SDI CORPORATION

ActiveX Control "LNComm.OCX" for LN Communications

External Specifications

First edition: June 2, 2003
Second edition: September 5, 2003
Third edition: September 8, 2003
Fourth edition: October 9, 2003
Fifth edition: October 10, 2003
Sixth edition: July 8, 2005
Seventh edition: July 28 2006
Eighth edition: January 26 2007

Revisions

Revision	Date	Version	Contents
First edition	June 2, 2003		Based on the Mk30LQ.OCX specifications, seventh edition, LN power supply (EDM/EDW/MC) integration OCX (LNComm.OCX) specifications created
Second edition	September 5, 2003		To keep compatibility with Mk30Comm.OCX, event and method return values modified Explanations on the HTTP connection deleted A sequence example added Supplementary information A added Specifications of "GetMacroParm" modified (to enable handling of null data) "GetDrSodickDatabase" deleted "GetCondSchDatabase" deleted
Third edition	September 8, 2003		Errors in the method return values that are newly added for LNCOMM corrected Description added to indicate that the file being edited cannot be overwritten with "Sendfile" Description about the difference in "ExecuteComplete" between LN1X and LN2X added
Fourth edition	October 9, 2003		The default value for "Poweron" property modified Description about the use of the status property and event added Explanations on "Username" and "Password" properties added Precautions on the use of "GetMacroParm" during NC execution added Description about the file extension modified
Fifth edition	October 10, 2003		Description about the setup modified Errors corrected
Sixth edition	July 8, 2005	V2.20	Add a postscript about parameter of “SendEmKey” Modify specifications about “LogStart” Add remarks about timeout in “Execute” and “SendFile” Modify errata
Seventh edition	July 28 2006	V2.21	Add a postscript about behavior that connection aborted
Eighth edition	January 26 2007	V2.22	Change a period until abort
		V2.23	Add a method “SetMacroParm” Modify errata

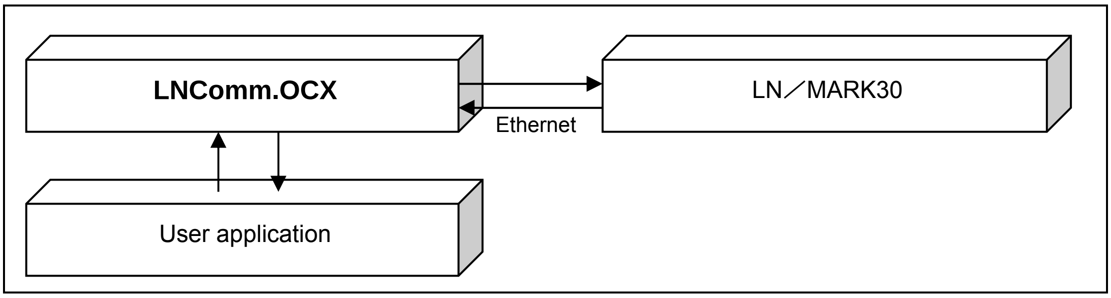
Table of Contents

1. Overview

ActiveX Control "LNComm.OCX" enables communications between Sodick LN series and WindowsNT4.0 compatibles. The LN series can be remote-controlled from a computer.

Program Name	LNComm.OCX
Applicable NC machines	Die sinking electrical discharge machine Sodick LN-EDM ("EDM") Wire-cut electrical discharge machine Sodick LN-WIRE ("EDW") Machining center Sodick Engineering LN2X ("MC2") Machining center Sodick Engineering LN1X version 7 or later ("MC1")
Applicable OS	Microsoft Windows NT4.0 (SP3 or above) Microsoft Windows 2000 Microsoft Windows XP Microsoft Windows 7 (The OCX corresponds only to the 32-bit version.)
Applicable computers	Computers operating on the above OS
Available language	Microsoft Visual Basic 5.0/6.0 Microsoft Visual C++ 5.0/6.0
Ethernet	Ethernet used as an interface between the computer and the NC machine
Protocol	Winsock 2.0
Functions	File transfer File deletion Program execution command Acquisition of machining history file Acquisition of offset term file

<Configuration>



2. File Configuration

・LNCOMM.OCX	ActiveX OCX
・LNCOMM.EXP	Export (.EXP) file Contains information related to the functions and data items to be exported.

3. Setup

3.1. Setup Procedure of LNCOMM.OCX

① Copy "LNCOMM.OCX" and "LNCOMM.EXP" into a folder.

These two files must be copied into the same folder.

② Start Command Prompt.

③ Execute "REGSVR32.EXE".

When these two files are copied into "C:\TEMP";

execute "REGSVR32 C:\TEMP\LNCOMM.OCX".

With the above procedure, OCX can be used on the client computer.

3.2. Possible Reasons for Setup Failure

There are a number of possibilities of setup failure; however, the following two points are the major reasons for most cases.

・"REGSVR32.EXE" cannot confirm the control.

・Some OLE support DLL files are missing.

If you have failed in setup, check the above points.

3.3. Version Update

When updating the version, follow the same steps as described in "3.1 Setup Procedure of LNCOMM.OCX". However, be sure to copy the files to the same folder where the previous version files are contained. Otherwise, the browse setting from the application software may not be performed correctly.

This is also the case with a downgrading to an earlier version.

If you have failed in updating, cancel the previous OCX registration and then try again.

To cancel registration of the previous OCX file that is located at "C:\TEMP", execute;

```
REGSVR32 C:\TEMP\LNCOMM.OCX /u
```

3.4. OLE Support DLL Files and Precautions

When using ActiveX (OCX) control, OLE support DLL files are required in addition to the control.

The following OLE support DLL files are required for operation of LNCOMM.OCX.

MFC42.DLL, MSVCRT.DLL, OLEPRO32.DLL

OLE support DLL files are available in multiple versions. Check to see which versions are used as well as where the files are installed. For more information, refer to the documents about MFC, such as MSDN.

4. Property, Method, Event

4.1. List of Properties, Methods, Events

Property

Name	Function
MachineStatus	Checks the LN machine state.
MyComputer	Refers to the computer name of the client computer.
Password	Sets a user password to enable access to the shared folder on the client computer.
PowerOn	Checks the LN power state.
RasPassword	Sets a user password to enable access to the shared folder on the LN.
RasUserName	Sets a user name to enable access to the shared folder on the LN.
UserName	Sets a user name to enable access to the shared folder on the client computer.
Version	Acquires the OCX version number.

Method

Name	Function
Connect	Establishes a connection with the LN.
DeleteFile	Deletes a file from the LN.
Disconnect	Cuts the connection from the LN.
Execute	Issues program execution command to the LN.
GetHistory	Acquires the latest machining history file from the LN (EDM/EDW).
GetLastError	Acquires the number of the error that has occurred most recently (for debugging).
GetOffset	Acquires the offset term file from the LN (EDM/EDW).
LogEnd	Ends communication logging (for debugging).
LogStart	Starts communication logging (for debugging).
Pause	Temporarily cuts and restarts the connection with the LN.
SendEmKey	Sends the [OFF]/[HALT]/[ACK] /[ENT]key data to the LN.
SendFile	Transfers a file to the LN.
GetToolNum	Acquires the tool number.
GetCoordSys	Acquires the current coordinate system.
GetCoordOrg	Acquires the origin of the coordinate system.
GetMachCoord	Acquires the machine coordinate value.
GetCoord	Acquires the coordinate value.
GetUpperGuideDistance	Acquires the distance between the table and the upper wire guide (EDW).
GetLowerGuideDistance	Acquires the distance between the table and the lower wire guide (EDW).
GetZAxisLimit	Acquires the Z-axis limit value (EDM/EDW).
GetInchUnit	Acquires the "Inch on/off" state (EDM/EDW/MC2).
GetDigit	Acquires the "Digit" state (EDM/EDW/MC2).
GetOffsetD	Acquires cutter compensation in a batch from the LN (MC2/MC1).
GetOffsetH	Acquires tool length compensation in a batch from the LN (MC2/MC1).
GetMacroParm	Acquires a system variable for a macro from the LN (MC2/MC1).
GetFeedrate	Acquires the machining speed from the LN (MC2/MC1).
GetSpindle	Acquires the spindle speed from the LN (MC2/MC1).
GetlineNumber	Acquires the number of the line that is currently being machined in the execution file (MC1).
SetMacroParm	Changes a system variable for a macro in the LN (MC2).

Event

Name	Function
DeleteFileComplete	Occurs on completion of file deletion from the LN.
DenyRemoteAccess	Occurs when the LN does not allow remote operation.
ExecuteComplete	Occurs on completion of program execution on the LN.
NcError	Occurs when an error or a halt arises on the LN.
NetworkError	Occurs when a network error arises.
SendFileComplete	Occurs on completion of file transfer to the LN.
VersionError	Occurs when the server version is not consistent with the client version at the time of connection with the LN.
MachineStatusChanged	Occurs when the LN machine state is changed.

* Properties, methods and events may be changed or added due to future extensions.

4.2. Property

MachineStatus

EDM/EDW/MC2/MC1

Function Checks the LN machine state.

Format *intMachineStatus* = *mLNComm*. **MachineStatus**

Setting Value Integer type

0	READY(execution possible state)
1	In executing
2	In a halt or feed hold
3	Waiting for ACK/RESET
(Default value)	-1

Usage Referencing

Explanation This property is updated at regular cycles.
If the machine state on the LN changes quickly, it may not be reflected correctly.
If it is necessary to know every change of the machine state precisely, use the **MachineStatusChange** event.

MyComputer

EDM/EDW/MC2/MC1

Function Refers to the computer name of the client computer.

Format *mLNComm*. **MyComputer** = *strMyComputer*

Setting Value String type

Client computer name	
(Default value)	Client computer name (host name)

Usage Setting and referencing

Explanation This property is used when the LN has access to a client computer.
The client computer name (host name) is automatically acquired at the startup of OCX.

Password

EDM/EDW/MC2/MC1

Function	Sets a user password to enable access to the shared folder on the client computer.
Format	*mLNComm*. **Password** = *strPasswd*
Setting Value	String type User password (Default value) " "
Usage	Setting and referencing
Explanation	This property is used when the LN has access to a client computer. It must be the password for the account that can be logged on from the LN to the client computer. For example, when transferring a file using the **SendFile** method, the LN logs in the client computer to read the file. In such a case, this password is used as the access password.

PowerOn

EDM/EDW/MC2/MC1

Function	Checks the LN power state.	
Format	*BlnPowerOn* = *mLNComm*. **PowerOn**	
Setting Value	Bool type	
	TRUE	Power ON
	FALSE	Power OFF
	(Default value)	FALSE
Usage	Referencing	
Explanation	This property is updated at regular cycles. If the power state on the LN changes quickly, it may not be reflected correctly.	

RasPassword

EDM/EDW/MC2/MC1

Function Sets a user password to enable access to the shared folder on the LN.

Format *mLNComm*. **RasPassword** = *strPasswd*

Setting Value String type
User password
(Default value) “enkaku”

Usage Setting and referencing

Explanation This property is used when having access to the shared folder on the LN.

RasUserName

EDM/EDW/MC2/MC1

Function Sets a user name to enable access to the shared folder on the LN.

Format *mLNComm*. **RasUserName** = *strUserName*

Setting Value String type
User name
(Default value) “rasperson”

Usage Setting and referencing

Explanation This property is used when having access to the shared folder on the LN.

UserName

EDM/EDW/MC2/MC1

Function Sets a user name to enable access to the shared folder on the client computer.

Format *mLNComm*.**UserName** = *strUserName*

Setting Value String type
User name
(Default value) User name to be used when the LN logs on

Usage Setting and referencing

Explanation This property is used when the LN has access to a client computer.
It must be the account that can be logged on from the LN to the client computer.
For example, when transferring a file using the **SendFile** method, the LN logs in the client computer to read the file.
In such a case, this password is used as the access account.

Version

EDM/EDW/MC2/MC1

Function Acquires the OCX version number.

Format *strVer* = *mLNComm.* **Version**

Setting Value String type
Version number

Usage Referencing

Explanation

4.3. Method

Connect

EDM/EDW/MC2/MC1

Function	Establishes a connection with the LN.	
Format	*intError* = *mLNComm*. **Connect**(*strMachineName*)	
Argument	strMachineName$	Character string of the machine name (computer name) to be connected
Return value	Integer type	
	-1	Connection successful
	0	Connection failure
Explanation	If connection has failed, the error number can be acquired by executing the **GetLastError** method.	

DeleteFile

EDM/EDW/MC2/MC1

Function	Deletes a file from the LN.	
Format	*intError* = *mLNComm*. **DeleteFile**(*strFileName*)	
Argument	strFileName$	Character string of the file name (including file extension) to be deleted
Return value	Integer type	
	-1	Deletion successful
	0	Deletion failure
Explanation	When this method has been completed, the **DeleteFileComplete** event occurs. If an invalid character string is used for the argument, etc., the return value is given for the error that has occurred before deleting the file. For the error that has occurred during file deletion, the error number is transferred as an argument in the **DeleteFileComplete** event.	

Disconnect

EDM/EDW/MC2/MC1

Function	Cuts the connection with the LN.
Format	*mLNComm*. **Disconnect**( )
Argument	None
Return value	None
Explanation	If failed, the error number can be acquired by executing the **GetLastError** method.

Execute

EDM/EDW/MC2/MC1

Function	Gives program execution command to the LN.	
Format	*intError* = *mLNComm*. **Execute**(*strCommand*)	
Argument	strCommand$	Character string for execution command ("ENKAKU.NC")
Return value	Integer type	
	-1	Execution successful
	0	Execution failure

Explanation

Before executing this method, the "ENKAKU.NC" file must be transferred to the LN.
When this method has been completed, the **ExecuteComplete** event occurs. (With MC1, however, the **ExecuteComplete** event does not occur if an error arises.)
If an invalid character string is used for the argument, etc., the return value "0" is given for the error that has occurred before issuing the program execution command. In this case, the error number can be acquired by executing the **GetLastError** method.
For the error that has occurred when the program execution command is issued, the error number is transferred as an argument in the **ExecuteComplete** event.
EDM/EDW: The contents of the "ENKAKU.NC" file describes the real file in the RAM of the LN in the Q command format. (Example) QTEST(0.000,10.000);
MC: The contents of the "ENKAKU.NC" file describe the real file in the RAM of the LN using a macro call. (Example) G65P1000 A0 B10.000;

GetHistory

EDM/EDW

Function Acquires the latest machining history file from the LN.

Format *nRet* = *mLNComm*. **GetHistory**(BSTR *pHistDat)

Argument	strHistory$	Character string of the data in the acquired machining history file (output)
Return value	Bool type	
	TRUE	Successful
	FALSE	Failure

Explanation Before executing this method, the **RasUserName** and **RasPassword** properties must be set.
File reading must not be attempted during NC execution.
The maximum size in file acquisition is not limited.
To read a file, use "FILE_SHARE_READ | FILE_SHARE_WRITE".
When acquiring a file, only the one recently used is acquired.
If failed in acquisition, the error number can be acquired by executing the **GetLastError** method.

GetLastError

EDM/EDW/MC2/MC1

Function Acquires the number of the error that has occurred most recently (for debugging).

Format *ret* = *mLNComm*. **GetLastError**

Argument None

Return value	Integer type	Error number

Explanation For more information on return values, refer to "4.6 **GetLastError** Method Error List".

LogEnd

EDM/EDW/MC2/MC1

Function	Ends communication logging (for debugging).
Format	*mLNComm*. **LogEnd**
Argument	None
Return value	None
Explanation	When **LogStart** is executed, be sure to execute this method.

LogStart

EDM/EDW/MC2/MC1

Function	Starts communication logging (for debugging).	
Format	*mLNComm*. **LogStart**	
Argument	None	
Return value	Bool type	
	TRUE	Start successful
	FALSE	Start failure
Explanation	The log file is created in the "TEMP" or "TMP" folder. The log file is named “LOG*yymmdd*.log”. (*yy*: year, *mm*: month, *dd*: date) The log file is recorded time, method name, argument and return value.	

GetOffset

EDM/EDW

Function — Acquires the offset term file from the LN.

Format — *Ret* = *mLNComm*. **GetOffset**(*strOffset*)

Argument — strOffset$ — Character string of the data in the acquired offset term file (output)

Return value — Bool type

TRUE	Acquisition successful
FALSE	Acquisition failure

Explanation — Before executing this method, the **RasUserName** and **RasPassword** properties must be set.
File reading must not be attempted during NC execution on the client computer.
The maximum size in file acquisition is not limited.
To read a file, use "FILE_SHARE_READ | FILE_SHARE_WRITE".
If failed in acquisition, the error number can be acquired by executing the **GetLastError** method.

Pause

EDM/EDW

Function — Temporarily cuts and restarts the connection with the LN.

Format — *mLNComm*. **Pause**(*blnPause*)

Argument — blnPause — Temporary cut and restart of the connection with the LN
TRUE (connection canceled)
FALSE (connection resumed)

Return value — None

Explanation — Use this method when manual intervention is required on the LN during communications between the LN and a computer.
This method ignores all the notices from the LN.
This method may not be supported in future.

SendEmKey

EDM/EDW/MC2/MC1

Function	Sends the [OFF]/[HALT]/[ACK]/[ENT] key data to the LN.	
Format	*intError* = *mLNComm*. **SendEmKey**(*strKey$*)	
Argument	strKey$	Character string of the key data to be sent to the LN EDM/EDW/MC2/MC1 "OFF" Execution stop "HALT" Temporary stop of execution "ACK" Error reset "ENT" Restart after temporary stop
Return value	Integer type	
	-1	Sending successful
	0	Sending failure

Explanation

If an invalid character string is used for the argument, etc., the return value "0" is given for the error that has occurred before issuing the program execution command.
In this case, the error number can be acquired by executing the **GetLastError** method.
Depending on the key character string that is sent, the **NcError** event may occur.

SendFile

EDM/EDW/MC2/MC1

Function	Transfers a file to the LN.	
Format	*intError* = *mLNComm*. **SendFile**(*strFileName*)	
Argument	strFileName$	Character string of the file name (including file extension) to be sent
Return value	Integer type	
	-1	Sending successful
	0	Sending failure

Explanation

Before executing this method, the **UserName** and **Password** properties must be set.
Only the files in the folder that can be shared with "PCSHARE" on the client computer can be transferred.
If the file specified for "strFileName" exists on the LN, the existing file on the LN will be overwritten. However, if the file is opened on the [Edit] screen, it is locked and cannot be overwritten. In this case, an error arises.
When this method has been completed, the **SendFileComplete** event occurs.
If an invalid character string is used for the argument, if the file to be sent does not exist, etc, the return value is given for the error that has occurred before transferring the file. In this case, the error number can be acquired by executing the **GetLastError** method.
For the error that has occurred during file transfer, the error number is transferred as an argument in the **SendFileComplete** event.

GetToolNum

EDM/MC1/MC2

Function	Acquires the tool number.	
Format	*LongNum* = *mLNComm*.***GetToolNum()***	
Argument	None	
Return value	Long integer type	The number of the currently selected tool
Explanation		

GetCoordSys

EDM/EDW/MC2/MC1

Function	Acquires the current coordinate system.	
Format	*LongCoord* = *mLNComm*. ***GetCoordSys()***	
Argument	None	
Return value	Long integer type	Current coordinate system
Explanation	For more information on the coordinate system, refer to "Relations between 'nCoordSys' and Coordinate Systems".	

GetCoordOrg

EDM/EDW/MC2/MC1

Function	Acquires the origin of the coordinate system (nCoordSys).	
Format	*LongRet = mLNComm.* ***GetCoordOrg*** *(long nCoordSys, double *pdblCoord)*	
Argument	Long integer type for input	Coordinate system: For more information on "nCoordSys", refer to "Relations between 'nCoordSys' and Coordinate Systems".
	Double precision type [8] for output	Coordinate origin: Array index: 0: X, 1: Y, 2: Z, 3: A, 4: B, 5: C, 6: U, 7: V
Return value	Long integer type	
	-1	Successful
	0	Failure
Explanation	If failed (return value "0"), the error number can be acquired by executing the **GetLastError** method.	

GetMachCoord

EDM/EDW/MC2/MC1

Function	Acquires the machine coordinate value.	
Format	*LongRet = mLNComm.* ***GetMachCoord****(double *pdblCoord)*	
Argument	Double precision type [8] for output	Array index: 0: X, 1: Y, 2: Z, 3: A, 4: B, 5: C, 6: U, 7: V
Return value	Long integer type	
	-1	Successful
	0	Failure
Explanation	If failed (return value "-1"), the error number can be acquired by executing the **GetLastError** method.	

GetCoord

EDM/EDW/MC2/MC1

Function	Acquires the coordinate value of the specified coordinate system.	
Format	*LongRet = mLNComm*. ***GetCoord****(long CoordSys, double *pdblCoord)*	
Argument	Long integer type for input CoordSys	Coordinate system:
	Double precision type [8] for output	Coordinate value: Array index: 0: X, 1: Y, 2: Z, 3: A, 4: B, 5: C, 6: U, 7: V
Return value	Long integer type -1 0	 Successful Failure
Explanation	If failed (return value "-1"), the error number can be acquired by executing the **GetLastError** method.	

GetUpperGuideDistance

EDW

Function	Acquires the distance between the table and the upper wire guide. (Varies along the X-axis travel during machining)	
Format	*mLNComm*. ***GetUpperGuideDistance****(long &pnUpperGuideDistance)*	
Argument	Output pnUpperGuideDistance	Pointer for the distance between the table and the upper wire guide Unit: "10000" is indicated for pnUpperGuideDistance when the distance is 1 mm.
Return value	None	
Explanation		

GetLowerGuideDistance

EDW

Function	Acquires the distance between the table and the lower wire guide. (Varies as "TL" in the NC code or when the setting has been changed)	
Format	*mLNComm*. ***GetLowerGuideDistance****(long &pnLowerGuideDistance)*	
Argument	Output pnLowerGuideDistance	Pointer of the distance between the table and the lower wire guide Unit: "10000" is indicated for pnLowerGuideDistance when the distance is 1 mm.
Return value	None	
Explanation		

GetZAxisLimit

EDM/EDW

Function	Acquires the Z-axis limit value.	
Format	*LongLimit = mLNComm*. ***GetZAxisLimit()***	
Argument	None	
Return value	Long integer type	Z-axis limit value Unit: "10000" is indicated for "1 mm" as the return value.
Explanation		

GetInchUnit

EDM/EDW/MC2

Function Acquires the "Inch on/off" state of the machine.

Format *bIsInch = mLNComm.* ***GetInchUnit*** *()*

Argument None

Return value Bool type

TRUE	inch
FALSE	mm

Explanation

GetDigit

EDM/EDW/MC2

Function Acquires the "Digit" state of the machine.

Format *nDigit = mLNComm.****GetDigit****()*

Argument None

Return value Long integer type

For mm: 0, 1, 2	The coordinate value is indicated to three, four or five decimal places.
For inch: 0, 1, 2	The coordinate value is indicated to four, five or six decimal places.

Explanation The number of decimal places for coordinate display is acquired as an integer from 0 to 2.

GetOffsetD

MC2/MC1

Function	Acquires cutter compensation data (1 - 99) in a batch from the LN.	
Format	*LongRet = mLNComm*. ***GetOffsetD(*** *strOffsetD_data)*	
Argument	strOffsetD_data for output	Character string of acquired cutter compensation data (1 - 99)
Return value	Long integer type	
	-1	Acquisition successful
	0	Acquisition failure
Explanation	If failed in acquisition, the error number can be acquired by executing the **GetLastError** method.	

GetOffsetH

MC2/MC1

Function	Acquires tool length compensation data (1 - 99) in a batch from the LN.	
Format	*LongRet = mLNComm*. **GetOffsetH**(*strOffsetH_data*)	
Argument	strOffsetH_data for output	Character string of acquired tool length compensation data (1 - 99)
Return value	Long integer type	
	-1	Acquisition successful
	0	Acquisition failure
Explanation	If failed in acquisition, the error number can be acquired by executing the **GetLastError** method.	

GetLineNumber

MC1

Function	Acquires the number of the line that is currently being machined in the execution file.	
Format	*LongRet = mLNComm*. ***GetLineNumber****(long LineNum)*	
Argument	Long integer type for output	Line number
Return value	Long integer type	
	-1	Successful
	0	Failure
Explanation	If failed (return value "0"), the error number can be acquired by executing the **GetLastError** method. Acquisition is possible only with MC1.	

GetFeedrate

MC2/MC1

Function	Acquires the machining speed.	
Format	*LongRet = mLNComm*. ***GetFeedrate****(long cmdFeedrate, long actFeedrate)*	
Argument	Long integer type for output	Specified machining speed
	Long integer type for output	Actual machining speed
Return value	Long integer type	
	-1	Successful
	0	Failure
Explanation	If failed (return value "0"), the error number can be acquired by executing the **GetLastError** method.	

GetSpindle

MC2/MC1

Function	Acquires the spindle speed.	
Format	*LongRet = mLNComm*. ***GetSpindle*** *(long cmdSpindle)*	
Argument	Long integer type for output	Specified spindle speed
Return value	Long integer type	
	-1	Successful
	0	Failure
Explanation	If failed (return value "0"), the error number can be acquired by executing the **GetLastError** method.	

GetMacroParm

MC2/MC1

Function	Acquires a system variable for a macro from the LN.	
Format	*LongRet = mLNComm*. **GetMacroParm**(*int_no*、*long_type*、*strMacro_data*)	
Argument	int_no for input	Macro variable number to be acquired
	Long integer type for output	Macro variable type to be acquired
	strMacro_data for output	Character string of macro variable data acquired
Return value	Long integer type	
	-1	Acquisition successful
	0	Acquisition failure
Explanation	If failed in acquisition, the error number can be acquired by executing the **GetLastError** method. Depending on the data type, strMacro_Data will be in the following format. 0: Double precision type data, 1: null data (NULL: data_type = 0 with LN1X, IsNull = 1 with LN2X)	

SetMacroParm

MC2

Function Changes a system variable for a macro in the LN

Format *LongRet* = *mLNComm*. **SetMacroParm**(*int_no*、*strMacro_data*)

Argument	int_no for input	Macro variable number to be changed
	strMacro_data for input	Character string of macro variable data changed

Return value	Long integer type	
	-1	Change successful
	0	Change failure

Explanation If failed in change, the error number can be acquired by executing the **GetLastError** method.
Depending on the data type, strMacro_Data will be in the following format. 0: Double precision type data, 1: null data
The number of the maximums which can be specified with strMacro_Data is 15.

4.4. Event

DeleteFileComplete

EDM/EDW/MC2/MC1

Function	Occurs on completion of file deletion from the LN.		
Format	*mLNComm_***DeleteFileComplete**(*intError,Cstring strError*)		
Argument	IntError	Error number	If an error arises during file deletion, the error number is passed; if no error arises, "19" is passed.
Argument	Cstring strError	Error message	If an error arises during file deletion, the error message is passed; if no error arises, the message "The file on LN machine RAM is deleted." is passed.
Explanation	This event occurs when the **DeleteFile** method has been executed.		

DenyRemoteAccess

EDM/EDW/MC2/MC1

Function	Occurs when the LN does not allow remote operation.
Format	*mLNComm* **DenyRemoteAccess**( )
Argument	None
Explanation	This event occurs when the LN does not allow remote operation on completion of execution of each method. This event occurs when remote operation is not permitted on the setting of the LN.

ExecuteComplete

EDM/EDW/MC2/MC1

Function	Occurs on completion of program execution on the LN.		
Format	*mLNComm* **ExecuteComplete**(*intError,Cstring strError*)		
Argument	IntError	Error number	If an error arises during program execution command, the error number is passed; if no error arises, "166" is passed.
Argument	Cstring strError	Error message	If an error arises during program execution command, the error message is passed; if no error arises, "Program end." is passed.
Explanation	This event occurs when the **Execute** method has been executed. This event does not occur with MC1 when an error arises. This event occurs with EDM, EDW or MC2 when an error arises.		

NetworkError

EDM/EDW/MC2/MC1

Function　Occurs when a network error arises.

Format　*MLNComm* **NetworkError**(*intError*)

Argument　IntError　Network error number

Explanation　This event occurs when the **Connect** method has been executed.

NcError

EDM/EDW/MC2/MC1

Function　Occurs when an error or a halt arises on the LN.

Format　*mLNComm* **NcError**(*strError*)

Argument　strError$　Character string of the error or halt

Explanation　This event does not occur with E00166 (program end).
This event occurs when **NCError** arises.

SendFileComplete

EDM/EDW/MC2/MC1

Function　Occurs on completion of file transfer to the LN.

Format　*mLNComm* **SendFileComplete**(*intError,Cstring strError*)

Argument	IntError	Error number	If an error arises during file transfer, the error number is passed; if no error arises, "17" is passed.
Argument	Cstring strError	Error message	If an error arises during file transfer, the error message is passed; if no error arises, the message "Writing is normally over on LN machine./" is passed.

Explanation　This event occurs when the **SendFile** method has been executed.
"17" is transferred at normal termination. "16" is transferred when the LN cannot have access to the specified file.

VersionError

EDM/EDW/MC2/MC1

Function Occurs when the LN server version is not consistent with the client version at the time of connection with the server and indicates the message box of "Version Error".

Format *mLNComm* **VersionError***(Int wServerVersionMajor, Int wServerVersionMinor, Int wClientVersionMajor, Int wClientVersionMinor)*

Argument	Int wServerVersionMajor	Server's major version	Major version of the server (MDComServer.exe on the LN)
Argument	Int wServerVersionMinor	Server's minor version	Minor version of the server (MDComServer.exe on the LN)
Argument	Int wClientVersionMajor	OCX's major version	Major version of the LNComm.ocx
Argument	Int wClientVersionMinor	OCX's minor version	Minor version of the LNComm.ocx

Explanation The LN server version and the client version are determined when a connection with the server has been established.
This event does not occur when the server and client versions are consistent.
When the server and client versions are consistent, this event occurs, and the message box of "Version Error" is displayed.

MachineStatusChanged

EDM/EDW/MC2/MC1

Function Occurs when the LN machine state is changed.

Format *LNComm.***MachineStatusChanged***(long oldValue, long newValue)*

Argument Output:
oldValue: State before change
newValue: Current state

Explanation Values for "newValue" and "oldValue":
0: READY
1: In executing
2: In a halt or feed hold
3: Waiting for ACK (RESET)
-1: Default value
This event occurs each time the machine state changes.
If a program syntax error, etc. occurs with MC1, pressing the [RESET] key triggers the error again; events of "3" → "0" and "0" → "3" occur continuously.

4.5. Relations between Methods and Events

Depending on the method that has been executed, the occurring event varies. Relations between methods and events are shown below.

Method		Error		Event
Connect	⇒	Error occurs before connection.	⇒	No event occurs. If connection has failed, the error number can be acquired by executing the **GetLastError** method.
	⇒	Error occurs at the time of connection.	⇒	The **NetworkError** event occurs.
	⇒	Connection successful	⇒	No event occurs.
DeleteFile	⇒	Error occurs before file deletion.	⇒	No event occurs.
	⇒	Error occurs at the time of file deletion.	⇒	The **DeleteFileComplete** event occurs.
	⇒	File deletion successful	⇒	The **DeleteFileComplete** event occurs.
Disconnect	⇒	Error occurs at the time of disconnection.	⇒	The **NetworkError** event occurs.
	⇒	Disconnection successful	⇒	No event occurs.
Execute	⇒	Error occurs before issuing program execution command.	⇒	No event occurs. If execution has failed, the error number can be acquired by executing the **GetLastError** method.
	⇒	Error occurs when the program execution command is issued.	⇒	The **ExecuteComplete** event occurs. (EDM/EDW/MC2)
	⇒	Program execution command successful	⇒	The **ExecuteComplete** event occurs.

GetHistory	⇒	Error occurs.	⇒	No event occurs. "False" is returned as the method return value.
	⇒	Successful.	⇒	No event occurs. "TRUE" is returned as the method return value.

GetOffset	⇒	Error occurs.	⇒	No event occurs. "False" is returned as the method return value.
	⇒	Successful.	⇒	No event occurs. "TRUE" is returned as the method return value.

Pause	⇒	Temporarily disconnected.	⇒	All events are invalid.
	⇒	Connection resumed.	⇒	All events are valid.

SendEmKey	⇒	Error occurs before sending key data.	⇒	No event occurs. If execution has failed, the error number can be acquired by executing the **GetLastError** method.
	⇒	Error occurs when key data is sent.	⇒	No event occurs. Depending on the key character string that is sent, the **NcError** event may occur.
	⇒	Sending key data successful.	⇒	No event occurs. Depending on the key character string that is sent, the **NcError** event may occur.

SendFile	⇒	Error occurs before file transfer.	⇒	No event occurs. If execution has failed, the error number can be acquired by executing the **GetLastError** method.
	⇒	Error occurs during file transfer.	⇒	The **SendFileComplete** event occurs.
	⇒	File transfer successful.	⇒	The **SendFileComplete** event occurs.

4.6. List of GetLastError Method Error

The following errors may occur when the **GetLastError** method is executed.

Return value	Explanation
0	No error is occurring.
1	Could not create the socket.
2	Could not connect to the socket.
3	Could not send the message to the LN.
4	Could not receive the message from the LN.
5	The parameter specified with the method is not correct.
6	Could not open the file.
7	Could not open the file in the shared holder on the LN.
8	Could not acquire the computer name of the client computer.
9	The user name that enables access to the shared folder on the client computer is not specified.
10	The password that enables access to the shared folder on the client computer is not specified.
11	Timeout occurs.
12	Connected with the LN already.
13	Cannot disconnect because connection with LN is not established.
14	Cannot read data because the LN is preparing data.
15	There is no response from the LN.
16	The macro variable cannot be found or the variable specification is not correct.
17	Cannot find the specified file.
18	The system variable cannot be changed. The LN's status is not READY.
19	You are going to set inaccurate data for the system variable.
20	You are forbidden change to the system variable, has been trying to remote access.

* Be careful not to confuse the return values with the following event values.

Value for the **SendFileComplete** event: 16 (file access failure), 17 (sending successful)

Value for the **DeleteFileComplete** event: 18 (deletion failure), 19 (deletion successful)

4.7. Relations between ”nCoordSys” and Coordinate Systems

nCoordSys	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Coordinate system:	54	55	56	57	58	59	154	155	156	157
nCoordSys	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
Coordinate system:	158	159	254	255	256	257	258	259	354	355
nCoordSys	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**
Coordinate system:	356	357	358	359	454	455	456	457	458	459
nCoordSys	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**
Coordinate system:	554	555	556	557	558	559	654	655	656	657
nCoordSys	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**
Coordinate system:	658	659	754	755	756	757	758	759	854	855
nCoordSys	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**
Coordinate system:	856	857	858	859	954	955	956	957	958	959

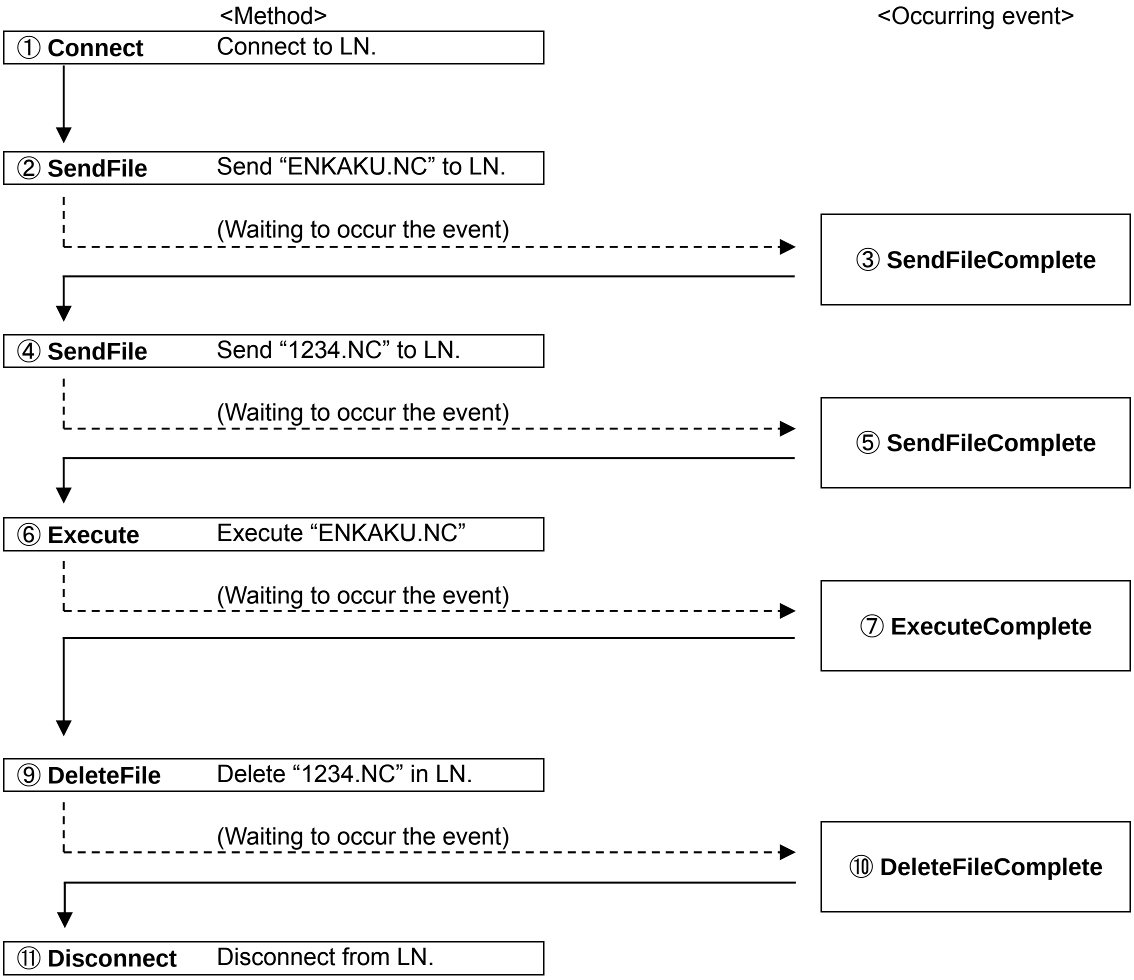
* With MC-LN1X, only coordinate systems 54 to 59 are usable.

5. Sequence Example

A sequence example using LNComm.OCX is given below.

This is an example of transferring "1234.NC" from the client computer to the LN2X, executing the file and then deleting the file.

It is assumed that the settings for connection (the OCX property setting and the setting on the LN) have been finished.



* Before step ②, "ENKAKU.NC" file must be created.

This file should describe the file (real file) to be transferred in step ④ using a macro call or in the Q Assist format.

(Example) G65P1234 A0 B10.000;

6. Remarks

- Do not execute the **GetOffset**, **GetOffsetD** or **GetOffsetH** method during NC execution.
- Do not execute the **DeleteFile** method during NC execution. If a failure occurs with this method, add a wait (2 sec or longer) immediately before processing.
- If the **GetMacroParm** method is used during NC execution, the response may be delayed and a timeout (for 4 sec) may occur. Use this method for cases except for NC execution. If it is absolutely necessary to use this method during NC execution, take appropriate action such as adding retrial processing on the calling side, etc.
- If **Execute** or **SendFile** method times out, you should be cope with it. (Timeout – **Execute**: 10 sec, **SendFile**: 20 sec) When LNComm.ocx detects timeout, retry the method.
- When communication is aborted, the system displays a dialog -”The connection was aborted due to timeout or other failure.”- for both LN and client PC. And **MachineStatus** property changes to default value. Client PC needs to run reconnection process.

7. Supplementary Information A

7.1. Setting the Shared Folder

With the LN series, the following shared folders are predetermined.

Shared name:	RAM	This is the folder to be used for file transfer and execution.
Shared name:	NCPROHD	This is the general NC program storage area. (This is not used with LNComm.ocx.)
Shared name:	HISTORY	This folder contains log files, etc.

Create the following shared folder and set the access account on the client computer.

Shared name:	PCSHARE	This is the area for file transfer.

* File transfer and execution of LNComm.ocx are performed between the PCSHARE folder on the client PC and the RAM folder on the LN (EDM, EDW or MC2) or the REMOTE folder on the LN (MC1).

7.2. Setting Properties

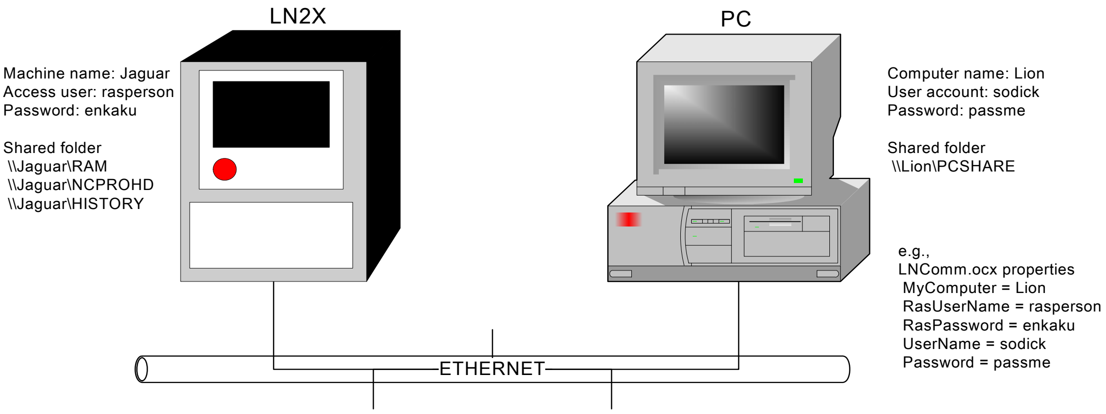
Set the above folders and the access account for the properties of LNComm.ocx.

The LN uses these property settings when having access to the client PC.

MyComputer	=	Computer name of the client PC (automatically acquired when the OCX starts up)
UserName	=	Access account to the shared folder (PCSHARE) on the client PC
Password	=	Access password to the shared folder (PCSHARE) on the client PC

The client PC uses the information when having access to the LN.

RasUserName	=	Access account to the shared folder on the LN series NC control unit
RasPassword	=	Access password to the shared folder on the LN series NC control unit


LN2X
Machine name: Jaguar
Access user: rasperson
Password: enkaku
Shared folder
\\Jaguar\RAM
\\Jaguar\NCPROHD
\\Jaguar\HISTORY
PC
Computer name: Lion
User account: sodick
Password: passme
Shared folder
\\Lion\PCSHARE
e.g.,
LNComm.ocx properties
MyComputer = Lion
RasUserName = rasperson
RasPassword = enkaku
UserName = sodick
Password = passme
ETHERNET