User Manual

Advantech ADAM series

Software Manual

Document History:

Doc Version	**Date**	**Comment**
1.0	2015/02/03	First edition

Chapter 1

1. Introduction

1.1. About This Manual

This is the Software Manual for the Advantech ADAM- 5550, APAX- 5560 products. Advantech provides ADAM library which allows developers and end users to connected I/O modules, perform configurations, and simple testing of the I/O.

This manual supplies information about Advantech ADAM I/O modules, including calling procedure of operating device and descriptions of each function, parameter and data structure.

This manual does not show you how to solve every possible programming problem. To use this manual, you should already be familiar with at least one of the supported programming environments and Windows 2000/XP/Vista/7/Embedded Standard/CE.

1.2. Organization of This Manual

This user manual is divided into the following sections:

- User functions
- Internal functions

User functions

This section provides information about user functions and how to use them.

Internal functions

This section provides information about internal functions.

It's highly recommended NOT to use these functions in your programs.

Functions Reference

- **Call flow**

  This section provides call flow and information about how to use functions.

- **Error Codes**

  This section describes system error codes.

- **Analog I/O Board Settings**
  This section provides range settings for Analog I/O Boards.

1.3. Installation

In order to save your development time, Advantech provides several examples that you can use it as reference to build your own C/C++, C# or VB application program. The default installation directory is **C:/Program Files (x86)/Advantech/AdamApax.NET Class Library** and all examples can be found in /**Sample Code** after installing AdamApax .NET Class Library from Advantech website at http://www.advantech.com in the download area under Support page. For Windows XP users, the examples are under “ADAM/Win32”. For Windows CE users, the examples are under “ADAM/WinCE”. The sample programs are all build with “Microsoft Visual Studio 2008” for Windows XP and Windows CE.

Chapter 2

2. User functions

2.1. ADAMDrvOpen

Users can use this function to open ADAM device. The function returns a handle that can be used to access the ADAM device.

Syntax

ADAMDrvOpen(
  LONG *handle
);

Parameters

Name	Direction	Description
handle	Output	[out] The driver handler.

Return Value

If driver initialization succeeded, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Remarks

Use the **ADAMDrvClose** function to close the ADAM device.

Example

```
LONG lDriverHandle = NULL; /* Driver handler */
if(ERR_SUCCESS != ADAMDrvOpen(&lDriverHandle))
    printf("Fail to open driver\n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5013_17_18

2.2. ADAMDrvClose

Close the ADAM device by calling this function when operation is completed.

Syntax

ADAMDrvClose (

  LONG *handle
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.

Return Value

If driver termination succeeded, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
if(NULL != lDriverHandle) {
    ADAMDrvClose(&lDriverHandle);
    lDriverHandle = NULL;
}
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5013_17_18

2.3. AI_GetChannelEnabled

Users can use this function to get the channel status (enable/disable)

Syntax

AI_GetChannelEnabled (
  LONG handle,
  WORD i_wSlot,
  BOOL* o_iEnableStatus
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_iEnableStatus	Output	[out] The variable to hold channel status. ***Note:**

		If the value is 1, it means that the channel is enabled; 0 for disabled.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_iEnableStatus** contains burnout function status from channel-0 to the last channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelNum = 8; /* Set channel number */
WORD wSlotID = 1; /* Slot ID */
BOOL bChEnabled[iChannelNum];
LONG lResult = AI_GetChannelEnabled (lDriverHandle, wSlotID,
bChEnabled);
if (ERR_SUCCESS == lResult) {
    for (int iCnt = 0; iCnt < iChannelNum; iCnt++) {
        if (bChEnabled[iCnt]) {
            printf("The channel %d is enabled.\n", iCnt);
            Sleep(100);
        }
        else
            printf("The channel %d is disabled.\n", iCnt);
    }
}
else
    printf("Fail to get channel status \n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5013_17_18

2.4. AI_GetChannelTotal

Users can use this function to get the total number of channels.

Syntax

AI_GetChannelTotal (
  LONG handle,

  WORD i_wSlot,
  WORD* o_wChTotal
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_wChTotal	Output	[out] The variable to hold the total number of channels

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_wChTotal** contains burnout value. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChTotal = 0; /* The total number of channels */
LONG lResult = AI_GetChannelTotal (lDriverHandle, wSlotID, & wChTotal);
if (ERR_SUCCESS == lResult)
    printf("The total number of channels is %d.", wChTotal);
else
    printf("Fail to get total number of channels \n");
```

2.5. AI_GetCjcValue

Users can use this function to get the CJC value of the indicated slot.

Syntax

AI_GetCjcValue(
  LONG handle,
  WORD i_wSlot,
  float* o_fValue
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_fValue	Output	[out] The variable to hold the CJC values.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**; **o_fValue** contains CJC values. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelNum = 8; /* Set channel number */
WORD wSlotID = 1; /* Slot ID */
float fValue[iChannelNum] = 0; /* CJC value */
LONG lResult = AI_GetCjcValue (lDriverHandle, wSlotID, fValue);
if (ERR_SUCCESS == lResult) {
    for (int iCnt = 0; iCnt < iChannelNum; iCnt++) {
        printf("[Ch%d] The CJC value is %f.\n", iCnt, fValue[iCnt]);
    }
    Sleep(3000);
}
else
    printf("Fail to get CJC value\n");
```

2.6. AI_GetInputRange

Users can use this function to get the range code of the indicated slot and channel.

Syntax

AI_GetInputRange (
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BYTE* o_byRange
);

Parameters

Name	Direction	Description

handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_byRange	Output	[out] The variable to hold range code. ***Note:** See **APPENDIX C** for more detailed information.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_byRange** contains the range code of the indicated slot and channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1; /* Channel ID */
BYTE byRange = 0; /* Range code */
LONG lResult = AI_GetInputRange(lDriverHandle, wSlotID, wChannel,
&byRange);
if (ERR_SUCCESS == lResult) {
    printf("The channel range code is %d.\n", byRange);
}
else
    printf("Fail to get channel range code\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5013_17_18

2.7. AI_GetRangeIntegrationTime

Users can use this function to get AI integration time of the indicated slot.

Syntax

AI_GetRangeIntegrationTime (
  LONG handle,
  WORD i_wSlot,
  WORD *o_wIntegration
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_wIntegration	Output	[out] The variable to hold AI integration time (Hz).

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wIntegration = 0; /* Hz */
if(ERR_SUCCESS == AI_SetIntegrationTime (lDriverHandle, wSlotID, 
&wIntegration))
    printf("The AI integration time is %d (Hz)\n", wIntegration);
```

2.8. AI_GetValues

Users can use this function to get all AI value of the indicated slot.

Syntax

AI_GetValues(
  LONG handle,
  WORD i_wSlot,
  float* o_fValues
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_fValues	Output	[out] The variable to hold the returned AI value.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_wValues** contains AIO values from channel-0 to the last channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelNum = 8; /* Set channel number */
WORD wSlotID = 1; /* Slot ID */
float fValue[iChannelNum] = {0};
LONG lResult = AIO_GetValues(lDriverHandle, wSlotID, fValue);

if (ERR_SUCCESS == lGetValueResult) {
    for (int iCnt =0; iCnt < iChannelNum ; iCnt++) {
        printf("[Ch %d] AI value is %f\n", iCnt, fValue[iCnt]);
    }
else
    printf("Fail to get values\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5013_17_18

2.9. AI_SetChannelEnabled

Users can use this function to set the channel status (enable/disable)

Syntax

AI_SetChannelEnabled (
  LONG handle,
  WORD i_wSlot,
  BOOL* i_iEnableStatus
);

Parameters

Name	**Direction**	**Description**
Handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_iEnableStatus	Input	[in] The channel status. ***Note:** Set value to 1 for “Enable”; 0 for “Disabled”.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **i_iEnableStatus** contains burnout function status from channel-0 to the last channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelNum = 8; /* Set channel number */
WORD wSlotID = 1; /* Slot ID */
BOOL bChEnabled[iChannelNum] = {0}; /*Disable all*/
if (ERR_SUCCESS != AI_SetChannelEnabled (lDriverHandle, wSlotID,
bChEnabled)
    printf("Fail to set channel status \n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5013_17_18

2.10. AI_SetCJCOffset

Users can use this function to set the CJC offset of the indicated slot.

Syntax

```
AI_SetCJCOffset (
  LONG handle,
  WORD i_wSlot,
  float i_fOffset
);
```

Parameters

Name	Direction	Description
Handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_fOffset	Input	[in] The offset value to be set.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
float fValue = 0.5; /* CJC offset value */

if(ERR_SUCCESS != AI_GetChValOffset (lDriverHandle, wSlotID, fValue))
    printf("Fail to set CJC offset.\n");
```

2.11. AI_SetInputRange

Users can use this function to set the range code of the indicated slot and channel.

Syntax

AI_SetInputRange (
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BYTE* i_byRange
);

Parameters

Name	**Direction**	**Description**
Handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
i_byRange	Input	[in] The range code. ***Note:** See **APPENDIX C** for more detailed information.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1; /* Channel ID */
```

```
BYTE byRange = 8; /* -10 ~ 10 V */
if (ERR_SUCCESS != AI_SetInputRange(lDriverHandle, wSlotID, wChannel,
&byRange)
    printf("Fail to set channel range.\n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5013_17_18

2.12. AI_SetRangeIntegrationTime

Users can use this function to set AI integration time of the indicated slot.

Syntax

AI_SetRangeIntegrationTime (
 LONG handle,
 WORD i_wSlot,
 WORD i_wIntegration
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wIntegration	Input	[in] The AI integration time (Hz).

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wIntegration = 20; /* Hz */
if(ERR_SUCCESS != AI_SetIntegrationTime (lDriverHandle, wSlotID,
wIntegration))
    printf("Fail to set AI integration time.\n");
```

2.13. AO_GetChannelTotal

Users can use this function to get the total number of channels.

Syntax

AO_GetChannelTotal (
  LONG handle,
  WORD i_wSlot,
  WORD* o_wChTotal
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_wChTotal	Output	[out] The variable to hold the total number of channels

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_wChTotal** contains burnout value. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChTotal = 0; /* The total number of channels */
LONG lResult = AO_GetChannelTotal (lDriverHandle, wSlotID, & wChTotal);
if (ERR_SUCCESS == lResult)
    printf("The total number of channels is %d.", wChTotal);
else
    printf("Fail to get total number of channels \n");
```

2.14. AO_GetOutputRange

Users can use this function to get the range code of the indicated slot and channel.

Syntax

AO_GetOutputRange (
  LONG handle,

WORD i_wSlot,
WORD i_wChannel,
BYTE* o_byRange
);

Parameters

Name	Direction	Description
Handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_byRange	Output	[out] The variable to hold range code. ***Note:** See **APPENDIX C** for more detailed information.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_byRange** contains the range code of the indicated slot and channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1; /* Channel ID */
BYTE byRange = 0; /* Range code */
LONG lResult = AO_GetOutputRange (lDriverHandle, wSlotID, wChannel,
&byRange);
if (ERR_SUCCESS == lResult) {
    printf("The channel range code is %d.\n", byRange);
}
else
    printf("Fail to get channel range code\n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5024

2.15. AO_GetValue

Users can use this function to get AO value of the indicated slot and channel.

Syntax

AO_GetValue(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  float* o_fValue
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_fValue	Output	[out] The variable to hold the returned AO value.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_wValue** contains AO value of the indicated slot and channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
float fValue = 0.0;
LONG lResult = AO_GetValue(lDriverHandle, wSlotID, wChannelID, &fValue);

if (ERR_SUCCESS == lGetValueResult) {
        printf("[Ch %d] AI value is %f\n", wChannelID, fValue);
else
    printf("Fail to get AO value\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5024

2.16. AO_GetValues

Users can use this function to get all AO value of the indicated slot.

Syntax

AO_GetValues(
  LONG handle,
  WORD i_wSlot,
  float* o_fValues
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_fValues	Output	[out] The variable to hold the returned AO value.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_wValues** contains AO values from channel-0 to the last channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelNum = 8; /* Set channel number */
WORD wSlotID = 1; /* Slot ID */
float fValue[iChannelNum] = {0};
LONG lResult = AO_GetValues(lDriverHandle, wSlotID, fValue);

if (ERR_SUCCESS == lGetValueResult) {
    for (int iCnt =0; iCnt < iChannelNum ; iCnt++) {
        printf("[Ch %d] AO value is %f\n", iCnt, fValue[iCnt]);
    }
else
    printf("Fail to get values\n");
```

2.17. AO_SetOutputRange

Users can use this function to set the range code of the indicated slot and channel.

Syntax

AO_SetOutputRange (
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BYTE* i_byRange
);

Parameters

Name	Direction	Description
Handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
i_byRange	Input	[in] The range code. ***Note:** See **APPENDIX C** for more detailed information.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1; /* Channel ID */
BYTE byRange = 0x32; /* 0 ~ 10 V */
if (ERR_SUCCESS != AO_SetOutputRange (lDriverHandle, wSlotID, wChannel,
&byRange)
    printf("Fail to set channel range.\n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5024

2.18. AO_SetValue

Users can use this function to get AO value of the indicated slot and channel.

Syntax

AO_SetValue(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  float i_fValue
);

Parameters

Name	Direction	Description
Handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
i_fValue	Input	[in] The AO value to be set.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
float fValue = 1.5;

if (ERR_SUCCESS != AO_SetValue (lDriverHandle, wSlotID, wChannelID,
fValue))
    printf("Fail to get AO value\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5024

2.19. AO_SetValues

Users can use this function to set all AO value of the indicated slot.

Syntax

AO_SetValues(
  LONG handle,

WORD i_wSlot,
float* i_fValues
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_fValues	Input	[in] The AO value to be set.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelNum = 8; /* Set channel number */
WORD wSlotID = 1; /* Slot ID */
float fValue[iChannelNum] = {0};

if (ERR_SUCCESS != AO_SetValues (lDriverHandle, wSlotID, fValue))
    printf("Fail to set AO values\n");
```

2.20. CNT_ClearAlarmFlag

Users can use this function to clear the counter alarm of the indicated slot and channel.

Syntax

CNT_ClearAlarmFlags(
LONG handle,
WORD i_wSlot,
WORD i_wChannel
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.

i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */

if(ERR_SUCCESS != CNT_ClearAlarmFlag(lDriverHandle, wSlotID,
wChannelID))
    printf("Fail to clear counter alarm \n");
```

2.21. CNT_ClearOverflow

Users can use this function to clear the counter overflow of the indicated slot and channel.

Syntax

CNT_ClearOverflow(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
if(ERR_SUCCESS != CNT_ClearOverflow(lDriverHandle, wSlotID,
wChannelID))
    printf("Fail to clear overflow\n");
```

2.22. CNT_ClearValue

Users can use this function to clear the counter value to startup value of the indicated slot and channel.

Syntax

CNT_ClearValue(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
if(ERR_SUCCESS != CNT_ClearValue (lDriverHandle, wSlotID, wChannelID))
    printf("Fail to clear counter value\n");
```

2.23. CNT_GetAlarmEnable

Users can use this function to get the counter alarm status of the indicated slot and channel.

Syntax

CNT_GetAlarmEnable (
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BOOL* o_iEnableStatus
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_iEnableStatus	Output	[out] The variable to hold channel status. ***Note:** If the value is 1, it means that the channel is enabled; 0 for disabled.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
BOOL bChEnabled = 0;
LONG lResult = CNT_GetAlarmEnable(lDriverHandle, wSlotID, wChannelID,
&bChEnabled);
if (ERR_SUCCESS == lResult) {
    if (bChEnabled) {
            printf("The channel %d is enabled.\n", wChannelID);
    else
            printf("The channel %d is disabled.\n", wChannelID);
    }
}
```

2.24. CNT_GetAlarmFlag

Users can use this function to get the counter alarm of the indicated slot and channel.

Syntax

CNT_GetAlarmFlag(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BOOL* o_iFlag
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_iFlag	Output	[out] The variable to hold the counter alarm flag.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
BOOL iFlag = 0;

if(ERR_SUCCESS == CNT_GetAlarmFlag(lDriverHandle, wSlotID, wChannelID ,
&iFlag))
    printf("[Ch%d] The alarm flag is %d\n", wChannelID, iFlag);
```

2.25. CNT_GetAlarmLimit

Users can use this function to get the limit count number of the indicated slot and channel.

Syntax

CNT_GetAlarmLimit(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  DWORD * o_dwLimit
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_dwLimit	Output	[out] The variable to hold the limit count number.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
DWORD dwLimit = 0;

if(ERR_SUCCESS == CNT_GetAlarmLimit (lDriverHandle, wSlotID, wChannelID ,
&dwLimit))
    printf("[Ch%d] The limit count number is %d\n", wChannelID, dwLimit);
```

2.26. CNT_GetAlarmMap

Users can use this function to get the mapping channel index of the indicated slot and channel.

Syntax

CNT_GetAlarmMap(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BYTE * o_byMap
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_byMap	Output	[out] The variable to hold the mapping channel index.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
BYTE byMap = 0;

if(ERR_SUCCESS == CNT_GetAlarmMap (lDriverHandle, wSlotID, wChannelID ,
& byMap))
    printf("[Ch%d] The mapping channel index =%d\n", wChannelID, byMap);
```

2.27. CNT_GetAlarmType

Users can use this function to get the local alarm type of the indicated slot and channel.

Syntax

CNT_GetAlarmType(
  LONG handle,
  WORD i_wSlot,

WORD i_wChannel,
BYTE * o_byType
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_byType	Output	[out] The variable to hold the local alarm type. ***Note:** 1 for “High alarm”; 0 for “Low alarm”.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannelID = 1; /* Channel ID */
BYTE byType = 0;

if(ERR_SUCCESS == CNT_GetAlarmType (lDriverHandle, wSlotID, wChannelID ,
& byType)) {
    if (byType)
        printf("[Ch%d] The alarm type is High alarm \n"wChannelID)
    else
        printf("[Ch%d] The alarm type is Low alarm \n"wChannelID)
}
```

2.28. CNT_GetDoValue

Users can use this function to get the DO value of the indicated slot and channel.

Syntax

CNT_GetDoValue (
LONG handle,

  WORD i_wSlot,
  WORD i_wChannel,
  BOOL* o_bValue
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID
o_bValue	Output	[out] The variable to hold the DO value.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_bValue** contains DO value. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1;
BOOL bHoldValue = false;
LONG lResult = CNT_GetDoValue (lDriverHandle, wSlotID, wChannel,
&bHoldValue);
if (ERR_SUCCESS == lResult) {
    if (bHoldValue)
        printf("Channel %d is true.\n", wChannel);
    else
        printf("Channel %d is false.\n",wChannel);
}
else
    printf("Fail to get DO single channel value \n");
```

2.29. CNT_GetDoValues

Users can use this function to get all DO values of the indicated slot.

Syntax

CNT_GetDoValues (

LONG handle,
WORD i_wSlot,
BYTE* o_bValue
);

Parameters

Name	Direction	Description
Handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_bValue	Output	[out] The variable to hold all the DO values.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_bValue** contains DO value. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelCNTNum = 4;
WORD wSlotID = 1; /* Slot ID */
BYTE bHoldValue = 0;
LONG lResult = CNT_GetDoValues (lDriverHandle, wSlotID, &bHoldValue);
if (ERR_SUCCESS == lResult) {
    for(int iCnt = 0; iCnt < iChannelCNTNum ; iCnt++) {
        if (bHoldValue & (0x01 << iCnt))
            printf("Channel %d is true.\n", iCnt);
        else
            printf("Channel %d is false.\n", iCnt);
    }
}
else
    printf("Fail to get all DO channel values \n");
```

2.30. CNT_GetFilter

Users can use this function to get the counter filter width of the indicated slot.

Syntax

CNT_GetFilter(

  LONG handle,
  WORD i_wSlot,
  DWORD* o_dwWidth
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_dwWidth	Output	[out] The variable to hold the counter digital filter width(us).

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
DWORD dwWidth = 0; /* filter width */
LONG lResult = CNT_GetFilter(lDriverHandle, wSlotID, &dwWidth);
if (ERR_SUCCESS == lResult) {
    printf("The CNT filter width is %d (us).\n", dwWidth);
        Sleep(1000);
    }
    else
        printf("Fail to get CNT filter width\n");
```

2.31. CNT_GetFreqAcqTime

Users can use this function to get the counter frequency acquisition time of the indicated slot.

Syntax

CNT_GetFreqAcqTime(
  LONG handle,
  WORD i_wSlot,
  DWORD *o_dwFreqAcqTime

);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_dwFreqAcqTime	Output	[out] The variable to hold frequency acquisition time.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
DWORD dwFreqAcqTime = 0;
if(ERR_SUCCESS == CNT_GetFreqAcqTime(lDriverHandle, wSlotID,
&dwFreqAcqTime))
    printf("The CNT frequency acquisition time is %ld\n",
    dwFreqAcqTime);
```

2.32. CNT_GetOverflow

Users can use this function to get overflow flag of the indicated slot and channel.

Syntax

CNT_GetOverflow (
 LONG handle,
 WORD i_wSlot,
 WORD i_wChannel,
 BOOL* o_bFlag
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.

i_wChannel	Input	[in] The channel ID.
o_bFlag	Output	[out] The variable to hold the overflow flag. Returns true if overflow occurred and false otherwise.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_bFlag** contains overflow flag. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1;
BOOL bHoldValue = false;
LONG lResult = CNT_GetOverflow (lDriverHandle, wSlotID, wChannel,
&bHoldValue);
if (ERR_SUCCESS == lResult) {
    if (bHoldValue)
        printf("[Ch %d] The overflow occurred.\n", wChannel);
}
```

2.33. CNT_SetRange

Users can use this function to set the channel range of the indicated slot and channel.

Syntax

```
CNT_SetRange(
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BYTE i_byRange
);
```

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 15.
i_wChannel	Input	[in] The channel ID.

i_byRange	Input	[in] The range code to be set.

Value	Meaning
0x00	Bi-directory
0x01	Up/Down
0x02	Up
0x03	Frequency
0x04	A/B-1X
0x05	A/B-2X
0x06	A/B-4X

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1;
BYTE bHoldValue = 3; /* Frequency */
if (ERR_SUCCESS == CNT_SetRange (lDriverHandle, wSlotID, wChannel,
bHoldValue))
    printf("[Ch %d] Succeed to set range code.\n", wChannel);
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam5081

2.34. CNT_SetStartup

Users can use this function to set the counter startup value of the indicated slot and channel.

Syntax

CNT_SetStartup (
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  DWORD i_dwStartupValue
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
i_dwStartupValue	Input	[in] The variable to hold counter startup value

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1;
DWORD dwStartupValue = 100;
if(ERR_SUCCESS == CNT_SetStartup (lDriverHandle, wSlotID, wChannel,
dwStartupValue))
    printf("[Ch %d] Succeed to set startup value.\n", wChannel);
```

2.35. CNT_SetState

Users can use this function to set the counter status of the indicated slot and channel.

Syntax

CNT_GetState (
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BOOL i_bCounting
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.

i_wChannel	Input	[in] The channel ID.
i_bCounting	Input	[in] The counter status to be set. ***Note:** Set true for "start to count"; false for "stop to count".

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1;
BOOL bHoldValue = true; /* start to count */
if(ERR_SUCCESS == CNT_GetState(lDriverHandle, wSlotID, wChannel,
bHoldValue))
    printf("[Ch %d] Succeed to set counter state.\n", wChannel);
```

2.36. DI_GetValue

Users can use this function to get the DI value of the indicated slot and channel.

Syntax

DI_GetValue (
  LONG handle,
  WORD i_wSlot,
  WORD i_wChannel,
  BOOL* o_bValue
);

Parameters

Name	**Direction**	**Description**
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
o_bValue	Output	[out] The variable to hold the DI value.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_bValue** contains DI value. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelDINum = 8;
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1;
BOOL bHoldValue = false;
LONG lResult = DI_GetValue(lDriverHandle, wSlotID, wChannel,
&bHoldValue);
if (ERR_SUCCESS == lResult) {
    if (bHoldValue)
        printf("Channel %d is true.\n", wChannel);
    else
        printf("Channel %d is false.\n",wChannel);
}
else
    printf("Fail to get DI single channel value \n");
```

2.37. DI_GetValues

Users can use this function to get all DI values of the indicated slot.

Syntax

DIO_GetValues (
 LONG handle,
 WORD i_wSlot,
 DWORD* o_dwValue
);

Parameters

Name	**Direction**	**Description**
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_dwValue	Output	[out] The variable to hold the DI values.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_dwValue** contain DI values from channel 0 to the last channel. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelDINum = 8;
WORD wSlotID = 1; /* Slot ID */
DWORD dwValue = 0x0;
LONG lResult = DI_GetValues(lDriverHandle, wSlotID, &dwValue);
if (ERR_SUCCESS == lResult) {
    for (int iCnt = 0; iCnt < iChannelDINum ; iCnt++) {
        if (dwValue & (0x0001 << iCnt))
            printf("Channel %d is true.\n", iCnt);
        else
            printf("Channel %d is false.\n", iCnt);
    }
}
else
    printf("Fail to get DI channel values \n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam50XXDIO

2.38. DIO_GetUniversalStatus

Users can use this function to determine whether the channel is DI or DO of the indicated slot.

Syntax

DIO_GetUniversalStatus(
  LONG handle,
  WORD i_wSlot,
  WORD* o_wStatus
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_wStatus	Output	[out] The variable to hold the universal status. ***Note:** Returns true for “DI module”; false for “DO module”.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)** and **o_wStatus** contains universal status. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
const int iChannelNum = 16;
WORD wSlotID = 1; /* Slot ID */
WORD dwValue = 0x0;
LONG lResult = DIO_GetUniversalStatus (lDriverHandle, wSlotID,
&dwValue);
if (ERR_SUCCESS == lResult) {
    for (int iCnt = 0; iCnt < iChannelNum ; iCnt++) {
        if (dwValue & (0x01 << iCnt))
            printf("Channel %d is DI.\n", iCnt);
        else
            printf("Channel %d is DO.\n", iCnt);
    }
}
else
    printf("Fail to get universal status\n");
```

2.39. DO_SetValue

Users can use this function to set DO value of the indicated slot and channel.

Syntax

DO_SetValue (
  LONG handle,

WORD i_wSlot,
WORD i_wChannel,
BOOL i_bValue
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_wChannel	Input	[in] The channel ID.
i_bValue	Input	[in] The DO value to be set.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wChannel = 1; /* Channel ID */
BOOL bSetValue = true; /* DO value to be set */

if(ERR_SUCCESS != DO_SetValue(lDriverHandle, wSlotID, wChannel,
bSetValue))
    printf("Fail to set single DO value.\n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam50XXDIO

2.40. DO_SetValues

Users can use this function to set all DO values of the indicated slot.

Syntax

DO_SetValues (
LONG handle,
WORD i_wSlot,
DWORD i_dwValue

);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
i_dwValue	Input	[in] The DO values to be set.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
DWORD dwValue = 0xFF; /*Turn on all channels*/
if(ERR_SUCCESS != DO_SetValues(lDriverHandle, wSlotID, dwValue))
    printf("Fail to set DO multiple channel values.\n");
```

2.41. SYS_GetModuleID

Users can use this function to get the module ID of the indicated slot.

Syntax

SYS_GetModuleID (
  LONG handle,
  WORD i_wSlot,
  WORD * o_wModuleID
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_wModuleID	Output	[out] The variable to hold module ID.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
WORD wModuleID = 0;
if (ERR_SUCCESS == SYS_GetModuleID(lDriverHandle, wSlotID, &wModuleID))
{
    switch (wModuleID) {
        case 0x0004:
            printf("ADAM-5017\n");
        case 0x0005:
            printf("ADAM-5018\n");
        case 0x0008:
        case 0x0009:
            printf("ADAM-5013\n");
        case 0x000C:
            printf("ADAM-5017H\n");
        case 0x000F:
            printf("ADAM-5052\n");
        case 0x0010:
            printf("ADAM-5050\n");
        case 0x0011:
            printf("ADAM-5051\n");
        case 0x0012:
            printf("ADAM-5056\n");
        case 0x0013:
            printf("ADAM-5068\n");
        case 0x0014:
            printf("ADAM-5060\n");
        case 0x0015:
            printf("ADAM-5055\n");
        case 0x0017:
            printf("ADAM-5017UH\n");
        case 0x0018:
            printf("ADAM-5024\n");
        case 0x001E:
            printf("ADAM-5080\n");
        case 0x0069:
            printf("ADAM-5069\n");
        case 0x0002:
            printf("ADAM-5202\n");
        case 0x0048:
            printf("ADAM-5240\n");
        case 0x0001:
            printf("ADAM-5081\n");
        case 0x0053:
            printf("ADAM-5053\n");
        case 0x0091:
            printf("ADAM-5091\n");
        case 0x0038:
            printf("AI modules\n");
            /* call "SYS_GetModuleName" to determine AI module name */
        default:
            printf("unknown module\n");
    }
}
else
    printf("Fail to get module ID\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam_PAC_GetSlotInfo

2.42. SYS_GetModuleName

Users can use this function to determine AI module name (ADAM-5017P / ADAM-5018P) of the indicated slot.

Syntax

SYS_GetModuleID (
  LONG handle,
  WORD i_wSlot,
  CHAR * o_strModuleID
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_wSlot	Input	[in] The slot ID which is ranged from 0 to 6.
o_strModuleID	Output	[out] The variable to hold module name.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wSlotID = 1; /* Slot ID */
char byName[8];
/* call "SYS_GetModuleID" to get module ID
   If module ID is equal to 0x0038, then call SYS_GetModuleName
*/
if (ERR_SUCCESS == SYS_GetModuleName(lDriverHandle, wSlotID, byName)) {
    if(byName[0]=='7' && byName[1]=='P')
        printf("ADAM-5017P\n");
    else if(byName[0]=='8' && byName[1]=='P')
        printf("ADAM-5018P\n");
    else
        printf("AI module\n");
}
else
    printf("Fail to get module name\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam_PAC_GetSlotInfo

2.43. SYS_GetNodeID

Users can use this function to get the DIP switch setting of Node ID.

Syntax

SYS_GetModuleID (

  LONG handle,
  WORD *o_wNodeID
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
o_wNodeID	Output	[out] The variable to hold Node ID.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
WORD wNodeID = 0; /* Node ID */
if (ERR_SUCCESS == SYS_GetNodeID(lDriverHandle, &wNodeID)) {
    printf("Node ID = 0x%04X\n", wNodeID);
else
    printf("Fail to get node ID\n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\
Adam_PAC_SystemLED

2.44. SYS_GetPWRLED

Users can use this function to get the status of Power LED.

Syntax

SYS_GetPWRLED (
  LONG handle,
  BOOL *o_bStatus
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.

o_bStatus	Output	[out] The variable to hold the status of Power LED.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
BOOL bStatus = 0;
if (ERR_SUCCESS == SYS_GetPWRLED(lHandle, &bStatus)) {
    if (bStatus == TRUE)
        printf("Power LED is ON \n");
    else
        printf("Power LED is OFF \n");
}
else
    printf("Fail to get LED status\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam_PAC_SystemLED

2.45. SYS_GetRUNLED

Users can use this function to get the status of Run LED.

Syntax

SYS_GetRUNLED (
  LONG handle,
  BOOL *o_bStatus
);

Parameters

Name	**Direction**	**Description**
handle	Input	[in] The driver handler.
o_bStatus	Output	[out] The variable to hold the status of Run LED.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
BOOL bStatus = 0;
if (ERR_SUCCESS == SYS_GetrunLED(lHandle, &bStatus)) {
    if (bStatus == TRUE)
        printf("Run LED is ON \n");
    else
        printf("Run LED is OFF \n");
}
else
    printf("Fail to get LED status\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam_PAC_SystemLED

2.46. SYS_SetPWRLED

Users can use this function to set the status of Power LED.

Syntax

SYS_GetPWRLED (
  LONG handle,
  BOOL i_bStatus
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_bStatus	Input	[in] The LED status to be set.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
BOOL bStatus = TRUE; /* Turn on LED*/
if (ERR_SUCCESS == SYS_SetPWRLED(lHandle, bStatus)) {
    printf("Power LED is ON \n");
}
else
    printf("Fail to set LED status\n");
```

For more detailed information regarding this function, please see $(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Adam_PAC_SystemLED

2.47. SYS_SetRUNLED

Users can use this function to set the status of Run LED.

Syntax

SYS_GetPWRLED (
  LONG handle,
  BOOL i_bStatus
);

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_bStatus	Input	[in] The LED status to be set.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

Example

```
BOOL bStatus = TRUE; /* Turn on LED*/
if (ERR_SUCCESS == SYS_SetRUNLED(lHandle, bStatus)) {
    printf("Run LED is ON \n");
}
```

```
else
    printf("Fail to set LED status\n");
```

For more detailed information regarding this function, please see
$(Install directory)\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\
Adam_PAC_SystemLED

Chapter 3

3. Internal functions

3.1. SYS_GetCPLDVer

Users can use this function to get CPLD firmware version.

Syntax

SYS_GetCPLDVer(
  LONG handle,
  WORD *i_usCPLDVer
)

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_usCPLDVer	Output	[out] The CPLD firmware version.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

3.2. SYS_GetRetainMemBase

Users can use this function to get the reserved memory start address and length.

Syntax

SYS_GetRetainMemBase (
  LONG handle,
  DWORD *i_dwRetainRAMBase,
  DWORD * i_dwRetainRAMLen
)

Parameters

Name	Direction	Description
handle	Input	[in] The driver handler.
i_dwRetainRAMBase	Output	[out] The memory start address.
i_dwRetainRAMLen	Output	[out] The memory length.

Return Value

If the function succeeds, the return value is **0 (ERR_SUCCESS)**. If the function fails, the return value is **325 (ERR_INTERNAL_FAILED)**. To get extended error information, call **GetLastError** function.

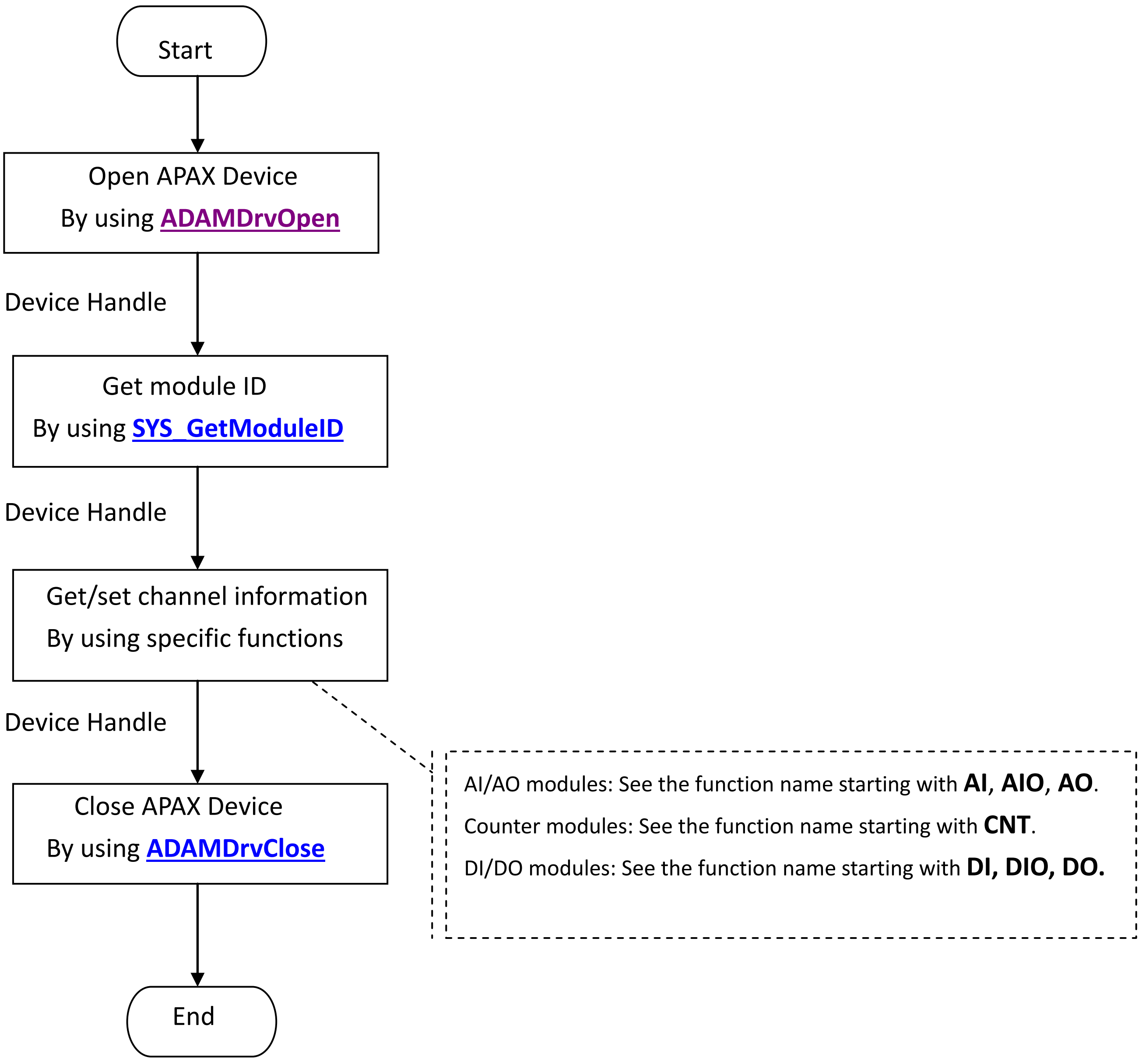
Appendix A

A1. Call flow

Users can directly access drivers with ADSDIO API. Necessary files for developing applications are listed below. Suppose installation paths of all header files in the example are C:\Program Files(x86)\Advantech\AdamApax.NET Class Library\Sample Code\ADAM\Win32\CPlusPlus\ADAM-PAC-Sample\Include

Common Call Flow

The following figure describes the common call flow of the APAX modules.


Start
Open APAX Device
By using ADAMDrvOpen
Device Handle
Get module ID
By using SYS_GetModuleID
Device Handle
Get/set channel information
By using specific functions
Device Handle
Close APAX Device
By using ADAMDrvClose
End
AI/AO modules: See the function name starting with AI, AIO, AO.
Counter modules: See the function name starting with CNT.
DI/DO modules: See the function name starting with DI, DIO, DO.

Appendix B

B1. Error Codes

The information on this page is intended to be used by programmers so that the software they write can better deal with errors.
The following list describes system error codes. They are returned by the **GetLastError** function when many functions fail.

ERROR_SUCCESS
  0 (0x0)
  The operation completed successfully.

ERR_MALLOC_FAILED
  300 (0x12C)
  The system fails to allocate memory.

ERR_MAPADDR_FAILED
  301 (0x12D)
  The system fails to map address.

ERR_HANDLE_INVALID
  302 (0x12E)
  The handle is invalid.

ERR_MODULE_INVALID
  303 (0x12F)
  The module is invalid.

ERR_SLOT_INVALID
  304 (0x130)
  The slot index is invalid.

ERR_CHANNEL_INVALID
  305 (0x131)
  The channel index is invalid.

ERR_FUNC_INVALID
  306 (0x132)
  The function is invalid.

ERR_INTRINIT_FAILED

307 (0x133)

The internal Initialization is failed.

ERR_FREQMEASU_FAILED

308 (0x134)

The system fails to measure frequency.

ERR_PARAM_INVALID

309 (0x135)

The parameter is invalid.

ERR_FIFO_NOTREADY

310 (0x136)

The internal fifo is not ready.

ERR_FIFO_FULL

311 (0x137)

The internal fifo is full.

ERR_FIFO_DATAFAILED

312 (0x138)

The fifo data is failed.

ERR_ACQSTOP_FAILED

313 (0x139)

The system fails to stop data acquisition.

Appendix C

C1. Analog I/O Board Settings

Range Settings for Analog I/O Boards. These ranges are provided for reference. Not all boards support all ranges. Please see hardware manual for valid ranges for a particular board.

	Setting Type	Value (Hex)
ADAM-5013	Pt(385) -100~100 'C	0x20
	Pt(385) 0~100 'C	0x21
	Pt(385) 0~200 'C	0x22
	Pt(385) 0~600 'C	0x23
	Pt(392) -100~100 'C	0x24
	Pt(392) 0~100 'C	0x25
	Pt(392) 0~200 'C	0x26
	Pt(392) 0~600 'C	0x27
	Ni(518) -80~100 'C	0x28
	Ni(518) 0~100 'C	0x29
ADAM-5017	4~20 mA	0x07
	+/- 10V	0x08
	+/- 5 V	0x09
	+/- 1 V	0x0A
	+/-500 mA	0x0B
	+/-150 mA	0x0C
	+/-20 mA	0x0D
ADAM-5017H	+/- 10 V	0x00
	0~10 V	0x01
	+/- 5 V	0x02
	0~5 V	0x03
	+/- 2.5 V	0x04
	0~2.5 V	0x05
	+/- 1 V	0x06
	0~1 V	0x07
	+/- 500 mV	0x08
	0~500 mV	0x09
	4~20 mA	0x0A
	0~20 mA	0x0B
ADAM-5017P	4~20 mA	0x07

	+/- 10 V	0x08
	+/- 5 V	0x09
	+/- 1 V	0x0A
	+/- 500 mV	0x0B
	+/- 150 mV	0x0C
	+/- 20 mA	0x0D
	+/- 15 V	0x15
	0~10 V	0x48
	0~5 V	0x49
	0~1 V	0x4A
	0~500 mV	0x4B
	0~150 mV	0x4C
	0~20 mA	0x4D
	0~15 V	0x55
ADAM-5017UH	4~20 mA	0x07
	+/- 10V	0x08
	0~500 mV	0x2B
	0~20 mA	0x2E
	0~10 V	0x30
ADAM-5018	+/- 15 mV	0x00
	+/- 50 mV	0x01
	+/- 100 mV	0x02
	+/- 500 mV	0x03
	+/- 1 V	0x04
	+/- 2.5 V	0x05
	+/- 20 mA	0x06
	T/C TypeJ 0~760 'C	0x0E
	T/C TypeK 0~1370 'C	0x0F
	T/C TypeT -100~400 'C	0x10
	T/C TypeE 0~1000 'C	0x11
	T/C TypeR 500~1750 'C	0x12
	T/C TypeS 500~1750 'C	0x13
	T/C TypeB 500~1800 'C	0x14
ADAM-5018P	+/- 15 mV	0x00
	+/- 50 mV	0x01
	+/- 100 mV	0x02

	+/- 500 mV	0x03
	+/- 1 V	0x04
	+/- 2.5 V	0x05
	+/- 20 mA	0x06
	4~20 mA	0x07
	T/C TypeJ 0~760 'C	0x0E
	T/C TypeK 0~1370 'C	0x0F
	T/C TypeT -100~400 'C	0x10
	T/C TypeE 0~1000 'C	0x11
	T/C TypeR 500~1750 'C	0x12
	T/C TypeS 500~1750 'C	0x13
	T/C TypeB 500~1800 'C	0x14
ADAM-5024	0~20 mA	0x30
	4~20 mA	0x31
	0~10 V	0x32