HMA OS800

Display Design Guide

Version 1.1, march 9, 2006

Released by ted

HØGLUND
marine automasjon as

CONTENTS

1 1 Introduction

1.1 Intention and scope

This document is primarily intended as a user manual for designing displays in Delphi for use with HMA OS800, and presenting information about the functions for display design.

The installed options, may wary for different projects.

This document is describing the different HMA functions in Delphi.

For additional information, see separate documents listed in **Related Documentation** at the end of this document.

1.2 Notice

The information in this document is subject to change without notice and should not be construed as a commitment by Høglund Marine Automation AS. Høglund Marine Automation AS assumes no responsibility for any errors that may appear in this document. Actual implementations may vary according to agreement.

Copyright © 2005 Høglund Marine Automation AS.

1.3 Abbreviations and definitions

HC800	HMA Concept 800 - an automation concept with integrated operator stations and process controllers.
Form	Display designed in Delphi
GMR	Graphical Manoeuvre Recorder
HMA	Høglund Marine Automation A/S.
VDU	Visual Display Unit.
Process Display	Main background in screen centre showing a part of the process
Top Display	The upper part of the monitor, above the process display
Display Dialog	Below the process display, bottom part of the monitor, left. Contains indication and commands regarding the process display
Object Display	Free floating window with details on the currently selected object in the process display.

1.4 Trademarks

This system, including, but not limited to, software, hardware, and documentation is copyright Høglund Marine Automation AS, Tønsberg, Norway.

1.5 Introduction to display design

The main purpose of the VDU display is so the operator (engineer) quickly can get an overview of the selected ship process. On the VDU all the important signals an operator may need to control the process safely are displayed. The displays should be presented in an easy and user-friendly manner. Try to avoid moving graphics and flashing objects when not in alarm state. This may look fancy but draws the attention from the rest of the display. The same goes for the colour scheme. Reds and greens should be avoided in drawings and used only for signal presentation.

There are some factors to consider before starting the design of a VDU display.

1) The layout: This includes the colour scheme, presentation of objects etc. The colour schemes presented are recommended, but can be changed if the customer uses a different system.
2) What will be included in the presentation? Some ship-owners like to include the entire PI&D with static objects of for example manually operated valves. Others prefer only to include the dynamic objects such as remote valves, motors etc.

2 Displays

The thing to think about concerning the display design is the size. It all boils down to the screen resolution. Another factor is the top display and bottom display size.

The screens used in the HMA design have a resolution of 1280x1024 pixels. The screen is divided in 3 parts from top to bottom; the top display, the process display and the bottom display (in the object display folder, name starting with Dialog_xxxxxx).

When deducting the size of the top and bottom display, the approx size left for the process display is 1271x859 pixels.

2.1 Top Display

This display shows the data the designer want shown in every VDU display. This could be the clock and date, Silence button for alarm, Show the last alarm, thruster indication etc.

2.2 Process display

The process display describes the process with simple drawings and symbols so the selected process can be easily understood. This is different for every process.

2.3 Bottom display

This has fast select buttons for quickly moving from one process display to another. It can be different for every OS or more accurately to whom man be logged on. I.E. If the operator is logged on as an engineer the bottom display will show fast select buttons for engine process displays.

2.4 Folder structure

It's important to use a certain structure when saving the forms so GMR100 can recognize them correctly. The folder names may not be changed.

The structure is as follows:

```
GMR    Displays    BasicDisplays
                   ObjDisplays
                   ProcDisplays
                   TopDisplays
```

2.4.1 BasicDisplay folder

This folder contains the basic displays. IE start up displays. HMA have 3 BasicDisplays when using a 3-screen configuration. Some deliveries do not use this folder. Instead a process display is the first display shown.

2.4.2 ObjDisplay folder

This folder contains the displays of each object. IE motors, valves etc. These displays are popup displays activated by the user. This folder also contains the bottom displays.

2.4.3 ProcDisplays

This folder contains the process displays. These are the displays for each process; cooling water, main engine and so on.

2.4.4 TopDisplays

The displays are shown in every display overlaying the process display in the top of the screen. Usually used to present data the designers want shown constantly.

3 Starting up Delphi

The program used by HMA for VDU design is Delphi 5.

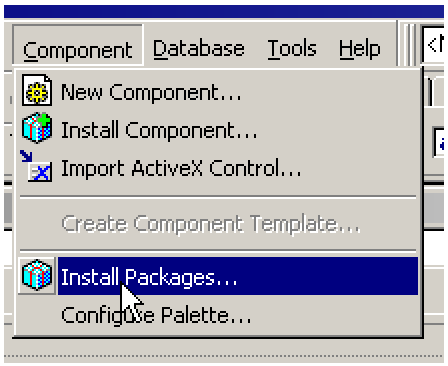
Delphi programs source files are usually stored in two file types: ASCII code files (.pas) and resource files (.dfm). This means that Delphi makes two files for each Form file (files with .dfm extension) and one or more Unit files (.pas extension). IE, one form has one .pas file and one .dfm file. Both must be in the display folder. Dfm files contain the details (properties) of the objects contained in a form. The DFM file is the binary data used to set up initial data for components (IE, the properties you set in design mode rather than in code).

Every time we change a form's position, a button's caption or assign an event procedure to a component, Delphi writes those modifications in a DFM.

3.1 Loading the design package into Delphi

Before working with VDU's, you have to load the OSPACK.BPL design package. This file is currently in version 50. The version number is seen by right-clicking the file in the windows explorere, choosing properties and version. Having the file available, this is what you must do to install it in Delphi:

Use "Components / Install packages" as shown.

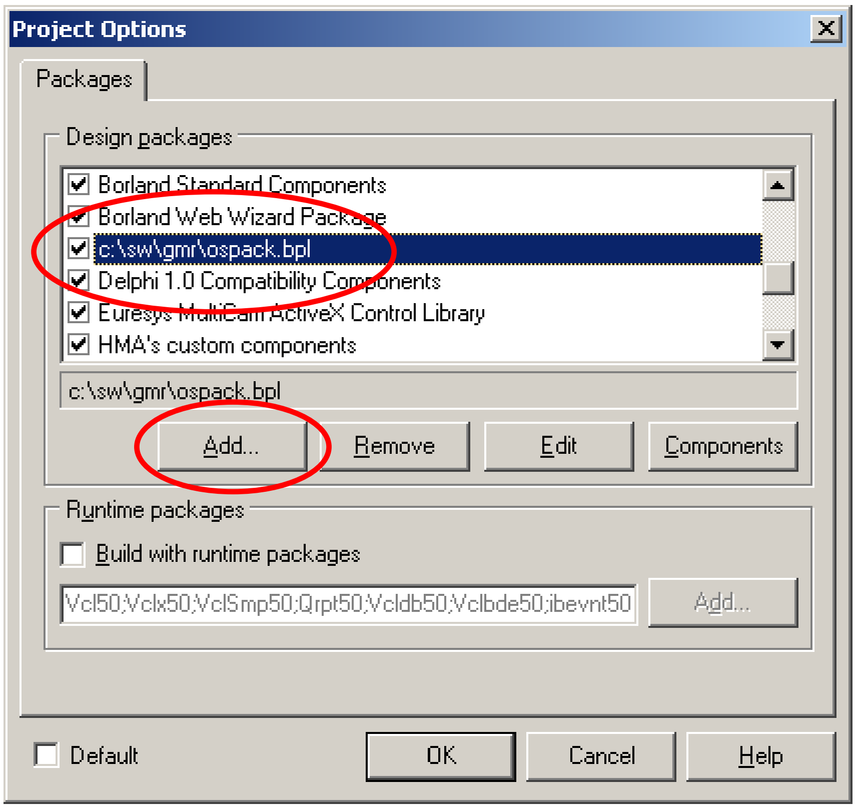


HMA`s Delphi package library (BPL file):

This is a HMA package and new versions are released continuously.

This file contains the latest objects and functionalities. HMA uses a folder called latest BPL witch we copy the latest BPL file down to. IE when Delphi is started the latest BPL file is loaded automatically. (All other Delphi applications must be shut down to upgrade the BPL file.)

NB!! New BPL cannot be used for old projects!! After delivery of a system the latest BPL file must be downloaded to the project folder. This BPL file must be used to edit this project. Remember to disable the latest BPL.


Project Options

Packages

Design packages

- ☑ Borland Standard Components
- ☑ Borland Web Wizard Package
- ☑ c:\sw\gmr\ospack.bpl
- ☑ Delphi 1.0 Compatibility Components
- ☑ Euresys MultiCam ActiveX Control Library
- ☑ HMA's custom components

c:\sw\gmr\ospack.bpl

Runtime packages

☐ Build with runtime packages

Vcl50;Vclx50;VclSmp50;Qrpt50;Vcldb50;Vclbde50;ibevnt50

☐ Default

The HMA objects from the BPL are then found on the palette tool bar

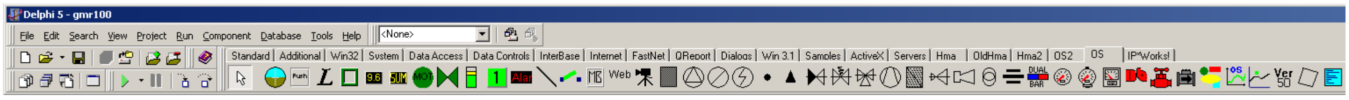

Delphi 5 - gmr100
File Edit Search View Project Run Component Database Tools Help
<None>
Standard | Additional | Win32 | System | Data Access | Data Controls | InterBase | Internet | FastNet | QReport | Dialogs | Win 3.1 | Samples | ActiveX | Servers | Hma | OldHma | Hma2 | OS2 | OS | IP*Works!

Fig. 1.0 Palette toolbar in Delphi

The main functions are found under “OS”, and “OS2” pallette tabs. There are also some standard objects used frequently, for example the image object in the additional tab.

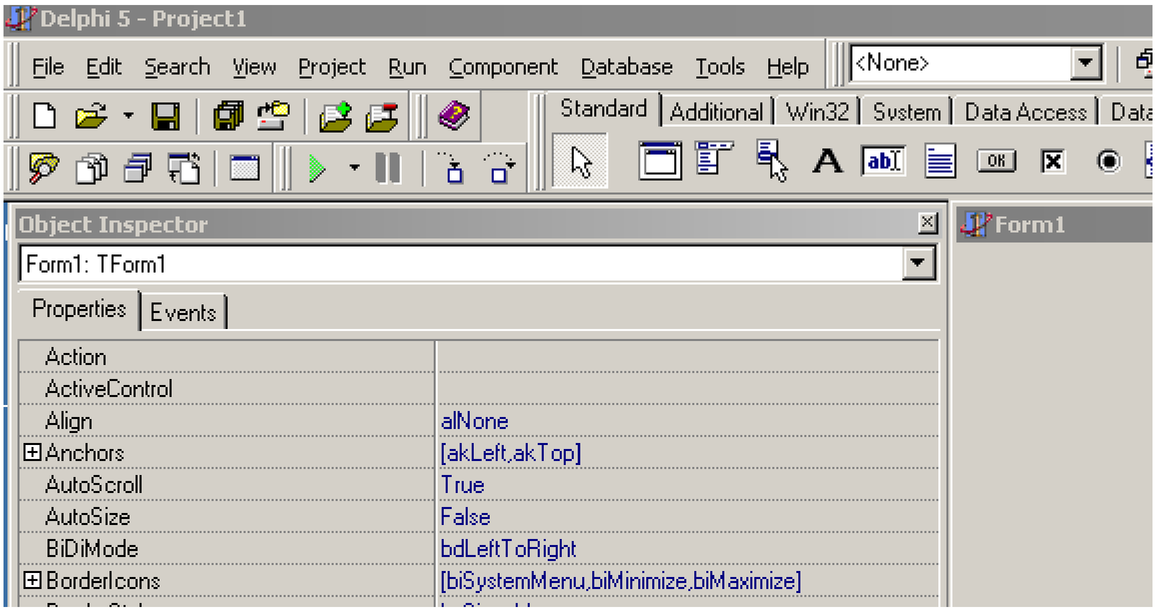
For more instructions in how to use Delphi check out the Borland Delphi website:
http://www.borland.com/delphi/

3.2 Fast keys in Delphi

Ctrl [C]	Copy
Ctrl [V]	Paste
Ctrl [X]	Delete
Shift	Hold the shift button and click on objects to highlight more than one object. With this you can set the same property for many objects. Highlight the objects first, select/type a property in the object inspector and press enter. Now all the highlighted objects have the same property.
Ctrl	Hold the Ctrl Button to move highlighted objects by the arrow keys.

3.3 Basic options

When opening Delphi the designer will get a blank form named Form1. The Object Inspector will also pop up. If this is not the case press F11 or use the View option on the main toolbar and select Object Inspector.


Delphi 5 - Project1
File Edit Search View Project Run Component Database Tools Help
<None>
Standard | Additional | Win32 | System | Data Access | Data
Object Inspector
Form1
Properties
Events

Form1: TForm1

Action	
ActiveControl	
Align	alNone
⊞Anchors	[akLeft,akTop]
AutoScroll	True
AutoSize	False
BiDiMode	bdLeftToRight
⊞BorderIcons	[biSystemMenu,biMinimize,biMaximize]

Fig. 1.1 Opening Delphi

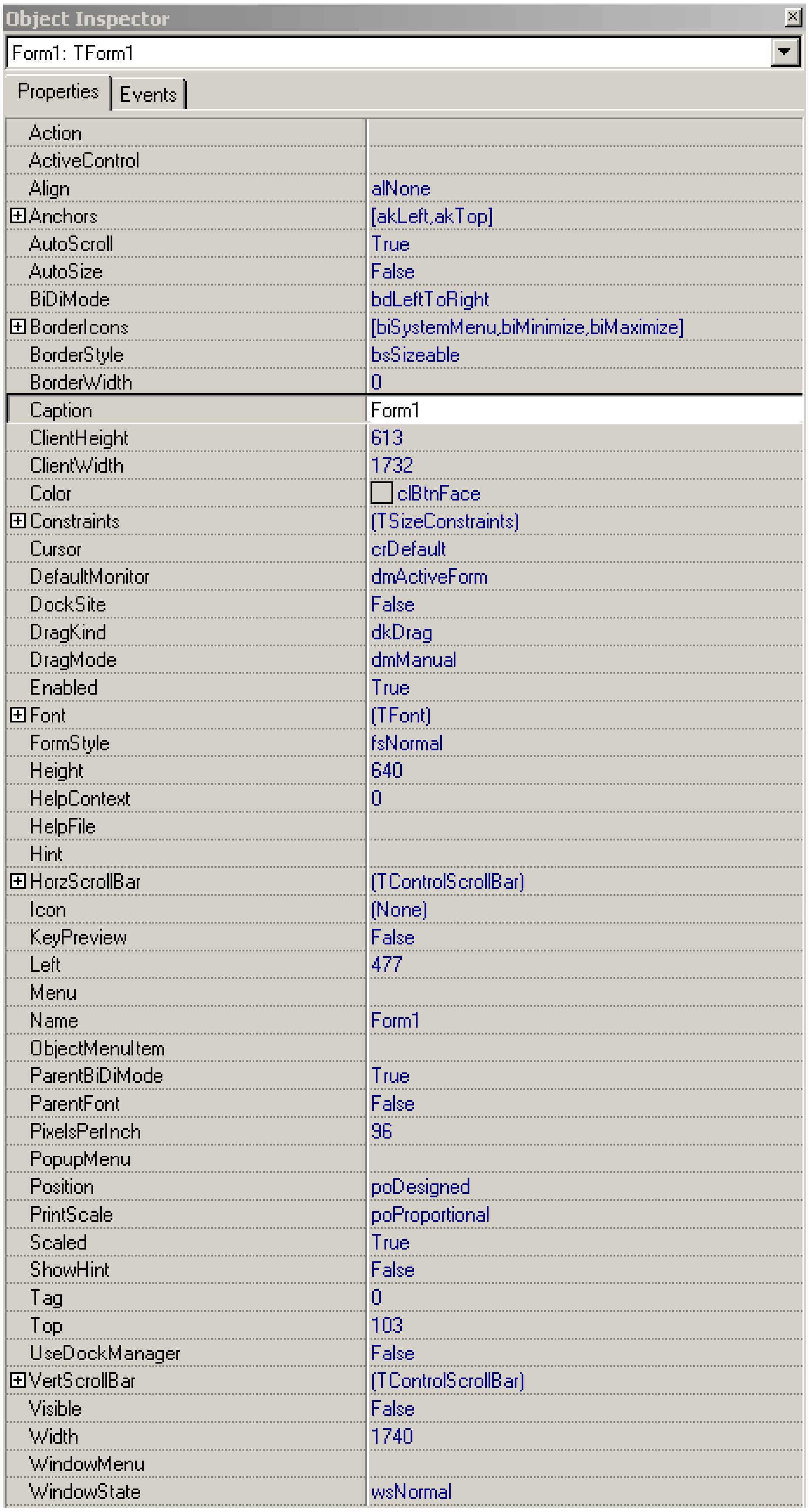
3.3.1 Object Inspector

The object inspector is the place to configure all objects, forms etc.

Here the designer set the size of objects and the properties, i.e. the variables used for "triggering the dynamic functions etc."

The number of properties is different for each object.

The various objects with unique properties will be listed later in this document.



Object Inspector

Form1: TForm1

Properties | Events

Action	
ActiveControl	
Align	alNone
⊞Anchors	[akLeft,akTop]
AutoScroll	True
AutoSize	False
BiDiMode	bdLeftToRight
⊞BorderIcons	[biSystemMenu,biMinimize,biMaximize]
BorderStyle	bsSizeable
BorderWidth	0
Caption	Form1
ClientHeight	613
ClientWidth	1732
Color	☐ clBtnFace
⊞Constraints	(TSizeConstraints)
Cursor	crDefault
DefaultMonitor	dmActiveForm
DockSite	False
DragKind	dkDrag
DragMode	dmManual
Enabled	True
⊞Font	(TFont)
FormStyle	fsNormal
Height	640
HelpContext	0
HelpFile	
Hint	
⊞HorzScrollBar	(TControlScrollBar)
Icon	(None)
KeyPreview	False
Left	477
Menu	
Name	Form1
ObjectMenuItem	
ParentBiDiMode	True
ParentFont	False
PixelsPerInch	96
PopupMenu	
Position	poDesigned
PrintScale	poProportional
Scaled	True
ShowHint	False
Tag	0
Top	103
UseDockManager	False
⊞VertScrollBar	(TControlScrollBar)
Visible	False
Width	1740
WindowMenu	
WindowState	wsNormal

Fig 1.2 Object Inspector

3.4 Creating a form

This chapter will present how to create a process display. The top and bottom display are done in the same way.

As previously mentioned the approx size left for the process display is 1271x859 pixels.

After opening Delphi a standard form pops up. This is named Form1. We will create a form as an example throughout this Manual. There will be a basic form using the most common options.

First click the form and the designer will get all the options for a form in the object inspector.

In the object inspector rename the form to Testform. Set its size to height 859 and width 1271. Then set the background colour. HMA uses a different colour than listed the standard palette. (The colours can be set in the CMYK using a $ sign in front of them.) HMA uses a dark background colour typed with CMYK. ($00333333)

For GMR100 to recognize this form, as display the designer must add a special HMA object. This can be found in the HMA library tab in the palette toolbar. The object looks like this:

Place this anywhere on the form and the form will be recognized by GMR100.

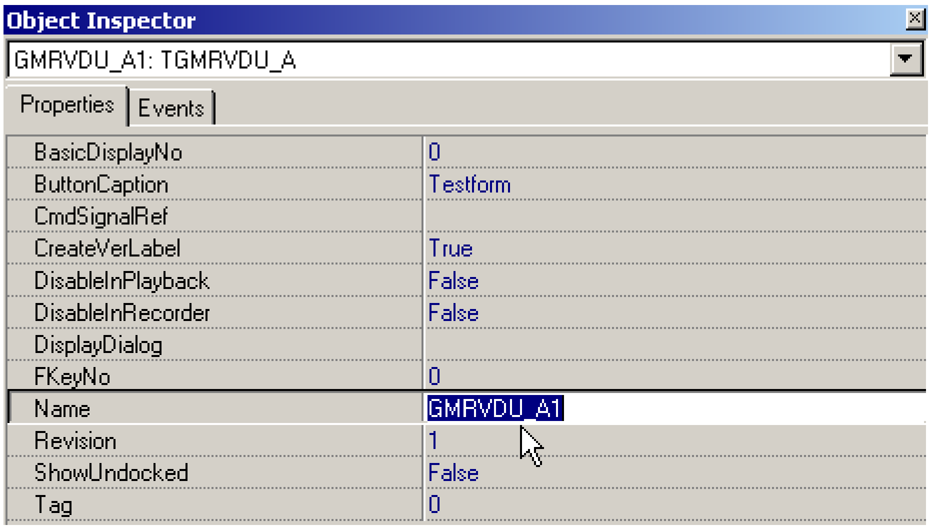
This object also has other useful options. When selecting this object the object inspector will look like this:

The FkeyNo is used if one would like to use function key on the keyboard to access this display.

ShowUndocked is an indication if the designer would like the display to be free floating on the screen, for instance when on an object display.

DisplayDialog Indicates the bottom display wanted for this VDU display.

The revision number is also important to follow up.



BasicDisplayNo	0
ButtonCaption	Testform
CmdSignalRef	
CreateVerLabel	True
DisableInPlayback	False
DisableInRecorder	False
DisplayDialog	
FKeyNo	0
Name	GMRVDU_A1
Revision	1
ShowUndocked	False
Tag	0

Fig 1.3 Object Inspector with GMRVDU

The form should now look something like this.

The creation of this form continues in chapter 3.6 (Adding objects to a form)

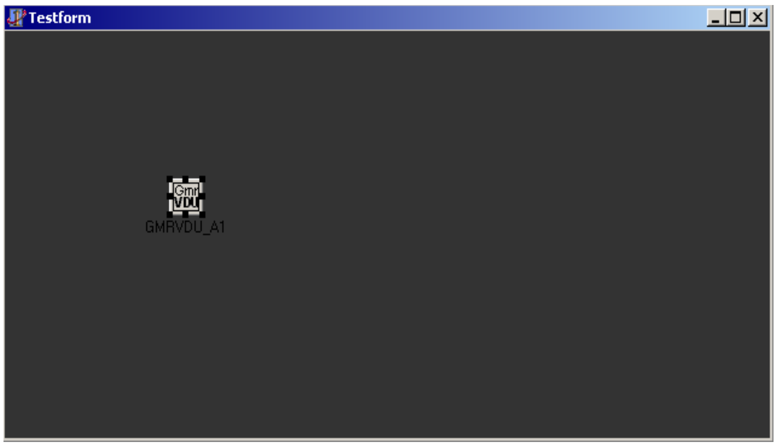
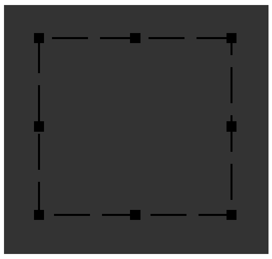
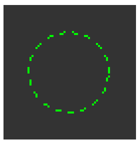
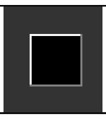


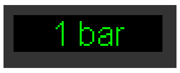
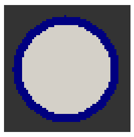
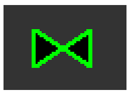
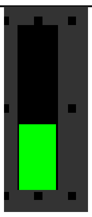
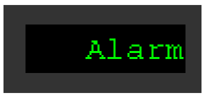
Fig 1.4 Test form with OSVDU

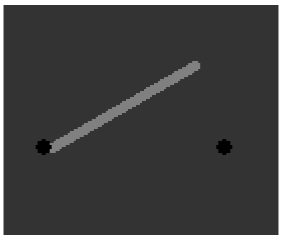
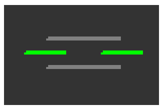
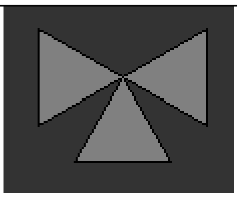
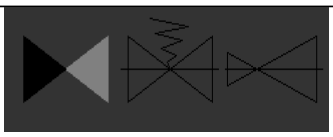
3.5 Objects in a form

Objects are selected from the palette toolbar shown in figure 1.0.

3.5.1 The objects and functions most used

Image: In the image object one can put image files. The files must be in jpg, jpeg, bmp, ico, emf or wmf format. HMA uses the emf format for best resolution, it has small a data file and because its vector. Delphi can also resize and stretch the image. The image object is mostly used for inserting static drawings.		Found in the additional tab
HMAShape: The HMA shape is an outline tool. The shape can be circle, rectangle, framed rectangle or rounded rectangle. One can also use different pen styles. IE dot, dash, solid, clear, dash dot, dash dot dot and inside frame. One can also set a tagname for making the shape invisible. The HMAShape object can be filled with any colour.		Found in the HMA tab
OSPushButton: A very useful object with many options. Can also be made invisible and placed over an image to make a part of an image clickable. Can write scripts, write commands, ask values, jump to any display etc. An image can be inserted in the button.	PushButton	Found in the OS tab
Label and OSLabel: For simply marking objects in the displays use the label function from the standard tab. For more advanced functions such as blinking or invisible use the OSLabel from the OS tab.	TOSLabel	Found in the OS tab and in the Standard tab.
OSDI: Digital input marker. Used for displaying digital inputs. Have 3 different states. OFF, SET, ALARM. The colours for the 3 states		Found in the OS tab

are standard and should not be changed. Has many functions similar to a pushbutton, but does not have the clickable functions like the pushbutton has.		
OSAI: Analogue input marker. Used to show measured data from an analogue input or a calculated analogue input. Can also be used for displaying text. When indicating a measured or calculated value the unit is set in the UnitHma option in the object inspector. If no unit set – here.		Found in the OS tab
OSMotor: The OSMotor object is basically used for motor/generator indication. This object has many states. The outer ring has 5 different colours indications indicating the state of the motor. Blue for manually operated, grey for auto, green for available, white for local and red for alarm. The objects centre fills with the same colour as the outer ring when the motor is running. The motor object also have an option to be set as an duty/stby motor, and will then indicate with a D for duty or S for stby. Can also have an arrow for indicating pump direction.		Found in the OS tab
OSValve: Basic valve can be used for all valves with feedback. Can also have a manual/auto indication for automatic valves that can be set to manual.		Found in the OS tab
OSBar: The bar is mainly used for indicating for example a volume in percent 0-100%. The can also indicate a set range. It can show the setpoint of 4 alarm states. IE lowlow, low, high, highhigh. These alarm limits must be set to true in the object inspector.		Found in the OS tab
OSLastProcAlarm: Used for indicating the last process alarm. Usually placed in the top display.		Found in the OS tab
OSLine: Generally used for indicating pipelines but can also be used to draw outlines. If used for outlines LineWidth is set to 1 for activating options for making the lines dotted. The angle of the line can be changed by a horizontal vertical option in the object inspector. When used for indicating pipelines: The default on/off colour is green/grey. Some want the colours to be different. The On en Off colour also has a setting called ColorConst. HMA DESIGERS usually set this to the name of the process display. IE For the Mud process display the ColorConst will be MudON for ColorOn and MudOff for ColorOFF. Make one horizontal line and one vertical line like this and copy/paste them on the rest of the process display. The colours can then be set later in GMR100.xls For toggling the line colours on/off we use the connecting remote valves. When a valve changes state the connecting line will also change state. Done by inserting these valve tags in PropValue (TagCollection).		Found in the OS tab

OSSwitch: OSSwitch are used for indicating remote bus-ties and breakers in the main switchboard.		Found in the OS tab
OSSeutValve: The OSSeutValve is used to indicate the seut valves. These are not remote or have feedback. The operator sets the status by right clicking on the valve and select open or closed.		Found in the OS tab
OS3Wvalve: 3-way valve. Same options as a regular valve. Have 3 individual properties, each side of the valve that can be controlled.		Found in the OS tab
Static valves for indication purpose: Can be rotated in any angle.		Found in the OS tab

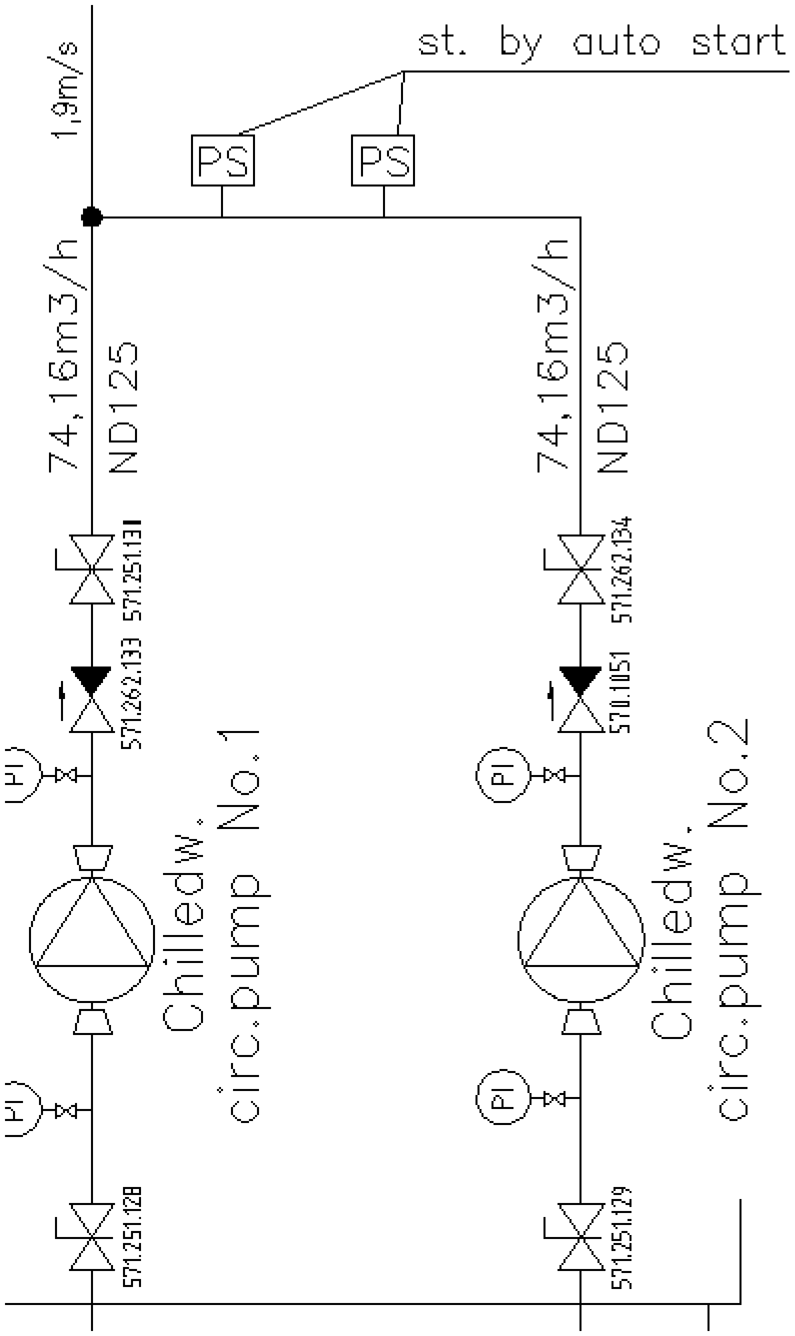
3.6 Adding objects to a form

The objects are inserted by “Drag and Drop” The objects size colour and so forth are set in the object inspector.

But just a blank grey display isn’t much fun so now its time to start adding objects to describe the process in the best possible manner.

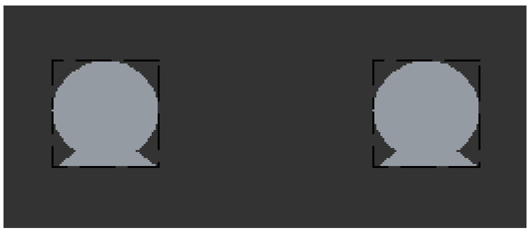
First we take a look at the PI&D.

This out cut from a PI&D shows to duty/standby pumps. Only the pumps have feedback in the system and the rest is static. To find out what is dynamic and what is static the whole PI&D page must be studied. All items on the PI&D are marked and have a reference. Most commonly used is the SFI code.



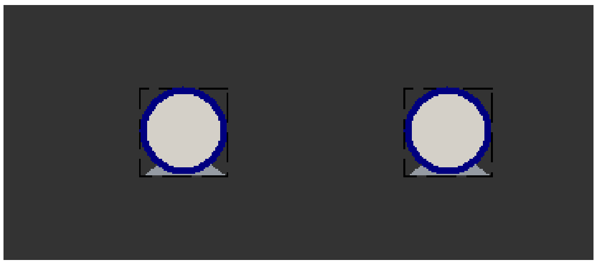
Now we try to replicate this into our testform.

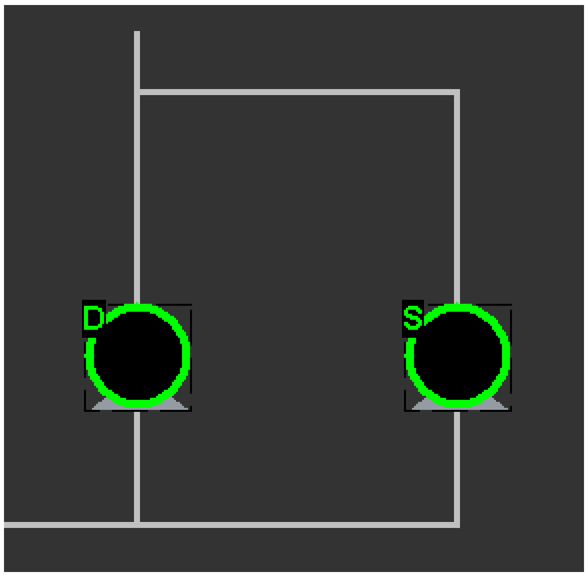
First insert an image, and then insert a picture of a symbol of a pump into the image object. When you are happy with the look: copy/paste it. It should now look something like this.



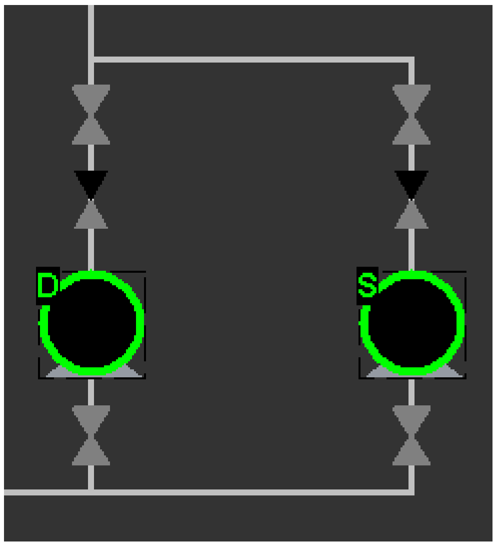
The next thing HMA DESIGERS do is inserting the motor object. Making sure to activate the duty standby function in object inspector.

Now it should look something like this.

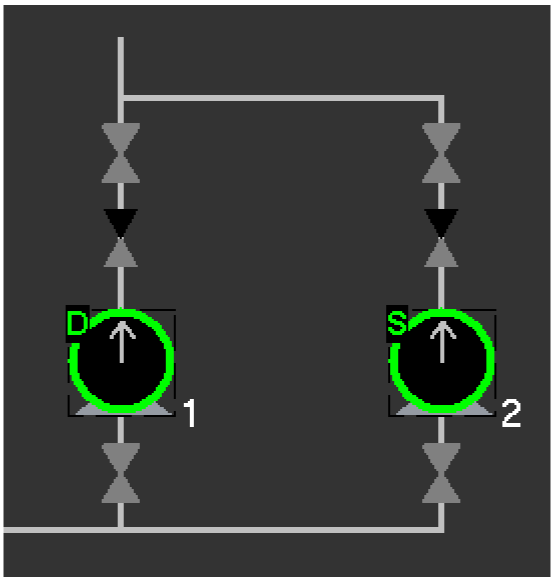




The next thing to do is to insert the lines. The motors background colour has been set to black. The duty standby function has been activated in the object inspector.



Inserting the valves. The grey valves are standard OSValve with colour set to grey. This to indicate that these are manually operated with no feedback signal. The non return valves are static objects from the OS tab.

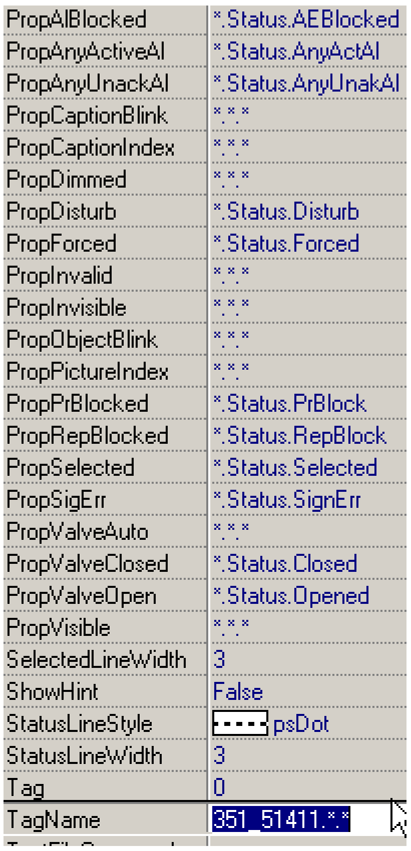


You then add the numbering using the label from the standard tab with colour set to white. Add arrows to indicate the flow direction.

If you are happy with the look of the pumps its time to tag to dynamic objects and name the static ones. HMA usually name the static parts also. The names (SFI No) will then show when the user holds the mouse pointer over them.

3.6.1 Tagging dynamic objects

Tagging the dynamic objects is done in Delphi. There are different options for different objects when it comes to tagging, in this manual will try and explain the most common.



PropAlBlocked	*.Status.AEBlocked
PropAnyActiveAl	*.Status.AnyActAl
PropAnyUnackAl	*.Status.AnyUnakAl
PropCaptionBlink	*.*.*
PropCaptionIndex	*.*.*
PropDimmed	*.*.*
PropDisturb	*.Status.Disturb
PropForced	*.Status.Forced
PropInvalid	*.*.*
PropInvisible	*.*.*
PropObjectBlink	*.*.*
PropPictureIndex	*.*.*
PropPrBlocked	*.Status.PrBlock
PropRepBlocked	*.Status.RepBlock
PropSelected	*.Status.Selected
PropSigErr	*.Status.SignErr
PropValveAuto	*.*.*
PropValveClosed	*.Status.Closed
PropValveOpen	*.Status.Opened
PropVisible	*.*.*
SelectedLineWidth	3
ShowHint	False
StatusLineStyle	psDot
StatusLineWidth	3
Tag	0
TagName	351_51411.*.*

This picture is taken from the object inspector after pressing the OSValve object.

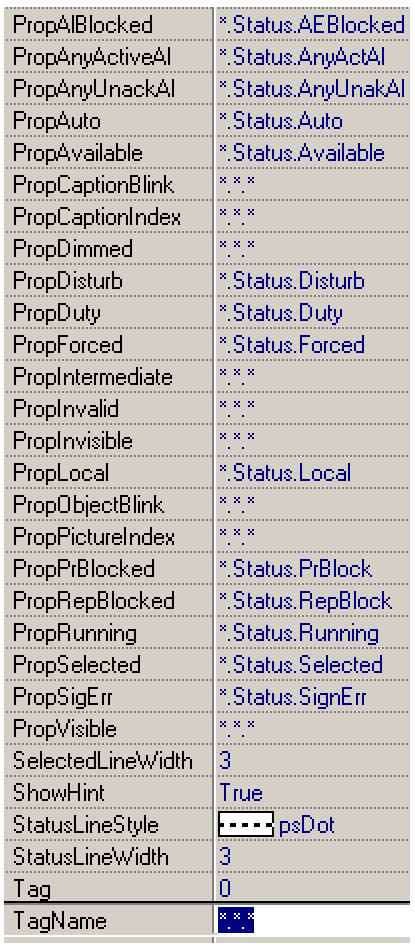
As the designer can se it's a lot of properties that can be activated. The default properties are the one activated here. One can insert properties by changing the *.*.* with *.Status.xxxxxxx.

In the bottom of the picture one will find the TagName. The number is taken from the SFI code. Its important that de tagname ends with .*.* or else the tag will lose its properties. Usually the .*.* will pop in automatically after entering the tag.

Show Hint is set to false. Set this to true if the designer want the tagname and description to popup when the user holds the mouse pointer over it.

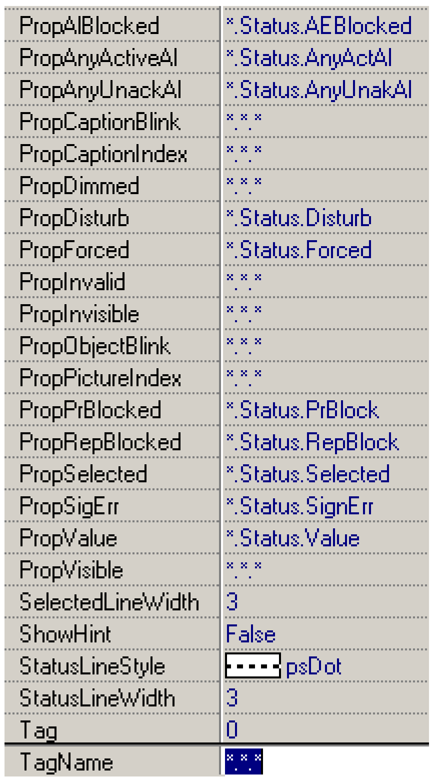
This picture is taken from the object inspector after pressing the OSMotor object.

As the designer can see the properties are a bit different. They can however be changed in the same way.

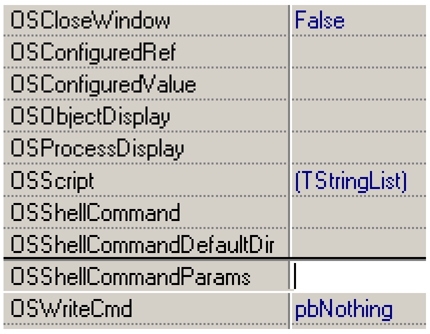


PropAlBlocked	*.Status.AEBlocked
PropAnyActiveAl	*.Status.AnyActAl
PropAnyUnackAl	*.Status.AnyUnakAl
PropAuto	*.Status.Auto
PropAvailable	*.Status.Available
PropCaptionBlink	*.*.*
PropCaptionIndex	*.*.*
PropDimmed	*.*.*
PropDisturb	*.Status.Disturb
PropDuty	*.Status.Duty
PropForced	*.Status.Forced
PropIntermediate	*.*.*
PropInvalid	*.*.*
PropInvisible	*.*.*
PropLocal	*.Status.Local
PropObjectBlink	*.*.*
PropPictureIndex	*.*.*
PropPrBlocked	*.Status.PrBlock
PropRepBlocked	*.Status.RepBlock
PropRunning	*.Status.Running
PropSelected	*.Status.Selected
PropSigErr	*.Status.SignErr
PropVisible	*.*.*
SelectedLineWidth	3
ShowHint	True
StatusLineStyle	psDot
StatusLineWidth	3
Tag	0
TagName	*.*.*

Taken from an OSDI object. NOTE! PropValue set to Status.Value.



PropAlBlocked	*.Status.AEBlocked
PropAnyActiveAl	*.Status.AnyActAl
PropAnyUnackAl	*.Status.AnyUnakAl
PropCaptionBlink	*.*.*
PropCaptionIndex	*.*.*
PropDimmed	*.*.*
PropDisturb	*.Status.Disturb
PropForced	*.Status.Forced
PropInvalid	*.*.*
PropInvisible	*.*.*
PropObjectBlink	*.*.*
PropPictureIndex	*.*.*
PropPrBlocked	*.Status.PrBlock
PropRepBlocked	*.Status.RepBlock
PropSelected	*.Status.Selected
PropSigErr	*.Status.SignErr
PropValue	*.Status.Value
PropVisible	*.*.*
SelectedLineWidth	3
ShowHint	False
StatusLineStyle	psDot
StatusLineWidth	3
Tag	0
TagName	*.*.*



OSCloseWindow	False
OSConfiguredRef	
OSConfiguredValue	
OSObjectDisplay	
OSProcessDisplay	
OSScript	(TStringList)
OSShellCommand	
OSShellCommandDefaultDir	
OSShellCommandParams	
OSWriteCmd	pbNothing

Taken from OSPushButton. In additional properties as described before the Pushbutton have some extra options. These are special functions for the Push Button. One can insert a name of a process display in OSProcessDisplay. Change OSWriteCmd to send true. And when the button is clicked in GMR100 the process display will change to the display typed in OSProcessDisplay. The Push Button has a lot of complex functions and is versatile.

The designer will find more options as he/she goes along. All the options and varieties make to long a list to type in this document. Try/Test is the recommended approach for special functions.

3.6.2 Naming static objects

Naming static objects are done if they are included in the process displays. Normally to use to help the operators identify different parts of the process display. To utilize the Hint function must be set to true. In the box that says hint the designer types in the information he/she want shown. Usually the SFI No. This is not necessary for dynamic objects with its tag number entered and the ShowHint property set to true.

3.6.3 Unit HMA

This option in the object inspector Sets the unit in an TOSAI. For instance we show an value in bar simply type bar in the UnitHMA. The value will them be shown in front of the unit. If one wants the unit in front of the value, insert a < in front of the unit in UnitHMA.

4 Tools

Useful tools when designing with Delphi.

4.1 FormFix

If the designer gets a problem with either form in the process of designing, it's likely that some parameters are wrong. This can result in problems closing Delphi. HMA has written a program called FormFix to repair the form. If Delphi fails to close do the following:

To solve this save the other opened forms manually. Delphi must then be closed by the task manager in windows. For an easy fix of this problem run the FormFix programme and enter the folder of the faulty form. This solves the problem automatically.

4.2 PropFix

A nice little programme, created by HMA. Used for changing properties. It's a very versatile programme that easily and effortlessly can save a lot of work. Functions similar to macro functions and easy search and replace functions.

NB! This programme can cause a lot of damage. Don't use this unless you have a backup of ALL forms!

5 Related Documentation

Document title	Description
HMA Concept 800	Gives a brief introduction to the HMA Concept 800 when connected to ABB Control IT process controllers.
HMA OS800 Product Guide	Gives an introduction to the HMA OS800 product.
HMA Concept 800 Solution Library	Describes in detail the HMA Concept 800 standard solutions.

6 Contact information

Høglund Marine Automation AS is a Norwegian company founded 1992, and is capable of developing customized software and hardware solutions as well as delivering standardized solutions. The mainly activities are:

- consultant services within marine automation
- supplier of automation products
- supplier complete automation systems (IAS)
- power management systems

See **www.hma.no** for more information.

Contact info

Phone	: +47 33414150	Organization number	: NO 966107650
Fax	: +47 33414946	Location	: Stålveien 11, N-3124 Tønsberg, Norway
Web	: *www.hma.no*	Manager	: Kåre Høglund
E-mail	: *mail@hma.no*		