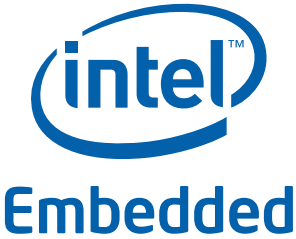

Intel® Platform Controller Hub EG20T

IEEE1588 Hardware Assist Driver for Windows* Programmer's Guide

March 2011

Order Number: 324259-002US

Contents

Tables

Revision History

Date	Revision	Description
March 2011	002	Updated Section 2.0, “Operating System (OS) Support” on page 7
September 2010	001	Initial release

1.0 Introduction

This document describes the IEEE1588 hardware assist driver interfaces exposed to the user-mode applications and how to use those interfaces to drive the IEEE1588 hardware to achieve time synchronization on Ethernet and CAN.

In a distributed control system containing multiple clocks, individual clocks tend to drift apart. A correction mechanism is necessary to synchronize the individual clocks to maintain global time, which is accurate to some requisite clock resolution. The IEEE 1588-2008 standard defines a precision clock synchronization protocol for networked measurement and control systems, including several messages used to exchange timing information. The hardware assist logic required to achieve precision clock synchronization using the IEEE1588-2008 standard depends on the implementation.

2.0 Operating System (OS) Support

The IEEE1588 driver is supported by the following operating system:

No	OS	Notes
1	Microsoft Windows XP*	Service Pack 3
2	Windows Embedded Standard*	2009
3	Windows Embedded POSReady*	2009
4	Microsoft Windows 7*	
5	Windows Embedded Standard7	

3.0 Dependencies

The Intel® Platform Controller Hub EG20T IEEE 1588 Hardware Assist driver is dependent upon either the Intel® Platform Controller Hub EG20T Gigabit Ethernet or Intel® Platform Controller Hub EG20T Controller Area Network implementation. Time synchronization is performed by either inspecting packets on the Gigabit Ethernet port or by inspecting packets on the CAN port. Thus, IEEE 1588 Hardware Assist implementation is also dependent upon either the Gigabit Ethernet driver or the Controller Area Network driver. Additionally, if CAN is used for the implementation of time synchronization, there is a dependency on a CAN client application to control the CAN driver and hardware block.

4.0 IEEE1588 Driver API Details

The IEEE1588 driver exposes the interfaces through Input/Output Controls (IOCTLs), which can be called from the user-mode applications. The following sections provide information about the IOCTLs of the driver and how to use them to successfully drive the IEEE1588 hardware.

4.1 Features

- Conforms to the IEEE1588-2008 standard
  - — Selecting IEEE1588-2002 message or IEEE1588-2008 message can be done by hardware
- Support IEEE1588 over Ethernet
- Support IEEE1588 over CAN
- Support auxiliary snapshots

4.2 Driver Configuration

The following driver entries must be modified in the inf file “iohieee1588.inf” or with the corresponding IEEE1588 driver registry to enable support for time stamping over TCP/IP and CAN networks.

Table 1. Driver Configuration

No.	Entry	Description
1	EnableEthernet	DWORD value to Enable/Disable time stamping of TCP/IP packets 1 = Enable Time stamping of TCP/IP packets 0 = Disable Time stamping of TCP/IP packets
2	EnableCAN	DWORD value to Enable/Disable time stamping of CAN packets 1 = Enable Time stamping of CAN packets 0 = Disable time stamping of CAN packets
3	StationAddress	String value indicating the MAC address of machine

The above configurations can be set after the IEEE1588 driver installation by modifying the value in the below registry entries:

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\PCI\VEN_8086&DEV_8819&SUBSYS_00000000&REV_00\5&335da31&0&6400B8\Device Parameters\EnableEthernet
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\PCI\VEN_8086&DEV_8819&SUBSYS_00000000&REV_00\5&335da31&0&6400B8\Device Parameters\EnableCAN
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\PCI\VEN_8086&DEV_8819&SUBSYS_00000000&REV_00\5&335da31&0&6400B8\Device Parameters\StationAddress

For example:

To enable support for time stamping over Ethernet and CAN networks:

```
Windows Registry Editor Version 5.1
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\PCI\VEN_8086&DEV_8819&SUBSYS_00000000&REV_00\5&335da31&0&6400B8\Device Parameters]
"EnableCAN" = dword:00000001
"EnableEthernet"=dword:00000001
```

"StationAddress"="XX:XX:XX:XX:XX:XX"

Note: Put the Physical Address of the Gigabit Ethernet Controller in "XX:XX:XX:XX:XX:XX".

Go to start-> run-> regedit, traverse to the path stated above and edit the entries' values accordingly.

4.3 Interface Details

Table 2 on page 10 lists the IOCTLs supported by the driver.

Table 2. Supported IOCTLs (Sheet 1 of 2)

No.	Interface	Description
1	IOCTL_1588_PORT_CONFIG_SET	Enables the time stamping on a particular PTP port.
2	IOCTL_1588_PORT_CONFIG_GET	Identifies the time stamping capability of a PTP port by obtaining its mode of operation.
3	IOCTL_1588_RX_POLL	Poll for the availability of a time stamp on the received Sync (Slave) or Delay_Req (Master) messages.
4	IOCTL_1588_TX_POLL	Poll for the availability of a time stamp on the transmitted Sync (Master) or Delay_Req (Slave) messages.
5	IOCTL_1588_CAN_POLL	Poll for the availability of a time stamp on a CAN port.
6	IOCTL_1588_SYS_TIME_GET	Get the System Time for Hardware Assist block.
7	IOCTL_1588_SYS_TIME_SET	Set the System Time to a given value.
8	IOCTL_1588_TICK_RATE_SET	Set the Tick Rate (Frequency Scaling Value).
9	IOCTL_1588_TICK_RATE_GET	Get the Tick Rate (Frequency Scaling Value).
10	IOCTL_1588_TARG_TIME_INTRPT_ENABLE	Enable target/aux time interrupt.
11	IOCTL_1588_TARG_TIME_INTRPT_DISABLE	Disable target/aux time interrupt.
12	IOCTL_1588_TARG_TIME_POLL	Poll the target time reached/hit condition status.
13	IOCTL_1588_TARG_TIME_SET	Set the Target Time to a given value.
14	IOCTL_1588_TARG_TIME_GET	Get the Target Time to a given buffer.
15	IOCTL_1588_AUX_TIME_INTRPT_ENABLE	Enable the Auxiliary Master/Slave Time stamp Interrupt.
16	IOCTL_1588_AUX_TIME_INTRPT_DISABLE	Disable the Auxiliary Master/Slave Time stamp Interrupt.
17	IOCTL_1588_AUX_TIME_POLL	Polls for the Time stamp in the appropriate Auxiliary Snapshot Registers.
18	IOCTL_1588_RESET	Reset IEEE 1588 Hardware Assist block.
19	IOCTL_1588_CHNL_RESET	Behaves the same as function IOCTL_1588_RESET.
20	IOCTL_1588_STATS_GET	Return the statistics of the received or transmitted messages.
21	IOCTL_1588_STATS_RESET	Resets the statistics counters.
22	IOCTL_1588_SHOW_ALL	Display the current configuration of the IEEE 1588 registers.
23	IOCTL_1588_AUX_TARG_TIME_INTRPT_ENABLE	Implementation returns an error to notify unsupported function.

Table 2. Supported IOCTLs (Sheet 2 of 2)

No.	Interface	Description
24	IOCTL_1588_AUX_TARG_TIME_INTRPT_DISABLE	Implementation returns an error to notify unsupported function.
25	IOCTL_1588_AUX_TARG_TIME_POLL	Implementation returns an error to notify unsupported function.
26	IOCTL_1588_AUX_TARG_TIME_SET	Implementation returns an error to notify unsupported function.
27	IOCTL_1588_AUX_TARG_TIME_GET	Implementation returns an error to notify unsupported function.
28	IOCTL_1588_PULSE_PER_SEC_INTRPT_ENABLE	Enable the pulse per second interrupt.
29	IOCTL_1588_PULSE_PER_SEC_INTRPT_DISABLE	Disable the pulse per second interrupt.
30	IOCTL_1588_TARG_TIME_NOTIFY	Notify on a Target Time interrupt event. The behavior of this IOCTL is in such a way that, the IOCTL will block until either, 1) A Target Time interrupt event is generated 2) The IOCTL_1588_TARG_TIME_CLR_NOTIFY IOCTL is called.
31	IOCTL_1588_AUX_TIME_NOTIFY	Notify on an Auxiliary Time interrupt event. The behavior of this IOCTL is in such a way that, the IOCTL will block until either, 1) An Auxiliary Time interrupt event is generated 2) The IOCTL_1588_AUX_TIME_CLR_NOTIFY IOCTL is called.
32	IOCTL_1588_AUX_TARG_TIME_NOTIFY	Implementation returns an error to notify unsupported function.
33	IOCTL_1588_PULSE_PER_SEC_NOTIFY	Notify on pulse per second interrupt.
34	IOCTL_1588_TARG_TIME_CLR_NOTIFY	Force a release of a blocked IOCTL_1588_TARG_TIME_NOTIFY IOCtl call.
35	IOCTL_1588_AUX_TIME_CLR_NOTIFY	Force a release of a blocked IOCTL_1588_AUX_TIME_NOTIFY IOCTL call.
36	IOCTL_1588_AUX_TARG_TIME_CLR_NOTIFY	Implementation returns an error to notify unsupported function.
37	IOCTL_1588_PULSE_PER_SEC_CLR_NOTIFY	Force a release of a blocked IOCTL_1588_PULSE_PER_SEC_NOTIFY IOCtl call.
38	IOCTL_1588_PULSE_PER_SEC_TIME_GET	Get the PPS time.
39	IOCTL_1588_PULSE_PER_SEC_TIME_SET	Set the PPS time.
40	IOCTL_1588_PORT_VERSION_SET	Sets the version of the PTP port. Valid values are 1 and 2.
41	IOCTL_1588_PORT_VERSION_GET	Get the version support of PTP port. Valid values are 1 and 2.
42	IOCTL_1588_PORT_OPERATION_MODE_SET	Set the operation mode of the channel.
43	IOCTL_1588_PORT_OPERATION_MODE_GET	Get the operation mode of the channel

4.4 IOCTL Usage Details

This section provides the details to configure the IEEE1588 interface and initiate IEEE1588 operations. The following files contain the details of the IOCTLs and data structures used:

- ioh_1588_ioctls.h – contains IOCTL definitions
- ioh_1588_common.h – data structures and other variables used by the IOCTLs

For the programming details refer to Chapter 5.0, “Programming Guide”.

4.5 Structures and Enumerations

This section provides the structures and enumerations used by interfaces exposed by the IEEE1588 driver. All the structures and enumerations used by the interfaces are defined in ioh_1588_common.h.

4.5.1 Structures

4.5.1.1 IOH1588PORTCFGIOCTL

IEEE 1588 Hardware Assist port configuration data.

Table 3. IOH1588PORTCFGIOCTL structure

Name	Description
IOH1588PTPPORT ptpPort	PTP Communication Port
IOH1588PTPPORTMODE ptpPortMode	PTP Port mode (Master, Slave. Any Mode)

4.5.1.2 IOH1588TIMEVALUE

Structure to hold 64-bit system time and time stamp value.

Table 4. IOH1588TIMEVALUE structure

Name	Description
ULONG timeValueLowWord	Lower 32 bits of time value
ULONG timeValueHighWord	Upper 32 bits of time value

4.5.1.3 IOH1588UUID

Structure to hold 48-bit UUID values captured in Sync or Delay_Req messages.

Table 5. IOH1588UUID structure

Name	Description
ULONG uuidValueLowWord	The lower 32 bits of UUID
ULONG uuidValueHighHalfword	The upper 16 bits of UUID

4.5.1.4 IOH1588PTPMSGDATA

Structure for data from the PTP message returned when TimeStamp available.

Table 6. IOH1588PTPMSGDATA structure

Name	Description
IOH1588PTPMSGTYPE ptpMsgType	PTP Messages type
IOH1588TIMEVALUE ptpTimeStamp	64-bit TimeStamp value from PTP Message
IOH1588UUID ptpUuid	48-bit UUID value from the PTP Message
UINT16 ptpSequenceNumber	16-bit Sequence Number from PTP Message

4.5.1.5 IOH1588RXTXPOLLIOCTL

Structure to pass PTP message data.

Table 7. IOH1588RXTXPOLLIOCTL structure

Name	Description
IOH1588PTPPORT ptpPort	IEEE 1588 PTP Communication Port
IOH1588PTPMSGDATA ptpMsgData	PTP message data

4.5.1.6 IOH1588CANPOLLIOCTL

Structure to pass CAN timestamp data.

Table 8. IOH1588RXTXPOLLIOCTL structure

Name	Description
IOH1588PTPPORT ptpPort	IEEE 1588 PTP Communication Port
IOH1588TIMEVALUE ptpTimeStamp	CAN PTP timestamp

4.5.1.7 IOH1588TIMEPOLLIOCTL

Structure to pass timestamp data.

Table 9. IOH1588TIMEPOLLIOCTL structure

Name	Description
ULONG pollFlag	Time event
IOH1588TIMEVALUE timeVal	Timestamp value
IOH1588AUXMODE auxMode	Master or Slave mode

4.5.1.8 IOH1588STATS

Provides the number of times timestamps are locked for rx and tx PTP messages.

Table 10. IOH1588STATS structure

Name	Description
ULONG rxMsgs	Count of timestamps for received PTP Msgs
ULONG txMsgs	Count of timestamps for transmitted PTP Msgs

4.5.1.9 IOH1588AUXTIMEIOCTL

Structure to pass aux time data.

Table 11. IOH1588AUXTIMEIOCTL structure

Name	Description
IOH1588AUXMODE auxMode	Aux mode: master or slave
IOH1588TIMEVALUE auxTime	Aux time snapshot

4.5.1.10 IOH1588VERSIONIOCTL

Structure to pass PTP port version information.

Table 12. IOH1588VERSIONIOCTL structure

Name	Description
IOH1588PTPPORT ptpPort	IEEE 1588 PTP Communication Port
IOH1588PTPVERSION ptpVersion	PTP Version value

4.5.1.11 IOH1588OPERATIONMODEIOCTL

Structure to pass the operation mode information.

Table 13. IOH1588OPERATIONMODEIOCTL structure

Name	Description
IOH1588PTPPORT ptpPort	IEEE 1588 PTP Communication Port
IOH1588PTPOPERATIONMODE ptpOpMode	IEEE 1588 operation mode

4.5.2 Enumerations

This section lists the enumerations exposed by the interface.

4.5.2.1 IOH1588PTPPORT

IEEE 1588 PTP Communication Port (Channel).

Table 14. IOH1588PTPPORT enumeration

Name	Description
IOH_1588_GBE_0_1588PTP_PORT	PTP Communication Port on GBE-0
IOH_1588_CAN_0_1588PTP_PORT	PTP Communication Port on CAN-0
IOH_1588_PORT_INVALID	Invalid PTP Communication Port

4.5.2.2 IOH1588PTPPORTMODE

PTP port mode information.

Table 15. IOH1588PTPPORTMODE enumeration

Name	Description
IOH_1588PTP_PORT_MASTER	Master Mode
IOH_1588PTP_PORT_SLAVE	Slave Mode
IOH_1588PTP_PORT_ANYMODE	Timestamp all messages
IOH_1588PTP_PORT_MODE_INVALID	Invalid PTP Port Mode

4.5.2.3 IOH1588PTPMSGTYPE

PTP Messages types that can be detected on communication port.

Table 16. IOH1588PTPMSGTYPE enumeration

Name	Description
IOH_1588PTP_MSGTYPE_SYNC	PTP Sync message sent by Master or received by Slave
IOH_1588PTP_MSGTYPE_DELAYREQ	PTP Delay_Req message sent by Slave or received by Master
IOH_1588PTP_MSGTYPE_UNKNOWN	Other PTP and non-PTP message sent or received by both Master and/or Slave

4.5.2.4 IOH1588AUXMODE

PTP Messages types that can be detected on communication port.

Table 17. IOH1588AUXMODE enumeration

Name	Description
IOH_1588_AUXMODE_MASTER	Auxiliary Master Mode
IOH_1588_AUXMODE_SLAVE	Auxiliary Slave Mode
IOH_1588_AUXMODE_INVALID	Invalid Auxiliary Mode

4.5.2.5 IOH1588PTPVERSION

PTP version value that can be detected on communication port.

Table 18. IOH1588PTPVERSION enumeration

Name	Description
IOH_1588_PTP_VERSION_0	Support version 1 only
IOH_1588_PTP_VERSION_1	Support both version 1 and version 2
IOH_1588_PTP_VERSION_INVALID	Invalid version

4.5.2.6 IOH1588PTPOPERATIONMODE

PTP operation mode value that can be detected on communication port.

Table 19. IOH1588PTPOPERATIONMODE enumeration

Name	Description
IOH_1588PTP_OP_MODE_SYNC_DELAYREQ_MSGS	Timestamp version 1 SYNC and DELAYED_REQ only
IOH_1588PTP_OP_MODE_V1_ALL_MSGS	Timestamp version 1 all messages
IOH_1588PTP_OP_MODE_V1_V2_EVENT_MSGS	Timestamp version 1 and 2 event messages only
IOH_1588PTP_OP_MODE_V1_V2_ALL_MSGS	Timestamp version 1 and 2 all messages
IOH_1588PTP_OP_MODE_INVALID	Invalid mode

4.6 Error Handling

Since the IOCTL command is implemented using Windows* APIs, the return value of the call is dependent on and defined by the OS. On Windows*, the return value is a non-zero value. If the error is detected within or outside the driver, an appropriate system defined value is returned by the driver.

5.0 Programming Guide

This section describes the basic procedure to use the IEEE1588 driver from a user-mode application. All operations are through the IOCTLs exposed by the IEEE1588 driver. Refer to Section 4.3 for details on the IOCTLs. The steps involved in accessing the IEEE1588 driver from the user mode application are described below:

- Open the device
- Configure the device for different modes of operations
- Close the device

5.1 Opening the Device

The IEEE1588 Device is opened using CreateFile Win32 API. To get the device name, refer to Section 5.1.1.

5.1.1 Using GUID Interface Exposed by the Driver

A device interface class is a way of exporting device and driver functionality to other system components, including other drivers, as well as user-mode applications. A driver can register a device interface class, and then enable an instance of the class for each device object to which user-mode I/O requests might be sent. The Intel® PCH EG20T IEEE1588 driver registers the following interface.

No	Interface Name
1	GUID_DEVINTERFACE_IOHIEEE1588

This is defined in ioh_1588_common.h.

Device interfaces are available to both kernel-mode components and user-mode applications. User-mode code can use **SetupDi**Xxx functions to find out about registered, enabled device interfaces.

Please refer to the following site for details about SetupDiXxx functions.

http://msdn.microsoft.com/en-us/library/dd406734.aspx

5.2 Device Functionality

This section describes the configuration of the device to achieve time synchronization on Ethernet and CAN.

5.2.1 PTP Port Mode Configuration

This section describes set and get IOCTLs that are used to configure and query Ethernet PTP ports. The IOCTLs used for port mode configuration are:

- IOCTL_1588_PORT_CONFIG_SET
- IOCTL_1588_PORT_CONFIG_GET

The following code snippet sets the PTP port mode as master. Other PTP modes are:

- IOH_1588PTP_PORT_SLAVE
- IOH_1588PTP_PORT_ANYMODE

```
iohConfig.ptpPort = IOH_1588_GBE_0_1588PTP_PORT;
```

```
iohConfig.ptpPortMode = IOH_1588PTP_PORT_MASTER;
bRet = DeviceIoControl( hDevice,
    IOCTL_1588_PORT_CONFIG_SET,
    &iohConfig,
    sizeof(IOH1588PORTCFGIOCTL),
    NULL,
    0,
    &dwRet,
    NULL);
```

To get the current PTP port configuration details use IOCTL_1588_PTP_CONFIG_GET.

```
iohConfigIn.ptpPort = IOH_1588_GBE_0_1588PTP_PORT;
iohConfigIn.ptpPortMode = IOH_1588PTP_PORT_MASTER;
IOH1588PORTCFGIOCTL iohConfgOut = {0};
bRet = DeviceIoControl( hDevice,
    IOCTL_1588_PORT_CONFIG_GET,
    &iohConfigIn,
    sizeof(IOH1588PORTCFGIOCTL),
    &iohConfigOut,
    sizeof(IOH1588PORTCFGIOCTL),
    &dwRet,
    NULL);
```

5.2.2 PTP Message TimeStamp Data

The following IOCTLs are used for getting Receive and Transmit timestamp information.

- IOCTL_1588_TX_POLL

```
IOH1588RXTXPOLLIOCTL rtPoll = {0};
rtPoll.ptpPort = IOH_1588_GBE_0_1588PTP_PORT;

bRet = DeviceIoControl(hDevice,
        IOCTL_1588_TX_POLL,
        &rtPoll,
        sizeof(IOH1588RXTXPOLLIOCTL),
        &rtPoll,
        sizeof(IOH1588RXTXPOLLIOCTL),
```

```
        &dwRet,
        NULL);
```

- IOCTL_1588_RX_POLL

```
IOH1588RXTXPOLLIOCTL rtPoll = {0};
rtPoll.ptpPort = IOH_1588_GBE_0_1588PTP_PORT;

bRet = DeviceIoControl(hDevice,
        IOCTL_1588_RX_POLL,
        &rtPoll,
        sizeof(IOH1588RXTXPOLLIOCTL),
        &rtPoll,
        sizeof(IOH1588RXTXPOLLIOCTL),
        &dwRet,
        NULL);
```

5.2.3 CAN PTP Message TimeStamp Data

To get the CAN PTP message timestamp, use the following IOCTL.

- IOCTL_1588_CAN_POLL

```
IOH1588CANPOLLIOCTL iohCanPoll = {0};
iohCanPoll.ptpPort = IOH_1588_CAN_0_1588PTP_PORT;

bRet = DeviceIoControl( hDevice,
        IOCTL_1588_CAN_POLL,
        &iohCanPoll,
        sizeof(IOH1588CANPOLLIOCTL),
        &iohCanPoll,
        &sizeof(IOH1588CANPOLLIOCTL),
        &dwRet,
        NULL);
```

5.2.4 System Time

The following IOCTLs are used to set and get system time information from the IEEE1588 hardware assist.

- IOCTL_1588_SYS_TIME_SET

```
IOH1588TIMEVALUE iohTimeVal = {0};
iohTimeVal.timeValueHighWord = 0xaa;
```

```
iohTimeVal.timeValueLowWord = 0xbb;

bRet = DeviceIoControl(hDevice,

            IOCTL_1588_SYS_TIME_SET,

            &iohTimeVal,

            sizeof(IOH1588TIMEVALUE),

            NULL,

            0,

            &dwRet, NULL);
```

- IOCTL_1588_SYS_TIME_GET

```
IOH1588TIMEVALUE iohTimeVal = {0};


bRet = DeviceIoControl(hDevice,

               IOCTL_1588_SYS_TIME_GET,

               NULL,

               0,

               &iohTimeVal,

               sizeof(IOH1588TIMEVALUE),

               &dwRet, NULL);
```

5.2.5 Frequency Scaling Value

This section describes the set and get IOCTLs that are used to configure and get the frequency scaling value (tick rate). The following IOCTLs are used for this operation:

- IOCTL_1588_TICK_RATE_SET

```
ULONG uRate = 0xabcd1234;


bRet = DeviceIoControl(hDevice,

        IOCTL_1588_TICK_RATE_SET,

        &uRate,

        sizeof(ULONG),

        NULL,

        0,

        &dwRet,NULL);
```

- IOCTL_1588_TICK_RATE_GET

```
ULONG uRate = 0;

bRet = DeviceIoControl(hDevice,
```

```
        IOCTL_1588_TICK_RATE_GET,

        NULL,

        0,

        &uRate,

        sizeof(ULONG),

        &dwRet,

        NULL);
```

5.2.6 Target Time

This section describes the set, get poll, and notification IOCTLs that are used to configure, query, and poll the target time. It also describes the functionality to enable or disable the interrupt processing of the target time reached/hit condition. The following IOCTLS are used for the operation:

- IOCTL_1588_TARGET_TIME_SET

```
IOH1588TIMEVALUE iohTimeVal ={0};


iohTimeVal.timeValueHighWord = 0xbb;

iohTimeVal.timeValueLowWord = 0xaa;


bRet = DeviceIoControl(hDevice,

        IOCTL_1588_TARG_TIME_SET,

        NULL,

        0, &iohTimeVal,sizeof(IOH1588TIMEVALUE),&dwRet,NULL);
```

- IOCTL_1588_TARGET_TIME_GET

```
IOH1588TIMEVALUE iohTimeVal ={0};


bRet = DeviceIoControl(hDevice,

        IOCTL_1588_TARG_TIME_GET,

        NULL,

        0,

        &iohTimeVal,

        sizeof(IOH1588TIMEVALUE),

        &dwRet,NULL);
```

- IOCTL_1588_TARG_TIME_INTRPT_ENABLE

```
bRet = DeviceIoControl(hDevice,

        IOCTL_1588_TARG_TIME_INTRPT_ENABLE,
```

```
        NULL,
        0,
        NULL,
        0,
        &dwRet,
        NULL);
```

- IOCTL_1588_TARG_TIME_INTRPT_DISABLE

```
bRet = DeviceIoControl(hDevice,
        IOCTL_1588_TARG_TIME_INTRPT_DISABLE,
        NULL,
        0,
        NULL,
        0,
        &dwRet,
        NULL);
```

- IOCTL_1588_TARG_TIME_POLL

```
IOH1588TIMEPOLLIOCTL iohTimePoll = {0};
DWORD dwSize = sizeof(IOH1588TIMEPOLLIOCTL);

bRet = DeviceIoControl(hDevice,
        IOCTL_1588_TARG_TIME_POLL,
        &iohTimePoll,
        dwSize,
        &iohTimePoll,dwSize,&dwRet,NULL);
```

- IOCTL_1588_TARG_TIME_NOTIFY
  IOCTL_1588_TARG_TIME_INTRPT_ENABLE must be called before calling this IOCTL.

```
IOH1588TIMEVALUE iohTimeVal = {0};

bRet = DeviceIoControl(hDevice,
        IOCTL_1588_TARG_TIME_NOTIFY,
        NULL,
        0,
        &iohTimeVal,
        dwSize,
```

```
        &dwRet,NULL);
```

- IOCTL_1588_TARG_TIME_CLR_NOTIFY
  This must be called from a separate thread, since the thread with the call IOCTL_1588_TARG_TIME_NOTIFY is waiting to get the notification from the driver.

```
bRet = DeviceIoControl(hDevice,

        IOCTL_1588_TARG_TIME_CLR_NOTIFY,

        NULL,

        0,

        NULL,

        0,

        &dwRet,

        NULL);
```

5.2.7 Pulse Per Second (PPS)

This section describes the IOCTLs used to get and set the pulse per second time value, enable or disable pulse per second notification interrupt, and clear the notification interrupt. The following IOCTLs are used for the operations:

- IOCTL_1588_PULSE_PER_SEC_TIME_SET

```
ULONG uTime =0xffffffff;

DWORD dwSize = sizeof(ULONG);


bRet = DeviceIoControl(hDevice,

        IOCTL_1588_PULSE_PER_SEC_TIME_SET,

        &uTime,dwSize, NULL,0,&dwRet, NULL);
```

- IOCTL_1588_PULSE_PER_SEC_TIME_GET

```
ULONG uTime =0;

DWORD dwSize = sizeof(ULONG);

bRet = DeviceIoControl(hDevice,

        IOCTL_1588_PULSE_PER_SEC_TIME_GET,

        NULL,0,&uTime, dwSize,&dwRet,NULL);
```

- IOCTL_1588_PULSE_PER_SEC_INTRPT_ENABLE

```
bRet = DeviceIoControl(hDevice,

        IOCTL_1588_PULSE_PER_SEC_INTRPT_ENABLE,

        NULL,0,NULL,0,&dwRet,NULL);
```

- IOCTL_1588_PULSE_PER_SEC_INTRPT_DISABLE

```
bRet = DeviceIoControl(hDevice,

        IOCTL_1588_PULSE_PER_SEC_INTRPT_DISABLE,
```

```
        NULL,0,NULL,0,&dwRet,NULL);
```

- IOCTL_1588_PULSE_PER_SEC_NOTIFY
  Before calling this, IOCTL_1588_PULSE_PER_SEC_INTRPT_ENABLE must be called.

```
ULONG uPPS = 0;


bRet = DeviceIoControl( hDevice,

        IOCTL_1588_PULSE_PER_SEC_NOTIFY,

        NULL,0,&uPPS,dwSize,&dwRet,NULL );
```

- IOCTL_1588_PULSE_PEC_SEC_CLR_NOTIFY
  This must be called from a separate thread, since the thread with the call IOCTL_1588_TARG_TIME_NOTIFY is waiting to get the notification from the driver.

```
bRet = DeviceIoControl(hDevice,

        IOCTL_1588_PULSE_PER_SEC_CLR_NOTIFY,

        NULL,0,NULL,0,&dwRet,NULL);
```

5.2.8 Reset

The following IOCTLs are used for resetting the device and the channel:

- IOCTL_1588_RESET

```
bRet = DeviceIoControl( hDevice,

        IOCTL_1588_RESET,NULL,0,NULL,0,&dwRet,NULL);
```

- IOCTL_1588_CHNL_RESET

```
bRet = DeviceIoControl( hDevice,

        IOCTL_1588_CHNL_RESET,

        NULL,0,NULL,0,&dwRet,NULL);
```

5.3 Closing the Device

Once all the operations related to the IEEE1588 driver are complete, the application must free the device handle by calling the Win32 API CloseHandle.

```
CloseHandle(hHandle);
```