



Optimizing your iX application

KI00324 2015-05

1 Function and area of use

This document discusses several different aspects of how an iX application can be improved to run smoother and different pit falls that a designer should be aware of when working with iX Developer.

This document can be seen as a compliment to iX Help documentation.

2 About this Start-Up document

This Start-Up document should not be considered as a complete manual. It is an aid to be able to start up a normal application quickly and easily. For further information we refer to the manual for iX Developer 2.0. This document and other Start-Up documents can be downloaded from www.beijerelectronics.com.

Please use the address manuals@beijerelectronics.com for feedback on our Start-Up documents.

3 Driver performance

Different signal types

Driver communication can be divided in two types of devices, Static and Dynamic and these are updated in different ways.

Static tags

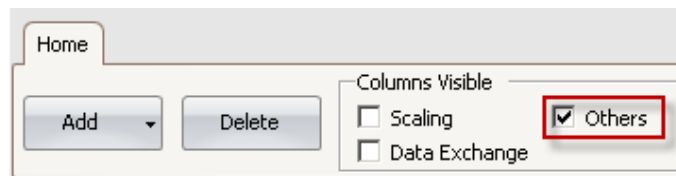
Static tags are always updated even if they are not shown on the HMI screen.

This type of devices is connected to Trends, Data loggers, Alarms, Data Exchange and system devices.

Any external tag which has been configured as **Always Active** will also be considered a Static Device.

Others			
Description	Poll Group	Always Active ▼	Non-volatile
	PollGroup1	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	PollGroup1	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	PollGroup1	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	PollGroup1	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	PollGroup1	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	PollGroup1	<input type="checkbox"/>	<input type="checkbox"/>

The **Always Active** setting is found under the “Others” Columns in the tags editor:



Tags which have actions configured on them are also treated as **Always Active**.

Tags used in scripts are also handled as **Always Active**.

Minimizing the amount of static devices is one way of improving the performance of your HMI application. Another way to improve the application is to keep your static devices in consecutive order.

Dynamic tags

Dynamic tags are only updated when they are shown on the screen and the most common example of this type of signal is a tag which is connected to an Analog numeric object; it is only updated when it is shown on the screen.

4 Keep signals in consecutive order

When the signals are transferred to the controller, all signals are not transferred simultaneously. Instead they are divided into packages with a number of signals in each package. To decrease the number of packages that have to be transferred and make the communication efficient this number has to be considered.

The number of signals in each package depends on the used driver and information about this can be found in the driver help for the used driver:

The screenshot shows the Allen-Bradley DF1 help interface. The left sidebar contains a table of contents with 'Efficient Communication' expanded to show 'Efficient packaging of signals'. The main content area has the title 'Efficient packaging of signals' and a paragraph: 'To make the communication quick and efficient the following should be read and what that can be done to optimize the reading.' Below this is a table:

	Analog signals	Digital signals
No. of signals/package	42	42
Waste	16	16

To make the communication as fast as possible the number of packages has to be minimized.

Consecutive signals require a minimum of used packages but it is not always possible to have consecutive devices.

In such cases the so-called waste between two signals has to be considered.

5 Waste

Waste is the maximum gap between two signals that still keeps them in the same package/telegram. The waste depends on the used driver and information about this can be found in the driver help for the used driver.



	Analog signals	Digital signals
No. of signals/package	29	124
Waste	20	0

To make communication times shorter, you also need to consider station handling compared to adding a separate controller for each station you wish to poll.

Since all our communication drivers are sequential, they will poll all devices from one station, followed by the next station in the list and so on.

If a lot of stations are configured and you use a variety of signals in each, this will lead to a long poll cycle for the configured controller. In this case, it's a good idea to split the stations into 2 or more separate controllers (not possible for serial interface drivers).

Note:

Each controller will take up a certain amount of resources (memory and CPU load).

This may require that you use more powerful hardware, for example upgrading from a T7A to a T7B.

6 Bit-addressed words compared to normal bit-devices

In most drivers it is better to use bit addressed words instead of normal bit devices. The reason for this is that when using bit address words you can fit more digital devices in one telegram compared to using bit devices.

Example:

	Analog signals	Digital signals
No. of signals/package	29	124
Waste	20	0

This specific driver can have 29 Analog devices in one telegram or 124 Digital devices(bits).

If you where to use bit addressed word devices in the selected driver you could fit 464 (29*16) digital devices in one package. This is almost four times as many devices.

7 ASCII Strings

ASCII strings are transmitted in separate telegrams and having a large number of strings will affect the communication performance negatively.

If an ASCII string only has a small number of different string values then it may be a good idea to use the **Text Library** functionality in iX with predetermined content to maximize the projects performance.

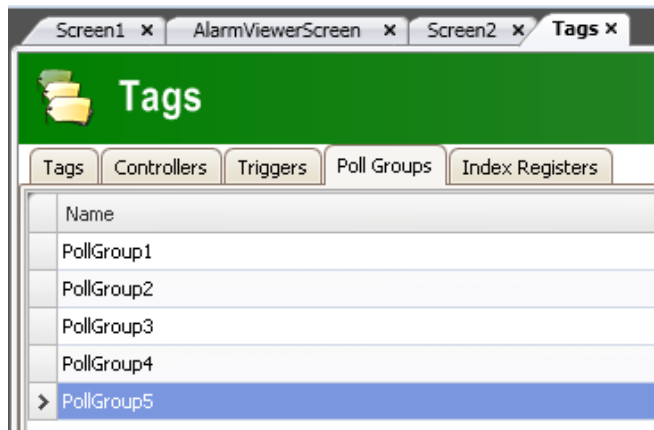
8 Poll groups

The default setting in iX Developer is that all signals update every 500ms.

If there are certain signals that don't need to be updated this fast it is possible to assign them to another poll group and slow down the update time.

If some need to be update more often, they can be configured in a faster poll group.

Poll groups are configured in the Tags editor, you can configure up to 5 different poll groups.



It is generally a good decision to let alarm tags be updated less regularly to optimize performance.

This is especially true if you have an application with a large amount of alarms.

9 Screen update time

Optimizing screen update time has to do with minimizing CPU load and Flash operations. This can make a big difference in a graphics intense project that approaches the limits of the panel's capabilities.

If the tags used in the relevant screens are connected to different poll groups the performance in the application could get better. To get the best performance try different poll groups and time interval with the controller used. The controller capacity effects the performance as well which means that the same settings may not work as good with another controller type. Always test the poll group and time interval with the specific controller in use to get the best performance.

An alarm viewer screen with a long list (default: 1000) will have a longer loading time compared to if it was configured to only show the last 100 alarms.

10 Resize symbols

The resize function in iX Developer is a good function to use if a small number of symbols are used.

If you are planning on incorporation a large number a symbols it is recommended to change the size before using the symbol in the application.

This can have a big impact on the performance of the panel.

Example:

If you are designing an application for an iX T7A, which has a resolution of 800*480 pixels, there is little point in using pictures with full HD resolution (1920*1080 pixels).

11 Gradients

Using objects with gradients creates a slightly higher load on the CPU and if a large number of them are used it will make the screen update/load time slower.



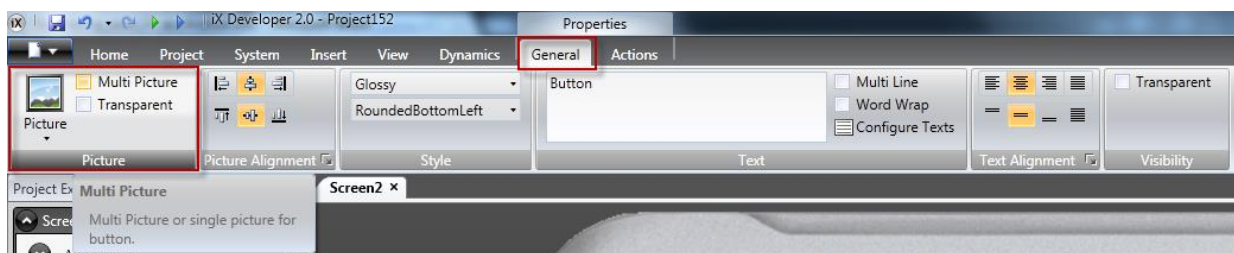
12 Consider how your Objects/Controls are used

Avoid “object stacking”. Placing several buttons on top of each other and controlling which is shown via dynamics is a bad way of solving a design problem.

A better solution (performance wise) would be to only have one button and write a script to handle the different actions needed.

The same is true for stacking symbols/multisymbols on top of buttons.

A better solution is to use the embedded symbols in the buttons, which can be configured under the General ribbon settings for the button:



Some objects are more CPU intense than others.

For example, the AnimatedLabel, RollerPanel, TouchComboBox and TouchListBox are more CPU intense than your average basic control, like the AnalogNumeric object.

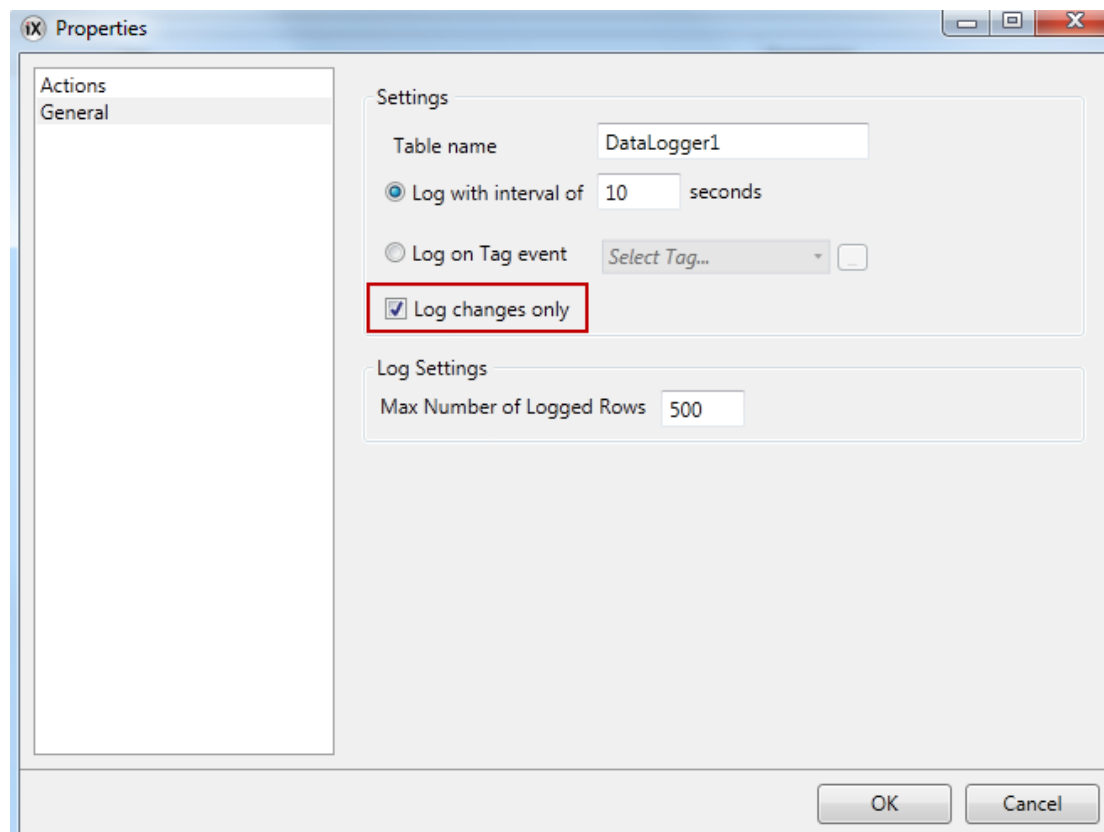
As a general rule, try to limit the use of CPU intense objects on each screen, especially if you are using entry level hardware like the TxA series.

13 Trends and Data logger

Using a lot of trends and data loggers will cause the panel to do a lot of flash operations and thereby slow down the screen update time.

One way to remedy this is the **Log changes only** option.

This can potentially reduce the number of write operations to the flash, if the tags configured in the datalogger does not change very often.



It is more efficient to have one large Datalogger containing many log values (tags), compared to having several Data loggers with a few log values in each.

14 Data Exchange

To maximize the performance of your data exchange, you should try to only use consecutive signals and you should use a data trigger to control the data exchange.

Value change triggered data exchange will greatly impact the performance of the HMI if the tags configured for data exchange change values frequently in the PLC.