GARZ FRICKE

Embedded Systems

Garz & Fricke

Embedded Computer Systems

Flash-N-Go · User Manual

Version 15.1-r8673
Built on 24.09.2019

freescale Enabled

Zuverlässige Qualität Made in Germany

Important hints

Thank you very much for purchasing a Garz & Fricke product. Our products are dedicated to professional use and therefore we suppose extended technical knowledge and practice in working with such products.

The information in this manual is subject to technical changes, particularly as a result of continuous product upgrades. Thus this manual only reflects the technical status of the products at the time of printing. Before design-in the device into your or your customer's product, please verify that this document and the therein described specification is the latest revision and matches to the PCB version. We highly recommend contacting our technical sales team priorto any activity of that kind. A good way getting the latest information is to check the release notes of each product and/or service. Please refer to the chapter [► 6 Related documents and online support].

The attached documentation does not entail any guarantee on the part of Garz & Fricke GmbH with respect to technical processes described in the manual or any product characteristics set out in the manual. We do not accept any liability for any printing errors or other inaccuracies in the manual unless it can be proven that we are aware of such errors or inaccuracies or that we are unaware of these as a result of gross negligence and Garz & Fricke has failed to eliminate these errors or inaccuracies for this reason. Garz & Fricke GmbH expressly informs that this manual only contains a general description of technical processes and instructions which may not be applicable in every individual case. In cases of doubt, please contact our technical sales team.

In no event, Garz & Fricke is liable for any direct, indirect, special, incidental or consequential damages arising out of use or resulting from non-compliancy of therein conditions and precautions, even if advised of the possibility of such damages.

Before using a device covered by this document, please carefully read the related hardware manual and the quick guide, which contain important instructions and hints for connectors and setup.

Embedded systems are complex and sensitive electronic products. Please act carefully and ensure that only qualified personnel will handle and use the device at the stage of development. In the event of damage to the device caused by failure to observe the hints in this manual and on the device (especially the safety instructions), Garz & Fricke shall not be required to honour the warranty even during the warranty period and shall be exempted from the statutory accident liability obligation. Attempting to repair or modify the product also voids all warranty claims.

Before contacting the Garz & Fricke support team, please try to help yourself by the means of this manual or any other documentation provided by Garz & Fricke or the related websites. If this does not help at all, please feel free to contact us or our partners as listed below. Our technicians and engineers will be glad to support you. Please note that beyond the support hours included in the Starter Kit, various support packages are available. To keep the pure product cost at a reasonable level, we have to charge support and consulting services per effort.

Shipping address:
Garz & Fricke GmbH
Tempowerkring 2
21079 Hamburg
Germany

Support contact:
Phone +49 (0) 40 / 791 899 - 30
Fax +49 (0) 40 / 791 899 - 39
Email ► support@garz-fricke.com
URL ► www.garz-fricke.com

Contents

Important hints **2**

1 Introduction **4**

2 Overview **5**
2.1 The bootloader 5
2.2 The Linux kernel 5
2.3 The root file system 5
2.4 The partition layout 5
2.5 Further information 6

3 Flash-N-Go Boot **7**
3.1 Boot-Script Commands 7
3.1.1 Comments 7
3.1.2 Load Command 7
3.1.3 Devtree Command 8
3.1.4 Exec Command 8
3.1.5 Testram Command 9
3.1.6 Testbench Command 9

4 Flash-N-Go System **10**
4.1 Booting Flash-N-Go System 10
4.2 Configuration tools 10
4.2.1 bootselect 10
4.2.2 sconfig 11
4.2.3 xconfig 11
4.3 Manual System Update 13
4.3.1 Example 1: GUF-Yocto V15.0-r4523-9 for SANTARO 14
4.3.2 Example 2: Windows Embedded Compact 7 V3.0r477-0 for SANTARO 14
4.3.3 Example 3: Upgrade to Flash-N-Go System 8.0 15
4.3.4 Example 4: Upgrade Flash-N-Go Boot to version 9.0 15
4.3.5 Example 5: Android 7.1.1 guf-4.0-0 for SANTARO via Internet 16
4.3.6 Example 5: Android 7.1.1 guf-4.0-0 for SANTARO via USB stick 16
4.4 Security Considerations 17
4.4.1 Hardware Access 17
4.4.2 Services 17
4.4.3 Serial Console 17
4.4.4 Password 17
4.4.5 Autojob 18

5 Flash-N-Go Update **20**
5.1 Quickstart 20
5.1.1 Network Setup 20
5.1.2 Flash-N-Go Update GUI elements 21
5.1.3 Installation 21
5.1.4 Reboot the system 22

6 Related documents and online support **24**

A GNU General Public License v2 **25**
A.1 Preamble 25
A.2 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION 25
A.3 END OF TERMS AND CONDITIONS 28
A.3.1 How to Apply These Terms to Your New Programs 28

B Standard Device Configuration **30**
B.1 Operating System 30
B.2 Bootloader 30
B.3 Boot Logo 30
B.4 Serial diagnostic port 30

B.5 IPv4 Settings 30
B.6 Services 30
B.7 Display 31

1 Introduction

Garz & Fricke systems can be used with an adapted version of several operating systems like Linux, Windows Embedded or Android. The corresponding board support package (BSP) as provided by Garz & Fricke is based on extensions by Freescale. Furthermore, Garz & Fricke has made several modifications and extensions to the BSPs. Nevertheless, the full source code usually is available as a board support package (BSP) from Garz & Fricke.

A Garz & Fricke device normally comes with a pre-installed Garz & Fricke operating system. If a BSP is provided by Garz & Fricke the user is able to build the complete BSP from source, modify it according to his needs and replace the pre-installed operating system with a custom one. Beyond the OS each platform needs its corresponding bootloader like RedBoot, U-Boot or Flash-N-Go-Boot. Usually the sources of the bootloaders are not provided by the Garz & Fricke BSPs. The bootloader is responsible for loading the kernel of the desired operating system into memory and provide the kernel with information it needs to work correctly. In most cases it holds possibilities to configure the target system.

This manual contains information about the usage of the Garz & Fricke **Flash-N-Go** solution which consists of three submodules:

- **Flash-N-Go Boot**: A proprietary small and fast bootloader of Garz & Fricke GmbH. This bootloader usually boots directly to the target OS or optionally into **Flash-N-Go System**.
- **Flash-N-Go System**: A RAM-disk-based slim Linux system based on PTXdist. Its scope of operation includes flashing, configuring and testing the device.
- **Flash-N-Go Update**: A proprietary software package for automatic updates of the operating system on the device and/or its **Flash-N-Go** software.

In addition to this manual, please also refer to the dedicated hardware manuals which can be found on the Garz & Fricke website as well.

2 Overview

A Garz & Fricke Flash-N-Go System generally consists of four basic software components:

- the bootloader
- the Linux kernel
- the root file system
- the device configuration

These software components are usually installed on separate partitions on the backing storage of the embedded system.

Newer Garz & Fricke devices are shipped with a separate small ramdisk-based Linux system called **Flash-N-Go System** which is installed in parallel to the main operating system. The purpose of Flash-N-Go is to provide the user a comfortable and secure update mechanism for the main operating system components.

2.1 The bootloader

There are several bootloaders available for the various Linux and Windows platforms. For desktop PC Linux systems, GRUB or LILO are commonly used. Those bootloaders are started by hardwired PC-BIOS.

Embedded Systems do not have a PC-like BIOS. In most cases they are started from raw flash memory or an eMMC device. For this purpose, there are certain open source boot loaders available, like RedBoot, U-Boot or Barebox. Furthermore, Garz & Fricke provides its own bootloader called **Flash-N-Go Boot** for its newer platforms (e.g. SANTARO and SANTOKA).

2.2 The Linux kernel

The Linux OS kernel includes the micro kernel specific parts of the Linux OS and several internal device and subsystem drivers.

2.3 The root file system

The root file system for **Flash-N-Go System** is a RAM based file system. It contains the Linux file system hierarchy folders and files. The root file system contains tools for configuring, flashing and testing the device.

Flash-N-Go System is equipped with **no gui framework**.

2.4 The partition layout

As already stated in chapter [► 2 Overview], the different components of the embedded Linux system are stored in different partitons of the backing-storage. The backing-storage type of Flash-N-Go is eMMC. In addition to the partitions for the basic Linux components there may be some more partitions depending on the system configuration.

The partition layout for the Flash-N-Go platform is:

Partition	File System	Contents
mmcblk0boot0	none	bootloader image
mmcblk0boot1	FAT32	XML configuration parametes (config.xml) and touchscreen configuration (ts.conf)
mmcblk0p1	FAT32	Flash-N-Go Linux kernel image file (linuximage), bootloader command file (boot-alt.cfg) and Flash-N-Go ramdisk file (root.cpio.gz)

Flash-N-Go Boot can start the following Linux kernel image and Windows Embedded image types:

- **zImage** compressed image
- **uImage** compressed image with u-boot header
- **Image** uncompressed image
- **BNX** Garz & Fricke bnx image for Windows Embedded

2.5 Further information

For readers who are not familar with Linux in general, the following link may be helpful:

- http://tldp.org/LDP/intro-linux/html

Information regarding embedded Linux systems can be found in the following book:

- "Building Embedded Linux systems 2nd Edition", Karim Yaghmour, John Masters, Gilad Ben-Yossef, Philippe Gerum, O'Reilly, 2008, ISBN: 978-0-596-52968-0

Information regarding Linux infrastructure issues in general can be found at:

- http://tldp.org/LDP/Pocket-Linux-Guide/html
- http://www.linuxfromscratch.org

3 Flash-N-Go Boot

Flash-N-Go Boot is a simple, non-interactive bootloader. It will be started by the SoC-Boot-ROM from MMC/SD-Card storage media (usually the internal eMMC of the system), runs completely from SoC-internal SRAM, initializes the system clocks and DRAM, loads files from its boot-medium into any place in RAM and is able to execute certain kinds of binary images.

Even though it uses a serial port to output informative and debugging messages, it does not provide any kind of console interface, network drivers, display drivers or even the capability to write any data to storage media. All installation, configuration, update and other maintenance tasks usually provided by bootloaders like RedBoot, U-Boot, or Barebox are missing in **Flash-N-Go Boot** on purpose and provided by **Flash-N-Go System** instead, which is always expected to be installed on a device using **Flash-N-Go Boot**.

All actions necessary to boot any kind of OS are controlled by one of two simple ASCII-text boot-scripts, which **Flash-N-Go Boot** looks for in all FAT12/16/32-partitions on the storage medium it is booted from itself. Normally **Flash-N-Go Boot** looks for a boot-script called **boot.cfg**. In three special cases it looks for a boot-script called **boot-alt.cfg**, instead:

- if no **boot.cfg** can be found or loaded and executed successfully,
- if the **SW2**-button of the device is pressed during power-up or reset, or
- if some software – usually a tool called **bootselect** (refer to chapter 4.2.1) provided in any OS – has set a special flag in a particular place of non-volatile storage of the system.

Flash-N-Go Boot itself only uses the SoC-internal SRAM for its code and all runtime heap, stack, and data. All other RAM-areas of a device are free to use by boot-scripts and operating systems and as soon as **Flash-N-Go Boot** has started another binary executable, this in turn may also use the SoC-internal SRAM for its own purposes.

3.1 Boot-Script Commands

The boot-scripts support some few, simple commands to load and execute files or perform some simple tests. Please note, that **Flash-N-Go Boot** only provides a simple script parser with only basic error-checking and some commands may prevent other commands to be executed.

3.1.1 Comments

Each line in the boot-script starting with a #-character will be treated as comment and ignored.

3.1.2 Load Command

```
load [-b <address>] [-p <partition name>] [-r] [-o] <filename>
```

This command loads the file **<filename>** from a FAT-partition of the boot-medium. If no **<partition name>** is specified, **Flash-N-Go Boot** searches on all FAT-partitions for the specified file and loads the first one it finds. Otherwise it will only look on the specified partition.

The parameter **-b <address>** specifies the (physical) address in the SoC address-space where the file should be loaded to. Usually this should be a place in DRAM. Particular addresses will usually depend on the specific file and operating system, which is going to use it later on. In most cases the **-b**-parameter will be specified explicitely, for Linux kernel images using U-Boot-compatible fileformats or Android boot images **Flash-N-Go Boot** is able to determine the load address from the binary file itself. In these cases the **-b**-parameter may be omitted.

If the option **-r** is specified, **Flash-N-Go Boot** assumes that the file to be loaded will be used as a Linux RAM-Disk and automatically passes start-address and length in RAM to a Linux kernel using the so-called ATAG-mechanism.

If the option **-o** is specified, **Flash-N-Go Boot** considers the file to be optional, i.e. if it fails for some reason to load the file, e.g. if it is not found anywhere, this error will simply be ignored and execution of the boot-script will simply continue. If this parameter is not specified, **Flash-N-Go Boot** will stop execution of the boot-script if it fails to load the file for any reason. If **Flash-N-Go Boot** is currently executing **boot.cfg**, it will instead try to execute **boot-alt.cg**. If such an error happens during execution of **boot-alt.cfg**, the system will simply halt.

The **load**-command may be used in a boot-script as many times as desired.

3.1.3 Devtree Command

(Only available starting with **Flash-N-Go Boot** v5.0r3274 an newer)

```
devtree -b <pfid>:<socid>:<address> [-p <partition name>] <filename>
```

This command works similar to the **load**-command but is intended to load so-called "device trees" for newer Linux kernels, only, therefore no **-r**-parameter is supported and as newer Linux kernels require a proper device-tree, files loaded by this command cannot be specified as optional.

The syntax for the **-b**-parameter is somewhat more complicated, though, and specifies platform- and SoC-IDs for which a particular device-tree file is intended. Boot-scripts will usually contain multiple **devtree**-lines, which specify different files to be loaded all to the same **<address>** in RAM but using different **<pfid>** and **<socid>** values. **Flash-N-Go Boot** "knows" on which particular kind of platform and SoC it is currently running and will execute only **devtree**-commands with matching IDs, all others are ignored.

Currently the following **<socid>**s are defined:

- **0x61** Freescale i.MX6 Solo/DualLite
- **0x63** Freescale i.MX6 Dual/Quad
- **0x6301** Freescale i.MX6 DualPlus/QuadPlus

Currently the following **<pfid>**s are defined:

- **0x0** HYDRA
- **0x1** HYDRA
- **0x2** SANTVEND (battery)
- **0x3** SANTINO-LT
- **0x4** SANTINO
- **0x5** SANTOKA
- **0x7** SANTARO
- **0xE** SANVITO
- **0xF** SANTVEND

Please note that customer specific board-designs will usually also have defined **<pfid>**s specific, which are not documented in this manual. If you don't find your board listed here, please refer to board-specific manuals or contact us for further information.

Successful execution of a **devtree**-command will result in **Flash-N-Go Boot** to automatically generate and append a kernel command-line parameter **devicetree=<address>** when executing an **exec**-command.

On **Flash-N-Go Boot** versions prior to v5.0r3274, which don't support the **devtree**-command, you can use the **load**-command to load a device-tree image, instead. In this case the kernel parameter **devicetree=<address>** must be passed explicitely via the **exec**-command.

3.1.4 Exec Command

```
exec [-b <address>] [-r <ramdisc start> -s <ramdisc length] "<command-line>"
```

This command starts execution of some binary code at some address. **Flash-N-Go Boot** establishes a runtime environment as specified for ARM Linux Kernels before jumping to the specified address, i.e. MMU and Caches will be turned-off, the serial debug console will be the only UART enabled in the system, a Linux Machine-ID and ATAG pointers providing information on available physical DRAM, an optional RAM-Disc, a Linux kernel Command Line, etc. are passed to the executed code.

Please note, though, that **Flash-N-Go Boot** for compatibility reasons passes Linux Device Trees in a non-standard way: While newer Linux kernels expect a device-tree to be passed instead of an ATAGs-pointer, **Flash-N-Go Boot** will always pass an ATAGs-pointer to the executed code and will append a **devicetree=<address>**-parameter to the kernel command line found in the ATAGs.

If a **load**-command using the **-r**-parameter has been executed before, the **-r** and **-s** parameters may be ommitted. **Flash-N-Go Boot** will pass the proper load-address and length of the RAM-Disc image automatically. If no RAM-Disc image is supposed to be loaded, these parameters may be omitted, as well, of course.

If a **load**-command has been executed before to load an executable binary image file that **Flash-N-Go Boot** knows about, e.g. a Linux kernel using U-Boot kernel Image format or Android Boot-Image format, the **-b**-parameter is optional. In all other cases the **-b**-parameter is mandatory and will be the address in memory **Flash-N-Go Boot** will jump to.

If you don't want to pass additional command-line parameters to the code to be executed, an empty string "" must be specified with the **exec**-command.

Note that boot-script execution will stop immediately after an **exec**-command. Boot-scripts may therefore contain only one **exec**-command.

3.1.5 Testram Command

```
testram [-s] -b <address> -l <length> -r <number of runs>
```

This command performs a memory test in the specified area. The **-b**, **-l**, and **-r** parameters are all mandatory. After execution of the **testram**-command boot-script execution will stop immediately, no further commands will be executed from the boot-script.

The optional **-s**-parameter may be specified to execute the RAM-test in SMP-mode. Without this option on multicore SoCs only one single SoC will execute the RAM-test while all other cores are idle. With the **-s**-option specified, all cores will execute a RAM-test in parallel.

The partitioning of the specified address space between the multiple cores in SMP-mode will not be defined and may change without notice with different versions of **Flash-N-Go Boot**. It is guaranteed, though, that each core will run the RAM-test on some private part of the specified address space.

Behavior, output-format and results are compatible with the same command already provided in our RedBoot bootloaders.

3.1.6 Testbench Command

```
testbench [-s] -b <address> -l <length>
```

This command performs a simple benchmark test in the specified area. The **-b**, and **-l** parameters are all mandatory. After execution of the **testbench**-command boot-script execution will stop immediately, no further commands will be executed from the boot-script.

The optional **-s**-parameter may be specified to execute the benchmark in SMP-mode. Without this option on multicore SoCs only one single SoC will execute the benchmark while all other cores are idle. With the **-s**-option specified, all cores will execute a benchmark test in parallel.

The partitioning of the specified address space between the mutliple cores in SMP-mode will not be defined and may change without notice with different versions of **Flash-N-Go Boot**. It is guaranteed, though, that each core will run the benchmark on some private part of the specified address space.

Behavior, output-format and results are compatible with the same command already provided in our RedBoot bootloaders.

4 Flash-N-Go System

Flash-N-Go System is a RAM-disk-based Linux distribution which is installed in parallel to the real operating system. It is intended to be used for service tasks e.g. operating system updates or setting up device configurations. Moreover it serves several tools for testing the devices interfaces which will be used by the Garz & Fricke production tests.

4.1 Booting Flash-N-Go System

There are two options of booting the device into Flash-N-Go System instead of booting the target OS. The first one is pressing down and holding the bootmode switch while the power supply is switched on. The location of the bootmode switch is shown in [► Figure 1]. After a few seconds a Flash-N-Go System prompt should be displayed on the serial console similar to this:

```
FLASH-N-GO:/
```

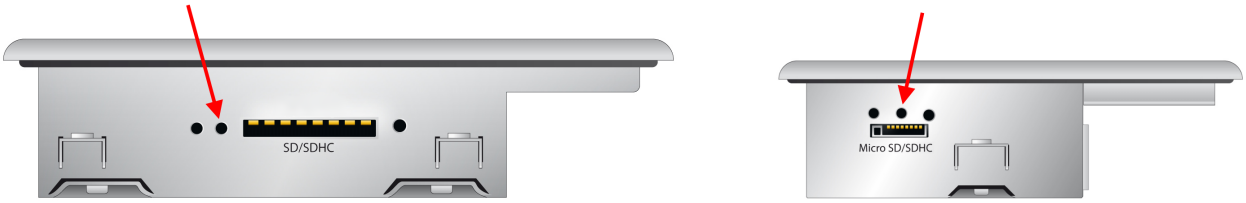

Figure 1: Location of the bootmode switch on the side of the device

Alternatively the bootselect tool can be used, which is usually provided by all Garz & Fricke operating systems using Flash-N-Go System as well as by Flash-N-Go System itself. See chapter [► 4.2.1 bootselect] for a detailled description of this tool.

4.2 Configuration tools

Garz & Fricke provides several tools on the Flash-N-Go System solution for configuring, testing and flashing the devices. This chapter describes some of these tools.

4.2.1 bootselect

This tool offers the possibility to switch the boot target between the flashed operation system (e.g. Linux, Windows Embedded) on the device and Flash-N-Go System itself. The following code block shows a list of all possible parameters.

```
FLASH-N-GO:/ bootselect -h
Usage: bootselect [-qv] [<bootmode>]

When called without parameters, bootselect reads the
current boot-mode setting from the RTC-register

Options:
    <bootmode>    Writes the given boot-mode into the RTC-register
                  May be one of: initialized regular alternative invalid
    -v            Turns on debug messages on stderr
    -vv           Turns on info messages on stderr
    -q            Turns off warning messages on stderr
    -qq           Turns off all messages on stderr
```

For selection of the target OS the bootselect tool should be called with the regular parameter:

```
FLASH-N-GO:/ bootselect regular
regular
```

For selection of Flash-N-Go System the bootselect tool should be called with the alternative parameter:

```
FLASH-N-GO:/ bootselect alternative
alternative
```

Note: Bootselect is also part of Garz & Fricke operating systems which use Flash-N-Go Boot as bootloader. Thus the selection may be done at OS runtime too.

4.2.2 sconfig

sconfig is a small command line tool which offers functionalities for device configurations like network settings, screen rotation etc. The following code block shows a list of all possible parameters.

```
FLASH-N-GO:/ sconfig -h
Usage: /usr/bin/sconfig {start | list | init | <setting> [value]}
   Call without [value] to read a setting, call with [value] to write it.
Available settings:
   serialdiag  switch serial debug console on or off
   dhcp        switch DHCP on or off
   ip          set IP address
   mask        set subnet mask
   gateway     set standard network gateway
   mac         set MAC address
   name        set device name
   serial      set serial number (affects MAC address and device name)
   rotation    set display rotation
If a 2.Ethernet is present, it may be configured via serial2, mac2, etc.
```

4.2.3 xconfig

xconfig is a command line tool to manipulate the XML-configuration data directly. Care should be taken when using these commands as miss-configured or corrupted XML- configuration data might render a system unusable. The following code block shows a list of all possible parameters.

```
FLASH-N-GO:/ xconfig
Usage: xconfig [command] [options]

Script to manipulate the XML-configuration data of GuF i.MX Boards
inside the NAND Flash.

xconfig expect one of the following commands to define its behavior:
  list           show the current contents of the XML-configuration as
                 obtained from NAND
  import         import an external XML-configuration file into the NAND
                 Flash
  addnode        add a new (empty) XML node
  delnode        delete XML node(s) and its/their children
  addattribute   add a new or change an existing attribute
  delattribute   delete an existing attribute
If no command and option given xconfig will display its version and exit.

Syntax:
  xconfig list [-v] [-p <path>]
        -v           show whole (sub)tree instead of summary, the format is
                     suitable to be used as input to "xconfig import"
        -p <path>    path to subtree, e.g.:
                     -p /configurationFile/variables/setting[@key='enable_serialdiag
                         ↪ ']
                     if omitted, the whole tree is shown
  xconfig import [-v] [-y] [-b] <file>
        -v           be verbose
        -y           assume yes - apply changes without user interaction
```

```
        <file>       path to file to import
        -b           deprecated - do not use
  xconfig addnode [-y] -p <path> -n <name>
        -y           same as above
        -p <path>    path to parent node
        -n <name>    name of new node
  xconfig delnode [-y] -p <path>
        -y           same as above
        -p <path>    path to node
  xconfig addattribute [-y] -p <path> -n <attr-name> -v <value>
        -y           same as above
        -p <path>    path to parent node
        -n <attr-name>   name of the attribute
        -v <value>   new value of attribute
  xconfig delattribute [-y] -p <path> -n <attr-name>
        -y           same as above
        -p <path>    path to parent node
        -n <attr-name>   name of the attribute
  xconfig listattributes -p <path>
        -p <path>    path to node
  xconfig getattribute -p <path> -n <attr-name>
        -p <path>    path to parent node
        -n <attr-name>   name of the attribute
```

xconfig list This command displays the whole XML-tree with its configuration data stored on the device.

```
xconfig list [-v] [-p <path>]
```

Example: Display the XML-configuration data

```
xconfig list -v
```

xconfig import This command imports an external XML configuration file into **Flash-N-Go System**'s current XML configuration. The source file must have been loaded to the destination file specified with the **-b**-option before. If the **-v**-option is specified, the command will output additional information about the changes made to the existing XML-configuration.

```
xconfig import [-v] [-y] -b <file>
```

Example: Import boot-logo license

```
tftp -g 192.168.1.100 -r rb-logolicense.xml -l ~/rb-logolicense.xml
xconfig import -b ~/rb-logolicense.xml
```

xconfig addnode This command adds a new, empty node with the name specified via the **-n**-option as a child to each of the nodes selected with <path>. As any node requires certain attributes to be functional, usage of this command is not recommended, because all mandatory attributes must be added to the node(s) manually, as well.

```
xconfig addnode [-y] -p <path> -n <name>
```

xconfig delnode This command deletes all nodes selected with the specified <path> and all child nodes.

```
xconfig delnode [-f] [-y] -p <path>
```

Example: Delete display configuration

```
xconfig delnode -p /variables/display
```

xconfig addattribute This command adds a new or changes an existing attribute with the name <attr-name> of all nodes selected with the specified <path> to the value specified with the **-v**-option.

```
xconfig addattribute [-y] -p <path> -n <attr-name> -v <value>
```

Example: Rotate display by 180 degree

```
xconfig addattribute -p /variables/display/rotation -n value -v 180
```

xconfig delattribute This command deletes existing attributes with the name <attr-name> from all nodes selected with the specified <path>.

```
xconfig delattribute [-y] -p <path> -n <attr-names>
```

Example: Remove the backlight look-up table, causing the system to revert to an internal hard-coded table

```
xconfig delattribute -p /variables/display/backlight -n lut
```

Note: For compatibility reasons with the xconfig implementation in **RedBoot**, the path can also be prepended with:

```
/configurationFile
```

4.3 Manual System Update

The Manual System Update considers a manual Operating System installation via a console. The update process described here requires an Ethernet connection of the device to a development PC, either directly or via intranet. Also a TFTP-Server is required on the development PC. There are several, also free, TFTP-Servers online available. Please make sure that the timeout settings of the TFTP-Server are set to 60 seconds at minimum.

The installation of target OSes, and Flash-N-Go System and Flash-N-Go Boot should work via console one- or two-liners. All device partitioning, deployment actions should be done by shell scripts executed by the simple command.

Accessing the target system via serial console Before updating the target operating system we first need an access to the device. The easiest way to access the target is via the serial console. Simply connect the first RS-232 port of your target system with a COM port of your PC or a USB-to-RS-232 converter using a null modem cable. With the serial connection set up start your favorite terminal program (e.g. minicom or tera term) with the following settings:

- 115200 baud
- 8 data bits
- no parity
- 1 stop bit
- no hardware flow control
- no software flow control

From the very first moment when the target is powered, you should see debug messages in the terminal.

Boot into Flash-N-Go System You need to force the target to boot into **Flash-N-Go System** either by pressing the **SW2** button while supply power or by using the **bootselect** tool as it is descibed in chapter [► 4.1 Booting Flash-N-Go System]. After the boot process has finished, you will see the **Flash-N-Go System** prompt.

```
FLASH-N-GO:/
```

Note: The SW2 button needs to be pressed on each boot process. The selection made with bootselect is persistent.

Establish a network connection The Ethernet can be set up with the sconfig command line tool, see also chapter [▶ 4.2.2 sconfig].

Example 1: Set IP Address 192.168.1.1 and netmask 255.255.255.0 and reboot the system to apply the network configuration:

```
FLASH-N-GO:/ sconfig ip 192.168.1.1
FLASH-N-GO:/ sconfig mask 255.255.255.0
FLASH-N-GO:/ sconfig dhcp off
FLASH-N-GO:/ reboot
```

Example 2: Set DHCP and reboot the system to apply the network configuration:

```
FLASH-N-GO:/ sconfig dhcp on
FLASH-N-GO:/ reboot
```

Note: It is also necessary to install a TFTP-Server on a host PC.

Download the image files of the desired OS Checkout the Garz & Fricke website for the latest release of your desired operating system e.g. Windows Embedded Compact or Yocto Linux and download all necessary files. Copy them to the root directory of your TFTP server (usually TFTP-Root).

4.3.1 Example 1: GUF-Yocto V15.0-r4523-9 for SANTARO

Download all files within the Linux-Yocto/Releases/Yocto-15.0-r4523-9/prebuilt_images folder of the Garz & Fricke FTP server.

- GUF-Yocto-15.0-r4523-9-SANTARO-boot.cfg
- GUF-Yocto-15.0-r4523-9-SANTARO-fng-install.sh
- GUF-Yocto-15.0-r4523-9-SANTARO-imx6-santaro-x1.dtb
- GUF-Yocto-15.0-r4523-9-SANTARO-imx6-santaro-x2.dtb
- GUF-Yocto-15.0-r4523-9-SANTARO-linuximage.bin
- GUF-Yocto-15.0-r4523-9-SANTARO-modules.tgz
- GUF-Yocto-15.0-r4523-9-SANTARO-pkg.py
- GUF-Yocto-15.0-r4523-9-SANTARO-root.tar.gz

Start the Linux installation process

```
export TFTP=<TFTP-Server IP>; curl tftp://$TFTP/GUF-Yocto-15.0-r4523-9-SANTARO-fng-
    ↪ install.sh | sh
```

Note: Replace <TFTP-Server IP> with the IP address of your TFTP server

4.3.2 Example 2: Windows Embedded Compact 7 V3.0r477-0 for SANTARO

Download the files listed bellow from the WindowsCE7/OS V3.0r477-0/ folder of the Garz & Fricke FTP server.

- SANTARO_CE7_3.0r477-0_28.01.2015.bnx
- SANTARO_CE7_3.0r477-0_boot.cfg
- SANTARO_CE7_3.0r477-0_fng-install.sh

Start the WEC7 installation process

```
export TFTP=<TFTP-Server IP>; curl tftp://$TFTP/SANTARO_CE7_3.0r477-0_fng-install.sh
    ↪ | sh
```

Note: Replace <TFTP-Server IP> with the IP address of your TFTP server

The installation procedure will take some minutes. The output of the installation procedure will appear on the terminal console. The installation procedure finishes by outputting the Flash-N-Go prompt again:

```
FLASH-N-GO:/
```

4.3.3 Example 3: Upgrade to Flash-N-Go System 8.0

Be aware that updating the Flash-N-Go System with the **self-init.sh** script deletes the previously installed target OS and user data. Download the Flash-N-Go System files listed bellow from the **Flash-N-Go/FNGSystem/FNGSystem-8.0** folder of the Garz & Fricke FTP server.

- Flash-N-Go-System-8.0-python-3.1.4.ext2
- Flash-N-Go-System-8.0-self-init.sh
- Flash-N-Go-System-8.0-self-update.sh
- Flash-N-Go-System-8.0.cpio.gz
- GF_Bootlogo_Flash-N-Go-System.png
- boot-alt.cfg
- fng_wipe_emmc.sh

Additionally download the Kernel Files files listed bellow from the **Flash-N-Go/FNGSystem/FNGSystem-8.0/GUF-Yocto-35.0-r5982-0** folder of the Garz & Fricke FTP server. Place them in a subfolder of the TFTP-Server root folder named **GUF-Yocto-35.0-r5982-0**

- GUF-Yocto-35.0-r5982-0-IMX6GUF-imx6dl-santaro.dtb
- GUF-Yocto-35.0-r5982-0-IMX6GUF-imx6dl-santino-lt.dtb
- GUF-Yocto-35.0-r5982-0-IMX6GUF-imx6dl-santino.dtb
- GUF-Yocto-35.0-r5982-0-IMX6GUF-imx6dl-santoka.dtb
- GUF-Yocto-35.0-r5982-0-IMX6GUF-imx6q-santaro.dtb
- GUF-Yocto-35.0-r5982-0-IMX6GUF-imx6q-santoka.dtb
- GUF-Yocto-35.0-r5982-0-IMX6GUF-linuximage.bin
- GUF-Yocto-35.0-r5982-0-IMX6GUF-modules.ext4
- ParamFile.txt
- md5sums.txt

Start the Flash-N-Go System upgrade process

```
export TFTP=<TFTP-Server IP>; curl tftp://$TFTP/Flash-N-Go-System-8.0-self-init.sh |
    ↪ sh -s -- --ParamFile=GUF-Yocto-35.0-r5982-0/ParamFile.txt
```

Note: Replace <TFTP-Server IP> with the IP address of your TFTP server

The installation procedure will take some minutes. The output of the installation procedure will appear on the terminal console. The installation procedure finishes by outputting the Flash-N-Go prompt again:

```
FLASH-N-GO:/
```

4.3.4 Example 4: Upgrade Flash-N-Go Boot to version 9.0

Download the Flash-N-Go Boot files listed bellow from the **Flash-N-Go/FNGBoot/FNGBoot-v9.0r3476/** folder of the Garz & Fricke FTP server.

- GF_iMX6x-FNGBoot-v9.0r3476_22-07-2016.bin
- fng_boot_update.sh
- md5sums.txt

Start the Flash-N-Go Boot upgrade process

```
export TFTP=<TFTP-Server IP>; curl tftp://$TFTP/fng_boot_update.sh | sh
```

Note: Replace <TFTP-Server IP> with the IP address of your TFTP server

The installation procedure will take some seconds. The output of the installation procedure will appear on the terminal console. The installation procedure finishes by outputting the Flash-N-Go prompt again:

```
FLASH-N-GO:/
```

Boot into the new OS After the successful installation of the new OS you can now boot into it. If the **bootselect** was used to boot into **Flash-N-Go System**, the system now has to be configured to boot into the normal OS.

```
bootselect regular
```

If the **SW2** button was used, the system can be rebooted immediately.

```
reboot
```

4.3.5 Example 5: Android 7.1.1 guf-4.0-0 for SANTARO via Internet

Most Garz & Fricke Operating Systems are directly installable via the Garz & Fricke support server. To do this set the **TFTP** environment variable to the URL where you can find the install script. In the following example this path is **http://support.garz-fricke.com/products/Santaro/Android-7.1/IMX6GUF_Android_7.1.1_guf-4.0-0/**.

Then use the **TFTP** variable in the curl command directly, and append the install script name

Supposing a System booted into Flash-N-Go System either via boot mode switch or via the **bootselect** tool, execute the following:

```
udhcpc
export TFTP=http://support.garz-fricke.com/products/Santaro/Android-7.1/
    ↪ IMX6GUF_Android_7.1.1_guf-4.0-0/
curl ${TFTP}/IMX6GUF_Android_7.1.1_guf-4.0-0_fng-install.sh | sh
```

4.3.6 Example 5: Android 7.1.1 guf-4.0-0 for SANTARO via USB stick

You can also use the install script from a mass storage device like a USB stick. For example copy all files from the support website ► http://support.garz-fricke.com/products/Santaro/Android-7.1/IMX6GUF_Android_7.1.1_guf-4.0-0/ to your USB stick.

Supposing a device booted into Flash-N-Go System either via boot mode switch or via the **bootselect** tool you can check the content of your USB-Stick:

```
cd mnt/mstick1/
ls -la
drwxr-xr-x    3 0        0             4096 Jan  1  1970 .
drwxr-xr-x    4 0        0                0 Aug 29 14:13 ..
-rwxr-xr-x    1 0        0         12688200 Aug 29  2018 IMX6GUF_Android_7.1.1_guf
    ↪ -4.0-0_boot.tar.gz
-rwxr-xr-x    1 0        0            43434 Aug 29  2018 IMX6GUF_Android_7.1.1_guf
    ↪ -4.0-0_fng-install.sh
```

```
-rwxr-xr-x    1 0        0              817 Aug 29  2018 IMX6GUF_Android_7.1.1_guf
    ↪ -4.0-0_pkg.py
-rwxr-xr-x    1 0        0        302845644 Aug 29  2018 IMX6GUF_Android_7.1.1_guf
    ↪ -4.0-0_system_raw.img.gz
-rwxr-xr-x    1 0        0              308 Aug 29  2018 IMX6GUF_Android_7.1.1_guf
    ↪ -4.0-0_userdata.tar.gz
-rwxr-xr-x    1 0        0              741 Aug 29  2018 md5sums.txt
```

You can then execute the the following lines (in the mstick1 directory):

```
unset TFTP
sh IMX6GUF_Android_7.1.1_guf-4.0-0_fng-install.sh
```

4.4 Security Considerations

The **Flash-N-Go System** is designed as management system to install and update target operating systems, and change the device configurations. Thus it runs with root privileges. This may be a concern for some scenarios. Therefore some security considerations are necessary.

4.4.1 Hardware Access

For embedded devices direct access to the hardware may generally be a security concern. For Garz & Fricke devices for example, one may want to restrict the access to the reset- and bootmode buttons. One should also consider to restrict the access to interfaces like serial line, USB or SD-Card slots as those may be used for scripts to alter your system. An alternative to this is to deactivate features by software (See 4.4.2). If not used for a product in the field at all, it may be possible to omit the interface in the manufacturing process altogether. Please feel free to contact your customer advisor to discuss possible options.

4.4.2 Services

By default **Flash-N-Go System** provides some services.

- ftpd - An FTP Server
- telnetd - A Telnet Server
- usbconsole - A shell provided on the micro-USB OTG port

The services can be disabled with setting in the system configuration XML file. Currently there is no sconfig abstraction for the settings and therefore the raw XML files need to be imported to disable the services. See **xconfig import** in the section 4.2.3 for further details on how to import XML settings.

The necessary files can be found under:

- http://support.garz-fricke.com/projects/Santaro/Configuration/System

The files are called **rbcfg-fng-disableFtpd-r3711.xml**, **rbcfg-fng-disableTelnetd-r3711.xml**, and **rbcfg-fng-disableTelnetd-r3711.xml**

4.4.3 Serial Console

It may be necessary to disable the serial console as well. The process to disable the serial console is the same as for the Services, by importing the corresponding XML setting. The file is called **rbcfg-fng-disableSerialconsole-r3711.xml**.

Be aware that by disabling all serial console connections you may lock yourself out of the **Flash-N-Go System**.

4.4.4 Password

You can configure the **Flash-N-Go System** to use a root password. But because the system is a RAM-based system you cannot simply create a persistent /etc/passwd file. Instead, the system checks for a file called **passwd_fngsystem** in the shared configuration partition (accessible through **/etc/shared**), copies the file to **/etc/passwd** and uses the file from there.

To create the file and set the password you may use the following procedure:

```
FLASH-N-GO:/# echo root:x:0:0:root:/:/bin/sh > /etc/passwd
FLASH-N-GO:/# chmod 755 /etc/passwd
FLASH-N-GO:/# passwd
Changing password for root
New password:
Retype password:
Password for root changed by root
FLASH-N-GO:/# cp /etc/passwd /etc/shared/passwd_fngsystem
```

After a reboot or the next time you login a password prompt appears.

4.4.5 Autojob

The Garz & Fricke Systems use a system of automatic scripts from internal and external storage devices, like USB-Sticks, and SD-Cards. The tools execute automatic copying of files, automatic start of applications and automatic start of the update tool. To prevent any of this feature from executing the autojob tools can be restricted.

One type of restriction is simply disabling the functions. This can be done similar to the Services by using the corresponding XML files.

Another way to restrict the autojob functions is to restrict the devices allowed to execute the autojob tools. I.e. one may want to restrict the autojob tools to USB-Sticks only. To do this use the device nodes begining with "sd" as a regular expression as a filter.

This can be achieved by the following XML file:

```
<?xml version="1.0" encoding="ASCII" standalone="yes" ?>
<configurationFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <variables>
    <setting key="autosource_fngsystem"
             title="FnG: Autojob source device"
             type="string"
             enable_key=""
             enable_sense="false"
             value="sd.*"
             />
  </variables>
</configurationFile>
```

The following example writes the XML data into a temporary file and imports it to the system configuration via xconfig:

```
FLASH-N-GO:/# cat > /tmp/xml.xml <<EOF
><?xml version="1.0" encoding="ASCII" standalone="yes" ?>
><configurationFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
>                   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
>  <variables>
>    <setting key="autosource_fngsystem"
>             title="FnG: Autojob source device"
>             type="string"
>             enable_key=""
>             enable_sense="false"
>             value="sd.*"
>             />
>  </variables>
></configurationFile>
>EOF
FLASH-N-GO:/# xconfig import /tmp/xml.xml
Update non-volatile configuration - continue (y/n)? y
Configuration successfully written.
FLASH-N-GO:/#
```

Similar to the source device the valid boot-mode can be restricted. To restrict the tools to the boot mode "alternative" only use the following file:

```
<?xml version="1.0" encoding="ASCII" standalone="yes" ?>
<configurationFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <variables>
   <setting key="autobootmode_fngsystem"
    title="FnG: Autojob bootmode filter"
    type="string"
    enable_key=""
    enable_sense="false"
    value="alternative"
    />
   </variables>
</configurationFile>
```

5 Flash-N-Go Update

Flash-N-Go Update is used to change or update the operating system (OS) of your Garz & Fricke device. It can be used with all **Garz & Fricke SAN* models**. The updater normally comes as USB flash drive and uses an autostart feature of the device to bring up the graphical update process. The available operating systems can be directly downloaded, if the device is setup up with internet access.

5.1 Quickstart

There are two ways to start Flash-N-Go Update.

1. When a **Garz & Fricke** Linux distribution is installed, simply plug in the update USB flash drive while the OS is running.
2. Alternatively, start the device in service-mode (**Flash-N-Go System**) and plug in the update USB flash drive. To start **Flash-N-Go System** directly, keep SW2 pressed while connecting the power supply. (See also 4.1)

When the device has no internet connection during the first start, a dialog may come up to enter the device's article number, which can be found on the back of the device (900-XXXXR).

5.1.1 Network Setup

Flash-N-Go Update is capable to do online updates from the Garz & Fricke Update Server. This feature is enabled by default and can be disabled in the IP Config panel with the **enable offline mode** button. For using online updates, **Flash-N-Go Update** needs access to the internet to reach the Garz & Fricke Update Server at the address: https://firmware.garz-fricke.com, port 4431. To change or view the IP settings select the **Options** button in the updater GUI.

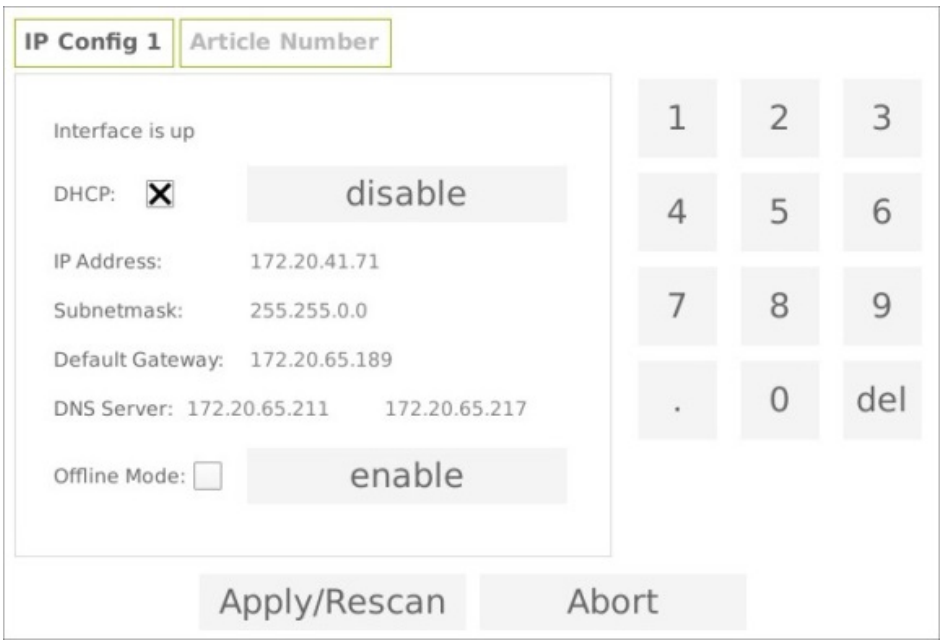

Figure 2: Dialog to change the IP settings of the device

After startup the Updater searches for firmware packages, online and locally stored, and displays them in the 'firmware packages' area. If no network is available during startup, a popup dialog is displayed. Select **OK** to continue using local packages only, **Retry** to do another connection attempt or **IP Settings** to configure the network interfaces.

5.1.2 Flash-N-Go Update GUI elements

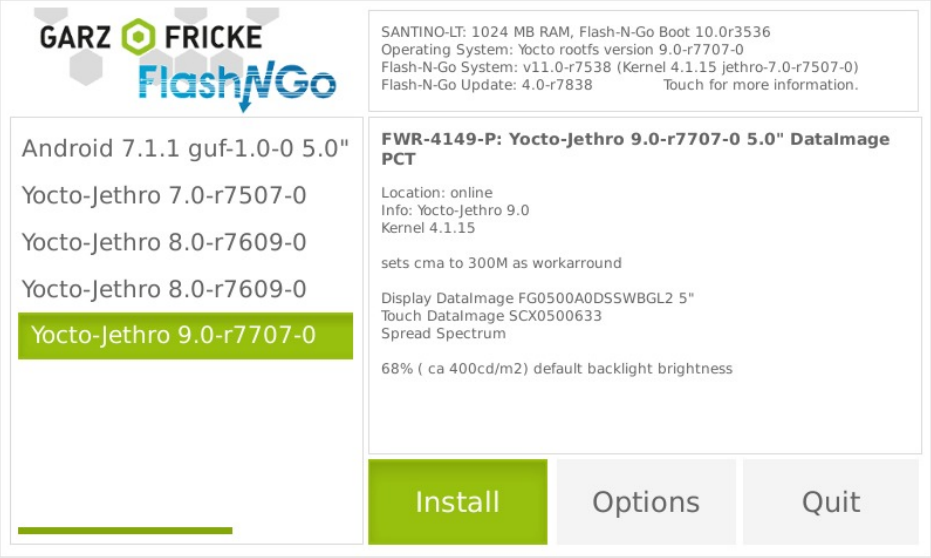

Figure 3: Graphical user interface of the Flash-N-Go Update tool

The GUI is divided into four parts:

1. System information: The area on top contains information about the system and the versions of the installed software. Touch this field to get a more detailed information page and access to the latest installation logs.
2. Firmware packages: On the left, there is list of available firmware packages. These may be different types of Operating Systems as well as different versions. If the device is connected to the internet, Flash-N-Go Update will display all firmware packages which are available online. Otherwise Flash-N-Go Update will display all suitable packages which are stored locally on the USB flash drive. Online packages are automatically cached during the update process and available offline afterwards.
3. Description: On the right the description area shows information about the selected firmware package. i It contains the firmware package number (FWR-1553), the name, if the package information was loaded from the internet (Location: online), and some additional information.
4. Buttons:
   - The **Install** button starts the installation of the selected firmware package.
   - The **Options** button shows the IP-Configuration page mentioned above.
   - The **Quit** button is used to quit the application. A dialog is displayed to choose if you want to reboot into the main OS of the device **Yes** or into Flash-N-Go System **Flash-N-Go**, **Abort** returns to the UI.
   - The small arrow in the left edge can be used to move dialogs to the left edge of the screen.

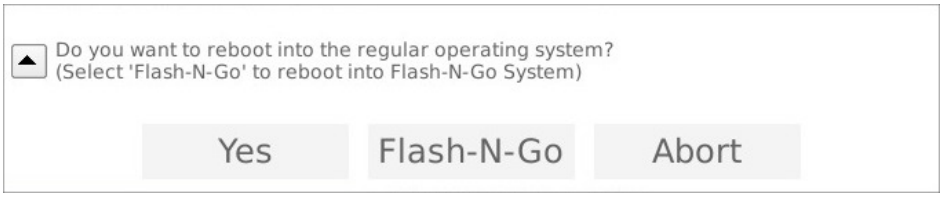

Figure 4: Quit dialog

This is useful on some dialogs to see the content of the field below, for example the logs when an error dialog is displayed.

5.1.3 Installation

To install a package, select it from the firmware packages list and press the **Install** button. A warning dialog is shown to remind you that all custom data will be overwritten.

Figure 5: Warning dialog before the installation

The **Force** button can be used to make sure parts of the OS like Flash-N-Go System and Flash-N-Go Boot are written, even if the same version is already installed. During the installation a reboot may be executed, but not user interaction is required here. The **Yes** button starts the default installation. Flash-N-Go Update will download and verify all needed files and cache them on the USB flash drive for further installations.

Figure 6: Process dialog before the installation

Note: The Flash-N-Go Update process deletes all custom data on the device!

During download and installation log messages and a progress bar inform about the executed steps.

5.1.4 Reboot the system

When the installation is finished, you can choose to reboot into the newly installed operating system by selecting **Yes**.

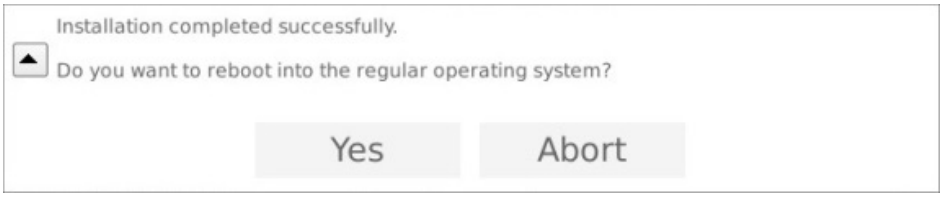

Figure 7: Dialog after a successful installation

If there was an error during the installation a dialog with the error message is displayed. **Retry** reboots the device and starts the installation again. The **Abort** returns to the main screen in all cases.

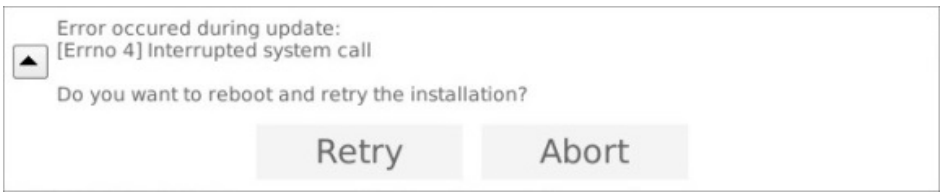

Figure 8: Error dialog during installation

Before the final reboot is executed, you have to unplug the Flash-N-Go Update USB flash drive from the device.

6 Related documents and online support

This document contains product specific information. Additional documentation is available for the use of embedded operating systems, the related tool chain and the bootloader (BIOS).

Title	Description
RedBoot User Manual	Contains relevant information about BIOS, boot logo, display settings, etc. in the case that RedBoot is used as BIOS.
U-Boot User Manual	Contains relevant information about BIOS, boot logo, etc. in the case that U-Boot is used as BIOS.
Windows OS Manual	Contains information about Windows Embedded CE, the tool chain, the development environment Visual Studio, Garz & Fricke tools, etc..
Linux OS Manual	Contains information about Linux BSP, the tool chain, Qt, etc..
SAM-BA User Manual	Contains relevant information about the usage on ATMEL's SAM-BA tool with Garz & Fricke devices in the case that an AT91SAM based platform is used.

Support for your Garz & Fricke embedded device is available on the Garz & Fricke website. You may find a list of the documents available, as well as their latest revision and updates for your system:

- http://www.garz-fricke.com/embedded-solutions_en.html

A GNU General Public License v2

Version 2, June 1991

A.1 Preamble

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

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

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

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

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

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

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

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

A.2 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

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

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

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

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

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

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

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

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

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

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

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

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

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

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

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

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

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

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

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

A.3 END OF TERMS AND CONDITIONS

A.3.1 How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and an idea of what it does.
Copyright (C) yyyy  name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

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

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it
starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'.  This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.
```

The hypothetical commands ‘show w’ and ‘show c’ should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ‘show w’ and ‘show c’; they could even be mouse-clicks or menu items–whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

B Standard Device Configuration

This section describes the default configuration of a Garz & Fricke device. It is considered as a template for further customer configuration, that may also done during the production test. Please request detailed information how to customize the configuration for series production from your contact person at Garz & Fricke GmbH.

B.1 Operating System

OS Type	PTXdist based Linux
OS Distribution Version	2011.09.0 (OSELAS.Toolchain-1.99.3)
OS Release Version	15.1-r8673

B.2 Bootloader

Bootloader Type	Flash-N-Go Boot
Bootloader Revision	9.0r3476

B.3 Boot Logo

Boot Logo Type	Flash-N-Go System logo
Boot Logo License	No license needed

Note: A custom boot logo always requires a boot logo license!

B.4 Serial diagnostic port

Activated by Default	☑
Serial Port	RS-232 #1
Baud Rate	115200
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	None

B.5 IPv4 Settings

DHCP	Deactivated
IPv4 Address	192.168.1.1
IPv4 Subnet mask	255.255.255.0
IPv4 Default Gateway	192.168.1.100

B.6 Services

Service	Port	Enabled by Default
Telnet Server	23	☑
SSH Server	22	☐
FTP Server	20	☑
HTTP Server	80	☐
SNMP Server	161	☐
OpenVPN	1194 (UDP)	☐
CUPS	631	☐

B.7 Display

The display orientation of the device is by default landscape. The most connectors of the device are at the bottom side. We consider this to be 0° rotation.

Note: When hardware is rotated clockwise, the display content must be rotated counterclockwise.