Advanced Card Systems Ltd.
Card & Reader Technologies

ACR122L

VisualVantage

NFC Reader with LCD

Application Programming Interface

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

Table of Contents

1.0. Introduction 4

2.0. ACR122L API 5

2.1. Overview 5
2.2. Reader 5
2.2.1. Define Documentation 5
2.2.1.1. ACR122_GetFirmwareVersion and ACR122_GetFirmwareVersionA 5
2.2.1.2. ACR122_Open and ACR122_OpenA 5
2.2.2. Function Documentation 5
2.2.2.1. ACR122_OpenA 5
2.2.2.2. ACR122_OpenW 6
2.2.2.3. ACR122_Close 6
2.2.2.4. ACR122_GetNumSlots 6
2.2.2.5. ACR122_GetBaudRate 7
2.2.2.6. ACR122_SetBaudRate 7
2.2.2.7. ACR122_GetTimeouts 8
2.2.2.8. ACR122_SetTimeouts 8
2.2.2.9. ACR122_GetFirmwareVersionA 8
2.2.2.10. ACR122_GetFirmwareVersionW 9
2.3. LCD 9
2.3.1. Define Documentation 9
2.3.2. Function Documentation 10
2.3.2.1. ACR122_DisplayLcdMessageA 10
2.3.2.2. ACR122_DisplayLcdMessageW 10
2.3.2.3. ACR122_DisplayLcdMessageExA 11
2.3.2.4. ACR122_DisplayLcdMessageExW 11
2.3.2.5. ACR122_DisplayLcdMessageGBA 12
2.3.2.6. ACR122_DisplayLcdMessageGBW 12
2.3.2.7. ACR122_DrawLcd 13
2.3.2.8. ACR122_StartLcdScrolling 14
2.3.2.9. ACR122_PauseLcdScrolling 15
2.3.2.10. ACR122_StopLcdScrolling 15
2.3.2.11. ACR122_ClearLcd 15
2.3.2.12. ACR122_EnableLcdBacklight 16
2.3.2.13. ACR122_SetLcdContrast 16
2.4. LED 17
2.4.1. Function Documentation 17
2.4.1.1. ACR122_EnableLed 17
2.4.1.2. ACR122_SetLedStates 17
2.4.1.3. ACR122_SetLedStatesWithBeep 18
2.5. Buzzer 19
2.5.1. Function Documentation 19
2.5.1.1. ACR122_Beep 19
2.6. Card 19
2.6.1. Function Documentation 19
2.6.1.1. ACR122_DirectTransmit 19
2.6.1.2. ACR122_ExchangeApdu 20
2.6.1.3. ACR122_PowerOffIcc 20
2.6.1.4. ACR122_PowerOnIcc 21

Appendix A. Data Structures 22

Appendix A.1. _ACR122_TIMEOUTS Struct Reference 22
Appendix A.2. _ACR122_LED_CONTROL Struct Reference 22

Appendix B. Error Codes Returned by High-Level APIs 23

Appendix C. Standard Program Flow 24

Figures

Figure 1: ACR122L Library Architecture..... 4

1.0. Introduction

This manual describes the use of ACR122L interface software to facilitate application development with the ACR122L reader. This interface software is supplied in the form of 32-bit and 64-bit DLL (Dynamic Link Library) which can be programmed using popular development tools like Java, Delphi, Visual Basic, Visual C++, Visual C# and Visual Basic .Net.

ACR122L can be connected to the PC via the RS232 interface.

The architecture of the ACR122L library can be visualized as the following diagram:

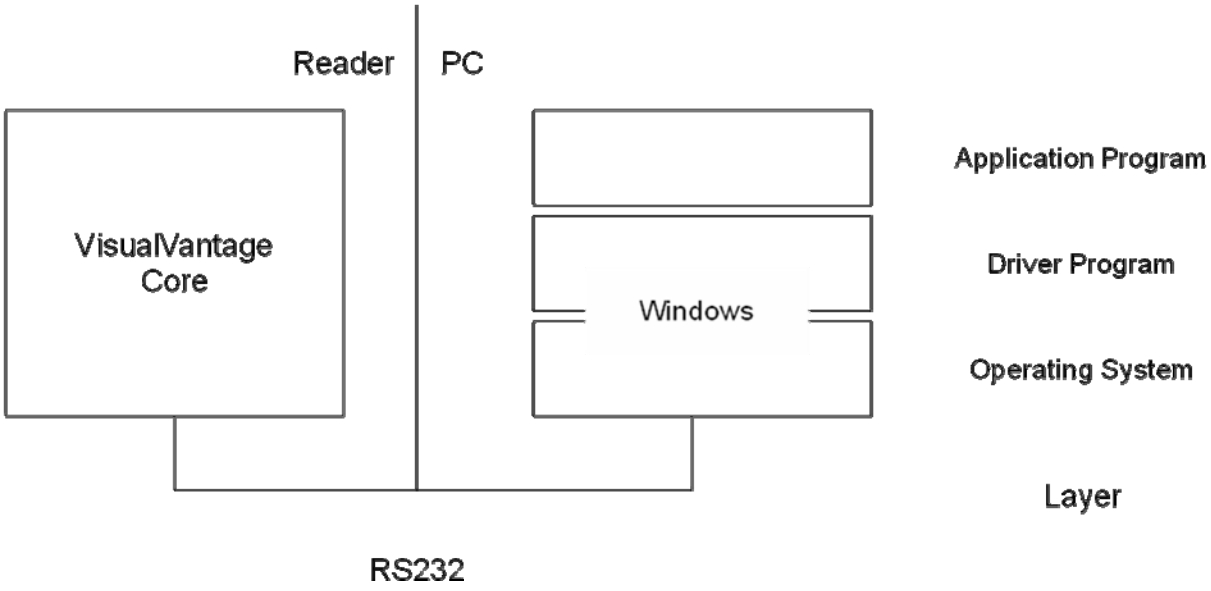

Figure 1: ACR122L Library Architecture

2.0. ACR122L API

2.1. Overview

The ACR122L DLL is a set of high-level functions provided for the application software to use. It supplies a consistent API (Application Programming Interface) for the application to operate on the ACR122L and the corresponding presented card. The DLL communicates with the ACR122L via the communication port facilities provided by the operating system.

The ACR122L API defines a common way of accessing the ACR122L. Application programs invoke the ACR122L through the interface functions and perform operations on the presented card.

The header file ACR122.h is available for the program developer, which contains all the function prototypes and macros described below.

2.2. Reader

2.2.1. Define Documentation

2.2.1.1. ACR122_GetFirmwareVersion and ACR122_GetFirmwareVersionA

Format:

```
#define ACR122_GetFirmwareVersion ACR122_GetFirmwareVersionA
```

Define Description:

`ACR122_GetFirmwareVersion` will be mapped to `ACR122_GetFirmwareVersionW()` function if UNICODE is defined.

Otherwise, it will be mapped to `ACR122_GetFirmwareVersionA()` function.

2.2.1.2. ACR122_Open and ACR122_OpenA

Format:

```
#define ACR122_Open ACR122_OpenA
```

Define Description:

`ACR122_Open` will be mapped to `ACR122_OpenW()` function if UNICODE is defined.

Otherwise, it will be mapped to `ACR122_OpenA()` function.

2.2.2. Function Documentation

2.2.2.1. ACR122_OpenA

Format:

```
DWORD WINAPI ACR122_OpenA ( LPCSTR   portName,
                            LPHANDLE phReader
                          )
```

Function Description: Open reader (ANSI).

This function opens a reader and returns a handle value as reference.

Parameters	Description	
[in] portName	Port name. "\\.\COM1" means that the reader is connected to COM1 in Windows.	
[out] phReader	Pointer to the HANDLE variable.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.2.2.2. ACR122_OpenW

Format:

```
DWORD WINAPI ACR122_OpenW ( LPCWSTR  portName,
                            LPHANDLE phReader
                          )
```

Function Description: Open reader (Unicode).

This function opens a reader and returns a handle value as reference.

Parameters	Description	
[in] portName	Port name. "\\.\COM1" means that the reader is connected to COM1 in Windows.	
[out] phReader	Pointer to the HANDLE variable.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.2.2.3. ACR122_Close

Format:

```
DWORD WINAPI ACR122_Close ( HANDLE hReader )
```

Function Description: Close reader.

This function closes the reader and releases the resources.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.2.2.4. ACR122_GetNumSlots

Format:

```
DWORD WINAPI ACR122_GetNumSlots ( HANDLE  hReader,
                                  LPDWORD pNumSlots
                                )
```

Function Description: Get number of slots.

This function retrieves the number of slots.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[out] pNumSlots	Pointer to a DWORD variable in which the number of slots is returned.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.2.2.5. ACR122_GetBaudRate

Format:

```
DWORD WINAPI ACR122_GetBaudRate ( HANDLE  hReader,
                                  LPDWORD pBaudRate
                                )
```

Function Description: Get baud rate.

This function retrieves the baud rate of reader.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[out] pBaudRate	Pointer to a DWORD variable in which the baud rate is returned.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.2.2.6. ACR122_SetBaudRate

Format:

```
DWORD WINAPI ACR122_SetBaudRate ( HANDLE hReader,
                                  DWORD baudRate
                                )
```

Function Description: Set baud rate.

This function sets the communication baud rate of reader. Actually, the reader supports 9600 bps and 115200 bps.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] baudRate	Baud rate must be 9600 bps or 115200 bps.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.2.2.7. ACR122_GetTimeouts

Format:

```
DWORD WINAPI ACR122_GetTimeouts ( HANDLE          hReader,
                                  PACR122_TIMEOUTS pTimeouts
                                )
```

Function Description: Get timeouts.

This function retrieves the timeout parameters for status and response operations of the reader.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[out] pTimeouts	Pointer to a ACR122_TIMEOUTS structure in which the timeout information is returned.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

Note: For PACR122_TIMEOUTS, please see _ACR122_TIMEOUTS Struct Reference.

2.2.2.8. ACR122_SetTimeouts

Format:

```
DWORD WINAPI ACR122_SetTimeouts ( HANDLE                hReader,
                                  const PACR122_TIMEOUTS pTimeouts
                                )
```

Function Description: Set timeouts.

This function sets the timeout parameters for status and response operations on the reader.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] pTimeouts	Pointer to a ACR122_TIMEOUTS structure that contains the new timeout values	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

Note: For PACR122_TIMEOUTS, please see _ACR122_TIMEOUTS Struct Reference.

2.2.2.9. ACR122_GetFirmwareVersionA

Format:

```
DWORD WINAPI ACR122_GetFirmwareVersionA ( HANDLE  hReader,
                                          DWORD   slotNum,
                                          LPSTR   firmwareVersion,
                                          LPDWORD pFirmwareVersionLen
                                        )
```

Function Description: Get firmware version (ANSI).

This function retrieves the firmware version in ANSI string of the slot.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] slotNum	Slot number.	
[out] firmwareVersion	A pointer to the buffer that receives the firmware version returned from the reader.	
[in,out] pFirmwareVersionLen	The length in number of bytes of the firmwareVersion parameter and receives the actual number of bytes received from the reader.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.2.2.10. ACR122_GetFirmwareVersionW

Format:

```
DWORD WINAPI ACR122_GetFirmwareVersionW ( HANDLE  hReader,
                                          DWORD   slotNum,
                                          LPWSTR  firmwareVersion,
                                          LPDWORD pFirmwareVersionLen
                                        )
```

Function Description: Get firmware version (Unicode).

This function retrieves the firmware version in Unicode string of the slot.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] slotNum	Slot number.	
[out] firmwareVersion	A pointer to the buffer that receives the firmware version returned from the reader.	
[in,out] pFirmwareVersionLen	The length in number of bytes of the firmwareVersion parameter and receives the actual number of bytes received from the reader.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.3. LCD

2.3.1. Define Documentation

Format:

```
#define ACR122_DisplayLcdMessage ACR122_DisplayLcdMessageA
```

Define Description:

ACR122_DisplayLcdMessage will be mapped to ACR122_DisplayLcdMessageW()function if

UNICODE is defined.

Otherwise, it will be mapped to `ACR122_DisplayLcdMessageA()` function.

2.3.2. Function Documentation

2.3.2.1. ACR122_DisplayLcdMessageA

Format:

```
DWORD WINAPI ACR122_DisplayLcdMessageA ( HANDLE hReader,
                                         DWORD  row,
                                         DWORD  col,
                                         LPCSTR message
                                       )
```

Function Description: Display LCD message in ANSI mode.

This function displays LCD message on the reader.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
`[in] row`	Row number must be from `0` to `1`.	
`[in] col`	Column number must be from `0` to `15`.	
`[in] message`	Message for display. The length of message must be less than or equal to 16 characters.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.
	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.2. ACR122_DisplayLcdMessageW

Format:

```
DWORD WINAPI ACR122_DisplayLcdMessageW ( HANDLE  hReader,
                                         DWORD   row,
                                         DWORD   col,
                                         LPCWSTR message
                                       )
```

Function Description: Display LCD message in Unicode mode.

This function displays LCD message on the reader.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
`[in] row`	Row number must be from `0` to `1`.	
`[in] col`	Column number must be from `0` to `15`.	
`[in] message`	Message for display. The length of message must be less than or equal to 16 characters.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.

Function Description: Display LCD message in Unicode mode. (cont.)

Return Value	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.3. ACR122_DisplayLcdMessageExA

Format:

```
DWORD WINAPI ACR122_DisplayLcdMessageExA ( HANDLE  hReader,
                                           DWORD   row,
                                           DWORD   col,
                                           DWORD   TableIndex,
                                           DWORD   BoldMode,
                                           LPCSTR  message
                                         )
```

Function Description: Display LCD message in Ex (ANSI) mode.

This function displays LCD message on the reader.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
`[in] row`	Row number must be from `0` to `1`.	
`[in] col`	Column number must be from `0` to `15`.	
`[in] TableIndex`	Select which fonts table be used, from `0` to `2`.	
`[in] BoldMode`	`1` – Bold Mode, `0` – Normal mode	
`[in] message`	Message for display. The length of message must be less than or equal to 16 characters.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.
	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.4. ACR122_DisplayLcdMessageExW

Format:

```
DWORD WINAPI ACR122_DisplayLcdMessageExW ( HANDLE  hReader,
                                           DWORD   row,
                                           DWORD   col,
                                           DWORD   TableIndex,
                                           DWORD   BoldMode,
                                           LPCWSTR message
                                         )
```

Function Description: Display LCD message in Ex (Unicode).

This function displays LCD message on the reader.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] row	Row number must be from 0 to 1.	
[in] col	Column number must be from 0 to 15.	
[in] TableIndex	Select which fonts table be used, from 0 to 2.	
[in] BoldMode	1 – Bold Mode, 0 – Normal mode	
[in] message	Message for display. The length of message must be less than or equal to 16 characters.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.5. ACR122_DisplayLcdMessageGBA

Format:

```
DWORD WINAPI ACR122_DisplayLcdMessageGBA ( HANDLE hReader,
                                           DWORD  row,
                                           DWORD  col,
                                           DWORD  BoldMode,
                                           LPCSTR message
                                         )
```

Function Description: Display LCD message in GB (ANSI) mode.

This function displays LCD message on the reader using Simplified Chinese font.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] row	Row number must be from 0 to 1.	
[in] col	Column number must be from 0 to 7.	
[in] BoldMode	1 – Bold Mode, 0 – Normal mode	
[in] message	Message for display. The length of message must be less than or equal to 8 characters.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.6. ACR122_DisplayLcdMessageGBW

Format:

```
DWORD WINAPI ACR122_DisplayLcdMessageGBA ( HANDLE  hReader,
```

```
                                    DWORD   row,
                                    DWORD   col,
                                    DWORD   BoldMode,
                                    LPCWSTR message
                                  )
```

Function Description: Display LCD message in GB (Unicode) mode.

This function displays LCD message on the reader using Simplified Chinese font.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] row	Row number must be from 0 to 1.	
[in] col	Column number must be from 0 to 7.	
[in] BoldMode	1 – Bold Mode, 0 – Normal mode	
[in] message	Message for display. The length of message must be less than or equal to 8 characters.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.7. ACR122_DrawLcd

Format:

```
DWORD WINAPI ACR122_DrawLcd ( HANDLE       hReader,
                              DWORD        lineIndex,
                              const LPBYTE pixelBuffer,
                              DWORD        pixelBufferLen
                            )
```

Function Description: Display LCD message in Graphic mode.

This function displays LCD message on the reader.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] lineIndex	The line to start update the LCD Display.	
[in] pixelBuffer	Buffer with pixel data to display, the length should not be larger than 128 bytes	
[in] pixelBufferLen	Length of pixel buffer in bytes.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.8. ACR122_StartLcdScrolling

Format:

```
DWORD WINAPI ACR122_StartLcdScrolling ( HANDLE hReader,
                                        DWORD  row,
                                        DWORD  col,
                                        DWORD  HRange,
                                        DWORD  VRange,
                                        DWORD  Speed,
                                        DWORD  Direction
                                      )
```

Function Description: Scroll current LCD display.

This function sets scrolling feature of the current LCD display.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open( ) function.	
[in] row	Row number must be 0.	
[in] col	Column number must be 0.	
[in] Hrange	Horizontal range (Unit: 8 pixels). The value must be 15.	
[in] Vrange	Vertical range (Unit: 1 pixel). The value must be 31.	
[in] Speed	Bit0 - Bit3: Number of pixels in pre-scrolling. Bit4 - Bit7: Scrolling period.	
[in] Direction	Scrolling direction, from 0 to 3	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

[in] Speed

Value (Bit4 - Bit7)	Scrolling Period
0	1 Unit
1	3 Units
2	5 Units
3	7 Units
4	17 Units
5	19 Units
6	21 Units
7	23 Units
8	129 Units
9	131 Units
10	133 Units
11	135 Units
12	145 Units
13	147 Units

14	149 Units
15	151 Units

`[in] Direction`

Value	Meaning
0	From left to right
1	From right to left
2	From top to bottom
3	From bottom to top

2.3.2.9. ACR122_PauseLcdScrolling

Format:

```
DWORD WINAPI ACR122_PauseLcdScrolling ( HANDLE hReader )
```

Function Description: Pause LCD scrolling.

This function pauses the LCD scrolling set previously.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.
	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.10. ACR122_StopLcdScrolling

Format:

```
DWORD WINAPI ACR122_StopLcdScrolling ( HANDLE hReader )
```

Function Description: Stop LCD scrolling.

This function stops the LCD scrolling set previously. The LCD display will return to normal display position.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.
	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.11. ACR122_ClearLcd

Format:

```
DWORD WINAPI ACR122_ClearLcd ( HANDLE hReader )
```

Function Description: Clear LCD.

This function clears the LCD display of the reader.

Parameters	Description
`[in] hReader`	A reference value returned from `ACR122_Open()` function.

Function Description: Clear LCD. (cont.)

Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

Note: For ACR122L firmware versions 307 and above, using the ACR122L_CLearLCD function successively with other LCD functions requires the application to handle an additional 100 ms time delay.

2.3.2.12. ACR122_EnableLcdBacklight

Format:

```
DWORD WINAPI ACR122_EnableLcdBacklight ( HANDLE hReader,
                                         BOOL   enabled
                                       )
```

Function Description: Enable LCD backlight.

This function enables or disables the LCD backlight of the reader.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] enabled	Set to TRUE to enable backlight. Otherwise, set to FALSE.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.3.2.13. ACR122_SetLcdContrast

Format:

```
DWORD WINAPI ACR122_SetLcdContrast ( HANDLE hReader,
                                     DWORD  level
                                   )
```

Function Description: Set LCD contrast.

This function sets the LCD contrast level of the reader.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] level	Contrast level. The value must be from 0x00 to 0x0F	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.4. LED

2.4.1. Function Documentation

2.4.1.1. ACR122_EnableLed

Format:

```
DWORD WINAPI ACR122_EnableLed ( HANDLE hReader,
                                BOOL   enabled
                              )
```

Function Description: Enable LED.

This function enables or disables LED control to the application. By default, LED is controlled by the firmware. Before calling `ACR122_SetLedStatesWithBeep()` and `ACR122_SetLedStates()`, the application needs to call this function in order to control the LED.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
`[in] enabled`	Set to TRUE to enable LED. Otherwise, set to FALSE.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.
	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.4.1.2. ACR122_SetLedStates

Format:

```
DWORD WINAPI ACR122_SetLedStates ( HANDLE hReader,
                                   DWORD *states,
                                   DWORD  numStates
                                 )
```

Function Description: Set LED states.

This function turns on or off the LEDs on the reader. `LED0`, `LED1`, `LED2` and `LED3` can be controlled.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
`[in] states`	A pointer to the array of states. Possible values are `ACR122_LED_STATE_OFF` and `ACR122_LED_STATE_ON`.	
`[in] numStates`	Number of states must be 4.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.
	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.4.1.3. ACR122_SetLedStatesWithBeep

Format:

```
DWORD WINAPI ACR122_SetLedStatesWithBeep ( HANDLE              hReader,
                                           PACR122_LED_CONTROL controls,
                                           DWORD               numControls,
                                           DWORD               t1,
                                           DWORD               t2,
                                           DWORD               numTimes,
                                           DWORD               buzzerMode
                                         )
```

Function Description: Set LED states with beep.

This function controls LED0, LED1 and buzzer operation on the reader.

Parameters	Description
[in] hReader	A reference value returned from ACR122_Open() function.
[in] controls	A pointer to the ACR122_LED_CONTROL data structure.
[in] numControls	Number of controls must be 2.
[in] t1	T1 in milliseconds. The value must be from 0 to 25500.
[in] t2	T2 in milliseconds. The value must be from 0 to 25500
[in] numTimes	Number of times. The values must be from 0 to 255.
[in] buzzerMode	A bitmask of buzzer mode. Possible values may be combined with the OR operation. **Value** / **Meaning** ACR122_BUZZER_MODE_OFF — The buzzer will not turn on. ACR122_BUZZER_MODE_ON_T1 — The buzzer will turn on during T1 duration. ACR122_BUZZER_MODE_ON_T2 — The buzzer will turn on during T2 duration.
Return Value	ERROR_SUCCESS — The operation completed successfully. Failure — An error code. See Windows API error codes and ACR122 error codes.

Note: For PACR122_LED_CONTROL, please see _ACR122_LED_CONTROL Struct Reference.

2.5. Buzzer

2.5.1. Function Documentation

2.5.1.1. ACR122_Beep

Format:

```
DWORD WINAPI ACR122_Beep ( HANDLE hReader,
                           DWORD  buzzerOnDuration,
                           DWORD  buzzerOffDuration,
                           DWORD  numTimes
                         )
```

Function Description: Beep.

This function controls the buzzer on the reader to generate the beep sound and it does not return control to its caller until the sound finishes.

Parameters	Description	
`[in] hReader`	A reference value returned from `ACR122_Open()` function.	
`[in] buzzerOnDuration`	Buzzer ON duration in milliseconds. The value must be from `0` to `25500`.	
`[in] buzzerOffDuration`	Buzzer OFF duration in milliseconds. The value must be from `0` to `25500`.	
`[in] numTimes`	Number of times.	
Return Value	`ERROR_SUCCESS`	The operation completed successfully.
	`Failure`	An error code. See Windows API error codes and ACR122 error codes.

2.6. Card

2.6.1. Function Documentation

2.6.1.1. ACR122_DirectTransmit

Format:

```
DWORD WINAPI ACR122_DirectTransmit ( HANDLE       hReader,
                                     const LPBYTE sendBuffer,
                                     DWORD        sendBufferLen,
                                     LPBYTE       recvBuffer,
                                     LPDWORD      pRecvBufferLen
                                   )
```

Function Description: Direct transmit TAG command.

This function sends TAG command and receives response from the reader.

Parameters	Description
`[in] hReader`	A reference value returned from `ACR122_Open()` function.

Function Description: Direct transmit TAG command. (cont.)

[in] sendBuffer	A pointer to the actual data to be written to the card.	
[in] sendBufferLen	The length in number of bytes of the sendBuffer parameter.	
[in] recvBuffer	A pointer to any data returned from the card.	
[in,out] pRecvBufferLen	The length in number of bytes of the recvBuffer parameter and receives the actual number of bytes received from the card.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.6.1.2. ACR122_ExchangeApdu

Format:

```
DWORD WINAPI ACR122_ExchangeApdu ( HANDLE       hReader,
                                   DWORD        slotNum,
                                   const LPBYTE sendBuffer,
                                   DWORD        sendBufferLen,
                                   LPBYTE       recvBuffer,
                                   LPDWORD      pRecvBufferLen
                                 )
```

Function Description: Exchange APDU.

This function sends APDU command and receives APDU response from the card.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open() function.	
[in] slotNum	Slot number.	
[in] sendBuffer	A pointer to the actual data to be written to the card.	
[in] sendBufferLen	The length in number of bytes of the sendBuffer parameter.	
[out] recvBuffer	A pointer to any data returned from the card.	
[in,out] pRecvBufferLen	The length in number of bytes of the parameter and receives the actual number of bytes received from the card.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.6.1.3. ACR122_PowerOffIcc

Format:

```
DWORD WINAPI ACR122_PowerOffIcc ( HANDLE hReader,
                                  DWORD  slotNum
                                )
```

Function Description: Power off ICC in slot.

This function powers off the card in the slot.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open( ) function.	
[in] slotNum	Slot number.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

2.6.1.4. ACR122_PowerOnIcc

Format:

```
DWORD WINAPI ACR122_PowerOnIcc ( HANDLE  hReader,
                                 DWORD   slotNum,
                                 LPBYTE  atr,
                                 LPDWORD pAtrLen
                               )
```

Function Description: Power on ICC in slot.

This function powers on the card in the slot and returns the ATR string from the card.

Parameters	Description	
[in] hReader	A reference value returned from ACR122_Open( ) function.	
[in] slotNum	Slot number.	
[out] atr	A pointer to the buffer that receives the ATR string returned from the card.	
[in,out] pAtrLen	The length in number of bytes of the atr parameter and receives the actual number of bytes received from the card.	
Return Value	ERROR_SUCCESS	The operation completed successfully.
	Failure	An error code. See Windows API error codes and ACR122 error codes.

Appendix A. Data Structures

Appendix A.1. _ACR122_TIMEOUTS Struct Reference

This data structure is used in **ACR122_GetTimeouts()** and **ACR122_SetTimeouts()** function.

- **DWORD _ACR122_TIMEOUTS::numResponseRetries**

  Number of response retries.

  Default is `1`.

- **DWORD _ACR122_TIMEOUTS::numStatusRetries**

  Number of status retries.

  Default is `1`.

- **DWORD _ACR122_TIMEOUTS::responseTimeout**

  Response timeout in milliseconds.

  Default is `10000 ms`

- **DWORD _ACR122_TIMEOUTS::statusTimeout**

  Status timeout in milliseconds.

  Default is `2000 ms`.

Appendix A.2. _ACR122_LED_CONTROL Struct Reference

This data structure is used in **ACR122_SetLedStatesWithBeep()** function.

- **BOOL _ACR122_LED_CONTROL::blinkEnabled**

  Enable blink.

  Set to `TRUE` to enable blink. Otherwise, set to `FALSE`.

- **DWORD _ACR122_LED_CONTROL::finalState**

  Final state.

  Possible values are `ACR122_LED_STATE_OFF` and `ACR122_LED_STATE_ON`.

- **DWORD _ACR122_LED_CONTROL::initialBlinkingState**

  Initial blinking state.

  Possible values are `ACR122_LED_STATE_OFF` and `ACR122_LED_STATE_ON`.

- **BOOL _ACR122_LED_CONTROL::updateEnabled**

  Enable update.

  Set to `TRUE` to update the state. Otherwise, set to `FALSE` to keep the state unchanged.

Appendix B. Error Codes Returned by High-Level APIs

- **ACR122_ERROR_NO_MORE_HANDLES ((DWORD) 0x20000001L)**
  The handle is invalid.
- **ACR122_ERROR_UNKNOWN_STATUS ((DWORD) 0x20000002L)**
  Reader unknown error.
- **ACR122_ERROR_OPERATION_FAILURE ((DWORD) 0x20000003L)**
  Operation failed.
- **ACR122_ERROR_OPERATION_TIMEOUT ((DWORD) 0x20000004L)**
  Timeout operation.
- **ACR122_ERROR_INVALID_CHECKSUM ((DWORD) 0x20000005L)**
  Checksum calculation error
- **ACR122_ERROR_INVALID_PARAMETER ((DWORD) 0x20000006L)**
  Incorrect parameter input.

Appendix C. Standard Program Flow

```
// ACR122L Test Program.cpp : Defines the entry point for the console
application.
//
#include "stdafx.h"
#include "acr122.h"
#include "conio.h"

int _tmain(int argc, _TCHAR* argv[])
{
	DWORD Error;
	DWORD i;

	HANDLE hReader;

	TCHAR Temp[255];

	DWORD LED_Status[4];
	ACR122_LED_CONTROL LED_CTRL[2];

	BYTE *cmd;

	BYTE PollCmd[] = {0xD4,0x4A,0x01,0x00};
	BYTE GetChallenge[] = {0x80,0x84,0x00,0x00,0x08};
	BYTE RecBuff[255];

	DWORD RecLen;

	TCHAR DisplayMessage1[] = _T("ACR122L TESTING ");
	TCHAR DisplayMessage2[] = _T("123456789ABCDEF@");

	TCHAR *FW_Ver;
	DWORD FW_Len;

	FW_Ver = Temp;
	FW_Len = 255;

	printf("1) Open COM1\n");
	printf("=========================================\n");
	Error = ACR122_Open(_T("\\\\.\\COM1"),&hReader);
	printf("Error Code: %d\n",Error);
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	printf("2) Test Firmware Version Slot 0\n");
	printf("=========================================\n");
	Error = ACR122_GetFirmwareVersion(hReader,0,FW_Ver,&FW_Len);

	printf("Error Code: %d\n",Error);

	if(Error == 0)
	{
		printf("Firmware Code: ");
		for (i=0;i<FW_Len;i++)
			printf("%s",Temp+i);
		printf("\n");
	}
	printf("Press any key to continuou\n");
	getch();
```

```
printf("\n");

printf("3) Test Firmware Version Slot 1\n");
printf("=======================================\n");
Error = ACR122_GetFirmwareVersion(hReader,1,FW_Ver,&FW_Len);

printf("Error Code: %d\n",Error);

if(Error == 0)
{
      printf("Firmware Code: ");
      for (i=0;i<FW_Len;i++)
            printf("%s",Temp+i);
      printf("\n");
}
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("4) Test Firmware Version Slot 2\n");
printf("=======================================\n");
Error = ACR122_GetFirmwareVersion(hReader,2,FW_Ver,&FW_Len);

printf("Error Code: %d\n",Error);

if(Error == 0)
{
      printf("Firmware Code: ");
      for (i=0;i<FW_Len;i++)
            printf("%s",Temp+i);
      printf("\n");
}
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("5) Display First Row LCD Message\n");
printf("=======================================\n");
Error = ACR122_DisplayLcdMessage(hReader,0,0,DisplayMessage1);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("6) Display Second Row LCD Message\n");
printf("=======================================\n");
Error = ACR122_DisplayLcdMessage(hReader,1,0,DisplayMessage2);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("7) Turn On Backlight\n");
printf("=======================================\n");
Error = ACR122_EnableLcdBacklight(hReader,true);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("8) Turn Off Backlight\n");
```

```
printf("=========================================\n");
Error = ACR122_EnableLcdBacklight(hReader,false);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("9) Clear LCD\n");
printf("=========================================\n");
Error = ACR122_ClearLcd(hReader);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

Error = ACR122_EnableLed(hReader,true);

printf("10) Turn ON LED0 \n");
printf("=========================================\n");

LED_Status[0] = ACR122_LED_STATE_ON;
LED_Status[1] = ACR122_LED_STATE_OFF;
LED_Status[2] = ACR122_LED_STATE_OFF;
LED_Status[3] = ACR122_LED_STATE_OFF;

Error = ACR122_SetLedStates(hReader,LED_Status,4);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("11) Turn ON LED1 \n");
printf("=========================================\n");

LED_Status[0] = ACR122_LED_STATE_OFF;
LED_Status[1] = ACR122_LED_STATE_ON;
LED_Status[2] = ACR122_LED_STATE_OFF;
LED_Status[3] = ACR122_LED_STATE_OFF;

Error = ACR122_SetLedStates(hReader,LED_Status,4);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("12) Turn ON LED2 \n");
printf("=========================================\n");

LED_Status[0] = ACR122_LED_STATE_OFF;
LED_Status[1] = ACR122_LED_STATE_OFF;
LED_Status[2] = ACR122_LED_STATE_ON;
LED_Status[3] = ACR122_LED_STATE_OFF;

Error = ACR122_SetLedStates(hReader,LED_Status,4);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("13) Turn ON LED3 \n");
printf("=========================================\n");
```

```
LED_Status[0] = ACR122_LED_STATE_OFF;
LED_Status[1] = ACR122_LED_STATE_OFF;
LED_Status[2] = ACR122_LED_STATE_OFF;
LED_Status[3] = ACR122_LED_STATE_ON;

Error = ACR122_SetLedStates(hReader,LED_Status,4);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("14) Turn ON All LEDs \n");
printf("=========================================\n");

LED_Status[0] = ACR122_LED_STATE_ON;
LED_Status[1] = ACR122_LED_STATE_ON;
LED_Status[2] = ACR122_LED_STATE_ON;
LED_Status[3] = ACR122_LED_STATE_ON;

Error = ACR122_SetLedStates(hReader,LED_Status,4);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("15) Turn OFF All LEDs \n");
printf("=========================================\n");

LED_Status[0] = ACR122_LED_STATE_OFF;
LED_Status[1] = ACR122_LED_STATE_OFF;
LED_Status[2] = ACR122_LED_STATE_OFF;
LED_Status[3] = ACR122_LED_STATE_OFF;

Error = ACR122_SetLedStates(hReader,LED_Status,4);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("16) Buzzer Test\n");
printf("=========================================\n");
Error = ACR122_Beep(hReader,500,500,2);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("17) Set LED with Buzzer \n");
printf("=========================================\n");
LED_Status[0] = ACR122_LED_STATE_OFF;
LED_Status[1] = ACR122_LED_STATE_OFF;
LED_Status[2] = ACR122_LED_STATE_OFF;
LED_Status[3] = ACR122_LED_STATE_OFF;

LED_CTRL[0].blinkEnabled = true;
LED_CTRL[0].finalState = ACR122_LED_STATE_OFF;
LED_CTRL[0].initialBlinkingState = ACR122_LED_STATE_ON;
LED_CTRL[0].updateEnabled = true;

LED_CTRL[1].blinkEnabled = true;
```

```
	LED_CTRL[1].finalState = ACR122_LED_STATE_OFF;
	LED_CTRL[1].initialBlinkingState = ACR122_LED_STATE_OFF;
	LED_CTRL[1].updateEnabled = true;

	Error                                                                    =
ACR122_SetLedStatesWithBeep(hReader,LED_CTRL,2,500,500,3,ACR122_BUZZER_MODE
_ON_T1);
	printf("Error Code: %d\n",Error);
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	Error = ACR122_EnableLed(hReader,false);

	printf("18) Direct Command - Poll Type A Command \n");
	printf("=========================================\n");
	cmd = PollCmd;
	RecLen = 255;
	Error = ACR122_DirectTransmit(hReader,cmd,4,RecBuff,&RecLen);
	printf("Error Code: %d\n",Error);

	if(Error == 0)
	{
		printf("Data Return: ");
		for (i=0;i<RecLen;i++)
		{
			printf("%02X",RecBuff[i]);
			printf(" ");
		}
		printf("\n");
	}
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	printf("19) Activate SAM A\n");
	printf("==========================================\n");
	RecLen = 255;
	Error = ACR122_PowerOnIcc(hReader,0,RecBuff,&RecLen);
	printf("Error Code: %d\n",Error);

	if(Error == 0)
	{
		printf("Data Return: ");
		for (i=0;i<RecLen;i++)
		{
			printf("%02X",RecBuff[i]);
			printf(" ");
		}
		printf("\n");
	}
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	printf("20) Activate SAM B\n");
	printf("==========================================\n");
	RecLen = 255;
	Error = ACR122_PowerOnIcc(hReader,1,RecBuff,&RecLen);
	printf("Error Code: %d\n",Error);
```

```
if(Error == 0)
{
	printf("Data Return: ");
	for (i=0;i<RecLen;i++)
	{
		printf("%02X",RecBuff[i]);
		printf(" ");
	}
	printf("\n");
}
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("21) Activate SAM C\n");
printf("=========================================\n");
RecLen = 255;
Error = ACR122_PowerOnIcc(hReader,2,RecBuff,&RecLen);
printf("Error Code: %d\n",Error);

if(Error == 0)
{
	printf("Data Return: ");
	for (i=0;i<RecLen;i++)
	{
		printf("%02X",RecBuff[i]);
		printf(" ");
	}
	printf("\n");
}
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("22) Get Challenge SAM A\n");
printf("=========================================\n");
cmd = GetChallenge;
RecLen = 255;
Error = ACR122_ExchangeApdu(hReader,0,cmd,5,RecBuff,&RecLen);
printf("Error Code: %d\n",Error);

if(Error == 0)
{
	printf("Data Return: ");
	for (i=0;i<RecLen;i++)
	{
		printf("%02X",RecBuff[i]);
		printf(" ");
	}
	printf("\n");
}
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("23) Get Challenge SAM B\n");
printf("=========================================\n");
cmd = GetChallenge;
RecLen = 255;
Error = ACR122_ExchangeApdu(hReader,1,cmd,5,RecBuff,&RecLen);
printf("Error Code: %d\n",Error);
```

```
if(Error == 0)
{
	printf("Data Return: ");
	for (i=0;i<RecLen;i++)
	{
		printf("%02X",RecBuff[i]);
		printf(" ");
	}
	printf("\n");
}
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("24) Get Challenge SAM C\n");
printf("=======================================\n");
cmd = GetChallenge;
RecLen = 255;
Error = ACR122_ExchangeApdu(hReader,2,cmd,5,RecBuff,&RecLen);
printf("Error Code: %d\n",Error);

if(Error == 0)
{
	printf("Data Return: ");
	for (i=0;i<RecLen;i++)
	{
		printf("%02X",RecBuff[i]);
		printf(" ");
	}
	printf("\n");
}
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("25) Deactivate SAM A\n");
printf("=======================================\n");
Error = ACR122_PowerOffIcc(hReader,0);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("26) Deactivate SAM B\n");
printf("=======================================\n");
Error = ACR122_PowerOffIcc(hReader,1);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("27) Deactivate SAM C\n");
printf("=======================================\n");
Error = ACR122_PowerOffIcc(hReader,2);
printf("Error Code: %d\n",Error);
printf("Press any key to continuou\n");
getch();
printf("\n");

printf("28) Change Baud Rate to 115200\n");
printf("=======================================\n");
```

```
	Error = ACR122_SetBaudRate(hReader,115200);
	printf("Error Code: %d\n",Error);
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	printf("29) Test Firmware Version Slot 0\n");
	printf("=======================================\n");
	Error = ACR122_GetFirmwareVersion(hReader,0,FW_Ver,&FW_Len);

	printf("Error Code: %d\n",Error);

	if(Error == 0)
	{
		printf("Firmware Code: ");
		for (i=0;i<FW_Len;i++)
			printf("%s",Temp+i);
		printf("\n");
	}
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	printf("30) Test Firmware Version Slot 1\n");
	printf("=======================================\n");
	Error = ACR122_GetFirmwareVersion(hReader,1,FW_Ver,&FW_Len);

	printf("Error Code: %d\n",Error);

	if(Error == 0)
	{
		printf("Firmware Code: ");
		for (i=0;i<FW_Len;i++)
			printf("%s",Temp+i);
		printf("\n");
	}
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	printf("31) Test Firmware Version Slot 2\n");
	printf("=======================================\n");
	Error = ACR122_GetFirmwareVersion(hReader,2,FW_Ver,&FW_Len);

	printf("Error Code: %d\n",Error);

	if(Error == 0)
	{
		printf("Firmware Code: ");
		for (i=0;i<FW_Len;i++)
			printf("%s",Temp+i);
		printf("\n");
	}
	printf("Press any key to continuou\n");
	getch();
	printf("\n");

	ACR122_Close(hReader);

	return 0;
}
```