

**EZ PC/SC Series  
Memory Card API  
Reference Manual  
v2.11**

# Table Of Contents

WARNING .....	4
About this manual.....	4
1. SLE4432/4442 Memory Card.....	5
SLE4442_Compare_Verification_Data .....	5
SLE4442_Read_Main_Memory.....	7
SLE4442_Update_Main_Memory .....	9
SLE4442_Update_Main_MemoryA .....	11
SLE4442_Read_Protection_Memory .....	13
SLE4442_Write_Protect_Memory.....	15
SLE4442_Read_Security_Memory.....	17
SLE4442_Update_Security_Memory.....	19
2. SLE4418/4428 Memory Card.....	21
SLE4428_PSC_Verification.....	21
SLE4428_Read_Data_Without_Protect_Bit .....	23
SLE4428_Write_Data_Without_Protect_Bit .....	25
SLE4428_Write_Data_Without_Protect_BitA .....	27
SLE4428_Read_Data_With_Protect_Bit.....	29
SLE4428_Write_Data_With_Protect_Bit.....	31
SLE4428_Write_Data_With_Protect_BitA .....	33
SLE4428_Write_Protect_Bit.....	35
3. SLE4404 Memory Card.....	37
SLE4404_Compare_User_Code.....	37
SLE4404_Read_Memory.....	39
SLE4404_Write_Memory.....	41
SLE4404_Write_MemoryA .....	43
SLE4404_Read_User_Memory .....	45
SLE4404_Write_User_Memory .....	47
SLE4404_Write_User_MemoryA .....	49
SLE4404_Erase_Memory.....	51
SLE4404_Compare_Memory_Code.....	53
SLE4404_Enter_Test_Mode.....	55
SLE4404_Exit_Test_Mode .....	56
SLE4404_Blow_Fuse .....	57

4.	SLE4406/4436/5536 Memory Card.....	58
	SLE4436_Read_Memory.....	58
	SLE4436_Read_Counter_Stages .....	60
	SLE4436_Write_Memory.....	62
	SLE4436_Write_Counter_Stage.....	64
	SLE4436_Reload .....	66
	SLE4436_Verify_Transport_Code.....	68
	SLE4436_Authentication.....	69
5.	I2C Memory Card .....	70
	I2C_Read_Memory.....	70
	I2C_Write_Memory .....	73
6.	Error Codes .....	75

## **WARNING**

Information in this document is subject to change without prior notice.

All trademarks mentioned are proprietary of their respective owners.

## **About this manual**

This manual describes the memory card API (Application Programming Interface) functions of EZ PC/SC series smart card reader; Application software developers should refer to this manual to develop their own software for read/write memory card via EZ PC/SC series smart card reader.

## 1. SLE4432/4442 Memory Card

### SLE4442\_Compare\_Verification\_Data

The **SLE4442\_Compare\_Verification\_Data** function will compare the entered data with the PSC(Personal Security Code) data in memory card. If success, all other operations of this memory card could work; otherwise, any operation would not work.

It only supports SLE4442 memory card, but SLE4432 does not.

```
LONG SLE4442_Compare_Verification_Data(  
    IN SCARDHANDLE hCard,  
    IN BYTE PSC1,  
    IN BYTE PSC2,  
    IN BYTE PSC3  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*PSC1*

The first byte of PSC is to be compared.

*PSC2*

The second byte of PSC is to be compared.

*PSC3*

The third byte of PSC is to be compared.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE PSC1, PSC2, PSC3;

    printf("    Compare Verify Data                ");

    PSC1 = 0x01;
    PSC2 = 0x02;
    PSC3 = 0x03;
    result = SLE4442_Compare_Verification_Data(ScardHandle,
                                                PSC1,
                                                PSC2,
                                                PSC3);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4442\_Read\_Main\_Memory

The **SLE4442\_Read\_Main\_Memory** function gets the data from the SLE4442 memory card.

It also supports SLE4432 memory card.

```
LONG SLE4442_Read_Main_Memory(  
    IN SCARDHANDLE hCard,  
    OUT LPBYTE pbRecvBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN OUT LPDWORD pcbRecvLength  
);
```

### Parameters

**hCard**

This is the reference value returned from **CasConnect**.

**pbRecvBuffer**

Points to a buffer that receives the returned data.

**aStartAddr**

Specify the address of the first byte of data that will be gotten. The range is between 0 to 255.

**pcbRecvLength**

Points to a DWORD that specifies the length of expected returned data and receives the actual number of bytes received from the memory card. The range of this value must be between 1 to 256.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE recvbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Read Main Memory (5 bytes)                ");
    startaddr = 0x60;
    length = 5;
    result = SLE4442_Read_Main_Memory(ScardHandle,
                                       recvbuffer,
                                       startaddr,
                                       &length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        if(length == 5)
            printf("Passed\n");
        else{
            printf("Fail\n");
        }
    }
}
```

## SLE4442\_Update\_Main\_Memory

The **SLE4442\_Update\_Main\_Memory function** updates the new data to memory card. This function will not check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address is not allowed to be updated.

It also supports SLE4432 memory card.

```
LONG SLE4442_Update_Main_Memory(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

hCard

This is the reference value returned from **CasConnect**.

pbTransmitBuffer

Points to a buffer that sends the data to the memory card.

aStartAddr

Specify the address of the first byte of data to be written. The range is between 0 to 255.

TransmitLength

Specifies the length of data that will be written. The range of this value must be between 1 to 256.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Update Main Memory (5 bytes)                ");
    startaddr = 0x60;
    sendbuffer[0] = 0x00;
    sendbuffer[1] = 0x01;
    sendbuffer[2] = 0x02;
    sendbuffer[3] = 0x03;
    sendbuffer[4] = 0x04;
    length = 5;
    result = SLE4442_Update_Main_Memory(ScardHandle[CurrentReader],
                                        sendbuffer,
                                        startaddr,
                                        length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
        return false;
    }else{
        printf("Passed\n");
    }
}
```

## SLE4442\_Update\_Main\_MemoryA

The **SLE4442\_Update\_Main\_MemoryA** function updates the new data to memory card. This function will check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address isn't allowed to be updated. In this condition, the function will return fail.

It also supports SLE4432 memory card.

```
LONG SLE4442_Update_Main_MemoryA(
    IN SCARDHANDLE hCard,
    IN LPBYTE pbTransmitBuffer,
    IN MM_ADDRESS aStartAddr,
    IN DWORD TransmitLength
);
```

### Parameters

#### *hCard*

This is the reference value returned from **CasConnect**.

#### *pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

#### *aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 255.

#### *TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 256.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Update Main Memory (5 bytes)                ");
    startaddr = 0x60;
    sendbuffer[0] = 0x00;
    sendbuffer[1] = 0x01;
    sendbuffer[2] = 0x02;
    sendbuffer[3] = 0x03;
    sendbuffer[4] = 0x04;
    length = 5;
    result = SLE4442_Update_Main_MemoryA(ScardHandle[CurrentReader],
                                          sendbuffer,
                                          startaddr,
                                          length);

    if(result != SCARD_M_SUCCESS && result !=
SCARD_M_CHECK_ERROR){
        printf("Fail\n");
        ErrorMessage(result);
        return false;
    }else{
        printf("Passed\n");
    }
}
```

## SLE4442\_Read\_Protection\_Memory

The **SLE4442\_Read\_Protection\_Memory** function gets the protected bits of the addresses with protect bit. The addresses is from address 0 to 31.

It also supports SLE4432 memory card.

```
LONG SLE4442_Read_Protection_Memory(  
    IN SCARDHANDLE hCard,  
    OUT LPBYTE pbRecvBuffer,  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbRecvBuffer*

Points to a buffer that receives the returned data. The buffer size must be larger than or equal to 4. The returned data is four bytes (equal to 32 bits).

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE recvbuffer[MAX_BUFFER_SIZE];

    printf("    Read Protection Memory (4 bytes)          ");

    result = SLE4442_Read_Protection_Memory(ScardHandle,
                                             recvbuffer);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4442\_Write\_Protect\_Memory

The **SLE4442\_Write\_Protect\_Memory function** will compare the input bytes with the data of the assigned address in memory card. If the same, the protected bit of the address will set to 0. If the protected bit is 0, the address will not be allowed to update any more. And the protected bit will never be set to 1.

It also supports SLE4432 memory card.

```
LONG SLE4442_Write_Protection_Memory(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

## Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 31.

TransmitLength

Specifies the length of data that will be written. The range of this value must be between 1 to 32.

## Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    //
    printf("    Write Protection Memory (2 byte)          ");
    startaddr = 0x03;
    length = 2;
    sendbuffer[0] = 0xaa;
    sendbuffer[1] = 0xaa;

    result = SLE4442_Write_Protection_Memory(ScardHandle[CurrentReader],
                                              sendbuffer,
                                              startaddr,
                                              length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4442\_Read\_Security\_Memory

The **SLE4442\_Read\_Security\_Memory** function gets the security data (PSC) from the SLE4442 memory card.

It only supports SLE4442 memory card, but SLE4432 does not.

```
LONG SLE4442_Read_Security_Memory(  
    IN SCARDHANDLE hCard,  
    OUT LPBYTE pbRecvBuffer,  
);
```

### Parameters

*hCard*

This is the reference value returned from **ScardConnect**. (See the **Microsoft MSDN**)

*pbRecvBuffer*

Points to a buffer that receives the returned data. The buffer size must be larger than or equal to 4. The returned data (PSC) is four bytes.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE recvbuffer[MAX_BUFFER_SIZE];

    printf("    Read Security Memory (4 bytes)          ");

    result = SLE4442_Read_Security_Memory(ScardHandle[CurrentReader],
                                           recvbuffer);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4442\_Update\_Security\_Memory

The **SLE4442\_Update\_Security\_Memory function** will update the security data (PSC) in memory card.

It only supports SLE4442 memory card, but SLE4432 does not.

```
LONG SLE4442_Update_Security_Memory(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 1 to 3.

TransmitLength

Specifies the length of data that will be written. The range of this value must be between 1 to 3.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Update Security Memory (3 bytes)           ");
    startaddr = 0x01;
    sendbuffer[0] = 0x02;
    sendbuffer[1] = 0x02;
    sendbuffer[2] = 0x02;

    length = 3;
    result = SLE4442_Update_Security_Memory(ScardHandle[CurrentReader],
                                             sendbuffer,
                                             startaddr,
                                             length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## 2. SLE4418/4428 Memory Card

### SLE4428\_PSC\_Verification

The **SLE4428\_PSC\_Verification function** will compare the entered data with the PSC data in memory card. If success, all other operations of this memory card could work; otherwise, any operation would not work.

It only supports SLE4428 memory card, but SLE4418 does not.

```
LONG SLE4428_PSC_Verification(  
    IN SCARDHANDLE hCard,  
    IN BYTE PSC1,  
    IN BYTE PSC2  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*PSC1*

The first byte of PSC is to be compared.

*PSC2*

The second byte of PSC is to be compared.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE PSC1, PSC2;

    printf("    PSC Verification                ");

    PSC1 = 0xff;
    PSC2 = 0xff;

    result = SLE4428_PSC_Verification(ScardHandle[CurrentReader],
                                      PSC1,
                                      PSC2);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4428\_Read\_Data\_Without\_Protect\_Bit

The **SLE4428\_Read\_Data\_Without\_Protect\_Bit** function gets the data from the SLE4428 memory card.

It also supports SLE4418 memory card.

```
LONG SLE4428_Read_Data_Without_Protect_Bit (  
    IN SCARDHANDLE hCard,  
    OUT LPBYTE pbRecvBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN OUT LPDWORD pcbRecvLength  
);
```

### Parameters

#### *hCard*

This is the reference value returned from **CasConnect**.

#### *pbRecvBuffer*

Points to a buffer that receives the returned data.

#### *aStartAddr*

Specify the address of the first byte of data that will be gotten. The range is between 0 to 1023.

#### *pcbRecvLength*

Points to a DWORD that specifies the length of expected returned data and receives the actual number of bytes received from the memory card. The range of this value must be between 1 to 1024.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE recvbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Read Memory Data (5 byte)                ");
    startaddr = 0x00;
    length = 5;
    result = SLE4428_Read_Data_Without_Protect_Bit(ScardHandle,
                                                    recvbuffer,
                                                    startaddr,
                                                    &length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        if(length == 1)
            printf("Passed\n");
        else
            printf("Fail\n");
    }
}
```

## SLE4428\_Write\_Data\_Without\_Protect\_Bit

The **SLE4428\_Write\_Data\_Without\_Protect\_Bit** function updates the new data to memory card. This function will not check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address is not allowed to be updated.

It also supports SLE4418 memory card.

```
LONG SLE4428_Write_Data_Without_Protect_Bit (
    IN SCARDHANDLE hCard,
    IN LPBYTE pbTransmitBuffer,
    IN MM_ADDRESS aStartAddr,
    IN DWORD TransmitLength
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 1023.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 1024.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Write Memory Data (256 byte)                ");
    startaddr = 0x00;
    for(i = 0; i < 256; ++i)
        sendbuffer[i] = (BYTE)i;    //data
    length = 256;
    result = SLE4428_Write_Data_Without_Protect_Bit(ScardHandle,
                                                    sendbuffer,
                                                    startaddr,
                                                    length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
        return false;
    }else{
        printf("Passed\n");
    }
}
```

## SLE4428\_Write\_Data\_Without\_Protect\_BitA

The **SLE4428\_Write\_Data\_Without\_Protect\_BitA** function updates the new data to memory card. This function will check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address isn't allowed to be updated. In this condition, the function will return fail.

It also supports SLE4418 memory card.

]

```
LONG SLE4428_Write_Data_Without_Protect_Bit (
    IN SCARDHANDLE hCard,
    IN LPBYTE pbTransmitBuffer,
    IN MM_ADDRESS aStartAddr,
    IN DWORD TransmitLength
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 1023.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 1024.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Write Memory Data (256 byte)                ");
    startaddr = 0x00;
    for(i = 0; i < 256; ++i)
        sendbuffer[i] = (BYTE)i;    //data
    length = 256;
    result = SLE4428_Write_Data_Without_Protect_BitA(ScardHandle,
                                                    sendbuffer,
                                                    startaddr,
                                                    length);

    if(result != SCARD_M_SUCCESS && result !=
    SCARD_M_CHECK_ERROR){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4428\_Read\_Data\_With\_Protect\_Bit

The **SLE4428\_Read\_Data\_With\_Protect\_Bit** function gets the data with protect bits from the SLE4428 memory card. Each address has its own protect bit. Once the protect bit of some address has written(set to 0), the address can not be written anymore, but read is always allowed.

It also supports SLE4418 memory card.

```
LONG SLE4428_Read_Data_With_Protect_Bit (  
    IN SCARDHANDLE hCard,  
    OUT LPBYTE pbRecvBuffer,  
    OUT LPBYTE pbProtectBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN OUT LPDWORD pcbRecvLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbRecvBuffer*

Points to a buffer that receives the returned data.

*pbProtectBuffer*

Points to a buffer that receives the protect bits of each address.

*aStartAddr*

Specify the address of the first byte of data that will be gotten. The range is between 0 to 1023.

*pcbRecvLength*

Points to a DWORD that specifies the length of expected returned data and receives the actual number of bytes received from the memory card. The range of this value must be between 1 to 1024.

## Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

### EXAMPLE:

```
#include <casmcard.h>
#include <stdio.h>
void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE recvbuffer[MAX_BUFFER_SIZE];
    BYTE protectbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Read Data with Protect Bit                ");
    startaddr = 0x25;
    length = 1;
    result = SLE4428_Read_Data_With_Protect_Bit(ScardHandle,
                                                recvbuffer,
                                                protectbuffer,
                                                startaddr,
                                                &length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        if(length == 1){
            printf("Passed\n");
        }else{
            printf("Fail\n");
        }
    }
}
```

## SLE4428\_Write\_Data\_With\_Protect\_Bit

The **SLE4428\_Write\_Data\_With\_Protect\_Bit** function updates the new data with protect bit written(set to 0) to memory card. This function will not check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address is not allowed to be updated.

It also supports SLE4418 memory card.

```
LONG SLE4428_Write_Data_With_Protect_Bit (  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 1023.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 1023.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Write Memory Data With Protect Bit (5 byte)  ");
    startaddr = 0x26;
    sendbuffer[0] = 0x64;    //data
    sendbuffer[1] = 0x65;
    sendbuffer[2] = 0x66;
    sendbuffer[3] = 0x67;
    sendbuffer[4] = 0x68;
    length = 5;
    result = SLE4428_Write_Data_With_Protect_Bit(ScardHandle,
                                                sendbuffer,
                                                startaddr,
                                                length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4428\_Write\_Data\_With\_Protect\_BitA

The **SLE4428\_Write\_Data\_With\_Protect\_BitA** function updates the new data with protect bit written (set to 0) to memory card. This function will check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address isn't allowed to be updated. In this condition, the function will return fail.

It also supports SLE4418 memory card.

```
LONG SLE4428_Write_Data_With_Protect_BitA (
    IN SCARDHANDLE hCard,
    IN LPBYTE pbTransmitBuffer,
    IN MM_ADDRESS aStartAddr,
    IN DWORD TransmitLength
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 1023.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 1024.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Write Memory Data With Protect Bit (5 byte)  ");
    startaddr = 0x26;
    sendbuffer[0] = 0x64;    //data
    sendbuffer[1] = 0x65;
    sendbuffer[2] = 0x66;
    sendbuffer[3] = 0x67;
    sendbuffer[4] = 0x68;
    length = 5;
    result = SLE4428_Write_Data_With_Protect_Bit A(ScardHandle,
                                                    sendbuffer,
                                                    startaddr,
                                                    length);

    if(result != SCARD_M_SUCCESS && result !=
SCARD_M_CHECK_ERROR){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4428\_Write\_Protect\_Bit

The **SLE4428\_Write\_Protect\_Bit function** will compare the data of the assigned address with entered data. If the same, the protect bit of the address will be written(set to 0).

It also supports SLE4418 memory card.

```
LONG SLE4428_Write_Protect_Bit (  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

## Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 1023.

TransmitLength

Specifies the length of data that will be written. The range of this value must be between 1 to 1024.

## Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    MM_ADDRESS startaddr;
    DWORD length;

    printf("    Write Protect Bit          ");

    startaddr = 0x25;
    sendbuffer[0] = recvbuffer[0];
    length = 1;
    result = SLE4428_Write_Protect_Bit(ScardHandle,
                                       sendbuffer,
                                       startaddr,
                                       length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
        return false;
    }else{
        printf("Passed\n");
    }
}
```

### 3. SLE4404 Memory Card

#### SLE4404\_Compare\_User\_Code

The **SLE4404\_Compare\_User\_Code function** will compare the entered data with the User Codes data in memory card. If success, the user memory without being written can be written once.

```
LONG SLE4442_Compare_User_Code(
    IN SCARDHANDLE hCard,
    IN BYTE Code1,
    IN BYTE Code2,
);
```

#### Parameters

##### *hCard*

This is the reference value returned from **CasConnect**.

##### *Code1*

The first byte of User Codes is to be compared.

##### *Code2*

The second byte of User Code is to be compared.

#### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE UserCode1, UserCode2;

    printf("    User Code Comparison                ");

    UserCode1 = 0xaa;
    UserCode2 = 0xaa;

    result = SLE4404_Compare_User_Code(ScardHandle,
                                       UserCode1,
                                       UserCode2);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4404\_Read\_Memory

The **SLE4404\_Read\_Memory** function gets the data from the SLE4404 memory card.

```
LONG SLE4404_Read_Memory(  
    IN SCARDHANDLE hCard,  
    OUT LPBYTE pbRecvBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN OUT LPDWORD pcbRecvLength  
);
```

### Parameters

hCard

This is the reference value returned from **CasConnect**.

pbRecvBuffer

Points to a buffer that receives the returned data.

aStartAddr

Specify the address of the first byte of data that will be gotten. The range is between 0 to 50.

pcbRecvLength

Points to a DWORD that specifies the length of expected returned data and receives the actual number of bytes received from the memory card. The range of this value must be between 1 to 51.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE recvbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Read Memory (26 bytes)                ");
    startaddr = 0x00;
    length = 26;
    result = SLE4404_Read_Memory(ScardHandle,
                                recvbuffer,
                                startaddr,
                                &length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        if(length == 26)
            printf("Passed\n");
        else
            printf("Fail\n");
    }
}
```

## SLE4404\_Write\_Memory

The **SLE4404\_Write\_Memory function** updates the new data to memory card. This function will not check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address is not allowed to be updated.

```
LONG SLE4404_Write_Memory(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 50.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 51.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Write Memory (26 bytes)           ");
    startaddr = 0x00;
    for(i = 0; i < 26; ++i)
        sendbuffer[i] = (BYTE)i;
    length = 26;
    result = SLE4404_Write_Memory(ScardHandle,
                                   sendbuffer,
                                   startaddr,
                                   length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4404\_Write\_MemoryA

The **SLE4404\_Write\_MemoryA** function updates the new data to memory card. This function will check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address isn't allowed to be updated. In this condition, the function will return fail.

```
LONG SLE4404_Write_MemoryA(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 50.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 51.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Write Memory with data check (26 bytes)          ");
    startaddr = 0x00;
    for(i = 0; i < 26; ++i)
        sendbuffer[i] = (BYTE)i;
    length = 26;
    result = SLE4404_Write_MemoryA(ScardHandle,
                                    sendbuffer,
                                    startaddr,
                                    length);

    if(result != SCARD_M_SUCCESS && result !=
SCARD_M_CHECK_ERROR){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4404\_Read\_User\_Memory

The **SLE4404\_Read\_User\_Memory** function gets data of user memory in the SLE4404 memory card.

```
LONG SLE4404_Read_User_Memory(  
    IN SCARDHANDLE hCard,  
    OUT LPBYTE pbRecvBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN OUT LPDWORD pcbRecvLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbRecvBuffer*

Points to a buffer that receives the returned data.

*aStartAddr*

Specify the address of the first byte of data that will be gotten. The range is between 0 to 25.

*pcbRecvLength*

Points to a DWORD that specifies the length of expected returned data and receives the actual number of bytes received from the memory card. The range of this value must be between 1 to 26.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE recvbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Read User Memory (26 bytes)                ");
    startaddr = 0x00;
    length = 26;
    result = SLE4404_Read_User_Memory(ScardHandle,
                                       recvbuffer,
                                       startaddr,
                                       &length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        if(length == 26)
            printf("Passed\n");
        else
            printf("Fail\n");
    }
}
```

## SLE4404\_Write\_User\_Memory

The **SLE4404\_Write\_User\_Memory function** updates the new data of user memory in the SLE4404 memory card. This function will not check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address is not allowed to be updated.

```
LONG SLE4404_Write_User_Memory(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 25.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 26.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Write User Memory (26 bytes)          ");
    startaddr = 0x00;
    for(i = 0; i < 26; ++i)
        sendbuffer[i] = (BYTE)i;
    length = 26;
    result = SLE4404_Write_User_Memory(ScardHandle,
                                        sendbuffer,
                                        startaddr,
                                        length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4404\_Write\_User\_MemoryA

The **SLE4404\_Write\_User\_MemoryA** function updates the new data of user memory in the SLE4404 memory card. This function will check if the data is really written in the memory card. Because it is possible that some address has been protected, the data of this address is not allowed to be updated.

```
LONG SLE4404_Write_User_MemoryA(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 25.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 26.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Write User Memory with data check (26 bytes)
");
    startaddr = 0x00;
    for(i = 0; i < 26; ++i)
        sendbuffer[i] = (BYTE)i;
    length = 26;
    result = SLE4404_Write_User_MemoryA(ScardHandle,
                                         sendbuffer,
                                         startaddr,
                                         length);

    if(result != SCARD_M_SUCCESS && result !=
SCARD_M_CHECK_ERROR){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4404\_Erase\_Memory

The **SLE4404\_Erase\_Memory** function erases the memory of any address in the SLE4404 memory card. This operation will work after **SLE4404\_Compare\_Memory\_Code**. After this operation, the memory that have been erased can be written once.

```
LONG SLE4404_Erase_Memory(  
    IN SCARDHANDLE hCard,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 50.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 51.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Erase Memory Counter (64 bits)           ");
    startaddr = 0x2c;
    length = 8;
    result = SLE4404_Erase_Memory(ScardHandle,
                                   startaddr,
                                   length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4404\_ Compare\_Memory\_Code

The **SLE4404\_ Compare\_Memory\_Code** function will compare the entered data with the Memory Codes data in memory card. If success, any address in the memory can be updated.

```
LONG SLE4404_ Compare_Memory_Code(
    IN SCARDHANDLE hCard,
    IN BYTE Code1,
    IN BYTE Code2,
    IN BYTE Code3,
    IN BYTE Code4
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*Code1*

The first byte of Memory Codes is to be compared.

*Code2*

The second byte of Memory Code is to be compared.

*Code3*

The third byte of Memory Codes is to be compared.

*Code4*

The forth byte of Memory Code is to be compared.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE MemoryCode1;
    BYTE MemoryCode2;
    BYTE MemoryCode3;
    BYTE MemoryCode4;

    printf("    Memory Code Comparison                ");

    MemoryCode1 = 0xff;
    MemoryCode2 = 0xff;
    MemoryCode3 = 0xff;
    MemoryCode4 = 0xff;

    result = SLE4404_Compare_Memory_Code(ScardHandle[CurrentReader],
                                         MemoryCode1,
                                         MemoryCode2,
                                         MemoryCode3,
                                         MemoryCode4);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4404\_Enter\_Test\_Mode

The **SLE4404\_Enter\_Test\_Mode** function will enter the test mode. During test mode, the entire EEPROM area, with the exception of the manufacturer ROM, can be changed and thus newly configured.

```
LONG SLE4404_Enter_Test_Mode(
    IN SCARDHANDLE hCard
);
```

## Parameters

*hCard*

This is the reference value returned from **CasConnect**.

## Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

## EXAMPLE:

```
#include <casmpcard.h>
```

```
#include <stdio.h>
```

```
void Test(SCARDHANDLE ScardHandle)
```

```
{
    long result;
    printf("  Enter Test Mode          ");
    result = SLE4404_Enter_Test_Mode(ScardHandle);
    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else
        printf("Passed\n");
}
```

## SLE4404\_Exit\_Test\_Mode

The **SLE4404\_Exit\_Test\_Mode** function will exit the test mode.

```
LONG SLE4404_Exit_Test_Mode(
    IN SCARDHANDLE hCard
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

### EXAMPLE:

```
#include <casocard.h>
```

```
#include <stdio.h>
```

```
void Test(SCARDHANDLE ScardHandle)
```

```
{
```

```
    long result;
```

```
    printf("    Exit Test Mode                ");
```

```
    result = SLE4404_Exit_Test_Mode(ScardHandle);
```

```
    if(result != SCARD_M_SUCCESS){
```

```
        printf("Fail\n");
```

```
        ErrorMessage(result);
```

```
    }else{
```

```
        printf("Passed\n");
```

```
    }
```

```
}
```

## SLE4404\_Blow\_Fuse

The **SLE4404\_Blow\_Fuse** function will blow the fuse.

```
LONG SLE4404_Blow_Fuse(
    IN SCARDHANDLE hCard
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

### EXAMPLE:

```
#include <casmcard.h>
#include <stdio.h>
```

```
void Test(SCARDHANDLE ScardHandle)
{
    long result;

    printf("    Enter Test Mode                ");
    result = SLE4404_Blow_Fuse(ScardHandle);
    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## 4. SLE4406/4436/5536 Memory Card

### SLE4436\_Read\_Memory

The **SLE4436\_Read\_Memory function** reads the data from the SLE4436 memory card. It also supports SLE4406/5536 memory card.

```
LONG SLE4436_Read_Memory(
    IN SCARDHANDLE hCard);
    OUT LPBYTE pbRecvBuffer,
    IN MM_ADDRESS aStartAddr,
    IN OUT LPDWORD pcbRecvLength
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbRecvBuffer*

Points to a buffer that receives the returned data.

*aStartAddr*

Specify the address of the first byte of data that will be gotten. The range is between 0 to 47.

*pcbRecvLength*

Points to a DWORD that specifies the length of expected returned data and receives the actual number of bytes received from the memory card. The range of this value must be between 1 to 48.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE recvbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Read All Memory (48 bytes)                ");
    startaddr = 0x00;
    length = 48;
    result = SLE4436_Read_Memory(ScardHandle,
                                recvbuffer,
                                startaddr,
                                &length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        if(length == 48){
            printf("Passed\n");
        }else{
            printf("Fail\n");
        }
    }
}
```

## SLE4436\_Read\_Counter\_Stages

The **SLE4436\_Read\_Counter\_Stages function** reads all counter stages (five stages) from the SLE4436 memory card.

It also supports SLE4406/5536 memory card.

```
LONG SLE4436_Read_Counter_Stages(  
    IN SCARDHANDLE hCard);  
    OUT BYTE Stage[5]  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*Stage[5]*

Array of five bytes that receives the five stages data.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE Stages[5];
    DWORD length;

    printf("    Read Counter Stages (5 bytes)           ");
    result = SLE4436_Read_Counter_Stages(ScardHandle,
                                         Stages);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4436\_Write\_Memory

The **SLE4436\_Write\_Memory function** writes the new data to memory card. It also supports SLE4406/5536 memory card.

```
LONG SLE4404_Write_Memory(  
    IN SCARDHANDLE hCard,  
    IN LPBYTE pbTransmitBuffer,  
    IN MM_ADDRESS aStartAddr,  
    IN DWORD TransmitLength  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*aStartAddr*

Specify the address of the first byte of data to be written. The range is between 0 to 47.

*TransmitLength*

Specifies the length of data that will be written. The range of this value must be between 1 to 48.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    DWORD length;

    printf("    Write Data Area (2 bytes)                ");
    startaddr = 0x0e;
    sendbuffer[0] = 0xfe;
    sendbuffer[1] = 0xf0;
    length = 2;
    result = SLE4436_Write_Memory(ScardHandle,
                                   sendbuffer,
                                   startaddr,
                                   length);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4436\_Write\_Counter\_Stage

The **SLE4436\_Write\_Counter\_Stage function** writes the value to the assigned stage of the SLE4436 memory card.

It also supports SLE4406/5536 memory card.

```
LONG SLE4436_Write_Counter_Stage(  
    IN SCARDHANDLE hCard);  
    IN BYTE Stage,  
    IN BYTE Data  
);
```

### Parameters

#### *hCard*

This is the reference value returned from **CasConnect**.

#### *Stage*

Indicates which stage will be written.

#### *Data*

The value that will be written into the assigned stage.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE stages;
    BYTE data;
    DWORD length;

    printf("    Write Counter Stage          ");
    stage = 1;
    data = 0xfe;
    result = SLE4436_Write_Counter_Stage(ScardHandle,
                                         stage,
                                         data);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## SLE4436\_Reload

The **SLE4436\_Reload function** erases all the bits of stage 1 and writes a bit of higher stage of the SLE4436 memory card.

It also supports SLE4406/5536 memory card.

```
LONG SLE4436_Reload(  
    IN SCARDHANDLE hCard);  
    IN BYTE BitAddr  
);
```

## Parameters

### *hCard*

This is the reference value returned from **CasConnect**.

### *BitAddr*

Indicates which bit will be written. The range is between 0 and 31.

0 - 7 is in stage 2

8 - 15 is in stage 3

16 - 23 is in stage 4

24 - 31 is in stage 5

## Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmcard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    BYTE bitAddr;

    printf("    Reload Counter Stages                ");
    bitAddr = 4;
    result = SLE4436_Reload(ScardHandle,
                            bitAddr);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

---

## SLE4436\_Verify\_Transport\_Code

The **SLE4436\_Verify\_Transport\_Code** function will compare the entered data with the transport codes in memory card..

It only supports SLE4436/5536 memory card, but SLE4406 does not.

```
LONG SLE4436_Verify_Transport_Code(  
    IN SCARDHANDLE hCard,  
    IN BYTE Code1,  
    IN BYTE Code2,  
    IN BYTE Code3  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*Code1*

The first byte to be compared.

*Code2*

The second byte to be compared.

*Code3*

The third byte to be compared.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

## SLE4436\_Authentication

The **SLE4436\_Authentication function** will compare the entered data with the PSC(Personal Security Code) data in memory card. If success, all other operations of this memory card could work; otherwise, any operation would not work.

It only supports SLE4436/5536 memory card, but SLE4406 does not.

```
LONG SLE4436_Authentication(
    IN SCARDHANDLE hCard,
    OUT BYTE ReturnCode[2],
    IN BYTE Key,
    IN BYTE ClockPulse,
    IN BYTE Code[6]
);
```

## Parameters

### *hCard*

This is the reference value returned from **CasConnect**.

### *ReturnCode[2]*

Array of two bytes that receives the return codes from the memory card.

### *Key*

The value that be sent into the memory card.

### *ClockPulse*

### *Code[6]*

Array of size bytes to be verified in the memory card.

## Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

## 5. I2C Memory Card

### I2C\_Read\_Memory

The **I2C\_Read\_Memory** function reads the data from the I2C memory card.

```
LONG I2C_Read_Memory(  
    IN SCARDHANDLE hCard,  
    IN BYTE ReadCommand,  
    IN MM_ADDRESS aStartAddr,  
    OUT LPBYTE pbRecvBuffer,  
    IN OUT LPDWORD pcbRecvLength,  
    IN BYTE Dummy,  
    IN BYTE AddrByteNum  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*ReadCommand*

Indicates the reading command of the I2C memory card.

*aStartAddr*

Specify the address of the first byte of data to be written

*pbRecvBuffer*

Points to a buffer that receives the returned data.

*pcbReturnLength*

Points to a DWORD that specifies the length of expected returned data and receives the actual number of bytes received from the memory card.

*Dummy*

Indicates if needs to use dummy write address. 0 is not used; otherwise, dummy is used.

*AddrByteNum*

Indicates the number of the address bytes. The value is between 0 to 2.

## Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

### EXAMPLE:

```
#include <casmscard.h>
#include <stdio.h>
```

```
void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE recvbuffer[MAX_BUFFER_SIZE];
    BYTE Command;
    BYTE AddrByteNum;
    BYTE Dummy;
    DWORD length;

    printf("    Read I2C Memory (5 bytes)                ");
    Command = 0xA1;
    startaddr = 0x00;
    length = 5;
    Dummy = 0x01;
    AddrByteNum = 0x01;
    result = I2C_Read_Memory(ScardHandle[CurrentReader],
                             Command,
                             startaddr,
                             recvbuffer,
                             &length,
                             Dummy,
                             AddrByteNum);
    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }
}
```

```
}else{  
    if(length == 5)  
        printf("Passed\n");  
    else  
        printf("Fail\n");  
}  
}
```

## I2C\_Write\_Memory

The **I2C\_Write\_Memory** function writes the data to the I2C memory card.

```
LONG I2C_Write_Memory(  
    IN SCARDHANDLE hCard,  
    IN BYTE WriteCommand,  
    IN MM_ADDRESS aStartAddr,  
    OUT LPBYTE pbTransmitBuffer,  
    IN DWORD TransmitLength,  
    IN BYTE AddrByteNum  
);
```

### Parameters

*hCard*

This is the reference value returned from **CasConnect**.

*WriteCommand*

Indicates the writing command of the I2C memory card.

*aStartAddr*

Specify the address of the first byte of data to be written

*pbTransmitBuffer*

Points to a buffer that sends the data to the memory card.

*TransmitLength*

Specifies the length of data that will be written.

*AddrByteNum*

Indicates the number of the address bytes. The value is between 0 to 2.

### Return Values

If the function...	The return value is...
Succeeds	SCARD_M_SUCCESS
Fails	An error code (see <b>Error Codes</b> for a list of all error codes).

**EXAMPLE:**

```
#include <casmscard.h>
#include <stdio.h>

void Test(SCARDHANDLE ScardHandle)
{
    long result;
    MM_ADDRESS startaddr;
    BYTE sendbuffer[MAX_BUFFER_SIZE];
    BYTE Command;
    BYTE AddrByteNum;
    DWORD length;

    printf("    Write I2C Memory (5 bytes)                ");
    Command = 0xA0;
    startaddr = 0x00;
    for(i = 0; i < 5; ++i)
        sendbuffer[i] = (BYTE)i;
    length = 5;
    AddrByteNum = 0x01;
    result = I2C_Write_Memory(ScardHandle,
                              Command,
                              startaddr,
                              sendbuffer,
                              length,
                              AddrByteNum);

    if(result != SCARD_M_SUCCESS){
        printf("Fail\n");
        ErrorMessage(result);
    }else{
        printf("Passed\n");
    }
}
```

## 6. Error Codes

<b>Error Values</b>	<b>Description</b>
SCARD_M_SUCCESS	Operation success
SCARD_M_CARD_ABSENT	The card is not in the reader
SCARD_M_NO_RESPONSE	The card does not response any thing
SCARD_M_POWER_FAIL	Power the card fail
SCARD_M_COMM_ERROR	Communication error
SCARD_M_VERIFY_FAIL	Verify codes fail
SCARD_M_TYPE_ERROR	Card Type Error
SCARD_M_COMMAND_ERROR	The command to the card Error
SCARD_M_COUNTER_EMPTY	Counter of card is empty
SCARD_M_CARD_LOCKED	The card is locked
SCARD_M_WRITE_ERROR	Write to the card error
SCARD_M_CHECK_ERROR	Check if previously written data is really written
SCARD_M_OTHER_FAIL	Other fail