



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

INTRODUCTION.....	2
Variables arrays range and system variables	5
Range of the variables arrays	5
General description of functions.....	7
General functions.....	8
QV_IsProcessControllerOn	9
QV_GetSelectedTarget	10
QV_SelectTarget	11
QV_GetSelectedProcess	12
QV_SelectProcess	13
QV_GetSelectedLayout.....	14
QV_SetExtraVideoID.....	15
Functions for LocalW variables (WORD).....	16
QV_ReadLocalWord.....	17
QV_ReadLocalWordStream	18
QV_WriteLocalWord.....	19
QV_WriteLocalWordBit	20
Functions for LocalD variables (DOUBLE).....	21
QV_ReadLocalDouble.....	22
QV_ReadLocalDoubleStream	23
QV_WriteLocalDouble.....	24
Functions for LocalA variables (ASCII)	25
QV_ReadLocalASCII.....	26
QV_WriteLocalASCII	27
Functions for SyMoW variables (WORD).....	28
QV_ReadSyMoWord.....	29
QV_WriteSyMoWord	30
QV_WriteSyMoWordBit.....	31
Functions for SyMoL variables (DOUBLE)	32
QV_ReadSyMoDouble	33
QV_WriteSyMoDouble	34
Functions for SyMoA variables (ASCII)	35
QV_ReadSyMoASCII	36
QV_WriteSyMoASCII	37
Functions for SyMoRW variables (WORD).....	38
QV_ReadSyMoRetWord	39
QV_WriteSyMoRetWord	40
QV_WriteSyMoRetWordBit	41
Functions for SyMoRL variables (DOUBLE)	42
QV_ReadSyMoRetDouble	43
QV_WriteSyMoRetDouble.....	44
Error codes	45



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

INTRODUCTION

Through the library described in this document it is possible to establish communications between the screen layouts managed by the WinNBI ProcessController and an application implemented by the OEM. In other words, it is possible to view custom data, control the images, interact by means of the buttons etc. provided in the screen pages of the ProcessController. The library also has auxiliary functions, e.g., to notify to the application the screen layout selected, the NC or the selected Process.

This functions enhance further the flexibility and customisation potential of the human interface of OSAI's numerical controls.

The LayoutBuilder and ProcessController applications of WinNBI make it possible to configure and view dynamic screen layouts based on individual graphic elements (boxes).

Some generic boxes (button, image, view/edit variable, progress bar, etc.) are controlled through variables.

With the LayoutBuilder you can select different types of variable for controlling such boxes. The variables may belong to the CNC being controlled, or be generic variables referred to as "local variables".

Local variables are managed by means of the **OSAI_Exchange** library described in this document.

The OSAI_Exchange DLL allocates in the memory WORD type (16 bit), DOUBLE type (high precision floating) and ASCII (characters) local variables and makes available variables access functions both to the ProcessController and to an external application implemented by the OEM.

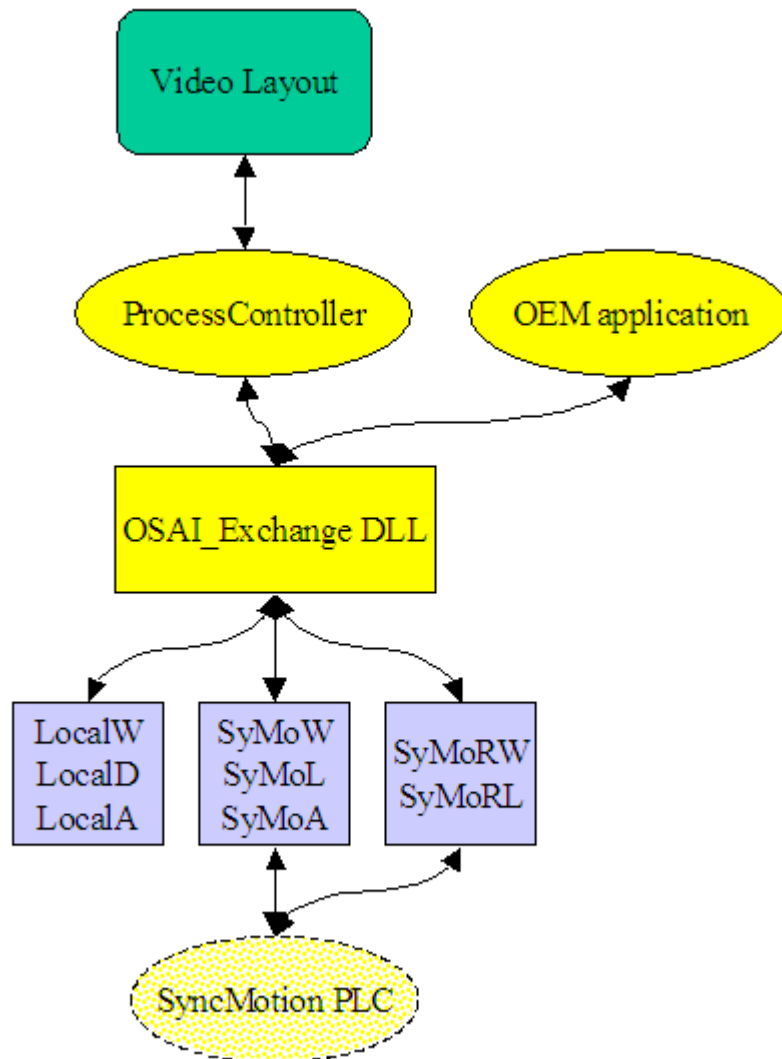
For SyncMotion product line application some variables are used to exchange data with the PLC. These variable, for CNC version ProcessController can be used as the other pure local variable..

The following table lists the variables managed by OSAI_Exchange DLL and it shows their use:

Nome	Formato	Usa per CNC	Usa per SyncMotion
LocalW	WORD	local	local
LocalD	DOUBLE	local	local
LocalA	ASCII	local	local
SyMoW	WORD	local	PLC memory
SyMoL	DOUBLE	local	PLC memory
SyMoA	ASCII	local	PLC memory
SyMoWR	WORD	local	retentive PLC memory
SyMoLR	DOUBLE	local	retentive PLC memory

The set-up described above is shown in the following page chart.

Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts



The Video Layout, through ProcessController, requires or send the values of the local variables to the OSAI_Exchange DLL. The DLL reads or writes the values in a shared memory when the variables are LocalW, LocalD or LocalA. The DLL reads or writes the values of the SyMoW, SiMoL, SyMoA, SyMoRW, SyMoRL variable either in a generic shared memory when ProcessController is used for a CNC or in the PLC shared memory when ProcessController is dedicated to SyncMotion product line machines. In the last case the SyMo variables are allocated in a volatile memory and the SyMoR variables are allocated in a retentive memory.

An OEM application interacting with OSAI_Exchange DLL can read and write the same variables managed by the Layout Video dispalyed by ProcessController.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

The following files are provided for the generation (compilation and link) of the applications that use the OSAI_Exchange library:

OSAI_Exchange.lib	Library to be "linked" to the application for checking and connecting the dynamic functions of OSAI_Exchange
OSAI_ExchangeUser.h	Include file for C language, containing the function statements and the definition of the constants needed for OSAI_Exchange
OSAI_ExchangeUser.bas	Include file for Visual Basic, containing the function statements and the definition of the constants needed for OSAI_Exchange

The OSAI_Exchange DLL is certified for applications developed in Microsoft C++ and Microsoft Visual Basic 6.0.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Variables arrays range and system variables

Some of the described variables are reserved and they have each one a specific meaning.
This paragraph describes how to use the reserved variables and the range of the variables arrays.

Range of the variables arrays

The following table shows the format of the available local variables, the range of free variables and the range of the reserved variables.

The item "Number x Number", used for ASCII variables indicates the number of strings times the number of character per string.

Nome	Formato	Numero	Libere	Riservate
LocalW	WORD	5000	0÷4899	4900÷4999
LocalD	DOUBLE	2000	0÷1899	1900÷1999
LocalA	ASCII	128x128	0÷128	-
SyMoW	WORD	8192	0÷8099	8100÷8191
SyMoL	DOUBLE	6143	0÷6142	-
SyMoA	ASCII	64 x 128	0÷63	-
SyMoWR	WORD	8192	0÷8191	-

The reserved LocalW and LocalD variables are **write protected**.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Reserved Variables (managed by the system)

The system variables used to this day are some LocalW variables (only for ProcessController connected with CNC/PowerGP) and one SyMoW variable (also for ProcessController used by SyncMotion product line).

Meaning of the reserved LocalW variables used to this day (only for the CNC/PowerGP version):

LocalW 4900 variable

value 1 = not all the NCs configured for the ProcessController are connected.

value 2 = all the NCs configured for the ProcessController are connected.

LocalW variables in the range 4901 ÷ 4908 are associated with the eight possible CNCs that the ProcessController can connect simultaneously. The meaning of each variable is as follows:

value 0 = NC not configured

value 1 = NC not connected, not selected

value 2 = NC not connected, selected

value 3 = NC connected, not selected

value 4 = NC connected, selected

These variables give the connection/selection status of the NCs configured for the ProcessController.

They can be used in a layout to graphically show the connection status of the configured targets (please note that the values of the variable can be directly used to select a bitmap inside an image box).

Meaning of the reserved SyMoW variables used to this day (for both CNC/PowerGP and SyncMotion versions):

SyMoW 8100 variable

The user (external application, SyncMotion PLC) can write in this variable the ID of the EXTRA screen it want to insert dynamically in the configured screens sequence.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

General description of functions

The paragraphs below illustrate the functions of the OSAI_Exchange DLL.

For each function, the “direction” along which the individual parameters are entered, i.e., who assigns the parameter, is specified.

The parameters can be:

- [in]** parameters compiled by the user and supplied to the function as inputs
- [out]** output parameters from the function (always pointers to a user memory area). The memory is compiled by the function during the execution.
- [in,out]** Function input/output parameters (always pointers to a user memory area). The memory is compiled by the user before calling the function, and is compiled by the function during the execution.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

General functions

This set of functions provides general information on the status of the ProcessController, the CNCs configured and connected, the screen layout being used.
It also makes it possible to use the ProcessController for the selection of specific CNCs, Processes and Screen Layouts.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_IsProcessControllerOn

Gives the status of the ProcessController application. The ProcessController may be active (running) or inactive. When the ProcessController is not running, the data contained in the exchange variables are not updated.

```
WORD QV_IsProcessControllerON (void);
```

Parameters

none

Value returned

0 if the ProcessController is running.
1 if the ProcessController is not running.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_GetSelectedTarget

Requests the data of the target (CNC) selected by the ProcessController. The ProcessController is able to connect simultaneously up to 8 CNCs. Only one of them is selected for the transmission of commands.

```
WORD QV_GetSelectedTarget (  
    LPSTR    pTargetName,  
    int      NameLen,  
    int      * pTargetID,  
    int      * pSelectedProcID  
);
```

Parameters

pTargetName

[out] Pointer to the string where the name of the selected target will be written. The string must be pre-allocated and at least 20 characters long.

NameLen

[in] Length of the buffer pointed by pTargetName. If the name of the chosen target is longer than NameLen, only the initial part of the name will be returned.

pTargetID

[out] Pointer to the variable where the identifier of the selected target (a number from 1 to 8) will be written.

pSelectedProcID

[out] Pointer to the variable where the identifier of the selected process (a number from 1 to 20) will be written.

Value returned

See paragraph on "Error codes".

See also

QV_SelectTarget, QV_GetSelectedProcess, QV_SelectProcess



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_SelectTarget

Requests the ProcessController to select a target. The ProcessController can connect simultaneously up to 8 CNCs. Only one of them is selected for the transmission of commands.

```
WORD QV_SelectTarget ( int TargetID );
```

Parameters

TargetID

[in] Identifier of the target to be selected (from 1 to 8).

Value returned

See paragraph on "Error Codes".

See also

QV_GetSelectedTarget



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_GetSelectedProcess

Requires the identifier of the process selected. The identifier depends on the target selected.

```
WORD QV_GetSelectedProcess (  
    int          * pSelectedProcID  
);
```

Parameters

pSelectedProcID

[out] Pointer to the variable where the identifier (a number from 1 to 20) for the process selected for the target selected will be written.

Value returned

See paragraph on "Error Codes".

See also

QV_GetSelectedTarget, QV_SelectTarget, QV_SelectProcess



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_SelectProcess

Requests the ProcessController to select a process for a specific target. The Process is selected even if the target specified is not the one selected, this means that when the ProcessController selects the target specified, the Process selected will be the one programmed with this function.

```
WORD QV_SelectProcess (  
    int      TargetID,  
    int      ProcessID  
);
```

Parameters

TargetID

[in] Identifier of the target specified (from 1 to 8).

ProcessID

[in] Identifier of the process to be selected (from 1 to 20).

Value returned

See paragraph on "Error Codes".

See also

QV_GetSelectedProcess, QV_SelectTarget



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_GetSelectedLayout

Requests the information on the screen layout selected by the ProcessController. The screen layout selected depends on the type of target connected and the configuration of the ProcessController.

This information is very important, since the user application must manage different variables (configured for the individual boxes) as a function of the screen layout selected by the ProcessController.

```
WORD QV_GetSelectedLayout (  
    int          * pLayoutID,  
    LPSTR        pLayoutName,  
    int          NameLen  
);
```

Parameters

pLayoutID

[out] Pointer to the variable where the identifier (number set through the LayoutBuilder) of the screen layout selected by the ProcessController will be written.

pLayoutName

[out] Pointer to the string where the filename of the screen layout selected will be written. The string must be pre-allocated and be long enough to contain a filename.

NameLen

[in] Length of the buffer pointed by pLayoutName. If the name of the screen layout selected is longer than NameLen, only the initial part of the name will be returned.

Value returned

See paragraph on "Error Codes".



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_SetExtraVideoID

Requests the selection of a screen layout that may/may not be on the list of screen layouts selected by the ProcessController. If the layout requested exists, the ProcessController selects it and activates it.

```
WORD QV_SetExtraVideoID (  
    short          ExtraVideoID,  
    short          StopVideoSelection  
);
```

Parameters

ExtraVideoID

[in] Identifier (number programmed through the LayoutBuilder) of the screen layout to be selected.

StopVideoSelection

[in] Indicates whether the ProcessController must block the F4 key and any other control for the selection of the screen layouts so as to prevent the user to change screen layout. Input values can be:

0 = does not block the selection of screen layouts

1 = blocks the selection of screen layouts

Value returned

See paragraph on "Error Codes".

See also

QV_GetExtraVideoID



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for LocalW variables (WORD)

This set of functions makes it possible to read/write LocalW WORD type local variables.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadLocalWord

Reads a WORD type LocalW variable.

```
WORD QV_ReadLocalWord (  
    long    Index,  
    short   * pValue  
);
```

Parameters

Index

[in] Index of the WORD variable to be read (from 0 to 4999).

pValue

[out] Pointer to a variable where the value of the variable will be written.

Value returned

See paragraph on "Error Codes".

See also

QV_ReadLocalWordStream



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadLocalWordStream

Reads an array of WORD type LocalW variables.

With this function it is not possible to read system variables (indices from 4900 to 4999).

```
WORD QV_ReadLocalWordStream (  
    long    Index,  
    long    NumVar,  
    short   * pValues  
);
```

Parameters

Index

[in] Index of the first WORD variable to be read (from 0 to 4899).

NumVar

[in] Number of variables to be read. If the number of variables requested, starting from Index, exceeds the size of the array, the function returns only the data available.

pValues

[out] Pointer to the WORD array where the values of the variables will be written.

Value returned

See paragraph on "Error Codes".

See also

QV_ReadLocalWord



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteLocalWord

Writes a WORD type LocalW variable.

```
WORD QV_WriteLocalWord (  
    long    Index,  
    short   Value  
);
```

Parameters

Index

[in] Index of the WORD variable to be written (from 0 to 4899).

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also

QV_WriteLocalWordBit



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteLocalWordBit

Writes a bit of a WORD type LocalW variable, leaving the other bits unaltered.

```
WORD QV_WriteLocalWordBit (  
    long      Index,  
    short     BitIndex,  
    short     Value  
);
```

Parameters

Index

[in] Index of the WORD variable in which to modify the bit (from 0 to 4899).

BitIndex

[in] Index (from 0 to 15) of the bit to be modified.

Value

[in] value (0 or 1) to be assigned to the bit.

Value returned

See paragraph on "Error Codes".

See also

QV_WriteLocalWord



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for LocalD variables (DOUBLE)

The set of functions described below makes it possible to read/write LocalD DOUBLE type local variables.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadLocalDouble

Reads a DOUBLE type LocalID variable.

```
WORD QV_ReadLocalDouble (  
    long    Index,  
    double  *pValue  
);
```

Parameters

Index

[in] Index of the DOUBLE variable to be read (from 0 to 1999).

pValue

[out] Pointer to a variable where the value of the variable will be written.

Value returned

See paragraph on "Error Codes".



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadLocalDoubleStream

Reads an array of DOUBLE type LocalID variables.

With this function system variables cannot be read (indices from 1900 to 1999).

```
WORD QV_ReadLocalDoubleStream (  
    long    Index,  
    long    NumVar,  
    double  * pValues  
);
```

Parameters

Index

[in] Index of the first DOUBLE variable to be read (from 0 to 1899).

NumVar

[in] Number of variables to read (from 1 to 1900). If the number of variables requested, starting from Index, exceeds the size of the array, the function returns only the available data.

pValues

[out] Pointer to the double array where the values of the variables will be written.

Value returned

See paragraph on "Error Codes".

See also

QV_ReadLocalDouble



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteLocalDouble

Writes a DOUBLE type LocalD local variable.

```
WORD QV_WriteLocalDouble (  
    long    Index,  
    double  Value  
);
```

Parameters

Index

[in] Index of the DOUBLE variable to be written (from 0 to 1899).

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for LocalA variables (ASCII)

The set of functions described below makes it possible to read/write LocalA ASCII type local variables. The LocalA variables are organized in an array of 128 strings. The single string, always zero terminated, can have a maximum length of 128 characters.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadLocalASCII

Reads an ASCII type LocalA variable.

```
WORD QV_ReadLocalASCII (  
    long    Index,  
    LPSTR   Text,  
    int     TextLen  
);
```

Parameters

Index

[in] Index of the string to be read (from 0 to 127).

Text

[out] Pointer to a char array where the text will be written.

TextLen

[in] Size of the "Text" char array (max 128 characters).

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteLocalASCII

Writes an ASCII type LocalA variable.

```
WORD QV_WriteLocalASCII (  
    long    Index,  
    LPSTR   Text  
    int     TextLen  
);
```

Parameters

Index

[in] Index of the string to be read (from 0 to 127).

Text

[in] Pointer to a char array containing the text to be written in the variable.

TextLen

[in] Length of the "Text" including terminator (max 128 characters). The string will be copied to the variable till either the terminator or the specified TextLen is reached.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for SyMoW variables (WORD)

The set of functions described below makes it possible to read/write SyMoW WORD type local variables. The SyMoW variables for CNC are allocated in a volatile shared memory while they are allocated in a volatile PLC shared memory for the SyncMotion machines.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadSyMoWord

Reads a WORD type SyMoW variable.

```
WORD QV_ReadSyMoWord (  
    long    Index,  
    short   * pValue  
);
```

Parameters

Index

[in] Index of the WORD variable to be read (from 0 to 8191).

pValue

[out] Pointer to a variable where the value of the variable will be written.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteSyMoWord

Writes a WORD type SyMoW variable.

```
WORD QV_WriteSyMoWord (  
    long      Index,  
    double    Value  
);
```

Parameters

Index

[in] Index of the WORD variable to be written (from 0 to 8191).

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteSyMoWordBit

Writes a bit of a WORD type SyMoW variable, leaving the other bits unaltered.

```
WORD QV_WriteSyMoWordBit (  
    long      Index,  
    short     BitIndex,  
    short     Value  
);
```

Parameters

Index

[in] Index of the WORD variable to be read (from 0 to 8191).

BitIndex

[in] Index (from 0 to 15) of the bit to be modified.

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for SyMoL variables (DOUBLE)

The set of functions described below makes it possible to read/write SyMoL DOUBLE type local variables. The SyMoL variables for CNC are allocated in a volatile shared memory while they are allocated in a volatile PLC shared memory for the SyncMotion machines.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadSyMoDouble

Reads a DOUBLE type SyMoL variable.

```
WORD QV_ReadSyMoDouble (  
    long    Index,  
    double  * pValue  
);
```

Parameters

Index

[in] Index of the DOUBLE variable to be written (from 0 to 6142).

pValue

[out] Pointer to a variable where the value of the variable will be written.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteSyMoDouble

Writes a DOUBLE type SyMoL variable.

```
WORD QV_WriteSyMoDouble (  
    long    Index,  
    double  Value  
);
```

Parameters

Index

[in] Index of the DOUBLE variable to be written (from 0 to 6142).

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for SyMoA variables (ASCII)

The set of functions described below makes it possible to read/write SyMoA ASCII type local variables. The SyMoA variables for CNC are allocated in a volatile shared memory while they are allocated in a volatile PLC shared memory for the SyncMotion machines. The SyMoA variables are organized in an array of 64 strings. The single string, always zero terminated, can have a maximum length of 128 characters.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadSyMoASCII

Reads an ASCII type SyMoA variable.

```
WORD QV_ReadSyMoASCII (  
    long    Index,  
    LPSTR   Text  
    int     TextLen  
);
```

Parameters

Index

[in] Index of the string to be read (from 0 to 63).

Text

[out] Pointer to a char array where the text will be written.

TextLen

[in] Size of the "Text" char array (max 128 characters).

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteSyMoASCII

Writes an ASCII type SyMoA variable.

```
WORD QV_WriteSyMoASCII (  
    long    Index,  
    LPSTR   Text  
    int     TextLen  
);
```

Parameters

Index

[in] Index of the string to be read (from 0 to 63).

Text

[in] Pointer to a char array containing the text to be written in the variable.

TextLen

[in] Length of the "Text" including terminator (max 128 characters). The string will be copied to the variable till either the terminator or the specified TextLen is reached.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for SyMoRW variables (WORD)

The set of functions described below makes it possible to read/write SyMoRW WORD type local variables. The SyMoRW variables for CNC are allocated in a volatile shared memory while they are allocated in a retentive PLC shared memory for the SyncMotion machines.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadSyMoRetWord

Reads a WORD type SyMoRW variable.

```
WORD QV_ReadSyMoRetWord (  
    long    Index,  
    short   * pValue  
);
```

Parameters

Index

[in] Index of the WORD variable to be read (from 0 to 8191).

pValue

[out] Pointer to a variable where the value of the variable will be written.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteSyMoRetWord

Writes a WORD type SyMoRW variable.

```
WORD QV_WriteSyMoRetWord (  
    long    Index,  
    double  Value  
);
```

Parameters

Index

[in] Index of the WORD variable to be read (from 0 to 8191).

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteSyMoRetWordBit

Writes a bit of WORD type SyMoRW variable, leaving the other bits unaltered.

```
WORD QV_WriteSyMoRetWordBit (  
    long      Index,  
    short     BitIndex,  
    short     Value  
);
```

Parameters

Index

[in] Index of the WORD variable to be read (from 0 to 8191).

BitIndex

[in] Index (from 0 to 15) of the bit to be modified.

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Functions for SyMoRL variables (DOUBLE)

The set of functions described below makes it possible to read/write SyMoRL DOUBLE type local variables. The SyMoRL variables for CNC are allocated in a volatile shared memory while they are allocated in a retentive PLC shared memory for the SyncMotion machines.



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_ReadSyMoRetDouble

Reads a DOUBLE type SyMoRL variable.

```
WORD QV_ReadSyMoRetDouble (  
    long    Index,  
    double  *pValue  
);
```

Parameters

Index

[in] Index of the DOUBLE variable to be written (from 0 to 5116).

pValue

[out] Pointer to a variable where the value of the variable will be written.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

QV_WriteSyMoRetDouble

Writes a DOUBLE type SyMoRL variable.

```
WORD QV_WriteSyMoRetDouble (  
    long    Index,  
    double  Value  
);
```

Parameters

Index

[in] Index of the DOUBLE variable to be written (from 0 to 5116).

Value

[in] value to be assigned to the variable.

Value returned

See paragraph on "Error Codes".

See also



Documentation on the OSAI_Exchange DLL: Interface with the ProcessController screen layouts

Error codes

The error codes that the library functions may return, with the relative explanations, are listed below. The notation used is that of C language and is taken from the OSAI_ExchangeUser.h user file.

#define QV_OK	0
Function executed with errors	
#define QV_ERROR_SYNCRO_ACCESS	1
System error – synchronisation error when accessing the variables	
#define QV_ERROR_PARAMETER_RANGE	2
One of the input parameters is out of range	
#define QV_ERROR_NOT_INIT	3
System error. Shared memory not allocated	
#define QV_ERROR_NOT_AVAILABLE_NOW	4
Information not available - ProcessController inactive	