Okuma America Corporation

THINC-API
Release Notes for Lathe

Document No.: S5015-008-23

Revision History

Date	Version	Description	Author
5/21/2007	S5015-008-00	Public release for Lathe THINC-API version 1.0.0.0	LHuynh
6/04/2007	S5015-008-01	Public release for Lathe THINC-API version 1.1.0.0	Lhuynh
08/15/2007	S5015-008-02	Public release for Lathe THINC-API version 1.2.0.0	Lhuynh
2/22/2008	S5015-008-03	Public Release 1.6.0.0 for Lathe THINC-API	Lhuynh
4/11/2008	S5015-008-04	Public Release 1.6.0.0 for Lathe THINC-API	Lhuynh
06/27/2008	S5015-008-05	Public Release 1.6.3.0 for Lathe THINC-API	Lhuynh
07/25/2008	S5015-008-06	Public Release 1.6.4.0 for Lathe THINC-API	Lhuynh
10/27/2008	S5015-008-07	Public Release 1.7.0.0 for Lathe THINC-API	Lhuynh
12/18/2008	S5015-008-08	Public Release 1.8.0.0 for Lathe THINC-API	Lhuynh
04/30/2009	S5015-008-09	Public Release 1.9.1.0 for Lathe THINC-API	Lhuynh
10/12/2009	S5015-008-10	Public Release 1.10.0.0 for Lathe THINC-API	Lhuynh
04/28/2010	S5015-008-11	Public Release 1.11.0.0 for Lathe THINC-API	Lhuynh
09/14/2010	S5015-008-12	Public Release 1.11.1.0 for Lathe THINC-API	Lhuynh
01/04/2011	S5015-008-13	Public Release 1.12.0.0 for Lathe THINC-API	Lhuynh
02/02/2011	S5015-008-14	Public Release 1.12.1.0 for Lathe THINC-API	Lhuynh
11/14/2011	S5015-008-15	Beta Release 1.14.0.0 for Lathe THINC-API	Lhuynh
01/15/2012	S5015-008-16	Beta Release 1.14.1.0 for Lathe THINC-API	Lhuynh
09/21/2012	S5015-008-17	Beta Release 1.14.2.0 for Lathe THINC-API	Lhuynh
12/04/2012	S5015-008-18	Public Release 1.15.0.0 for Lathe THINC-API	Lhuynh

03/06/2013	S5015-008-19	Beta Release 1.15.1.0 for Lathe THINC-API	Lhuynh
03/18/2013	S5015-008-20	Beta Release 1.15.2.0 for Lathe THINC-API	Lhuynh
09/25/2013	S5015-008-21	Public Release 1.16.0.0 for Lathe THINC-API	Lhuynh
01/15/2014	S5015-008-22	Public Release 1.17.0.0 for Lathe THINC-API	Lhuynh
04/01/2014	S5015-008-23	Public Release 1.17.1.0 for Lathe THINC-API	Lhuynh

Table of Contents

Release Notes for Lathe

1. Introduction

1.1 Disclaimer of Warranty

Okuma America Corporation makes no representations or warranties, either expressed or implied, by or with respect to anything in this document, and shall not be liable for any implied warranties of merchantability or fitness for a particular purpose or for any indirect, special or consequential damages.

1.2 Purpose

The purpose of the ***Release Notes*** document is to communicate major new features and changes in this release of the THINC-API for Lathe libraries. It also documents known problems and workarounds.

1.3 Scope

This document describes Public Release 1.17.1.0 of THINC-API for Lathe.

1.4 Definitions, Acronyms, and Abbreviations

GAC – Global Assembly Cache Windows folder located in 'C:\WINDOWS\assembly'

1.5 References

None.

2. About This Release

Public Release 1.17.1.0 of the THINC-API library for Lathe supports the following:

From this release, THINC-API libraries will check dependency libraries during installation and at run-time. THINC-API will fail to install or load if version of dependency libraries cannot support current version of THINC-API.

From this release and forward, API Notifier will delay the checking of API for an approximately of 1 minutes or so after NC is running.

All applications compiled with Beta Release from version 1.15.X.X must be compiled with Public Release version 1.17.0.0 or higher when it is available.

In current version of THINC-API, some of the existing functions related to ATC, Tool, TailStock, and Chuck Data from DATA-API or Command API might not function correctly on OSP-P300S (SLP) and OSP-P300L control unless they have been revised as listed in the Change section 3.2.

Please refer to the help file for detail usage and compatibility information of each function.
This version requires latest OSP system disk.

Version of Okuma.CLDATAPI.dll in this release is 2.6.1.0

Version of Okuma.CLCMDAPI.dll in this release is 2.0.0.0

Version of APINotifierService.exe in this release is 1.17.3.0

Version of APINotifierStatus.exe in this release is 1.1.1.0

Version of Okuma.Flexnet.dll in this release is 1.3.0.0

Version of Okuma.Apilog.dll in this release is 1.4.0.0

This release requires OCJ API version 003P

The following functions of Okuma.CLDATAPI.dll library will not be available in OSP-P100II control machines:

Classes	Interfaces
CMachine	OnOffStateEnum GetNCStatus(NCStatusEnum enNCStatus)
CMachine	CCurrentAlarm* GetCurrentAlarm()
CIO	OnOffStateEnum GetUserTaskIOVariable(IOTypeEnum enIO, Int32 intIndex) ;
CIO	void SetUserTaskOutputVariable(Int32 intIndex, OnOffStateEnum enValue) ;
CIO	OnOffStateEnum GetProtectedUserTaskOutputVariable(Int32 intIndex) ;
CIO	void SetProtectedUserTaskOutputVariable(Int32 intIndex, OnOffStateEnum enValue) ;

The following functions of Okuma.CLCMDAPI.dll library will not be available in OSP-P100II control machines:

Classes	Interfaces
CATC	Sub RegisterToolPot(ByVal intPotNo As Integer, ByVal intToolNo As Integer, ByVal enSettingToolKind As SettingToolKindEnum, ByVal enSettingToolSize As SettingToolSizeEnum, ByVal enReturnMagazine As ReturnMagazineEnum)
CATC	Sub SetNextTool(ByVal intToolNo As Integer, ByVal enSettingToolKind As SettingToolKindEnum, ByVal enSettingToolSize As SettingToolSizeEnum, ByVal enReturnMagazine As ReturnMagazineEnum)
CATC	Sub SetToolInStation(ByVal intToolNo As Integer, ByVal enSettingToolKind As SettingToolKindEnum, ByVal enSettingToolSize As SettingToolSizeEnum, ByVal enReturnMagazine As ReturnMagazineEnum, ByVal enTurretStation As TurretStationEnum)
CATC	Sub UnRegisterToolPot(ByVal intPotNo As Integer)
CProgram	CancelMainProgram()
CProgram	SelectMainProgramRSide(ByVal strMainProgramFileName As String, Optional ByVal strSubProgramFileName As String = "", Optional ByVal strSystemSubtituteProgramFileName As String = "", Optional ByVal strProgramName As String = "")
CProgram	SelectMainProgramLSide(ByVal strMainProgramFileName As String, Optional ByVal strSubProgramFileName As String = "", Optional ByVal strSystemSubtituteProgramFileName As String = "", Optional ByVal strProgramName As String = "")

CTools	CalcualteToolOffset(ByVal intToolNo As Integer, ByVal enAxisIndex As OffsetAxisIndexEnum, ByVal enSubSystem As SubSystemEnum, ByVal dblValue As Double)
CTools	AddConstantToolOffset(ByVal intOffsetNo As Integer, ByVal enAxisIndex As OffsetAxisIndexEnum, ByVal enSubSystem As SubSystemEnum, ByVal enCuttingPosition As CuttingPositionEnum)
CTools	AddConstantNoseRadiusCompensation(ByVal intOffsetNo As Integer, ByVal enAxisIndex As OffsetAxisIndex2Enum, ByVal enSubSystem As SubSystemEnum, ByVal enCuttingPosition As CuttingPositionEnum)
CTools	AddConstantToolWear(ByVal intOffsetNo As Integer, ByVal enAxisIndex As OffsetAxisIndex2Enum, ByVal enSubSystem As SubSystemEnum, ByVal enCuttingPosition As CuttingPositionEnum)
CTools	SubtractConstantToolOffset(ByVal intOffsetNo As Integer, ByVal enAxisIndex As OffsetAxisIndexEnum, ByVal enSubSystem As SubSystemEnum, ByVal enCuttingPosition As CuttingPositionEnum)
CTools	SubtractConstantNoseRadiusCompensation(ByVal intOffsetNo As Integer, ByVal enAxisIndex As OffsetAxisIndex2Enum, ByVal enSubSystem As SubSystemEnum, ByVal enCuttingPosition As CuttingPositionEnum)
CTools	SubtractConstantToolWear(ByVal intOffsetNo As Integer, ByVal enAxisIndex As OffsetAxisIndex2Enum, ByVal enSubSystem As SubSystemEnum, ByVal enCuttingPosition As CuttingPositionEnum)
CProgram	SelectScheduleProgramLSide(ByVal strScheduleProgramFileName As String)
CProgram	SelectScheduleProgramRSide(ByVal strScheduleProgramFileName As String)
CMachine	Public Sub SetUserAlarm(ByVal enAlarm As UserAlarmEnum, Optional ByVal strAlarmMessage As String = "", Optional ByVal enUserAlarmSubSystem As UserAlarmSubSystemEnum = 0)
CMachine	Public Sub ClearUserAlarmD(ByVal enUserAlarmSubSystem As UserAlarmSubSystemEnum)

Note: User Alarm will require OKUMA.Lathe.UserAlarm license feature in order to function on OSP-P200 machine if machine can support.

3. New Features

3.1.1 DATA-API

Control	Classes	Interfaces
P300L P300S	CATC	ToolPotProperty GetToolPot(Int32 intToolNo)
P100II P200 P300L P300s	CMachine	Boolean GetTDMode()
P300S	CMachine	Boolean SetTDMode()

P100II P200 P300S P300L	CSpindle	Double GetCurrentSpindlePosition()
P300L P300S	CTools	Boolean IsMountedTool(Int32 intToolNo)
P300L P300S	CTools	Boolean IsMultiEdgesTool(Int32 intToolNo)
P300L P300S	CTools	Boolean IsRegisteredEdge(Int32 intToolNo)
P300L P300S	CTools	Boolean IsRegisteredTool(Int32 intToolNo)
P300L P300S	CTools	Int32 GetRegisteredCuttingPositions(Int32 intToolNo, ToolEdgeEnum enEdgeNo)[]
P300L P300S	CTools	Int32 GetRegisteredToolEdges(Int32 intToolNo)[];
P300L P300S	CTools	ToolKind2Enum GetToolKind(Int32 intToolNo)
P300L P300S	CTools	Boolean IsActiveCuttingPosition(Int32 intToolNo, ToolEdgeEnum enEdgeNo, ToolCuttingPositionEnum enToolCuttingPosition)
P300S	CTools	Double GetToolOffset(Int32 intToolNo, ToolOffsetAxisIndexEnum enAxisIndex, ToolEdgeEnum enEdgeNo, ToolCuttingPositionEnum enToolCuttingPosition) void SetToolOffset(Int32 intToolNo, ToolOffsetAxisIndexEnum enAxisIndex, ToolEdgeEnum enEdgeNo, ToolCuttingPositionEnum enToolCuttingPosition, Double dblValue) void AddToolOffset(Int32 intToolNo, ToolOffsetAxisIndexEnum enAxisIndex, ToolEdgeEnum enEdgeNo, ToolCuttingPositionEnum enToolCuttingPosition, Double dblValue) void CalToolOffset(Int32 intToolNo, ToolOffsetAxisIndexEnum enAxisIndex, ToolEdgeEnum enEdgeNo, ToolCuttingPositionEnum enToolCuttingPosition, Double dblValue)

P300S	CTools	Double GetToolWearOffset(Int32 intToolNo, ToolWearOffsetAxisIndexEnum enAxisIndex, ToolCuttingPositionEnum enToolCuttingPosition , ToolEdgeEnum enEdgeNo) void AddToolWearOffset(Int32 intToolNo, ToolWearOffsetAxisIndexEnum enAxisIndex, ToolCuttingPositionEnum enToolCuttingPosition, ToolEdgeEnum enEdgeNo, Double dblValue) void SetToolWearOffset(Int32 intToolNo, ToolWearOffsetAxisIndexEnum enAxisIndex, ToolCuttingPositionEnum enToolCuttingPosition, ToolEdgeEnum enEdgeNo, Double dblValue)
P300S	CTools	Int32 GetReferenceToolOffset1(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetReferenceToolOffset1(Int32 intToolNo, ToolEdgeEnum enEdgeNo, Int32 intOffsetNo)
P300S	CTools	Int32 GetReferenceToolOffset1(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetReferenceToolOffset1(Int32 intToolNo, ToolEdgeEnum enEdgeNo, Int32 intOffsetNo)
P300S	CTools	Int32 GetReferenceToolOffset2(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetReferenceToolOffset2(Int32 intToolNo, ToolEdgeEnum enEdgeNo, Int32 intOffsetNo)
P300S	CTools	Int32 GetReferenceToolOffset2(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetReferenceToolOffset2(Int32 intToolNo, ToolEdgeEnum enEdgeNo, Int32 intOffsetNo)
P300S	CTools	Int32 GetReferenceToolOffset3(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetReferenceToolOffset3(Int32 intToolNo, ToolEdgeEnum enEdgeNo, Int32 intOffsetNo)
P300S	CTools	Int32 GetReferenceToolOffset3(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetReferenceToolOffset3(Int32 intToolNo, ToolEdgeEnum enEdgeNo, Int32 intOffsetNo)
P300S	CTools	String* GetToolNumberInGroup3(Int32 intGroupNo) void SetToolNumberInGroup3(Int32 intGroupNo, Int32 intToolNo, ToolEdgeEnum enEdgeNo)

P300S	CTools	String* GetToolNumberInGroup3(Int32 intGroupNo) void SetToolNumberInGroup3(Int32 intGroupNo, Int32 intToolNo, ToolEdgeEnum enEdgeNo)
P300S	CTools	Int32 GetGroupNo(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetGroupNo(Int32 intToolNo, ToolEdgeEnum enEdgeNo, Int32 intValue)
P300S	CTools	String* GetToolNumberInGroup3(Int32 intGroupNo) void SetToolNumberInGroup3(Int32 intGroupNo, Int32 intToolNo, ToolEdgeEnum enEdgeNo)
P300S	CTools	ToolCuttingPositionTypeEnum GetToolCuttingPosition(Int32 intToolNo, ToolEdgeEnum enEdgeNo, ToolCuttingPositionEnum enToolCuttingPosition)
P300S	CTools	String* GetToolGroupEntry3(Int32 intGroupNo)[]
P300S	CTools	Double GetActualToolLife(ActualToolLifeConditionEnum enActualToolLifeCondition, Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetActualToolLife(ActualToolLifeConditionEnum enActualToolLifeCondition, Int32 intToolNo, ToolEdgeEnum enEdgeNo, Double dblValue) void AddActualToolLife(ActualToolLifeConditionEnum enActualToolLifeCondition, Int32 intToolNo, ToolEdgeEnum enEdgeNo, Double dblValue)
P300S	CTools	Double GetToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo, ToolEdgeEnum enEdgeNo, Double dblValue) void AddToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo, ToolEdgeEnum enEdgeNo, Double dblValue)
P300S	CTools	ToolGaugeStatusEnum GetGaugeStatus(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetGaugeStatus(Int32 intToolNo, ToolEdgeEnum enEdgeNo, ToolGaugeStatusEnum enValue)

P300S	CTools	ToolLifeStatusEnum GetLifeStatus(Int32 intToolNo, ToolEdgeEnum enEdgeNo) void SetLifeStatus(Int32 intToolNo, ToolEdgeEnum enEdgeNo, ToolLifeStatusEnum enValue)
P300S P300L	CTurret	Int32 GetToolNo(Int32 intTurretStationNo)
P300S P300L	CTurret	Int32 GetToolNo(Int32 intTurretStationNo)
P300L P300S	CWorkpiece	Double GetBaseZeroOffset(AxisIndex4Enum enAxisIndex) void SetBaseZeroOffset(AxisIndex4Enum enAxisIndex, Double dblValue) void AddBaseZeroOffset(AxisIndex4Enum enAxisIndex, Double dblValue) void CalBaseZeroOffset(AxisIndex4Enum enAxisIndex, Double dblValue)

3.1.2 Command-API

Control	Classes	Interfaces
P300S P300L	CATC	Public Sub AttachTool(ByVal intToolNo As Integer, ByVal intTarget As Integer, ByVal enToolLocation As ToolLocationEnum)
P300S P300L	CATC	Public Sub DeleteTool(ByVal intToolNo As Integer)
P300S	CATC	Public Sub DeleteToolCuttingPosition(ByVal intToolNo As Integer, ByVal intEdgeNo As Integer, ByVal enToolCuttingPosition As Enumerations.ToolCuttingPositionEnum)
P300S P300L	CATC	Public Sub DeleteToolEdge(ByVal intToolNo As Integer, ByVal enToolEdge As ToolEdgeEnum)
P300S P300L	CATC	Public Sub DetachTool(ByVal intTarget As Integer, ByVal enToolLocation As ToolLocationEnum)
P300L	CATC	Public Sub RegisterTool(ByVal intToolNo As Integer, ByVal intTargetNo As Integer, ByVal enSettingToolKind As Enumerations.SettingToolKindEnum, ByVal enSpindleAxis As Enumerations.BasicPositionSpindleEnum, ByVal enToolLocation As ToolLocationEnum)

P300S-TD	CATC	Public Sub RegisterTool(ByVal intToolNo As Integer, ByVal intTargetNo As Integer, ByVal enEdgeNo As ToolEdgeEnum, ByVal enSettingToolKind As Enumerations.SettingToolKindEnum, ByVal enSettingToolSize As Enumerations.SettingToolSizeEnum, ByVal enSpindleAxis As Enumerations.BasicPositionSpindleEnum, ByVal enBaseCuttingPositionSetting As Enumerations.BaseCuttingPositionSettingEnum, ByVal dblAngle As Double, ByVal enToolLocation As ToolLocationEnum)
P300S P300L	CATC	Public Sub RegisterToolCuttingPosition(ByVal intToolNo As Integer, ByVal enEdgeNo As ToolEdgeEnum, ByVal enToolCuttingPosition As Enumerations.ToolCuttingPositionEnum, ByVal enSpindleAxis As Enumerations.BasicPositionSpindleEnum, ByVal enBaseCuttingPositionSetting As Enumerations.BaseCuttingPositionSettingEnum, ByVal dblAngle As Double)
P300S P300L	CATC	Public Sub RegisterToolEdge(ByVal intToolNo As Integer, ByVal enEdgeNo As ToolEdgeEnum, ByVal enSpindleAxis As Enumerations.BasicPositionSpindleEnum, ByVal enBaseCuttingPositionSetting As Enumerations.BaseCuttingPositionSettingEnum, ByVal dblAngle As Double)

3.2 Change

The following functions have been modified to support data items on OSP-P300 control types.

Please refer to the help file for detail usage and compatibility information of each function.

3.2.1 DATA-API

Control	Classes	Interfaces
P100II P200 P300S	CATC	Int32 GetPotNo(Int32 intToolNo)
P100II P200 P300S	CATC	Int32 GetToolNo(Int32 intPotNo)

P100II P200 P300L P300S	CATC	Int32 GetToolNo(Int32 intPotNo)
P100II P200 P300S	CATC	ToolKindEnum GetToolKind(Int32 intPotNo)
P100II P200 P300L P300S-TL	CTools	Int32 GetGroupNo(Int32 intToolNo) ; void SetGroupNo(Int32 intToolNo, Int32 intGroupNo) ;
P100II P200 P300L P300S	CTools	Int32 GetMaxToolOffset()
P100II P200 P300L P300S	CTools	Int32 GetCurrentToolNumber()
P100II P200 P300L P300S	CTools	Double GetToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo) void SetToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo, Double dblValue) void AddToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo, Double dblValue)
P100II P200 P300L P300S	CTools	Double GetActualToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo) void SetActualToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo, Double dblValue) void AddActualToolLife(ToolLifeConditionEnum enToolLifeCondition, Int32 intToolNo, Double dblValue)
P100II P200 P300L P300S	CTools	Boolean GetLifeStatus(Int32 intToolNo) void SetLifeStatus(Int32 intToolNo,Boolean blnOK)
P100II P200 P300L P300S	CTools	Boolean GetGaugeStatus(Int32 intToolNo) void SetGaugeStatus(Int32 intToolNo,Boolean blnOK)

3.2.2 Update

The following functions have been revised since version 1.15.0.0 for Lathe.
Functions:
CProgram.ScheduleProgramCycleComplete(MachineSideEnum enMachineSide)
CProgram.CycleComplete(MachineSideEnum enMachineSide)
CProgram.GetProgramRunningState(ByVal enMachineSide As MachineSideEnum)
CProgram.GetActiveScheduleProgramFileName(ByVal enMachineSide As MachineSideEnum) A
CMachine.GetMachineSelection(MachineSideEnum enMachineSide)
CMachine::GetCurrentAlarm(MachineSideEnum enMachineSide)

The above functions have been revised to use First System or Right side system as default value on single side or dual side machine for consistency across all API functions.

4. Known Bugs and Limitations

This section identifies known problems in this release and describes any work-arounds.

4.1 Defect

4.2 General Defect

DATA-API library, Okuma.CLDATAPI.dll, cannot create directly under ASP.NET web application.

Solution/Work-arounds: Create and initialize DATA-API in a separate thread. All function calls must be called from objects created inside separated thread mentioned above.

DATA-API library can only support applications designed with single-threaded apartment of COM threading model. The underlying library, LDATAPI.dll, cannot be loaded during call to CMachine::Init function when an MTAThread attribute is applied to the application.

Solution/Work-arounds: None

4.2.1 Data-API

4.2.1.1 MacMan.COperationHistory class

Function:

Int32 COperation* GetOperationHistory(Int32 intIndex) ;
ArrayList* GetOperationHistory(Int32 intFromIndex, Int32 intToIndex);
Int32 GetMaxCount() ;
Int32 GetCount() ;

Symptom: Failed to get correct data for Subsystem L and R side if MacMan screen is different than current setting of subsystem. It always gets the data from current MacMan screen.

Solution/Work around: None

4.2.1.2 CAxis class

Function:

Double GetActualPositionProgramCoord(AxisIndex1Enum enAxisIndex);
Double GetTargetPosition (AxisIndex1Enum enAxisIndex);

Symptom: The Z-axis position for sub system NC-AL and NC-AR does not return data correctly. It is based on the current selection of spindle in NC panel or command program G140/G141.

Solution/Work around: None

4.3 Defects Fixed in this Release

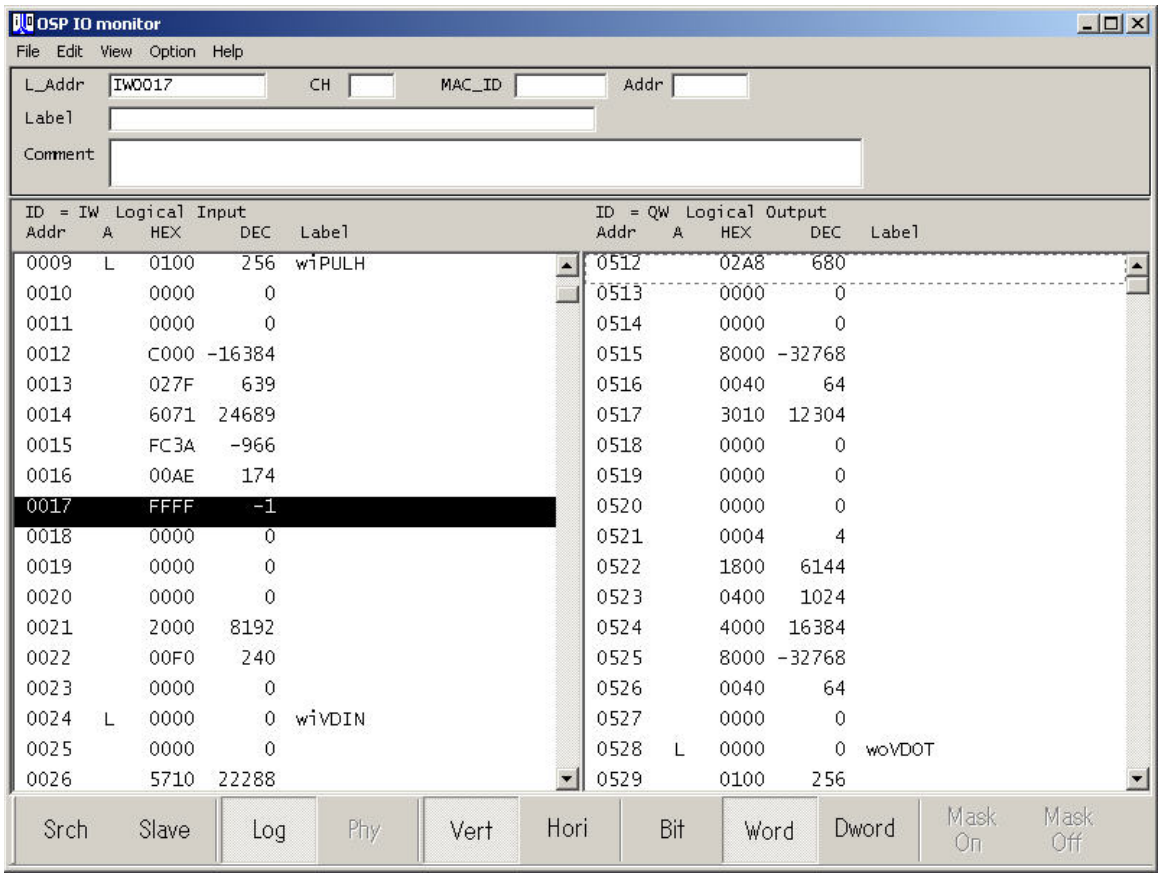
4.3.1 CIO Class

4.3.1.1 Int32 GetWordIO(IOTypeEnum enIO, Int32 intAddressIndex)

Function: Int32 GetWordIO(IOTypeEnum enIO, Int32 intAddressIndex)

Symptom: Function does not represent data correctly even the data has the same result from OSP IO Monitor application. The functions should return positive values only to correctly represent 2 to the power of N bits.
OSP IO Monitor application will return negative values when the actual data is greater than 32767.

For example: if bit 15 of word index 515 is ON then function will return 32768 meanwhile OSP IO Monitor displays -32768. Or if all bits of word index 17 are ON then function will return 65535 meanwhile OSP IO Monitor displays -1.



4.3.1.2 Int32 GetLongWordIO(IOTypeEnum enIO, Int32 intAddressIndex)

Symptom: Function does not represent data correctly even the data has the same result from OSP IO Monitor application.

The functions should return positive values only to correctly represent 2 to the power of N bits.
OSP IO Monitor application will return negative values when the actual data is greater than 2147483647.

For example: if a long word value is 80000000 in HEX then function will return 2147483648 meanwhile OSP IO Monitor displays - 2147483648.

The function has been also revised to support long data type.

4.3.2 CMachiningReport Class

Functions:

```
Int32 GetCount(ReportPeriodEnum enReportPeriod);
CMachining* GetMachiningReport(Int32 intIndex, ReportPeriodEnum enReportPeriod)
```

Symptom: The functions do not return any record after polling for a long period of time. CMachining class has been revised to support long data type for CuttingTime, OperatingTime, and RunningTime data item.

4.3.3 COperatingReport Class

Functions:

```
Int64 GetTodayOperatingReport(OperatingReportDataEnum enReportData) ;
Int64 GetPreviousOperatingReport(OperatingReportDataEnum enReportData);
Int64 GetPeriodOperatingReport(OperatingReportDataEnum enReportData);
```

The functions have been revised to support long data type.

4.3.4 COptionalParameter Class

Functions:
Overloads Public Sub New(ByVal enSubSystem As SubSystemEnum)
Symptom: If an object of this class is created using this constructor, all methods will not be accessible if calling in threads other than created thread.

4.3.5 CVariables Class

Functions:
Overloads Public Sub New(ByVal enSubSystem As SubSystemEnum)
Symptom: If an object of this class is created using this constructor, all methods will not be accessible if calling in threads other than created thread.