

marathon VF2

microEnable 5 marathon V Series

Camera Link HS F2 Standard, Version 1.1

User Manual

Imprint

Silicon Software GmbH
Steubenstraße 46
68163 Mannheim, Germany
Tel.: +49 (0) 621 789507 0
Fax: +49 (0) 621 789507 10

© 2016 Silicon Software GmbH. All rights reserved.

Document Version: 2.0
Document Language: en (US)

Last Change: September 2016

Contents

1	Welcome to marathon VF2.....	7
1.1	Short Product Overview	7
1.1.1	Main Features	7
1.1.2	Ports and LEDs.....	12
1.1.3	Possible Topologies.....	13
1.1.4	Available Product Variants.....	15
1.2	Delivery Contents	15
1.2.1	Standard Delivery Contents	15
1.2.2	Optional Add-Ons (ask for availability)	15
1.3	Requirements	16
1.3.1	System Requirements	16
1.3.2	Software and Environment	16
2	Hardware Installation.....	18
2.1	Connecting marathon VF2 Physically	18
2.1.1	Requirements.....	18
2.1.2	Installing the Board	19
3	Software Installation.....	22
3.1	Installing the Runtime Software under Windows.....	22
3.2	Installing the Runtime Software under Linux.....	25
4	Using marathon VF2.....	26
4.1	Connecting to the Camera	26
4.2	Configuring the Camera	27
4.2.1	Autodiscovery	27
4.2.2	Configuring the Camera	30

4.3	Preparing the File System.....	32
4.4	Installing an Applet onto marathon VF2 (Flashing).....	33
4.5	Basic Steps	40
4.6	FPGA Fall-Back Configuration.....	44
4.6.1	Automatic Load of Available Configuration	44
4.6.2	Configuration Switch.....	45
4.6.3	Re-Flashing marathon VF2	46
5	Control and Configuration via SDK.....	46
6	Trigger System.....	47
6.1	Introduction.....	47
6.2	Trigger Interfaces on marathon VF2	47
6.2.1	Front GPIO (Slot Bracket)	48
6.2.1.1	Pin Layout Front GPIO	50
6.2.1.2	Input Configuration	51
6.2.1.3	Trigger Output TTL.....	55
6.2.2	GPIO (34-Pin Flat Cable Connector)	56
6.2.2.1	Trigger Extension Boards.....	58
6.2.2.2	Signal Types	59
6.2.2.3	Available Trigger Boards.....	59
6.2.3	Configuring GPIO Units on a Frame Grabber via Command Line	60
6.2.3.1	Configuring a GPIO Unit	60
6.2.3.2	Re-Setting Default Values.....	64
7	VisualApplets: Designing Individual Functionality	65
7.1	Installation.....	65
7.1.1	System Requirements	66
7.1.2	Installing VisualApplets	67
7.1.3	Installing the Xilinx Tools.....	72
7.1.4	Connecting VisualApplets to Xilinx	74

7.2	Working with VisualApplets	75
7.2.1	Workflow.....	76
7.2.2	Writing Applets on your Own	80
7.3	Running Your Applet on Hardware	80
7.3.1	Testing your Applet in microDisplay	81
7.3.2	Starting the Applet in your own Software	82
7.4	Further Reading.....	82
8	Programming a Trigger System with VisualApplets.....	83
8.1	Trigger Connectors on marathon VF2	83
8.2	GPI/GPO Operators in VisualApplets	85
8.2.1	Parameter ConnectorType.....	86
8.2.2	Parameter Pin_ID	88
8.3	Trigger Signals Addressing the Front GPIO (Slot Bracket).....	88
8.3.1	Pin Configuration on the Front GPIO	89
8.3.2	Trigger INPUT on the Front GPIO	90
8.3.2.1	Mode „Differential“ – Individual Pins	93
8.3.2.2	Mode „Single-Ended“ – Individual Pins.....	94
8.3.3	Trigger OUTPUT “TTL” on the Front GPIO	95
8.4	Trigger Signals Addressing the Trigger Extension Boards (via GPIO)	96
8.4.1	Trigger Extension Boards.....	96
8.4.2	Pin Layout and Pin IDs in VisualApplets	97
8.4.2.1	TTL Trigger Extension Board.....	98
8.4.2.2	Opto Trigger Extension Boards.....	104
8.5	Configuring the Trigger System	114
9	Appendix	115
9.1	Additional Procedures	115
9.1.1	Installing a Trigger Extension Bboard	115
9.1.2	Adding an Applet Manually.....	118

9.1.3	Checking on Installed Applet	119
9.1.4	Silent Installation Under Windows	120
9.1.5	Resetting the Global Settings in microDisplay	125
9.1.6	Camera and Topology Configuration	126
9.1.6.1	Starting Link Topology Detection Manually	126
9.1.6.2	Using an External XML File	127
9.1.6.3	Configuring the Program Behavior of the GenICam Explorer at Program Start ...	128
9.1.7	Multi-Board Usage	129
9.2	Trouble-Shooting.....	130
9.2.1	Applet Installed on marathon VF2 Cannot be Loaded into microDisplay	130
9.2.2	Loading a New Applet (Flashing) gets not Completed.....	130
9.2.3	Disabling FPGA Live Reconfiguration	131
9.2.4	Starting Generic Service	132
9.2.5	Topology Mismatch.....	134
9.3	Pin Layout GPIO (On-Board Interface)	135
9.4	Where to Find Further Documentation	137
9.5	Additional Applets and Patches	139
9.6	Support	139

1 Welcome to marathon VF2



Starting with Installation

If you want to start hands-on setting up marathon VF2, skip this introduction and proceed with section [2](#).

1.1 Short Product Overview

1.1.1 Main Features

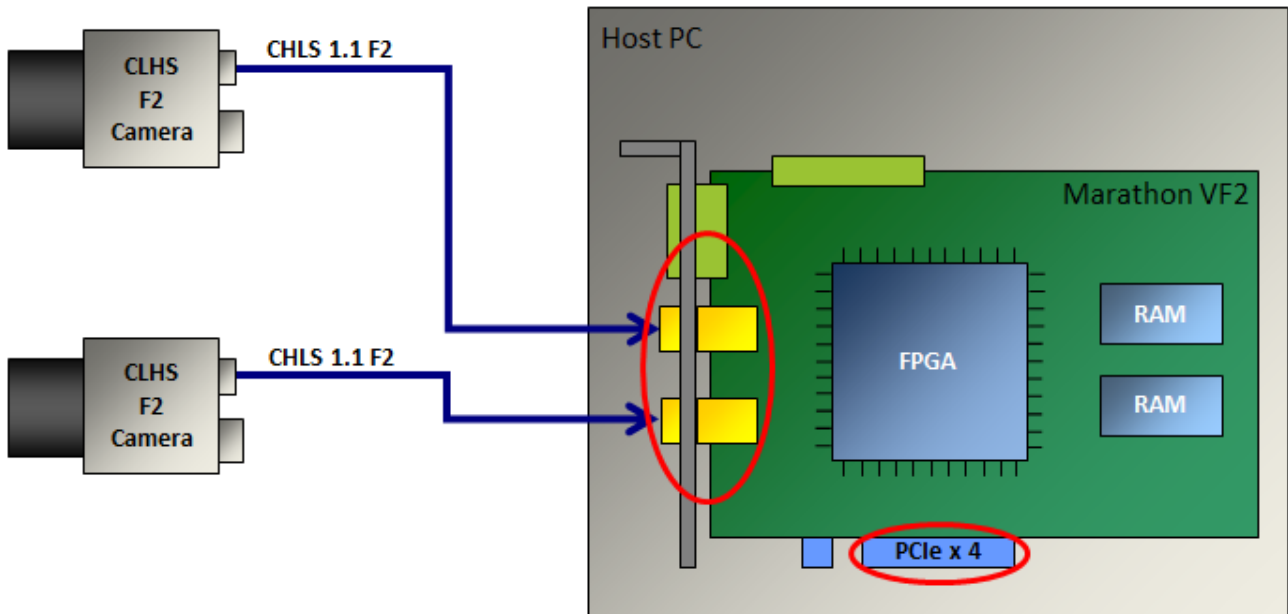
marathon VF2 is a programmable Camera Link HS frame grabber. In addition to the traditional frame grabber functionality, marathon VF2 offers the possibility to be programmed with individual image processing tasks.



Figure 1: microEnable 5 marathon VF2

For connection to the camera(s), marathon VF2 supports the Camera Link HS Standard 1.1 for fiber optic connection (F2) to the camera. Two SFP+ connectors are located directly on the slot bracket.

For connection to the PC, marathon VF2 is plugged into a PCIe 2.x (Gen 2) x 4 slot of your PC.



Technical Data at a Glance

Host Interface	PCIe x4 Gen2 (Direct Memory Access)
Bandwidth (theor.)	2GB/s
Bandwidth (typ./max.)	Up to 1,8 GB/s sustainable data bandwidth
On-board memory	2 GB DDR3-RAM
On-board FPGA processing capabilities	Applets programmed with VisualApplets, Acquisition Applets
Voltage, max. current (actual values depend on processing)	12 V, 1 A
Dimensions	PCIe standard height, half length card: 167.64 mm length x 111.15 mm height
Weight	200g
Camera interface	2 x Camera Link HS 1.1 F2 Standard (via 2 SFP+ Connectors)
Ambient Temperature	50° (0 LFM)** 60° (100 LFM) An adequate airflow in the PC is recommended.
FPGA operating temperature*	0°C to 85°C
Storage temperature	-50°C up to +80°C
Relative humidity	5%-90% non-condensing (operating), 0%-95% (storage)
Conformity	CE pending, RoHS

* Temperature being measured directly on the FPGA; the measured value can be read out in all applets available for marathon VF2 via applet parameter FG_SYSTEMMONITOR_FPGA_TEMPERATURE.

** LFM = Linear Feet per Minute, unit for measuring airflow velocity.


	<p>PCIe Bus Data Throughput</p> <p>The PCIe bus data throughput depends on the motherboard, the chip set, and the BIOS configuration of the host PC. It can differ from slot to slot. It can also depend on the number of installed PCIe boards, e.g., a PCIe x8 connector may support only x4 performance.</p> <p>Always check the mainboard manual thoroughly to be sure.</p>
---	--

Image Processing

marathon VF2 can be programmed to fulfill highly specific image processing tasks that are required by a specific application. The programming can be done by yourself using the easy-to-use graphical FPGA programming environment *VisualApplets*® (see section [Z](#) for details.), or by Silicon Software / a certified partner.

Production Line

Using marathon VF2, the host PC can be located in a protected area, away from the actual manufacturing plant. The possible distances depend on the fiber cables you use for connecting the camera with marathon VF2:

- ◆ With F2 Multi Mode Fiber cables: up to **300 m**

For integration into the production line, marathon VF2 offers several general purpose inputs and outputs for PLC and multi-board synchronization.

This allows for

- ◆ controlling peripheral devices by sending trigger signals (such as lighting, camera...),
- ◆ receiving various trigger signals from peripheral devices (such as shaft encoder, light barrier, ...), and
- ◆ synchronizing connected devices and/or other marathon VF2 devices in a daisy chain.

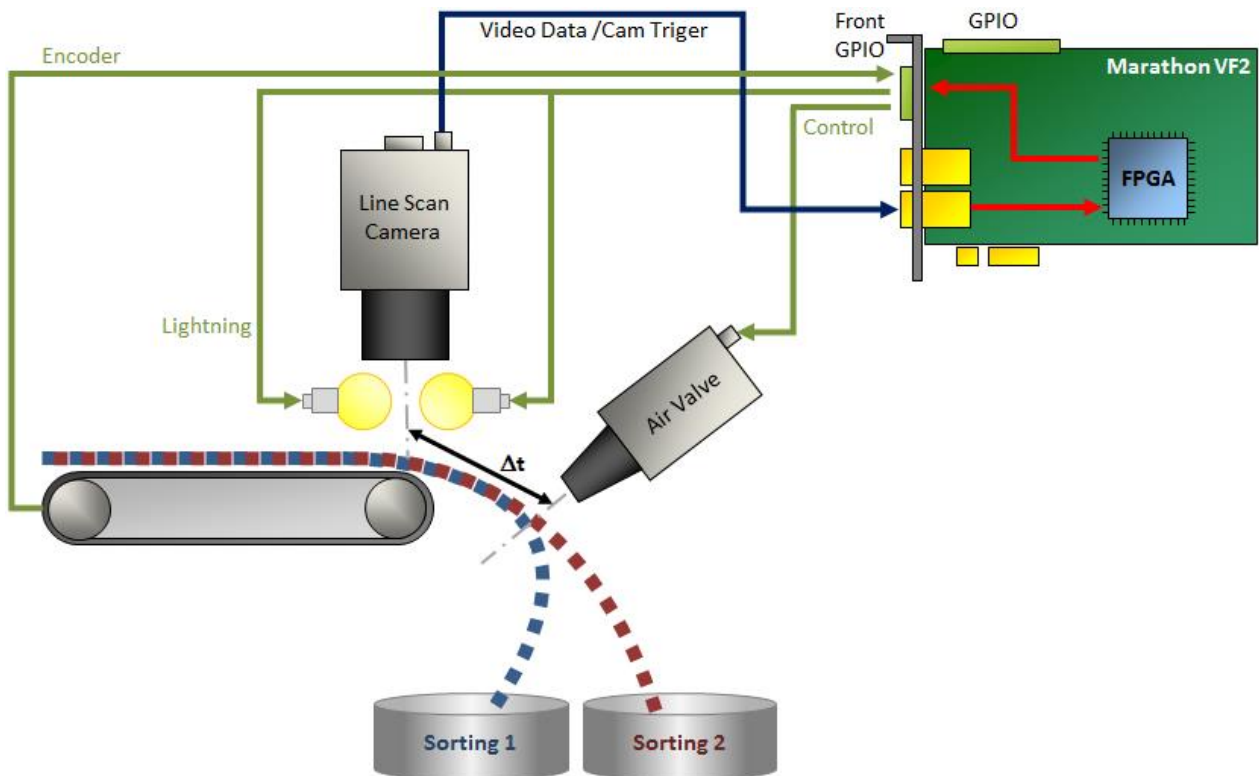


Figure 2: marathon VF2 within the production line (example: bulk sorting)

Software Programming and Configuration Interface

marathon VF2 offers an easy-to-use, C-based Software Development Kit, the standard Silicon Software SDK. The SDK allows for easy integration into any image processing software. For details on the marathon VF2 SDK programming interface, see the [SDK online documentation on the Silicon Software website](#).

1.1.2 Ports and LEDs

Ports

On marathon VF2, you have the following ports:

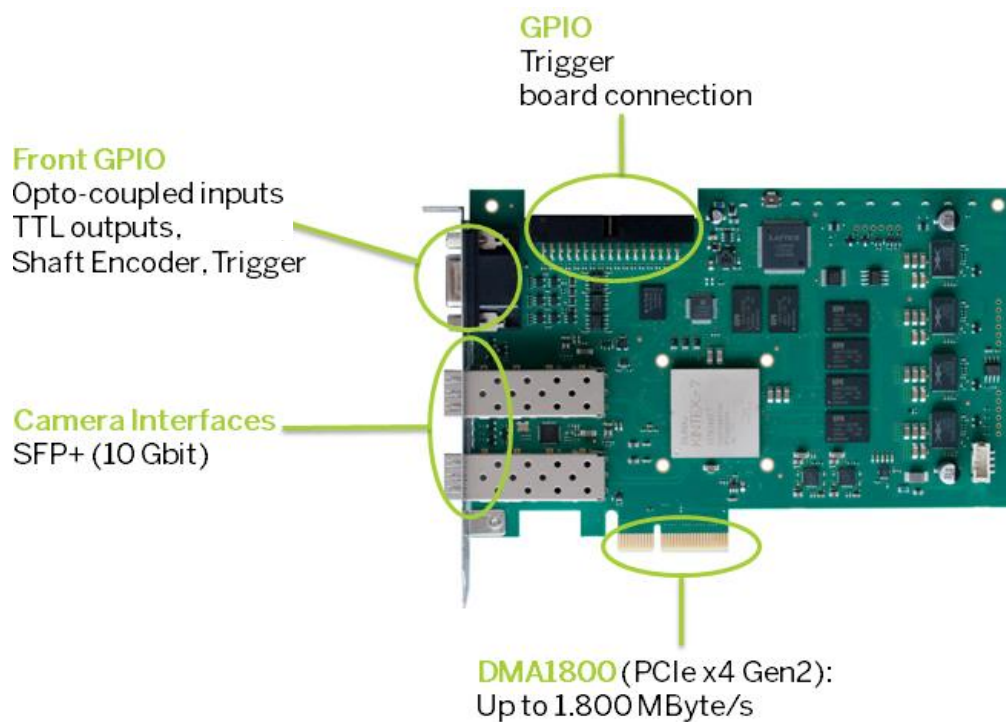


Figure 3: marathon VF2 Ports

LEDs

On the marathon VF2 slot bracket, you see 6 LEDs, three for each CLHS connector:



Figure 4: marathon VF2 slot bracket

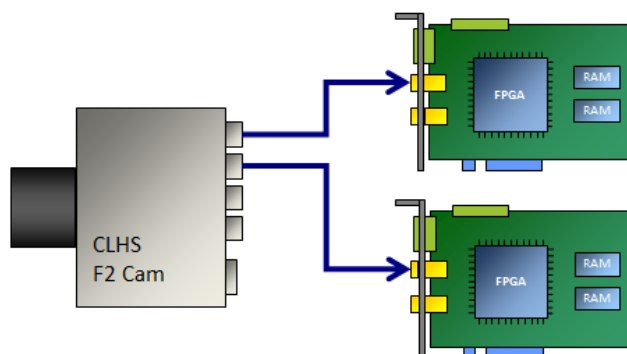
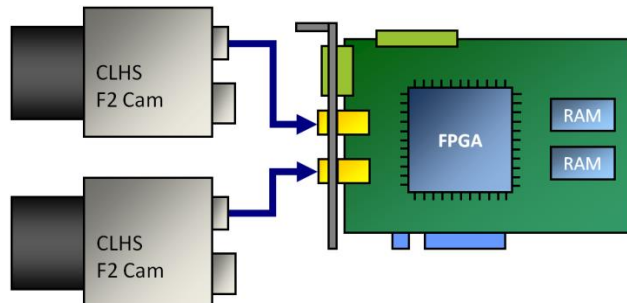
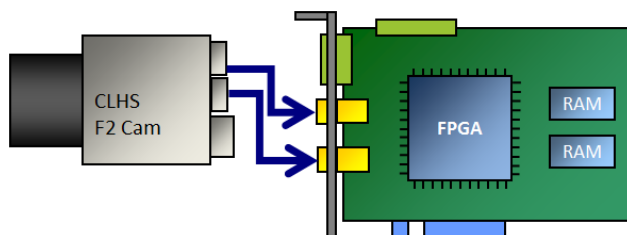
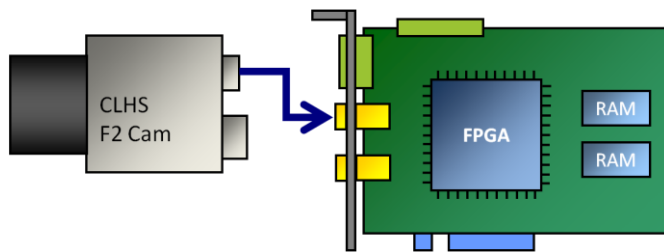
The state of the LEDs (off, blinking fast or medium speed, or constantly on) indicate the following conditions:

State of LED	Description	Blinking Speed	State Name
All LEDs off	Device is not powered and/or is waiting for software(Service is deactivated);	0 Hz	NON_OP
Constant green	Link established and ready for data transfer.	0 Hz	LINK_GOOD
Blinking green medium speed	Hardware is fine, but connection not established or recently broken.	2 Hz	LOOKING_FOR_LINK
Constant orange	The frame grabber doesn't detect camera (camera is off or in reset) -> receive channel (from camera to frame grabber) is dead or sends no revision messages.	0 Hz	FAR_END_RESET

1.1.3 Possible Topologies

You can operate marathon VF2 as follows:

- ◆ 1 camera and 1 marathon VF2
- ◆ 2 cameras and 1 marathon VF2
- ◆ 1 camera and 2 marathon VF2



1.1.4 Available Product Variants

marathon AF2 (A-Series)

marathon AF2 is a plug-and-play frame grabber. You simply need to load the applet supporting your camera topology. Configuration is possible via the additional tools microDisplay and microDiagnostics, or via SDK.

marathon VF2 (V-Series)

marathon VF2 offers all the functions provided by marathon AF2. In addition, marathon VF2 can be programmed to fulfill highly specific, complex image processing tasks that are required by a specific application. The programming can be done either by Silicon Software / a certified partner, or by yourself using the easy-to-use graphical FPGA programming environment *VisualApplets*®.

1.2 Delivery Contents

1.2.1 Standard Delivery Contents

If you ordered marathon VF2, the following items are contained in your delivery package:

- ◆ microEnable 5 marathon VF2 (order number 150721)

1.2.2 Optional Add-Ons (ask for availability)

Optionally, you may have ordered one of the following parts:

- ◆ External trigger boards:
 - ◆ [I/O Opto-Coupled Trigger Board](#)
 - ◆ [I/O TTL Trigger Board](#)

1.3 Requirements

1.3.1 System Requirements

Operating System on host PC: You can use marathon VF2 with a PC that has one of the following operating systems installed:

- ◆ Windows® 7 32bit/64bit
- ◆ Windows® 8 32bit/64bit
- ◆ Linux 32bit/64bit (Ubuntu 14.04 LTS or Red Hat Enterprise 7.x)¹

PCIe interface on host PC: The host PC needs to have one PCIe (Gen.2) x4 slot available.

1.3.2 Software and Environment

Silicon Software Runtime on host PC: To use marathon VF2, you need to install the Silicon Software Runtime Software (version 5.4.2 or higher) on the host PC. The Runtime Software installer is available as download on the download section of the Silicon Software website. The runtime installation contains, amongst other things, the required driver, additional tools, SDK, documentation, and SDK examples. For installation instructions, see section [3 Software Installation](#).

Applet on marathon VF2: When you start to adapt the system to your specific image acquisition requirements, you need to install an applet on marathon VF2.

For installation, only very few steps are required:

- ◆ Install the runtime on the host PC (see section [3](#))
- ◆ Copy the applet(s) you developed in VisualApplets to the host PC (see section [4.3](#))
- ◆ Install one applet on marathon VF2 (flashing) (see [4.4](#) Installing an Applet onto marathon VF2 (Flashing))

¹ All other Linux distributions may require additional configuration work by the user and can only be supported by Silicon Software to a limited extent.

The following figure shows which software/firmware modules are installed on PC and marathon VF2:

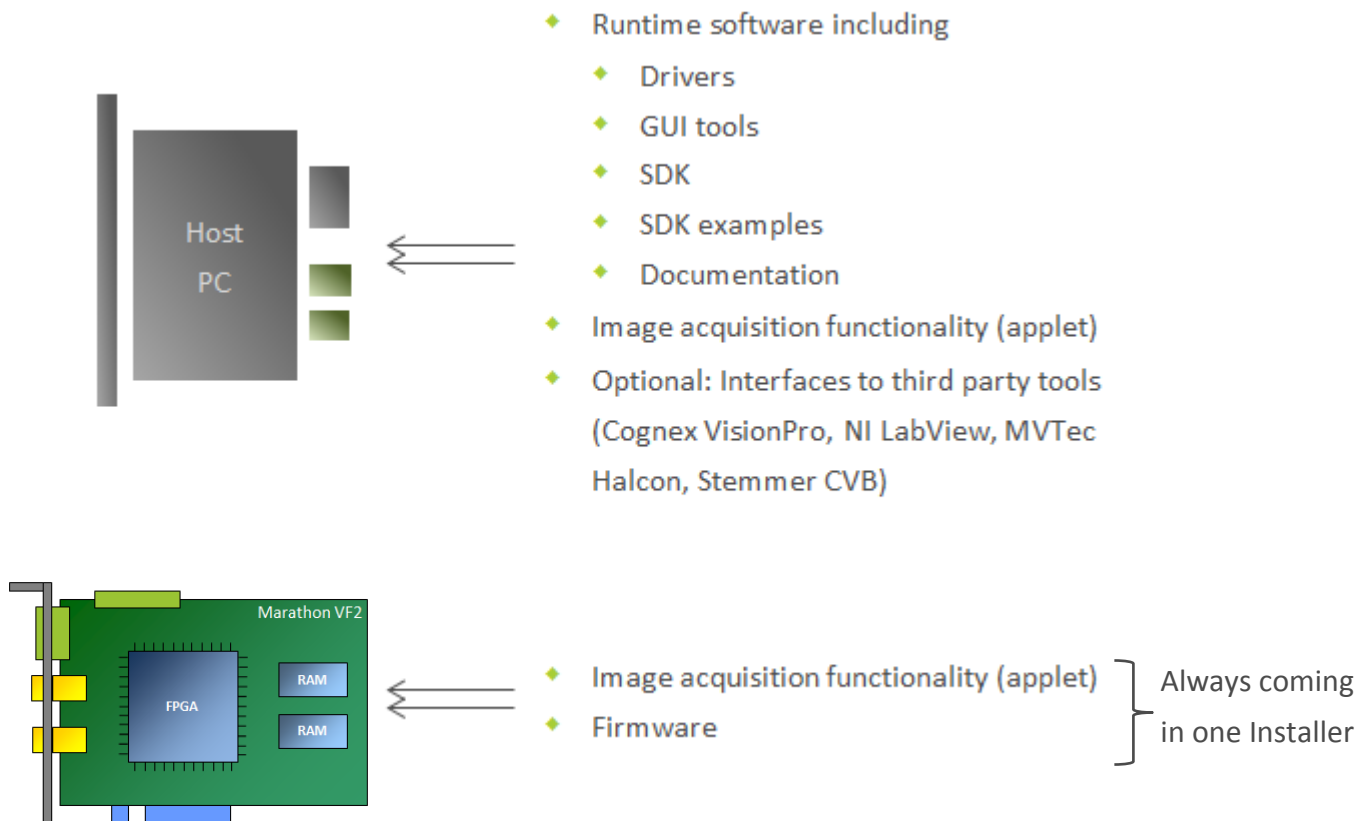


Figure 5: Software and firmware modules on individual devices

2 Hardware Installation

2.1 Connecting marathon VF2 Physically

2.1.1 Requirements

- ◆ Host PC running one of these operating systems:
 - Windows® 7 32bit/64bit, or
 - Windows® 8 32bit/64bit
 - Linux 32bit/64bit (Ubuntu 14.04 LTS²)
- ◆ One PCIe 2 (Gen. 2) x4 slot on the host PC (with at least 4 wired PCIe lanes to achieve the full performance of marathon VF2; refer your mainboard manual for detailed information).
- ◆ SFP+ transceiver(s) for fiber optical data (see Figure 6) – one transceiver per connected camera port
- ◆ Fiber optic cable(s) F2 Multi Mode for distances of up to 300 m between camera and host PC – one cable per connected camera port
- ◆ Camera that is compatible to Camera Link HS specification 1.1 F2




Figure 6: SFP+ Transceiver



Figure 7: Fiber Optic Cable

² All other Linux distributions may require additional configuration work by the user and can only be supported by Silicon Software to a limited extent.

2.1.2 Installing the Board

	<p>Caution</p> <p>Before installing hardware, ensure that</p> <ul style="list-style-type: none">• the system power is OFF and unplugged from the power outlet,• proper electrical grounding procedures have been followed.
---	--

To install the frame grabber hardware:

1. Shut down your computer.
2. Unplug your computer from the power outlet.
3. Plug your marathon VF2 frame grabber into a free PCIe 2.x (Gen2) x4 of your PC.

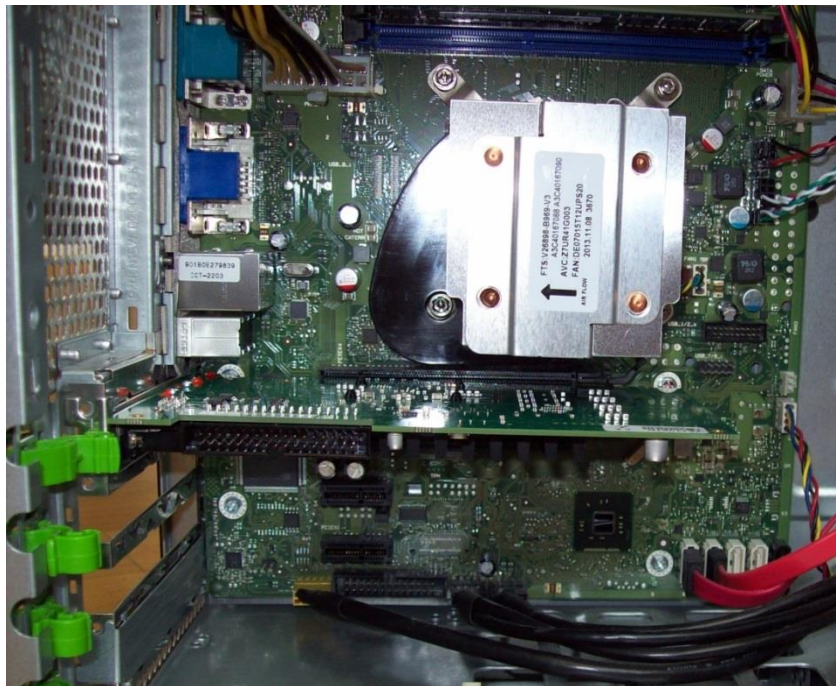


Figure 8: Plugged frame grabber board within a PC

Caution

Make sure you use an adequate ventilation system within your computer.

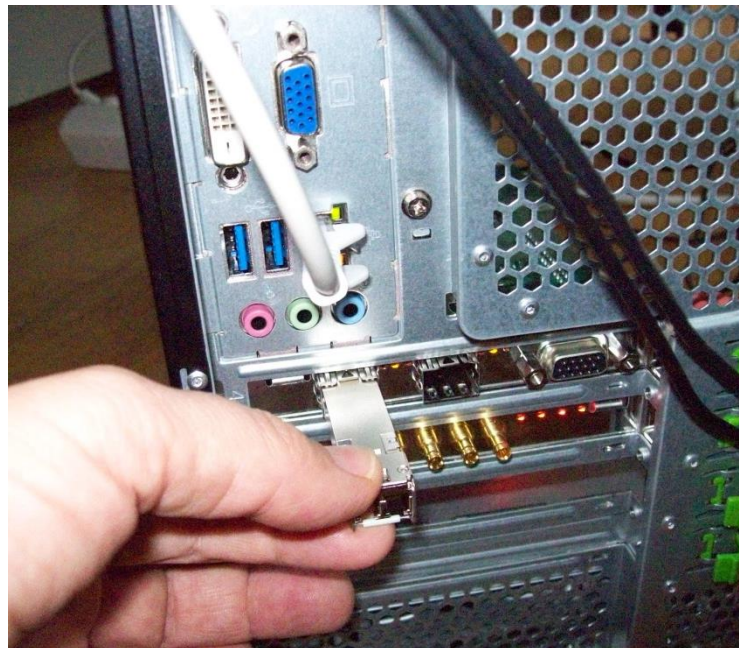
This is of special importance if



- there is little space between boards in a multi board installation,
- an installation is close to a graphics card.

We recommend installing a fan and leaving enough free space between boards.

4. Insert the optical SFP+ transceiver into the empty port (cage) on the slot bracket of marathon VF2. Make sure that it clicks firmly into place.



5. If you are going to use both ports, repeat the last step for the second port (for topology options, see section [1.1.3](#)).
6. Boot the system.

7. After booting, marathon VF2 is recognized in the *Windows Device Manager* as a *Multifunction adapter*:

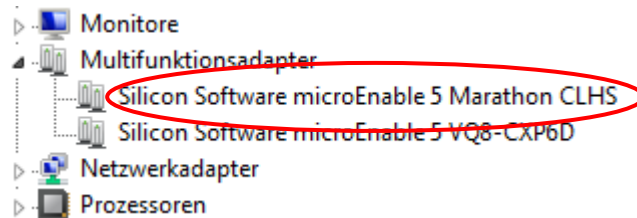


Figure 9: marathon VF2 in *Windows Device Manager*

8. There are two ways the frame grabber may be displayed under *Multifunction adapters*.

If the frame grabber is displayed

- as *Unknown device*: Proceed as described in section [3.1 Installing the Runtime Software](#) to install runtime 5.4.2 (or higher). The relevant driver will be installed together with the runtime.
- As *Silicon Software microEnable 5 marathon CLHS*:
 - Make sure runtime 5.4.2 (or higher) is already installed on your system (otherwise, proceed with section [3.1 Installing the Runtime Software](#)).
 - Make sure the version of the frame grabber's driver is the same as the one available in the installation folder of the runtime. If not, update the driver with the driver you find in the installation folder of the Silicon Software runtime (\SiliconSoftware\Runtime5.4.x\drivers\me5).

3 Software Installation

3.1 Installing the Runtime Software under Windows

The *Runtime Installer* is available on the download section of the Silicon Software website. The runtime software needs to be installed on the host PC in order to run marathon VF2.



Silent Installation

If you prefer silent installation, see section [9.1.4](#).




Installing under Linux

For runtime installation under a Linux operating system, refer to our [Installation Guide for installing the Silicon Software Runtime under Linux](#) on the Silicon Software website.

To install the runtime:

1. On the host PC, uninstall all Silicon Software runtime versions prior to the version you are going to install. Make sure you also delete all related subfolders in the Silicon Software installation folder.
2. Download the *Runtime Installer* from the download section of the Silicon Software website (version 5.4.2 or higher). If you have no restraints as to the size of the download package, just use the standard installer. The installer file has, depending on the operating system you are using, the following name:
 - a. RuntimeAppletsSetup_v5.4.x_Win64.exe
 - b. RuntimeAppletsSetup_v5.4.x_Win32.exe



Small Download Packages Available

If you want to download only small download packages, use the runtime stand-alone installer. The installer file has, depending on the operating system you are using, the following name:

- ◆ RuntimeSetup_v5.4.x_Win64.exe
- ◆ RuntimeSetup_v5.4.x_Win32.exe

3. Start the runtime installer by double-clicking on the file name RuntimeAppletsSetup_v5.4.x_Winxx.exe.
4. Follow the instructions of the installation wizard until the following window is displayed:

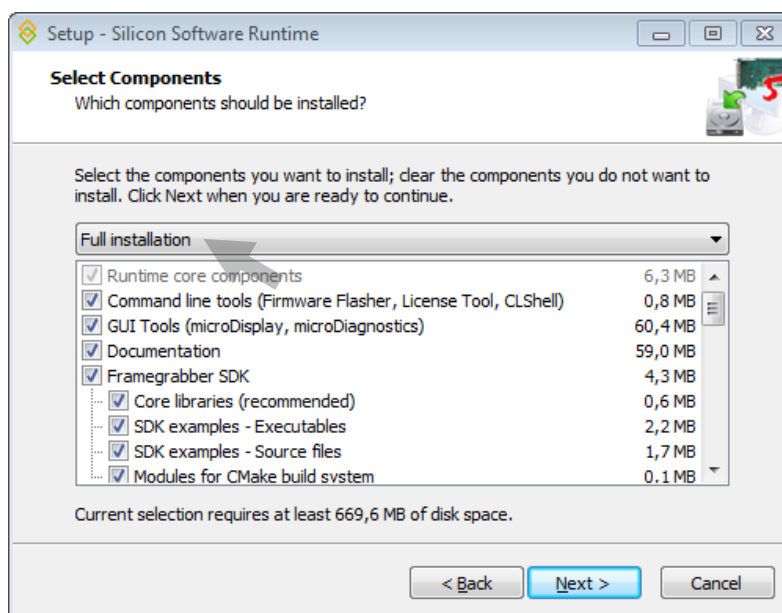


Figure 10: Installation Wizard with Default Option "Full installation"

5. **Full Installation** is the default option. If you want to save disk space, you can de-select the *Acquisition Applets* and *Advanced Acquisition Applets* that are not necessary for your application (i.e., applets for other frame grabbers than marathon VF2).

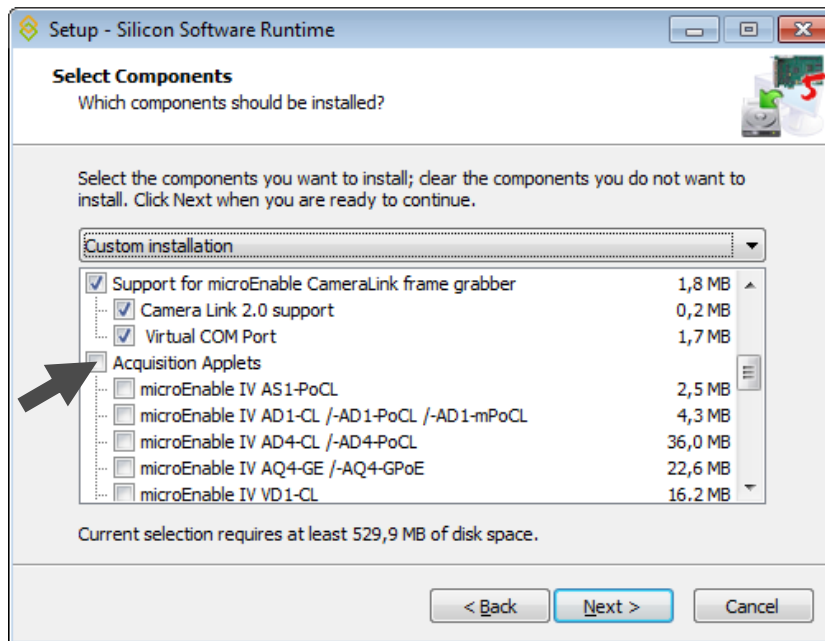


Figure 11: Custom Installation

6. Klick **Next** and follow the instructions of the installer.

The runtime software, the drivers, and all related software tools will be installed on your system now.

If you are asked for driver installations:

Make sure you do install the driver for microEnable 5.

Installing the driver for microEnable IV is optional.

7. If you are asked to re-start your PC system, do so to complete installation.

After installing the runtime software, marathon VF2 is recognized in the *Windows Device Manager* under *Multifunction adapters* with its full device name.

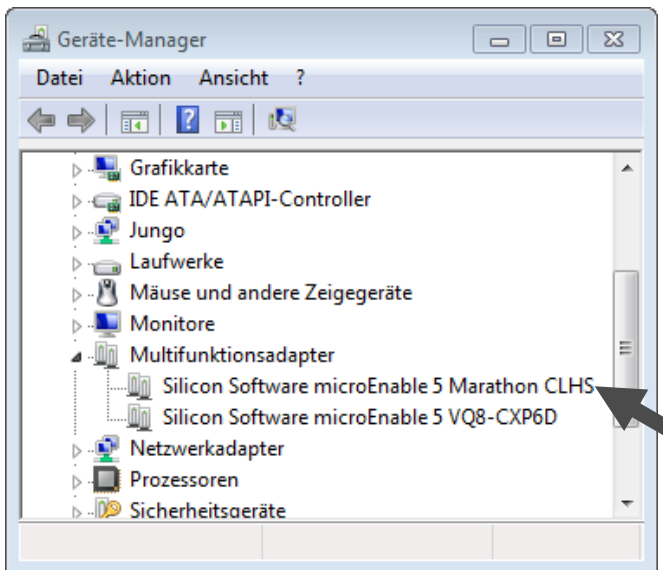


Figure 12: marathon VF2 as displayed in *Windows Device Manager* after runtime installation

3.2 Installing the Runtime Software under Linux

For runtime installation under a Linux operating system, refer to our [Installation Guide for installing the Silicon Software Runtime under Linux](#) on the Silicon Software [Live Documentation](#) site.

4 Using marathon VF2

4.1 Connecting to the Camera

1. Plug the optical SFP+ transceiver(s) into the camera port(s) on the slot bracket of your marathon VF2.
2. Plug the fiber-optic cable into the optical SFP+ transceiver on the slot bracket:

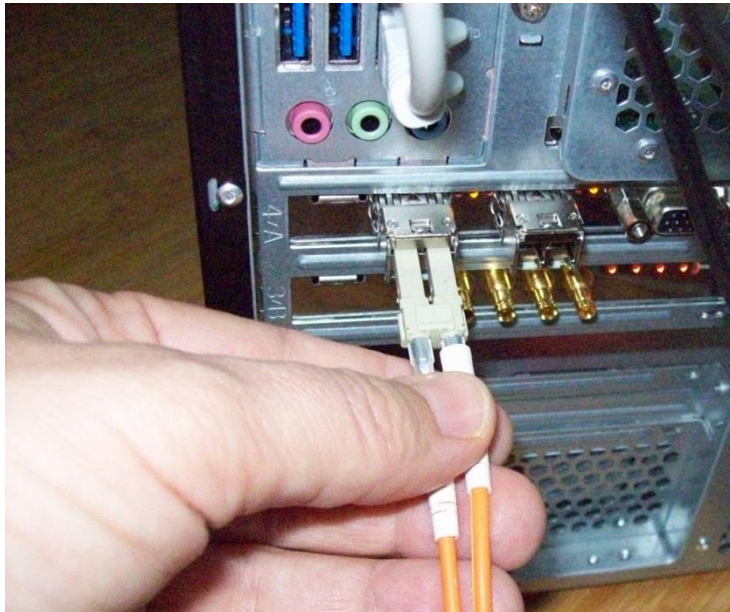
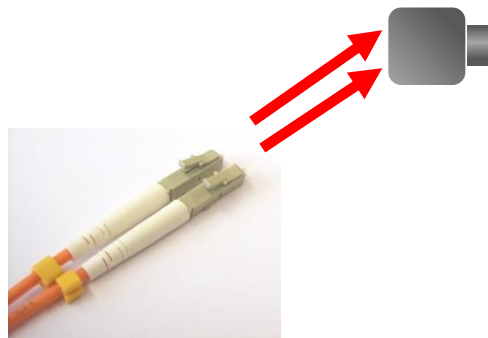


Figure 13: Inserting fiber optic cable into SFP+ connector

3. Plug the fiber-optic cable into the according camera cages of a CLHS Rev.1.1 F2 compatible camera.
4. Power the camera.
5. Start the camera.



4.2 Configuring the Camera

4.2.1 Autodiscovery

The Silicon Software marathon VF2 frame grabber offers the GenICam interface and comes together with a graphical tool for accessing this interface: the *GenICam Explorer*.

Using the *GenICam Explorer*, you can

- ◆ configure and control the camera connection.
- ◆ configure the camera.

The tool *GenICam Explorer* comes as part of the runtime installation. The *GenICam Explorer* discovers connected cameras automatically and provides direct access to the GenICam interface of the camera.

To connect the camera to the frame grabber:

1. Start the camera.
2. Open the *GenICam Explorer* (Start -> All Programs -> Silicon Software -> Runtime 5.4.x -> *GenICam Explorer*).


On program start:


- ◆ The start window of the *GenICam Explorer* opens.
- ◆ The *GenICam Explorer* starts the automatic camera discovery.
- ◆ The *GenICam Explorer* connects to the discovered camera.



Generic Service must be started

Generic service must be started for camera detection. If you decided during installation to disable the generic service, enable it now (see section [9.2.4](#)).

	<p>Important</p> <p>You can define if you want the GenICam Explorer to take all these steps automatically (default). If you prefer user interaction, you can configure the program behavior, see section 9.1.6.3 Configuring the Program Behavior of the GenICam Explorer at Program Start.</p>
---	--

	<p>Starting Full Discovery Manually</p> <p>If the camera cannot be discovered on starting the GenICam Explorer, you can start the discovery manually, see section 9.1.6.1 Starting Link Topology Detection Manually.</p>
---	---

You see the current status of the camera discovery and of the connecting process in the task bar:

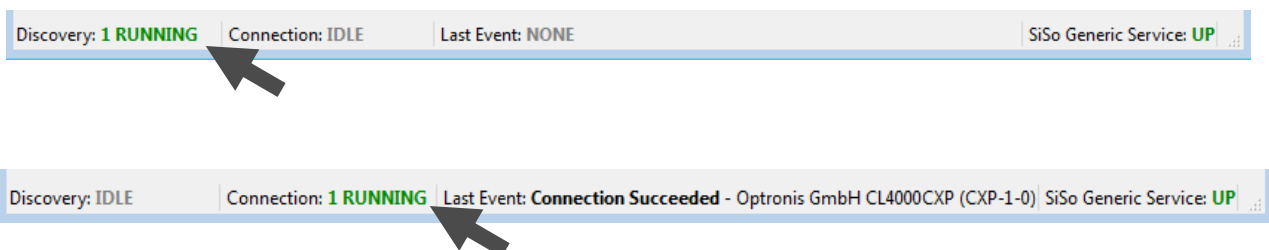


Figure 14: Taskbar of Program Window during automatic Camera Discovery and automatic Camera Connect

After successful camera discovery, information on the detected camera is displayed:

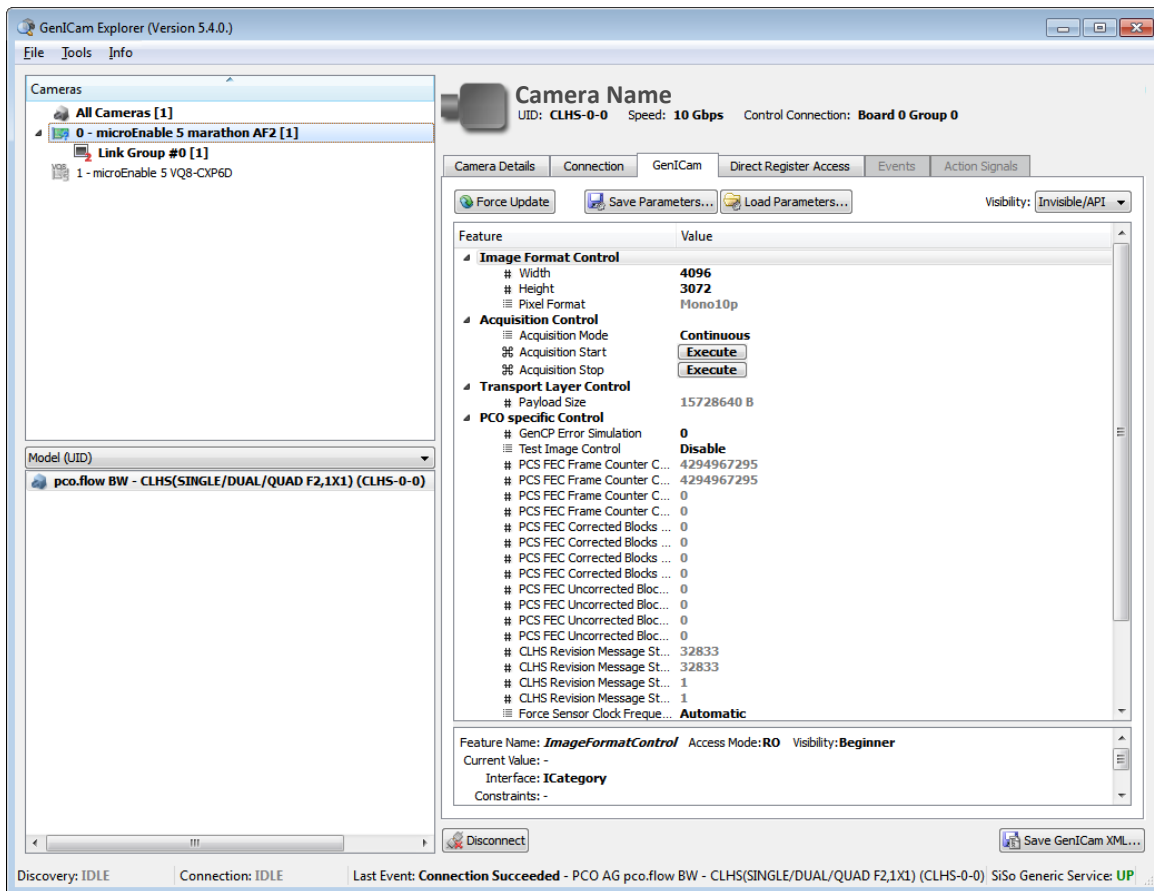
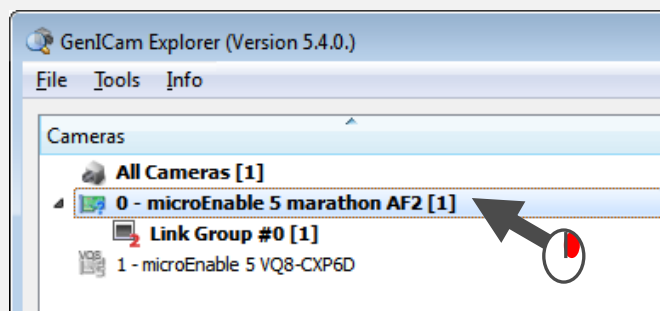


Figure 15: After successful camera discovery, camera information is displayed

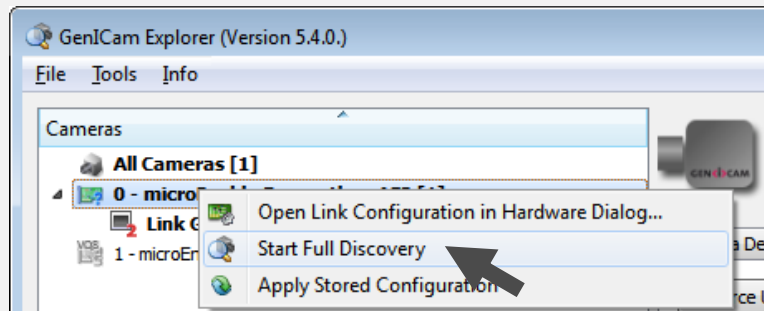
Camera Name Displays in Red

If the camera name is displayed in red:

1. Right-click on the board name in the left upper corner.



2. From the context menu, select *Start Full Discovery*.



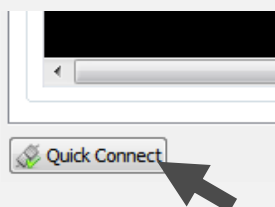
3. If more than one cameras are connected, you can select the camera in the left bottom panel *Model UID*.

4.2.2 Configuring the Camera

Prerequisites: The camera is started. The GenICam Explorer is started. On program start, the camera has been successfully discovered and connected.

Connecting the Camera Manually

If the camera is not connected yet, go to the *Connection* tab and click the **Quick Connect** button.




Using External XML Files

If you want to use an external configuration file for setting the camera parameters, see section [9.1.6.2 Using an External XML File](#).

To configure the camera:

1. In the GenICam Explorer, go to the **GenICam** tab.

The parameters of the camera's GenICam interface are displayed directly after connecting to the camera. The values that are currently set are displayed.



Parameter Set

Which parameters are displayed depends on the GenICam interface of the camera you are using.

2. Adapt the settings of the camera's GenICam parameters to your needs.
In the **Value** column, type in or select the new value.
3. Scroll down to access all parameters.

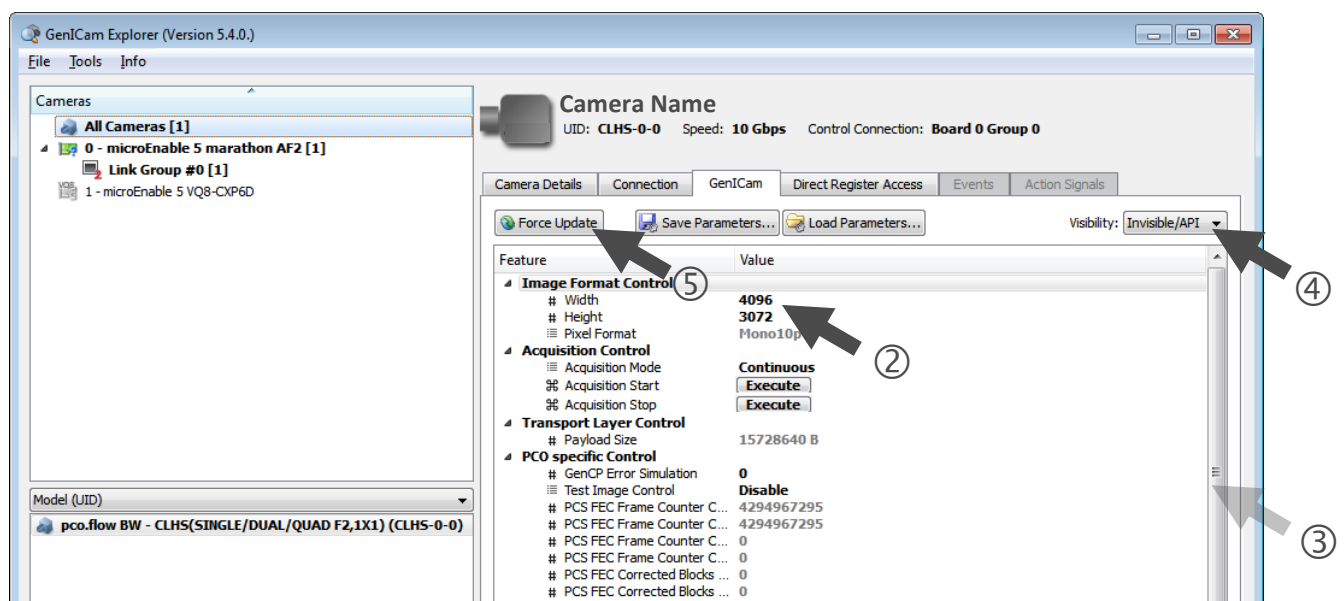
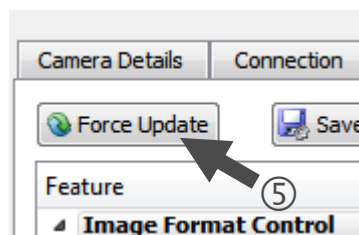


Figure 16: Changing parameter values under *Value*

4. Change the Visibility status to the value that let's you see all the parameters you want to access (this depends on the camera-specific GenICam implementation).

To see your changes after modifying parameter values:

5. Click on the **Force Update** button. The display will be updated immediately.



Writing Data Directly into the Camera

During image acquisition, the camera will use the settings you define here, since you are writing the data directly into the camera.

4.3 Preparing the File System

Before you can flash an applet you created in VisualApplets onto marathon VF2, you have to copy the applet (*.hap file) to the host PC.

To prepare your file system for flashing:

1. Go into the runtime installation directory and here into the subdirectory *Hardware Applets*.
2. Create here the following directory:
[runtime installation directory]/Hardware Applets/**mE5-MA-VF2**
3. Copy the applet (*.hap file) you created in VisualApplets into the new directory **mE5-MA-VF2**.

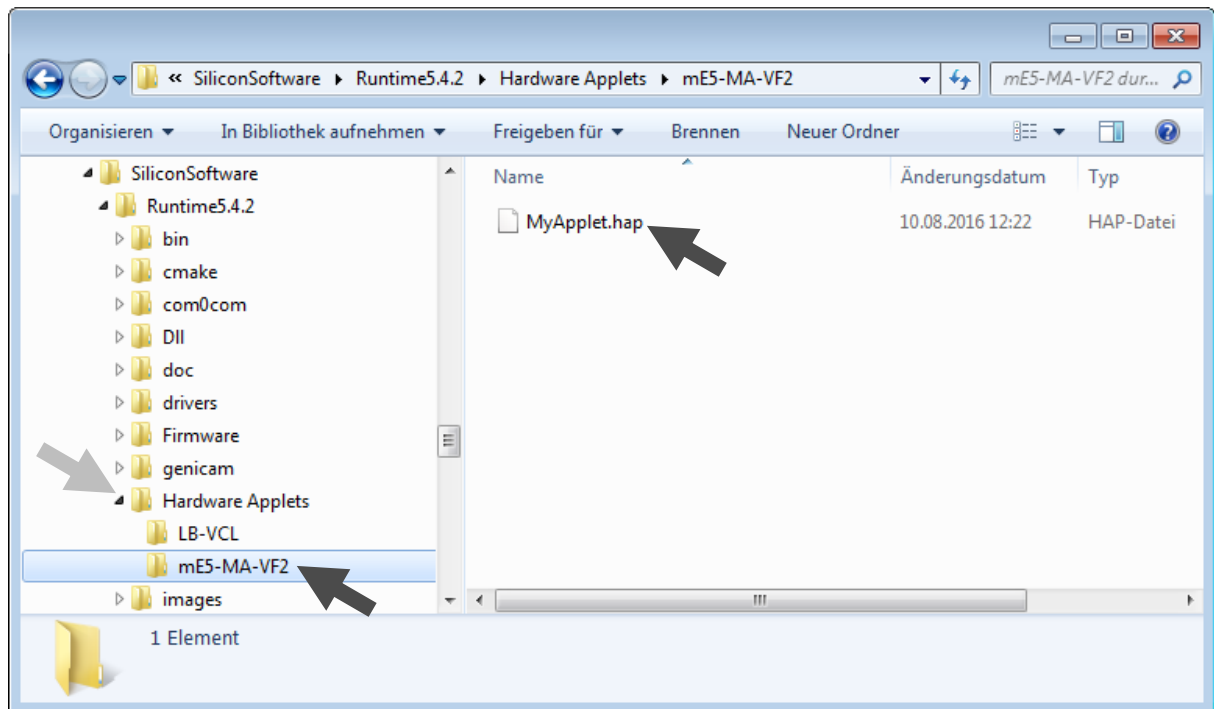


Figure 17: Copying an applet (*.hap) into the installation directory

4.4 Installing an Applet onto marathon VF2 (Flashing)



Adding an Applet Manually

If you got a specific applet not in form of an installer, but as a single *.hap or *.dll file, you have to copy it to your host file system before you can flash it onto marathon VF2. For instructions on how to do this, refer to section [9.1.2](#).

To flash an applet onto marathon VF2:

1. Start the tool microDiagnostics (e.g., via Windows Start -> Programms -> SiliconSoftware-> Runtime5.4.x -> microDiagnostics, or directly out of the installation directory ([runtime installation directory]/bin/microDiagnostics.exe)

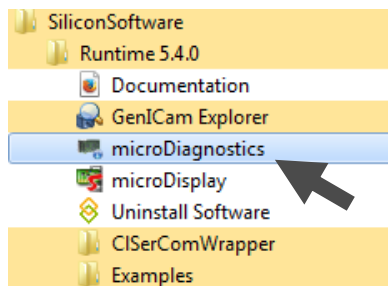


Figure 18: Starting microDiagnostics

In the microDiagnostics program window that opens:

2. Select the marathon VF2 device you want to install the applet on.

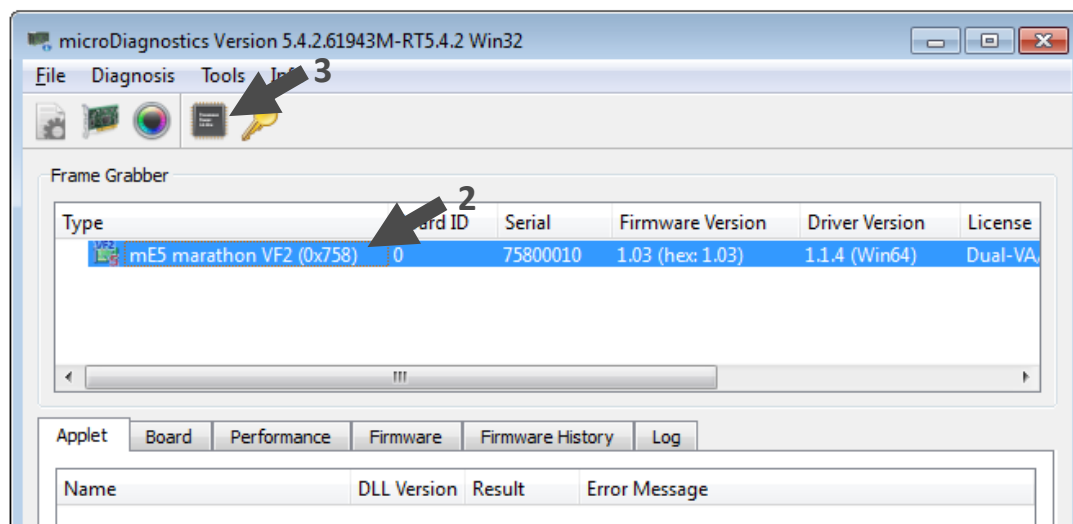
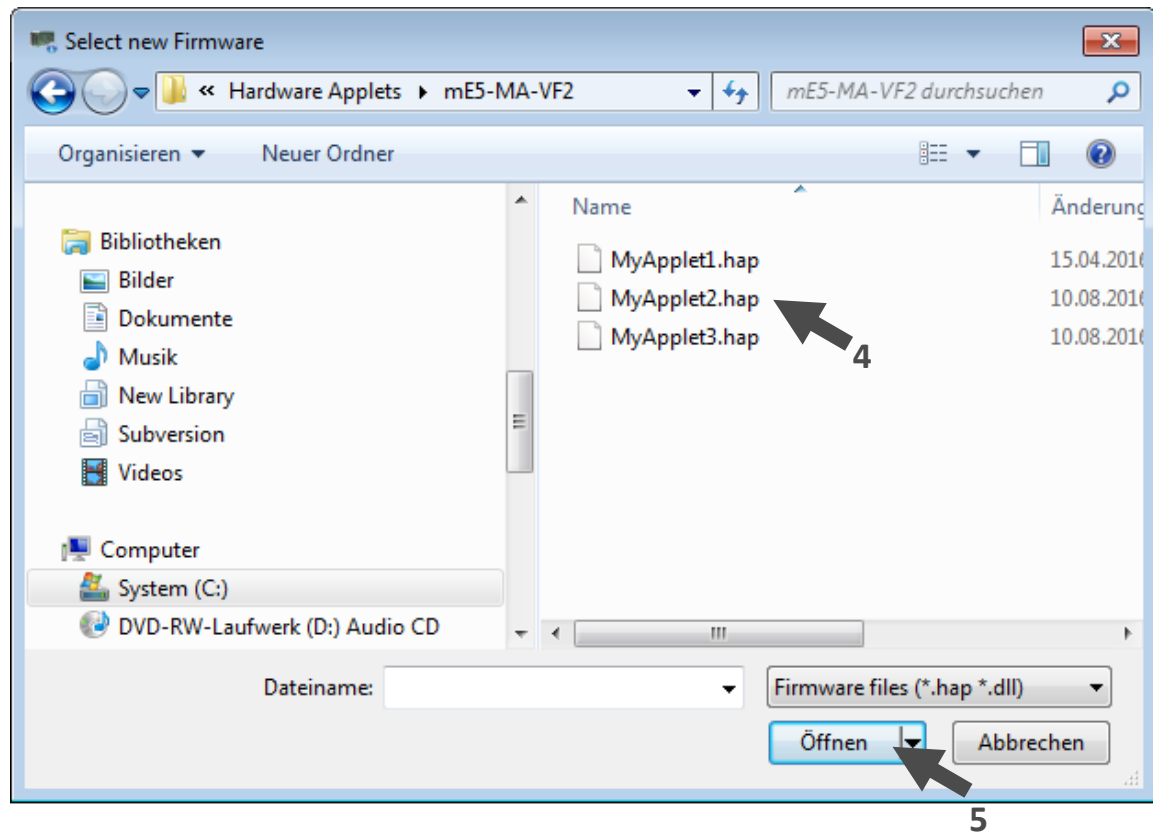


Figure 19: Start window of microDiagnostics

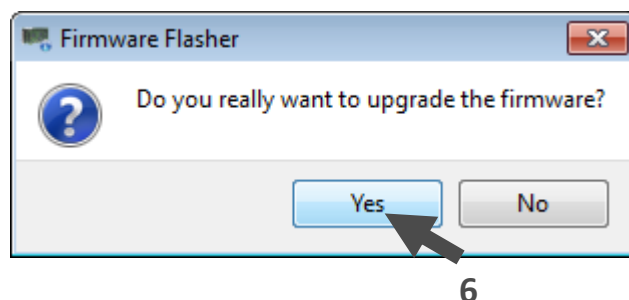
3. Click the button **Flash Board(s)**.



The program leads you to a file selection dialog:



4. Select the applet (*.hap file) you need by clicking on the file name.
5. Click on **Open**.
6. Confirm by clicking on **Yes**.



The flashing process gets started now:

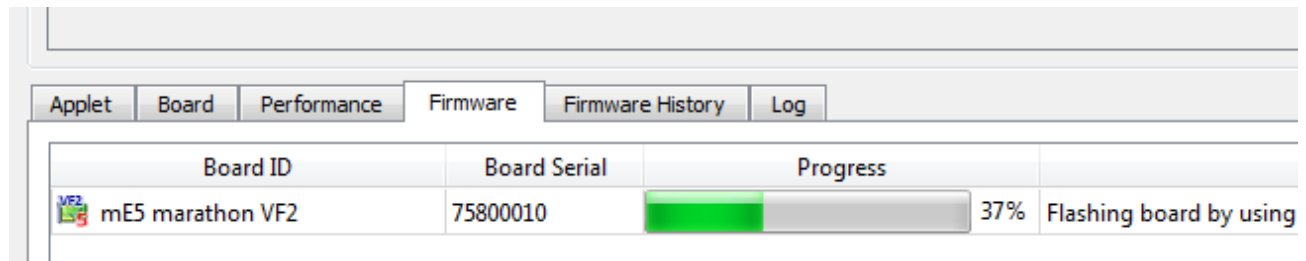


Figure 21: Flashing in progress – as displayed in microDiagnostics



Attention

If you experience any problems, keep marathon VF2 powered and call the Silicon Software Support department.

7. Wait until the new applet is completely installed.



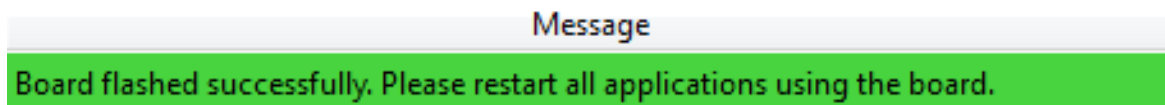
Windows or Linux

The following behaviour of the system depends on the operating system you use on the host PC.

- ◆ If you are using Windows, just follow the instructions below.
- ◆ If you are using Linux, the system will behave as described on page [39](#) (complete shutdown and restart required), since the FPGA live configuration has not been implemented for Linux.


With a host PC that supports live reconfiguration³ of the FPGA (see chipset lists below), you get one of the following messages (A, B or C) after flashing:

A) Restarting applications:



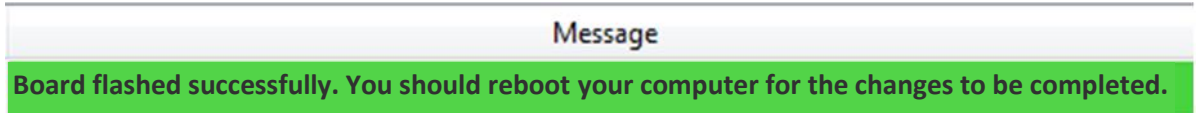
- Close all applications of the Silicon Software runtime environment (microDisplay, microDiagnostics, GenICam Explorer).

Now, the applet is successfully flashed onto marathon. After re-opening, the runtime programs will provide access to the applet you just flashed onto marathon. You can continue with configuring the applet via microDisplay (see section [4.4](#)) or via SDK.

	<p>PCs supporting Live Reconfiguration, Behavior A</p> <p>High-end mainboards with server level chipset support this behavior. The following chipsets are known to support FPGA live configuration without reboot in some mainboards:</p> <ul style="list-style-type: none"> ◆ Intel® X58 Express ◆ Intel® X79 Express ◆ Intel® Z87 (Z99) <p>We do not guarantee behaviour A with the aforementioned chipsets.</p>
---	--

³ FPGA live configuration is only available under Windows; FPGA live configuration can be disabled by user, see section [9.2.3](#).

B) Optionally rebooting host PC:




You will get this message if marathon has been flashed successfully, but the PCI speed is reduced⁴ after flashing.

To get the full PCI bandwidth⁵, you need to reboot your host PC:

➔ Reboot your host PC to get the full PCI bandwidth.

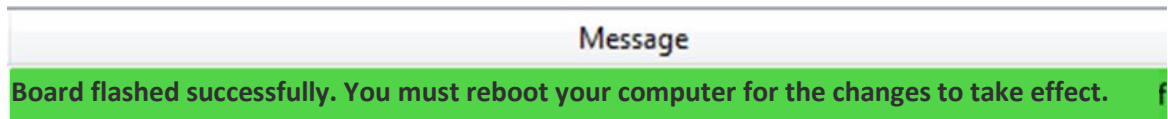
After re-boot, you get the full bandwidth on the PCI interface. The runtime programs will provide access to the applet you just flashed onto marathon. You can continue with configuring the applet via microDisplay (see section [4.4](#)) or via SDK.

	<p>PCs supporting Live Reconfiguration, Behavior B</p> <p>High-end mainboards with server level chipset support this behavior. The following chipsets are known to support FPGA live configuration without reboot in some mainboards:</p> <ul style="list-style-type: none"> ◆ Intel® X58 Express ◆ Intel® X79 Express ◆ Intel® Z87 (Z99) <p>We do not guarantee behaviour B with the aforementioned chipsets.</p>
---	--

⁴ PCI may be reduced from generation 2 to generation 1, or the link width changes (e.g., 8 to 4, 4 to 1,).


⁵ **Reboot not always required:** If you intentionally flashed an applet that reduces the PCI speed (in comparison to the applet that has been on marathon before flashing), you do not need to reboot your host PC.

C) Rebooting host PC:



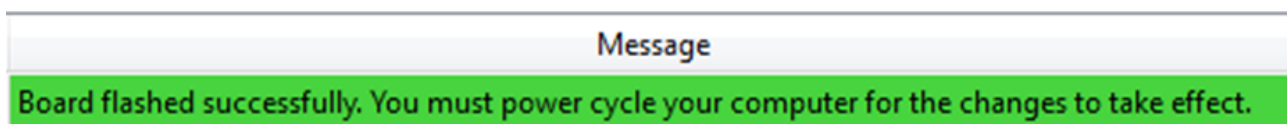
➔ Reboot your host PC.

Now, the applet is successfully flashed onto marathon. After re-boot, the runtime programs will provide access to the applet you just flashed onto marathon. You can continue with configuring the applet via microDisplay (see section [4.4](#)) or via SDK.

	<p>PCs Requiring Reboot, Behavior C</p> <p>The following chipsets are known to require a reboot after FPGA configuration:</p> <ul style="list-style-type: none"> ◆ Intel® C220 ◆ Intel® H87
---	--

Linux – Behavior after Flashing

Under Linux, after each flashing, you will now be prompted to power cycle (cold-boot) your host PC:



To power cycle your host PC:

1. Click **Shut Down** to shut down your host PC completely (the Restart option is not enough).
2. After the computer is completely off, wait for some seconds.
3. Start the host PC again.



Complete Shut Down Essential

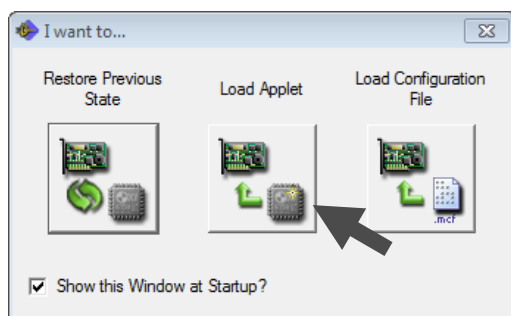
For power cycling, it is not enough use the *Restart* option. Complete shut down and following new start are essential when you need to power cycle your host PC.

4.5 Basic Steps

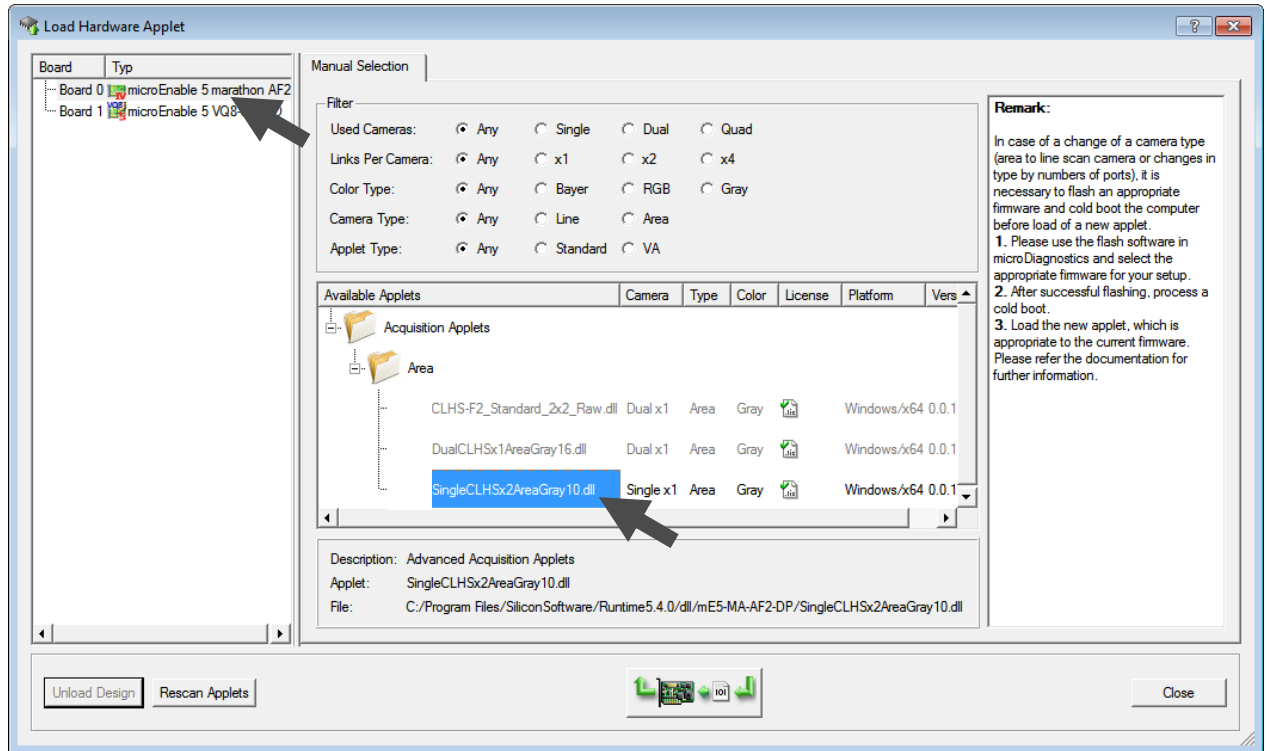
Before you start using your marathon VF2 applet in your own Software (via SDK), you might want to test the system set-up or the individual applet and its parameterization options. To do so, you use the tool *microDisplay* that has been installed on your host PC as part of the runtime environment.

To start image acquisition for test purposes:

1. Power and start the camera.
2. Start the tool *microDisplay*, e.g., by
 - ◆ clicking on START -> All programs -> SiliconSoftware -> RT 5.x.x -> *microDisplay*, or
 - ◆ directly out of the installation directory ([runtime installation directory]/bin/*microDiagnostics.exe*).
3. In the start dialog *I want to...*, select **Load Applet**.



The *Load Hardware Applet* dialog opens:



To load the hardware applet into microDisplay:

4. In the dialog *Load Hardware Applet* under *Board* (left upper corner), select the marathon VF2 device you want to use.

A list of applets is displayed. Only one applet is clickable. The others are grayed out to indicate that they are not available at the moment. Available is only the applet that is already installed on marathon VF2. (See section [4.4](#) for information on how to install an applet on marathon VF2.)

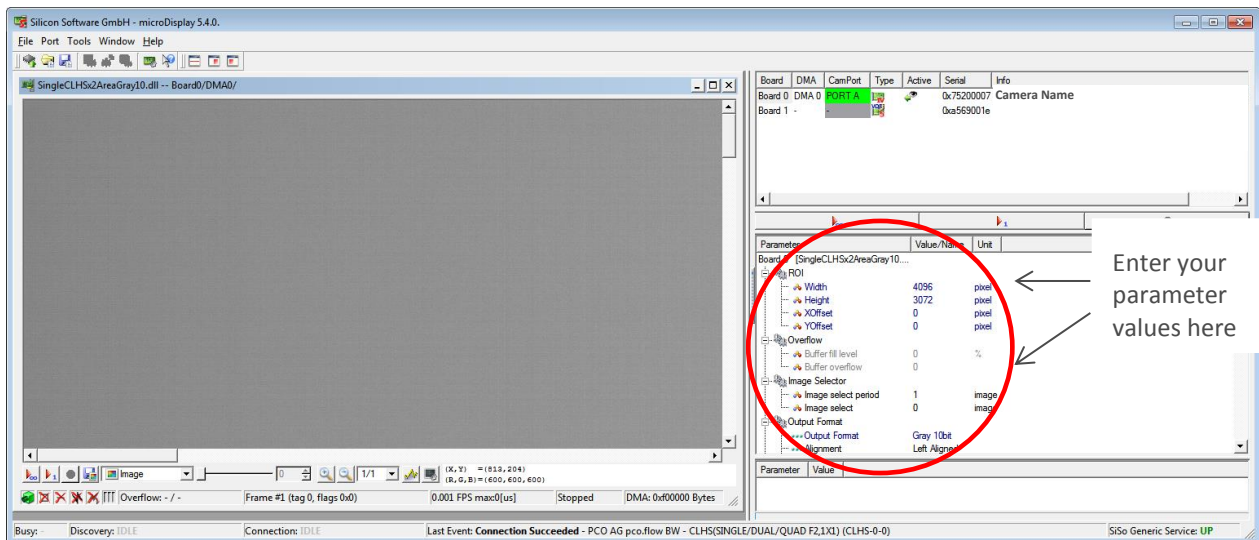
5. Select the clickable applet and click on the load button:



Figure 22: Load button in microDisplay

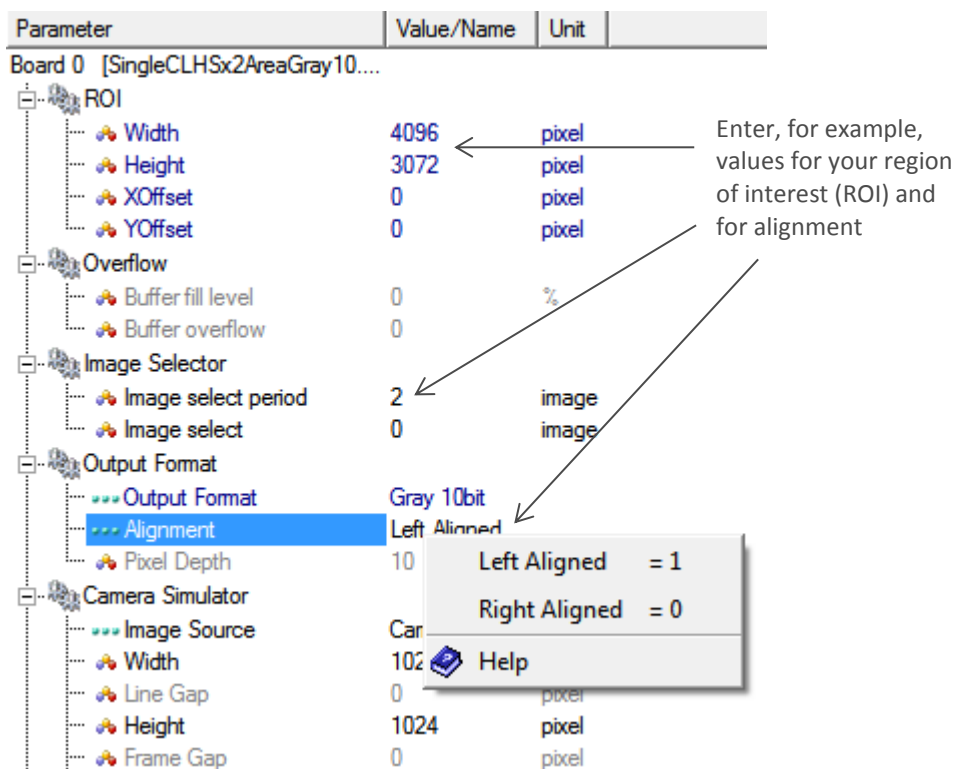
6. Close the *Load Hardware Applet* dialog.

7. In the *Parameter* panel (left bottom corner of program window), you can enter values for the individual applet parameters.



To configure the parameters of the applet you are using:

- a. Right-click directly on the value and select **Edit**.
- b. Enter the value.



8. Start image acquisition on marathon VF2 by clicking on the button **Grab and display an**

infinite number of frames  .

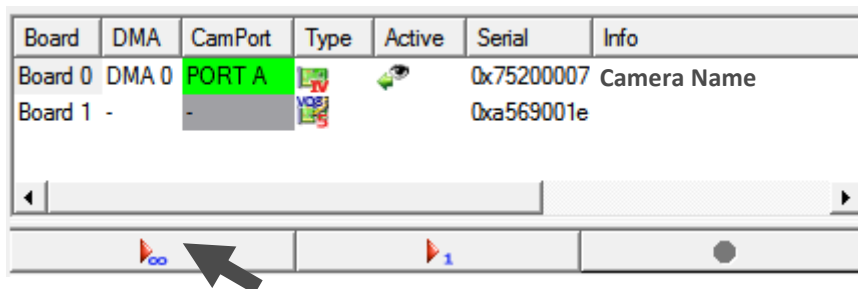


Figure 23: Starting image acquisition on the frame grabber

The grabbed images are now displayed in *microDisplay*:

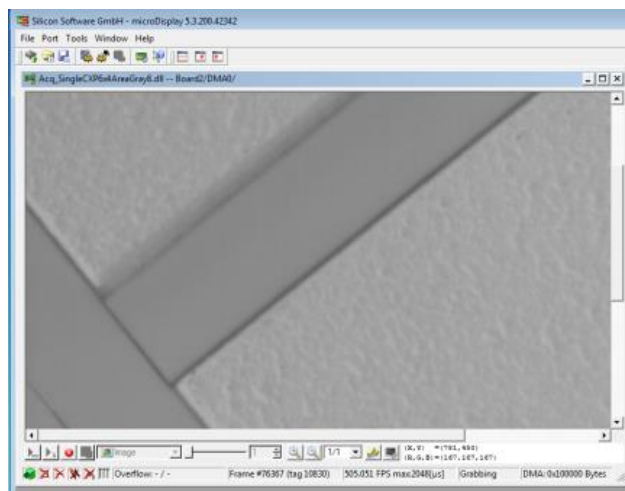


Figure 24: Display of grabbed images in the tool *microDisplay*

Image Display



You will only see an image in *microDisplay*, if the camera

- ◆ has been discovered by the system (see section [4.2.1 Autodiscovery](#))
- ◆ has been configured by you using the GenICam interface of marathon VF2 (see section [4.2.2 Configuring the Camera](#))

9. To stop the acquisition, click on the stop button in microDisplay:

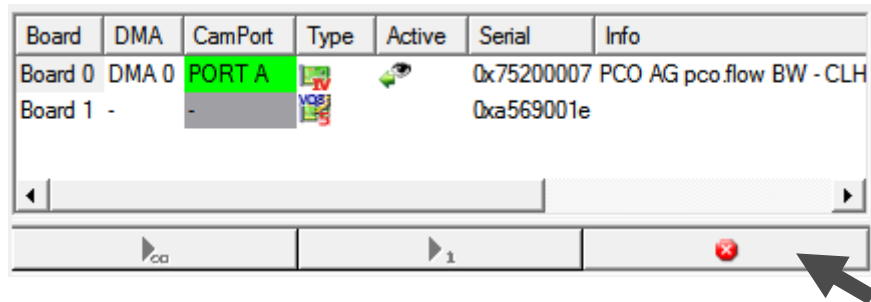


Figure 25: Stopping acquisition on marathon VF2 via microDisplay



Test Acquisition with Pre-Installed Applet

If the applet installed on the frame grabber supports the camera type and topology you are using, the according port(s) are highlighted green in microDisplay.

Loading the Applet you need for your specific Image Acquisition/Processing

If you need another applet than the one that is available in microDisplay: Install the applet you need, see section [4.4 Installing an Applet onto marathon VF2 \(Flashing\)](#).

4.6 FPGA Fall-Back Configuration


4.6.1 Automatic Load of Available Configuration

FPGAs lose their configuration at power down. The configuration is loaded anew at each cold start (or optionally at a later point of time). On marathon VF2, the FPGA loads its configuration from a BPI flash memory.

To ensure the frame grabber is working reliably also in exceptional cases, the BPI flash memory is divided into several parts. One of them contains the configuration the user is going to use.

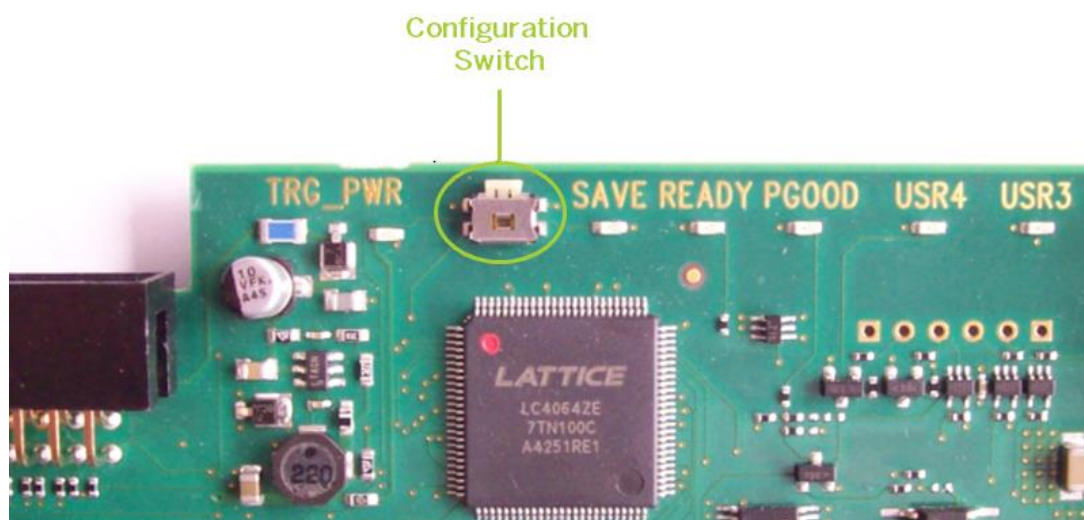
Another part contains a fall-back configuration (configuration *Save*). Configuration *Save* is loaded automatically in cases the intended configuration cannot be loaded.

This way, you always have software access to the FPGA, i.e., you can always re-flash marathon VF2 with a fully functional configuration (applet).

	<p>When do such Situations Occur?</p> <p>The most common cause why the intended configuration cannot be loaded is that marathon VF2 is disconnected from power while the BPI flash is loaded. Disconnection from power leaves the loading process unfinished. Afterwards, the FPGA cannot use the (incomplete) content of the flash for configuration. The FPGA uses the <i>Save</i> configuration instead.</p>
---	--

4.6.2 Configuration Switch

Any time, you can activate configuration *Save* via the Configuration switch which is located directly on the board. This is the ultimate fall-back alternative in the very unlikely case the *Save* configuration is *not* loaded automatically.



To use the Configuration switch:

1. Press the Configuration switch.
2. Reload the frame grabber's device driver, or re-start the host PC (warm start).

4.6.3 Re-Flashing marathon VF2

As soon as configuration *Save* is successfully activated (automatically or via switch), the according LED "SAVE" gives red light. This indicates marathon VF2 is available via software. You can re-flash marathon VF2 now with a fully functional version of the applet you want to use for image acquisition. For flashing, you use the software tool microDiagnostics.

1. Open microDiagnostics and load the applet you want to use (in a fully functional version) on marathon ACX QP as described in section [4.4 Installing an Applet onto marathon VF2 \(Flashing\)](#).

5 Control and Configuration via SDK

For a detailed description of the Software Development Kit (SDK) available for marathon VF2 as well as for all other Silicon Software image acquisition and processing devices, see the according online documentation on

http://www.siliconsoftware.de/download/live_docu/RT5/en/ind_sdk.html.

6 Trigger System

6.1 Introduction

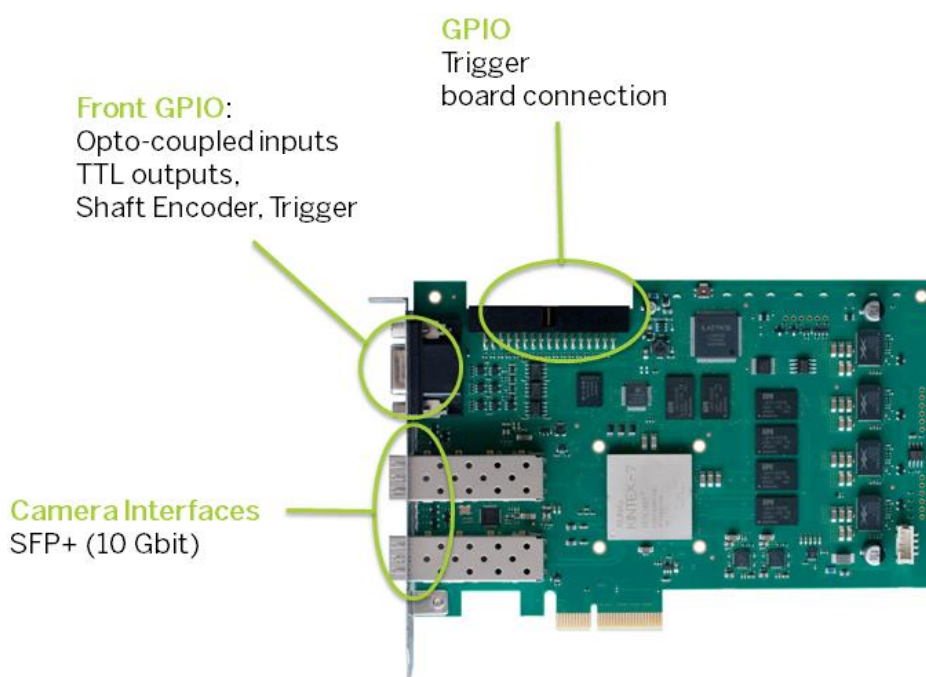
Using triggers, you are able to control the image acquisition process. You can, for example, acquire an image at a particular moment, define when to accept an image delivered by the camera, specify how many lines should be acquired, control the lighting, etc.

marathon VF2 is equipped with various trigger interfaces and connectors that allow to set up a detailed and complex trigger system.

With marathon VF2, you can

- ♦ **receive** trigger signals from external devices like [shaft encoders](#), light barriers, etc. (trigger IN).
- ♦ **send** trigger signals from marathon VF2 to external devices like camera, lighting, etc. (trigger OUT)

6.2 Trigger Interfaces on marathon VF2



marathon VF2 offers the following trigger interfaces:

- ◆ One 15-pin D-Sub socket (trigger unit “Front GPIO” on the slot bracket)
- ◆ One 34-pin socket (trigger unit “GPIO”) for connecting an additional trigger board ([opto-decoupled](#) or [TTL](#))
- ◆ Two software triggers (A and B)

6.2.1 Front GPIO (Slot Bracket)

The Front GPIO covers the basic trigger setup of your marathon VF2. Its trigger connectors allow to control peripheral devices (PLC).

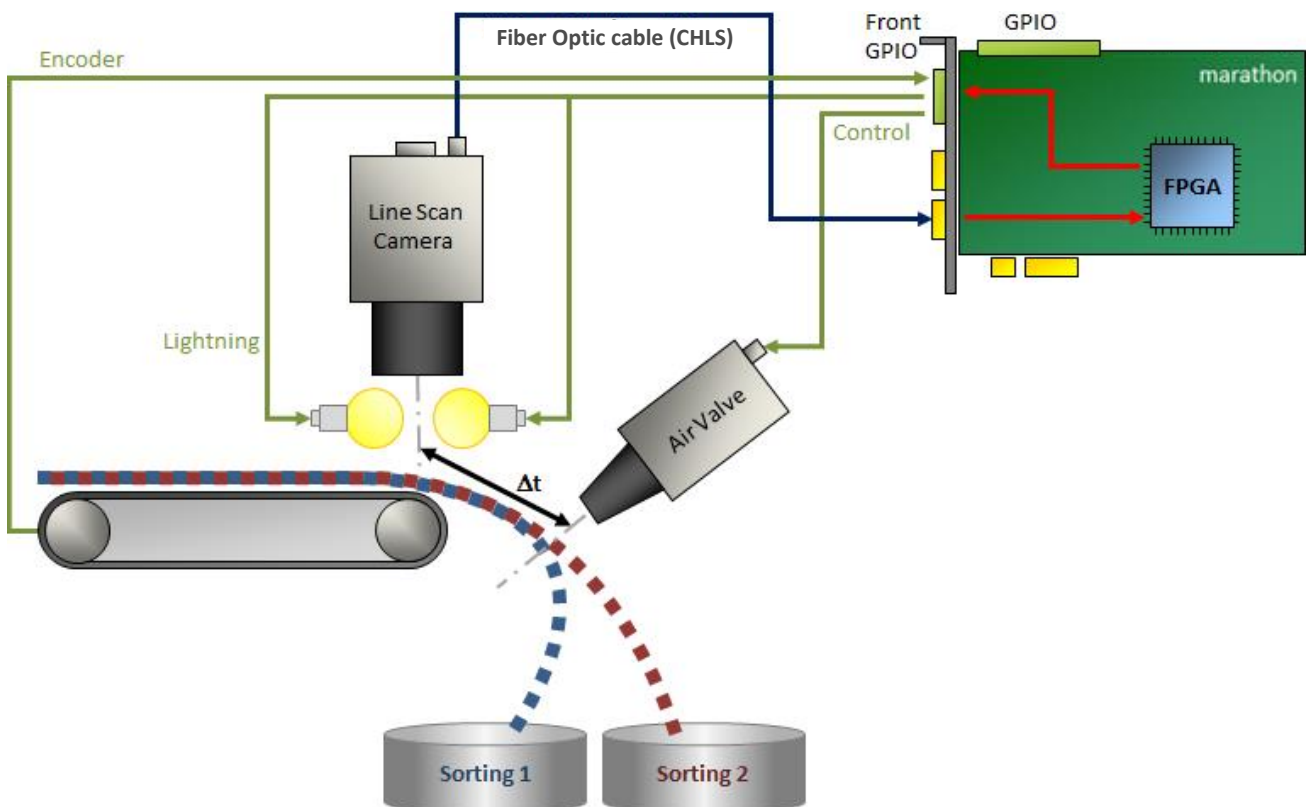


Figure 26: marathon VF2 using Front GPIO for trigger IN/trigger OUT (example: bulk sorting)

The socket is located directly on the slot bracket:

Front GPIO:
Opto-coupled inputs
TTL outputs

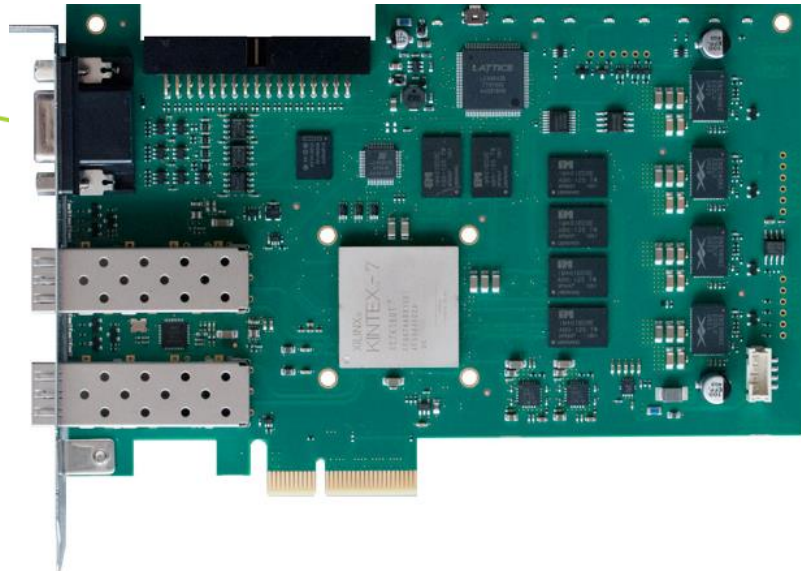


Figure 27: Front GPIO on slot bracket

In default configuration, the trigger connectors of the Front GPIO offer:

- ◆ 3 differential input signals **or** 2 differential and 1 single-ended input signal in pull-up mode
- ◆ Two TTL output signals

Custom Configuration

The physical interface of the Front GPIO is configurable. If you need another configuration of the input signals, i.e.:

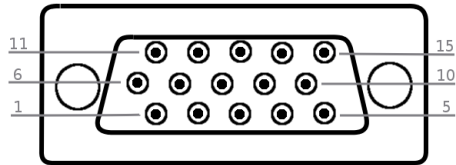
- ◆ Pull-down mode,
- ◆ 4 single-ended signals (and no differential signals),

you can easily configure the GPIOs according to your needs. For configuration, you use a command line tool which has been installed on your system together with the Silicon Software runtime.

For more information, see sections [6.2.1.2 Input Configuration](#) and [6.2.3 Configuring GPIO Units on a Frame Grabber via Command Line](#).



6.2.1.1 Pin Layout Front GPIO



Pin number	Signal	Reference Signal
1	GPO 0 (TTL)	5V / Global GND (pin 6)
2	GPO 1 (TTL)	5V / Global GND (pin 6)
3	Reserved for RS 485 (GND)	
4	Reserved for RS 485	
5	-	
6	GND (global GND)	
7	5V_Out (0,5 A max)	
8	GPI 2 + if used for	GPI voltage IN (pin 10) / GPI GND (pin 15)
9	GPI 3 - differential signal	GPI voltage IN (pin 10) / GPI GND (pin 15)
10	GPI voltage IN (4,5 – 28 V)	
11	GPI 0+	GPI voltage IN (pin 10) / GPI GND (pin 15)
12	GPI 0-	GPI voltage IN (pin 10) / GPI GND (pin 15)
13	GPI 1+	GPI voltage IN (pin 10) / GPI GND (pin 15)
14	GPI 1-	GPI voltage IN (pin 10) / GPI GND (pin 15)
15	GPI GND	

Electrically isolated

Extra Voltage IN/GND for Electrically Isolated Circuit

All *General Purpose Inputs* (GPIs) of the Front GPIO are electrically isolated. The incoming signals are transferred to the frame grabber via optocoupler. This way, the frame grabber is securely protected against high incoming voltages.

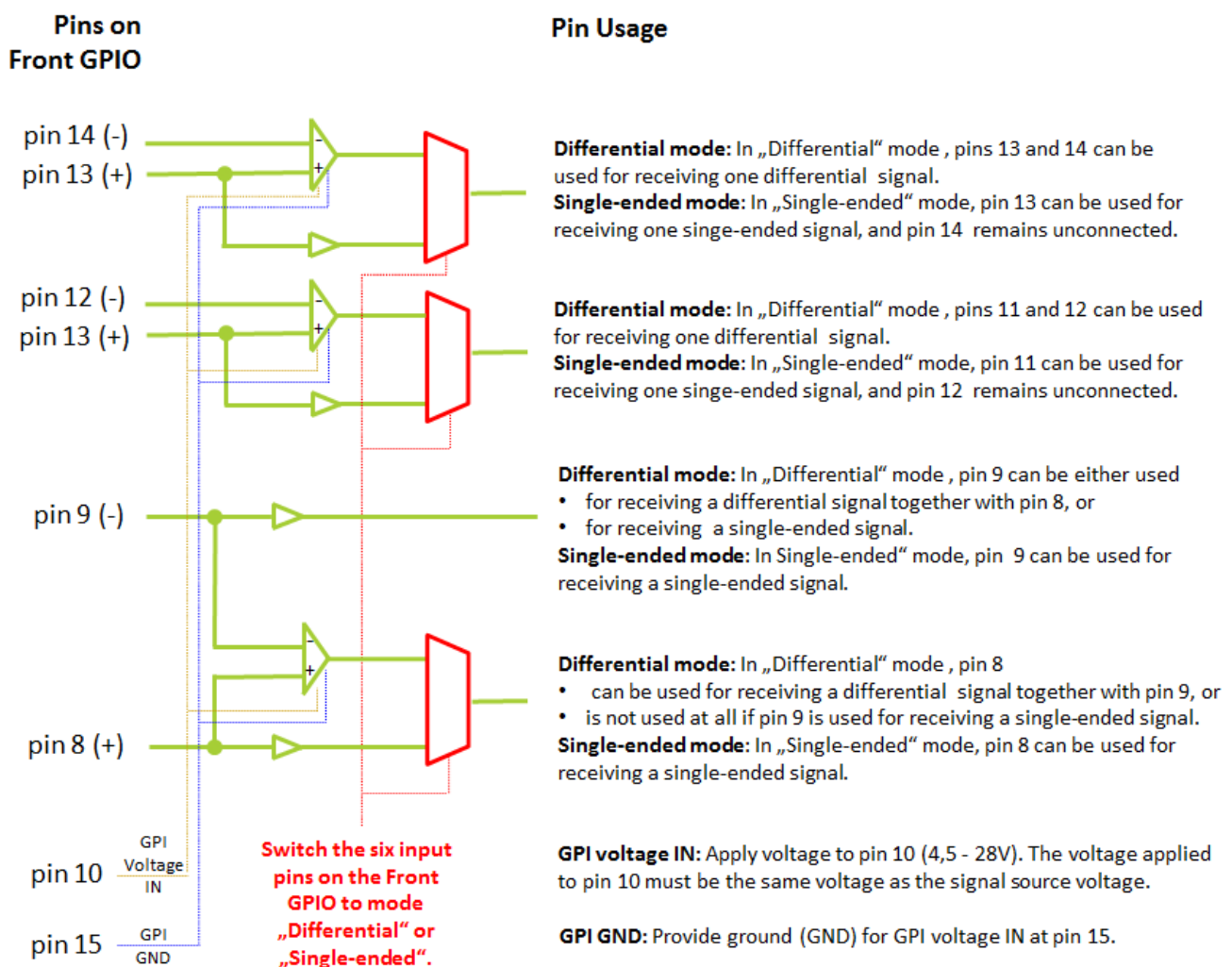
However, to operate the internal GPI circuits, you need to connect a voltage source and ground: On pin 10, you apply the voltage (**GPI voltage IN**) that serves as operating voltage for the internal operational amplifiers. On pin 15, you connect the according ground (**GPI GND**).

6.2.1.2 Input Configuration

For GPI configuration, you have two signal modes available: Mode "Differential", and mode "Single-ended". Mode "Differential" is the default mode.

In mode "Differential", you can use the GPIs on the Front GPIO for receiving either up to 3 differential signals, or for receiving two differential signals and one single-ended signal.

In mode Single-ended", you can use the GPIs on the Front GPIO for receiving up to 4 single-ended signals.



You can switch between the modes "Differential" and "Single-ended" using the command line tool as described in section and [6.2.3 Configuring GPIO Units on a Frame Grabber via Command Line](#).

Differential Mode (Default Mode)

In default configuration ("Differential" mode), all the six input pins of the Front GPIO are configured to receive differential signals (in pull-up mode):

- ◆ Pins 8/9 -> 1 differential signal (pin 8 +, pin 9 -)
- ◆ Pins 11/12 -> 1 differential signal (pin 11 +, pin 12 -)
- ◆ Pins 13/14 -> 1 differential signal (pin 13 +, pin 14 -)
- ◆ Pin 10 -> GPI voltage IN (4,5 - 28 V): The **GPI voltage IN** must be the same voltage as the signal source voltage.
- ◆ Pin 15 -> GPI GND

Alternatively, you can (in standard configuration) use pin pair 8/9 for receiving one single-ended signal:

- ◆ Pin 9 -> 1 single-ended signal
You connect the single-ended incoming signal to the physical pin 9; Pin 8 does not get connected.
- ◆ Pins 11/12 -> 1 differential signal (pin 11 +, pin 12 -)
- ◆ Pins 13/14 -> 1 differential signal (pin 13 +, pin 14 -)
- ◆ Pin 10 -> GPI voltage IN (4,5 - 28 V): The **GPI voltage IN** must be the same voltage as the signal source voltage.
- ◆ Pin 15 -> GPI GND

Single-Ended Mode

If you need more than 1 single-ended input signal, you can switch the GPIs of your Front GPIO into "Single-ended" mode. In "Single-ended" mode, the six input pins of the Front GPIO are configured to receive 4 single-ended signals:

- ◆ Pin 8 -> 1 single-ended signal
- ◆ Pin 9 -> 1 single-ended signal
- ◆ Pin 11 -> 1 single-ended signal (pin 12 not used)
- ◆ Pin 13 -> 1 single-ended signal (pin 14 not used)
- ◆ Pin 10 -> GPI voltage IN (4,5 - 28 V): The **GPI voltage IN** must be the same voltage as the signal source voltage.
- ◆ Pin 15 -> GPI GND

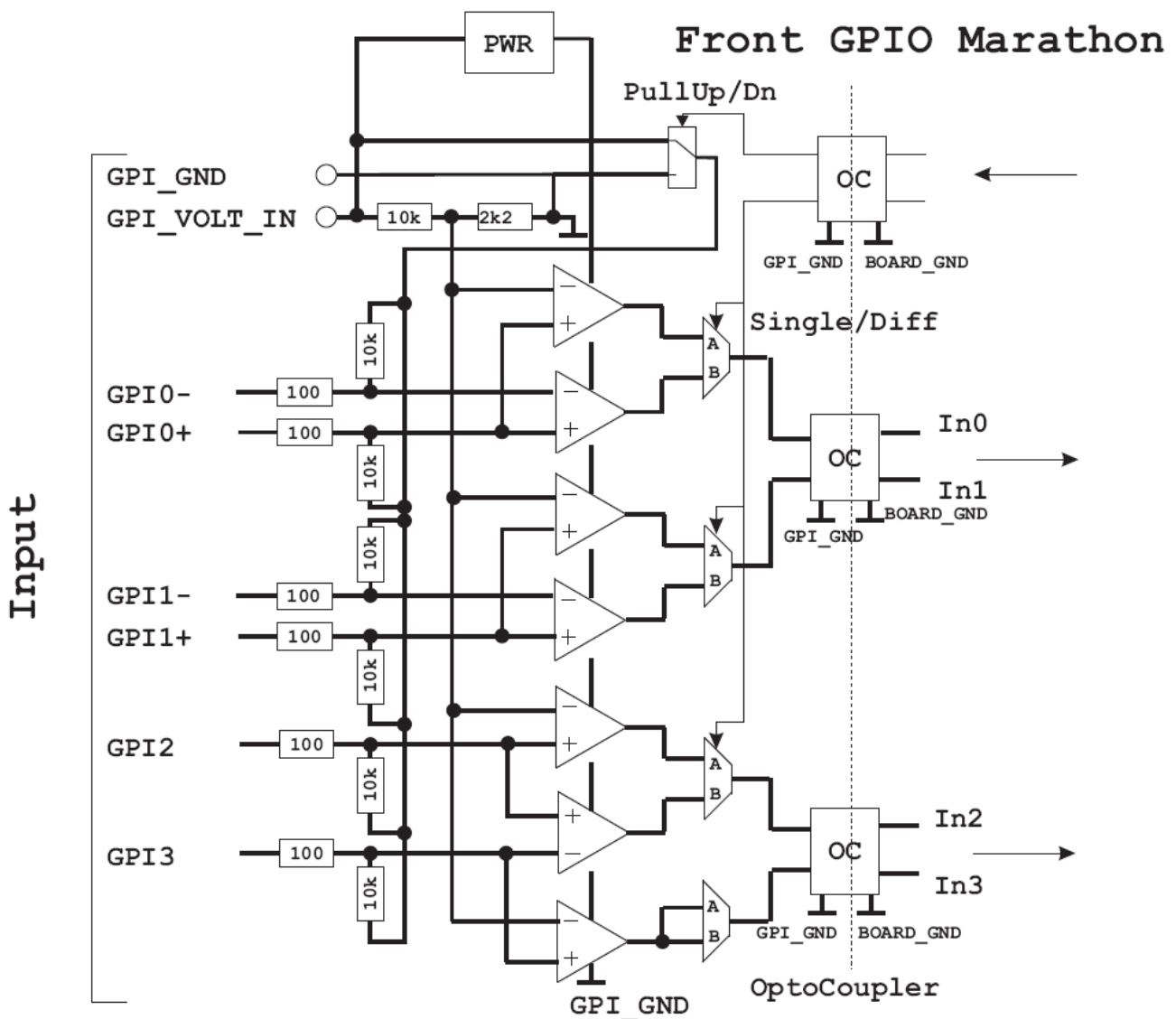
How to switch between "Single-ended" and "Differential" mode, see section and [6.2.3 Configuring GPIO Units on a Frame Grabber via Command Line](#).



Custom Configuration "Pull-Down"

Both configurations (Differential and Single-ended) can be used in pull-up and pull-down mode. Default is the pull-up mode. If you want to use the inputs in pull-down mode, you can configure the pins accordingly, see section and [6.2.3 Configuring GPIO Units on a Frame Grabber via Command Line](#).

Schematic circuit diagram:



Electrical Characteristics of the Physical Inputs:

Electrical	Minimum	Typical	Maximum	Unit
Supply voltage	4.5		28	V
Input threshold		20% Supply Voltage		V
Differential input offset voltage		10		mV
Current at GPI pin		4		mA
Total current at GPI voltage IN			50	mA

Timing Characteristics of the Physical Inputs:

Timing	Minimum	Typical	Maximum	Unit
Propagation delay	40	60	80	ns
Min. pulse width		200		ns
Max. frequency		2.5		MHz

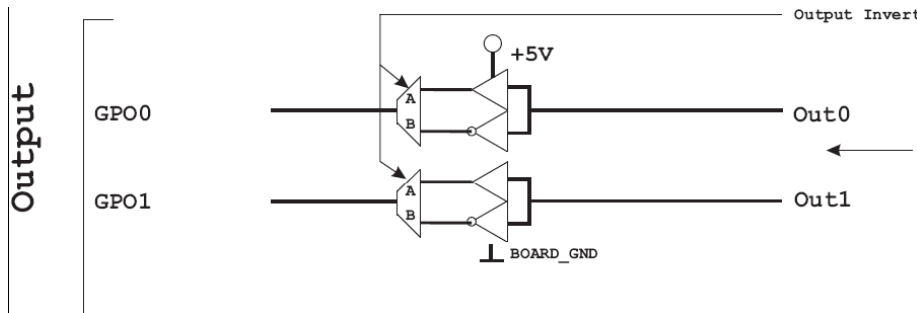


Attention

marathon VF2 has been designed with a varistor which opens at an input voltage of 30V (on the GPIOs) to let the onboard surge protector get active to protect the board. At a total supply voltage of 36V, the electronic chips would become defective.

6.2.1.3 Trigger Output TTL

Pin 1 and 2 can be used to send TTL trigger output signals (5 V).



Electrical Characteristics of the Physical Outputs:

Electrical	Minimum	Typical	Maximum	Unit
Output voltage high (V_{OH})	3.8		5	V
Output voltage low (V_{OL})			0.58	V
Output current (I_O)			32	mA

5V Out (max. current 400 mA): The Front GPOI offers a voltage out pin that provides 5 V. It can be used for a current of maximal 400 mA.

Timing Characteristics of the Physical Outputs:

Timing	Minimum	Typical	Maximum	Unit
Rise Time/ Fall Time		2.5		ns

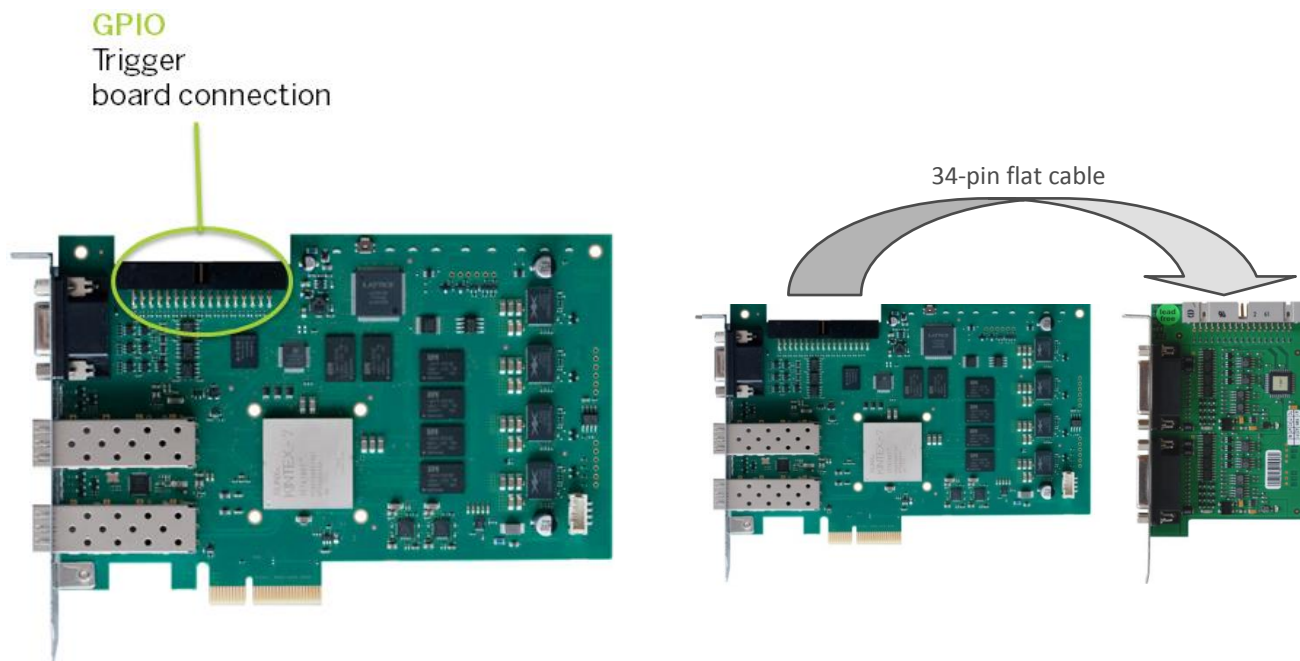


Refer to Texas Instruments Data Sheets

For details on the electrical and timing characteristics of the Front GPIO outputs, refer to the TI SN74LVC1G97 data sheet for details.

6.2.2 GPIO (34-Pin Flat Cable Connector)

The 34-pin flat cable connector links the frame grabber to the external trigger board.



The following figure shows how marathon VF2 may be used in conjunction with a trigger extension board:

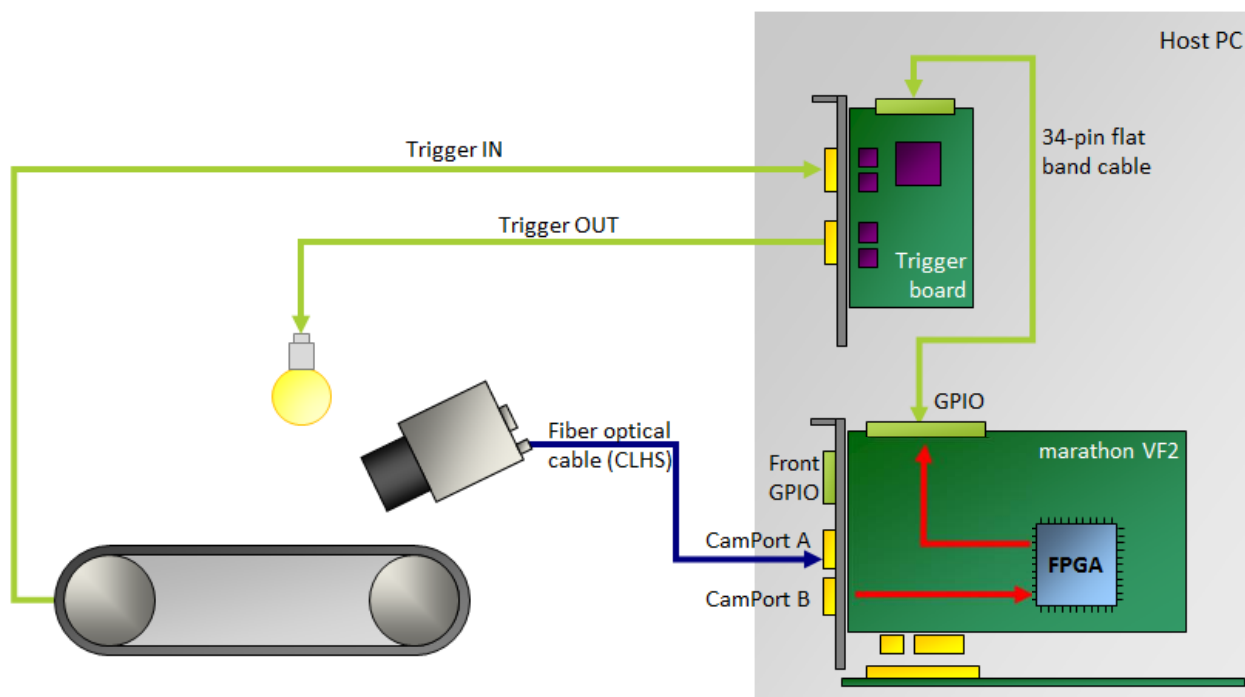






Figure 28: marathon VF2 with trigger extension board (example)

The GPIO connector, i.e., each trigger extension board, offers

- ◆ 8 digital inputs (IN 0 - 7) (8 single-ended signals or 4 differential signals)
- ◆ 8 digital outputs (OUT 0 - 7).

The eight inputs have the indices 0 to 7. The eight outputs also have indices from 0 to 7.

	<p>Using individual GPIs and GPOs</p> <p>The GPIO connector is used for connecting a trigger extension board. How the physical interface of the trigger extension board is configured (TTL or opto-decoupled, pull-up, pull-down, differential or single-ended signals) depends on the trigger extension board you are using (see our online documentation Opto-Trigger Board(s) / TTL Trigger Board).</p>
	<p>Applet-Specific Trigger Settings</p> <p>How the individual inputs and outputs of the trigger extension board you are using are employed you can configure via the applet you are using. Please refer to the documentation of the specific applet you are using.</p>
	<p>Allocation of GPIOs over Cameras</p> <p>The eight inputs and outputs are mapped to the camera ports. If a single-camera applet is used, all eight inputs and outputs can be used for one camera. In dual-camera applets, the first four inputs (0 to 3) and the first four outputs (0 to 3) can be used by the trigger system of the first camera (connected to port A), while the other four inputs (4 to 7) and the other four outputs (4 to 7) can only be used by the trigger system of the second camera (connected to port B).</p>
	<p>Multi-Board Usage</p> <p>For details on connecting one or multiple marathon VF2 device(s) to one or multiple trigger board(s), see section 9.1.7.</p>

6.2.2.1 Trigger Extension Boards

The trigger extension board you use for

- ◆ Controlling cameras
- ◆ Controlling peripheral devices
- ◆ synchronizing multiple marathon VF2 boards

at small latencies.

On all available trigger boards, the GPIs and GPOs are mapped to two ports, Port A and Port B:

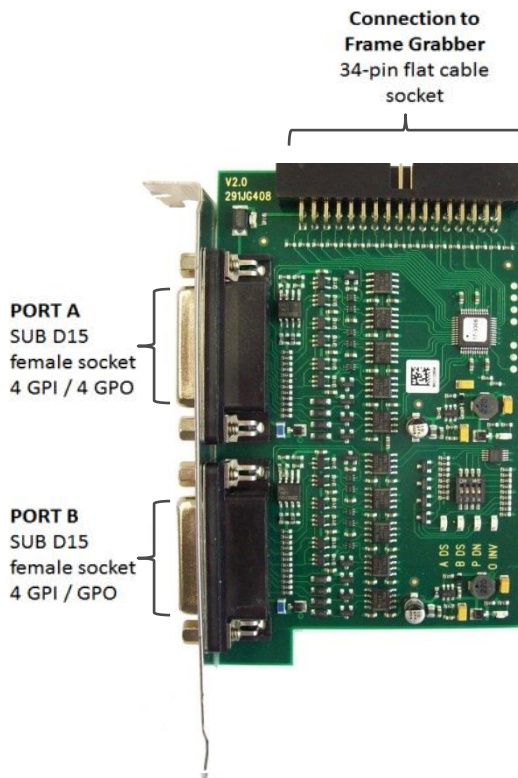


Figure 29: Opto-decoupled trigger board

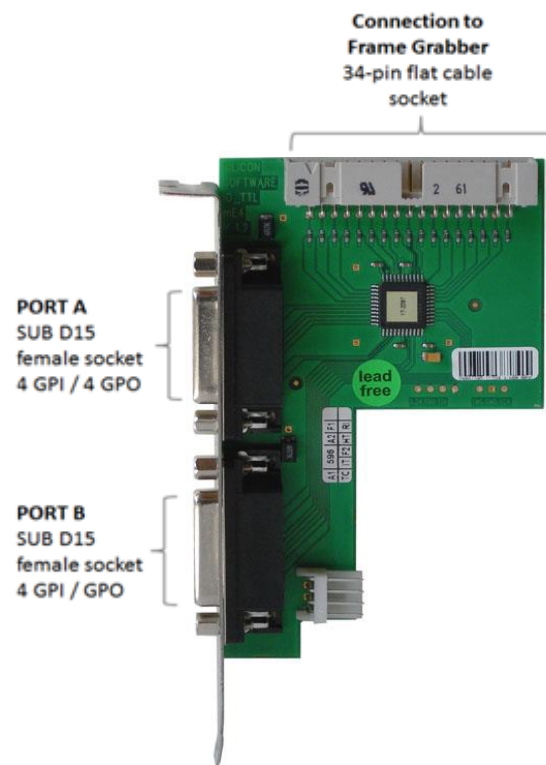


Figure 30: TTL trigger board

6.2.2.2 Signal Types

Each trigger extension board offers 8 digital inputs and 8 digital outputs. The input (GPI) pins of the trigger board can be used for various signal types. You can receive

- ◆ TTL signals or
- ◆ Opto-decoupled signals.

If you use opto-decoupled signals, you can receive

- ◆ differential signals in pull-up mode,
- ◆ differential signals in pull-down mode,
- ◆ single-ended signals in pull-up mode or
- ◆ single-ended signals in pull-down mode.

The signal type is defined by the trigger board you are using. For details, refer to our online documentation [Opto Trigger Board\(s\)](#) and [TTL Trigger Board](#).

6.2.2.3 Available Trigger Boards

For marathon VF2, the following trigger boards are available:

- ◆ [opto-decoupled](#) GPIO boards:
 - ◆ Opto Trigger board IV
 - ◆ Opto Trigger 5 *new!* (configurable) which can replace all flavors of its predecessor Opto Trigger board IV.
- ◆ TTL GPIO board: The [TTL](#) trigger board provides an adaption to the TTL signal level.

For details on the pin layout, the electrical and timing characteristics and the physical interface of a trigger extension board, refer to the documentation of the individual trigger boards in the Silicon Software Live documentation:

- ◆ [Opto-decoupled trigger extension boards](#)
- ◆ [TTLtrigger extension board](#)

For information on how to install a trigger extension board, see section [9.1.1](#).

6.2.3 Configuring GPIO Units on a Frame Grabber via Command Line

6.2.3.1 Configuring a GPIO Unit

For configuring the physical interface of a GPIO unit via command line, you work with the **gpioTool** which comes as part of the Silicon Software runtime environment.

To configure a GPIO unit via command line:

1. Open the command line window:



2. In the command line tool, go to the bin directory of your runtime installation:

```
cd /D %SISODIR5%\bin
```

3. Now, enter your commands and values as described in the following.

You have the following commands and parameters available:

```
gpioTool -b <board_index>  
-g  
-s <bank>:<settings>  
-h  
-v
```

Commands

With command **gpioTool -b** you specify which frame grabber board in your system you want to address. Specifying the addressed board is mandatory.

Via command **gpioTool -b <board_index> -g** you get the current GPIO bank settings of the specified frame grabber board displayed directly in the command line window.

Via command **gpioTool -b <board_index> -s <bank>:<settings>** you configure a specific GPIO bank on the specified frame grabber board.

Via command **gpioTool -h** you get help instructions displayed directly in the command line window.

Via command **gpioTool -b <board_index> -v** you get verbose output displayed directly in the command line window.

Value ranges:

<board_index>: Index of the frame grabber board you want to configure. The value range depends on the number of frame grabbers you have installed in your system. When you are configuring a trigger extension board (*Opto Trigger 5⁶* only), you need to specify the frame grabber the trigger extension board is connected to.

<bank>: The bank defines which GPIO bank on the frame grabber board/trigger extension board you want to address. You have the following values available:

⁶ **Opto Trigger 5** is a new Opto Trigger Board (not available yet at release of Runtime 5.4.1) that will allow complete configuration of its physical interface (pull-up/pull-down, single-ended/differential signals, inverted/not inverted outgoing signals)

Value	Addressed GPIO Bank
0	Front GPIO
1	Port A on external trigger extension board <i>Opto Trigger 5</i>
2	Port B on external trigger extension board <i>Opto Trigger 5</i>
all	Addresses all GPIO banks of the selected frame grabber. All addresses the front GPIO and (if working with <i>Opto Trigger 5</i>) both ports of the trigger extension board.

<settings>: You set here the values for three parameters, separated by comma. <settings> is in the form <signal>,<pull-up-down>,<inversion>

<signal> is either "ds" or "se" (GPIs receive differential or single-ended signals)

<pull-up-down> is either "pu" or "pd" (pull-up or pull-down)

<inversion> is either "in" (inverted) or "ni" (not-inverted)

Alternatively, "default" can be used for <settings> to set the default values for the bank, or "clear|dip-switch" can be used to control the settings on the GPIO extension board via the dip-switch (bank settings via software will be erased).

Parameter	Value	Effect of specific value
<signal>	se	You set all addressed GPIs to receive single-ended signals.
<signal>	ds	You set all addressed GPIs to receive differential signals.
<pull-up-down>	pu	You set to pull-up mode.

Parameter	Value	Effect of specific value
<pull-up-down>	pd	You set to pull-down mode.
<inversion>	ni	GPOs are not inverted.
<inversion>	in	GPOs are inverted.

Example:

```
C:\Users\nameofuser>cd /D %SISODIR5\bin
```

```
C:\SiliconSoftware\Runtime5.4.1\bin>gpioTool -b 0 -s 0:ds,pu,ni
```

Explanation:

- a) In the first line of the example, you go into the bin directory of your Silicon Software runtime installation.

In the second line:

- b) You call the GPIO configuration tool via **gpioTool**.
- c) You specify which frame grabber board in your system you want to configure. When you are configuring a trigger extension board, specify the frame grabber the trigger extension board is connected to. You first enter **-b** to say that the next input will be the board index of the frame grabber board. Then you enter the index number of the frame grabber you want to configure **{0, 1, 2 ...}**. If you have only one frame grabber board in your system, the board index is 0. In our example, you want to configure GPIOs on frame grabber 0. Specifying the frame grabber index is mandatory.
- d) With **-s** you announce that you are now starting the actual configuration.
- e) With the value that follows **{0,1,2,all}** you specify which GPIO bank on the selected frame grabber/trigger extension board you are going to configure. In the example, you selected the Front GPIO (0).

- f) With double dot: you separate the specification of the addressed GPIO bank and the actual configuration values.
- g) After the double dot you specify the values for the three parameters. In the example, the Front GPIO is configured to receive two differential IN signals (ds), to work in pull-up mode (pu), and to send the outgoing signals not inverted (ni).

6.2.3.2 Re-Setting Default Values

To re-set a bank to the default values:

1. Open the command line window:



2. In the command line tool, go to the bin directory of your runtime installation:

```
cd /D %SISODIR5%\bin
```

3. Now, enter the following string:

```
installdir\bin>gpioTool -b <board_index> -s <bank>:default
```


7 VisualApplets: Designing Individual Functionality

The design of marathon VF2 allows you to program your own individual image processing solutions (applets) using the graphical FPGA programming environment VisualApplets®.

VisualApplets® is an intuitive, graphical tool for programming FPGAs used in machine vision. You can develop applications handling very specific and highly complex image processing tasks in a time and cost effective manner. You are supported by features like visual debugging, analyzing options, and pixel-accurate simulation. An integrated SDK generator and the accompanying *microDisplay* program make the implementation on marathon VF2 easy.

7.1 Installation

Target PC

VisualApplets and some additional software needs to be installed on the PC that you use for applet development. This is typically another PC than the PC that you use as the host PC within your image processing application.

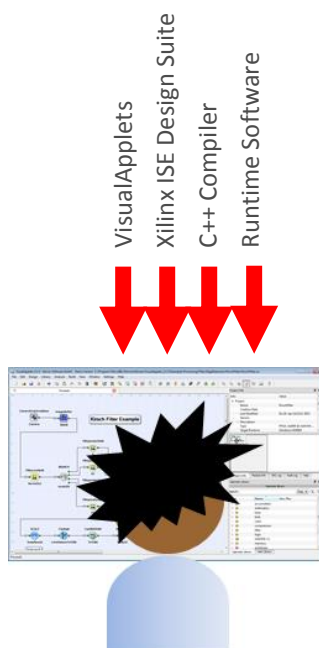


Figure 31: Software Installations required for Applet Development

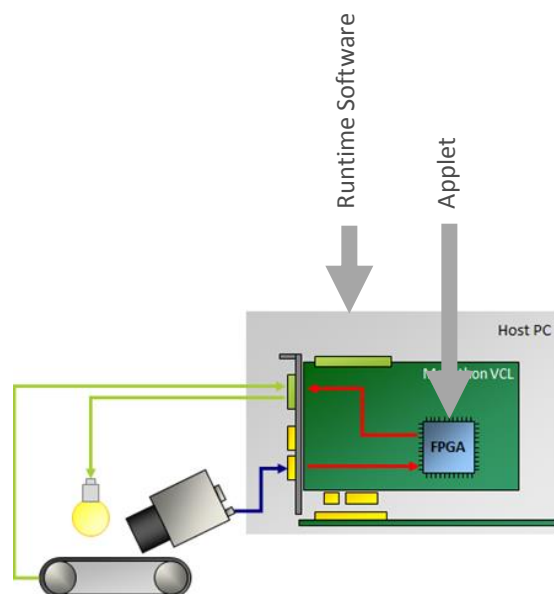


Figure 32: Installations required for image acquisition

You need to install the following software on the PC that you want to use for applet development:

1. The actual VisualApplets program (comes as an *.exe installer file)
2. Xilinx ISE Design Suite Logic Edition (required for building the applet)
3. C++ Compiler (required for compiling the SDK examples)
4. Silicon Software Runtime Software version 5.4 or higher
5. VisualApplets test design that you can use as a first basis for designing own image acquisition and processing applets for marathon VF2



VisualApplets Licensing and Hardware Dongle

For building the applet, you also need to acquire the SILICONSOFTWARE VisualApplets IDE license and to plug the USB-dongle version 2 into your machine. For details, see our website, section [VisualApplets Licensing](#).

7.1.1 System Requirements

The following system requirements have to be met:

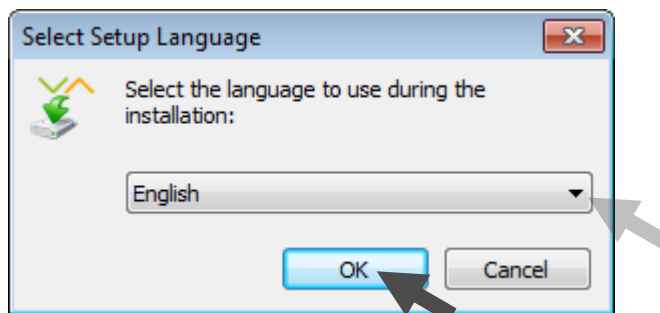
- ◆ Operation System: Windows 7 or Windows 8 (32bit or 64bit)
- ◆ Memory: Minimum 4 GByte, recommended: 8 GByte or better
- ◆ Minimum available hard disk space: 500 Mbyte
- ◆ SILICONSOFTWARE runtime environment, version 5.4

7.1.2 Installing VisualApplets

The VisualApplets installer has been provided to your company directly by Silicon Software.

To install VisualApplets:

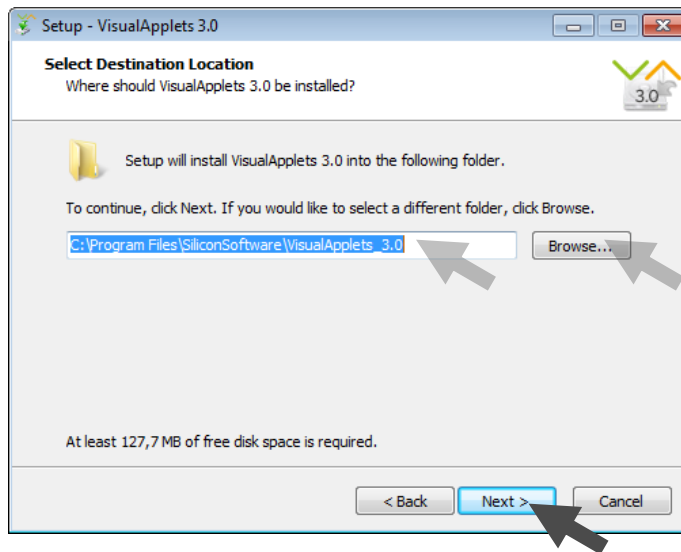
1. Run the installer that comes with the VisualApplets DVD or execute the VisualApplets installation file "Vasetup3.x.exe" by double clicking the program icon. You get the following dialog:



2. Select the desired language for the setup wizard and confirm with "OK".
The installation starts to collect information for the procedure:



3. Click "Next".



Full Access Rights Required

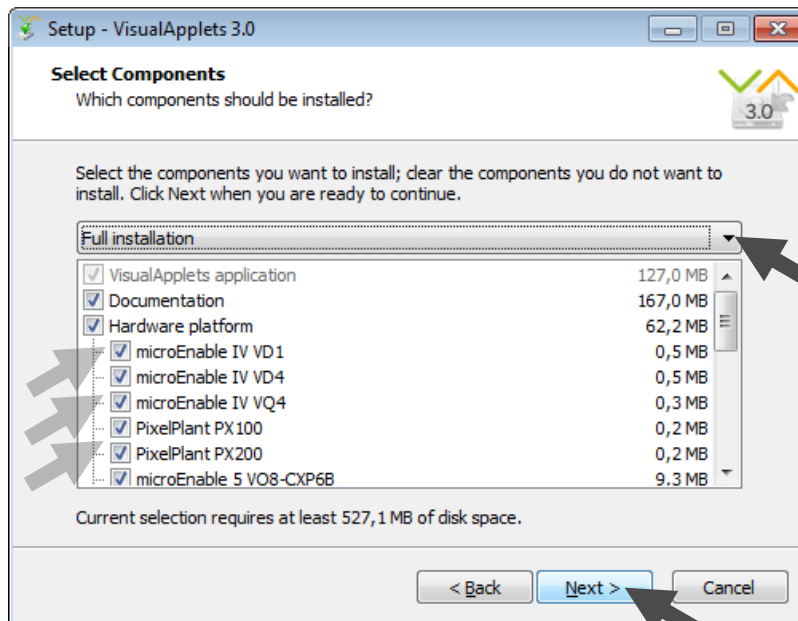


As destination folder for your VisualApplets installation, select

- the user folder or
- any other folder you have full access rights to.

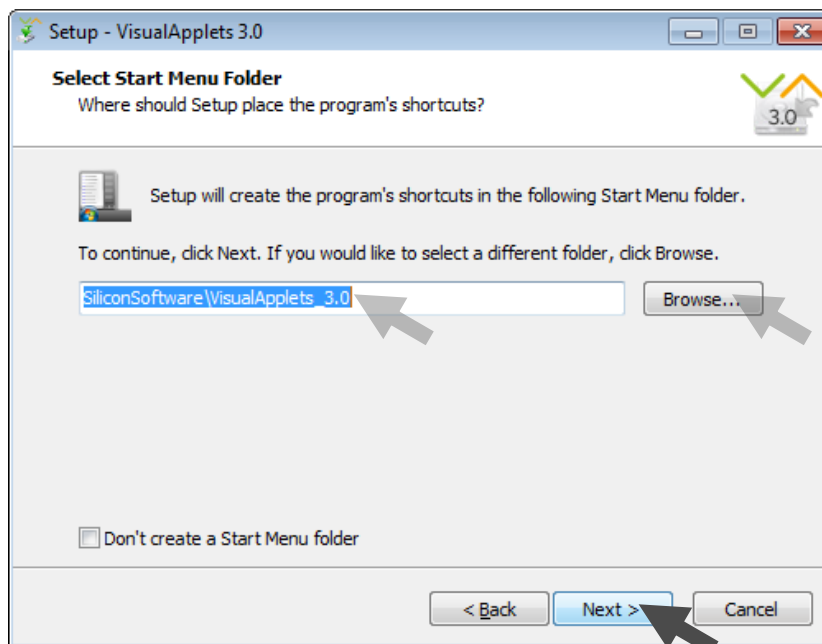
Alternatively, you are free to use any folder as your VisualApplets installation directory if you can start VisualApplets as administrator.

4. Select the destination folder. Confirm the default setting with "Next" or edit the path (manually via "Browse" button).
5. Select the installation components by predefined installation profiles:
 - ◆ You can simply use the Full installation profile.
 - ◆ If you want to reduce the volume of the installation, select your target frame grabber model and operating system, e.g., Windows 32bit, Windows 64bit, Linux 32bit, Linux 64bit, or QNX. The installation of the Microsoft Visual Studio Redistribution Package is mandatory.



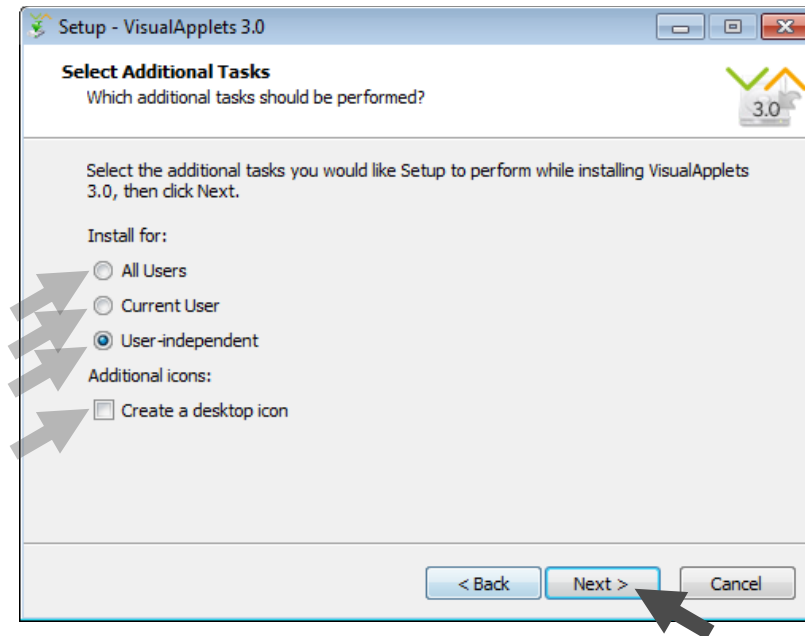
6. Confirm with **Next**.

7. Define a name for the group in the *Start Menu* folder:



8. Click **Next**.

9. Select further installation options:

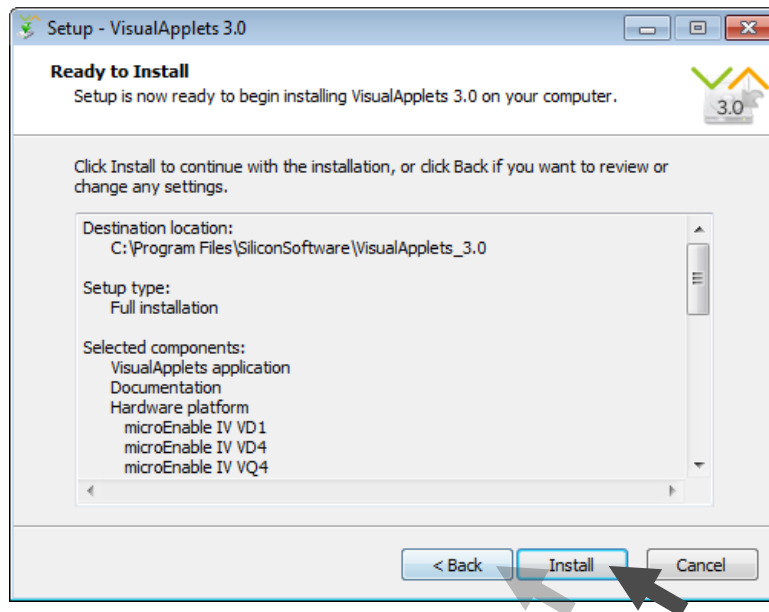


Options:

- ◆ Option 'All users/Current user': The file VisualApplets.ini will be written into the corresponding windows folder "All Users" or "Current User", considering the windows system user accounts.
- ◆ Option 'User-independent': The file VisualApplets.ini will be written into the VisualApplets installation folder. The installation is independent from the windows system user account management. This option allows the parallel installation of multiple VisualApplets versions.
- ◆ If you want to have an icon of VisualApplets on your desktop, check the box 'Create a desktop icon'.

10. Confirm with **Next**.

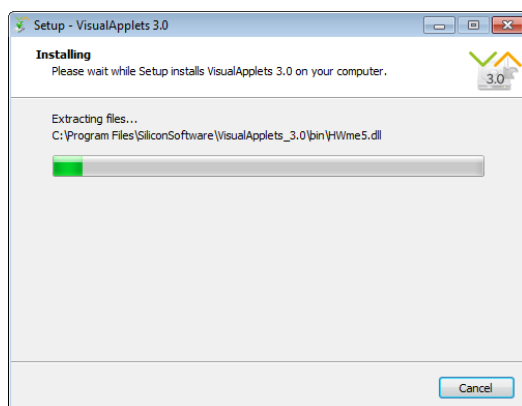
Your complete installation profile is listed:



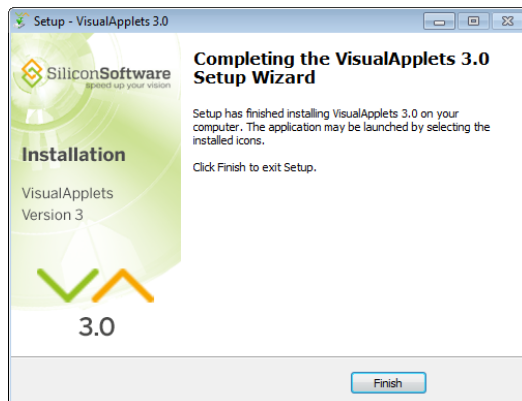
11. If you want to edit the list, use the **Back** button to change the settings in the according dialog windows.

12. To start the installation, click the **Install** button.


The progress bar displays the status of the current installation:



The installation is completed when the following screen appears:



13. Click **Finish**.



Full Access Rights Required

To enjoy all features of VisualApplets, you need to have full access rights to the installation folder. Alternatively, you can start VisualApplets by *right*-clicking on the Start Menu entry and selecting the **Run as administrator** option in the context menu.

Open

Troubleshoot compatibility

Open file location

Run as administrator

7.1.3 Installing the Xilinx Tools

You have successfully installed VisualApplets. Now, you can create applet designs as you want and your applications require.

However, before you can synthesize your designs created in VisualApplets, i.e., before you can transfer your designs into real firmware files, you need to install the Xilinx ISE® or Xilinx Vivado® tools.

Which tool and which tool version you need (and if the according license is free or not) depends on the target frame grabber you are developing for. In addition, a 30-day evaluation license for the ISE Design Suite Logic Edition version 14.7 is available directly from Silicon Software.

The table below lists all Silicon Software target hardware and the required Xilinx ISE licenses:

	Xilinx cost-free Editions		Xilinx Purchase Versions		Xilinx Eval Version
Frame Grabber Model	ISE® WebPACK Version 9.2 (free license)	Vivado® WebPACK Version 2016.1 (free license)	ISE® Design Suite Logic Edition Version 9.2 / 14.7	Vivado® Design Suite Version 2016.1	Xilinx ISE 30-Day Evaluation License Version 14.7
microEnable IV VD1-CL	ISE® WebPACK (free) Version 9.2, Download from Xilinx Website		ISE® Design Suite Logic Edition, recommended version: 9.2		
PixelPlant PX100, PixelPlant PX100e	ISE® WebPACK (free) Version 9.2, Download from Xilinx Website		ISE® Design Suite Logic Edition, recommended version: 9.2		
microEnable IV VD4-CL/-PoCL			ISE® Design Suite Logic Edition, recommended version: 9.2		
microEnable IV VQ4-GE/-GPoE			ISE® Design Suite Logic Edition, recommended version: 9.2		
PixelPlant PX200e			ISE® Design Suite Logic Edition, recommended version: 9.2		
microEnable 5 VQ8-CXP6D			ISE® Design Suite Logic Edition, Version 14.7 (minimum: version 14.x)		Xilinx ISE 30-day Eval, Version 14.7, available only at Silicon Software
microEnable 5 VD8-PoCL			ISE® Design Suite Logic Edition, Version 14.7 (minimum: version 14.x)		Xilinx ISE 30-day Eval, Version 14.7, available only at Silicon Software
microEnable 5 marathon VCL	ISE® WebPACK (free) Version 14.7, Download from Xilinx Website	Vivado® WebPACK (free) Version 2016.1, Download from Xilinx Website	ISE® Design Suite Logic Edition, Version 14.7 (minimum: version 14.x)	Vivado® Design Suite, version 2016.1 (minimum: version 2016.1)	Xilinx ISE 30-day Eval, Version 14.7, available only at Silicon Software
LightBridge VCL	ISE® WebPACK (free) Version 14.7, Download from Xilinx Website	Vivado® WebPACK (free) Version 2016.1, Download from Xilinx Website	ISE® Design Suite Logic Edition, Version 14.7 (minimum: version 14.x)	Vivado® Design Suite, version 2016.1 (minimum: version 2016.1)	Xilinx ISE 30-day Eval, Version 14.7, available only at Silicon Software

Detailed installation instructions you find here:

[Vivado® WebPACK \(free license\) and Vivado® Design Suite](#)

[ISE Full License](#)

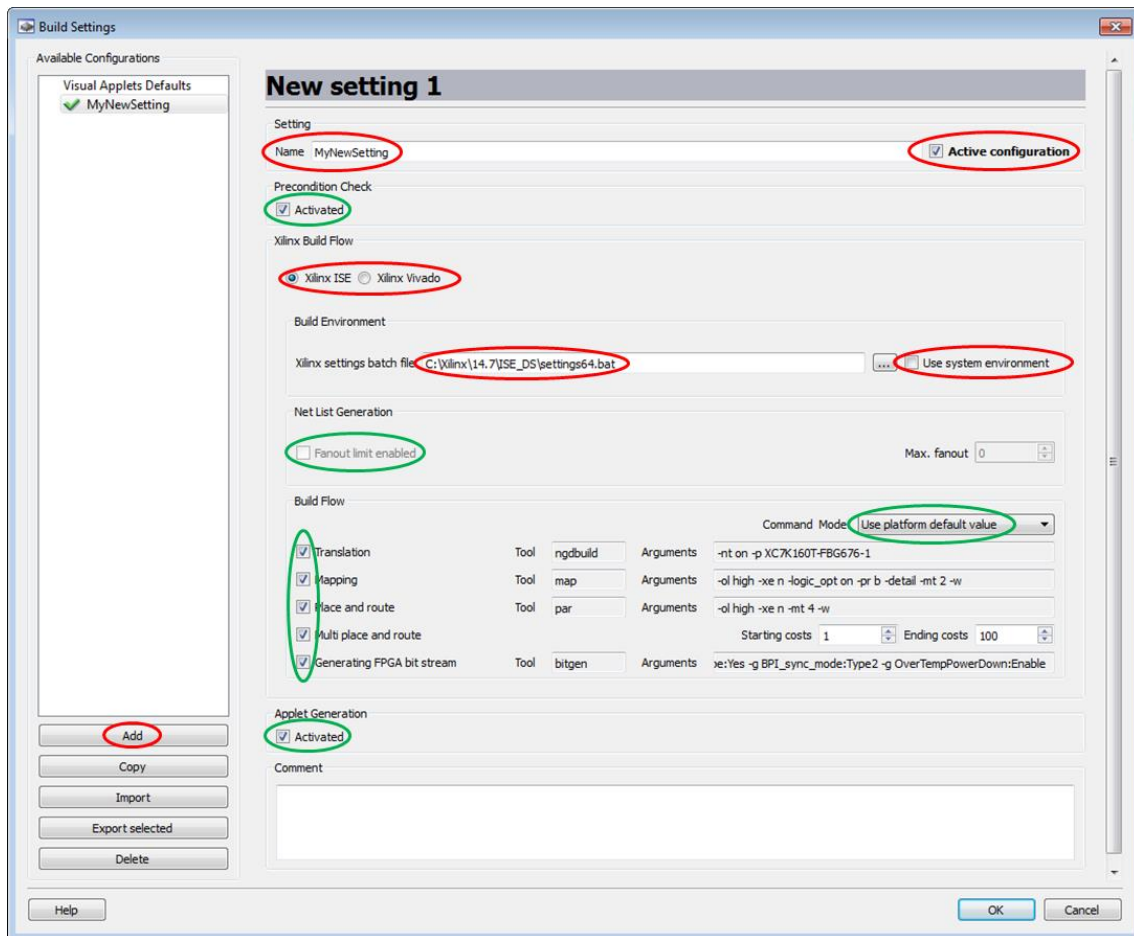
[ISE WebPACK Free License](#)

[ISE 30-day Evaluation License](#)

7.1.4 Connecting VisualApplets to Xilinx

Before you can use the Xilinx Tools within VisualApplets, you have to adapt some settings within VisualApplets:

1. Click menu **Settings** → **Build Settings**.
The **Build Settings** dialog opens.
2. Click on the **Add** button.
3. Select the target hardware platform (frame grabber) for these build settings on and confirm with **OK**.
4. Give a name to the new set of build settings you are just creating.
5. Activate the option **Active configuration** (directly behind the field where you entered the name).
6. Leave **Precondition Check** activated.
7. Under **Xilinx Build Flow**, select the Xilinx Tool you want to use (Xilinx ISE or Xilinx Vivado).
8. Disable the option **Use system environment instead**.
9. Select a Xilinx settings batch file from your file system. You find the batch file in the Xilinx installation folder.:
 - ◆ **ISE**®: \Xilinx\14.7\ISE_DS\settings64.bat.
 - ◆ **Vivado**®: \Xilinx\Vivado\2016.1\settings64.bat.
10. We recommend to use the 64-bit Windows operating system when developing applets for microEnable 5 platforms. Make sure you select the batch file that matches the operating system you are using, e.g., "settings64.bat" which is the file for the 64-bit Windows OP.
11. Keep all other settings as they are.
12. Click **OK**.



For further details on build settings options, refer to section [Build Settings](#).

7.2 Working with VisualApplets

Now you have all prerequisites installed, you are ready to actually work with VisualApplets.

In VisualApplets, you can design an image acquisition and image processing chain. You have various dedicated libraries available that you can use in a visual environment. Complex, pre-programmed functions are available to you in form of **operators**. Interfaces for signal input (image data, trigger input etc.) and signal output (image data, trigger output, etc.) are available to you in form of operators, too.

In a second step, you build your actual executable out of your design, using the VisualApplets **Build** option.

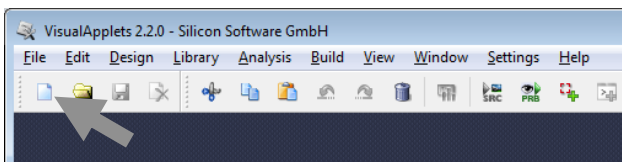


7.2.1 Workflow

To get a first impression on how to work with VisualApplets, this chapter gives you a short introduction into the tool. All steps required to generate an image processing application with VisualApplets are presented in a workflow. You will learn how fast applications can be realized and how easy this is in VisualApplets without any hardware-specific knowledge.

Step 1

You start a new project by creating a new project file.



VisualApplets will now start a new project. You will see a blank⁷ design area in the center of your program window. In the information panel on the right, you get information regarding your current project like project name, target hardware, target platform.

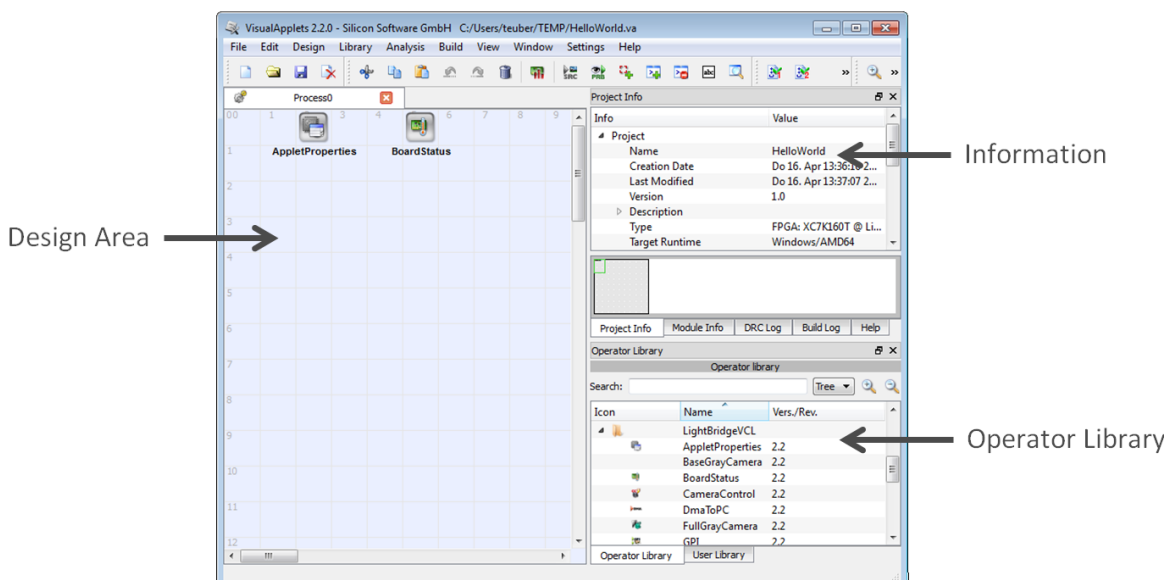


Figure 33: Empty Design in VisualApplets

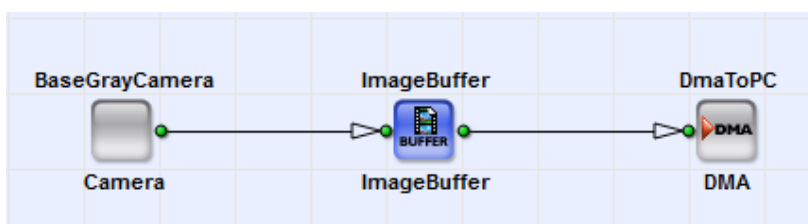
⁷ The two obligatory control operators **Applet Properties** and **Board Status** are automatically inserted into the empty design.

Step 2

You **select operators**, combine them to form your individual design, and **link them** to each other.

In VisualApplets, image processing operations are represented by operators. All these operators can be found in the operator library. Using drag-and-drop, you can very easily place operator instances into the design area.

Operators can have input and output ports. The input ports are always on the left-hand side, the output ports on the right-hand side. Use this ports to connect operator instances with each other (by clicking on one port and then on the second port of a link). Connections between operator instances are called **links** which are represented in the design window by arrows:



Operator instances and links together represent the image- (or signal-) processing pipeline; hence, the order of operations is determined by the order of operator instances and their linking.

Step 3

You **parameterize your operators and links**. (Otherwise, the design will use the preset default properties.)

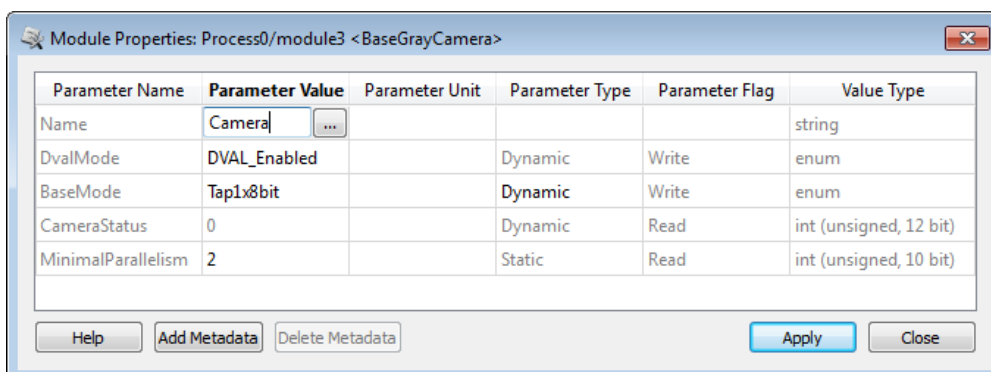


Figure 34: Parameters of an operator instance

Parameters are always operator-specific; therefore, the setting options differ from operator to operator. To open the *Module Properties* window for operator parameterization, or the *Link Properties* window for link parameterization, just double-click on the operator instance/link in your design.

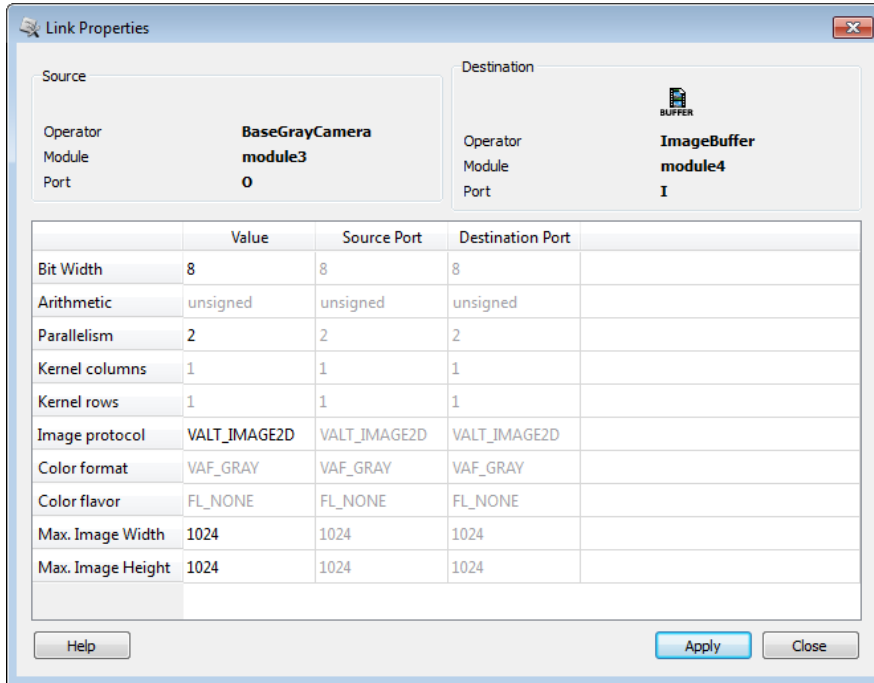


Figure 35: Parameters of a link between two operator instances

Step 4

Run the Design Rules Check (DRC).

You **check your design for errors**, using the Design Rules Check function of VisualApplets.

You simply have to click the according button , and the Design Rules Check will be run by VisualApplets.

In the **Project Info** window on the right, the DRC analysis results are displayed:



Figure 36: Successful DRC

If the DRC detects an error, you have to correct it before continuing with the next step. Detected errors are listed in link format. If you click on one of the listed errors, the respective module or link will be highlighted in the Design Window.

Step 5

You **build the final executable** hardware applet via mouse click.

VisualApplets will translate the application into the FPGA bitstream, i.e., the "program" or "applet".

After successful build, the applet is fully generated. It is saved automatically as a *.hap file to the default build output folder

[VisualApplets installation directory]/Designs

or an individual folder you can specify before you started the build.

At this point, you are done with VisualApplets and can use the applet in real FPGA hardware. To learn how to do this, proceed with section [7.3 Running Your Applet on Hardware](#).


7.2.2 Writing Applets on your Own

Our comprehensive [VisualApplets documentation](#) will support you in creating your own designs. You find the complete VisualApplets documentation, covering a manual, a tutorial, many examples, and the operator reference, on the download area of the Silicon Software website.

This documentation is divided into three major parts:

- I. The [User Manual](#) lists and explains all functions of VisualApplets. If you are new to VisualApplets, we recommend you start with '[Writing Your First Applet](#)'. To get an overview over the broad functionality VisualApplets offers, proceed with [Basic Functionality](#) and [Extended Functionality](#).
- II. [Tutorial and Examples](#) provides a deeper step-by-step introduction into VisualApplets.
- III. In the [Operator Reference](#), you find a complete and detailed description of all operators. All information provided in the Operator Reference is additionally available directly in the program as context-sensitive online help.

7.3 Running Your Applet on Hardware

	<p>Prerequisites</p> <ul style="list-style-type: none"> ◆ marathon VF2 is installed on your system (see section 2). ◆ The Silicon Software Runtime software is installed on your system (see section 3.1).
---	---

Before you can run your applet on marathon VF2, you have to load your *.hap file (applet) onto marathon VF2.

1. Install your applet on marathon VF2 as described in section [4.4](#).

After you have done this, you can

- ◆ test your applet and directly see image processing results of your applet with the Silicon Software tool microDisplay (which comes as part of the runtime software) (see section [7.3.1](#)).

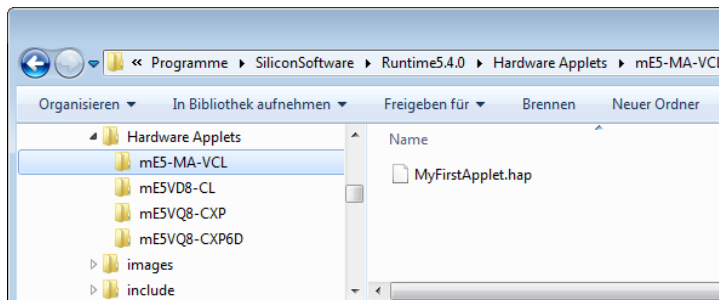
- ♦ start your applet via the application programming interface (see section [7.3.2](#)).

7.3.1 Testing your Applet in microDisplay


To test your applet and to set some first parameters, you can use the program microDisplay:

1. Save the *.hap.file you created in VisualApplets into the according folder in the installation directory:

[runtime installation directory]/Hardware Applets/mE5-MA-VF2



2. Start the tool microDisplay, either directly from VisualApplets by clicking on **Build** → **microDisplay** (**F5**), or via START -> All programs -> SiliconSoftware -> RT 5.x -> microDisplay.
3. In the dialog **I want to...**, select **Load Applet**.
4. In the **Load Hardware Applet** dialog, under **Board**, select the marathon VF2 device you want to use. Immediately, a list with applets is displayed. One applet (the one that is currently installed on the marathon VF2 device) is available. The other marathon VF2 applets are grayed out.
5. Select the available *.hap file.
6. Click on the **Load** button to load the selected applet (*.hap file) onto the frame grabber.
7. Close the **Load Hardware Applet** dialog.
8. Configure marathon VF2 (i.e., the applet) using the parameter panel of the microDisplay program window.

- ◆ When working with more than one camera, first select the port you are going to configure. Configure all ports you are using.
 - ◆ Set the parameters to your needs, e.g., image height and image width. To do so, right-click directly on the value and select **Edit**.
 - ◆ Configure specific operation modes you want to use.
9. Start image acquisition on the frame grabber by clicking on the button **Grab and display an infinite number of frames** .
- The grabbed images are now displayed in microDisplay.
10. To stop the acquisition, click on the **Stop** button in microDisplay.

7.3.2 Starting the Applet in your own Software

To start using the Applet in your own software:

1. On the Silicon Software application programming interface, use the call **Fg_Init** to start the applet.
Specify the path to the location of the *.hap file you created with VisualApplets.

7.4 Further Reading

You have just successfully implemented your first VisualApplets design. If you are interested in a deeper step-by-step introduction into VisualApplets, we recommend to read our tutorial which you can find in [Tutorial and Examples](#).

If you want to learn more about the functionalities of the program, just proceed by reading the VisualApplets [User Manual](#). Here you will find all functions of VisualApplets explained in detail.

All information provided in the [Operator Reference](#) you can also access directly in VisualApplets. Click on a **Help** button in the program and you will be automatically directed to the respective explanations in the documentation.

If you prefer a directed tool introduction, Silicon Software offers regular VisualApplets workshops and coachings. Feel free to contact us at support@silicon-software.de or sales@silicon-software.de.

8 Programming a Trigger System with VisualApplets

VisualApplets is a graphical software development environment from Silicon Software for programming FPGAs. Building applications with this development tool requires an according license, see [VisualApplets License](#).



Tip

If you are completely new to VisualApplets, we recommend you start with section [7 VisualApplets: Designing Individual Functionality](#).

8.1 Trigger Connectors on marathon VF2

Using triggers, you are able to control the image acquisition process. You can, for example, acquire an image at a particular moment, define when to accept an image delivered by the camera, specify how many lines should be acquired, control the lighting, etc.

You can use VisualApplets to build your specific trigger system for marathon VF2.

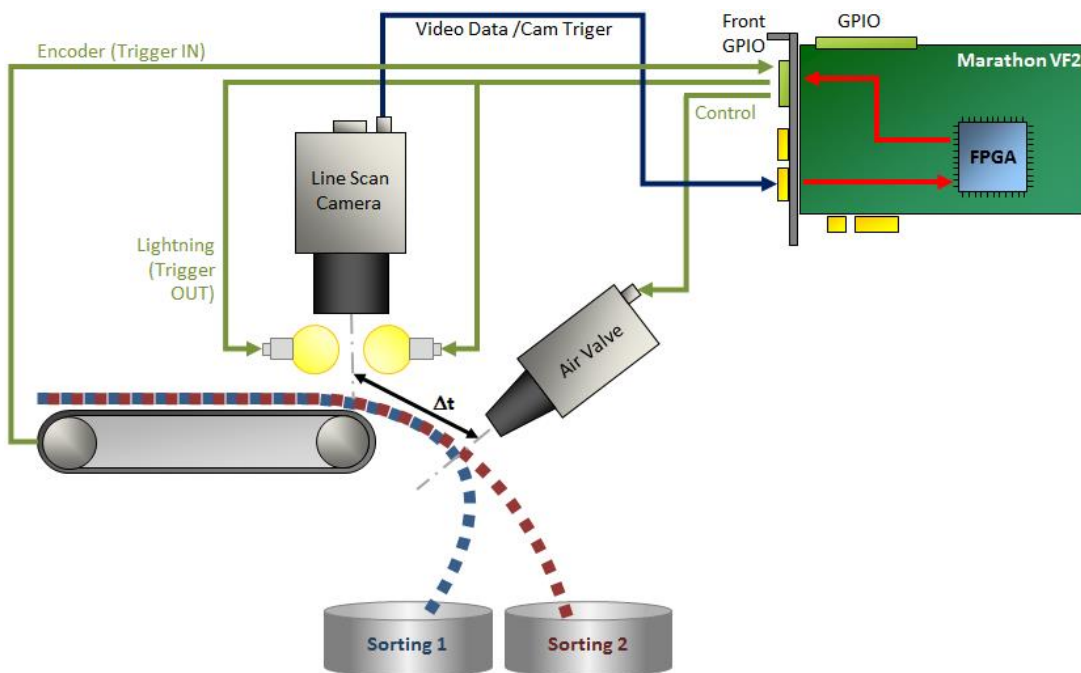


Figure 37: marathon VF2 within the production line

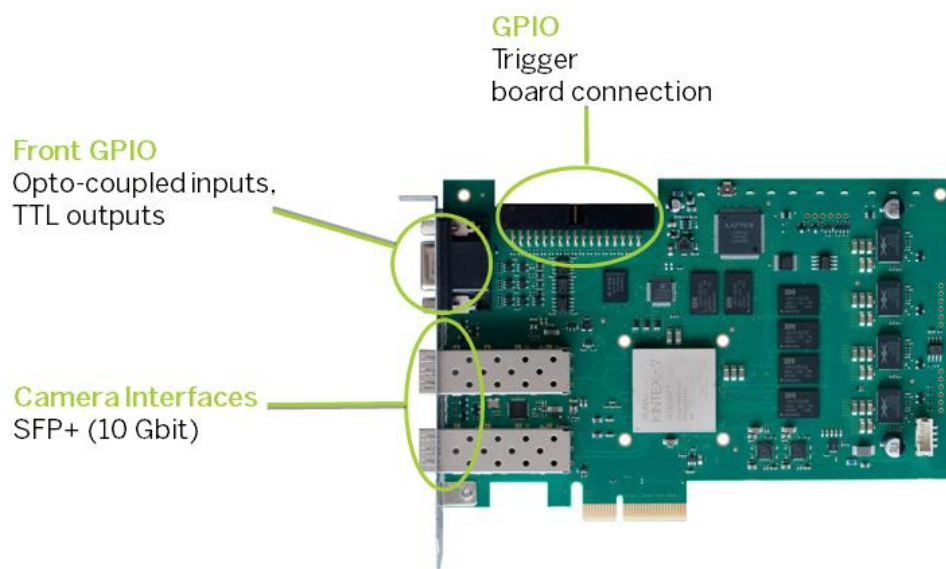
marathon VF2 is equipped with various trigger interfaces and connectors that allow to set up a detailed and complex trigger system.

With marathon VF2, you can

- ◆ **receive** trigger signals from external devices like [shaft encoders](#), light barriers, etc. (trigger IN).
- ◆ **send** trigger signals from marathon VF2 to external devices like camera, lighting, etc. (trigger OUT)

marathon VF2 offers the following trigger interfaces:

- ◆ One 15-pin D-Sub socket (trigger unit “Front GPIO” on the slot bracket)
- ◆ One 34-pin flat cable connector (trigger unit “GPIO”) for connecting an external trigger board
- ◆ Two software triggers (A and B)
- ◆ Four camera control signals (CC signals) per Camera Link port for controlling the camera



With VisualApplets, you can address the individual pins of the Front GPIO and the trigger extension board (as well as the software triggers and CC signals).

8.2 GPI/GPO Operators in VisualApplets

Using VisualApplets, you can design your own, application-specific trigger system. You can, e.g., determine which signal is sent or received by which individual physical input/output pin.

In VisualApplets, you have two operators available to wire up an individual pin with your applet design:

- ◆ Operator GPI (for trigger **IN** signals)
- ◆ Operator GPO (for trigger **OUT** signals)

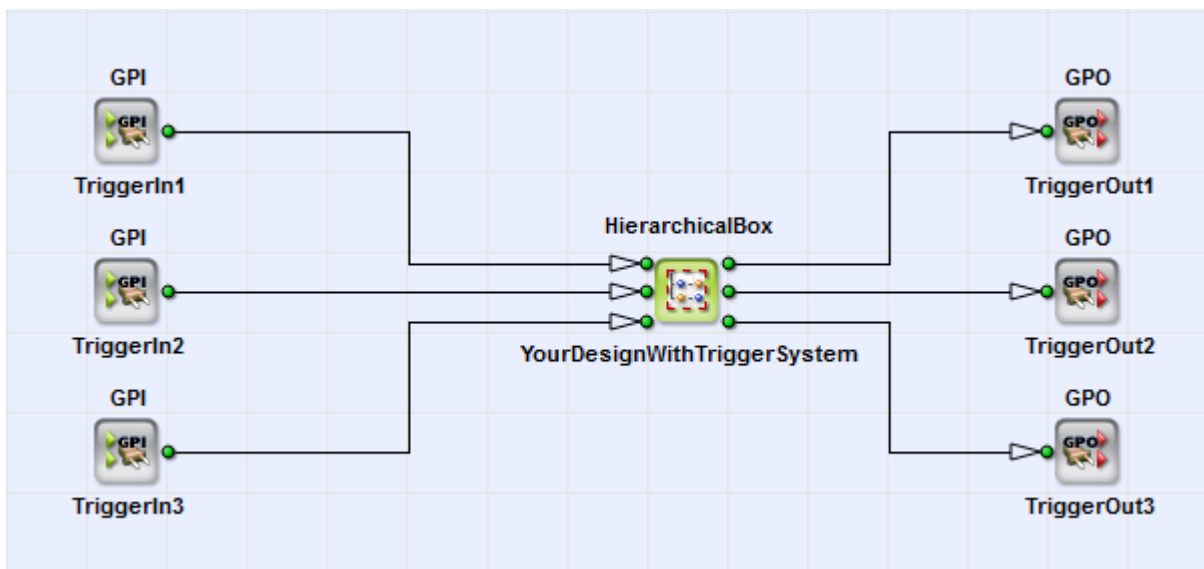


Figure 38: Using multiple instances of the GPI and GPO operators in a design

Each trigger IN / trigger OUT pin you want to use on the Front GPIO or the trigger extension board you have to interconnect with an instance of a GPI/GPO operator.



GPI and GPO Operators

Since operator GPI functions exactly as operator GPO, in the following, both operators are described together.

1. Double-click on the operator icon in your design to open the *Module Properties* window.



The *Module Properties* window opens:

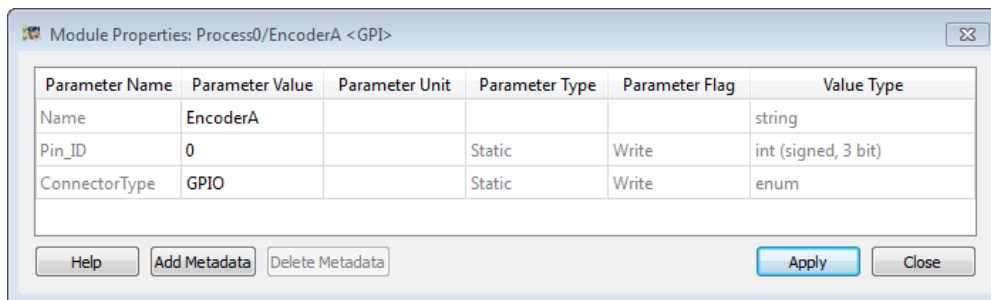


Figure 39: Module Properties window of Operator GPI

As you see, the operator has two parameters:

- ◆ Pin_ID
- ◆ ConnectorType

Parameter **ConnectorType** allows you to define which connector (Front GPIO or trigger extension board via GPIO) you want to address. Parameter **Pin_ID** allows you to define which individual pin on a connector you want to address.

8.2.1 Parameter ConnectorType

This parameter allows you to define which group of trigger pins you want to address, the ones provided in the 15-pin D-Sub socket ("Front GPIO"), or the ones on the trigger extension board ("GPIO"). The parameter has the value range {Front GPIO, GPIO}.

Value "Front GPIO"

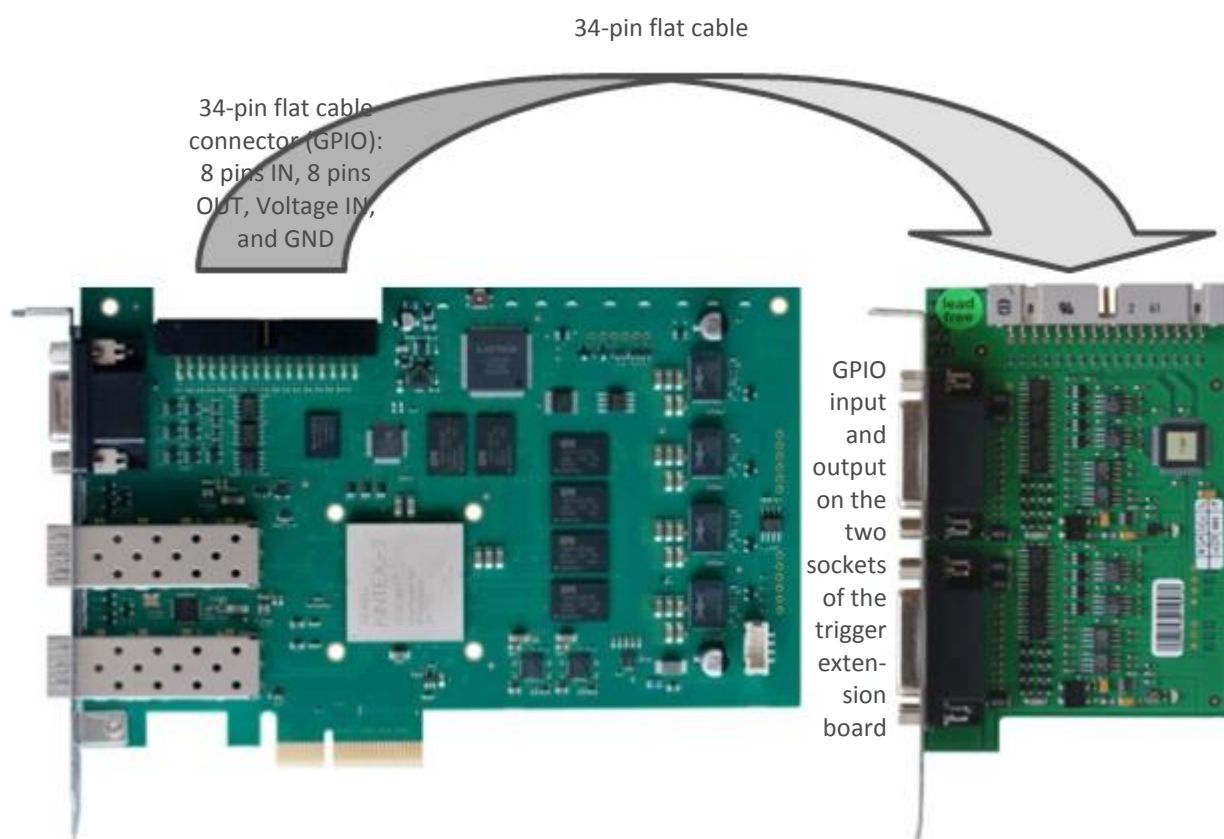
If you select value **Front GPIO**, you address the following connector (group of trigger pins) on the slot bracket of marathon VF2:



15-pin D-Sub 15 socket
(Front GPIO) for triggering
and synchronizing
peripheral devices,
including two TTL GPOs

Value “GPIO”


If you select value **GPIO**, you address the following group of trigger pins on marathon VF2:



8.2.2 Parameter Pin_ID

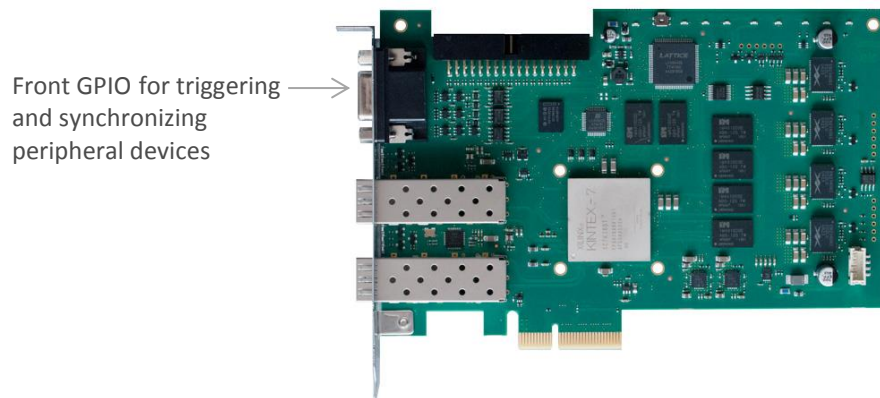
This parameter allows you to define which individual pin on the chosen connector (group of trigger pins) you address. The value range of parameter Pin_ID is {0,1,2,3,4,5,6,7} or less.

For a detailed description which pin is addressed by which value, see sections [8.3](#) and [8.4](#).

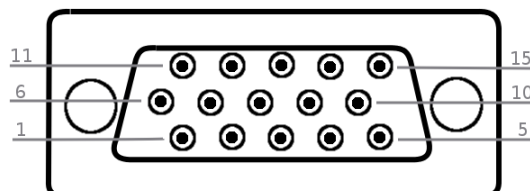
	<p>Attention</p> <p>Keep in mind that you have to set parameter Pin_ID in four different situations:</p> <ul style="list-style-type: none"> ◆ Operator GPI: <ul style="list-style-type: none"> ◆ The <i>inputs</i> of the Front GPIO (value range {0,1,2,3}) (Connector type: Front GPIO) ◆ The <i>inputs</i> of the GPIO (value range {0,1,2,3,4,5,6,7}) (Connector type: GPIO) ◆ Operator GPO: <ul style="list-style-type: none"> ◆ The <i>outputs</i> of the Front GPIO (value range {0,1}) (Connector type: Front GPIO) ◆ The <i>outputs</i> of the GPIO (value range {0,1,2,3,4,5,6,7}) (Connector type: GPIO)
--	--

8.3 Trigger Signals Addressing the Front GPIO (Slot Bracket)

The trigger pins available on the Front GPIO allow to controll peripheral devices (PLC). They offer various trigger input and trigger output options for [single ended](#) and/or [differential signals](#). Two pins are dedicated to TTL GPO signals. The socket is located on the slot bracket:



8.3.1 Pin Configuration on the Front GPIO



	Signal	PIN_ID in Visual Applets for single-ended signals	PIN_ID in Visual Applets for differential signals	Visual Applet's Operator	Reference Signal
1	GPO 0 (TTL)	0		GPO	5V / GND
2	GPO 1 (TTL)	1			5V / GND
3	-				
4	-				
5	-				
6	GND (global GND)				
7	5V_Out (0,5 A max)				

Electrically isolated	8	GPI 2 + <div>if used for</div>	2*	2	GPI	GPI voltage IN / GPI GND
	9	GPI 3 - <div>differential signal</div>	3*			GPI voltage IN / GPI GND
	10	GPI voltage IN (4,5 – 28 V)				
	11	GPI 0+	0	0		GPI voltage IN / GPI GND
	12	GPI 0-				GPI voltage IN / GPI GND
	13	GPI 1+	1	1		GPI voltage IN / GPI GND
	14	GPI 1-				GPI voltage IN/GPI GND
	15	GPI GND				

* In differential mode, pin 9 can be used to receive ONE single-ended signal on a GPI operator with PIN_ID=3, see section [8.3.2.1](#).



Attention

The marathon VF2 trigger system needs to get supply voltage on the **Voltage IN** pins. If you want to connect devices that have no PWR pin, you need to provide the power supply to the **Voltage IN** pin from an external source.

8.3.2 Trigger INPUT on the Front GPIO



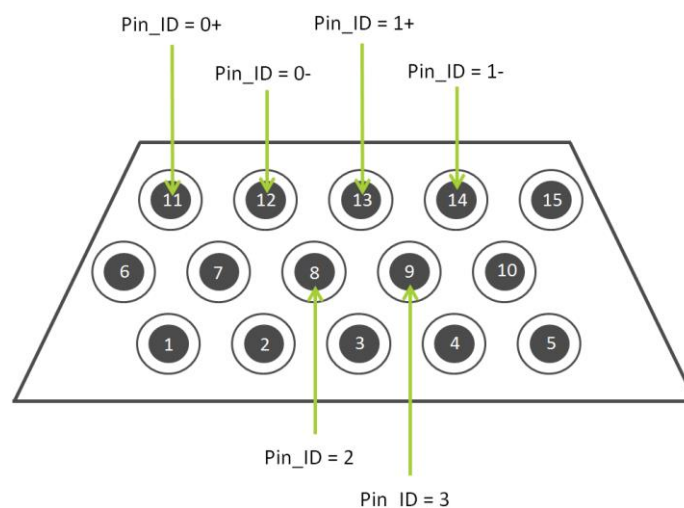
Pull-Up is Default

The default configuration for pull-up/pull-down of the GPI pins on the Front GPIO is pull-up. How to configure the GPI pins on the Front GPIO to pull-down, see section [6.2.3](#).

If you

- ◆ are using the **GPI** operator (for defining trigger *input* signals),
- ◆ have set parameter *Connector Type* to **Front GPIO**,

the values of parameter *Pin_ID* address the following pins:



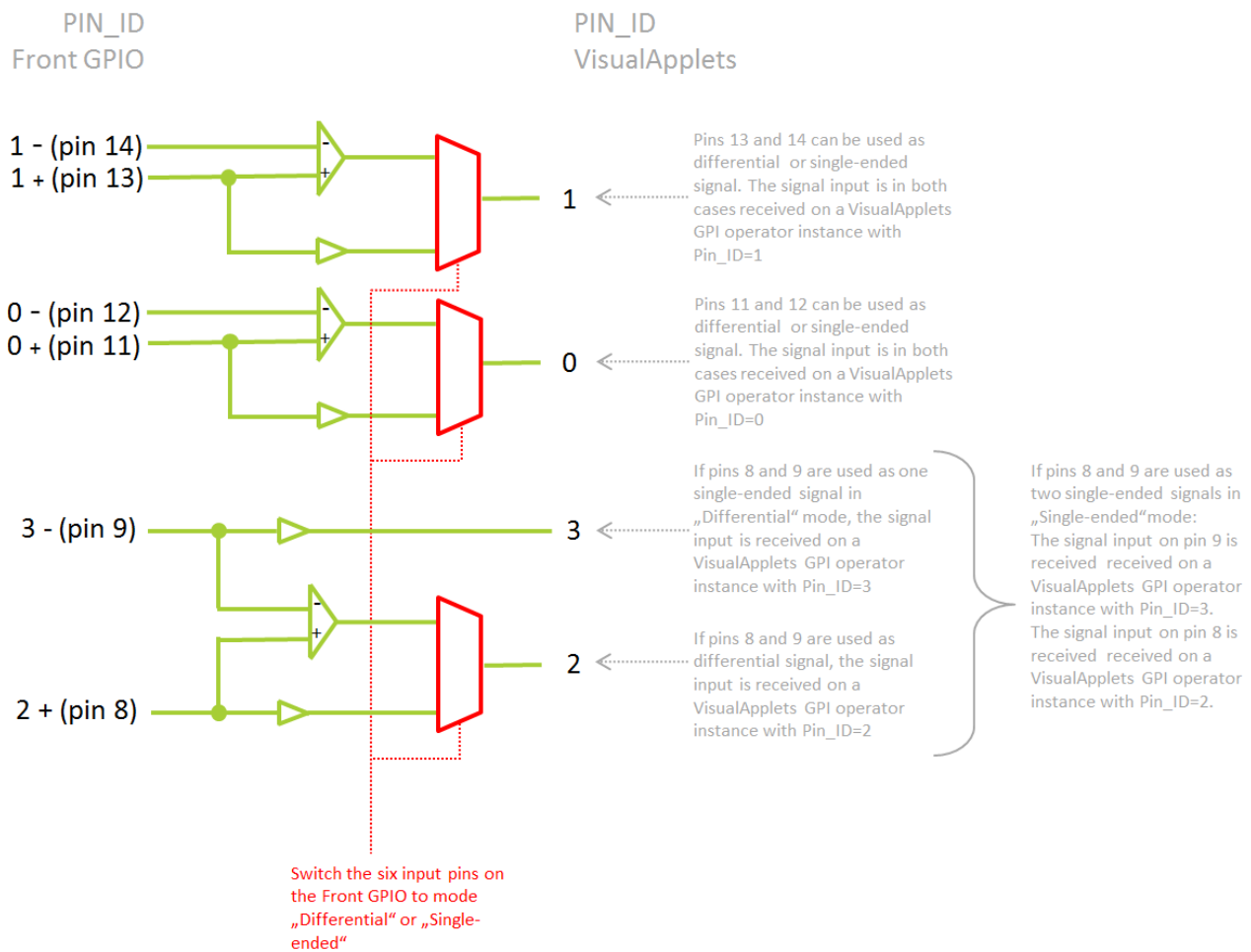


Figure 40: Usage of IN pins on Front GPIO

The input (GPI) pins of the Front GPIO are configurable. You can use these six input pins to receive either:

- ◆ 3 differential signals, or
- ◆ 2 differential signals and 1 single-ended signal, or
- ◆ 4 single-ended signals.

To configure the input pins on **Front GPIO** for a specific signal type (differential or single-ended), you have two modes available:

- ◆ Mode “Differential” (providing options a and b)
- ◆ Mode “Single-ended” (providing option c)

Default mode for the Front GPIO is mode „Differential“.



Configuring Physical Interface

For instructions how to configure the physical interface of the Front GPIO to mode “Single-ended” or back to mode “Differential”, see section [6.2.3](#).

Mode “Differential” (Default)

In mode “Differential” (default), all the six input pins of the Front GPIO are configured to receive differential signals:

- ◆ Pins 8/9 -> 1 differential signal
- ◆ Pins 11/12 -> 1 differential signal
- ◆ Pins 13/14 -> 1 differential signal

Alternatively, you can use pin pair 8/9 (in mode “Differential”) for receiving one single-ended signal:

- ◆ Pin 9 -> 1 single-ended signal
- ◆ Pins 11/12 -> 1 differential signal
- ◆ Pins 13/14 -> 1 differential signal

Mode “Single-ended”

In mode “Single-ended” (configurable), the six input pins of the Front GPIO are configured to receive 4 single-ended signals:

- ◆ Pin 8 -> 1 single-ended signal
- ◆ Pin 9 -> 1 single-ended signal
- ◆ Pin 11 -> 1 single-ended signal
- ◆ Pin 13 -> 1 single-ended signal

8.3.2.1 Mode „Differential“ – Individual Pins

Pins 11 and 12:

- ◆ The incoming signals on pins 11 and 12 are interpreted as one differential signal (pin 11 with the +, pin 12 with the - value).
- ◆ To receive this differential signal within VisualApplets, use a GPI operator instance with Pin_ID = 0.

Pins 13 and 14

- ◆ The incoming signals on pins 13 and 14 are interpreted as one differential signal (pin 13 with the +, pin 14 with the - value) .
- ◆ To receive this differential signal within VisualApplets, use a GPI operator instance with Pin_ID = 1.

Pins 8 and 9

- ◆ The incoming signals on pins 8 and 9 are interpreted as one differential signal (pin 8 with the +, pin 9 with the - value) .
- ◆ To receive this differential signal within VisualApplets, use a GPI operator instance with Pin_ID = 2.

Alternatively, you can use pin pair 8/9 for receiving one single-ended signal (although in mode “Differential”): In this case,

- ◆ you connect the single-ended incoming signal to the physical pin 9.
- ◆ To receive this single-ended signal within VisualApplets, use a GPI operator instance with Pin_ID = 3.

8.3.2.2 Mode „Single-Ended“ – Individual Pins

Signals on Pin 11 and 12:

- ◆ The incoming signal on pin 11 is interpreted as a single-ended signal.
- ◆ To receive this single-ended signal within VisualApplets, use a GPI operator instance with Pin_ID = 0.
- ◆ Pin 12 is not used in mode “Single-ended”.

Signals on Pin 13 and 14:

- ◆ The incoming signal on pin 13 is interpreted as a single-ended signal.
- ◆ To receive this single-ended signal within VisualApplets, use a GPI operator instance with Pin_ID = 1.
- ◆ Pin 14 is not used in mode “Single-ended”.

Pins 8 and 9

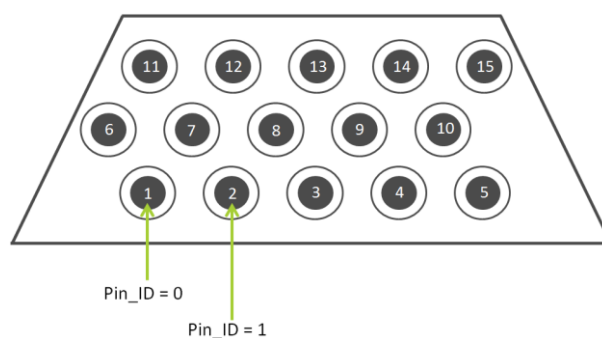
- ◆ The incoming signals on pins 8 and 9 are interpreted as two single-ended signals.
- ◆ To receive these single-ended signals within VisualApplets, use
 - ◆ a GPI operator instance with Pin_ID = 2 for pin 8, and
 - ◆ a GPI operator instance with Pin_ID = 3 for pin 9.

8.3.3 Trigger OUTPUT “TTL” on the Front GPIO

If you

- ◆ use the **GPO** operator for defining trigger *output* signals,
- ◆ you have set parameter *Connector Type* to **Front GPIO**,

the values of parameter *Pin_ID* address the following pins:



These two pins are sending TTL signals (5 V).

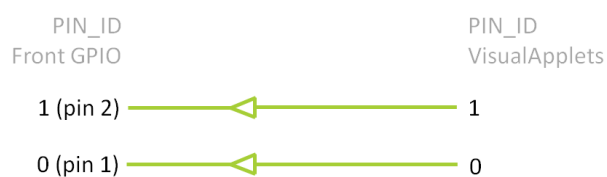


Figure 41: Outgoing TTL signals on Front GPIO (5 V)

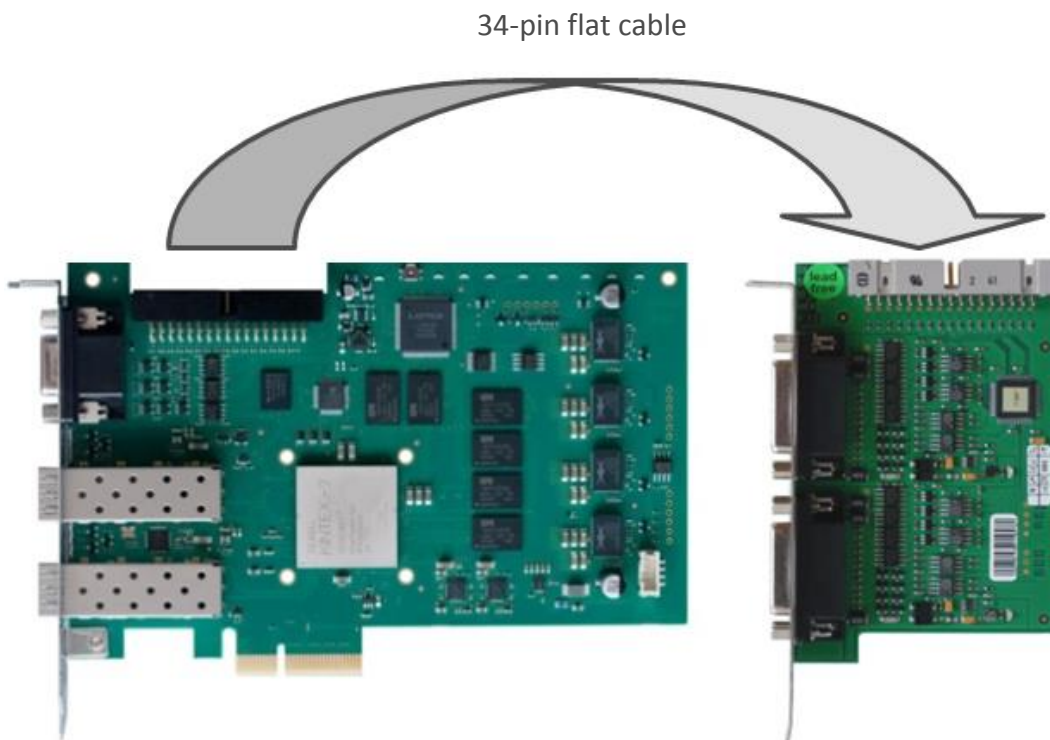
8.4 Trigger Signals Addressing the Trigger Extension Boards (via GPIO)

8.4.1 Trigger Extension Boards

The GPIO connector, i.e., each trigger extension board, offers

- ◆ 8 digital inputs (IN 0 - 7) (8 single-ended signals or 4 differential signals)
- ◆ 8 digital outputs (OUT 0 - 7)

The GPIO connector is the physical trigger interface of marathon VF2 that is designed for connection to a trigger extension board:



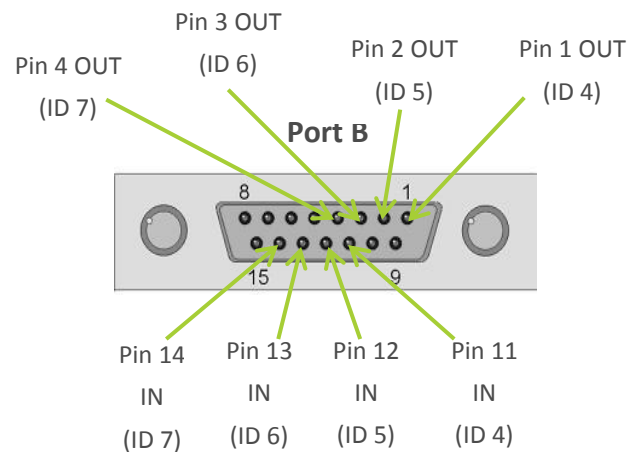
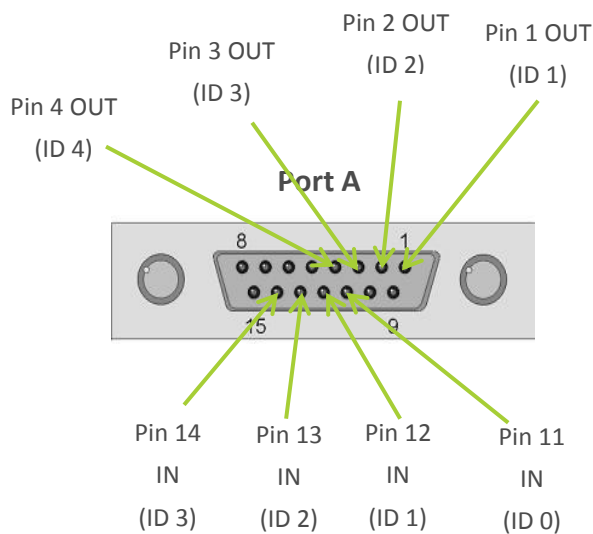
8.4.2 Pin Layout and Pin IDs in VisualApplets

There are two SUB-D15 female plugs on the trigger extension board, port **A** and port **B** (see also Figure 29 and Figure 30).

All trigger extension boards have 8 inputs and 8 outputs:

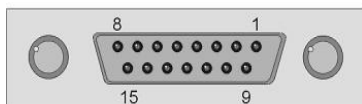
- ◆ 4 IN port A (indices 0-3)
- ◆ 4 OUT port A (indices 0-3)
- ◆ 4 IN port B (indices 4-7)
- ◆ 4 OUT port B (indices 4-7)

These signals are connected to the trigger inputs and outputs of the GPIO.



8.4.2.1 TTL Trigger Extension Board

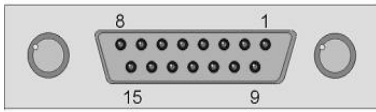
8.4.2.1.1 Pin Layout



Port A:

	SUB-D15 Port A	PIN_ID in Visual Applets
Pin #	Signal name	
1	Trigger Output 0 Port A <i>Flash Signal</i>	0
2	Trigger Output 1 Port A <i>For Area Cameras: Image Trigger (Ex Sync)</i> <i>For Line Cameras: Line Trigger (ExSync2)</i>	1
3	Trigger Output 2 Port A <i>For Area Cameras: HD Sync</i> <i>For Line Cameras: Line Trigger (Ex Sync)</i>	2
4	Trigger Output 3 Port A <i>User Output (DigitalOut Bit#0)</i>	3
5	nc	
6	GND	
7	nc	
8	+12V (for external devices only)	
9	+5V (for external devices only)	
10	nc	
11	Trigger Input 0 Port A	0
12	Trigger Input 1 Port A	1
13	Trigger Input 2 Port A	2
14	Trigger Input 3 Port A	3
15	GND	

Table 1: Pin Layout SUB-D15 female (Port A) on TTL trigger extension board



Port B:

	SUB-D15 Port B	PIN_ID in Visual Applets (single-ended signals)
Pin #	Signal name	
1	Trigger Output 4 Port B <i>Flash Signal</i>	4
2	Trigger Output 5 Port B <i>For Area Cameras: Image Trigger (Ex Sync)</i> <i>For Line Cameras: Line Trigger (ExSync2)</i>	5
3	Trigger Output 6 Port B <i>For Area Cameras: HD Sync</i> <i>For Line Cameras: Line Trigger (Ex Sync)</i>	6
4	Trigger Output 7 Port B <i>User Output (DigitalOut Bit#0)</i>	7
5	nc	
6	GND	
7	nc	
8	+12V (for external devices only)	
9	+5V (for external devices only)	
10	nc	
11	Trigger Input 4 Port B	4
12	Trigger Input 5 Port B	5
13	Trigger Input 6 Port B	6
14	Trigger Input 7 Port B	7
15	GND	

Table 2: Pin Layout SUB-D15 female (Port B) on TTL trigger extension board

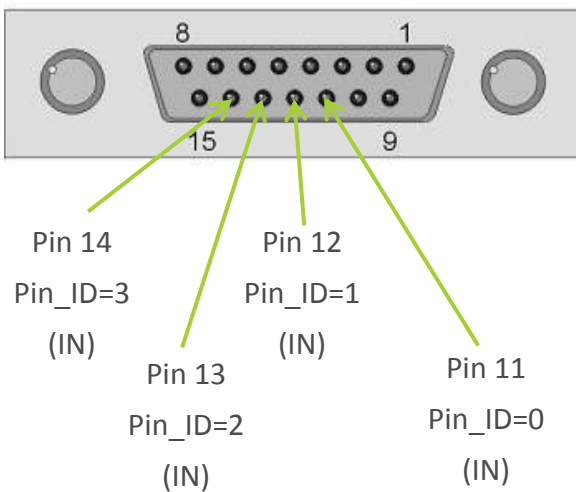
8.4.2.1.2 Trigger INPUT

If you

- ◆ are using the GPI operator (for defining trigger *input* signals),
- ◆ have set parameter *Connector Type* to **GPIO**,

the values of parameter *Pin_ID* address the following pins on the trigger extension board:

Port A:



Port B:

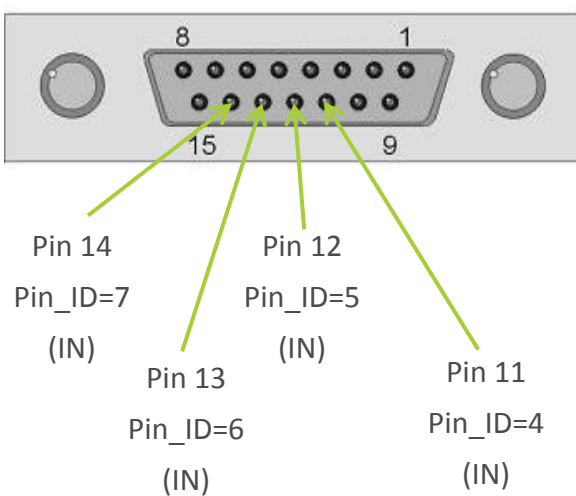


Figure 42: Pin IDs of operator GPI addressing physical pins of port A and B

To receive incoming signals on the IN pins of the TTL trigger extension board, in VisualApplets you have to set parameter Pin_ID of an GPI operator instance to the following values:

- ◆ The incoming signal on pin 11 port A is received within VisualApplets with a GPI operator instance with Pin_ID = 0.
- ◆ The incoming signal on pin 12 port A received within VisualApplets with a GPI operator instance with Pin_ID = 1.
- ◆ The incoming signal pin 13 port A is received within VisualApplets with a GPI operator instance with Pin_ID = 2.
- ◆ The incoming signal on pin 14 port A is received within VisualApplets with a GPI operator instance with Pin_ID = 3.
- ◆ The incoming signal on pin 11 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 4.
- ◆ The incoming signal on pin 12 port B received within VisualApplets with a GPI operator instance with Pin_ID = 5.
- ◆ The incoming signal pin 13 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 6.
- ◆ The incoming signal on pin 14 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 7.

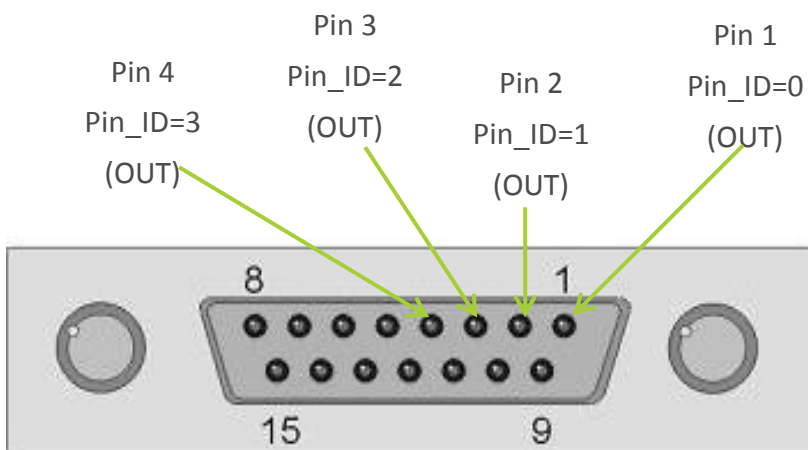
8.4.2.1.3 Trigger OUTPUT

If you

- ◆ are using the **GPO** operator (for defining trigger *output* signals),
- ◆ have set parameter *Connector Type* to **GPIO**,

the values of parameter *Pin_ID* address the following pins:

Port A:



Port B:

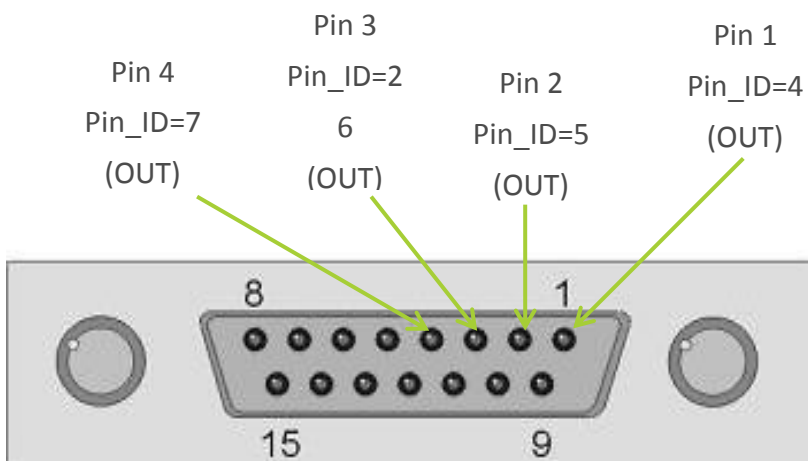


Figure 43: Pin IDs of operator GPO addressing physical pins of port A and B

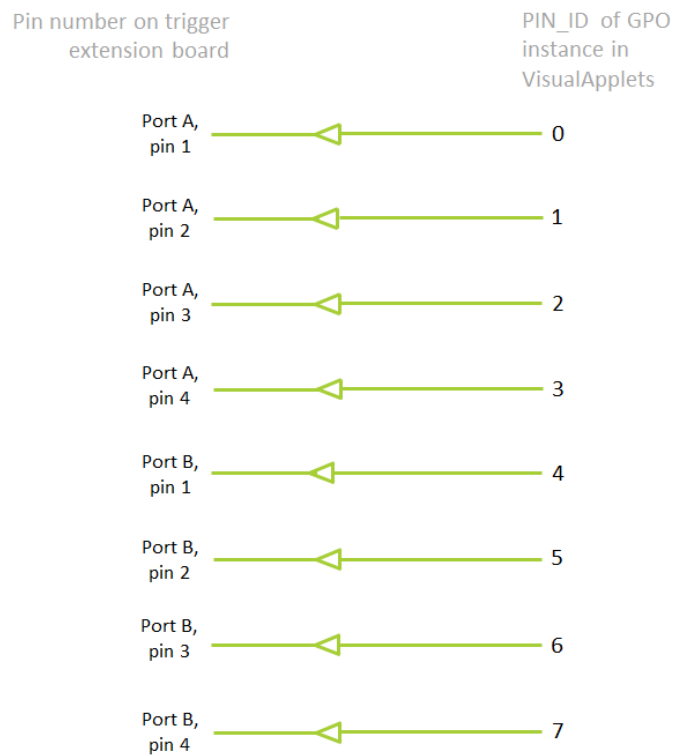
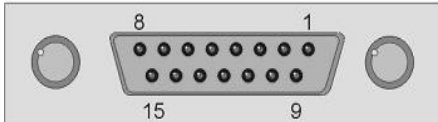


Figure 44: Outgoing Signals on Trigger Extension Board

8.4.2.2 Opto Trigger Extension Boards

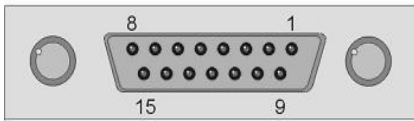
8.4.2.2.1 Pin Layout



Port A:

	SUB-D15 Port A	PIN_ID in Visual Applets (single-ended signals)	PIN_ID in Visual Applets (differential signals)
Pin #	Signal Name		
1	Trigger Output 0 Port A Flash Signal	0	
2	Trigger Output 1 Port A For Area Cameras: Image Trigger (Ex Sync) For Line Cameras: Line Trigger (ExSync2)	1	
3	Trigger Output 2 Port A For Area Cameras: HD Sync For Line Cameras: Line Trigger Line(Ex Sync)	2	
4	Trigger Output 3 Port A User Output (DigitalOut Bit#0)	3	
5	Voltage_IN (VCC Input Port A)		
6	GND Port A		
7	nc		
8	nc		
9	nc		
10	Voltage_IN (VCC Input Port A)		
11	Trigger Input 0 Port A (+ if used for diff. signal)	0	0
12	Trigger Input 1 Port A (- if used for diff. signal)	1	
13	Trigger Input 2 Port A (+ if used for diff. signal)	2	2
14	Trigger Input 3 Port A (- if used for diff. signal)	3	
15	GND Port A		



Table 3: Pin Layout SUB-D15 female (Port A) on Opto Trigger Extension Board



Port B:

	SUB-D15 Port B	PIN_ID in Visual Applets (single-ended signals)	PIN_ID in Visual Applets (differential signals)
Pin #	Signal Name		
1	Trigger Output 4 Port B Flash Signal	4	
2	Trigger Output 5 Port B For Area Cameras: Image Trigger (Ex Sync) For Line Cameras: Line Trigger (ExSync2)	5	
3	Trigger Output 6 Port B For Area Cameras: HD Sync For Line Cameras: Line Trigger Line(Ex Sync)	6	
4	Trigger Output 7 Port B User Output (DigitalOut Bit#0)	7	
5	Voltage_IN (VCC Input Port B)		
6	GND Port B		
7	nc		
8	nc		
9	nc		
10	Voltage_IN (VCC Input Port B)		
11	Trigger Input 4 Port B (+ if used for diff. signal)	4	4
12	Trigger Input 5 Port B (- if used for diff. signal)	5	
13	Trigger Input 6 Port B (+ if used for diff. signal)	6	6
14	Trigger Input 7 Port B (- if used for diff. signal)	7	
15	GND Port B		

Table 4: Pin Layout SUB-D15 female (Port B) on Opto Trigger Extension Board

	<p>Signal Type</p> <p>Which type of signal (differential or single-ended) is supported depends on the trigger board you are using. See our online documentation: Opto-decoupled Trigger Extension Boards or TTL Trigger Extension Board)</p>
	<p>Voltage In and GND required</p> <p>The marathon VF2 trigger system needs to get supply voltage on the Voltage IN and GND pins from an external source.</p>

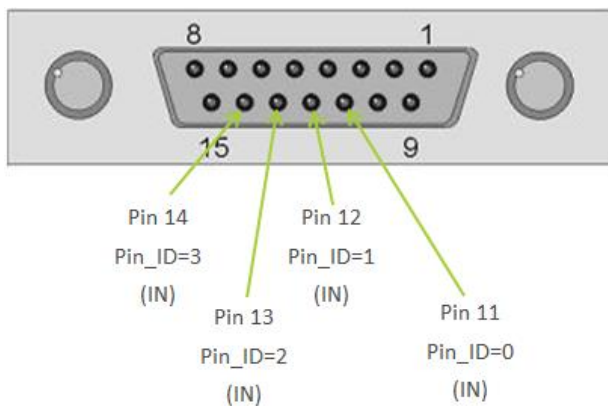
8.4.2.2.2 Trigger INPUT

If you

- ◆ are using the GPI operator (for defining trigger *input* signals),
- ◆ have set parameter *Connector Type* to **GPIO**,

the values of parameter *Pin_ID* address the following pins on the trigger extension board:

Port A:



Port B:

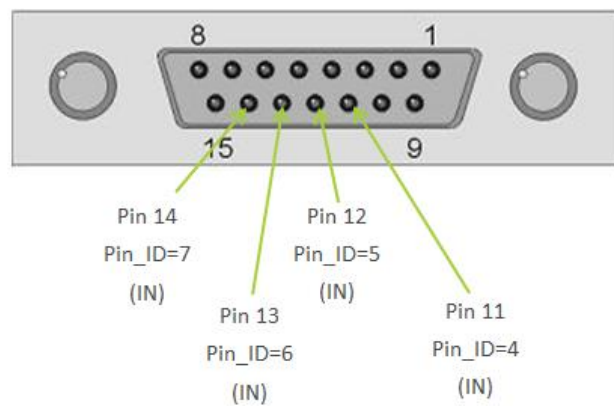
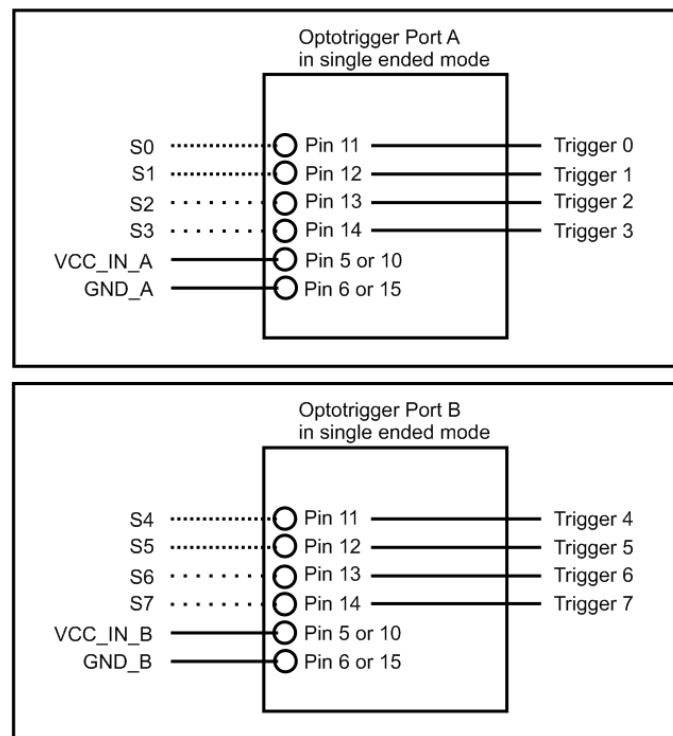


Figure 45: Pin IDs of operator GPI addressing physical pins of port A and B

8.4.2.2.3 Eight Single-Ended IN Signals – Individual Pins

If you configure trigger extension board **Opto Trigger 5** to receive 8 single-ended incoming signals (or if you use trigger extension board *Opto trigger board IV Classic*):

- ◆ The incoming signal on pin 11 port A is received within VisualApplets with a GPI operator instance with Pin_ID = 0.
- ◆ The incoming signal on pin 12 port A received within VisualApplets with a GPI operator instance with Pin_ID = 1.
- ◆ The incoming signal pin 13 port A is received within VisualApplets with a GPI operator instance with Pin_ID = 2.
- ◆ The incoming signal on pin 14 port A is received within VisualApplets with a GPI operator instance with Pin_ID = 3.
- ◆ The incoming signal on pin 11 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 4.
- ◆ The incoming signal on pin 12 port B received within VisualApplets with a GPI operator instance with Pin_ID = 5.
- ◆ The incoming signal pin 13 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 6.
- ◆ The incoming signal on pin 14 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 7.



8.4.2.2.4 Four Differential IN Signals – Individual Pins

If you configure trigger extension board **Opto Trigger 5** to receive 4 differential incoming signals (or if you use trigger extension board *Opto Trigger board IV DS*), you use the pin pairs 11/12 (GPI 0 /GPI 1) and 13/14 (GPI 2 / GPI 3) on port A and pin pairs 11/12 (GPI 4 /GPI 5) and 13/14 (GPI 6 / GPI 7) on port B for receiving differential signals.

Differential signals on Port A:

- ◆ You use pins 11 (GPI 0) and 12 (GPI 1) on port A for one differential signal. To receive this signal, you have to set the value Pin_ID of the receiving GPI operator instance (in VisualApplets) to Pin_ID = 0.
- ◆ You use pins 13 (GPI 2) and 14 (GPI 3) on port A for one differential signal. To receive this signal, you have to set the value Pin_ID of the receiving GPI operator instance (in VisualApplets) to Pin_ID = 2.



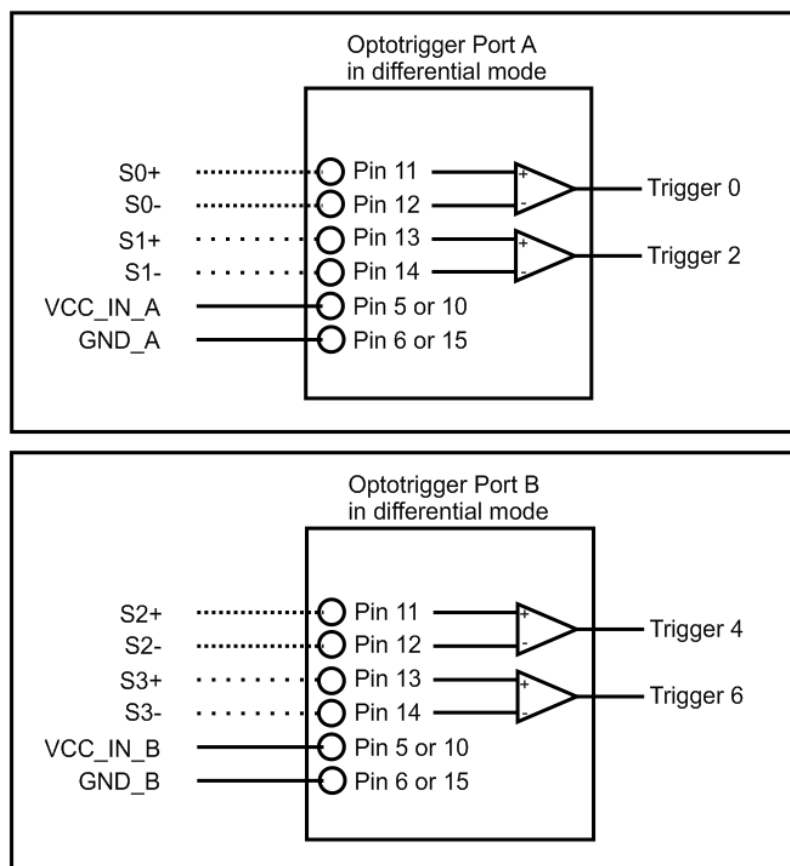
Don't use any GPI operator instances with Pin_ID = 1 or Pin_ID = 3, since the incoming signals in these cases are undefined.

Differential signals on Port B:

- ◆ You use pins 11 (GPI 4) and 12 (GPI 5) on port B for one differential signal. To receive this signal, you have to set the value Pin_ID of the receiving GPI operator instance (in VisualApplets) to Pin_ID = 4.
- ◆ You use pins 13 (GPI 6) and 14 (GPI 7) on port B for one differential signal. To receive this signal, you have to set the value Pin_ID of the receiving GPI operator instance (in VisualApplets) to Pin_ID = 6.



Don't use any GPI operator instances with Pin_ID = 5 or Pin_ID = 7, since the incoming signals in these cases are undefined.



8.4.2.2.5 Two Differential and Four Single-Ended IN Signals – Individual Pins

If you configure trigger extension board **Opto Trigger 5** to receive 2 differential and 4 single-ended incoming signals (or if you use trigger extension board *Opto trigger board IV DS/SE*):

Differential signals on Port A:

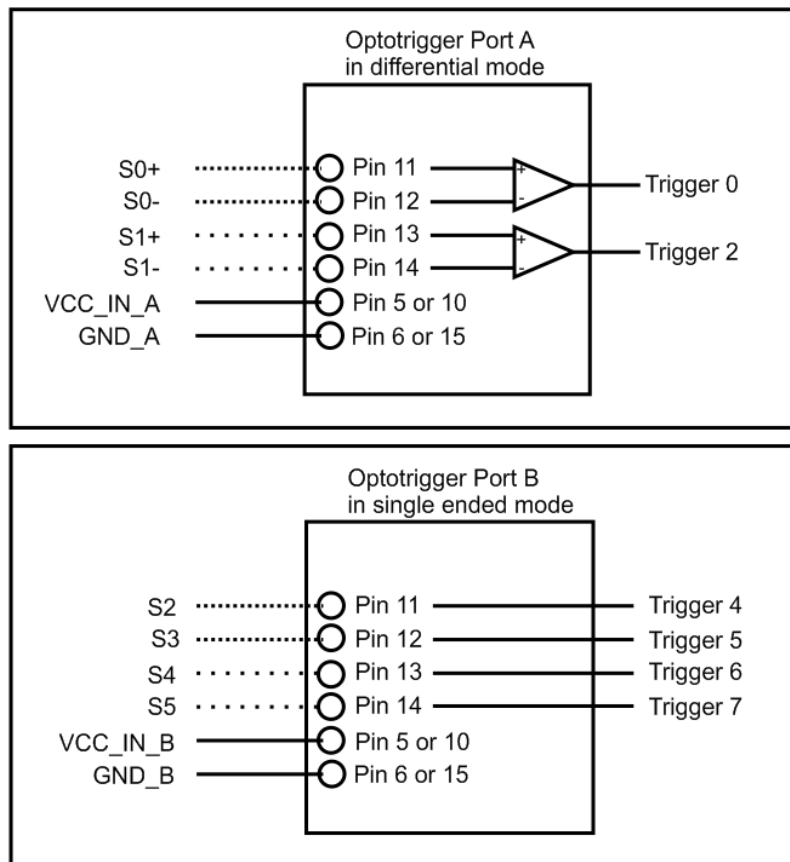
- ◆ You use pins 11 (GPI 0) and 12 (GPI 1) on port A for one differential signal. To receive this signal, you have to set the value Pin_ID of the receiving GPI operator instance (in VisualApplets) to Pin_ID = 0.
- ◆ You use pins 13 (GPI 2) and 14 (GPI 3) on port A for one differential signal. To receive this signal, you have to set the value Pin_ID of the receiving GPI operator instance (in VisualApplets) to Pin_ID = 2.



Don't use any GPI operator instances with Pin_ID = 1 or Pin_ID = 3, since the incoming signals in these cases are undefined.

Single-ended signals on Port B:

- ◆ The incoming signal on pin 11 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 4.
- ◆ The incoming signal on pin 12 port B received within VisualApplets with a GPI operator instance with Pin_ID = 5.
- ◆ The incoming signal pin 13 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 6.
- ◆ The incoming signal on pin 14 port B is received within VisualApplets with a GPI operator instance with Pin_ID = 7.



8.4.2.2.6 Trigger OUTPUT on the Trigger Extension Board

If you

- ◆ are using the **GPO** operator (for defining trigger *output* signals),
- ◆ have set parameter *Connector Type* to **GPIO**,

the values of parameter *Pin_ID* address the following pins:

Port A:

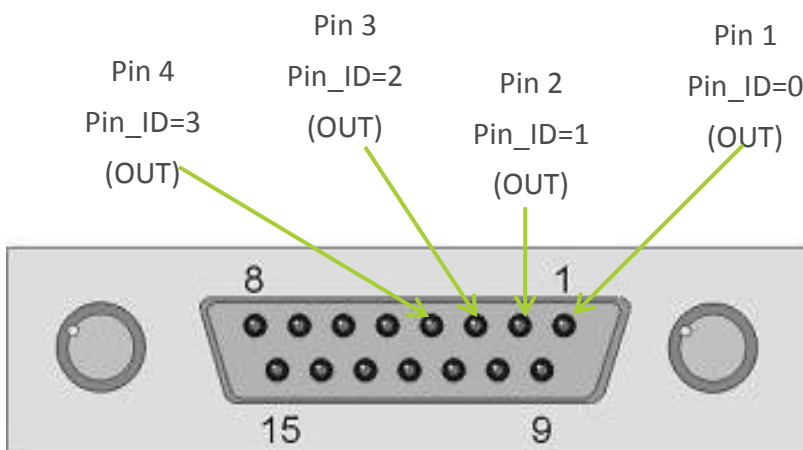


Figure 46: Pin IDs of operator GPO addressing physical pins of port A

Port B:

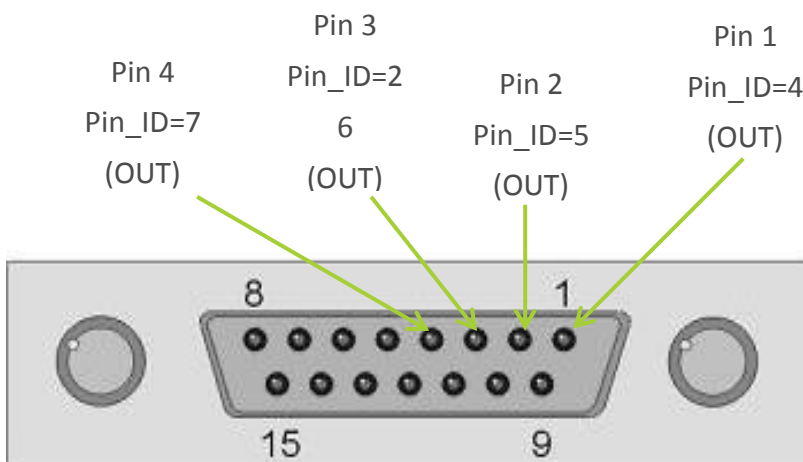


Figure 47: Pin IDs of operator GPO addressing physical pins of port B

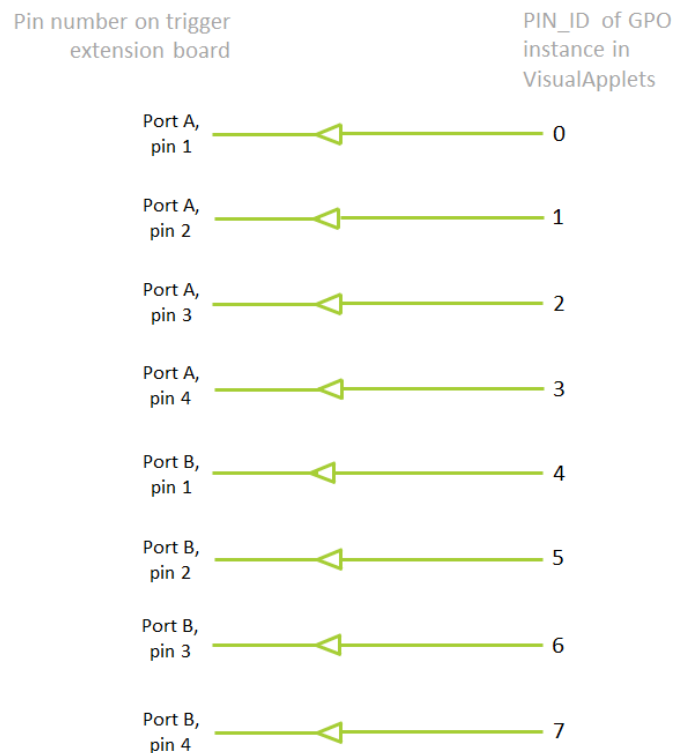


Figure 48: Outgoing signals on Trigger Extension Board



Important

For details on electrical and timing characteristics of the trigger extension boards, see our online documentation: [Opto-decoupled Trigger Extension Boards](#) or [TTL Trigger Extension Board](#))



8.5 Configuring the Trigger System

You can access the trigger configuration via two ways:

- ◆ **microDisplay⁸**: Here, you can enter trigger parameter values via GUI.
- ◆ **SDK**: For implementing the marathon VF2 programming interface into a specific image processing software application.

The demands on the trigger system vary, depending on your application:

- ◆ For **Area Scan**, only one trigger type (frame trigger) is required.
- ◆ For **Line Scan**, two trigger types are required: Line trigger (sends a trigger pulse for each line) and frame trigger. In a line scan application, the frame trigger controls how many lines are appended to an image.


	<p>Detailed Information on Trigger System</p> <p>For deeper information about the microEnable trigger system, refer to the Live documentation, sections Trigger System, Trigger Modes Area Scan and Trigger Modes Line Scan.</p>
	<p>Refer to Applet Documentation</p> <p>For information which parameters are available (and which effects specific settings may cause), refer to the documentation of the applet you are using.</p>

⁸ microDisplay is a tool that has come to you as part of the delivery package. microDisplay is part of the runtime environment of marathon VF2 and is automatically installed when you install the Silicon Software runtime software.

9 Appendix

9.1 Additional Procedures

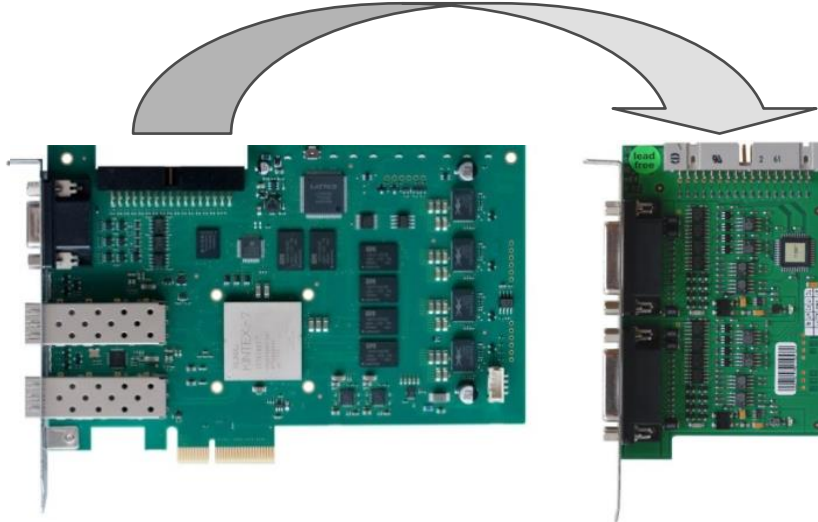
9.1.1 Installing a Trigger Extension Bboard

	<p>Writing Data Directly into the Camera</p> <p>Make sure you use an adequate ventilation system within your computer. This is of special importance if</p> <ul style="list-style-type: none">♦ there is little space between boards in a multi board installation♦ an installation is close to a graphics card. <p>We also recommend leaving enough free space between boards.</p>
---	---

You need the following components:

- ♦ One marathon VF2
- ♦ One trigger extension board
- ♦ One 34-pin flat cable you received from Silicon Software

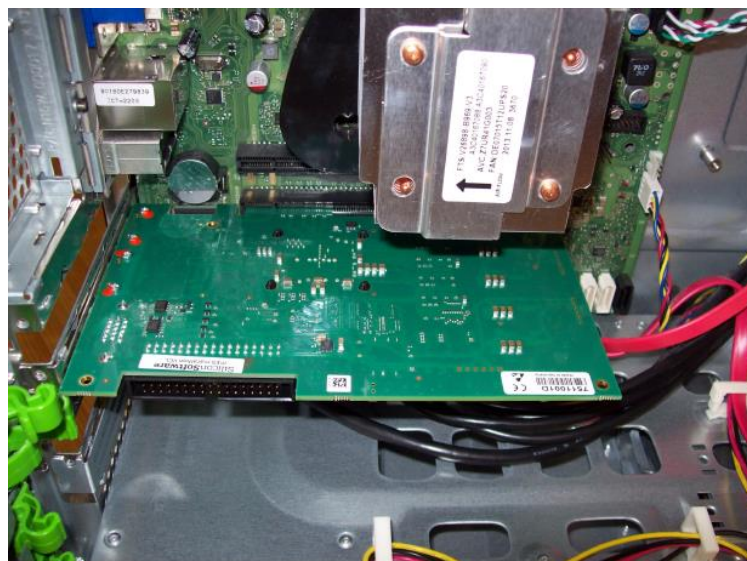
34-pin flat cable



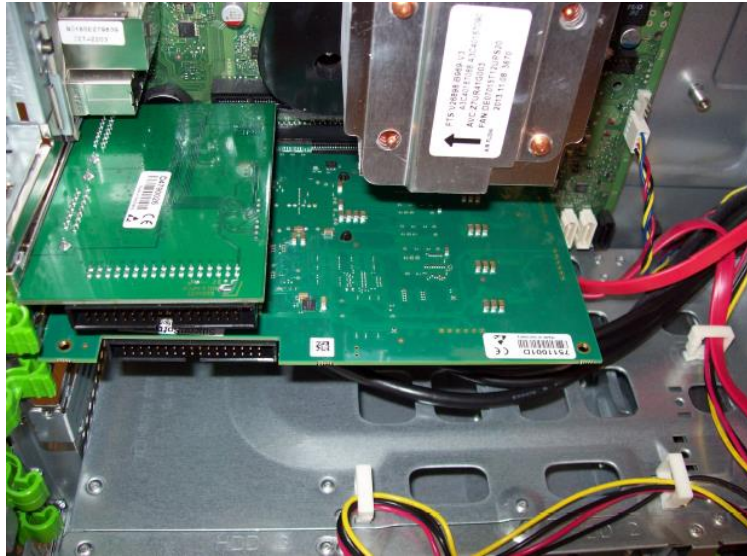
To install your trigger extension board:

1. Shut down the host PC.
2. Unplug the host PC from the power outlet.
3. Plug the trigger board into an empty slot on the slot bracket of the host PC.

The trigger board doesn't need a connection to the motherboard of the computer, so it is possible to use a PCIe, a PCI, or an ISA slot.



marathon frame grabber and empty slots in host PC



marathon frame grabber board and trigger board in host PC

4. Fasten the trigger board to the PC chassis by screws.
5. Connect the trigger board with the GPIO socket of marathon VF2, using the 34-pin flat cable.



marathon frame grabber and trigger board being connected via 34-pin flat cable

6. Reboot your host PC.

Now, the trigger board is ready for use.

9.1.2 Adding an Applet Manually

To add an applet you received as *.hap or *.dll file (and not in form of an installer) to your host file system:

1. Go into the runtime installation directory.
2. Create a subdirectory for the applet you have received from Silicon Software:
 - a) If you received an applet file with file extension *.hap, go into subdirectory *Hardware Applets* and create here the following directory:
[runtime installation directory]/Hardware Applets/mE5-MA-VF2
 - b) If you received an applet file with file extension *.dll, go into subdirectory *Dll* and create the following subdirectory:
[runtime installation directory]/Dll/mE5-MA-VF2
3. Get the *.hap or *.dll file you have received from Silicon Software.
4. Copy this file into the directory you just created:
 - a) *.hap files into subdirectory *Hardware Applets/mE5-MA-VF2*
 - b) *.dll files into subdirectory *Dll/mE5-MA-VF2*

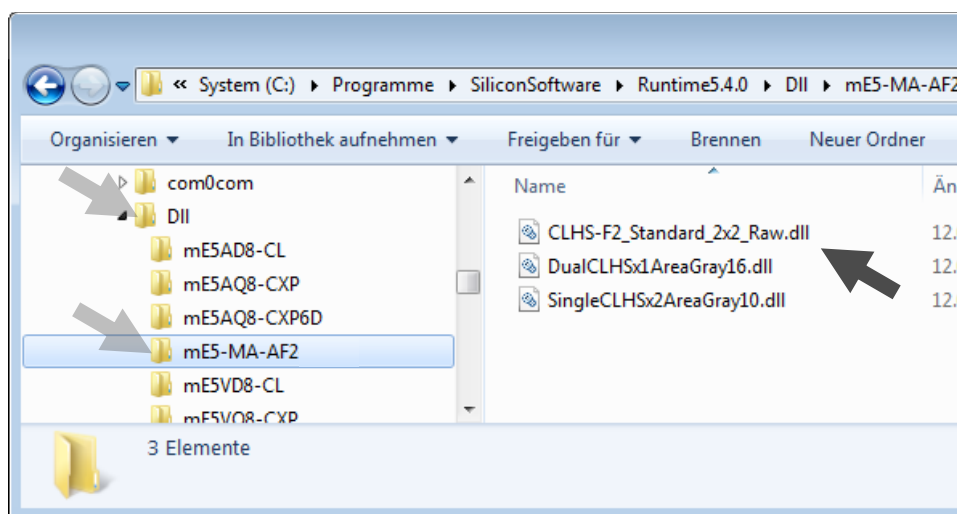


Figure 49: Copying an applet (*.dll) into the installation directory


Now, the applet can be flashed onto marathon VF2 as described in section [4.4 Installing an Applet onto marathon VF2 \(Flashing\)](#).

9.1.3 Checking on Installed Applet

Sometimes, you might need to know which applet is installed on marathon VF2.

To check which applet is installed on marathon VF2:

1. Start the tool microDisplay. (e.g., via Windows Start -> Programms -> SiliconSoftware -> Runtime5.x -> microDisplay)
2. In the dialog *I want to...*, select **Load Applet**.
3. In the *Load Hardware Applet* dialog, select the marathon VF2 device you want to get information about (left upper corner).
4. The currently installed applet is displayed in black, while all other applets are greyed out.

	<p>Self-explaining File Name</p> <p>For each combination of camera type (area scan, line scan, color, gray, CLHS, Camera Link Base/Medium/Full/Full 10 Tap ...) and link topology, a specific applet has to be installed on marathon VF2. The name of the applet file informs you which camera and topology are supported by the applet, e.g.:</p> <ul style="list-style-type: none"> ◆ Acq = acquisition applet ◆ line/area, color/gray = Information on the camera ◆ Last number before file name extension = Bit depth per pixel
---	---

9.1.4 Silent Installation Under Windows

The Runtime installer supports silent installation. The setup program accepts optional command line parameters. These can be useful for system administrators and other programs calling the setup program.

Setup Command Line Parameters

/SILENT, /VERYSILENT

Instructs the setup to be *silent* or *very silent*.

- Silent setup: The wizard and the background window are not displayed, but the installation progress window is visible on screen.
- Very silent setup: Wizard and background window are not displayed; even the installation progress window is not displayed.

Everything else is normal, e.g., error messages during installation are displayed, as well as the startup prompt (if you haven't disabled it with `DisableStartupPrompt` or the `'/SP-'` command line option).

If a restart is necessary and the `'/NORESTART'` command isn't used (see below):

- Silent setup: A "Reboot now?" message box is displayed.
- Very silent setup: The system reboots without asking.

/SUPPRESSMSGBOXES

Instructs the setup to suppress message boxes. This command line parameter has only an effect when combined with `'/SILENT'` or `'/VERYSILENT'`.

The used defaults are the following:

- 'Keep newer file?' **Yes**
- 'File exists, confirm overwrite.' **No**
- Abort/Retry: **Abort**
- Retry/Cancel: **Cancel**
- DiskSpaceWarning/ DirExists/ DirDoesntExist/ /NoUninstallWarning/ ExitSetupMessage/ ConfirmUninstall: **Yes (=continue)**
- FinishedRestartMessage/ /UninstalledAndNeedsRestart: **Yes (=restart)**

5 message boxes are not suppressible:

- The “About Setup” message box,
- The “Exit Setup?” message box, and
- The “FileNotInDir2” message box which is displayed when setup requires a new disk to be inserted and the disk was not found.
- Any (error) message box displayed before Setup (or Uninstall) could read the command line parameters.
- Any message box displayed by [Code] support function **MsgBox**.

/LOG="filename"

Same as /LOG, except that this parameter allows you to specify a fixed path/filename to use for the log file. If a file with the specified name already exists, it will be overwritten. If the file cannot be created, setup will abort with an error message.

/NORESTART

Instructs setup not to reboot even if ia reboot is necessary.

/DIR="x:\dirname"

Overrides the default directory name displayed on the *Select Destination Location* wizard page. A fully qualified pathname must be specified.

/GROUP="folder name"

Overrides the default folder name displayed on the *Select Start Menu Folder* wizard page. If the [Setup] section directive DisableProgramGroupPage was set to **yes**, this command line parameter is ignored.

/NOICONS

Instructs setup to initially check the *Don't create a Start Menu Folder* check box on the *Select Start Menu Folder* wizard page.

/COMPONENTS="comma separated list of component names"

Overrides the default component settings. Using this command line parameter causes the setup to automatically select a custom type. If no custom type is defined, this parameter is ignored.

Only the specified components will be selected; the rest will be deselected. If a component name is prefixed with a "*" character, any child components will be selected as well (except for those that include the dontinheritcheck flag). If a component name is prefixed with a "!" character, the component will be deselected.

This parameter does not change the state of components that include the fixed flag.

Usage:

<SISO_INSTALLER_EXE.exe> /Components = "<component1>,<component2>"

Available components:

Component	Description
core	Installation of core components (required)
tools_cli	Installation of command line tools
tools_gui	Installation of GUI tools
doc	Installation of documentation
gige	Support for GigE Vision frame grabber
dev\core	Installation of libs and header files
dev\examples	Installation of SDK examples
dev\examples_source	Installation of the source code of the examples
dev\examples_bin	Installation of example binaries
dev\cmake	Installation of cmake files
acq_applets	Installation of AcquisitionApplets
acq_applets\me4as1cl	Installation of frame grabber specific AcquisitionApplets
acq_applets\me4ad1cl	Installation of frame grabber specific AcquisitionApplets
acq_applets\me4ad4cl	Installation of frame grabber specific AcquisitionApplets
acq_applets\me4aq4ge	Installation of frame grabber specific AcquisitionApplets
acq_applets\me4vd1cl	Installation of frame grabber specific AcquisitionApplets
acq_applets\me4vd4cl	Installation of frame grabber specific AcquisitionApplets
acq_applets\me4vq4ge	Installation of frame grabber specific AcquisitionApplets
acq_applets\me5aq8cxp	Installation of frame grabber specific AcquisitionApplets
acq_applets\me5vq8cxp	Installation of frame grabber specific AcquisitionApplets
acq_applets\me5aq8cxp6d	Installation of frame grabber specific AcquisitionApplets

Component	Description
acq_applets\me5vq8cxp6d	Installation of frame grabber specific AcquisitionApplets
acq_applets\me5ad8cl	Installation of frame grabber specific AcquisitionApplets
acq_applets\me5vd8cl	Installation of frame grabber specific AcquisitionApplets
acq_applets\lightbridgeVF2	Installation of frame grabber specific AcquisitionApplets
acq_applets\me5maVF2	Installation of frame grabber specific AcquisitionApplets
acq_applets\me5maVF2dp	Installation of frame grabber specific AcquisitionApplets
advanced_acq_applets	Installation of Advanced AcquisitionApplets
advanced_acq_applets\acq me4aq4ge	Installation of frame grabber specific AcquisitionApplets
advanced_acq_applets\acq me4vq4ge	Installation of frame grabber specific AcquisitionApplets
advanced_acq_applets\acq me4ad4cl	Installation of frame grabber specific Advanced AcquisitionApplets
advanced_acq_applets\acq me4vd4cl	Installation of frame grabber specific Advanced AcquisitionApplets
UpdateEnvironment	Update of the environment variables
CompCLStandardVersion_2	Installation of CLser as defined in Camera Link 2.0
Com_0_Com	Installation of virtual null modem
bin_libs	Installation of libs into the system directory
redist_package	Installation of redistributable packages
lbservice	LightBridge Interface Service, supporting the LightBridge Interface board

Notes:

- Multiple components are applied by a comma separated list.
- The list may not contain any blanks.

/TASKS="comma separated list of task names"

Specifies a list of tasks that should be initially selected.

Only the specified tasks will be selected; the rest will be deselected. Use the /MERGETASKS parameter instead if you want to keep the default set of tasks and only select/deselect some of them.

If a task name is prefixed with a "*" character, any child tasks will be selected as well (except for those that include the dontinheritcheck flag). If a task name is prefixed with a "!" character, the task will be deselected.

Usage:

<SISO_INSTALLER_EXE.exe> /Tasks = "<task1>,<task2>"

Available tasks:

Task	Description
taskDesktopicon	installs a desktop icon
taskDrvInstall64	update of the device drivers

Example for a silent installation:

```
RuntimeSetup_v5.2.2_Win64.exe  
/components=core,tools_cli,acq_applets\me4ad1cl,acq_applets\me4vd1  
cl /silent
```

9.1.5 Resetting the Global Settings in microDisplay

To change microDisplay's *Settings* Dialog:

1. Select **Tools** -> **Settings**.

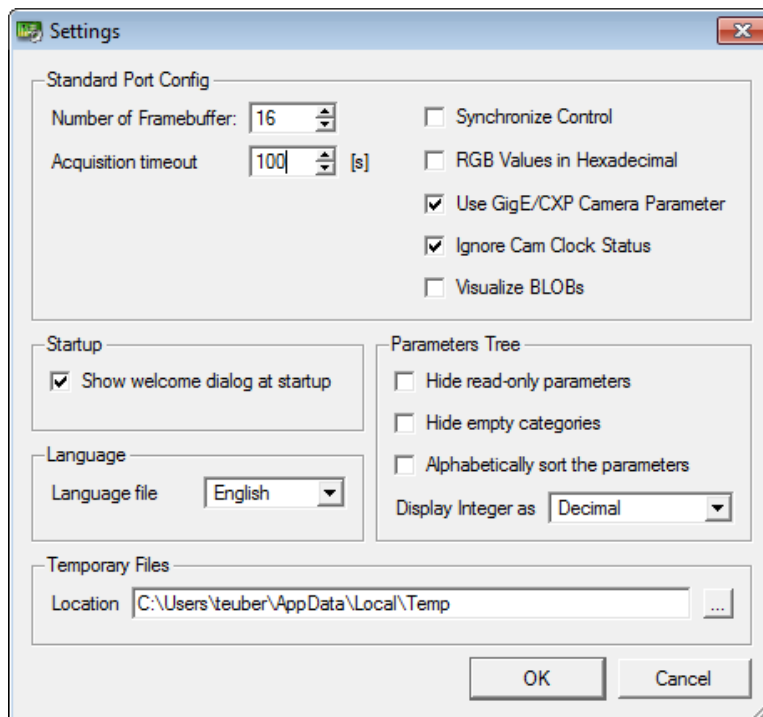


Figure 50: Global Settings Dialog in microDisplay

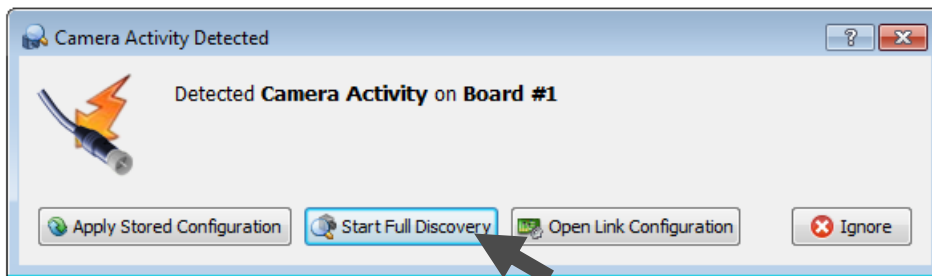
2. Make your changes as required.
3. Click the **OK** button.

9.1.6 Camera and Topology Configuration

9.1.6.1 Starting Link Topology Detection Manually

If the current link topology cannot be detected, or after you changed cable connections, one of the following two situations will appear. Proceed as described below to discover the camera and the link topology.

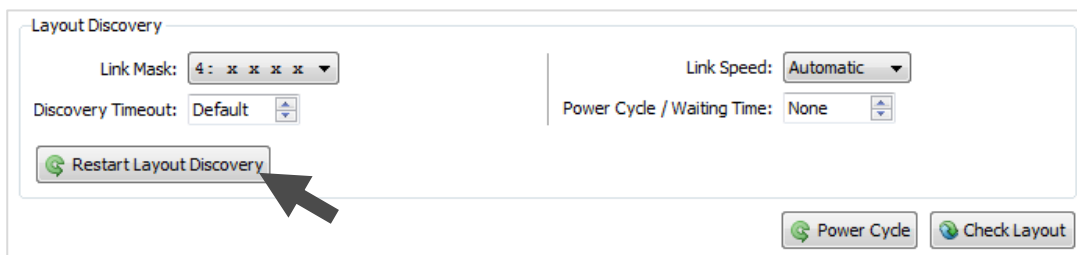
- a) The following dialog appears. In this case, simply click on **Start Full Discovery**.



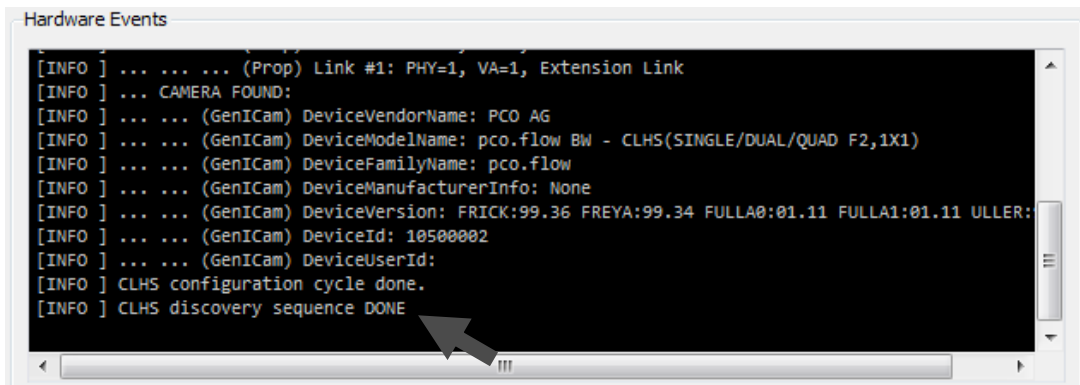
- b) There is no dialog.

In this case, to discover the current link topology:

1. Open the *GenICam Explorer*.
2. On the **Tools** menu, select **Hardware Setup**. The *Hardware Setup* dialog opens.
3. In the left upper corner, select the frame grabber you are working with.
4. Go to the **Link Configuration** tab.
5. Click on **Restart Layout Discovery**.



6. Wait until the process is finished. You get an according message:



The current link topology is displayed now.

9.1.6.2 Using an External XML File

Alternatively, you can also use an external XML file to configure the camera. In this case, you load the XML file from your file system and not from the camera.

To load an external XML file into the GenICam Explorer and on the camera:

1. In the GenICam Explorer, go to the **Connection** tab.
2. Activate the radio button **User Supplied GenICam XML File**.
3. Select the GenICam XML file you want to use. (Use only files supplied by the camera vendor).
4. Click on **Connect**.

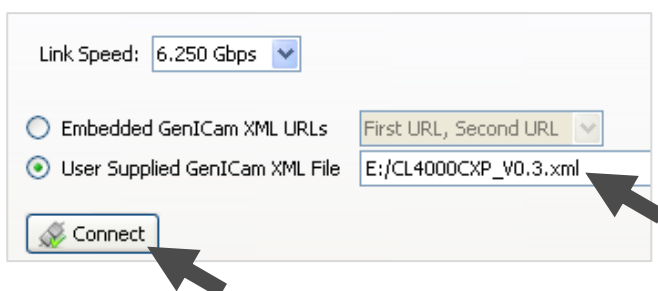


Figure 51: Loading an external XML file

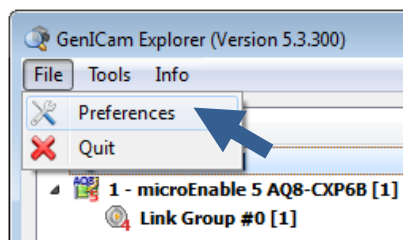
It might take some seconds to load the file. The parameters of the GenICam interface with current settings are displayed. You are ready to start the actual camera configuration.

5. Proceed with step [2](#) in section [4.2.2 Configuring the Camera](#).

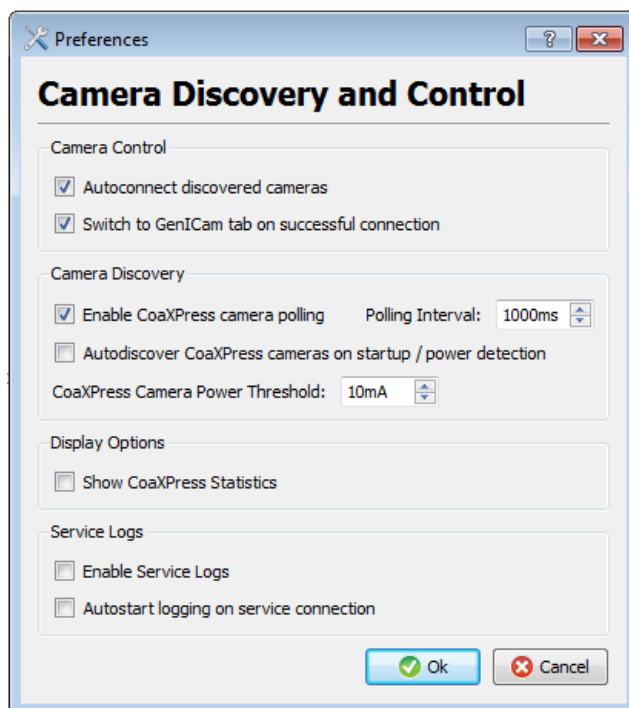
9.1.6.3 Configuring the Program Behavior of the GenICam Explorer at Program Start

To configure the program behavior at program start:

1. In the **File** menu, select **Preferences**.



2. Select the options according to your needs:




If you want to go on with getting your camera ready, proceed with section [4.2.1 “You see the current status of the camera discovery ...”](#).

9.1.7 Multi-Board Usage

You have various options to connect multiple marathon VF2 boards and/or multiple trigger extension boards:

- ◆ Connecting 1 trigger board (outputs are wired-ORed) to multiple marathon VF2 boards (synchronization),
- ◆ Connecting several trigger boards (inputs are wired-ORed) to marathon VF2, thus you can use more GPIOs than available on one single trigger board,
- ◆ Connecting multiple marathon VF2 boards without trigger board: In this case, one master marathon VF2 board synchronizes all its slave marathon VF2 boards.

marathon VF2 (or multiple marathon VF2 devices) is/are connected to the trigger board(s) by a specific 34-pin flat cable. The signals of this flat cable use open collector drivers.

	<p>Attention</p> <ol style="list-style-type: none"> 1. Do NOT connect marathon VF2 devices residing in different PCs. 2. To connect to several marathon VF2 devices, a <i>custom-made single flat cable</i> with a connector for the trigger board and multiple connectors for the marathon VF2 devices must be used. Please contact the Silicon Software sales department for ordering information.
---	---

Using one trigger extension board for several marathon VF2 devices makes synchronizing the marathon VF2 devices most easy. By sharing the trigger input lines of the trigger extension board, all marathon VF2 devices in the PC are accurately synchronized. There is no limit how many marathon VF2 devices can be connected to one trigger extension board. However, all connected marathon VF2 devices must be seated in the same PC. Commonly, the trigger extension board supports all marathon VF2 devices in one PC.

The scheme described above is useful if the acquisition is controlled by an external generator. However, there are applications where no external source is available or desired. In this cases, a master marathon VF2 device needs to synchronize all its slave marathon VF2 devices.

9.2 Trouble-Shooting

9.2.1 Applet Installed on marathon VF2 Cannot be Loaded into microDisplay

If you expect a certain applet to be available because you installed it on marathon VF2 as described in section [4.4](#) Installing an Applet onto marathon VF2 (Flashing), but it is nevertheless grayed out:

1. Use the tool tip information provided in microDisplay to get information why it is not available:

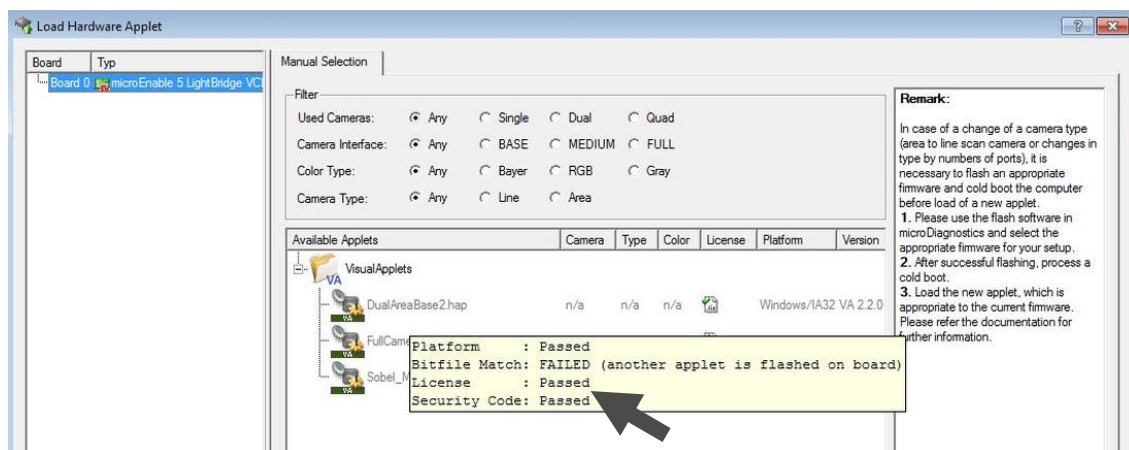


Figure 52: Tool tip informing why applet cannot be loaded in microDisplay

2. Open microDiagnostics and load the applet you want to use once again onto marathon VF2 (see section [4.4](#) Installing an Applet onto marathon VF2 (Flashing)).

9.2.2 Loading a New Applet (Flashing) gets not Completed

If you experience problems while loading an applet onto marathon VF2, i.e., if the loading (flashing) process doesn't come to an end:

1. Keep marathon VF2 powered. DON'T switch off marathon VF2.
2. Call the Silicon Software Support department.

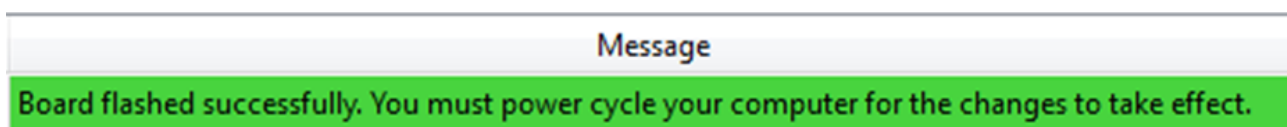
9.2.3 Disabling FPGA Live Reconfiguration

You can always disable the live reconfiguration of the FPGA. Afterwards, you will need to power cycle (cold-boot) your host PC after flashing the frame grabber board

To disable the live reconfiguration of the FPGA:


1. Add the following system environment variable to your system:
SISO_ENABLE_RECONFIGURATION
2. Set it to "NO": **SISO_ENABLE_RECONFIGURATION=NO**

After each flashing, you will now be prompted to power cycle (cold-boot) your host PC:



To power cycle your host PC:

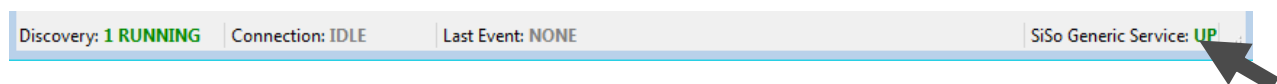
1. Click **Shut Down** to shut down your host PC completely (the Restart option is not enough).
2. After the computer is completely off, wait for some seconds.
3. Start the host PC again.

	<p>Complete Shut Down Essential</p> <p>For power cycling, it is not enough use the <i>Restart</i> option of Windows. Complete shut down and following new start are essential when you need to power cycle your host PC.</p>
---	---

9.2.4 Starting Generic Service

If there is no board displayed in the GenICam Explore after starting the GenICam Explorer, make sure the Silicon Software Generic Service (GS) is started.

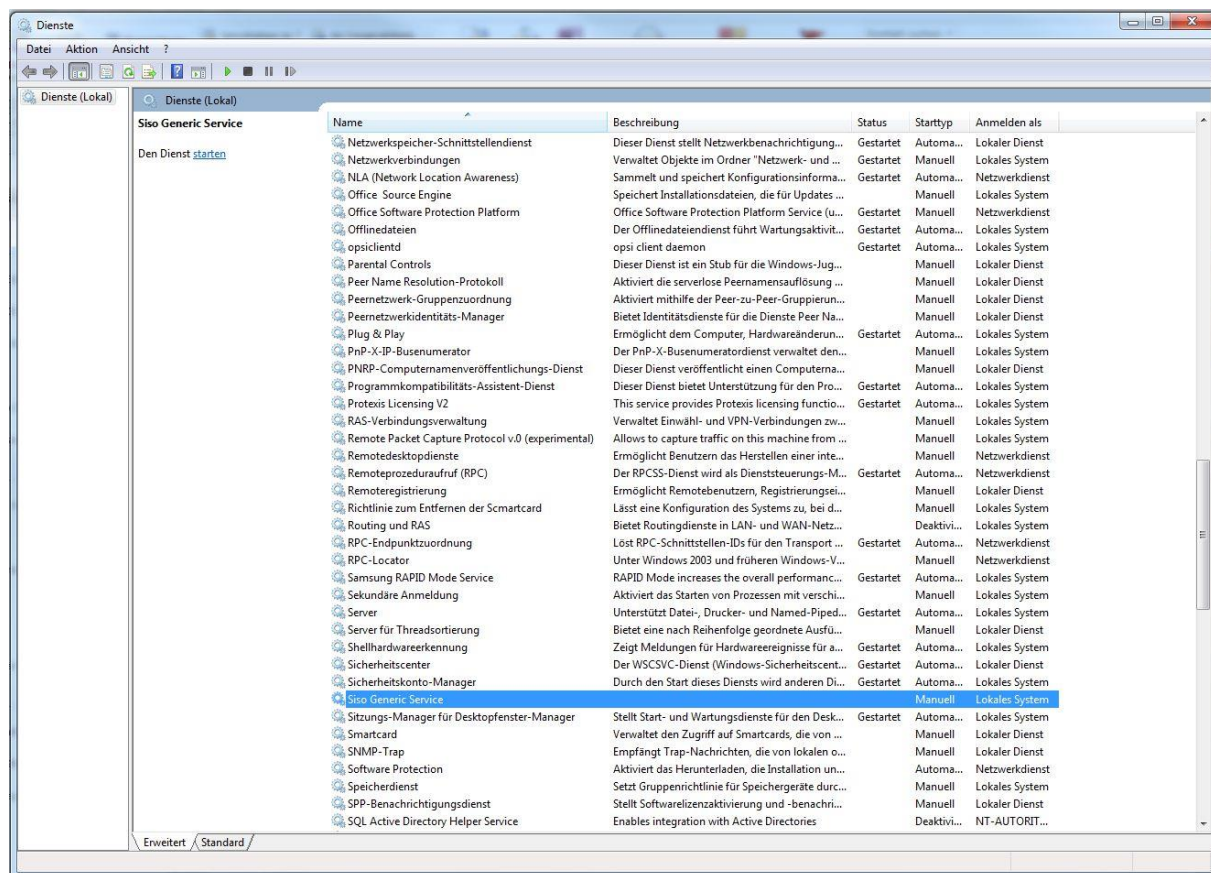
If the service is started you get an according message in GenICam explorer's task bar:



Enabling Automatic Start of Generic Service at Windows Start

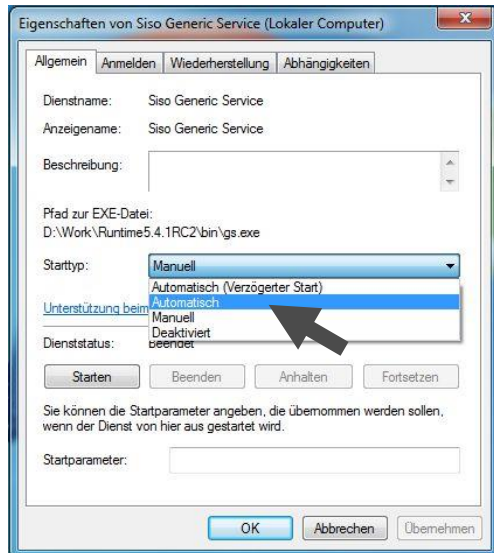
1. Open the Windows *Start* menu.
2. Enter "Services" (in the language you have selected for Windows OS).

The following window listing the available services opens:



3. Select „Siso Generic Service“ out of the list.

4. Right-click on the list entry. The windows *Properties* opens:



5. Under Start Type, select “Automatic”.

Now, the service is started automatically at each start of Windows OS.

Starting the Generic Service Once via Command Line

To start the Silicon Software Generic Service (GS):

1. Open the Windows command shell (cmd).
2. Go to your runtime installation directory and here in the subdirectory bin:

```
cd %SISODIR5%/bin
gs run
```

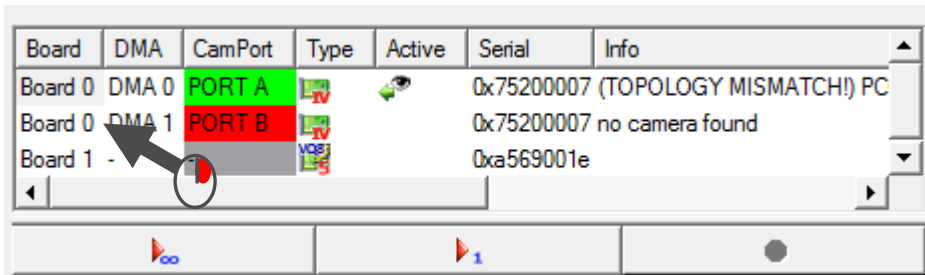
(Under Linux, use command `./gs run`)

Now, the Silicon Software generic service is started. As soon as you close the Windows command shell, the generic service is disabled again.

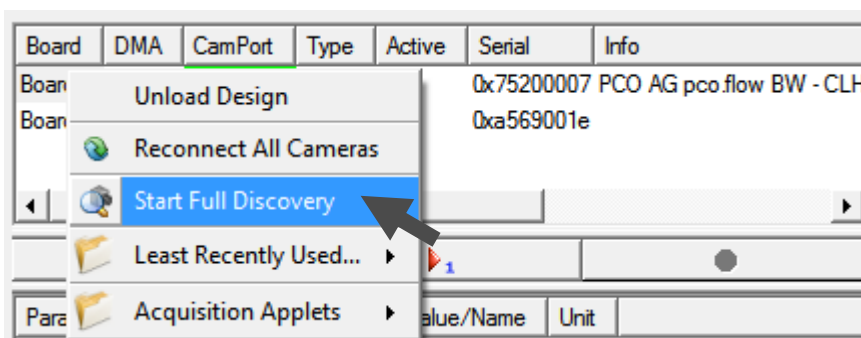
9.2.5 Topology Mismatch

If you get the message “Topology Mismatch” in microDisplay:

1. Right-click on the board that shows the problem:



2. From the context menu, select *Start Full Discovery*:

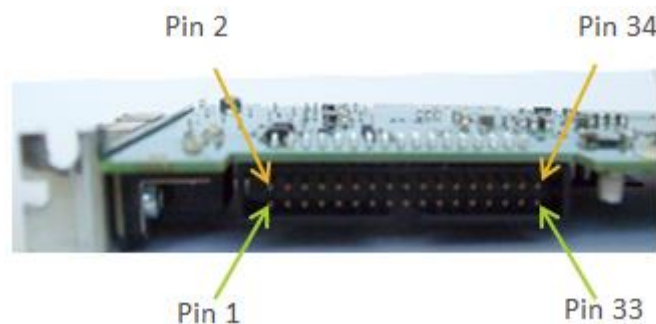


9.3 Pin Layout GPIO (On-Board Interface)



Use GPIO via Trigger Board

Keep in mind that the GPIO connector is designed for connecting the frame grabber to a trigger extension board.



The pins of the 34-pin flat cable connector are connected to the following inputs and outputs:

Odd Pin Numbers

Pin Number	I/O Name
1	Trigger Output 0
3	Trigger Output 1
5	Trigger Output 2
7	Trigger Output 3
9	Trigger Input 0
11	Trigger Input 1
13	Trigger Input 2
15	Trigger Input 3
17	Trigger Output 4
19	Trigger Output 5
21	Trigger Output 6
23	Trigger Output 7
25	Trigger Input 4
27	Trigger Input 5
29	Trigger Input 6
31	Trigger Input 7
33	Presence Detect

Even Pin Numbers

Pin Number	I/O Name
2	+3.3 V
4	+3.3 V
6	GND
8	GND
10	GND
12	GND
14	GND
16	GND
18	GND
20	GND
22	GND
24	GND
26	GND
28	GND
30	GND
32	VCCIO (+2.5 V / 3.3 V)*
34	VCCIO (+2.5 V / 3.3 V)*


**Attention**

The marathon VF2 trigger system needs to get supply voltage on the ***Voltage IN*** pins. If you want to connect devices that have no PWR pin, you need to provide the power supply to the ***Voltage IN*** pin from an external source.

9.4 Where to Find Further Documentation

Silicon Software provides deep and comprehensive documentation for its frame grabber series.

The documentation is part of the installation package. After installation of the runtime software package, you find the documentation in the Windows start menu.

	<p>Where to find the Documentation</p>
	<p><i>START -> All Programs -> SiliconSoftware -> Runtime 5.x -> Documentation</i></p>

The most relevant information for running your frame grabber for the first time is:



Product Documentation Site

Welcome

Dear Customer,

The Silicon Software Documentation is intended to provide deep and complete information about the Silicon Software frame grabber products - from installation to usage. It covers the frame grabber series microEnable 5 CXP, microEnable 5 Camera Link, microEnable IV Camera Link, microEnable IV GigE Vision, and microEnable IV LVDS, as well as all add-ons [\[more\]](#).



Figure 53: Documentation set for Frame Grabbers & Runtime

You get the following information:

1) Introduction and Installation

Quick start Guides for microEnable IV and microEnable 5 frame grabbers, containing all information you need to get your system running:

- Installation of hardware and runtime software
- All steps required to start image acquisition
- Applets guide that makes it easy to find the optimal applet for your specific image acquisition system (microEnable 5 ironman only)

2) Image Acquisition

Information on

- How frame grabber applets work
- Which applets (supporting specific image acquisition and processing functions) are available for individual frame grabber models
- How to set up an image acquisition
- Trigger boards and how to use them

3) Frame Grabber Hardware

Information on

- Individual frame grabber boards
- Accessories like external trigger boards

4) Software Development Kit (API)

- Introduction to this powerful image acquisition library
- SDK Manual
- SDK Reference
- SDK examples as a quick and simple starting point for your own C/C++ projects

5) Tools

Information on

- Camera configuration with *GenICam Explorer*
- Image acquisition with *microDisplay*
- Functionality of *microDiagnostics*

9.5 Additional Applets and Patches

Our products are under continuous development. New applets, providing advanced new features, are constantly added to our portfolio to meet the needs of our customers.

As the microEnable 5 frame grabber family is constantly improved and enhanced, there are also some patches available.

If you want to get information on new applets and patches in advance, or if you want to get these enhancements prior to the next release, feel free to contact our Silicon Software support team.

9.6 Support

For technical support please contact our support team:

<mailto:support@silicon-software.de>

Phone: +49 621 789 50 70

Contact Details

Silicon**Software** GmbH

Steubenstrasse 46

D - 68163 Mannheim, Germany

Phone: +49(0)621.789 507 0

Fax: +49(0)621.789 507 10

Email: info@silicon.software

Web: www.silicon.software

Silicon**Software** Inc.

1 Tara Boulevard, Suite 200

Nashua, NH 03062, USA

Phone: +1 603 324 7172

Fax: +1 603 324 7101

Email: info@silicon.software

Web: www.silicon.software

Disclaimer

While every precaution has been taken in the preparation of this manual, Silicon Software GmbH assumes no responsibility for errors or omissions. Silicon Software GmbH reserves the right to change the specification of the product described within this manual and the manual itself at any time without notice and without obligation of Silicon Software GmbH to notify any person of such revisions or changes.

Trademarks

All trademarks and registered trademarks are the property of their respective owners.

Copyright Note

© Copyright 2016 Silicon Software GmbH. All rights reserved. This document may not in whole or in part, be reproduced, transmitted, transcribed, stored in any electronic medium or machine readable form, or translated into any language or computer language without the prior written consent of Silicon Software GmbH.