

# **Intel<sup>®</sup> 6300ESB I/O Controller Watchdog Timer Driver Specification**

**Ref. No. 0.1**

Copyright © Intel Corporation 2004



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The Intel Watchdog Timer Application Programming Interface may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

MPEG is an international standard for video compression/decompression promoted by ISO. Implementations of MPEG CODECs, or MPEG enabled platforms may require licenses from various entities, including Intel Corporation.

This Datasheet as well as the software described in it is furnished under license and may only be used or copied in accordance with the terms of the license. The information in this manual is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Intel Corporation. Intel Corporation assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document.

Except as permitted by such license, no part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the express written consent of Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature may be obtained by calling

1-800-548-4725 or by visiting Intel's website at <http://www.intel.com>.

AnyPoint, AppChoice, BoardWatch, BunnyPeople, CablePort, Celeron, Chips, CT Media, Dialogic, DM3, EtherExpress, ETOX, FlashFile, i386, i486, i960, iCOMP, InstantIP, Intel, Intel Centrino, Intel logo, Intel386, Intel486, Intel740, IntelDX2, IntelDX4, IntelSX2, Intel Create & Share, Intel GigaBlade, Intel InBusiness, Intel Inside, Intel Inside logo, Intel NetBurst, Intel NetMerge, Intel NetStructure, Intel Play, Intel Play logo, Intel SingleDriver, Intel SpeedStep, Intel StrataFlash, Intel TeamStation, Intel Xeon, Intel XScale, IPLink, Itanium, MCS, MMX, MMX logo, Optimizer logo, OverDrive, Paragon, PC Dads, PC Parents, PDCharm, Pentium, Pentium II Xeon, Pentium III Xeon, Performance at Your Command, RemoteExpress, SmartDie, Solutions960, Sound Mark, StorageExpress, The Computer Inside., The Journey Inside, TokenExpress, VoiceBrick, VTune, and Xircom are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

\*Other names and brands may be claimed as the property of others.

Copyright © 2004, Intel Corporation



## Introduction

This document covers the Intel® 6300ESB I/O Controller Watchdog Timer Driver. A basic familiarity with Windows device driver concepts is assumed. The Intel Watchdog Timer (WDT) driver is a kernel mode driver designed run on Microsoft Windows NT\*, Windows 2000\* and Windows XP\* platforms and provide OS base control of the CICH2 Watchdog Timer device. In the event of a lockup condition, the watchdog device can signal the platform using an interrupt mechanism or, if necessary, reboot the platform.

## Scope

This specification describes the WDT Driver Architecture and IOCTL device codes that are used to control the driver from a user mode program. This document will describe how the watchdog device is integrated into the following components and operating systems:

- Intel I/O Control Hub (CICH2) based chipsets.
- Microsoft Windows 2000 and Microsoft Windows XP families.

## Supporting Documents

- *Windows XP 2600 DDK*
- *Windows Platform SDK*
- *Intel wdt api.DOC*

## Interface of Driver

The device driver has a named device object that can be accessed through the symbolic link "\.\\SAWD1". Applications can access the driver by first obtaining a device handle, by calling *CreateFile()* with the file name "\.\\SAWD1". This will cause the symbolic link to be accessed and cause an IRP to be passed to the driver. On success, the handle can be used with *DeviceIoControl()* to direct IOCTL calls to the driver.

The driver supports the following IOCTLs, which are accessed in the IRP\_MJ\_DEVICE\_CONTROL dispatch handler.

IOCTL_ENABLE_WDT	Start timer counting down
IOCTL_DISABLE_WDT	Stop timer from counting down
IOCTL_LOAD_COUNTER	Load the counters
IOCTL_PING_THE_WDT	Ping the Timer to prevent timeout
IOCTL_SET_PRESEALAR	Select presecalar to use
IOCTL_READ_DOWN_COUNTER	Read current count in down counter
IOCTL_SET_EXTERNAL_OUT	Enable/disable the output pin if Stage 2 times out
IOCTL_LOCK_DEVICE	Lock WDT to prevent changes
IOCTL_ROUTE_INTERRUPT	Select how the Stage 1 interrupt will be handled.
IOCTL_WRITE_CONFIG	Low-level configuration control
IOCTL_USER_HANDLE	Obtain kernel pointer to user-mode event handle.
IOCTL_INTR_CONNECT	Manually connect driver to interrupt logic



IOCTL_GET_STATUS	Return misc. device status bits
IOCTL_SET_MODE	Select either Watchdog or Free running timer
IOCTL_GET_TIMEOUT_STATUS	Check if Timeout occurred

## IWDTLIB.DLL Interface

The IWDTLIB.DLL is a dynamic link library packaged with the Watchdog driver that provides a high-level interface to control the Watchdog driver from a Windows ring 3 application. The import library IWDTLIB.LIB is provided to link the DLL to a user application. In addition, the header file wdt.h is also provided.

The following APIs can be imported into your application. Refer to Intelwdt api. for full details on the use of the IWDTLIB library and coding examples.

WDT_IMPORT HANDLE	WdtInitLibrary(PWDT);
WDT_IMPORT HANDLE	WdtGetDeviceHandle();
WDT_IMPORT HANDLE	WdtGetStatus( HANDLE <i>wdHandle</i> , SAWD_CTRL *);
WDT_IMPORT ULONG	WdtGetDriverVersion(HANDLE);
WDT_IMPORT ULONG	WdtGetLibraryVersion(VOID);
WDT_IMPORT BOOL	WdtGetCapabilities(HANDLE );
WDT_IMPORT BOOL	WdtSetPreloadValues(HANDLE,WDT);
WDT_IMPORT BOOL	WdtPing(HANDLE,ULONG);
WDT_IMPORT BOOL	WdtEnable(HANDLE, bool);
WDT_IMPORT BOOL	WdtLockDevice(HANDLE);
WDT_IMPORT BOOL	WdtStageOneNotify(HANDLE, SIFUNCTPTR );
WDT_IMPORT BOOL	WdtCancelNotify();
WDT_IMPORT int	WdtCheckTimeOutStatus(HANDLE, BOOL);
WDT_IMPORT BOOL	WdtSetPrescaler(HANDLE, ULONG);
WDT_IMPORT BOOL	WdtRouteInterrupt(HANDLE, ULONG);
WDT_IMPORT BOOL	WdtSetMode(HANDLE xhndFile, ULONG);
WDT_IMPORT BOOL	WdtSetOutputEnable(HANDLE, ULONG);

The following structure is used to communicate data from user mode applications to the kernel mode driver.

```
typedef struct _SAWD_CTRL {
```

HANDLE	UserEvent; // Handle to a Ring 3 user event
unsigned long	Version; // version of interface used
unsigned long	Flags; // flags defined below
unsigned long	Preload1; // Preload register # 1
unsigned long	Preload2; // Preload Register # 2
unsigned long	DownCount; // Holds current down count
unsigned long	ConfigReg; // 16-bit Configuration register



```
unsigned short    DeviceStatus; // store device status bits
unsigned short    ReloadReg; // 16-bit reload register
unsigned short    LockReg; // 8-bit Lock register
unsigned long     InterruptCount; // Count of times the stage 1 interrupt has occurred

unsigned short    IRQ;
short             BusType;
unsigned short    BusNumber;

unsigned short    IrqOffset;

} SAWD_CTRL,      *PSAWD_CTRL;
```

## Installing the Watchdog Driver

Install the watchdog timer driver using the method described in the release notes documentation file.