TIMER Driver User Guide V1.00.01

1. Function

DrvTIMER_GetTimerCLK

Prototype

int32_t DrvTIMER_GetTimerCLK(TIMER_CHANNEL ch);

Description

This function is used to get the timer clock

Parameter

ch [in]

TIMER channel TMR0/ TMR1

Include

Driver/DrvTimer.h

Return Value

clk: Current timer clock.

E_DRVTIMER_CHANNEL: Invalid Timer channel.

DrvTIMER_GetStatus

Prototype

int32_t DrvTIMER_GetStatus(TIMER_CHANNEL ch);

Description

This function is used to return read TIMER TISR register to get timer interrupt status.

Parameter

channel [in]

TIMER channel TMR0/ TMR1

Include

Driver/DrvTimer.h

Return Value

The data of register TISR.

DrvTIMER_SetTimerEvent

Prototype

int32_t DrvTIMER_SetTimerEvent(

TIMER_CHANNEL channel,

uint32_t uTimeTick,

TIMER_CALLBACK pvFun ,
uint32_t parameter
);

Description

This function is used to install a event to timer0, timer1.

Parameter

channel [in]

TMR0/ TMR1

uTimeTick [in]

The tick value which want to execute event.

pvFun [in]

The event function pointer.

parameter [in]

An parameter, was defined by user, which send to callback function.

Include

Driver/DrvTimer.h

Return Value

The event number which contains this event

DrvTIMER_ClearTimerEvent

Prototype

void DrvTIMER_ClearTimerEvent(
TIMER_CHANNEL channel,
uint32_t uTimeEventNo
);

Description

This function is used to remove an installed event.

Parameter

channel [in]

TIMER channel TMR0/ TMR1

uTimeEventNo [in]

EVENT No. it could be 0 ~ TIMER_EVENT_COUNT-1.

Include

Driver/DrvTimer.h

Return Value

None

DrvTIMER_EnableInt

Prototype

int32_t DrvTIMER_EnableInt (E_TIMER_CHANNEL ch)

Description

This function is used to enable the specified timer interrupt.

Parameter

ch [in]

TIMER channel TMR0/ TMR1

Include

Driver/DrvTIMER.h

Return Value

E_SUCCESS: Operation successful

E_DRVTIMER_CHANNEL: Invalid timer channel

DrvTIMER_ResetTicks

Prototype

Int32_t DrvTIMER_ResetTicks(TIMER_CHANNEL channel);

Description

This function is used to reset TIMER Tick.

Parameter

channel [in]

TIMER channel TMR0/ TMR1.

Include

Driver/DrvTimer.h

Return Value

E_SUCCESS: Success

E_DRVTIMER_CHANNEL: Unsupported timer channel

DrvTIMER_Init

Prototype

void DrvTIMER_Init(void);

Description

This function is used to initial TIMER when system boot up.

Parameter

None

Include

Driver/DrvTimer.h

Return Value

None

DrvTIMER_Open

Prototype

int32_t DrvTIMER_Open(
TIMER_CHANNEL channel,
uint32_t uTicksPerSecond,
TIMER_OPMODE mode
);

Description

This function is used to set and start TIMER.

Parameter

channel [in]

TIMER channel TMR0/ TMR1

uTickPerSecond [in]

Tick per second.

Mode [in]

Operation Mode One-Shot / Periodic / Toggle. It could be ONESHOT_MODE, PERIODIC_MODE ,TOGGLE_MODE or UNINTERREUPT_MODE.

Include

Driver/DrvTimer.h

Return Value

E_SUCCESS: Success.
E_DRVTIMER_CMD: Command error.
E_DRVTIMER_EIO: Timer is not initialized by DrvTIMER_Init().

DrvTIMER_GetTicks

Prototype

uin32_t DrvTIMER_GetTicks(TIMER_CHANNEL channel);

Description

This function is used to return Timer ticks.

Parameter

channel [in]

TIMER channel TMR0/ TMR1.

Include

Driver/DrvTimer.h

Return Value

Return the current ticks of TIMER0/TIMER1.

DrvTIMER_Delay

Prototype

void DrvTIMER_Delay (uint32_t uTicks);

Description

This function is used to set a delay time if necessary. The TIMER0 is used in this delay function thus it needs to be opened and initialized first.

Parameter

uTicks [in]

The delay time, and it is depend on Timer CLK.

Include

Driver/DrvTimer.h

Return Value

None

DrvTIMER_SetEXTClockFreq

Prototype

void DrvTIMER_SetEXTClockFreq (uint32_t u32ClockValue)

Description

Set the external clock frequency, it's used for timer clock source. User can change the timer clock source from the external clock source by calling DrvSYS_SetIPClockSource (…).

Parameter

u32ClockFreq [in]

Set the clock frequency (Hz) for external clock source

Include

Driver/DrvTIMER.h

Return Value

None

DrvTIMER_Ioctl

Prototype

```
int32_t DrvTIMER_Ioctl(
TIMER_CHANNEL channel,
TIMER_CMD uCmd,
UINT32 uArg1
);
```

Description

To process the general control of timer. The following table listed the command, parameters and relative descriptions.

Command	Argument	Description
TIMER_IOC_START_COUNT	Not used	Start timer counter
TIMER_IOC_STOP_COUNT	Not used	Stop timer counter
TIMER_IOC_ENABLE_INT	Not used	Enable the timer interrupt
TIMER_IOC_DISABLE_INT	Not used	Disable the timer interrupt
TIMER_IOC_RESET_TIMER	Not used	Reset timer counter
TIMER_IOC_SET_PRESCALE	uArg1	uArg1 is the pre-scale value for timer counter. The value could be 0 ~ 255 and resulting the counter clock to be divided by 1 ~ 256.
TIMER_IOC_SET_INITIAL_COUNT	uArg1	This command is used to specify the initial value of timer counter. Due to the timer counter is 16-bit, the uArg1 could be 0 ~ 65535 and the timer counter will down count to 0 from the initial count value when timer start.

Parameter

channel [in]

TIMER channel TMR0/ TMR1.

uCmd [in]

The I/O control commands, e.x. TIMER_IOC_START_COUNT.

uArg1 [in]

The first parameter for specified command.

pvFun [in]

The event function pointer.

Include

Driver/DrvTimer.h

Return Value

E_SUCCESS: Success.

E_DRVTIMER_CHANNEL: Invalid timer channel

E_DRVTIMER_CMD: Invalid command.

DrvTIMER_Close

Prototype

int32_t DrvTIMER_Close(TIMER_CHANNEL channel);

Description

The function is used to disable timer.

Parameter

channel [in]

TIMER channel TMR0/ TMR1.

Include

Driver/DrvTimer.h

Return Value

E_SUCCESS: Success.

E_DRVTIMER_CMD: Invalid command

DrvWDT_InstallISR

Prototype

void DrvWDT_InstallISR (TIMER_CALLBACK pvNewISR)

Description

The function is used to install WDT interrupt service routine.

Parameter

pvNewISR [in]

The function pointer of the interrupt service routine

Include

Driver/DrvTIMER.h

Return Value

None

DrvWDT_SetInterval

Prototype

void WDT_SetInterval (WDT_INTERVAL nWdtInterval)

Description

The function is used to set the “Interval Level” of WDT Time-out

Parameter

nWdtInterval [in]

Watch Dog time-out Interval.

Include

Driver/DrvTIMER.h

Return Value

E_SUCCESS: Success

DrvWDT_Open

Prototype

int32_t DrvWDT_Open(int32_t hander ,WDT_INTERVAL level);

Description

The function is used to set WDT Interval and Star WDT Timer to count.

Parameter

hander [in]

Reserved.

level [in]

WDT time-out level. It could be LEVEL0 ~ 7 .

Include

Driver/DrvTimer.h

Return Value

E_SUCESS: Success

DrvWDT_Ioctl

Prototype

int32_T DrvWDT_Ioctl(int32_t hander ,WDT_CMD uCmd , uint32_t uArg1);

Description

The function is used to I/O Control API.

Parameter

hander [in]

Reserved.

uCmd [in]

WDT IO control command.

uArg1 [in]

First argument of the command.

Include

Driver/DrvTimer.h

Return Value

E_SUCCESS: Success

E_DRVTIMER_CMD: Invalid I/O command

DrvWDT_Close

Prototype

void DrvWDT_Close(void);

Description

The function is used to Stop WatchDog Timer and Disable WDT Interrupt.

Parameter

None

Include

Driver/DrvTimer.h

Return Value

None

DrvWDT_ResetCount

Prototype

void DrvWDT_ResetCount(void);

Description

This function is used to reset WDT Tick to avoid time-out to restart system.

Parameter

None

Include

Driver/DrvTimer.h

Return Value

None

DrvTIMER_GetVersion

Prototype

uint32_t DrvTimer_GetVersion (void);

Description

Return the current version number of driver.

Include

Driver/DrvTimer.h

Return Value

Version number :

31:24	23:16	15:8	7:0
00000000	MAJOR_NUM	MINOR_NUM	BUILD_NUM

2. Revision History

Version	Date	Description
1.00.01	Mar. 2011	Preliminary TIMER Driver User Guide of ISD9160