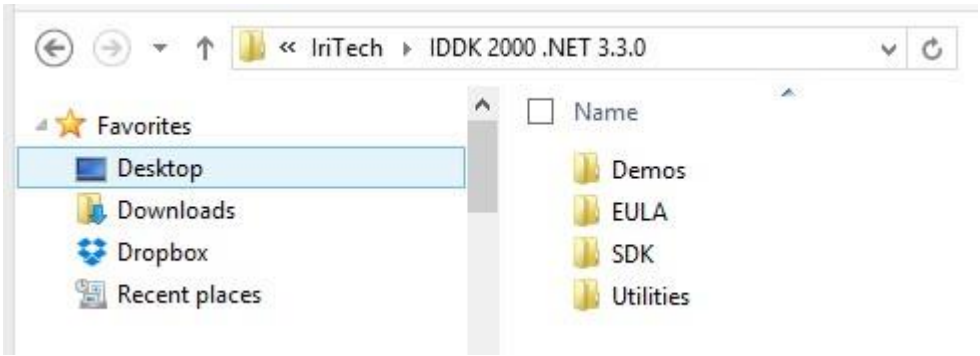
2. If you want to install the SDK in a different folder instead of the one set as default, click "**Browse**" and specify the desired folder. If not, click "**Next**" to continue.



3. Follow the instructions from the installer to finish the setup process. When the following wizard appears, click "**Close**" to successfully complete the IDDK 2000 .NET installation.



4. When finished, you can verify if the program has been properly installed by checking the installation folder.



2.2 Device driver installation

NOTE: The following sections give instructions on how to install IriShield driver in USB operation mode on different host platforms.

2.2.1. MS Windows XP and MS Windows 7

The details on how to install driver on these operating systems can be found in the Basic User's Guide.

2.2.2. MS Windows CE

Please contact IriTech, Inc. and provide us with more specific information about the platform you are using to develop your host system. The binaries mentioned in the following instructions are supposed to be completely compatible with your platform. They are included in the IDDK 2000 installation package. Please see 2.1.1.2 for the SDK setup process.

2.2.2.1 Driver Installation for End-user

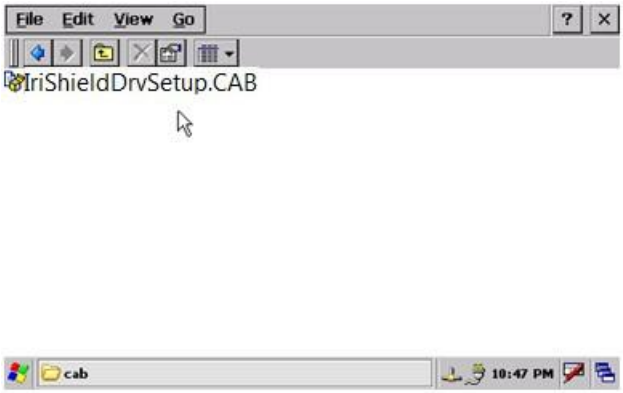
In embedded environments such as Windows CE, driver installation may be demanded when host device has been already deployed and in use. The driver package consists of the following file.

- **IriShieldDrvSetup.cab**: An installation cabinet for IriShield driver and all required settings.

NOTE: Installation using cabinet file only works when run-time image of the host device supports Window CE loader (SYSGEN_WCELOAD).

User should follow the below steps to install USB driver for IriShield.

1. Copy the installation cabinet file *IriShieldDrvSetup.cab* to the host device.
2. On host device, use File Explorer to view and open *IriShieldDrvSetup.cab*.



3. Click OK when prompted with the following pop-up window. Driver file will be copied to your device and proper registry keys will be configured to automatically recognize IriShield.

If Windows CE loader is not available in host device, direct installation from .cab file will fail.

2.2.2.2 Built-in Driver for Windows CE run-time image

IriShield driver can be easily built-in into the run-time image of the host device so that it is always available along with the operating system. This integration needs to be done during the development of host device's run-time image.

1. Copy *icamm7d0drv.dll* to a folder on your development computer such as *C:\<YOUR_DIR>.*
2. Ask Platform Builder to integrate *icamm7d0drv.dll* into the final run-time image by adding the following lines to *platform.bib* of your BSP which may be located in *%_PLATFORMROOT%\< YOUR_BSP>\Files* directory

   (Ex: *C:\WINCE600\PLATFORM\SMDK6410\FILES\platform.bib*):

```
;-------------- IriShield USB Driver ----------------------------------
icamm7d0drv.dll C:\ <YOUR_DIR>\ icamm7d0drv.dll  NK  SHK
;----------------------------------------------------------------------
```

3. Register *icamm7d0drv.dll* as driver binary for all IriShield devices of which your device is supposed to support by adding the appropriate registry settings to

platform.reg of your BSP which may be located in the *%_PLATFORMROOT%\<YOUR_BSP>\Files*

(Ex: *C:\WINCE600\PLATFORM\SMDK6410\FILES\platform.reg*):

```
;-------------- IriShield USB Driver ----------------------------------
[HKEY_LOCAL_MACHINE\Drivers\USB\ClientDrivers\IriShield]
    "DLL"=" icamm7d0drv.dll"
    "Prefix"="IRD"
"IClass"="{573E8C73-0CB4-4471-A1BF-FAB26C31D384}"
```

4. Rebuild your run-time image.

2.2.3. Linux

There are two driver alternatives for IriShield devices in Linux environment: using the driver developed by IriTech or using usbfs (USB Device File System).

2.2.3.1 Using IriTech driver

1. Copy device driver folder to your host computer (e.g., /home/iritech/Downloads).
   Note: The IriShield driver folder is copied to the PC when installing the IDDK package in 2.1.1.3.
2. Open a Terminal. **Change to root privilege.** Go to the directory that contains the installation package (e.g., /home/iritech/Downloads/drivers/icamm7d0):

```
root@Mozart-computer:/home/iritech/Downloads/icamm7d0# ls
built-in     installdriver.sh  iritech.db     iritech.run     readme
driver.conf  install.sh        iritech.rules  iritech.run.64  root.sh
root@Mozart-computer:/home/iritech/Downloads/icamm7d0# chmod 777 install.sh installdriver.s
h root.sh iritech.run iritech.run.64
root@Mozart-computer:/home/iritech/Downloads/icamm7d0#
```

3. Modify the driver configuration file “driver.conf” depending on your host computer.

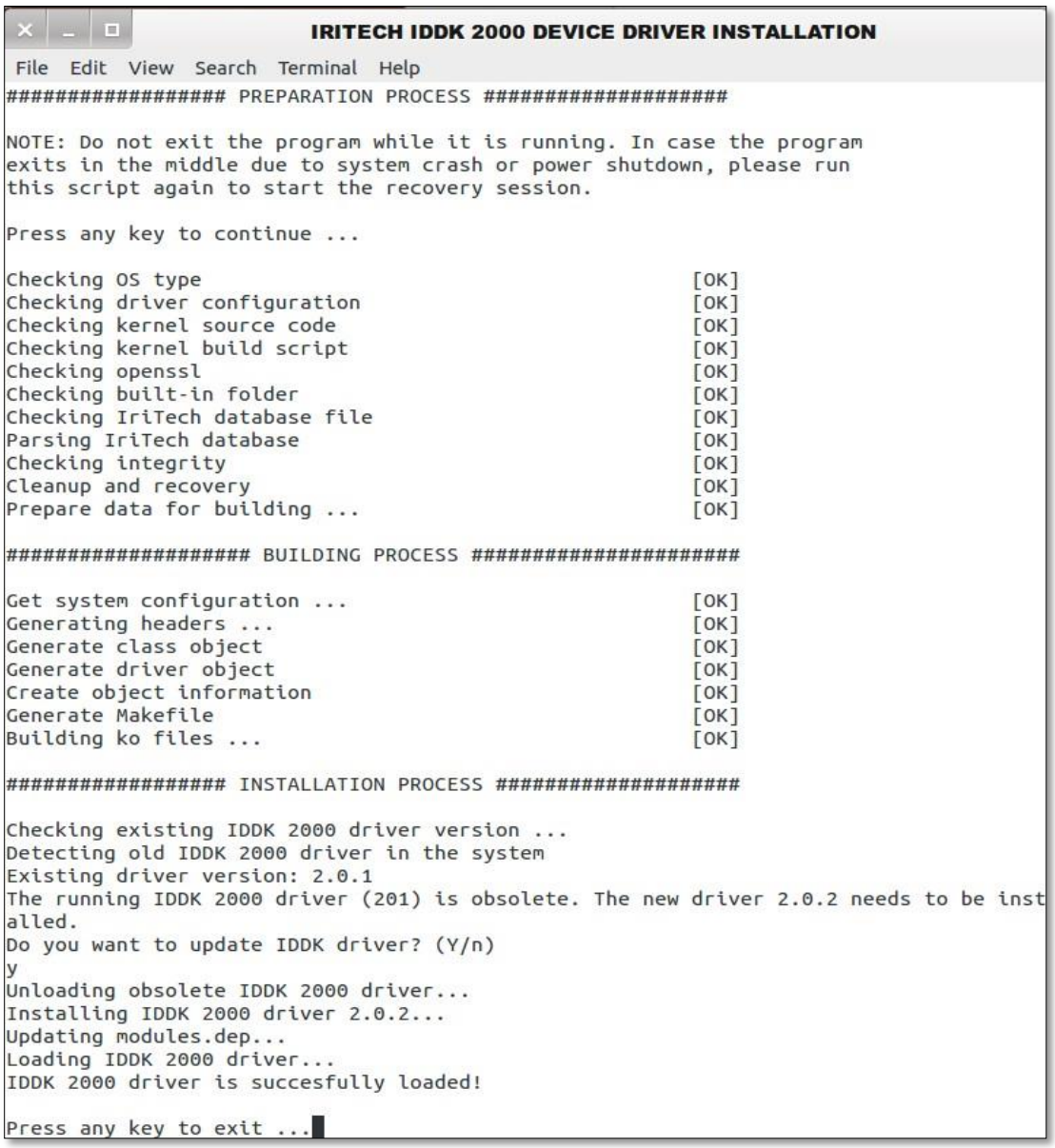
```
################################################################################################
#                                                                                              #
#       CONFIGURATION FILE, PLEASE MODIFY THESE VALUES ACCORDING TO YOUR SYSTEM                #
#                                                                                              #
#                        Copyright (c) IriTech, Inc. 2000 - 2013                               #
#                                                                                              #
################################################################################################

# This is the path to the built kernel source code
# In case ELinux, you have to build the kernel souce code by yourself in advance and
# specify the path to the kernel source code (was compiled) here
# For example: KERNEL_BUILD_PATH=/lib/modules/`uname -r`/build
KERNEL_BUILD_PATH=/lib/modules/`uname -r`/build

# This is the target platform, we only support "Linux" or "ELinux"
# For example: TARGET_PLATFORM=Linux
TARGET_PLATFORM=Linux

# The path to the toolchain, to build driver for ELinux, must specify the path to the toochain
# For example: CROSS_TOOLCHAIN=/opt/iritech/bin/arm-linux-uclibcgnueabi-
CROSS_TOOLCHAIN=gcc

# Architecture value is used to build driver for ELinux
# For example: ARCHITECTURE=arm
ARCHITECTURE=x86
```

There are four variables that must be modified before building and installing IriShield drivers on host computer:

a. KERNEL_BUILD_PATH is the path to the kernel source code. Note that the kernel source code must be already compiled in advance.
b. TARGET_PLATFORM is used to specify the target running system is Linux or Embedded Linux
c. CROSS_COMPILE is used only for Embedded Linux. This variable specifies the path to the toolchain that is used to compile the system's kernel source code
d. ARCHITECTURE is used only for Embedded Linux. This variable specifies the target running platform.

4. Build and install IriShield drivers into the target system.

```
root@Mozart-computer:/home/iritech/Downloads/icamm7d0# ./install.sh
```



2.2.3.2 Using usbfs (USB Device File System)

usbfs is enabled by default in most up-to-date Linux desktop system such as Fedora, Ubuntu, etc. No further steps are needed.

2.2.4. Mac OS X

usbfs is enabled by default. No further steps are needed.

2.2.5. Embedded Linux

There are two driver alternatives for IriShield devices in embedded Linux environment: using the driver developed by IriTech or using usbfs (USB Device File System).

2.2.5.1 Using IriTech driver

1. Copy device driver folder to your host computer (e.g.,/home/iritech/Downloads).
2. Open a Terminal. **Change to root privilege.** Go to the directory that contains the installation package (e.g., /home/iritech/Downloads/drivers/icamm7d0):

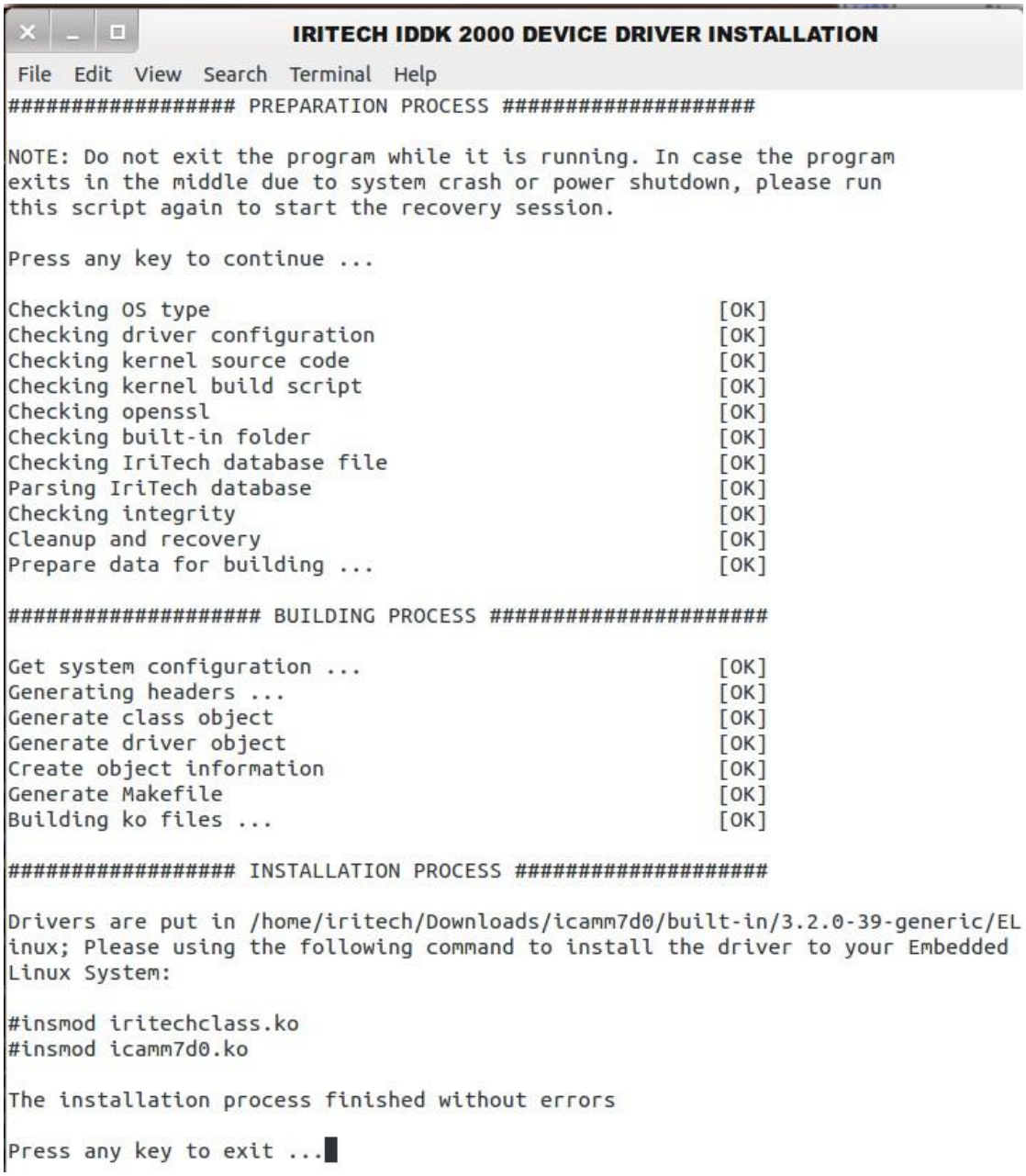
```
root@Mozart-computer:/home/iritech/Downloads/icamm7d0# ls
built-in     installdriver.sh  iritech.db     iritech.run     readme
driver.conf  install.sh        iritech.rules  iritech.run.64  root.sh
root@Mozart-computer:/home/iritech/Downloads/icamm7d0# chmod 777 install.sh installdriver.s
h root.sh iritech.run iritech.run.64
root@Mozart-computer:/home/iritech/Downloads/icamm7d0#
```

3. Modify the driver configuration file “driver.conf” depending on your host computer.

```
#############################################################################################
#                                                                                           #
#       CONFIGURATION FILE, PLEASE MODIFY THESE VALUES ACCORDING TO YOUR SYSTEM             #
#                                                                                           #
#                         Copyright (c) IriTech, Inc. 2000 - 2013                           #
#                                                                                           #
#############################################################################################

# This is the path to the built kernel source code
# In case ELinux, you have to build the kernel souce code by yourself in advance and
# specify the path to the kernel source code (was compiled) here
# For example: KERNEL_BUILD_PATH=/lib/modules/`uname -r`/build
KERNEL_BUILD_PATH=/home/iritech/Downloads/Test/source/kernel/linux-2.6.29.0

# This is the target platform, we only support "Linux" or "ELinux"
# For example: TARGET_PLATFORM=Linux
TARGET_PLATFORM=ELinux

# The path to the toolchain, to build driver for ELinux, must specify the path to the toochain
# For example: CROSS_TOOLCHAIN=/opt/iritech/bin/arm-linux-uclibcgnueabi-
CROSS_TOOLCHAIN=/opt/toolchain/arm-2009q3/bin/arm-none-linux-gnueabi-

# Architecture value is used to build driver for ELinux
# For example: ARCHITECTURE=arm
ARCHITECTURE=arm
```

4. Build IriShield drivers on your host computer.

```
root@Mozart-computer:/home/iritech/Downloads/icamm7d0# ./install.sh
```



5. Copy two driver files to the Embedded System and install them using the following commands:

```
[root@EDK-SC100 icamm7d0]$insmod iritechclass.ko
[root@EDK-SC100 icamm7d0]$insmod icamm7d0.ko
[root@EDK-SC100 icamm7d0]$
```

6. Plugin IriShield to the Embedded System, wait for 10 seconds, and do as follows:

```
[root@EDK-SC100 icamm7d0]$cd /sys/class/iritech/irishield0/
[root@EDK-SC100 irishield0]$ls
dev        device     subsystem  uevent
[root@EDK-SC100 irishield0]$cat dev
252:0
[root@EDK-SC100 irishield0]$
```

In some Embedded System, device node (usually `irishield0`) is not created automatically by the system due to lack of `udev`. We can check the existence of this device node in directory `/dev`.

```
[root@EDK-SC100 irishield0]$ls /dev/
apm_bios     hda9         log          pts          snd          ttySAC1
appdev       hdb          loop0        ptyp0        sndstat      ttySAC2
camera       hdb1         loop1        ptyp1        spi          ttySAC3
console      hdb10        mem          ptyp2        spi0         ttyST0
dsp          hdb11        misc         ptyp3        spi1         ttyST1
fb           hdb12        mmcblk0      ptyp4        touch        ttyST2
fb0          hdb13        mmcblk0p1    ptyp5        touchscreen  ttyST3
fb1          hdb14        mmcblk1      ptyp6        tty          ttyUSB0
fb2          hdb2         mmcblk1p1    ptyp7        tty0         ttyUSB1
fb3          hdb3         mtd0         ptyp8        tty1         ttyUSB2
fb4          hdb4         mtd1         ptyp9        tty2         ttyUSB3
hda          hdb5         mtd2         ram          tty3         ttyp0
hda1         hdb6         mtd3         ram0         tty4         ttyp1
hda10        hdb7         mtd4         ram1         tty5         ttyp2
hda11        hdb8         mtd5         ram2         tty6         ttyp3
hda12        hdb9         mtdblock0    ram3         tty7         ttyp4
hda13        hsspi        mtdblock1    random       ttyP0        ttyp5
hda14        i2c-0        mtdblock2    rtc          ttyP1        ttyp6
hda2         i2c-1        mtdblock3    rtc0         ttyP2        ttyp7
hda3         i2c-2        mtdblock4    sda0         ttyP3        ttyp8
hda4         initctl      mtdblock5    sda1         ttyS0        ttyp9
hda5         input        net          sda2         ttyS1        urandom
hda6         irda         null         sda3         ttyS2        video
hda7         iricamm0     psaux        sda4         ttyS3        watchdog
hda8         kmem         ptmx         shm          ttySAC0      zero
[root@EDK-SC100 irishield0]$
```

In case there is no device node in directory `/dev`. We can make it manually by the following command:

```
[root@EDK-SC100 irishield0]$mknod /dev/irishield0 c 252 0
[root@EDK-SC100 irishield0]$chmod 777 /dev/irishield0
[root@EDK-SC100 irishield0]$
```

The number 252 and 0 can be found in /sys/class/iritech/irishield0/dev.

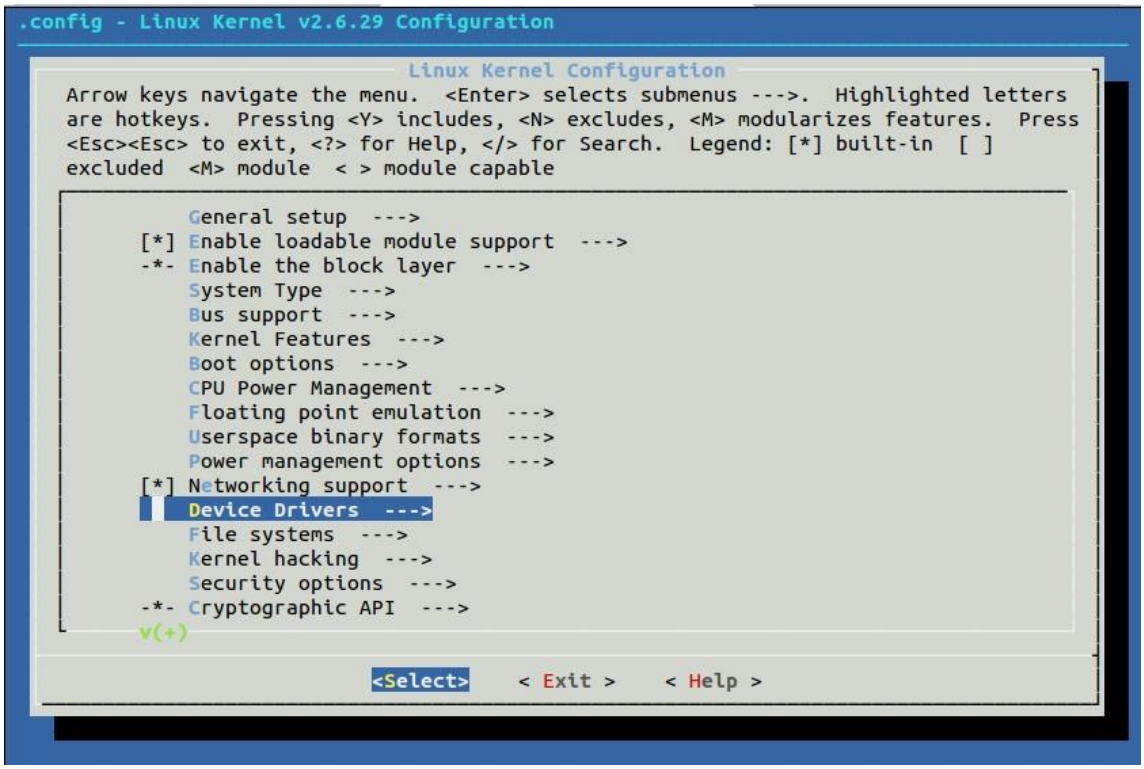
2.2.5.2 Using usbfs (USB Device File System)

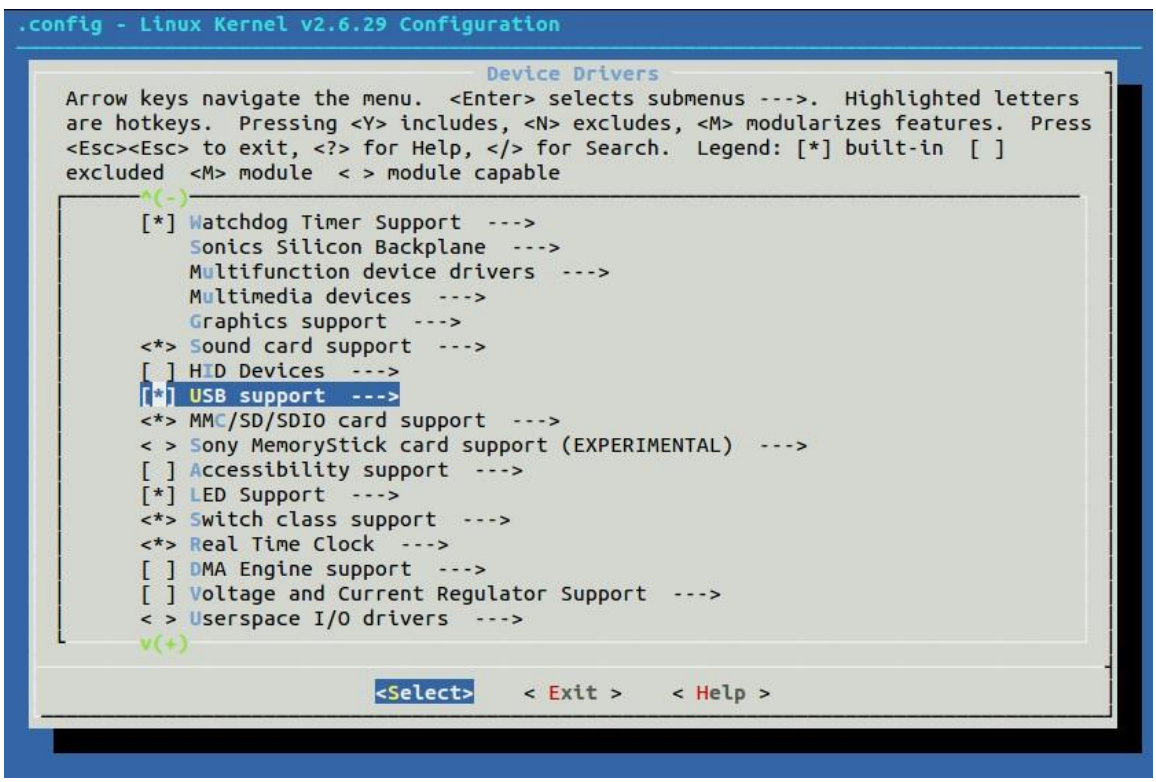
Some embedded Linux systems may not enable this usbfs by default. Developers must make further steps in the process of building their kernel images to make sure usbfs is enabled.

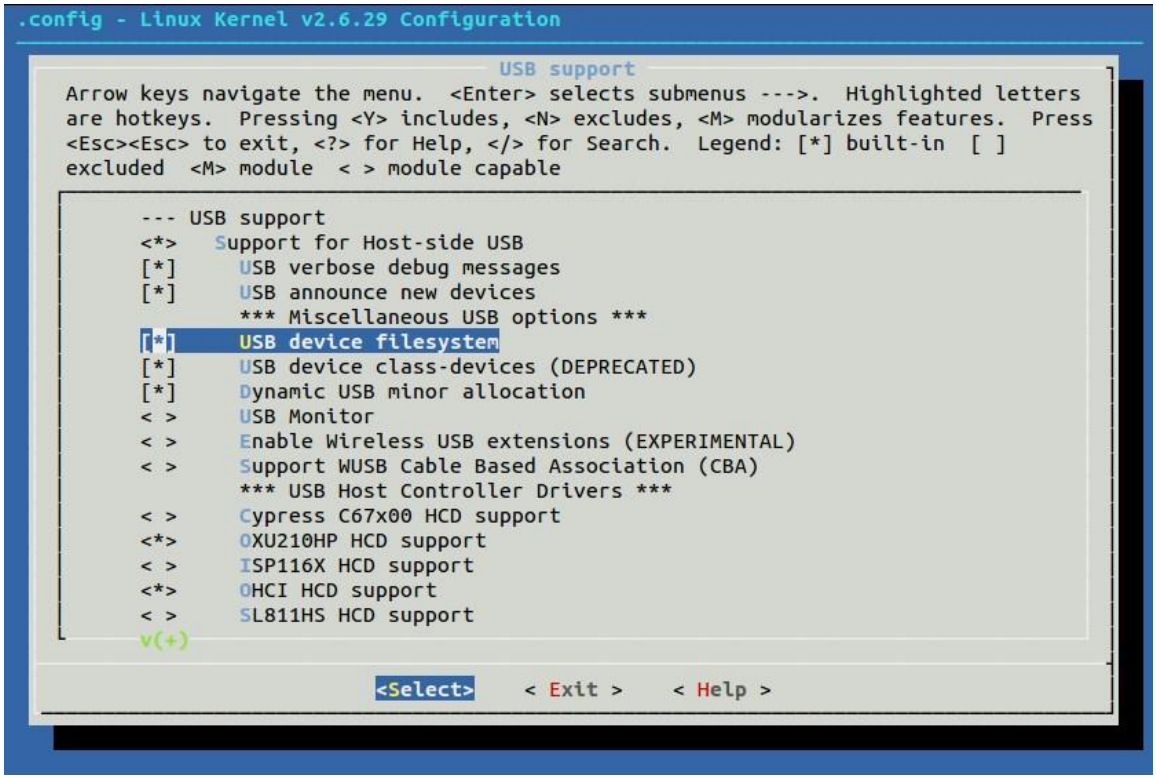
The following guidance shows how to configure the kernel image in general to support usbfs. However, it may differ from system to system. Therefore, please use it as a reference only.

```
#make ARCH=arm CROSS_COMPILE=<path to toolchain> menuconfig
```

When the configuration dialog appears, select ***Device Drivers → USB Support → USB device filesystem***.





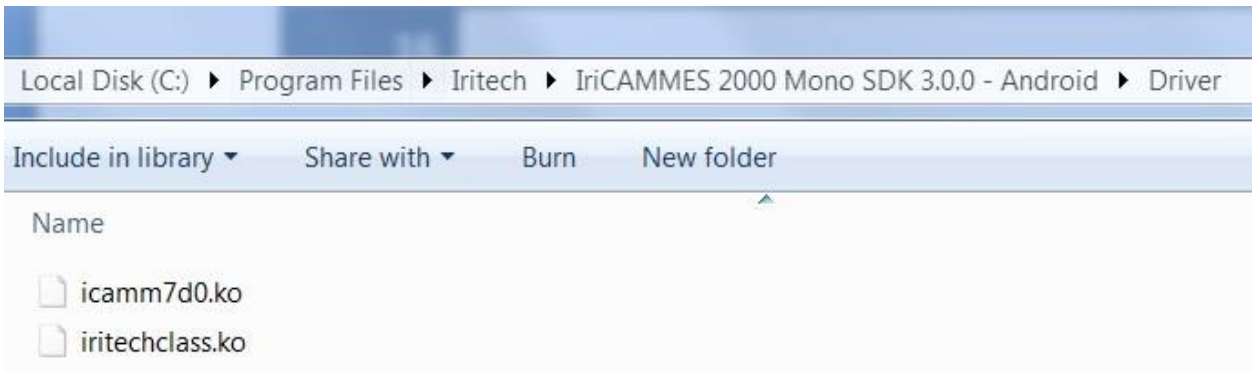


2.2.6. Android

2.2.6.1 Android - Native Code

First, please refer to 1.2.12 for a special note for Android platforms.
The binaries mentioned in the following instructions are supposed to be completely compatible with your platform. They are included in the IDDK 2000 for Android installation package. Please contact IriTech to get the SDK installation package.

After installing IDDK 2000 for Android, open folder “Driver” to find two loadable modules: iritechclass.ko and icamm7d0.ko.



Please copy two files into your embedded system and use two following commands to install the device driver: (iritechclass.ko is installed before icamm7d0.ko)


```
C:\Program Files\Iritech\IriCAMMES 2000 Mono SDK 3.0.0 - Android\Driver>adb push iritechclass.ko /iritech
3680 KB/s (7537 bytes in 0.002s)

C:\Program Files\Iritech\IriCAMMES 2000 Mono SDK 3.0.0 - Android\Driver>adb push icamm7d0.ko /iritech
455 KB/s (10718 bytes in 0.023s)

C:\Program Files\Iritech\IriCAMMES 2000 Mono SDK 3.0.0 - Android\Driver>adb shell
# cd /iritech
cd /iritech
# ls
ls
iritechclass.ko
icamm7d0.ko
#

# insmod iritechclass.ko
insmod iritechclass.ko
# insmod icamm7d0.ko
insmod icamm7d0.ko
#
```

Next, copy the following line to */ueventd.rc*:

```
/dev/irishield*              0777   root       root
```

2.2.6.2 Android - USB Host Mode

This alternative requires no driver installation. Instead, SDK utilizes Android USB Host APIs to communicate with IriShield.

2.2.7. Other Operating Systems and Non-OS Platforms

Please refer to 1.2.11.

3. Software Specification

3.1 Library Description

IDDK 2000 provides libraries of ready-to-use APIs to be easily called through applications programmed in C/C++, Java, and .NET programming languages (Visual C++, Visual C#, Visual Basic .NET).

Developers can use this library to easily control and customize the functionalities of IriTech's device, such as device management, capturing process, template generation, gallery management, 1:1 and 1:N matching, and secured communication.

3.1.1 C/C++

3.1.1.1 Windows XP and Windows 7

The library includes the following components:

- Three header files: Iddk2000Apis.h, Iddk2000Defs.h, and IddkBase.h
- One import library file: Iddk2000.lib
- One binary file: Iddk2000.dll

3.1.1.2 Windows CE

The library includes the following components:

- Three header files: Iddk2000Apis.h, Iddk2000Defs.h, and IddkBase.h
- One import library file: CEIddk2000.lib
- One binary file: CEIddk2000.dll

3.1.1.3 Linux

The library includes the following components:

- Three header files: Iddk2000Apis.h, Iddk2000Defs.h, and IddkBase.h
- One dynamic shared library file: libIddk2000.so

3.1.1.4 Mac OS X

The library includes the following components:

- Three header files: Iddk2000Apis.h, Iddk2000Defs.h, and IddkBase.h
- One dynamic shared library file: libIddk2000.dylib

3.1.1.5 Embedded Linux

The library includes the following components:

- Three header files: Iddk2000Apis.h, Iddk2000Defs.h, and IddkBase.h
- One dynamic shared library file: libEIddk2000.so

3.1.2 Java

The library includes the following components:

- Two binary files: Iddk2000.dll, Iddk2000jni.dll
- One library package: Iddk2000-<Version>.jar

3.1.3 Net

The library includes the following components:

- Two binary files: Iddk2000.dll, Iddk2000DotNet.dll
- One documentation file: Iddk2000DotNet.XML

3.2 Standard Capturing Procedure

IDDK 2000 offers different operation modes, capturing modes, and quality tolerance levels to help developers balance ease of the capturing procedure against high iris image quality.

3.2.1 Operation Modes

- **Auto Capture Mode:** As soon as the capturing process is activated, every image from the camera is automatically streamed through a Quality Measurement module (QM) for real-time evaluation, to be classified as good or bad, and to have the best image selected at the end. All activities in this operation mode are automated by the device without the outside support.

- **Operator-Initiated Auto Capture Mode:** In case a user is unfamiliar with the system and has difficulties presenting eyes correctly to the sensor, the Operator-Initiated Auto Capture Mode allows the assistance from an operator to control the evaluation process. Specifically, at first the images from the camera are streamed to host for reviewing purpose but not to be evaluated yet. Whenever the user is in the proper position in front of camera, the operator will send a command to the device to initiate the automated capturing process.

3.2.2 Capturing Modes

- **Frame-Based Search Mode:** In the Frame-Based Search Mode, the developer specifies the number of qualified images that must be captured before the capturing process ends. An image is considered qualified when it passes the basic level of qualities from the QM module. Once the desired number of qualified images is reached, the best image will be selected and checked against the Minimum Quality Tolerance condition. If satisfied, it will be returned as the captured image. Otherwise, no image is returned and user must initiate the capturing process again.

- **Timed-Based Search Mode:** In the Timed-Based Search Mode, once the first eye image is detected, the device will continuously capture frames across a pre-defined period of time. When the period elapses, the best image will be selected among the captured frames and checked against the Minimum Quality Tolerance (as defined below). If satisfied, the image will be returned as the captured image. Otherwise, no image is returned and user must initiate the capturing process again.

3.2.3 Minimum Quality Tolerance

The Minimum Quality Tolerance parameter specifies the minimum quality level for the accepted iris images. They are classified as follows:

- **Normal:** This is the least stringent quality requirement for the accepted frames and is suitable for everyone regardless of their eye colors and their races. With this choice, the camera operation works in the most user-friendly condition, but the tradeoff is that the system may accept a lower quality image as output. Frankly, most images accepted in this mode will be of sufficiently good quality. This mode is recommended by IriTech for speed and usability.

- **High:** In this setting, the iris quality thresholds are more stringent, but the camera operation may be less user-friendly. On the contrary, the system will exclude most low quality images as output. The “High” Minimum Quality Tolerance condition provides better analysis for the people with darker eyes such as Arabs, Africans, Asians, and Eastern Europeans.

- **Very High:** In “Very High” Minimum Quality Tolerance condition, the capturing process tends to accept only the highest quality images, which may require more than one trial for some users. However, using the “Very High” mode, the quality of every output image is in a perfect condition. This mode is recommended when user’s convenience is not a priority.

Developers may adjust the minimum quality threshold to a level suitable for their system’s capturing environment and desired iris image quality.

3.2.4 Standard Capturing Procedure

The IriShield device is equipped with an iris camera that captures one iris image at a time. A standard capturing process will lead to the following stages:

- **Ready:** In this stage, the capturing process is initiated. The infrared LED is turned on to lighten the texture inside the irises. The subject should place his/her eye at about 5 cm (for short range devices) away from the front of the camera in perpendicular direction and follow the capturing procedure mentioned in the Basic User's Guide, so that the camera sensor can detect the eye. In Auto Capture Mode, images from iris camera start streaming through Quality Measurement module for live evaluation. For detailed information on capturing distance range, please refer to the Hardware Specification document.

- **Capture:** Whenever the first eye image is detected, capturing process will change to the capturing stage. In this stage, the subject must continue to move slowly towards the camera if fixed-focused devices are used or stand still and allow camera lenses to move on their own if auto-focused devices are used.

- **Complete:** When the conditions for completing the capturing process have been met (i.e., a reasonable duration of time or a reasonable number of qualified eye images detected), the capturing process stops and enters the timeout/finish stage. The infrared LED will turn off.

- **Abort:** If something abnormal happens (e.g., no images from iris camera or capturing process is cancelled), it will cause the capturing process to terminate before finishing its normal routine and this stage will be entered. The infrared LED will turn off. The user should start the procedure over again.

More details on how to carry out the capturing procedure properly and on how to improve the quality of captured images can be found in the Basic User's Guide.

Every iris image satisfying the quality standards from the live QM module will be qualified for iris recognition. However, to enhance matching accuracy, not all of qualified images are selected. Only the image with the best quality is selected and used as the output image of that capturing process.

3.3 Iris Recognition Procedure

3.3.1 Template Generation

The captured images will be inputted into the Iris Recognition module of the IriShield. Here, the template generation process will be carried out at a very high speed. Template is a compact data structure in which all important iris features are extracted, compressed and encoded. Matching process is only conducted on templates.

Currently, the device supports a uniform type of template, N-template.

- **N-template:** The N-template contains sophisticated information of iris features of different iris images. It is very compact in size. It is about 4000 bytes for one-iris template and for each additional iris, a similar amount of bytes are added to the template. N-template also contains eye-side information, i.e., left, right, or unknown.

3.3.2 Template Gallery

IriShield owns a template gallery that can accommodate thousands of iris slots - depending on the NAND memory capacity - to serve for 1:N and 1:1 matching. The gallery is structured as a very simple database where templates are retrieved quickly using user-input IDs. Users can enroll/unenroll templates into/from the gallery and save the changes permanently.

3.3.3 Iris Recognition

Generally, iris recognition process can be classified into two different tasks: identification and verification.

- Identification (1-to-N) is to compare one template against a list of enrolled templates to find out whether it belongs to any of the enrollees and if so, what the identity of the found enrollee is.

- Verification (1-to-1) is to compare the two templates to see whether they are from the same eye or not.

Instead of providing binary identification and verification results, IriShield provides one matching distance for every pair of compared templates. The lower is the matching distance; the higher is the possibility that the two compared templates are from the same eye. The decision will be done by the application system using one distance threshold (dichotomous decision) or two (trichotomous).

Choosing a decision threshold value is a tradeoff between system security (avoiding false positive matches) and user convenience (minimizing false negative matches). For more details, please refer to the IDDK API Reference manual.

Actual comparison is computed only when the two templates have the same eye-side or when at least one of them has unknown eye-side. Therefore, when developers try to compare a left-eye template with a right-eye one, a big distance will be returned constantly. If one or both templates consist of multiple irises, the comparison result between them is the minimum distance among all distances of every possible pair of irises contained in the two templates.

3.4 Security Infrastructure

To secure data exchange between the device and the user (host), IriShield applies state-of-the-art cryptographic standards on all functions related to image and template transmission on USB bus. This security infrastructure is based on Public Key Infrastructure (PKI), employing 2048-bit RSA key pair for asymmetric encryption and signature, 256-bit AES for session encryption, and SHA1 for data digest. RSA key pairs are imported into the device in the form of an X509 certificate (for public key) and PFX/PKCS #12 certificate (for private key). IriShield,

however, only concerns cryptography keys inside each certificate. Expiry, Certification Authority (CA) verification, issuers and subjects as well as other parts of each certificate are left to customers' consideration.

3.4.1 PKI Key Distribution

In the cryptography mode, the device and its user must be provided with pairs of cryptographic keys - a public key and a private key - separately. The private keys must be kept confidential by each side while the paired public keys are made publicly available to the other party.

To implement this security feature, the device utilizes the following key notations:

- **CRCam** (private key of device): a private key used by the device to sign data sent to the user and to decrypt the data received from the user.
- **CUCam** (public key of device): a public key used by the user to verify data received from the device and to encrypt the data sent to the device.
- **CRCust** (private key of user): a private key used by the user to sign the data sent to the device and to decrypt the data received from the device.
- **CUCust** (public key of user): a public key used by the device to verify the data received from user and to encrypt the data sent to the user.

The device needs to be provided with the keys **CRCam** and **CUCust** before the user is able to call any function in cryptographic version. The user also needs to manage the corresponding keys **CRCust** and **CUCam** at his/her own site (Figure 1).

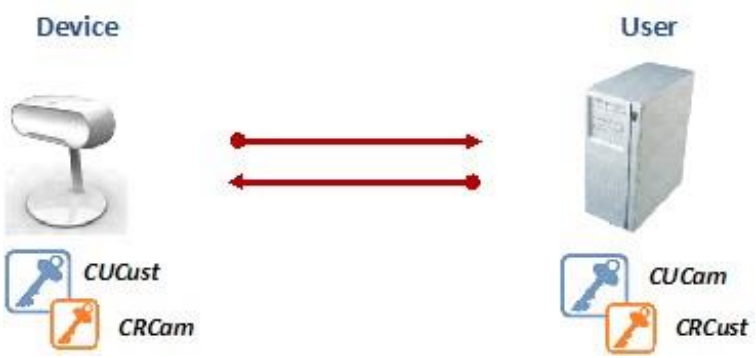


Figure 1. Distribution of keys between the device and the user

The valid keys are to be installed in the device using the appropriate API functions provided by IDDK 2000 (Figure 2).

- **CUCust** has to be imported first in an X509 DER-encoded format. To prevent unauthorized modification on **CUCust** from outside, any update on **CUCust** from the second time needs to be signed by the old **CRCust**.

- Device automatically generates a unique RSA 2048-bit key pair ***CRCam*** and ***CUCam*** when manufactured. ***CUCam*** can be acquired by calling to function ***GetCameraCertificate***(). Customers can keep and use this key pair or choose to provide the device with their own key pair in a PKCS #12 (PFX) packet. In the latter case, the PKCS #12 packet must be signed by ***CRCust*** and its corresponding password must be encrypted by the current ***CUCam***. The PKCS #12 or X509 packet must contain only the intended keys and certificate.

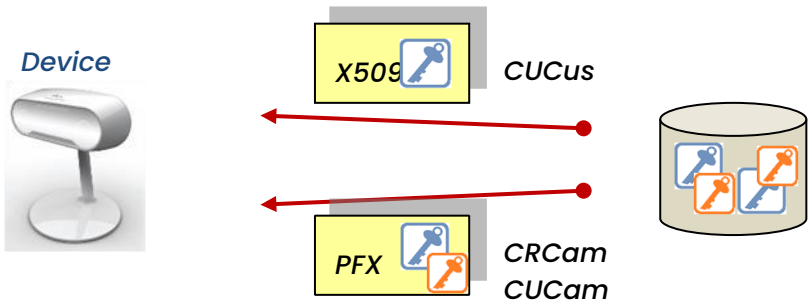


Figure 2. Importing keys into the device

In order to be imported into the device, X509 and PKCS #12 certificates must be signed and organized as follows.

Certificate Length (4 bytes)	Certificate Data (X509)	Signature Length (4 bytes)	Signature Data (Using SHA)

- Certificate Length: Size in bytes of the Certificate Data
- Certificate Data: Byte array containing certificate information
- Signature Length: Size in bytes of the signature signed on Certificate Data
- Signature Data: Signature signed on Certificate Data

To prevent the password from being exposed during network transmission, it must be encrypted by a random IV and a random AES-256 key in CBC mode. The AES key and IV are then encrypted by ***CUCam***.

Encrypted Signed Data Length (4 bytes)	Encrypted Signed Data	Encrypted Session Key Length (4 bytes)	Encrypted Session Key Data (RSA-encrypted)

- Encrypted Password Length: Total length of Encrypted Password
- Encrypted Password: The password encrypted by AES-256
- Encrypted Session Key Length: Size of the Encrypted Session Key Data
- Encrypted Session Key Data: Encrypted IV and 256-bit AES key. After "Encrypted Session Key Data" is decrypted, the first 16 bytes of plain data will be the IV and the next 32 bytes (256 bits) is the AES key itself.

***Note**: Each above key pair is supposed to be valid for both encryption and signature; otherwise, the import will be failed.*

3.4.2 End-to-End Security

Once all necessary cryptography keys are available at both sides, templates and captured images can be exchanged through USB bus in encrypted signed data streams. The cryptography structure based on PKI involves the use of asymmetric key algorithms and symmetric key algorithms. Briefly, one side (device or user) has to attach its signature to the data (template or image) needed to be sent to the other side. The signed data are in turn encrypted with a random 256-bit AES key (called session key) and a random Initialization Vector using CBC cipher mode and PKCS #5 padding scheme. The key and initialization vector are then encrypted with the recipient's RSA public key before being sent together with the encrypted data. At the recipient side, the received encrypted signed data will be decrypted and verified using the appropriate keys to get the necessary data which have been kept confidential and consistent.

To make the secured communication clear between device and user, the following diagram illustrates the processes of sending and receiving data (template or image) between device and user (Figure 3).

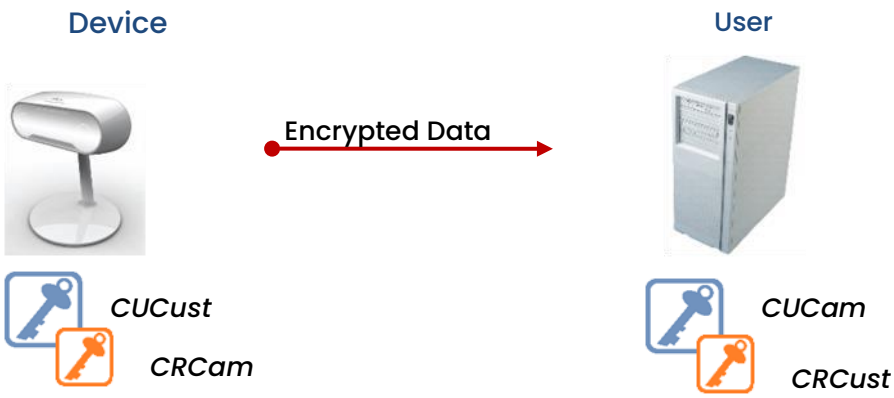


Figure 3. Sending data from the device to the user

3.4.2.1 Sending data from device to user

To send data (template or image) from the device to the user, the data needs to be signed and encrypted (Figure 4).

- The Data is first hashed using SHA1 to form the Data Digest.
- Next, the Data Digest will be encrypted using device's private key **CRCam** (sender's private key) to form the Data Signature. The Data and Data Signature are structured as a Signed Data BLOB as in following:

Data Length (4 bytes)	Data	Data Signature Length (4 bytes)	Data Signature

- Data Length: Size of to-be-exchanged Data
- Data: Data in binary
- Data Signature Length: Size of Data Signature
- Data Signature: Binary form of Data Signature signed by the private key **CRCam**

- Next, the Signed Data BLOB will be encrypted using a random 256-bit AES key and a random Initialization Vector to form an Encrypted Signed Data. The symmetric encryption uses CBC cipher mode and PKCS #5 padding scheme.

- Next, the AES key and Initialization Vector will be encrypted with user's public key **CUCust** (receiver's public key). The asymmetric encryption uses PKCS #1 block type 2.

- Lastly, the ready-to-be-exchanged data including the Encrypted Signed Data and the Encrypted Session Key Data will be organized as an Encrypted Data and sent to user.

Encrypted Signed Data Length (4 bytes)	Encrypted Signed Data	Encrypted Session Key Length (4 bytes)	Encrypted Session Key Data (RSA-encrypted)

- Encrypted Signed Data Length: Total length of Encrypted Signed Data o Encrypted Signed Data: The encrypted signed data generated from the above o Encrypted Session Key Length: Size of the Encrypted Session Key Data
- Encrypted Session Key Data: Encrypted Initialization Vector and 256-bit AES key. After this "Encrypted Session Key Data" is decrypted, the first 16 bytes of plain data will be the Initialization Vector and the next 32 bytes (256 bits) is the AES key itself.

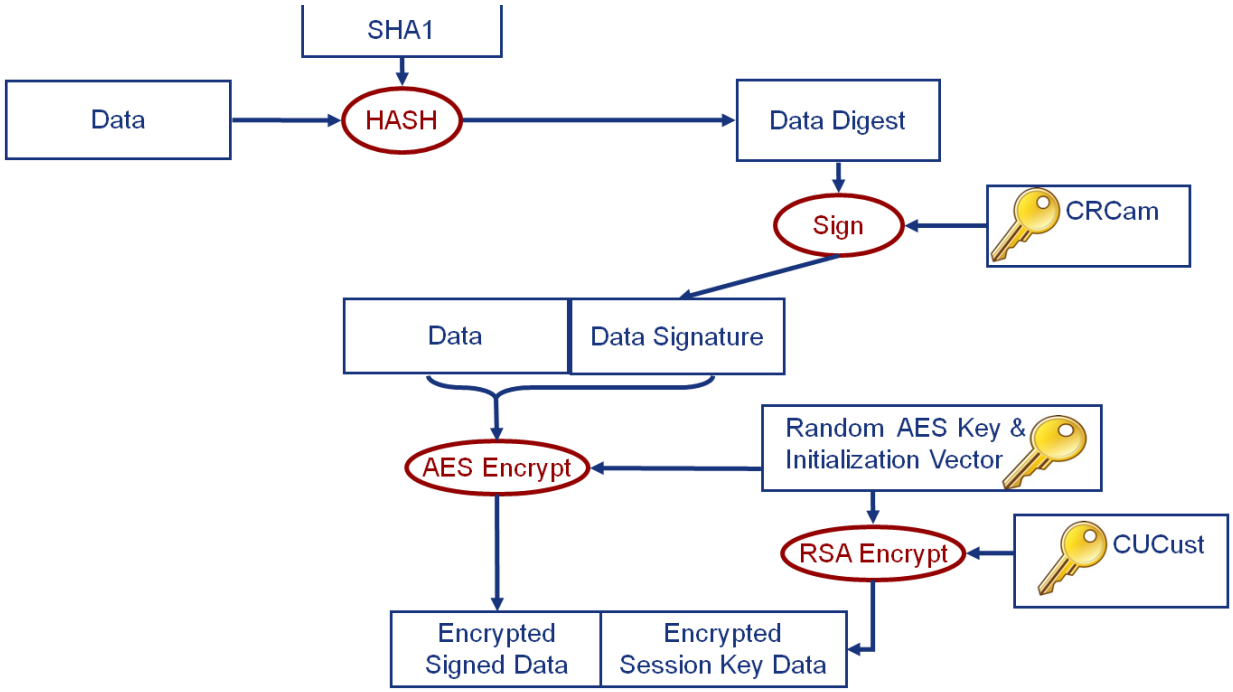


Figure 4. Signing and encryption of data

Note: *All data are in little endian byte order.*

3.4.2.2 Receiving data from device

At the user side, the encrypted data will be decrypted to obtain the original one and verified to ensure the integrity of that data.

- The Encrypted Session Key Data will first be decrypted using user's private key **CRCust** (receiver's private key) to extract the AES key and Initialization Vector.

- Next, the Encrypted Signed Data will be decrypted using the extracted AES key and Initialization Vector to obtain the Data and Data Signature.

- The Data will be hashed to form the Data Digest.

- Lastly, the Data Signature will be decrypted using device's public key **CUCam** (sender's public key) and verified with the Data Digest to check the integrity of the received data.

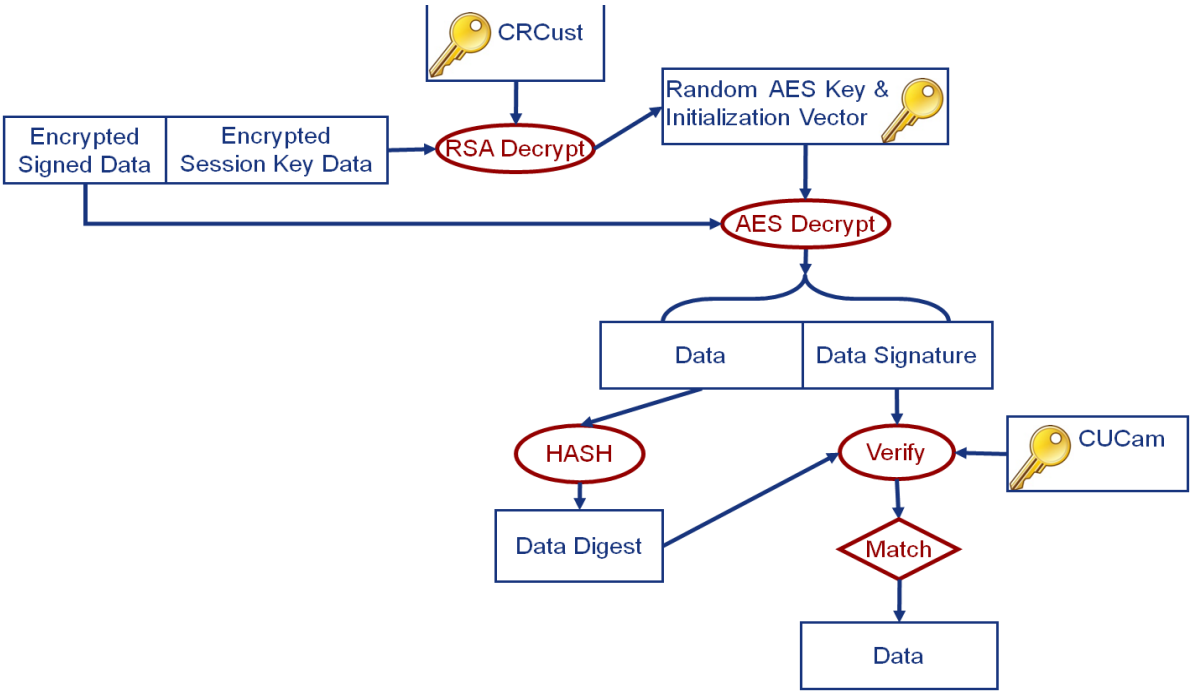


Figure 5. Decryption and verifying of data

The process of sending data from user to device can be carried out similarly in the reverse direction using the appropriate keys.

3.4.3 Administrative Audits

IriShield also supports users with audit information so the usage and performance of each capturing station and each operator can be easily monitored and traced. Users are allowed to add their own managing data such as timestamp, capturing station, capturing operator ID, and other user-specific data inside each encrypted data piece as a header. The user header is organized into volatile data and nonvolatile data.

- Volatile data such as timestamp changes every time encrypted data is retrieved.
- Nonvolatile data such as device location seldom changes.

IDDK 2000 supplies one API to update the nonvolatile data, while the volatile information must be inputted every time a cryptographic function is called.

3.5 Supported Image Types

- Captured image with original size (K1)
- VGA image (K2)
- Cropped image (K3)
- Cropped and Masked Image (K7)

Please refer to the specification of ISO/IEC 19794-6:2011 for more detailed information.

3.6 Supported Image Formats

3.6.1 IriTech Standard Image Format

IriTech has its own Standard Image Format and only manipulates on the type called the IriTech image. This standard format encapsulates images captured from IriTech's devices. Besides the raw image data, it contains the header information and an image data signature, which differentiates the IriTech image from any other data format.

By employing its own IriTech image, IriShield device can operate more reliably and precisely. Before carrying out any functions on the image, the image sections are examined to verify whether any part is corrupted or not. If any, the image will be rejected. Moreover, the device will not accept any other image formats except the IriTech image. This IriTech image cannot be processed anywhere else except in an IriTech device. This ensures the operations of the IriShield devices to be secure and safe.

IriTech image can be used compatibly among IriTech's products.

3.6.2 Raw Image Format

IriShield devices output raw image directly and also allow users to covert an IriTech image into a raw image format (size 640x480). This raw format simply contains the plain image data with no other information. It allows users themselves to manipulate on the captured image data as desired.

3.6.3 JPEG2000 Image

Image can be outputted in JPEG2000 format with compression quality from 1 to 100.

3.6.4 Iris ISO Standard Image Format

In the year of 2005, the International Standards Organization (ISO) defined the iris standard format ISO/IEC 19794-6 to facilitate data exchange among biometric authentication systems that implement iris recognition. IriShield is capable of producing images in this standard format, allowing easy integration with other biometric systems.

ISO/IEC 19794-6 specifies image compression including JPEG and JPEG2000. IriShield offers two options in gray scale format: raw image and compressed JPEG2000 image.

Recently, ISO announced the iris standard format ISO/IEC 19794-6 version 2011 with some modifications from the old version 2005. The most important modification made was to support a variety of Image Kinds as follows: K1, K2, K3, and K7.

Image Kind	Definition
K1	Iris image of any size
K2	Iris image of size VGA 640x480 pixels
K3	Iris image where non-iris parts are cropped
K7	Iris image where non-iris parts are cropped and masked

With the introduction of K3 and K7, customers are offered with more options to store just the useful parts of iris images in which it also leads to improving storage ability as well as lowering overhead transmission images through communication channels.

4. Demonstration Code and Utilities

The following step-by-step tutorials and the corresponding demonstration code show you how to use IDDK 2000 in a basic manner. We will purposely cover all the major abilities of the IDDK to provide the best support for customers. The demonstration code was written in C++, .NET and Java, and packaged as a full solution on each platform. Users can compile, execute or parse each part of the code for easy investigation.

The program is self-explanatory using a menu-based structure which makes it easy to operate even during the first-time running. To distinguish the difference between the normal and securityinfrastructure-related functions, we divided the demonstration code into two separate projects. The first represents all the basic functions of the IDDK such as login/logout, device management, IDDK and device information retrieval, capturing process, iris recognition, and power management. The second project is written as a useful guide to aid the developers to get accustomed to our security infrastructure easily. As a supplementary tutorial, we described the demonstration code in details as follows.

Before running these demonstrations, please make sure the appropriate driver is loaded, the device is connected to the host system, and the host is able to recognize the device correctly.

4.1 Demonstration with non-cryptographic functions

For Windows XP/Windows 7 platform

- Go to Start →All Programs →IriTech→IDDK 2000 (version)→Demos → Iddk2000Demo.sln to open the C++ demonstration solution.
- Or go to C:\Program Files\Iritech\IDDK 2000 <version> Java\Iddk2000Demo to build Java demonstration code by running *build.bat* and execute *run.bat* to start the demonstration.
- Or go to Start→All Programs→IriTech→IDDK 2000 .NET (version)→Demos → Iddk2000Demo.sln to open the C# demonstration solution.

For Windows CE platform

1. Go to Start→All Programs→IriTech→IDDK 2000 (version) WinCE_ARMV4I →Demos →Iddk2000Demo_ce.sln to open the demonstration code solution.
2. Build the project to produce a console application *Iddk2000Demo_ce.exe*.
3. Copy the application to your Windows CE host device and execute it.

***NOTE**: Since the demonstration project produces a console application, host device has to support console processor (SYSGEN_CMD) and console window (SYSGEN_CONSOLE) to have this application run properly. For more information on how to integrate console processor and console windows into your run-time image, please refer to MSDN guidance (i.e., http://msdn.microsoft.com/en-US/library/gg156507).*

For Linux platform

1. Make sure that g++ and corresponding library (libstdc++) are installed in your system.
2. Go to /opt/IriTech/IDDK-2000-<version>/demo/source (or /opt/IriTech/IDDK-2000-Java<version>/demo/source in Java).
3. Type "**make**" to build the demonstration project into an executable binary *Iddk2000Demo* (or execute the command **./build** to build the demonstration source code and **./run** to execute the demonstration program in Java)**.**

For Mac OS X platform

1. Make sure you have already installed Command Line Tools to build C/C++ source code.
2. Go to /opt/IriTech/IDDK-2000-<version>/demo/source.
3. Type "**sudo make**" to build the demonstration project into an executable binary *Iddk2000Demo***.**

For Android platform, it is strongly recommended for the developers to refer to Java demonstration code on Windows or Linux to attain the adequate instruction.

***Note**: The SDK version is subject to change without notice.*

4.1.1. Main Menu

If the program is compiled and runs successfully, an introduction message will appear.

```
**************************************************************************
Usage:
        This demonstration shows how to use the IDDK 2000 library in a basic
        manner. This demonstration only works with the IriShield devices.

        Copyright(C) 2012 by IriTech, Inc. All rights reserved.
**************************************************************************
Please press ENTER to continue ...
```

User then chooses the communication method (USB or UART) and accesses the main menu.

```
Your attached device can be accessed through:
        1. USB (default)
        2. UART
Enter your choice:
1

Reset on open device:
        1. Yes (default)
        2. No
Enter your choice:
1

Scan devices ... 1 device(s) found !
        1. IriShield Bino #1

Open device IriShield Bino #1 ... done.

MAIN MENU: Select one of the features below
        1. Login
        2. Logout
        3. Device Management
        4. Device & SDK Information
        5. Capturing Process
        6. Iris Recognition
        7. Power Management
        8. Exit
Enter your choice:
```

Main menu includes:

1. Login: Authenticates an Administrator or a Superuser
2. Logout: Leave authenticated session
3. Device Management: Configures and manages device and Administrators
4. Device & SDK Information: Describes details of Device and SDK
5. Capturing Process: Demonstrates a capturing process with customizable inputs
6. Iris Recognition: Enrolls, unenrolls and compares irises
7. Power Management: Keeps a device asleep or awake

Capturing process from this menu is the elaborated one in which the user is asked to provide every input and make decision on processing the output. In other operations that require iris capture, a simple capturing process is used with default inputs (time-based with 3 seconds timeout, normal quality, etc.) and no output processing is required.

4.1.2. Login/Logout

If one or more Administrators or Superusers have been registered in the device, they need to login before performing their granted tasks and must log out promptly when they finish their tasks.

During login, a capturing process is automatically activated to acquire iris of Administrator or Superuser. If device has no Administrator enrolled in advance, please go to menu item "Device Management" and proceed "Enroll Admin" first. The next two illustrations demonstrate the usages of:

- Iddk_Login
- Iddk_Logout

```
*************** Enroll Admin ***************

To enroll as an Administrator, you need to capture your iris first. Capturing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................................................................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 89
        2. Usable area of the right eye: 83
        3. Total score of the left eye: 95
        4. Usable area of the left eye: 86

Enter admin ID (less than 32 characters): iritech

Admin "iritech" just enrolled !

************************************************
```

```
*************** Login ***************

To login the device, you need to capture your iris first. Capturing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................................................................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 90
        2. Usable area of the right eye: 74
        3. Total score of the left eye: 98
        4. Usable area of the left eye: 78

Login as:
        1. Administrator (default)
        2. Superuser
Enter your choice: 1

Enter Administrator ID (less than 32 characters): iritech

Hi "iritech" !!! Login successfully!

Please press any key to continue ...
```

```
*************** Logout ***************

Logout successfully!

Please press any key to continue ...
```

4.1.3. Device Management

Most functionalities in this menu require Administrator's privileges. If the device is registered with at least one Administrator, he/she needs to log in first.

```
*************** Device Management ***************


DEVICE MANAGEMENT: Please select one menu item below
        1. Main Menu
        2. Get Device Configuration
        3. Set Device Configuration
        4. Lock Device
        5. Unlock Device
        6. Reset Certificates
        7. Unenroll Admin
        8. Enroll Admin
        9. Exit
Enter your choice:
```

- Get Device Configuration: Obtains the current configuration of the device

```
**************** Get Device Configuration ****************


Current device configuration:
        + Encryption mode: No encryption
        + Deduplication mode: Disabled
        + Deduplication threshold: 1.100000
        + Authentication threshold: 1.000000
        + Supervised enrollment: Disabled
        + Iris data closure: Disabled
        + Stream images: Yes
        + Stream scale: 1
        + Stream format: IDDK_IFORMAT_MONO_RAW
        + UART Baudrate: 115200
        + Compression Quality: 100
        + UART Hardware Flow Control: Disabled
```

- Set Device Configuration: Changes the current configuration of the device. Administrator's privileges are required. User may skip the changing configuration parameters if he/she is not interested to do so by pressing "Enter" on those parameters.

```
**************** Set Device Configuration ***************


Set encryption mode?
        1. No encryption
        2. Force encryption
Enter your choice (Press ENTER to skip):
Current value: No encryption

Set deduplication mode?
        1. Disable
        2. Enable
Enter your choice (Press ENTER to skip): 2

Set deduplication threshold (Press [ENTER] for current value):
1.1

Set authentication threshold (Press [ENTER] for current value):
1.0

Set supervised enrollment mode?
        1. Disable
        2. Enable
Enter your choice (Press ENTER to skip):
Current value: Disabled

Set iris data closure mode?
        1. Disable
        2. Enable
Enter your choice (Press ENTER to skip):
Current value: Disabled

Do you want to stream images?
        1. Yes
        2. No
Enter your choice (Press ENTER to skip):
Current value: Yes

Set stream scale?
        1. 1
        2. 2
        3. 4
        4. 8
        5. 16
Enter your choice (Press ENTER to skip):
Current value: 1

Enter UART baudrate (not less than 56000, press ENTER for current value):
115200

Set UART hardware flow control?
        1. Disable
        2. Enable
Enter your choice (Press ENTER to skip):
Current value: Disabled

Enter compression quality (Press ENTER to skip):
100
New device configuration was set successfully !
```

- Lock/Unlock Device: Locks or unlocks the device to prevent the device from being accessed. Administrator's privileges are required.

```
*************** Lock Device ***************

Device was locked successfully !

************************************************
```

```
*************** Unlock Device ***************

Device was unlocked successfully !

************************************************
```

- Reset Certificates: Regenerates Camera Public Key and Private Key and/or deletes Customer Public Key

```
*************** Reset Certificates ***************


Delete customer certificate?
        1. No (default)
        2. Yes
Enter your choice: 2

Regenerate camera certificate?
        1. No (default)
        2. Yes
Enter your choice: 2
Camera certificates are reset !
```

- Enroll/Unenroll Admin: Enrolls or unenrolls Administrators. Administrator's privileges are required. During the enrollment, capturing process is automatically activated and image quality is checked carefully.

```
*************** Enroll Admin ***************

To enroll as an Administrator, you need to capture your iris first. Capturing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................................................................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 89
        2. Usable area of the right eye: 83
        3. Total score of the left eye: 95
        4. Usable area of the left eye: 86

Enter admin ID (less than 32 characters): iritech

Admin "iritech" just enrolled !


************************************************
```

```
*************** Logout ***************

Logout successfully!

Please press any key to continue ...
```

Device management demonstrates the usages of:

- Iddk_GetDeviceConfig
- Iddk_SetDeviceConfig
- Iddk_LockDevice
- Iddk_UnlockDevice
- Iddk_ResetCertificates
- Iddk_EnrollAsAdmin
- Iddk_UnenrollAdmin
- Iddk_GetResultQuality

4.1.4. Device & SDK Information

```
*************** Device & SDK Information ***************


SDK version:3.2.5

SDK description: IriShield SDK version 3.2.5.
Copyright(C) 2012 by IriTech, Inc. All rights reserved.

Current SDK configuration:
        Communication type: USB
        Logging: Disabled

Device information:
        Product ID                      0xF005
        Product Name                    IriShield Bino
        Device Model                    Binocular
        Serial Number                   0000000000000000
        Device Properties Flag          0x03010000
        Kernel Version                  3.20
        Device Additional Features:
                - Template generation
                - Template comparison

Please press any key to continue ...
```

This demonstrates the usages of:

- Iddk_GetSdkVersion
- Iddk_GetSdkDescription
- Iddk_GetSdkConfig
- Iddk_GetDeviceInfo.

4.1.5. Capturing Process

The codes demonstrate how developers can easily customize their own capturing process with different capture modes, quality alternatives, indicating LED controls and output formats (see 3.2.2, 3.2.3 and 3.2.4). They can query the image quality including total score and usable area at the end of each capture session as well.

The result images are saved in the same directory as the demonstration assembly. If IriTech image is required, developers can extract the image data from IriTech image.

```
*************** Capturing Process ***************


Init Camera:

        Image width: 640
        Image height: 480

Parameters for capturing process
Capture mode:
        1. IDDK_TIMEBASED (default)
        2. IDDK_FRAMEBASED
Enter your choice:
1

Enter the duration since iris detected (from 1 to 600 seconds, enter for 3):
3

Quality mode:
        1. Normal (default)
        2. High
        3. Very High
Enter your choice:
1

Enable auto led?
        1. Yes (default)
        2. No
Enter your choice:
1

Specify eye subtype to capture:
        1. Left Eye
        2. Right Eye
        3. Both eye(default)
Enter your choice:
3

Put your eyes in front of the camera
        Scanning for eyes.............................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 94
        2. Usable area of the right eye: 83
        3. Total score of the left eye: 100
        4. Usable area of the left eye: 86
```

Various output formats (including raw image, IriTech image and ISO/IEC 19794-6:2005 and 2011) and image kinds (K1, K2, K3 and K7) are available for result images (see 3.5 and 3.6).

```
Do you want to get the result image?
        1. No (default)
        2. Yes
Enter your choice: 2

Select image kind:
        1. Original Image - K1 (default)
        2. VGA Image - K2
        3. Cropped Image - K3
        4. Cropped and Masked Image - K7
Enter your choice:
1

Select image format:
        1. Mono JP2 Image (default)
        2. Mono Raw Image
        3. IriTech JP2 Image
        4. IriTech Raw Image
Enter your choice:
1

Enter compress ratio (enter for default):
100


Get result image ...done
               Saved ./RightEyeImage_1.jp2
               Saved ./LeftEyeImage_1.jp2
```

```
Do you want to get result ISO image:
        1. No (default)
        2. Yes
Enter your choice: 2

Select image kind:
        1. Original Image - K1 (default)
        2. VGA Image - K2
        3. Cropped Image - K3
        4. Cropped and Masked Image - K7
Enter your choice:
1

Select ISO revision:
        1. Iso 2005 (default)
        2. Iso 2011
Enter your choice: 2

Select image format:
        1. Mono Raw
        2. Mono Jpeg2000 (default)
Enter your choice: 2

Specify eye label:
        1. Left Eye
        2. Right Eye
        3. Unknown eye(default)
Enter your choice:
3

Enter compress ratio (enter for default):
100

Get result ISO image ... done.

        Saved ./IsoImage_1.bin
```

Capturing process demonstrates the usages of:

- Iddk_OpenDevice
- Iddk_InitCamera
- Iddk_StartCapture
- Iddk_GetCaptureStatus
- Iddk_GetStreamImage
- Iddk_GetDeviceConfig
- Iddk_GetResultImage
- Iddk_GetImageData
- Iddk_GetResultIsoImage
- Iddk_StopCapture
- Iddk_GetResultQuality
- Iddk_DeinitCamera
- Iddk_CloseDevice

4.1.6. Iris Recognition

The codes instruct developers how to perform iris recognition on IriTech device including template generation, enrollment/unenrollment, and various iris comparison operations. Capturing process with default settings for high usability is automatically activated for operations requiring iris image.

```
MAIN MENU: Select one of the features below
        1. Login
        2. Logout
        3. Device Management
        4. Device & SDK Information
        5. Capturing Process
        6. Iris Recognition
        7. Power Management
        8. Exit
Enter your choice: 6


*************** Iris Recognition ***************


Loading gallery ... done.

IRIS RECOGNITION: Please select one menu item
        1. Main Menu
        2. Get Gallery Information
        3. Enroll Captured Iris
        4. Get Result Template
        5. Enroll Template
        6. Unenroll Templates
        7. Matching
        8. Exit
Enter your choice: 2
```

a. Device has two separate enrollment galleries: one for Administrators only and the other for Superusers and Users. Information in both galleries is open for developers to check its enrollment statuses and capacities.

```
*************** Get Gallery Information ***************

Which gallery do you want to get information?
        1. User (default)
        2. Administrator
Enter your choice: 1

Get gallery information ... done.

Enrollee IDs: ivan peter
Number of enrollees = 2
Number of used slots = 8
Maximum number of slots = 10000
```

b. To generate a template, the user needs to finish the capturing process. Template generated from the captured image will be then retrieved and stored in the same directory as the demonstration assembly. The template can be used later for enrollment and comparison.

```
**************** Get Result Template ****************

Get result template ... done.

        Saved ./ResultTemplate_1.iri.

*************************************************
```

c. To enroll Superuser/User, user first finishes the capturing process. It is strongly recommended to carefully check the image quality before the enrollment to ensure the enrolled images are clear and eyes had been widely open to allow matching to achieve the best accuracy. User may sometimes see the warning message like the one in below:

```
**************** Enroll Captured Iris ****************

There is no captured iris image in the device, you need to capture your iris(es) first. Capturing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................................
        Eyes are detected.
Left eye image is not qualified. Quality of the current captured image:
        1. Total score of right eye: 53
        2. Usable area of right eye: 58
User chose to capture both eyes but only one is qualified for the enrollment.
The captured image(s) is enrollable but not in sufficient quality to warrant the best accuracy.
The subject is recommended to have his/her iris image recaptured with the eye opened widely.
Do you want to proceed anyway?
        1. Yes.
        2. No (default).
Enter your choice:
```

Or even rejection message for images with quality lower than expected:

```
**************** Enroll Captured Iris ****************

There is no captured iris image in the device, you need to capture your iris(es) first. Capturing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................................................................................
        Eyes are detected.
Left eye image is not qualified. Quality of the current captured image:
        1. Total score of right eye: 78
        2. Usable area of right eye: 47

No captured iris image has acceptable quality for the enrollment. Please try to capture your iris(es) again!
```

After being enrolled, enrollee becomes a User. The role can be changed by a Superuser if a supervised enrollment is a must.

```
*************** Enroll Captured Iris ***************

There is no captured iris image in the device, you need to capture your iris(es) first. Capturing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 100
        2. Usable area of the right eye: 84
        3. Total score of the left eye: 95
        4. Usable area of the left eye: 92

Do you want to enroll the current captured iris image(s) into the gallery?
        1. Yes (default)
        2. No
Enter your choice: 1

Enter enrollee ID (less than 32 characters): ivan

Enroll iris ... done.

Do you want to set user role for the current captured iris?
        1. No (default)
        2. Yes
Enter your choice: 2

Which user role?
        1. Superuser (default)
        2. User
Enter your choice: 1
ivan is set to Superuser
```

d. User can also be enrolled by existing templates.

```
*************** Enroll Template ***************


Please specify the template file you want to enroll.
Template file (enter empty path to quit): ./template.iri
Reading file ./template.iri...done.

Enter enrollee ID (less than 32 characters): peter

Enroll template ... done
Commit gallery changes...done.
```

e. There are three types of iris comparisons. Compare11 is for verifying the captured iris image with a known enrollee in the main gallery.

```
*************** Matching ***************


There is no captured iris image in the device, you need to capture your iris(es) first. Cap
turing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.........................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 100
        2. Usable area of the right eye: 86
        3. Total score of the left eye: 88
        4. Usable area of the left eye: 76

Select the kind of comparison:
        1. Compare11 (default)
        2. Compare1N
        3. Compare11WithTemplate
Enter your choice: 1

Enter the enrollee ID (less than 32 characters): peter
Compare the captured iris images with the templates of the enrollee 'peter'...done.

        Compare Distance = 2.000000

No match found!
```

Compare1N is for comparing the captured iris image against the gallery in attempt to establish the identity of the person with this image.

```
**************** Matching ***************


There is no captured iris image in the device, you need to capture your iris(es) first. Cap
turing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 75
        2. Usable area of the right eye: 64
        3. Total score of the left eye: 96
        4. Usable area of the left eye: 92

Select the kind of comparison:
        1. Compare11 (default)
        2. Compare1N
        3. Compare11WithTemplate
Enter your choice: 2

Please specify max distance (Enter or value 0 for no maximum):
Compare the captured iris images with all templates in the gallery...done.

Comparison result:
        Enrollee ID = ivan, Distance = 0.629205
        Enrollee ID = peter, Distance = 1.372803

Matched with 'ivan'!!!
```

Compare11WithTemplate is for comparing an existing iris template with the current captured iris image to verify if the two are matched.

```
**************** Matching ***************


There is no captured iris image in the device, you need to capture your iris(es) first. Cap
turing process starts ...

Init Camera:

        Image width: 640
        Image height: 480

Put your eyes in front of the camera
        Scanning for eyes.....................
        Eyes are detected.
Quality of the current captured images:
        1. Total score of the right eye: 96
        2. Usable area of the right eye: 87
        3. Total score of the left eye: 35
        4. Usable area of the left eye: 88

Select the kind of comparison:
        1. Compare11 (default)
        2. Compare1N
        3. Compare11WithTemplate
Enter your choice: 3

Please specify the template file you want to compare.
Template file (enter empty path to quit): ./template.iri
Reading file ./template.iri...done.
Compare the captured iris image with the specified template...done.
        Compare Distance = 2.000000

No match found!
```

f. User can unenroll a specific enrollee or all enrollees in the gallery.

```
*************** Unenroll Templates ***************

Unenroll all templates in gallery?
        1. No (default)
        2. Yes
Enter your choice: 1

Enter enrollee ID (less than 32 characters): peter

Unenroll templates of enrollee peter ... done.
Commit gallery changes...done.
```

The codes demonstrate the usages of:

- Iddk_LoadGallery
- Iddk_CommitGallery
- Iddk_GetGalleryInfo
- Iddk_GetResultTemplate
- Iddk_EnrollCapture
- Iddk_SetUserRole
- Iddk_EnrollTemplate
- Iddk_UnenrollTemplate
- Iddk_Compare11

- Iddk_Compare1N
- Iddk_Compare11WithTemplate
- Iddk_GetResultQuality

4.1.7. Power Management

The codes demonstrate how to turn the device into Standby or Sleep mode and how to wake it up.

- Iddk_SleepDevice
- Iddk_Recovery

```
*************** Power Management ***************


Select your action?
        1. Standby (default)
        2. Sleep
        3. Wakeup device
Enter your choice: 2
Device is slept with mode: Sleep

Please press any key to continue ...
```

4.2 Demonstration with Security Functionality

This project makes use of the security infrastructure in order to secure data communication between device and the intended recipient of biometric data, e.g., host or remote server.

The project is only available on Windows XP/Windows 7 platform. User can go to Start→(All) Programs →IriTech →IDDK 2000 (version) →Demos →Iddk2000SIDemo.sln to open the demonstration solution.

This demonstration focuses mainly on security functionalities. All iris images and templates are properly signed and encrypted before being transmitted from/to the device. Therefore, we employ a library called SampleCryptoAPI to perform signing, verifying, encryption, and decryption on data. Due to some legal issues, we cannot expose the source code of this library. The library must be used for the demonstration purpose only. Developers have to implement their own cryptography library. IriTech does not hold responsibility for SampleCryptoAPI.

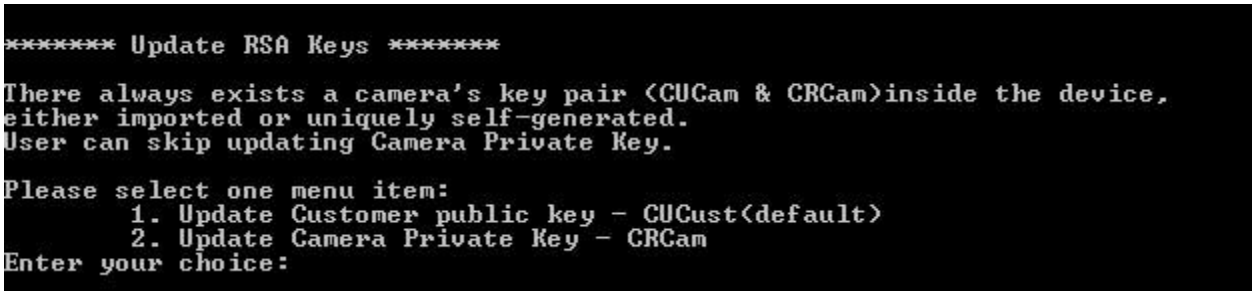
4.2.1. Main Menu

1. Update RSA Keys: Installs proper CUCust and CRCam into the device
2. Capturing Process: Activates a simple capturing process
3. Get Encrypted Capturing Result: Retrieves the captured images and templates
4. Enroll Encrypted Template: Performs enrollment using existing template
5. Unenroll Templates
6. Compare11 with Encrypted Template: Compares the current captured iris image with an existing template
7. Update Nonvolatile Information: Updates user's nonvolatile data (see 3.4.3)

8. Get Encrypted Enrollee Template: Retrieves the enrolled templates of a specific enrollee

4.2.2. Update RSA Keys

Before any further operations, user has to deploy proper keys to the device including **CUCust** and **CRCam** using X509 and PKCS #12 certificates. **CUCust** is a must and has to be imported first. **CRCam** can be skipped if user wishes to use the default Camera keys that are uniquely generated for each device in factory. For demonstration purpose, user can use IDDK 2000 Utility to generate sample certificates for testing.



a. Update Customer Public Key – **CUCust**

When device is new, there is no **CUCust** imported into the device. User may freely update it.


```
******* Update RSA Keys *******

There always exists a camera's key pair (CUCam & CRCam)inside the device,
either imported or uniquely self-generated.
User can skip updating Camera Private Key.

Please select one menu item:
        1. Update Customer public key - CUCust(default)
        2. Update Camera Private Key - CRCam
Enter your choice: 1

Is this the first time device is updated with Customer Public Key?
        1. NO (default)
        2. YES
Enter your choice: 2

Please enter the path to the customer X509 file: C:\Certificate\CUCust.der
Reading certificate... done.

Customer Public Key (CUCust) is updated.
```

However, if a device has been registered with a **CUCust** and user wishes to change it, they must sign the new X509 certificate with their old **CRCust**. In this case, user will be asked for their old PKCS #12 which contains **CRCust**.

```
******* Update RSA Keys *******

There always exists a camera's key pair (CUCam & CRCam)inside the device,
either imported or uniquely self-generated.
User can skip updating Camera Private Key.

Please select one menu item:
        1. Update Customer public key - CUCust(default)
        2. Update Camera Private Key - CRCam
Enter your choice: 1

Is this the first time device is updated with Customer Public Key?
        1. NO (default)
        2. YES
Enter your choice: 1

Please enter the path to the customer X509 file: C:\Certificate\CUCust.der
Reading certificate... done.

Please enter the path to the old Customer PFX(PKCS#12): C:\Certificate\CRCust.pf
x
Reading ... done.
Password to decode PFX(PKCS#12): 123456

Customer Public Key is updated.
```

b. Update Camera Private Key – **CRCam**

CRCam must be put in a PKCS #12 certificate. This PKCS #12 has to be signed by the current in-use **CRCust** before being sent to device.

```
******** Update RSA Keys ********

There always exists a camera's key pair (CUCam & CRCam)inside the device,
either imported or uniquely self-generated.
User can skip updating Camera Private Key.

Please select one menu item:
       1. Update Customer public key - CUCust(default)
       2. Update Camera Private Key - CRCam
Enter your choice: 2

Please enter the path to camera PKCS#12 file: C:\Certificate\CRCam.pfx
Reading certificate... done.
Password to decode PKCS#12: 123456

Please enter the path to Customer PKCS#12 for data signature:C:\Certificate\CRCu
st.pfx
Reading certificate... done.
Password to decode Customer PKCS#12: 123456

Camera Private Key (CRCam) is updated.
```

The codes demonstrate the usages of:

- Iddk_UpdateCustomerCertificate
- Iddk_GetCameraCertificate
- Iddk_UpdateCameraCertificate

4.2.3. Update Non-volatile Information

Please refer to 3.4.3.

```
******** Update Nonvolatile Information ********


Importing Camera Public Key ... done
Please enter the path to Customer PKCS#12: C:\Certificate\CRCust.pfx
Password to decode Customer PKCS#12: 123456
Reading certificate ... done.

Please enter nonvolatile information (empty string will clear nonvolatile data):
 iritech

Updating nonvolatile information ... done.
```

The codes demonstrate the usages of:

- Iddk_UpdateNonvolatileInfo

4.2.4. Get Encrypted Capturing Result

The codes demonstrate the usages of the following to obtain the captured images and templates.

- Iddk_GetEncryptedResultImage
- Iddk_GetEncryptedResultTemplate
- Iddk_GetEncryptedResultISOImage

The encrypted data are stored in the same directory as the demonstration assembly. User needs to provide **CRCust** in order to decrypt and verify the data. The plain data is saved in the same directory.

```
******* Get Encrypted Capturing Result *******


Please select one menu item:
        1. Get Encrypted Result Image (default)
        2. Get Encrypted Result Template
        3. Get Encrypted Result ISO Image
Enter your choice: 1

Select image kind:
        1. Original Image - K1 (default)
        2. VGA Image - K2
        3. Cropped Image - K3
        4. Cropped and Masked Image - K7
Enter your choice: 1

Select image format:
        1. JP2 Image (default)
        2. Raw Image
        3. IriTech JP2 Image
        4. IriTech Raw Image
Enter your choice: 1

Enter compress ratio (enter for default):
100

Please enter volatile data: 2013-5-16


Get encrypted result image...done
                Both left & right images are qualified.

        Saved d:\DeviceSWTeam\IDDK2000\SampleCodes\Iddk2000BinoSIDemo\Bin/Encryp
tedEyeImage_Right.bin

        Saved d:\DeviceSWTeam\IDDK2000\SampleCodes\Iddk2000BinoSIDemo\Bin/Encryp
tedEyeImage_Left.bin

Importing Camera Public Key ... done
Please enter the path to Customer PKCS#12: C:\Certificate\CRCust.pfx
Password to decode Customer PKCS#12: 123456
Reading certificate ... done.
Decrypting ... done.

        Volatile Data: 2013-5-16

        Non Volatile Data: iritech

        Saved ./RightEyeImage.jp2

Importing Camera Public Key ... done
Please enter the path to Customer PKCS#12: C:\Certificate\CRCust.pfx
Password to decode Customer PKCS#12: 123456
Reading certificate ... done.
Decrypting ... done.

        Volatile Data: 2013-5-16

        Non Volatile Data: iritech

        Saved ./LeftEyeImage.jp2
```

```
******* Get Encrypted Capturing Result *******


Please select one menu item:
        1. Get Encrypted Result Image (default)
        2. Get Encrypted Result Template
        3. Get Encrypted Result ISO Image
Enter your choice: 2

Please enter volatile data: 2013-5-16

Get result template ... done.

        Saved ./EncryptedResultTemplate.bin

Importing Camera Public Key ... done
Please enter the path to Customer PKCS#12: C:\Certificate\CRCust.pfx
Password to decode Customer PKCS#12: 123456
Reading certificate ... done.
Decrypting ... done.

        Volatile Data: 2013-5-16

        Non Volatile Data: iritech

        Saved ./DecryptedResultTemplate.iri.
```

4.2.5. Enroll Encrypted Template

The codes demonstrate how to enroll an existing template into a device. The template needs to be signed and encrypted properly by **CRCust** and **CUCam** before being sent to the device.

```
******* Enroll Encrypted Template *******


Importing Camera Public Key ... done
Please enter the path to Customer PKCS#12: C:\Certificate\CRCust.pfx
Password to decode Customer PKCS#12: 123456
Reading certificate ... done.

Please specify the template files you want to enroll.
Template file (enter empty path to quit): C:\Templates\Test.iri
Reading file C:\Templates\Test.iri...done.

Enter enrollee ID (less than 32 characters): peter

Enroll templates ... done

Commit gallery changes...done.
```

The following API is used in the demonstration:

- **Iddk_EnrollEncryptedTemplate**

4.2.6. Compare With Input Encrypted Template

The codes demonstrate how to compare the captured image with an existing template. The template needs to be signed and encrypted properly by **CRCust** and **CUCam** before being sent to the device.

```
******* Compare11 with Encrypted Template *******


Importing Camera Public Key ... done
Please enter the path to Customer PKCS#12: C:\Certificate\CRCust.pfx
Password to decode Customer PKCS#12: 123456
Reading certificate ... done.

Please enter template file to be compared: C:\Templates\Test.iri
Reading file C:\Templates\Test.iri...done.

        Compare Distance = 0.649764
Matched!!!
```

The codes demonstrate the usages of:

- **Iddk_Compare11WithEncryptedTemplate**

4.2.7. Get Encrypted Enrollee Template

The codes demonstrate how to get encrypted templates of a specific enrollee from the device. The encrypted template will be stored in the same directory as the demonstration assembly. User must provide **CRCust** in order to decrypt and verify the data. The plain template is saved in the same directory.

```
******* Get Encrypted Enrollee Template *******


Please enter volatile data: 2013-5-16

Enter enrollee ID (less than 32 characters): peter

Get enrollee template ... done.

        Saved ./EncryptedEnrolleeTemplate.bin

Importing Camera Public Key ... done
Please enter the path to Customer PKCS#12: C:\Certificate\CRCust.pfx
Password to decode Customer PKCS#12: 123456
Reading certificate ... done.
Decrypting ... done.

        Volatile Data: 2013-5-16

        Non Volatile Data: iritech

        Saved ./DecryptedResultTemplate.iri.
```

4.3 IDDK 2000 Utility

In order to aid users in programming IriTech device with ease, the IDDK 2000 Utility is readily provided in the IDDK package. Currently, this utility only runs on Windows 7/Windows XP.

- For Windows XP/Windows 7 package, users can look for the utility from Start menu → (All) Programs → Iritech → IDDK 2000 <version> → Utilities → IDDK 2000 Utility.

- For Windows CE package, users can look for the utility from Start menu → (All) Programs → Iritech → IDDK 2000– WinCE_ARM4VI → Utilities → IDDK 2000 Utility, and run it on the development PC which installs Windows XP/Windows 7 OS.

4.3.1. Device Info

This utility helps users to quickly get the detailed information of the current device.

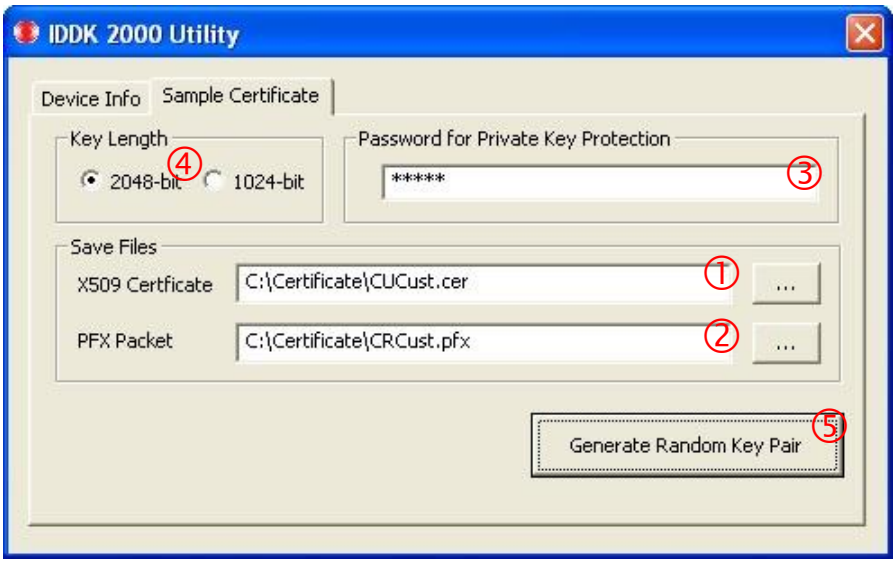
The following steps should be followed to use the utility.

- Connect device to host
- Click "**Refresh**" ① and choose the device from the combo box ②
- Lastly, click "**Get Device Info**"③

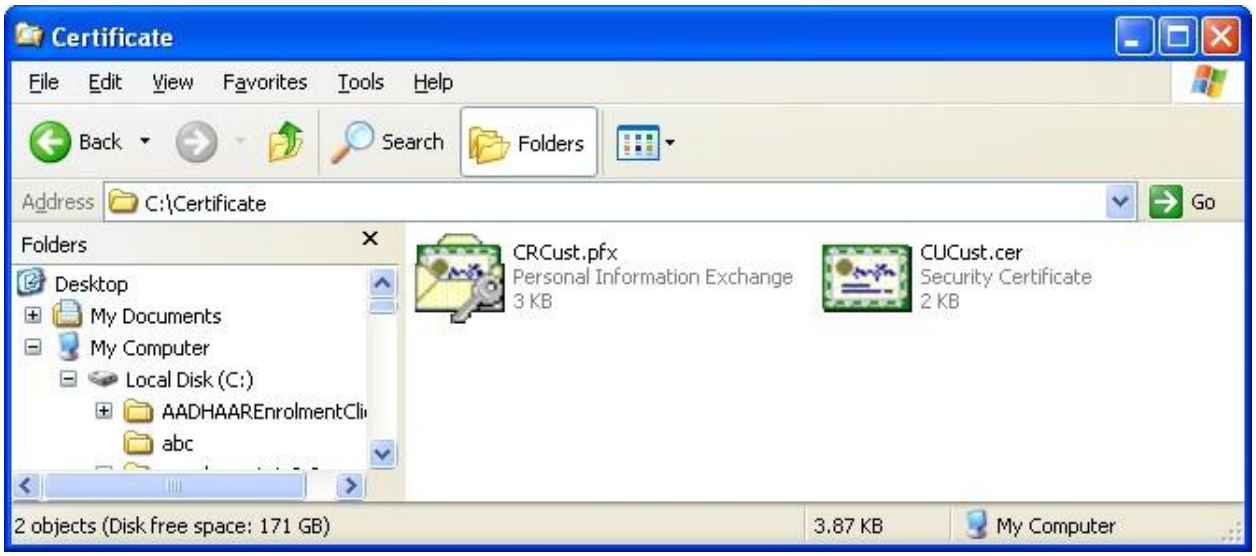
4.3.2. Sample Certificate

In order to help users approach the security functionalities with more ease, this tool generates valid X509 and PKCS #12 certificates that can be used in the cryptographic functions. An X509 certificate encodes the public key and other personal information. PKCS #12 (PFX) packet encodes both public information and a private key protected by password.

- In the box "**X509 Certificate**" under "**Save Files**" section, specify the location to store the generated X509 certificate ①
- In the box "**PFX Packet**" under "**Save Files**" section, specify the location to store the generated PKCS #12 packet ②
- Type password to protect the private key in "**Password for Private Key Protection**" section ③
- Select the key length for the generated key pair in the "**Key Length**" section ④
- Lastly, click "**Generate Random Key Pair**" ⑤



A random RSA key pair will be generated in the specified location.



Note:

- *The certificates generated by this tool are self-signed certificates.*
- *The certificates generated by this tool are only for sample. IriTech does not hold any responsibilities related to security issues if users utilize these certificates for business.*
- *Currently, this utility is not supported on Linux platform.*

5. Software Warnings and Precautions

IriShield has been tested successfully with robustness, reliability, speed, and high accuracy. However, any intentional or unintentional wrong usages of the device and the SDK will lead to unexpected results or even mal-function states. Users need to be aware of the below phenomena and follow the suitable instructions to overcome the problems.

5.1 Red-Eye Effect Cautions

Similar to flash color photography, capturing iris images in dark environments suffers from "red-eye effect" which causes the pupil's appearance to be too bright, leading to potential matching errors. To avoid such effect, we recommend for users to have their iris images captured in sufficient ambient light. Particularly during enrollment, system operators are strongly recommended to reject red-eye (brightpupil) images.

5.2 Enhance Matching Accuracy

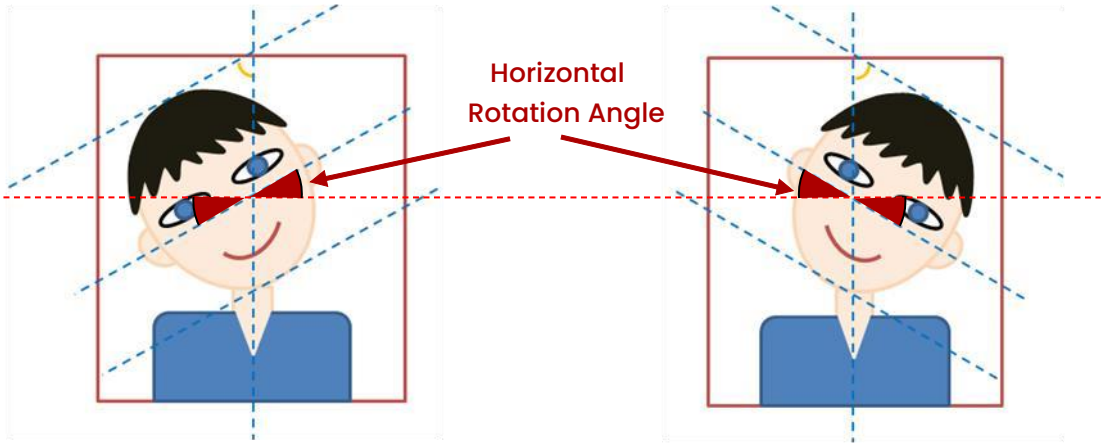
To prevent faulty matching, the following guidelines should be followed carefully by developers. Please be warned that the threshold numbers given in this section are only for reference. Developers should make their own decisions to allow a balance between accuracy and usability based on their iris database.

5.2.1. Enrollment

- *Image quality*: The template used for enrollment is very important for the successive operations such as identifying and verifying. Therefore, enrolled templates must be generated from highlyqualified images in which iris texture is clearly emerged with high sharpness and wide-open eyes without much occlusion. Before enrollment, developers should examine the quality scores of the captured images. If both total score and usable area are greater than 70 (>70), the captured image is good for enrollment. If either total score or usable area is less than or equal to 50 (<=50), the image should be rejected.

- *Pupil dilation*: Since iris muscles control the size of the pupil, iris pattern shown in a captured iris image deforms significantly during pupil dilation/constriction due to too bright or too dark environment. Comparison of an iris image with highly constricted pupil and that of the same eye with dilated pupil may produce a very big distance as if they are from two different eyes. Therefore, each subject is recommended to enroll multiple iris templates captured in different lighting conditions: bright, normal, and dark to prevent false negative results.

- *Horizontal rotation (head roll)*: IriTech device supports a specific horizontal rotation range. During capturing process, users should make efforts to provide eye images as horizontal as possible. For enrollment, horizontal rotation angle of eye postures should not exceed ±10 degrees.



- *Deduplication*: Before enrolling a template, please make sure that very template does not match with other enrollees'. Deduplication can be performed automatically by device or manually by developers. Deduplication threshold should be carefully adjusted in a way that it is high enough to prevent all duplications (minimizing false negative rate), but not too high to make the enrollment frustrating. The choice of deduplication threshold is a tradeoff between accuracy and usability. With a good quality enrollment database, the recommended deduplication threshold value is 1.10.

5.2.2. Matching

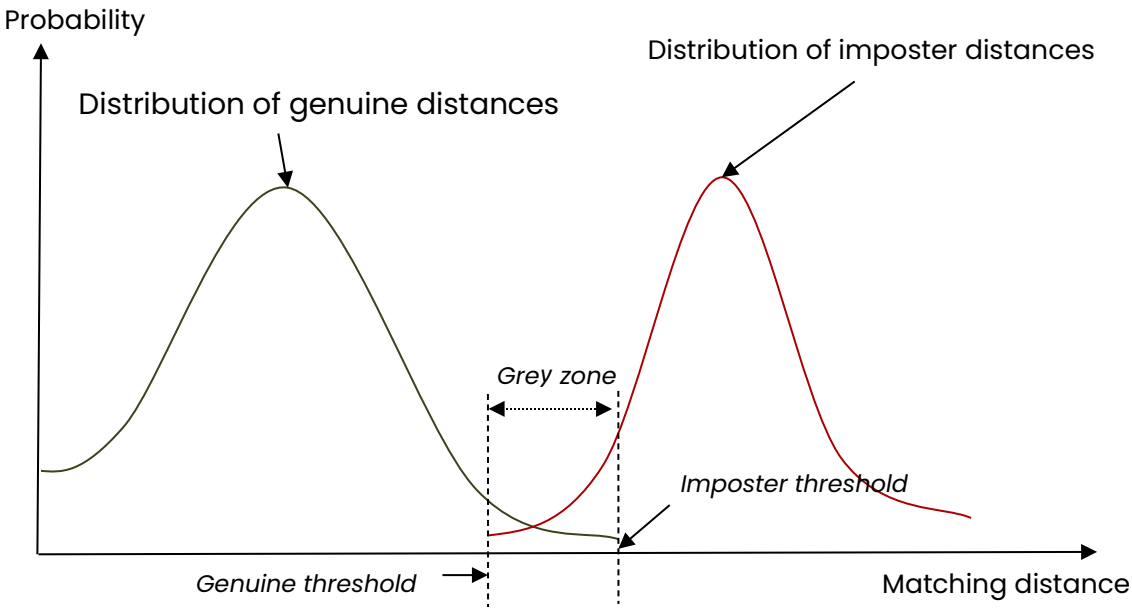
- *Image quality*: Images captured during identification stage are not required to be of high quality as in enrollment stage, but they should be still decent. If either `totalScore` or `usableArea` is less than or equal to 30 (<=30), the image should be rejected.

- *Horizontal rotation*: During capturing process, users should make efforts to provide eye images as horizontal as possible. For comparison, horizontal rotation angle of eye postures should not exceed ±20 degrees.

- *Matching decision*: The device does not make any matching decision. Instead, it returns the comparison distances for 1:1 and 1:N comparisons and lets the applications make decision. Different from deduplication, matching concerns more about false positives to reject unauthorized accesses. Matching threshold needs to be carefully chosen. It should be low enough to accept only true matches (avoiding false positives), but not too low to cause the user inconvenience. We recommend developers to:
    - Investigate the distributions of genuine and imposter distances on their sample iris database. There is usually an overlap between the two distributions.
    - Select two thresholds: Genuine threshold and imposter threshold. Ideally, genuine threshold is selected in a way for all the genuine matches to have distances lower than or equal to it. Imposter threshold is selected in a way for all the imposter matches to have distances greater than it. However,

depending on the accuracy of each biometric application, the thresholds can be relaxed to the level that reduces false acceptance rate as much as possible. Distances that are greater than genuine threshold and less than imposter threshold are in the *grey zone*. With a good quality enrollment database, the two recommended thresholds are 1.00 and 1.10, respectively.

- Get the smallest distance among those returned from 1:1 and 1:N comparisons. If the distance is less than or equal to genuine threshold, a positive match is claimed. If the distance is greater than imposter threshold, the query image is from an imposter. If the distance is in the grey zone, there is some uncertainty due to, e.g., low image quality, occlusion, or pupil dilation. User should be asked to recapture his/her iris image with the eyes wide open and do the matching again.



5.3 Device IO Failure Cautions

Due to mistakes made with using the device or unexpected problems from the system itself, the device may become out of order or be brought into the malfunction states. Such issues cause device I/O failure on successive operation. There may be several reasons as to why the device fails such as version incompatibility, unstable USB links, and user's mistaken operation.

5.3.1. Hardware

The major issue having been reported up to date is power insufficiency or power glitch which can happen without customer's notice during its long duration of operation. Power insufficiency not only causes device to behave unexpectedly, but also harms USB hardware component of the device in which it halts the communication between the host and device.

To prevent such failure, users should carefully refer to the hardware specifications for electronic details. In general, the following tips should be considered with care:

- The USB cables should not be plugged into the computer via the USB hub, but plugged directly into the computer.
- The USB cable's length must not be too long. Users are encouraged to use the cables provided by IriTech.
- The system must not be in a power save or standby mode that causes the reduction of power supply to the device.
- Users must carefully check the power supply to device to make sure it is sufficient to do the work properly.

5.3.2. Software

IO failures occur when application tries to access an invalid device handle of which may occur when the device is hard reset (i.e., due to power glitch or insufficiency). Especially after the device has slept, since it is clock-lowered, any access to it before it awakes will result in IO failures. IO failures are also due to version incompatibility among the device, its driver, and the library.

When the failure occurs, synchronization between the device and the host is lost. Data and underlying USB IO buffers may be corrupted.

Developers can limit these failures by:

- Avoiding cases that cause the application to crash or kill the application in the middle of device's communication.
- Avoiding improper thread termination while it is accessing to the device (a function in this library is called but not yet finished).
- Performing a proper wait on time-critical functions, such as ***Iddk_Sleep***, ***Iddk_Recovery***, or waiting between two captures. Do not flood USB link with request for a long time.
- Avoiding virtual machines such as VM Ware or Virtual Box. Virtual USB controller does not work properly all the time.
- Assigning the device handle to NULL as soon as it is closed.
- Contacting IriTech to be advised about the version compatibility.
- Waking device up before accessing it.
- Listening to PNP events to have appropriate action when device is removed from host.

To recover from a failure, unplugging and plugging the device again is the strongest and always-work solution. However, tender approaches also usually work. First, the host should wait for around 10s and use recovery methods provided by IDDK 2000 once or multiple times with increased severity levels.

- Cancel all pending IOs and clear internal buffers. When the failure happens, the requested command execution is corrupted with invalid data. All the IO requests made for this command should be cancelled and invalid data should be discarded.

- Reset USB pipes. To perform a command, there are several IN/OUT data transferred between the host and the device. For example, the host first sends the request and waits for device's response to see whether it can execute that command. If yes, the device changes its state to wait for input data. After receiving sufficient data, the device executes the command and sends the result to the host. In each step, the device has to wait for data or receipt acknowledgment from the host before it can proceed to the next step. If it cannot receive the valid data after some time (around 10s), it has to halt data endpoints, drop the command, and wait for another command. Resetting USB pipes clears these halting conditions.
- Reset a USB Port. If a reset on USB pipes does not help, try resetting a USB port to have the USB device reset and re-enumerated.
- Power cycle a USB Port. If a reset on USB port does not help, try to power cycle a USB port. The device will act as if it is being unplugged and then plugged back in, but it is not electrically disconnected (firmware is not reloaded). The handle to open the device will become invalid.
- Soft reset the device. It is not a POR (Power-On Reset). Firmware is not reloaded. The processor only jumps to the execution of boot routine. Current states and data of the device are lost as if the device were powered off and then on again. Host receives device removal and arrival notifications when boot routine finishes.

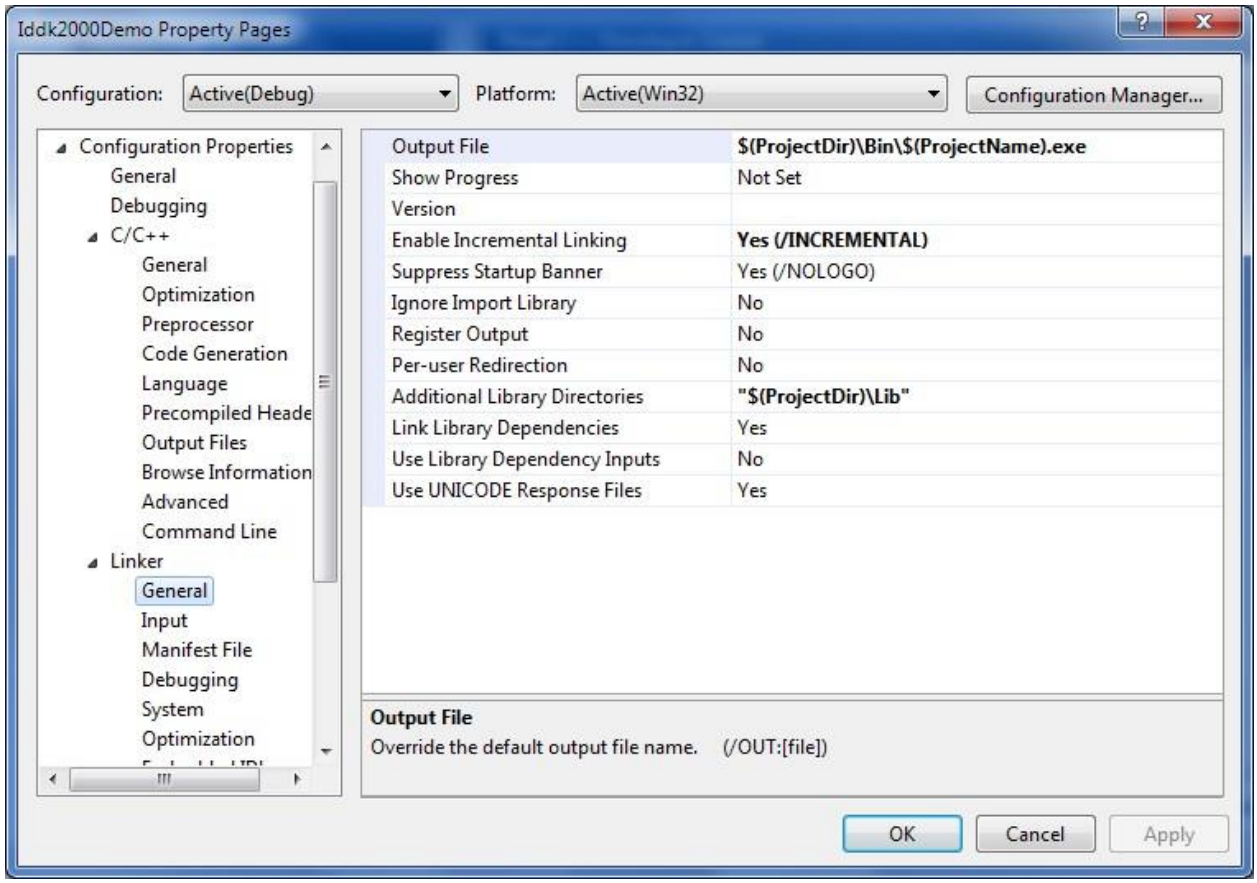
6. Troubleshooting and FAQ

1. **Question:**

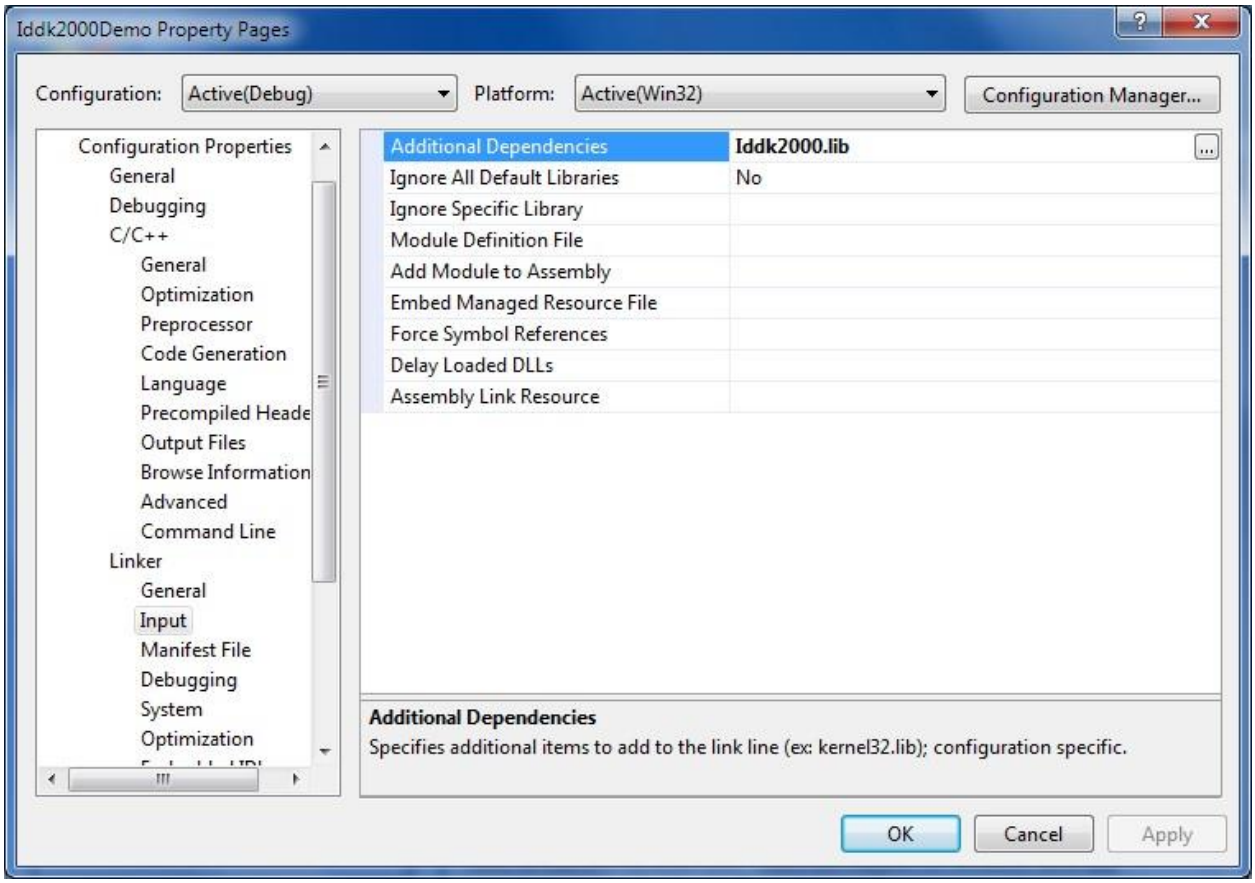
   How do I use Visual Studio 2008 to configure a project?

Answer:

1. Open project under Visual Studio 2008.
2. Open project properties: Select the project, go to Project menu and choose Properties.
3. In the Property Pages window, select "Configuration Properties->Linker->General->Additional Library Directories," then specify the location of library file *Iddk2000.lib.*



4. Select “Configuration Properties->Linker->Input->Additional Dependencies” and configure *Iddk2000.lib* as “Additional Dependencies.”



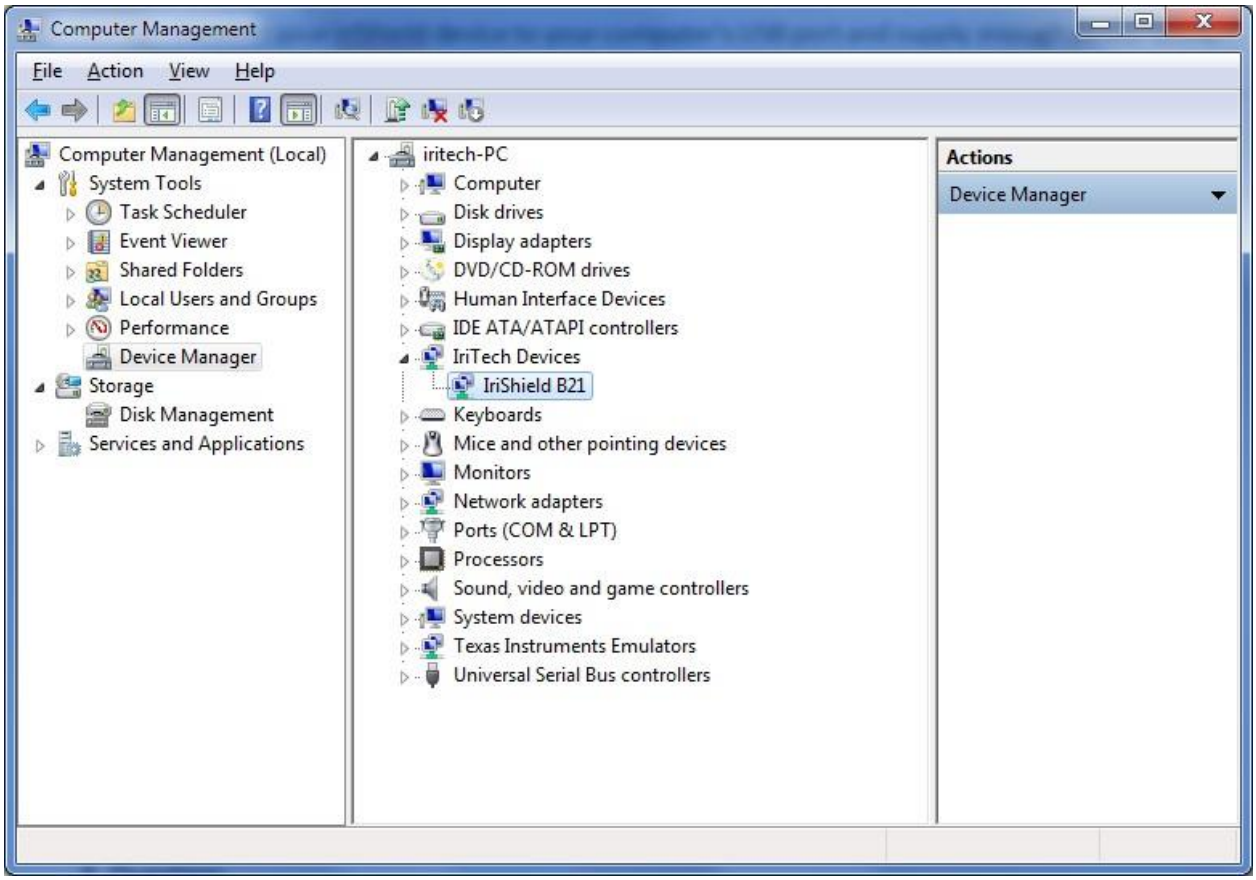
5. Rebuild the project.

2. Question:

How do I confirm that the device is correctly plugged into the computer?

Answer:

1. Connect your IriShield device to your computer's USB port and supply enough power to the device.
2. Right-click on "My Computer" from your desktop or Windows Explorer, and select "Manage."
3. Select "Device Manager" in the left pane of the Computer Management window.
4. Locate and expand "IriTech Devices" in the right pane.
5. If you see the device's name, the device has been plugged into the computer correctly and ready for use.



3. Question:

What does the "Device not found" error mean?

Answer:

The message indicates that the program cannot find any appropriate device in the host system. First of all, users need to check the cables and power supply and make sure the device has been plugged into the host computer properly.

Secondly, IriTech provides its customers with various iris-recognition devices, each of which shares several major functionalities with the others yet carries its own characteristics. The SDKs for different product lines are different from each other.

If a customer has purchased multiple products from IriTech and installed them in the same host system, he/she should pay attention to use the correct SDK to control each product. If the device attached to the host is a binocular device, but using an IriShield Mono SDK to control it will return "Device not found."

7. Legal Notice

7.1 Warranty Terms

IriTech	IriCAMM / IriTerminal / IriMobile / IriMagic/ IriShield / IriHerald
Warranty Length	1 Year Limited: Parts & Labor, Mail in or Carry in
Hardware Technical Support	1 Year
Software Support	90 days
Website	http://www.iritech.com
Service Phone Number	+1 703 877 2135 (United States) +82 2 872 3812 (Korea)

THIS AGREEMENT CONTAINS A MANDATORY AND BINDING ARBITRATION PROVISION IN WHICH YOU AND IRITECH AGREE TO RESOLVE ANY DISPUTES BETWEEN YOU AND IRITECH BY BINDING ARBITRATION. PLEASE SEE SECTION 10 BELOW.

This Agreement ("Agreement") is between the original purchaser ("You") and IriTech, Inc. ("IRITECH") and applies to IRITECH products ("Products") and services purchased by you from IRITECH or affiliates or an IRITECH authorized reseller ("Reseller"). This limited warranty extends only to You, the original purchaser, and is not transferable to anyone who subsequently purchases, leases, or otherwise obtains the Product from You.

The term of this Limited Warranty (the "Limited Warranty Period") is identified in the reference table included with this Agreement (Warranty Reference Table"). The Limited Warranty Period commences on the date of purchase by You. Your original purchase invoice (sales receipt) showing the date of purchase of the Product is your proof of the date of purchase.

Product Limited Warranty. IRITECH warrants that its Products will be free from defects in materials and workmanship for the Limited Warranty Period. During the Limited Warranty Period, IRITECH will, as its option; (i) provide replacement parts necessary to repair the Product; (ii) repair the Product or replace it with a comparable product; or (iii) refund the amount You paid for the Product, LESS DEPRECIATION, upon its return. Replacement parts and Products will be new or serviceably used, comparable in function and performance to the original part or Product and warranted for the remainder of the original warranty period or, if longer, 90 days after they are shipped to you.

Technical Support. During the Limited Warranty Period, IRITECH will provide product technical support by e-mail. The scope of technical support consists of helping you diagnose and resolve problems with defects in Products covered by this Agreement,

and, for Camera Products, reinstalling the factory-installed operating system and software to restore it to the original factory configuration. IRITECH may provide technical support via on-line and other methods. IRITECH may change the means through which it provides technical support at any time.

THIS LIMITED WARRANTY DOES NOT COVER MISUSE OR MINOR IMPERFECTIONS IN UNITS THAT MEET DESIGN SPECIFICATIONS OR IMPERFECTIONS THAT DO NOT MATERIALLY ALTER FUNCTIONALITY.

THIS LIMITED WARRANTY DOES NOT COVER AND IRITECH IS NOT RESPONSIBLE FOR:

- DAMAGES CAUSED BY MISUSE, ABUSE, ACCIDENTS, FIRE, THEFT, DISAPPEARANCE, MISPLACEMENT, POWER SURGES, VIRUSES, RECKLESS, WILLFUL, OR INTENTIONAL CONDUCT.
- DAMAGES CAUSED BY SERVICING NOT AUTHORIZED BY IRITECH.
- DAMAGES CAUSED BY USAGE THAT IS NOT IN ACCORDANCE WITH PRODUCT INSTRUCTIONS.
- DAMAGES CAUSED BY FAILURE TO FOLLOW THE PRODUCT INSTRUCTIONS OR FAILURE TO PERFORM PREVENTIVE MAINTENANCE.
- DAMAGES CAUSED BY THE COMBINATION OF IRITECH PRODUCTS WITH OTHER NON-IRITECH BRANDED PRODUCTS, ACCESSORIES, PARTS OR COMPONENTS.
- SOFTWARE, INCLUDING THE OPERATING SYSTEM AND SOFTWARE ADDED TO YOUR PRODUCT THROUGH OUR FACTORY-INTEGRATION SYSTEM, THIRD-PARTY SOFTWARE, OR THE RELOADING OF SOFTWARE.
- ANY EQUIPMENT OR COMPONENTS THAT WERE NOT INCLUDED IN YOUR PRODUCT AS ORIGINALLY SOLD TO YOU.
- NORMAL WEAR AND TEAR.
- COSMETIC DAMAGE THAT DOES NOT AFFECT FUNCTIONALITY.
- PRODUCTS WHERE THE IRITECH SERIAL NUMBER IS MISSING, ALTERED OR DEFACED.

1. **Instructions for Warranty Service.** For specific instructions on how to obtain warranty service for your product, please refer to the Warranty Reference Table contained in this Agreement and go to: http://www.iritech.com.

To obtain warranty service:

- You must assist IRITECH in diagnosing issues with your product and follow IRITECH's warranty processes.

- You must obtain warranty service from IRITECH or an authorized service center specified by IRITECH. IRIECH will not reimburse you for service performed by others.

- You may be required to deliver and retrieve your product to and from IRITECH or an authorized service facility specified by IRITECH at your expense. When sending a product to IRITECH or the authorized service facility specified by IRITECH, you must deliver the product, freight prepaid, in either its original packaging or packaging affording an equal degree of protection. You are responsible for properly packaging your product, paying all shipping costs, loss or damage to the product during shipping, and any other taxes, fees or charges associated with transporting the product to an authorized IRITECH service facility. **YOU ARE RESPONSIBLE FOR ANY DAMAGE TO YOUR IRITECH PRODUCT DURING SHIPMENT TO US.**

- Before providing your product to IRITECH for service, remove any confidential, proprietary or personal information.

- If IRITECH asks you to return defective parts or products, you must do so within 7 days after you receive the replacement parts or products. IRITECH will charge you for replacement parts or products if you fail to do so.

IT IS YOUR RESPONSIBILITY TO BACK UP THE CONTENTS OR DATA RECORDED ON THE DEVICE BEFORE SERVICES ARE PERFORMED AND REMOVE ANY DATA FROM PARTS OR PRODUCTS RETURNED TO IRITECH. It is possible that the contents recorded will be lost or reformatted in the course of service and IRITECH will not be responsible for any damage to or loss of any programs, data, or other information stored on any media or any part of any product serviced. IF DURING THE REPAIR OF THE PRODUCT THE CONTENTS OF THE DEVICE ARE ALTERED, DELETED, OR IN ANY WAY MODIFIED, IRITECH IS NOT RESPONSIBLE FOR ANY LOSS OF YOUR DATA

WHATSOEVER. YOUR PRODUCT WILL BE RETURNED TO YOU CONFIGURED AS ORIGINALLY PURCHASED (SUBJECT TO AVAILABILITY OF SOFTWARE).

2. **Implied Warranties. EXCEPT TO THE EXTENT PROHIBITED BY APPLICABLE LAW, ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE IS LIMITED IN DURATION TO THE DURATION OF THIS WARRANTY.**

Commercial Purchasers: IRITECH extends the above limited warranty to purchasers of Products for industrial, commercial and business use upon the same terms and conditions and exclusions applicable to consumer purchasers. **HOWEVER, WITH RESPECT**

TO COMMERCIAL PURCHASERS, ALL IMPLIED WARRANTIES, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE EXCLUDED AND DISCLAIMED.

3. **Limitation of Liability. IRITECH SHALL NOT BE LIABLE FOR ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR BREACH OF ANY EXPRESS OR IMPLIED WARRANTY.** IRITECH'S AND YOUR MAXIMUM LIABILITY TO THE OTHER IS LIMITED TO PURCHASE PRICE YOU PAID FOR PRODUCTS OR SERVICES PLUS INTEREST ALLOWED BY LAW. NEITHER YOU NOR IRITECH IS LIABLE TO THE OTHER IF YOU OR IT ARE UNABLE TO PERFORM DUE TO EVENTS YOU OR IT ARE NOT ABLE TO CONTROL, SUCH AS ACTS OF GOD OR FOR VIRUSES, PROPERTY DAMAGE, LOSS OF USE, INTERRUPTION OF BUSINESS, LOST PROFITS, LOST DATA OR OTHER CONSEQUENTIAL, PUNITIVE OR SPECIAL DAMAGES, HOWSOEVER CAUSED, WHETHER FOR BREACH OF WARRANTY, CONTRACT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY OR OTHERWISE.

Some states or jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, or allow limitations on how long an implied warranty lasts, so the above limitations may not apply to you. This Limited Warranty gives you specific legal rights and you may have other rights which vary from state to state or jurisdiction to jurisdiction.

4. **Binding Arbitration.** THIS AGREENET PROVIDES THAT ALL DISPUTES BETWEEN YOU AND IRITECH WILL BE RESOLVED BY MANDATORY AND BINDING ARBITRATION TO THE FULLEST EXTENT PROVIDED BY LAW.
   YOU THUS GIVE UP YOUR RIGHT TO GO TO COURT TO ASSERT OR DEFEND YOUR RIGHTS UNDER THIS AGREEMENT (EXCEPT FOR MATTERS THA MAY BE TAKEN TO SMALL CLAIMS COURT AS FURTHER SET FORTH BELOW IN THIS SECTION 7).

To the fullest extent provided by law, and except as otherwise provided below, You and IRITECH agree that any Dispute between You and IRITECH will be resolved exclusively and finally by binding arbitration administered by the American Arbitration Association (AAA) and conducted in accordance with AAA's Supplementary Procedures for Consumer-Related Disputes of the Commercial Arbitration rules and the Consumer Due Process Protocol. YOUR RIGHTS WILL THEREFORE BE DETERMINED BY A NEUTRAL ARBITRATOR AND NOT A JUDGE OR JURY. You and IRITECH will agree on another arbitration forum if AAA ceases operations.

The arbitration will be conducted before a single arbitrator, and will be limited solely to the Dispute between You and IRITECH. Arbitration is a process whereby a dispute is submitted to an arbitrator, for a final and binding determination, known as the award. The arbitrator is an individual, similar to a judge, who reviews and weighs evidence provided by both parties, and renders an award enforceable in court. Arbitrator decisions are as enforceable as any court order and are subject to VERY LIMITED REVIEW BY A COURT. YOU ACKNOWLEDGE THAT, BY WAY OF THIS AGREEMENT, YOU AND IRITECH WAIVE ALL RIGHTS TO A JURY TRIAL.

The arbitration, or any portion of it, will not be consolidated with any other arbitration and will not be conducted on a class-wide or class action basis. The arbitration shall be held at any reasonable location near your residence by submission of documents, by telephone, online or in person whichever method of presentation You choose. Under the AAA Supplementary Procedures for Consumer-Related Disputes and Consumer Due Process Protocol, You retain the right to seek relief in a small claims court for Disputes within the scope of the small claims court's jurisdiction. If you prevail in the arbitration of any Dispute with IRITECH, IRITECH will reimburse You for any fees You paid to AAA in connection with the arbitration.

ANY DECISION RENDERED IN SUCH ARBITRATION PROCEEDINGS WILL BE FINAL AND BINDING ON THE PARTIES, AND JUDGMENT MAY BE ENTERED THEREON IN ANY COURT OF COMPETENT JURISDICTION.

Should either party bring a Dispute in a forum other than AAA, the arbitrator may award the other party its reasonable costs and expenses, including legal fees, incurred in staying or dismissing such other proceedings or in otherwise enforcing compliance with this dispute resolution provision. **YOU UNDERSTAND THAT, IN THE ABSENCE OF THIS PROVISION, YOU WOULD HAVE HAD A RIGHT TO LITIGATE DISPUTES THROUGH A COURT**, INCLUDING THE RIGHT, IF ANY AND SUBJECT TO THE RULES OF YOUR JURISDICTION, TO LITIGATE CLAIMS ON A CLASS-WIDE OR CLASS-ACTION BASIS, **AND THAT YOU HAVE EXPRESSLY AND KNOWINGLY WAIVED THOSE RIGHTS AND AGREED TO RESOLVE ANY DISPUTES THROUGH BINDING ARBITRATION IN ACCORDANCE WITH THE PROVISIONS OF THIS PARAGRAPH.**

This arbitration provision shall be governed by the Federal Arbitration Act, 9 U.S.C. Section 1, *et seq*. For the purposes of this provision, the term "Dispute" means any dispute, controversy, or claim arising out of or relating to (i) this Agreement, its interpretation, or the breach, termination, applicability or validity thereof, (ii) the related order for, purchase, delivery, receipt or use of any product or service from IRITECH, or (iii) any other dispute arising out of or relating to the relationship between you and IRITECH; the term "IRITECH" means IRITECH, Inc, its parents, subsidiaries, affiliates, directors, officers, employees, beneficiaries, agents, assigns, component suppliers (both hardware and software), and/ or any third party who provides products or services purchased from or distributed by IRITECH; and the term "You" means the original purchaser and those in privacy with the original purchaser, such as family members or beneficiaries. Information may be obtained from the AAA on line at www.adr.org, by calling 800-778-7879 or writing to American Arbitration Association, 1633 Broadway, 10th Floor, New York, NY, 10019.

5. **General.** IRITECH may assign this Agreement and/or any associated service plan without your consent and without notice to You. If IRITECH does assign this Agreement and/or any associated service plan, the assignee will assume all

obligations to You, IRITECH will be released of all obligations, and You agree to look solely to the assignee for the performance of all obligations under this Agreement and/or any associated service plan. IRITECH, Inc. and its subsidiaries and affiliates are intended beneficiaries of this Agreement. If there is any inconsistency between this Agreement and any other agreement included with or relating to products or services purchased from IRITECH, this Agreement shall govern. This Agreement may not be modified, altered or amended without the written agreement of IRITECH. Any additional or altered terms shall be null and void, unless expressly agreed to in writing by IRITECH. If any term of this Agreement is illegal or unenforceable, the legality and enforceability of the remaining provisions shall not be affected or impaired. This Agreement shall be interpreted under the laws of the State of South Dakota, without giving effect to conflicts of law rules.

6. **Privacy Notice.** You can review IRITECH's Privacy Policy on our web site, located at www.iritech.com. IRITECH will maintain and use your customer information in accordance with its Privacy Policy.

7. **For Residents of Canada:** This Agreement is subject to the applicable provisions of Canadian consumer protection laws that cannot be derogated from by private agreement.

8. **International Support:** You must comply with all applicable export laws and regulations if you export the product from Korea, the United States or Canada. IRITECH does not accept for return any products purchased from a reseller. Customers may be responsible for paying all freight charges incurred in shipping, importing/exporting and receiving replacement products and parts and for arranging and paying for the shipment of any defective part(s) back to the IRITECH.

All international customers are responsible for all customs duties, VAT and other associated taxes and charges.

Please send correspondence about this Agreement to:

IRITECH Customer Service Department
Attn: Warranty Services
3951 Pender Dr., Suite 120A
Fairfax, VA 22030 or
Suite 701 Cheongdong Bldg., 1922 Nambusunhwan-Ro,
Gwanak-Gu, Seoul 151-832, Korea

Current information on technical support and warranty policies, phone numbers and other service information is available on our web sites: www.iritech.com

7.2 End-User License Agreement

IMPORTANT-READ CAREFULLY:

THIS IS A LEGAL AGREEMENT BETWEEN YOU (EITHER AN INDIVIDUAL OR A SINGLE ENTITY) AND IRITECH, INC.

("IRITECH") STATING THE TERMS THAT GOVERN YOUR USE OF THIS SOFTWARE SERVICE. THIS AGREEMENT - TOGETHER WITH ALL UPDATES, ADDITIONAL TERMS, SOFTWARE LICENSES, AND ALL OF IRITECH, INC.'S RULES AND POLICIES-COLLECTIVELY CONSTITUTES THE "AGREEMENT" BETWEEN YOU AND IRITECH, INC.

IF YOU DO NOT AGREE TO THESE TERMS AND CONDITIONS, DO NOT OPEN THE PACKAGE AND DO NOT INSTALL, COPY OR USE THE SOFTWARE; YOU MAY RETURN IT TO YOUR PLACE OF PURCHASE WITHIN FOURTEEN (14) DAYS OF PURCHASE FOR A REFUND OF ITS PURCHASE PRICE. IF THE PACKAGE HAS BEEN OPENED OR THIS SOFTWARE HAS BEEN USED WITHOUT OPENING OF THE PACKAGE, YOU SHALL BE DEEMED TO ACCEPT AND AGREE TO THESE TERMS AS PRESENTED TO YOU HEREINAFTER.

Definitions

The following terms used in this Agreement, to all intents and purposes of this Agreement, have the meaning as specified below respectively except where the context requires otherwise:

(1) "Products" means hardware products such as including, but not limited to, IriCAMM products, IriTerminal products, IriMobile products, IriMagic products, IriShield products, IriHerald products, any and all iris-recognition cameras that are developed and/or manufactured by IRITECH.
(2) "Embedded Software" means any and all software embedded, recorded, contained or enclosed in hardware products and any related printed or electronic material developed and/or provided by IRTECH, and any updates, modifications, revisions, copies, documentation and design data of the foregoing.
(3) "Supporting Software" means any and all software recorded, contained or enclosed in media products and any related printed or electronic material developed and/or provided by IRTECH, and any updates, modifications, revisions, copies, documentation and design data of the foregoing. Supporting Software specifically excludes Embedded Software.
(4) "Software" means any and all of Embedded Software and/or Supporting Software.
(5) "License Tool" means a file or hardware device provided by IRITECH to you so that it may be used to activate the Software.
(6) "Computer" means the computer, server, workstation, embedded system, or any device on which any software can be used.

Any terms used in this Agreement in the singular form includes the meaning of plural and vice versa, if the context requires.

OWNERSHIP, COPYRIGHT, AND INTELLECTUAL PROPERTY RIGHTS.

Although you own the hardware and/or physical media in which the Software is embedded, recorded, contained or enclosed, the Embedded Software and/or Supporting Software is the property of IRITECH.

Title to and intellectual property rights of (including copyrights to) the Embedded Software and Supporting Software, in whole and in part, and all copies thereof, and all modifications, enhancements, and other alterations (“Modifications) and derivatives of the Embedded Software and Supporting Software, if any, are, and shall remain, the sole and exclusive property of IRITECH regardless of who made such modifications or derivatives.

GRANT OF LICENSES.

IRITECH grants you a license authorizing the use of the Software provided that you comply with all terms and conditions of this EULA.

Nothing contained herein shall be construed as transferring any technology including patent, utility model, trademark, design, copyright or trade secrets on or relation to the Embedded Software and Supporting Software.

You may not use the Embedded Software and/or Supporting Software (i) to build any product or software that competes with Products, Embedded Software, and/or Supporting Software; or (ii) to help or assist a person, natural or legal, who directly or indirectly competes with IRITECH or its business or products. You may not use the Products and/or Software for any illegal purpose.

RESTRICTIONS ON USE OF SOFTWARE.

IRITECH licenses the Embedded Software on a per-Product basis. You may not copy the Embedded Software for any purpose.

IRITECH licenses only one copy of the Supporting Software for each Product purchased. You may install and use only one copy of Supporting Software on one Computer for each Product purchased. You may not copy the Supporting Software for any purpose.

In case there is a license activation mechanism required by IRITECH, the Supporting Software must be activated in accordance with the procedure of the License Tool provided by IRITECH.

You may not, and you will not encourage, assist or authorize any other person to modify, reverse assemble, decompile, reverse compile, or otherwise reverse engineer the Product or the Software, or attempt to discover in any way the underlying code of, the Software, whether in whole or in part, create any derivative works from or of the Software, or bypass, modify, defeat or tamper with or circumvent any of the functions or the licensing and/or Software protection mechanism of the Products.

SOFTWARE TRANSFER.

Internal. You may move the Software to a different Computer. After the transfer, you must completely remove the Software from the former Computer.

Transfer to Third Party. The initial user of the Software may make a one-time permanent transfer of this EULA and Software to another end user, provided the initial user retains no copies of the Software. The transfer may not be an indirect transfer such as a consignment. Prior to the transfer, the end user receiving the Software must agree to all the EULA terms.

TERMINATION.

Without prejudice to any other rights, IRITECH may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such event, you must destroy all copies of the Software and all of its component parts.

LIMITED WARRANTY.

IRITECH warrants that any Product provided will be free from defects in material and workmanship for a period of twelve (12) months from the date of initial purchase from IRITECH, or its distributors.

THE SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF INTELLECTUAL PROPERTY. IRITECH DOES NOT WARRANT THAT SOFTWARE WILL MEET YOUR REQUIREMENTS OR THAT OPERATION OF SOFTWARE WILL BE UNINTERRUPTED OR ERROR FREE. IRITECH MAKES NO WARRANTIES WITH RESPECT TO SERVICES.

IN ANY CASE, IRITECH'S ENTIRE LIABILITY AND YOUR EXCLUSIVE REMEDY SHALL BE, AT IRITECH'S SOLE OPTION, EITHER:

(1) REPAIR OR REPLACEMENT, OR;
(2) REFUND OF THE PRICE PAID UPON RETURN OF SOFTWARE TO IRITECH

FOR THE PRODUCTS REPAIRED, RECTIFIED OR REPLACED UNDER THIS AGREEMENT SHALL BE WARRANTED FOR THE BALANCE OF THE ORIGINAL PRODUCTS WARRANTY.

LIMITATION OF LIABILITY.

EXCEPT WHERE THIS EXCLUSION OR RESTRICTION OF LIABILITY WOULD BE VOID OR INEFFECTIVE UNDER APPLICABLE STATUTE OR REGULATION, IN NO EVENT SHALL IRITECH BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS OR

SAVINGS) WHETHER BASED ON CONTRACT, TORT OR ANY OTHER LEGAL THEORY, EVEN IF IRITECH HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

ALL THE LIABILITIES ARISING OR ALLEGED IN CONNECTION WITH THE USE OF SOFTWARE SHALL NOT EXCEED THE PRICE OF RELEVANT PRODUCTS PAID BY YOU, UPON WHICH SUCH LIABILITIES ARE BASED. IF SUCH LIABILITY RELATES TO PARTICULAR ITEMS OF PRODUCTS, SOFTWARE OR SERVICES PROVIDED BY IRITECH, SUCH LIABILITY SHALL BE LIMITED TO THE PRICES OR FEES PAID FOR THE RELEVANT PRODUCTS, SOFTWARE OR SERVICES.

IN THE CASE WHERE NO AMOUNT WAS PAID, IRITECH SHALL HAVE NO LIABILITY FOR ANY DAMAGES WHATSOEVER.

THE PROVISIONS OF THIS SECTION SHALL SURVIVE TERMINATION, EXPIRATION OR CANCELLATION OF THE LICENSE AND/OR THE AGREEMENT.

Governing Law and Jurisdiction.

This Agreement shall be executed in English text as the controlling text and governed by and construed in accordance with the laws of the Commonwealth of Virginia. Any dispute, controversy or difference which may arise between the parties hereto out of or in relation to or in connection with this Agreement, Products or Software, shall be settled amicably through negotiations between the parties. If such negotiations should fail, then such disputes, controversies or differences shall be submitted to a court that has jurisdiction over the place where IRITECH has the relevant office.

ENTIRE AGREEMENT AND SEVERABILITY.

This EULA (including any addendum or amendment to this EULA which is included with the Software) is the entire agreement between you and IRITECH relating to the Software and the support services (if any) and they supersede all prior or contemporaneous oral or written communications, proposals and representations with respect to the Software or any other subject matter covered by this EULA. To the extent the terms of any IRITECH policies or programs for support services conflict with the terms of this EULA, the terms of this EULA shall control. If any provision of this EULA is held to be void, invalid, unenforceable or illegal, the other provisions shall continue in full force and effect.

- The end of the document -

Contact information

Headquarters

11166 Fairfax Boulevard,
Suite 302, Fairfax, VA
22030, USA
Tel: +1 703-877-2135
Fax: +1 703-877-2136

South Koreo office

A-801, Daesung Dipolis Knowledge
Industry Center, 606, Seobusaet-gil,
Geumcheon-gu, Seoul 08504, KOREA
Tel: +82 2-872-3812
Fax: +82 2-872-3815

Viet Nam office

3th Floor, VP1-03, BCONS TOWER
Building, 176/1 - 176/3 Nguyen Van
Thuong St., Ward 25, Binh Thanh
District, Ho Chi Minh City, Vietnam.
Tel: +84 8-6297-9480

India office

320, Raheja Arcade, Koramangala,
Bangalore - 560095
Landline: +91 80-41643057
Phone: +91 98-45025278

Get in touch

Email: info@iritech.com
Website: www.iritech.com