



ORILINK® MONITORING SYSTEM

SCRIPT SERVICE, 23475

(Covers only OriLink® WinTools R7 and later with flash chips in all modules)
(All modules manufactured can be updated to flash chips)
(Some features may need a module software update)

Table of Contents

1. INTRODUCTION	3
2. WHY SCRIPTING ?	3
2.1. SOFTWARE MODIFICATION METHOD.	3
2.2. SCRIPTING METHOD	3
2.3. CSL-SCRIPTING	3
3. TECHNICAL BASICS	3
3.1. TO USE THE ORILINK® SCRIPT SERVICE SOFTWARE THE FOLLOWING IS NEEDED.	3
3.2. TO USE THE ORILINK® SCRIPT SERVICE SOFTWARE THE FOLLOWING IS RECOMMENDED.	4
4. INSTALLATION	4
4.1. DEFAULT FILES FOR R7 OR LATER	4
4.2. LOADING THE SCRIPT SERVICE	4
4.3. SETTING THE FILE ASSOCIATIONS	5
4.3.1. By Notepad	5
4.3.2. By Crimson editor	6
4.3.3. By Crimson Editor, x64 Windows OS	6
5. SCRIPT CONVENTIONS	6
5.1. GENERAL SCRIPT CONVENTIONS	6
5.2. SCRIPT NAMING FOR ORILINK®	6
5.2.1. Scripts directly started from a Keypad	6
5.2.2. Scripts indirectly started from a Keypad	6
5.2.3. Fixed Scripts started from the PC	6
5.2.4. Parametric Script start in Run-Time.	7
5.2.5. Automatic scripts	7
5.2.6. Time scheduled scripts	7
5.3. OCP-2.xx API PARAMETERS	8
5.3.1. Caller parameters	8
5.3.2. Transaction forwarded parameters	8
5.4. ORILINK® SPECIFIC COMMANDS (SCRIPT.DLL)	9
5.4.1. System commands	9
5.4.2. Dispense point control	10
5.4.3. KP100 Commands	14
5.4.4. PM100, 101, 200 and 201 commands	16
5.4.5. General Commands	17
5.5. OLDER SCRIPT CALLS SYNTAX LEFT FOR BACKWARD COMPATIBILITY.	18
5.5.1. KPSetGlobalResolverMessage (Seconds,'LCD Top Row ', 'LCD Bottom ROW ');	18
5.5.2. KPString(Seconds,'LCD Top Row ', 'LCD Bottom ROW ')	18
5.5.3. KPMessage(Seconds,'LCD Top Row ', 'LCD Bottom ROW ')	18
5.5.4. KPMessageQuit(Seconds,'LCD Top Row ', 'LCD Bottom ROW ')	18
5.5.5. KPFollowKey(''+Some parameter+' '?','LCD Bottom row')	18
6. WRITING SCRIPTS	19
6.1. MY FIRST SCRIPT	19
6.2. HOW DOES IT WORK?	20
6.3. IF I HAVE MADE SOMETHING WRONG IN MY SCRIPT ?	20
7. UPDATING THE PC COMPUTER	20
7.1. PATCH THE OS TO THE LATEST VERSION	20
7.2. INSTALL / UPDATE MDAC	21
7.3. INSTALL / UPDATE MSJET	21
7.4. SHOW HIDDEN FILES AND EXTENSIONS	21
8. GNU LICENSE	23

1. Introduction

The OriLink® Script Service is used when there is a demand for custom functionality. It utilises an extremely powerful tool for customisation.

Examples of this is,

- Opening 2 or more dispense points with on command.
- Mixing of two or more fluids according to Keypad input or from database.
- Additional questions such as mileage, registration number, running hours,
- Validating against a custom database.
- Fetching information from a custom database.
- Alternative input order.
- Keypad inputs that changes depending on answers.
- Remote starting of dispensing, in a network by a non OriLink® application

The software is modularised like the OriLink® hardware. This enables the possibility to only use functions needed and by this have software that has lowest possible complexity.

2. Why scripting ?

There are two main roads to obtaining custom functionality, modify the software or using scripting.

2.1. Software modification method.

It is very easy to use when the customisation is done but it has to be done by a skilled programmer using development software. It could introduce errors for all users when a custom function is done for one user. It is inflexible, very expensive and needs a lot of administration.

2.2. Scripting method

Some basic programming knowledge is needed but it can be customized without a skilled programmer and is done using a standard text editor such as Microsoft Notepad. The modification does not influence the standard software at all. The flexibility is infinite and it is inexpensive.

2.3. CSL-scripting

We have chosen to use a script language called CSL (C Script Language). The base for this is a script interpreter and a set of software libraries. This software is free software, it can be redistribute and/or modify under the terms of the GNU General Public License version 2 as published by the Free Software Foundation. For a complete GNU License see chapter "GNU License"

3. Technical basics

To be able to use the OriLink® Script service some basic things must or is recommended to be present.

3.1. To use the OriLink® Script Service software the following is needed.

An OriLink® system.

A PC computer with properly installed and configured Microsoft Windows 95/ 98SE/ NT4/ 2000/ XP operating system. It is recommended that the OS should be patched to the latest level.

The PC should have a performance suitable for the used Operating system.

For the OriLink® WinTools the minimum demands are Intel Pentium 2-333Mhz compatible, Windows98SE (Second edition) and 128Mb RAM. The recommend demand is a standard PC of today.

General rule –The more things that are running in a PC the higher performance is needed.

MDAC 2.5 or later should be installed, (Microsoft Data Access Component).

MSJET should be installed, (Microsoft database drivers).

The PC must have one free 16550 compatible serial port. If it is a laptop without a serial port use a PC-card to serial port adapter. **Do not use an USB to serial port adapter, it may not work at all or give random erratic behaviour!**

An OriLink® PC-interface (SIO part number 23 403)

A null-modem serial cable (part number 203 02 80), included in part number 23 403.

OriLink® WinTools Professional version validated for script service.

3.2. To use the OriLink® Script Service software the following is recommended.

OriLink® WinTools R7 or later

All hardware modules originally equipped with or upgraded to Flash chip technology.

4. Installation

The OriLink® Script service is installed at the OriLink® WinTools installation. It comes with a default script set (Preset) that uses the Pre-sets managed by the OriLink® WinDB Manager. To use the Script Service a license for OriLink® pro + Script Service must be present in the C:\Orilink folder.

4.1. Default files for R7 or later

During the OriLink® WinTools installation the following files are installed.

C:\Orilink\	Contains
<i>Script.dll</i>	<i>Script Service, load in engine properties.</i>
<i>Script.ini</i>	<i>Script Service configuration file.</i>
<i>OcpRun.exe</i>	<i>Command program to run direct activated parametric scripts.</i>
<i>Preset.ini</i>	<i>Preset script set configuration file.</i>

C:\Orilink\CSL\ Contains the install default Pre-set filling script set.

<i>!Base.csl</i>	<i>Basic script must be present.</i>
<i>_EUP.csl</i>	<i>Initialises the script service, must be present.</i>
<i>_EKP.csl</i>	<i>Resolves KP input to a script, must be present.</i>
<i>_EDOWN.csl</i>	<i>Runs when script service is unloaded</i>
<i>!Preset.csl</i>	<i>Basic preset functions called from !PresetKP.csl</i>
<i>!PresetKP.csl</i>	<i>Pre-Set keypad conversation.</i>
<i>!UpdateTankKP.csl</i>	<i>Makes it possible to update tank stock from a keypad</i>
<i>RemoteOneReel.csl</i>	<i>Demo script for OcpRun remote script calls</i>
<i>CSL.exe</i>	<i>Command program to run direct activated fixed scripts.</i>

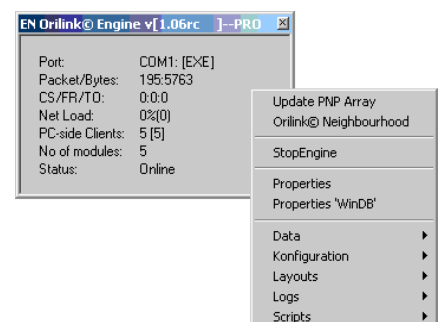
4.2. Loading the Script service

When the files are properly copied to the hard drive as above and there is a License.dat file valid for the Script service start the engine and load script.dll.

Open the OriLink® Engine window. Right-click some where in the engine window or on the icon in the Systray to open the engine control menu.

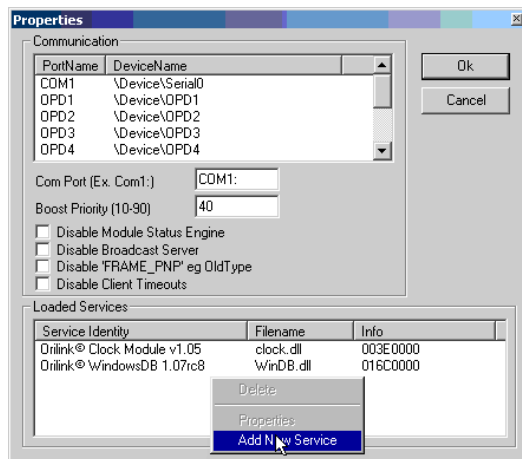
The result will be like this.

Select properties.

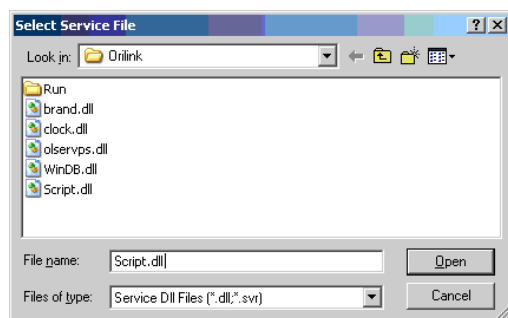


In the lower part of the properties window there is a frame named "Loaded Services". This frame contains a table of loaded services. In this window services can be added or removed.

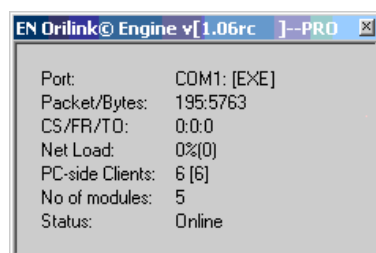
Right-click the white window.



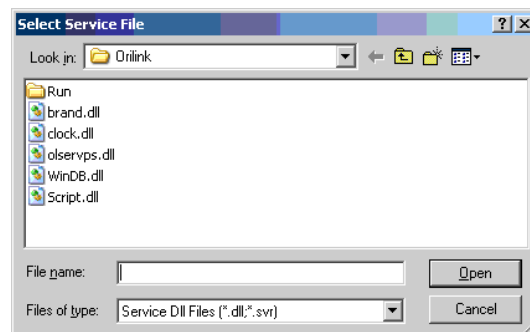
Efgfwrgwgrwer.



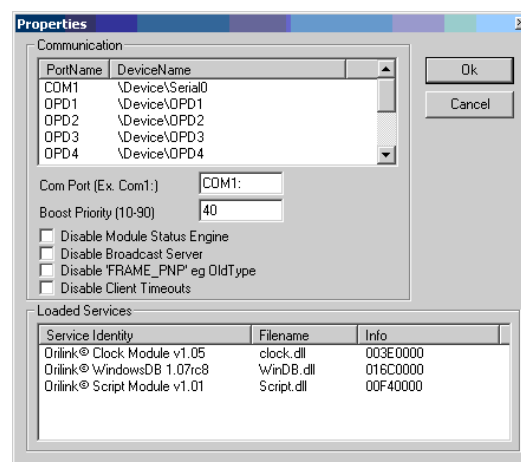
The result will be this.



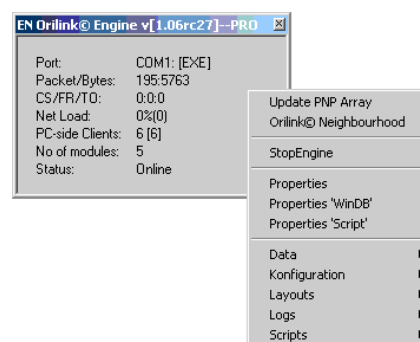
Select service dll file and left-click Open-button.



The result will be this, left-click OK button.



Now the Engine control menu will look something like this.



4.3. Setting the file associations

If you want to be able to edit and or develop script sets there are two possible ways.

4.3.1. By Notepad

Double click on a script file (****.csl) in the C:\OriLink\CSL folder. The first time you do this a window pops up asking you to specify which program to use for opening this type of file. Select Notepad and if you want that to be permanent leave the check box “ ” checked.

4.3.2. By Crimson editor

On the OriLink® CD there is a folder “CD:\EXTRAS\Scripting” and in that folder there is a subfolder \Crimson Editor\ that contains a OriLink® Script Service adapted advanced script editor “Crimson Editor”.

To use it copy the whole folder to some place on your hard drive. You can put it anywhere but we recommend putting it in either C:\OriLink or better C:\Tools. When you have done that open the Crimson editor folder and run the two Windows registry files (.reg) named according to where you put the \Crimson Editor\ folder.

For example:

You have added it as

C:\Tools\Crimson Editor

Then you run

“C:\Tools\Crimson Editor\Register Crimson in Tools.reg”

“C:\Tools\Crimson Editor\Register CSL extention Tools.reg”

If you have chosen to put Crimson Editor in some folder without .reg files for it you have to create that and run them.

4.3.3. By Crimson Editor, x64 Windows OS

On some Windows OS the version of the copy, paste and register method does not work. Then a later version must be installed. You can do that by running the “cedt-286-setup.exe” in the folder CD:\EXTRAS\Scripting\Later Crimson\. This will install the standard Crimson Editor version SVN286.

To adapt it to OriLink® ,

You must create .reg files according to where you installed it and run them.

You must copy the csl.key and csl.spc from \Crimson Editor\spec to the same place where you installed Crimson Editor.

5. Script conventions

5.1. General Script conventions

Read the CSL manual from the CD: \Scripting\Crimson Editor\Manual, if you have installed Crimson Editor C:\Tools\Crimson Editor\Manual and/or search the web.

5.2. Script naming for OriLink®

The naming of a script file has to follow some basic rules.

5.2.1. Scripts directly started from a Keypad

A script that should be started from a Keypad must begin with a “_” (underscore).

The next character has to be either “S” for start or “K” for stop (Kill).

After the two first characters you can put any accepted windows character(s) followed by .csl. These characters will be the ones you should type in on the Keypad to start the script.

Example:

There are two script files _SS80.csl and _KS80.csl.

If you type S80 followed by ENTER key on a keypad the script _SS80.csl will start.

If you type S80 followed by STOP key on a keypad the script _KS80.csl will start.

NOTE ! DO NOT NAME A SCRIPT SAME AS AN EXISTING DISPENSE POINT IN THE SYSTEM!

5.2.2. Scripts indirectly started from a Keypad

The Script Service will always check on incoming events and event script names starts with _E so by creating a _EKP.csl script it will be started on an incoming event from a keypad.

5.2.3. Fixed Scripts started from the PC

A script that should be started by double-clicking on the script, short-cut to the script or from some software can have a name with any accepted windows character(s) followed by .csl

5.2.4. Parametric Script start in Run-Time.

This can be used to start a specific script with input parameters as switches. It is done by running the OcpRun.exe file with the needed switches (parameters). The parameter

The syntax is

```
ocprun -mLOCALHOST -eScriptName.csl "const _ResolveName = '1234ABCD';"  
"const _ReelNo = '1';" "const _RVolume = '5.85';"
```

Switch	Stands for	Scope
-m	(M)achine	Should be set to the name of the PC running the OriLink® system. (The OriLink® server)
-e	(E)vent	Should be set to the name of the script that should be run.
Const_Parameter=	Input parameter declaration	Parameter stands for the name of parameter (Switch).
'XXXXX.....'	Value of parameter	XXXXX..... sets the value of the parameter (Switch).

The sample run-time instruction above runs the script “ScriptName.csl” locally on the OriLink® server. The script opens Reel number 1 with the Job number “1234ABCD” for 5.85 liters.

5.2.5. Automatic scripts

Script that should start automatically when the reset button of a module is pressed more than 5 seconds should begin with _A.

Example:

_AKP100.csl default programs a KP100 module if the reset button is pressed more than 5 seconds.

5.2.6. Time scheduled scripts

To make an OriLink® system run time scheduled tasks is very simple. Creating scripts that begin with _T placed in the C:\OriLink\CSL folder do this.

Example:

You want to dispense 1.00 L of oil every day at 12:00.

Create a new text file with the name “_T1200.csl” in the C:\Orilink\CSL folder. Edit it like this,

```
#loadScript '!Base.csl'  
main()  
{  
    sysLog('_T1200.csl');  
    sysLog('Every day at 12:00');  
    ocpOpenReel(1005,1,2222,'CL12',1.00);  
}
```

Make sure that the Script Service (Script.dll) is properly loaded. Restart the Engine.

This will open Port 1 of MPDM with address 1005 with the PIN code 2222 and Job number CL12 for 1.00 L every day at 12:00.

5.3. OCP-2.xx API parameters

In the OCP-2.xx context there are some predefined dynamic parameters. All such parameter starts with an underscore “_”.

5.3.1. Caller parameters

Caller	Parameter	Description
Keypad	_ResolveName	The keypad input
	_ResolveType	S if keypad <ENTER> has been pressed, K if keypad <STOP> has been pressed.
	_ResolveRemoteAddress	The address of the calling keypad.
	_ResolveRemoteSphere	The sphere of the calling keypad.
OcpRun	_ResolveName	The first and identifying parameter of a dynamic external call.

5.3.2. Transaction forwarded parameters

Service	Parameter	Source	Sample	Description
WinDB	_TransNo	OriLink® database	‘123’	The transaction number
	_JobNo	Dispense point	‘EH3456.01’	The used object identifier
	_WorkOrder	Dispense point	‘EH3456’	Extracted from _JobNo based on set splitting in WinDb
	_LineNo	Dispense point	‘01’	Extracted from _JobNo based on set splitting in WinDb
	_UserId	OriLink® database	‘15678’	Employee number extracted by the used PIN
	_UserName	OriLink® database	‘Some Name’	User name extracted by the used PIN
	_ReelNo	Dispense point	‘3’	Number of the reel / dispense point that created the record.
	_SphereNo	Dispense point	‘0’	Sphere of the reel / dispense point that created the record.
	_FluidNo	Dispense point	‘4’	The Tank/Fluid number associated with the reel / dispense point that created the record.
	_FuidName	OriLink® database	‘Engine oil’	Fluid name extracted by the used Fluid/Tank number of the reel / dispense point that created the record.
	_FluidPartNo	OriLink® database	‘EO234578’	Part number extracted by the used Fluid/Tank number of the reel / dispense point that created the record.
	_DVolume	Dispense point	‘2.76’	Dispensed volume
	_DVolumeTotal	OriLink® database	‘15.65’	Sum of all dispenses of the same part number on the current object.
	_RVolume	Dispense point	‘3.00’	Requested volume
	_Y	OS Date and time	‘2012’	Year 4 figures
	_y	OS Date and time	‘12’	Year 2 figures
	_m	OS Date and time	‘11’	Month
	_d	OS Date and time	‘15’	Day
	_H	OS Date and time	‘14’	Hours
	_M	OS Date and time	‘27’	Minutes
	_S	OS Date and time	‘45’	Seconds
	_StopCode	Dispense point	‘1’	Stop code

5.4. OriLink® specific commands (Script.dll)

There are some specific OriLink® commands that is explained here.

5.4.1. System commands

5.4.1.1. PNPBuildArray()

This function returns the number of found items in the PNP Array.

5.4.1.2. PNPGetAddress(n)

This function returns the address of the n'th module in the PNP Array.

5.4.1.3. PNPGetType(n)

This function returns the type of the n'th module in the PNP Array.

5.4.1.4. GetIni('Header','Parameter',IniFilename)

This function retrieves the value of a parameter setting of a configuration file (.ini)

If the SomeName.ini file contains.

```
[Header]
Parameter=SomeValue
```

GetIni(SomeHeader, SomeParameter, SomeName.ini) will return the value "SomeValue".

5.4.1.5. ocpLog('Flag','String to print in a log file')

This can be used to print selectable by a flag strategic log rows in a log file.

5.4.1.6. ocpPrint('String to print in window')

This can be used to print strategic log rows in the pop-up real-time log window of a script.

5.4.1.7. ocpAnd(A,B)

Returns the result of a logical A AND B operation

5.4.1.8. ocpOr(A,B)

Returns the result of a logical A OR B operation

5.4.1.9. ocpWildCard(A,B)

Returns the result TRUE if A matches the wildcard B otherwise FALSE

5.4.1.10. ocpWinPrintOut('PrinterQueue','String to print')

This can be used to print customized receipts, reports, etc on any available printer on the PC or a network the PC is connected to.

5.4.1.11. ocpPcVerification('Title', 1st Prompt, 1st Parameter,, 5th Prompt, 5th Parameter)

Will send up a window on the PC screen that calls for attention by beeping. The beep is defined by the sound setting for Windows event Question.

Here is an example of the window and what it can be used for.

```
ocpPcVerification(
    'Dispense request',
    'Requester','Guest Starring',
    'Workbay','3002',
    'Work order','A1234',
    'Part number','EO1234',
    'Requested volume','2.00')
```

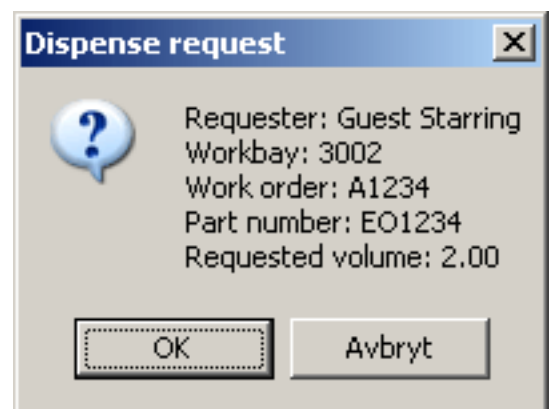
It returns

1 for <OK>

0 for <CANCEL>

5.4.1.12.

Returns the value of a specific fcode parameter



5.4.2. Dispense point control

There are several different ways to open and close a dispense point. There is one overhead difference between them. The ones starting with mon... are the latest ones and should be used all others are there only for backwards compatibility.

5.4.2.1. **monOpenReel('AAAA','N','Pin','JobNo','Vol')**

This opens a dispense point based on module address (AAAA) and port (N) and checks validation according to the setting of the dispense point.

5.4.2.2. **monOpenReel1('AAAA','N','Pin','JobNo','Vol')**

This opens a dispense point based on module address (AAAA) and port (N) regardless of the configuration of the dispense point.

5.4.2.3. **monOpenReel2(ReelNo,UserID,JobNo,Vol[,PreOpen]);**

This opens a dispense point based on ReelNo.

The optional parameter *[,PreOpen]* puts the dispense point in PreOpen mode, that is it allocates the dispense point, sets the requested volume and shows it on the LED and then waits for an OK to open the valve. The OK is normally a signal on the Input B on the allocated port.

This command has status replay where

- 1 No contact with system, communication error, no reel, or similar.
- 0 OK
- 1 System closed, key in offline position.
- 2 Reel already open.
- 3 Reel in Pre-open mode.
- 31 Requested volume is lower than Min volume.
- 32 Requested volume is lower than Max volume.

5.4.2.4. **monCloseReel('AAAA','N');**

This closes a dispense based on module address (AAAA) and module port number (N)

5.4.2.5. **monCloseReel2('ReelNo');**

This closes a dispense based on the reel number (ReelNo).

5.4.2.6. **monReelSvStatus('AAAA','N');**

Checks the status of the solenoid valve of a dispense point based on the module address (AAAA) and the port number (N).

5.4.2.7. **monReelSvStatus2('ReelNo');**

5.4.2.8. monSetReg(AAAA, Number, Value)

The instruction is used to set the status for I/O-pins on a module PCB.

AAAA = The address of the module

Value = 1 eller 0

Module	Location	I/O-pin	Number	Value	Result (Bold is default)
MPDM, TCM, TSM	Port 1	Sol	31768	1	ON
				0	OFF
	Port 2	Sol	31769	1	ON
				0	OFF
	Port 3	Sol	31770	1	ON
				0	OFF
	Port 4	Sol	31771	1	ON
				0	OFF
MPDM	Optional connector	RC0	31760	1	HIGH
				0	LOW
	Optional connector	RC1	31761	1	HIGH
				0	LOW
	Optional connector	RC2	31762	1	HIGH
				0	LOW
	Optional connector	RE0	31776	1	HIGH
				0	LOW
MPDM	Optional connector pin mode	TRISC0	31904	0	FLOAT
				1	OUT
	Optional connector pin mode	TRISC1	31905	0	FLOAT
				1	OUT
	Optional connector pin mode	TRISC2	31906	0	FLOAT
				1	OUT
	Optional connector pin mode	TRISE0	31920	0	FLOAT
				1	OUT
	Optional connector pin mode	TRISE1	31921	0	FLOAT
				1	OUT
	Optional connector pin mode	TRISE2	31922	0	FLOAT
				1	OUT

Example:

```

monSetReg(1001,31904,1);   Sets optional connector pin RC0 mode to OUT
monSetReg(1001,31760,0);   Sets optional connector pin RC0 to LOW

monSetReg(1001,31760,1);   Sets optional connector pin RC0 to HIGH
monSetReg(1001,31904,0);   Sets optional connector pin RC0 mode back to FLOAT

```

5.4.2.9. ocpOpenReel('AAAA','N','PPP','JJJJ','VVVV')

This command is used to open a dispense point regardless of the configuration of the dispense point.

5.4.2.10. ocpOpenReel(ReelNo,SphereNo,UserID,JobNo,Vol[,ocpPreOpenReel]);

The optional parameter *[,ocpPreOpenReel]* puts the dispense point in PreOpen mode, that is it allocates the dispense point, sets the requested volume and shows it on the LED and then waits for an OK to open the valve. The OK can be a signal on the Input B on the allocated port or running the command again with the optional parameter *[,ocpTrigg]* instead of *[,ocpPreOpenReel]*.

This command has status replay where

- 1 No contact with system, communication error, no reel, or similar.
- 0 OK
- 1 System closed, key in offline position.
- 2 Reel already open.
- 3 Reel in Pre-open mode.
- 31 Requested volume is lower than Min volume.
- 32 Requested volume is lower than Max volume

5.4.2.11. ocpOpenReel2(ReelNo,UserID,JobNo,Vol[,ocpPreOpenReel]);

The optional parameter *[,ocpPreOpenReel]* puts the dispense point in PreOpen mode, that is it allocates the dispense point, sets the requested volume and shows it on the LED and then waits for an OK to open the valve. The OK is normally a signal on the Input B on the allocated port.

This command has status replay where

- 1 No contact with system, communication error, no reel, or similar.
- 0 OK
- 1 System closed, key in offline position.
- 2 Reel already open.
- 3 Reel in Pre-open mode.
- 31 Requested volume is lower than Min volume.
- 32 Requested volume is lower than Max volume.

5.4.2.12. ocpCloseReel('AAAA','N');

This closes a dispense based on module address (AAAA) and module port number (N)

5.4.2.13. ocpCloseReel(ReelNo,SphereNo)

This closes a dispense based on the reel and sphere numbers.

5.4.2.14. ocpCloseReel2(ReelNo);

Where ReelNo is the reel number of the dispense point to close.

5.4.2.15. OpenReel('AAAA','N','PPP','JJJJ','VVVV');

This command is used to open a dispense point without any validation.

Where AAAA is the address, N (1-4) is the port number, PPPP is the Pin-code, JJJJ is the Job-number and VVVV is the requested volume. The ' ' can be omitted if there are only figures.

Set the dispense point to Demand volume only.

5.4.2.16. StopReel(1XXX,N);

Where 1XXX is the address and N (1-4) is the port number

5.4.2.17. ReelMV(1XXX,N);

Where 1XXX is the address and N (1-4) is the port number

5.4.2.18. ocpGetReelAddress(ReelNo,SphereNo,Address,Port);

Fetches the MPDM address and Reel port number

5.4.2.19. ocpGetReelStatus(ReelNo,SphereNo);

This command is used to query the status of a reel

Return values are

ocpStatusClosed if the reel is closed

ocpStatusPreOpen if the reel is in PreOpen mode

ocpStatusOpen if the reel is in use

5.4.3. KP100 Commands

This section describes commands only valid for keypad modules.

5.4.3.1. ocpKpSetDefaultMessage('LCD Top row ','LCD Bottom row ',Seconds)

This sets a local default message to be shown on keypad LCD when nothing else is shown.

“LCD Top row” is max 15 characters, “LCD Bottom row” is max 16 characters and “Seconds” is the show time in seconds.

OBSERVE ! do not set Seconds = 0

5.4.3.2. ocpKpMessage('LCD Top row ','LCD Bottom row ',Seconds)

Use this if you want to show a message on keypad LCD

LCD Top ROW is max 15 characters, LCD Bottom ROW is max 16 characters and TT is the show time in seconds.

OBSERVE ! do not set Seconds = 0

5.4.3.3. ocpKPString('LCD Top row ','LCD Bottom row ','')

Use this to show a question on the keypad followed by a *ocpKpGetValue()* statement to read the answer.

5.4.3.4. ocpKpPassword('LCD Top row ','LCD Bottom row ','')

Use this to show a question on the keypad and hide the typed characters with “*”. Use *ocpKpGetValue()* statement to read the answer.

5.4.3.5. ocpKpGetValue();

This is used to fetch the answer from a *ocpKPString()* statement.

To fetch and store it in the parameter KeypadInput do like this.

```
const KeypadInput = ocpKpGetValue();
```

5.4.3.6. ocpKpFollowKey("+Some parameter+'?', 'LCD Bottom row')

Is used for scroll and select from a list.

Example;

```
var Items[10];
var ItemsCount=0
var ItemsIndex;
while(1)
{
    switch(KPFollowKey("+Items[ItemsIndex]+'?', 'Up Down ENT. EXIT'))
    {
        case ocpKpUp:
            ItemsIndex--;
            if(ItemsIndex== -1)
                ItemsIndex=ItemsCount-1;
            break;

        case ocpKpDown:
            ItemsIndex++;
            if(ItemsIndex==ItemsCount)
                ItemsIndex=0;
            break;

        case ocpKpExit:
            exit();
            break;

        case ocpKpEnter:
            {
                return Items[ItemsIndex];
            }
            break;
    }
}
```

5.4.3.6.1. ocpKpUp

Return value for the arrow up key (Blue) on keypad.

5.4.3.6.2. ocpKpDown

Return value for the arrow down key (Yellow) on keypad.

5.4.3.6.3. ocpKpExit

Return value for the EXIT key on keypad.

5.4.3.6.4. ocpKpEnter

Return value for the ENTER key on keypad.

5.4.3.7. ocpKpMessageQuit('LCD Top row ','LCD Bottom row ',Seconds)

This must be used when leaving a keypad session and shows a quit message on the keypad LCD

“LCD Top row” is max 15 characters, “LCD Bottom row” is max 16 characters and “Seconds” is the show time in seconds.

OBSERVE !

Must be used when leaving a keypad session.

Do not set Seconds = 0

5.4.3.8. FastMenu('3XXX',N,ADDRESSFCODE,'NAME');

This can be used to add, edit or delete items in the Fast Menu of a keypad.

Where

3XXX is the keypad address

ADDRESS is address of target

FCODE is the proper fast code for the target (se manual for module)

NAME is the FastMenu text.

5.4.3.9. ocpLaunchScriptKP(SomeScriptName.csl)**5.4.3.10.**

5.4.4. PM100, 101, 200 and 201 commands

This section describes commands only valid for printer modules.

5.4.4.1. SetTankData(2XXX,N,Capacity,Current,Reorder,Stop,OilName);

This can be used to set up a tank in a printer module.

Where 2XXX is the address, Capacity is tank volume, Current is the current stock, Reorder is the reorder volume, Stop is the stop volume and Oil Name is the name or part number of the oil in the tank.

5.4.4.2. Print(2XXX,'Some String \n');

This can be used to send a print out to a printer module.

Where

2XXX is the address

'Some String' can be any type of string such as text, variables, parameters, etc.

\n' ends the string and sends a LineFeed (Carriage Return) to the printer.

5.4.5. General Commands

This section describes commands valid for several modules.

5.4.5.1. Language(AAAA,TYPE,LANGUAGE);

This can be used to change language of a module.

Were AAAA is the address, TYPE is the module type (LED100, KP100,) and LANGUAGE is a valid language name in the language file lang.lan.

5.4.5.2. LanguageProgramAllByPNP(Language);

Use this if you want to change the language of all modules found by PNP

Were Language is a valid language name in the language file lang.lan.

5.4.5.3. GetPPU(XXXX,FCODE);

This can be used to get the current PPU from modules such as MPDM and LED

Were XXXX is the address, FCODE is the proper fast menu code (se manual for module) and the function returns the current PPU value.

5.4.5.4. SetPPU(XXXX,FCODE,PPU);

This can be used to set the PPU in modules such as MPDM and LED.

Were XXXX is the address, FCODE is the proper fast menu code (se manual for module) and PPU is the value. If it supports Integer (328) or Float (328.00) depends on the software in the module.

5.4.5.5. GetInt(XXXX,FCODE);

This can be used to get integer parameters from modules such as Mask, Rights

Were XXXX is the address, FCODE is the proper fast code (se manual for module) and the function returns the value requested.

5.4.5.6. SetInt(XXXX,FCODE,INT);

This can be used to set integer parameters in modules such as Mask, Rights,

Were XXXX is the address, FCODE is the proper fast code (se manual for module) and INT is the value.

5.4.5.7. GetString(XXXX,FCODE);

This can be used to get current TEXT parameters from modules such as Names, part numbers, job numbers,.....

Were XXXX is the address, FCODE is the proper fast menu code (se manual for module) and the function returns the value requested.

5.4.5.8. SetString(XXXX,FCODE,STRING);

This can be used to set TEXT parameters in modules such as Names, part numbers, job numbers,.....

Were XXXX is the address, FCODE is the proper fast menu code (se manual for module) and STRING is the value.

5.4.5.9. GetVolume(XXXX,FCODE);

This can be used to get current volume settings from modules such as MinVolume and MaxVolume from a dispense point..

Were XXXX is the address, FCODE is the proper fast menu code (se manual for module) and function returns the value requested.

5.4.5.10. SetVolume(XXXX,FCODE,VOL);

This can be used to set decimal parameters in modules such as Volumes.

Were XXXX is the address, FCODE is the proper fast code (se manual for module) and VOL is the value.

5.4.5.11. SetAddress(XXXX,0x800,YYYY);

This is used to change the address of a module.

Were XXXX is the present address (DFF0 if reset button has been pressed for more than 5 sec), 0X800 is the memory address for the address for a module (fixed) and YYYY is the new address of the module.

The address field in OriLink® is 16-bit hexadecimal so address range is 0000 to FFFF but we strongly recommend following the addressing recommendations stated in the module manuals.

Adress	Module
0000 – 0xxx	Forbidden
1000 – 1xxx	MPDM module
2000 – 2xxx	Printer module
2999	PC database
3000 – 3xxx	Keypad module
4000 – 4xxx	LED module
5000 – 5xxx	PLC-Modules
6000 – 6xxx	<i>Reserved</i>
7000 – 7xxx	<i>Reserved</i>
8000 – 8xxx	TCM/TSM module
9000 – 9xxx	<i>Reserved</i>
A000 – FFFF	Forbidden

5.4.5.12. MemSetInt('XXXX', MEMADDRESS,INT);

This can be used to set INTEGER parameters in modules such as Mask, Rights, PPU,.....

Were XXXX is the address, MEMADDRESS is a proper memory address and INT is the value.

5.4.5.13. MemSetString(XXXX, MEMADDRESS,STRING);

This can be used to set TEXT parameters in memory of modules such as Names, part numbers, job numbers,.....

Were XXXX is the address, MEMADDRESS is a proper memory address and STRING is the value.

5.4.5.14. MemSetVolume(XXXX, MEMADDRESS,VOL);

This can be used to set DECIMAL parameters in the memory of a module such as MinVol, MaxVol,....

Were XXXX is the address, MEMADDRESS is a proper memory address and VOL is the value.

5.4.5.15. MemArrayFill(XXXX, FROM, TO, VALUE);

This can be used to fill a part of the memory in a module with a value.

Were XXXX is the address, FROM is the start memory address, TO is the end memory address and VALUE is the value.

5.5. Older script calls syntax left for backward compatibility.

5.5.1. KPSetGlobalResolverMessage (Seconds,'LCD Top Row ','LCD Bottom ROW ');

Use this if you want to set a global resolve message for the keypad LCD

Where Seconds is the show time, LCD Top ROW is max 15 characters and LCD Bottom ROW is max 16 characters.

5.5.2. KPString(Seconds,'LCD Top Row ','LCD Bottom ROW ')

Use this if you want to show a question on a keypad LCD and use the input as a parameter.

Where Seconds is the show time, LCD Top ROW is max 15 characters and LCD Bottom ROW is max 16 characters.

5.5.3. KPMessage(Seconds,'LCD Top Row ','LCD Bottom ROW ')

Use this if you want to show a message on keypad LCD

Where Seconds is the show time, LCD Top ROW is max 15 characters and LCD Bottom ROW is max 16 characters.

5.5.4. KPMessageQuit(Seconds,'LCD Top Row ','LCD Bottom ROW ')

Use this if you want to leave a keypad session and show a quit message on the keypad LCD

Where Seconds is the show time, LCD Top ROW is max 15 characters and LCD Bottom ROW is max 16 characters. The parameter is the answer to the LCD Top ROW question

5.5.5. KPFollowKey("+Some parameter+ '?','LCD Bottom row')

6. Writing scripts

Before you start writing your own scripts try out the scripts that comes with the installation first.

To try them set up a system with one module of each with their addresses set to XXX5, MPDM = 1005, PM = 2005, etc..

Make sure that you are running OriLink® WinTools professional validated for script service and that you have loaded WinDB service.

Set up all dispense points to only ask for volume.

Hint! Run a script and see what happens, start with a simple one such as !UpdateTank, !Preset or RemoteOneReel. Then open it by right click on it and select edit. Try to understand how it is written then run it again, etc.....

When you start to understand how it works try to make a own script.

6.1. My first script

Start with a simple one.

I want to open 2 dispense points at the same time. One for 1 L and the other for 0.5 L

Begin with opening a new text document and save it as

C:\Orilink\CSL_SMIX.csl

Load the needed basic script. This is done by the sign #

```
#loadScript '!Base.csl'
```

Then set the startpoint of the script.

```
#loadScript '!Base.csl'
main()
```

Then set the start of the commands

```
#loadScript '!Base.csl'
main()
{
```

First 3 commands is not necessary but good for information and error fixing. Command lines must end with ;

```
#loadScript '!Base.csl'
main()
{
    sysSleep(50);
    ocpPrint('-> Starting _SMIX.csl');
    ocpPrint(' Open Mix 33%');
```

Then add the dispense commands

```
#loadScript '!Base.csl'
main()
{
    sysSleep(50);
    ocpPrint('-> Starting _SMIX.csl');
    ocpPrint(' Open Mix 33%');

    ocpOpenReel(1,2,"",0.50);
    ocpOpenReel(2,2,"",1.00);
```

Then set the end of the commands

```
#loadScript '!Base.csl'
main()
{
    sysSleep(50);
    ocpPrint('-> Starting _SMIX.csl');
    ocpPrint(' Open Mix 33%');
```

```

    ocpOpenReel(1,2,"",0.50);
    ocpOpenReel(2,2,"",1.00);

    ocpPrint('<- _SMIX Ended');
    return 1;
}

```

Now save and exit.

To be able to stop this script if something goes wrong you must create a Stop script. We take this in one step.

Open a new text document write this

```

#loadScript '!Base.csl'
main()
{
    sysSleep(50);
    ocpPrint('-> Starting _KMIX.csl');
    ocpPrint(' Closeing Mix33%');
    ocpCloseReel(1,2);
    ocpCloseReel(2,2);
    ocpPrint(' Closeing Mix33%');
    ocpPrint('<- _KMIX Ended');
    return 1;
}

```

Save it as C:\OriLink\CSL_KMIX.csl

Now type MIX followed by ENTER key on the Keypad.

6.2. How does it work?

When you type a reel number or a word on a keypad the keypad launches a question for this on the OriLink® network. If there is something that can handle this request it will start a communication with the keypad.

If it is a script that matches this request it starts talking directly with the keypad. When the script has received answers to all its questions the script does the opening of reels etc.

6.3. If I have made something wrong in my script ?

The script service comes with a advanced error handling tool. This tool is started by right-clicking the OriLink® engine window and selecting the Properties “Script” item in the menu. In the window that opens there are two white fields. In the upper one you can see running scripts and if there is an error. If there is one or more errors mark the script in the upper field and there will be an error explanation in the lower field. There is also coordinates for the error in the format “ROW”: “CHARACTER POSITION”.

There is also a “Always show output dialog” check box. If this is checked a real-time log window will come up for each script running. By placing strategic *ocpPrint()*; statements in the script the script process can be visualised in the log window.

7. Updating the PC computer

The OriLink® WinTools software is based on standard Microsoft® software components such as DCOM, MFC6.2, MDAC and MSJET. This is done to simplify integration between OriLink® and other Windows® software. It also reduces the dedicated program code for OriLink® which reduces possible software errors.

It is recommended to update the OS to the latest level. Dependant of which OS and other software and how they are installed some standard components may be missing or incomplete.

By updating (patching) the Operating system, the OS, of the PC the risk of having error in these and other components is minimised.

7.1. Patch the OS to the latest version

The OS of the PC can be updated in several different ways. The easiest way is through the Internet. If the PC is not connected to the Internet it can be done through a CD.

It is not possible for us, Alentec & Orion AB, to have the updates on the OriLink® WinTools CD because the updates are from Microsoft® and language specific.

All Windows® OS can be updated with the correct version automatically through the web site “windowsupdate.microsoft.com”. Windows 98 and later has a shortcut to this web site with a name like “Windows Update” in the Start-menu.

Windows® 2000 /XP can be set-up for automatic updating.

The updates can also be downloaded from the web and put on a CD for use on computers without Internet connection. This can be done from <http://www.microsoft.com/technet/treeview/default.asp>

7.2. Install / Update MDAC

The MDAC (Microsoft Data Access Component) is a standard software package for database access that many different database software's uses. For example it is normally installed when installing Microsoft Office and in many cases together with the OS.

7.3. Install / Update MSJET

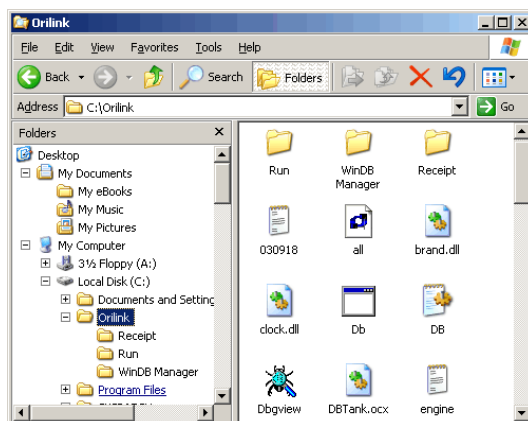
The MSJET (Microsoft JET) is a software package with ODBC drivers for several different databases such as SQL, ACCESS, PARADOX, FOXBORO, etc. It is normally installed when installing Microsoft Office and in many cases together with the OS.

7.4. Show hidden files and extensions

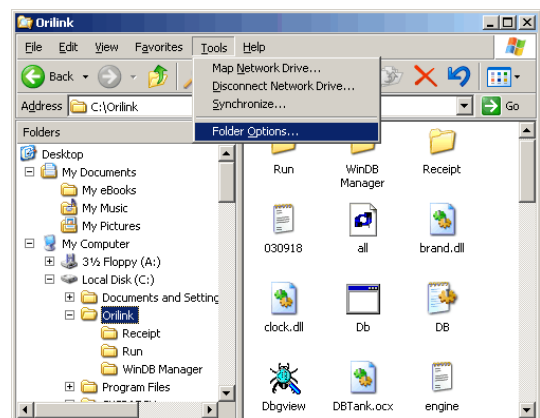
When you want to work with files from the Windows Explorer sometimes the file extensions (.XXX) are hidden. This is due to the fact that file extensions for registered file types are hidden as default in Windows.

Changing folder options can change this. Folder options are changed a little different depending on the Windows version (9X/NT/2K/XP). Below is a description of how to do this in Windows XP.

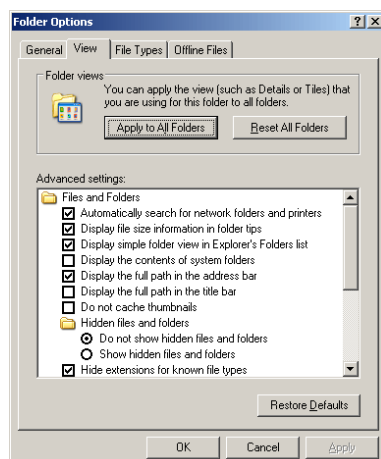
First open Windows Explorer and browse to *C:\OriLink*.



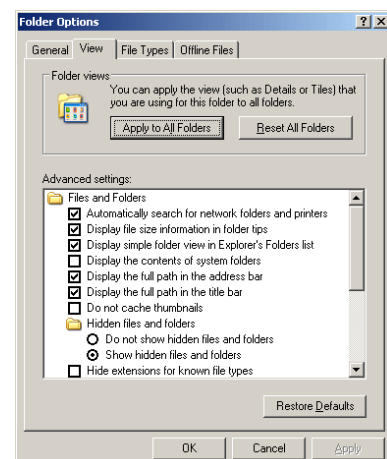
Then select Tools -> Folder Options.



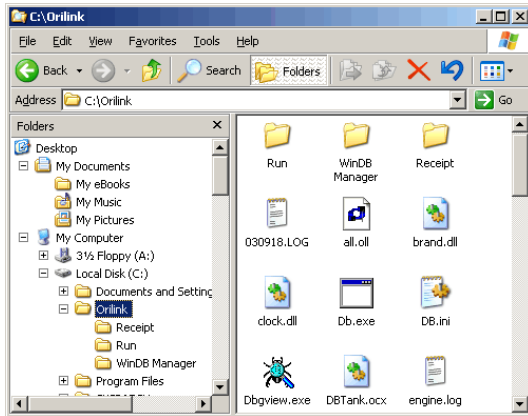
The default looks like this.



Check “Display the full path in the titlebar” and “Show hidden files and folders”. Uncheck “Hide extensions for known file types”. Like this,



Click on OK-button.



Now you can see all files and file extensions.

When you are finished with the files you can do the same thing backwards to restore it as it was before.

8. GNU License

Copyright © 1998-2001, Informatik-Buero Koch (IBK) - Landquart - Switzerland

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License version 2 as published by the Free Software Foundation.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License below for more details.

The General Public License (GPL) Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc. 675 Mass Ave, Cambridge, MA 02139, USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the

Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
 - a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
 - b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
 - c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
 - a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS