User Guide for EAPI Software Toolkit

Author: Hans Wu ,Nick Fang

Table of Content

Version History

Version	Date	Remark	Author
1.0	2016/8	Revision 1.0	Hans

Introduction

Purpose of this document

The document is introduction the Portwell EAPI toolkit (PEAPI), and how to development the software to control the Portwell hardware and uses it. EAPI provide the API function and it can easy use development applications. The document explains the API function how to use and provide the sample code.

Package contents

In Windows version, the software package contain below:
Driver: portwell.sys and portwellx64.sys
Library: PEAPI_1.dll, PEAPI_1.lib, PEAPI_x64_1.dll and PEAPI_x64_1.lib
Header file: Eapi_type.h, pet_Eapi.h, Type_Define.h, PET_Type.h
Sample code: Eapi_TESTAP.cpp

Operating Systems

The PEAPI work on module series boards, what boards made by Portwell. PEAPI is run the Windows 7, Windows 8 and Ubuntu well.

PEAPI Function Guide

The PEAPI provide the dynamic link library on our platform. All the function have to initialized “EApiLibInitialize()” before used. If the function didn’t called after “EApiLibInitialize()”, all the function would not work.

Return values

EAPI_STATUS_NOT_INITIALIZED	0xFFFFFFFF
EAPI_STATUS_INITIALIZED	0xFFFFFFFE
EAPI_STATUS_ALLOC_ERROR	0xFFFFFFFD
EAPI_STATUS_DRIVER_TIMEOUT	0xFFFFFFFC
EAPI_STATUS_INVALID_PARAMETER	0xFFFFFEFF
EAPI_STATUS_INVALID_BLOCK_ALIGNMENT	0xFFFFFEFE
EAPI_STATUS_INVALID_BLOCK_LENGTH	0xFFFFFEFD
EAPI_STATUS_INVALID_DIRECTION	0xFFFFFEFC
EAPI_STATUS_INVALID_BITMASK	0xFFFFFEFB
EAPI_STATUS_RUNNING	0xFFFFFEFA
EAPI_STATUS_UNSUPPORTED	0xFFFFFCFF
EAPI_STATUS_NOT_FOUND	0xFFFFFBFF
EAPI_STATUS_TIMEOUT	0xFFFFFBFE
EAPI_STATUS_BUSY_COLLISION	0xFFFFFBFD
EAPI_STATUS_READ_ERROR	0xFFFFFAFf
EAPI_STATUS_WRITE_ERROR	0xFFFFFAFE
EAPI_STATUS_MORE_DATA	0xFFFFF9FF
EAPI_STATUS_ERROR	0xFFFFF0FF
EAPI_STATUS_SUCCESS	0

Initialization

EApiLibInitialize

Prototype

```
PEApiStatus_t
PEAPI
EApiLibInitialize (void);
```

Description

Initial the application programming library of PEAPI. Programmer has to call function before calling other functions.

Parameters

None.

EApiLibUnInitialize

Prototype

```
PEApiStatus_t
PEAPI
EApiLibUnInitialize (void);
```

Description

Before an application is end, it must un-initialized the PEAPI library.

Parameters

None.

Board information

PEPApiBoardGetStringA

Prototype

```
PEApiStatus_t
PEAPI
EApiBoardGetStringA (uint32_t Id, char *pBuffer, uint32_t *pBufLen);
```

Description

Get the hardware platform information in text format.

Parameters

in/out	parameter name	description
in	Id	Selects the gets string function Id
out	pBuffer	Receive the string buffer
in/out	pBufLen	[in] the buffer size for receive string; [out] the string size request of buffer.

Select the Id string for sub function.

Id	Description
EAPI_ID_BOARD_MANUFACTURER_STR	Board Manufacturer Name
EAPI_ID_BOARD_NAME_STR	Board Name
EAPI_ID_BOARD_SERIAL_STR	Serial Number
EAPI_ID_BOARD_BIOS_REVISION_STR	Board BIOS Revision
EAPI_ID_BOARD_PLATFORM_TYPE_STR	Platform ID

EApiBoardGetValue

Prototype

PEApiStatus_t PEAPI EApiBoardGetValue (uint32_t Id, uint32_t *pValue);

Description

Get the hardware platform information in value format.

Parameters

In/out	Parameter name	Description
In	Id	Selects the gets value
Out	pValue	Receive the value

Select the Id value for sub function

Id	Description
EAPI_ID_GET_EAPI_SPEC_VERSION	EAPI Specification Version used to implement API
EAPI_ID_BOARD_BOOT_COUNTER_VAL	Boot Counter
EAPI_ID_BOARD_RUNNING_TIME_METER_VAL	Running Time Meter (Unit Minutes)
EAPI_ID_BOARD_PNPID_VAL	Board Vendor PNPID
EAPI_ID_BOARD_PLATFORM_REV_VAL	Platform Specification Version of the Board
EAPI_ID_BOARD_DRIVER_VERSION_VAL	Vendor Specific Driver Version
EAPI_ID_BOARD_LIB_VERSION_VAL	Vendor Specific Library Version
EAPI_ID_HWMON_CPU_TEMP	CPU Temperature (Unit 0.1 Kelvins)
EAPI_ID_HWMON_SHIPSET_TEMP	Chipset Temperature (Unit 0.1 Kelvins)
EAPI_ID_HWMON_SYSTEM_TEMP	System Temperature (Unit 0.1 Kelvins)
EAPI_ID_HWMON_VOLTAGE_VCORE	CPU Core Voltage (Unit millivolts)
EAPI_ID_HWMON_VOLTAGE_2V5	2.5V Voltage (Unit millivolts)
EAPI_ID_HWMON_VOLTAGE_3V3	3.3V Voltage (Unit millivolts)
EAPI_ID_HWMON_VOLTAGE_VBAT	Battery Voltage (Unit millivolts)

EAPI_ID_HWMON_VOLTAGE_5V	5V Voltage (Unit millivolts)
EAPI_ID_HWMON_VOLTAGE_5VSB	5V Standby Voltage (Unit millivolts)
EAPI_ID_HWMON_VOLTAGE_12V	12V Voltage (Unit millivolts)
EAPI_ID_HWMON_FAN_CPU	CPU Fan (Unit RPM)
EAPI_ID_HWMON_FAN_SYSTEM	System Fan (Unit RPM)

I2C Bus Function

EApiI2CReadByte

Prototype

```
PEApiStatus_t
PEAPI
EApiI2CReadByte (uint8_t Addr, uint32_t cmd, uint8_t *pBuffer);
```

Description

Read from a register in the selected I2C device. Read from the I2C device at I2C address Addr, using the device command cmd, and read data to the buffer pBuffer.

Parameters

in/out	parameter name	description
in	Addr	Encode 7bit I2C device address
in	cmd	Command/Offset
in/out	pBuffer	[in] the buffer for read data; [out] the buffer for write data.

EApiI2CWriteByte

Prototype

```
PEApiStatus_t
PEAPI
EApiI2CWriteByte (uint8_t Addr, uint32_t cmd, uint8_t *pBuffer);
```

Description

Write to a register in the selected I2C device. Write to the I2C device at I2C address Addr, using the device command cmd, and write data from the buffer pBuffer.

Parameters

in/out	parameter name	description
in	Addr	Encode 7bit I2C device address
in	cmd	Command/Offset
in/out	pBuffer	[in] the buffer for read data; [out] the buffer for write data.

EApiSMBusReadByte

Prototype

```
PEApiStatus_t
PEAPI
EApiSMBusReadByte (uint8_t Addr, uint8_t cmd, uint8_t *pBuffer);
```

Description

Read from a register in the selected SMBus device. Read from the SMBus device at SMBus address Addr, using the device command cmd, and read data to the buffer pBuffer.

Parameters

in/out	parameter name	description
in	Addr	Encode 7bit SMBus device address
in	cmd	Command/Offset
in/out	pBuffer	[in] the buffer for read data; [out] the buffer for write data.

EApiSMBusWriteByte

Prototype

```
PEApiStatus_t
PEAPI
EApiSMBusWriteByte (uint8_t Addr, uint8_t cmd, uint8_t *pBuffer);
```

Description

Write to a register in the selected SMBus device. Write to the SMBus device at SMBus address Addr, using the device command cmd, and write data from the buffer pBuffer.

Parameters

in/out	parameter name	description
in	Addr	Encode 7bit SMBus device address
in	cmd	Command/Offset
in/out	pBuffer	[in] the buffer for read data; [out] the buffer for write data.

WATCHDOG

EApiWDogGetCap

Prototype

```
PEApiStatus_t
PEAPI
EApiWDogGetCap (uint32_t *pMaxDelay, uint32_t *pMaxEventTimeout, uint32_t
*pMaxResetTimeout);
```

Description

Get the watchdog timer capabilities.

Parameters

in/out	parameter name	description
out	pMaxDelay	Receives maximux supported initial delay time of WDT in millseconds.
out	pMaxEventTimeout	Receives maximux supported event delay time of WDT in millseconds.
out	pMaxResetTimeout	Receives maximux supported reset delay time of WDT in millseconds.

EApiWDogStart

Prototype

```
PEApiStatus_t
PEAPI
EApiWDogStart (uint32_t Delay, uint32_t EventTimeout, uint32_t ResetTimeout);
```

Description

Set the watchdog timer. When the EApiWDogStart called by programmer, if need to reset the parameters by EApiWDogStart, programmer would called the function EApiWDogStop before EApiWDogStart.

Parameters

in/out	parameter name	description
in	Delay	Set the delay time for watchdog timer.
in	EventTimeout	Set the event time for watchdog timer.
in	ResetTimeout	Set the reset time for watchdog timer.

EApiWDogTrigger

Prototype

```
PEApiStatus_t
PEAPI
EApiWDogTrigger (void);
```

Description

Trigger the watchdog timer.

Parameters

None

EApiWDogStop

Prototype

```
PEApiStatus_t
PEAPI
EApiWDogStop (void);
```

Description

Stop the operation of the watchdog timer. EApiWDogStop can stop when EApiWDogStart and EApiWDogTrigger call.

Parameters

None

GPIO

EApiGPIOGetDirectionCaps

Prototype

PEApiStatus_t PEAPI EApiGPIOGetDirectionCaps (uint32_t Id, uint32_t *pInputs, uint32_t *pOutputs);

Description

Get the GPIO capabilities from the select GPIO interface. The pInputs and pOutputs get the value “1” means the port can set input or output by EApiGPIOSetDirection.

Parameters

in/out	parameter name	description
in	Id	Select GPIO pin.
out	pInputs	Receive the bit mask of the supported inputs.
out	pOutputs	Receive the bit mask of the supported output.

The Id parameter can select

define string	description
EAPI_ID_GPIO_GPIO00	GPIO pin 0 bit mapped to bit 0
EAPI_ID_GPIO_GPIO01	GPIO pin 1 bit mapped to bit 1
EAPI_ID_GPIO_GPIO02	GPIO pin 2 bit mapped to bit 2
EAPI_ID_GPIO_GPIO03	GPIO pin 3 bit mapped to bit 3
EAPI_ID_GPIO_GPIO04	GPIO pin 4 bit mapped to bit 4
EAPI_ID_GPIO_GPIO05	GPIO pin 5 bit mapped to bit 5
EAPI_ID_GPIO_GPIO06	GPIO pin 6 bit mapped to bit 6
EAPI_ID_GPIO_GPIO07	GPIO pin 7 bit mapped to bit 7
EAPI_ID_GPIO_BANK00	GPIO pin 0-7 bit mapped to bit 0-7

EApiGPIOGetDirection

Prototype

```
PEApiStatus_t
PEAPI
EApiGPIOGetDirection (uint32_t Id, uint32_t Bitmask, uint32_t *pDirection);
```

Description

Read the current configuration of the select GPIO pin.

Parameters

in/out	parameter name	description
in	Id	Select GPIO pin.
in	Bitmask	Bit mask of affected bits.
out	pDirection	Get the current direction.

Bitmask define value

define string	description
EAPI_GPIO_BITMASK_NOSELECT	Do not get the selected pin for this operation.
EAPI_GPIO_BITMASK_SELECT	Get the selected pin for this operation.

EApiGPIOSetDirection

Prototype

PEApiStatus_t PEAPI EApiGPIOSetDirection (uint32_t Id, uint32_t Bitmask, uint32_t Direction);

Description

Set the current configuration of the select GPIO pin.

Parameters

in/out	parameter name	description
in	Id	Select GPIO pin.
in	Bitmask	Bit mask of affected bits.
out	Direction	Set the direction.

Bitmask define value

define string	description
EAPI_GPIO_BITMASK_NOSELECT	Do not get the selected pin for this operation.
EAPI_GPIO_BITMASK_SELECT	Get the selected pin for this operation.

define string	description
EAPI_GPIO_INPUT	Select GPIO pin is an input.
EAPI_GPIO_OUTPUT	Select GPIO pin is an output.

EApiGPIOGetLevel

Prototype

```
PEApiStatus_t
PEAPI
EApiGPIOGetLevel (uint32_t Id, uint32_t Bitmask, uint32_t *pLevel);
```

Description

Get the level from GPIO pin.

Parameters

in/out	parameter name	description
in	Id	Select GPIO pin.
in	Bitmask	Bit mask of affected bits.
out	pLevel	Receive the level value from selected.

Bitmask define value

define string	description
EAPI_GPIO_BITMASK_NOSELECT	Do not get the selected pin for this operation.
EAPI_GPIO_BITMASK_SELECT	Get the selected pin for this operation.

EApiGPIOSetLevel

Prototype

PEApiStatus_t PEAPI EApiGPIOSetLevel (uint32_t Id, uint32_t Bitmask, uint32_t Level);

Description

Set the level from GPIO pin.

Parameters

in/out	parameter name	description
in	Id	Select GPIO pin.
in	Bitmask	Bit mask of affected bits.
in	Level	Set the level value to select pin.

Bitmask define value

define string	description
EAPI_GPIO_BITMASK_NOSELECT	Do not get the selected pin for this operation.
EAPI_GPIO_BITMASK_SELECT	Get the selected pin for this operation.

define string	description
EAPI_GPIO_LOW	Set selected GPIO pin to low.
EAPI_GPIO_HIGH	Set selected GPIO pin to high.