APU Developers Manual

Smart Sensors

PSxxx-90+
PACxxx-90-y-zz+
PS Lightweight

Triple-IN GmbH | Poppenbütteler Bogen 64, 22399 Hamburg

Document changes

Version	Date	Authors	Last changes
1.00.01	2016-01-16	SC	Initial versions. IDE and board setup
1.00.02	2016-12-28	SC	Updated for the latest kernel version and Eclipse NEON
1.00.03	2017-05-10	SC	Description of various configurations
1.00.04	2018-05-04	SC	Documents merged into new format
1.00.05	2018-05-07	SC	Updated to APU Firmware Version 04.00.18
1.00.06	2018-06-13	SC/TT	Fixed some typos

Content

1 Introduction

1.1 About this document

This document applies to all Triple-IN's Smart Sensors.
This document describes the development system of the Smart Sensors family.
This document is related to the models:

```
PS Firmware Version     3.04.05
APU Firmware Version    4.00.xx
```

This APU Developers Manual is part of a set of documents:

Manual	Targeted persons	Content
User's Manual	Technical personnel	Transport, mounting and installation Wiring and maintenance Operating means, system configuration Technical data
Programmer's Manual	Software developers	Data formats Commands and responses
APU Developer's Manual	Software developers	Developer environment setup Specific APU features

If you or your colleagues have any comments on this manual, we would be grateful to hear from you. Please write to:


```
Triple-IN GmbH
Poppenbütteler Bogen 64
D-22399 Hamburg - Germany
Telefon +49(0)40 50091998
Mail info@triple-in.de
```


1.2 Web server with Documents and Firmware Updates

The latest version of this and related documents, and the latest firmware updates can be downloaded from Triple-IN's web server.

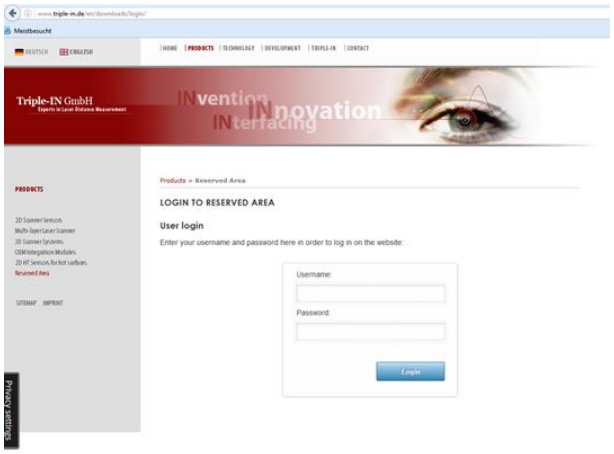

Figure 1: Triple-IN's web server Reserved Area login

Please contact Triple-IN to get access to the reserved area of the web server:

```
info@triple-in.de
```

1.3 PSDemoProgram source code project

Triple-IN's web server offers a C/C++ source code project for free download. PSDemoProgram is a very basic software to communicate and control Triple-IN Laser Scanners over the Ethernet.
PSDemoProgram comes as simple command line application. However, it incorporates the most important PS Laser Scanners commands and an example to design the scan procedure. The program is placed into the public domain and may be used for any purpose.

❖ Important

No warranty or support is provided by Triple-IN or its distributors on the PSDemoProgram package.

1.4 Prerequisites

Some basic knowledge is required and assumed to take full advantage of this manual about the following topics:

- Basic Linux, such as terminal commands, directories, permissions

- Basic C/C++
- Microsoft Windows

This manual is intended for setting up a development system on a Microsoft Windows host machine.

2 Safety Instructions

The following are general safety instructions. Please refer to your sensor's User Manual for dedicated instructions.

Caution

Before using the Smart Sensors, the user manual must be read, and all the instructions must be carefully observed.
The Smart Sensors must be installed, configured and serviced only by qualified personnel.
National and international rules and regulations must be applied according to the field of application and usage.
Smart Sensors cannot be used as safety devices.

Caution

The measurement laser is a laser class 1M or 1 product, depending on the sensor model (see sensor's User Manual). Do not look directly into the laser beam!
Red laser marker is a laser class 2 product. Emits visible light (635 to 678 nm). Do not look directly into the laser beam!

Caution

To reduce the risk of electric shock, do not remove the cover. Device contains high voltage components!
Connect and disconnect electrical linkages only under de-energized conditions.

Warning

Do not open the Smart Sensors.
If opened, the mechanical adjustment will be damaged, and warranty will get void!

3 Tools installation

The Smart Sensors' development board is a Linux machine with a TI Sitara AM335X microprocessor. The APU board is currently running a customized kernel version tagged 4.4.91-triplein with a customized rootfs based on *busybox*.
With that in mind, the user can freely choose any language and any compiler which is able to generate code for such machine.
In this document we focus on enabling the user to write, build and install an application written in C/C++ language.

3.1 Toolchain

The gcc toolchain we recommend is one from the Linaro (www.linaro.org) database. It can be downloaded from this link:

https://releases.linaro.org/archive/14.09/components/toolchain/binaries/gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_win32.zip.xz

It must be decompressed to a folder of user choice. This folder will be used for the project setup.
For the purpose of example, in this manual we'll use the folder:

```
C:\Programs\linaro\gcc-linaro-arm-linux-gnueabihf\
```

3.2 Eclipse

Using an IDE tool such Eclipse (open source tool maintained by the Eclipse Foundation) can drastically simplify the development phase.
The suggested Eclipse version is the latest available one. At the time of writing this manual, the latest Eclipse version is the Oxygen. The version with CDT is in order and can be freely downloaded from

http://www.eclipse.org/downloads/packages/eclipse-ide-cc-developers/oxygen3a

Download the zip format of 64 bits or 32 bits version (depending on your host operating system) and decompress it in a folder of your choice.
Launch eclipse and install Java if it's not yet installed in the system.

❖ **Important**

Even if newer Eclipse versions should work without issues, Triple-IN cannot provide any guarantee nor any support.

3.2.1 Working with different Eclipse versions

One of the advantages of Eclipse is that all the needed files are stored in one single directory. This makes possible to have different folders for different Eclipse's versions to work with.

4 Connecting Eclipse to the APU

We need to create a new remote connection on Eclipse to easily transfer the application files and debug it.

4.1 Create a new Remote Connection

Locate the Remote Systems panel. If not available, make it visible following the `Window -> Show view -> Others…` menu and searching for the Remote Systems folder. Click on Remote Systems and press `Open`.

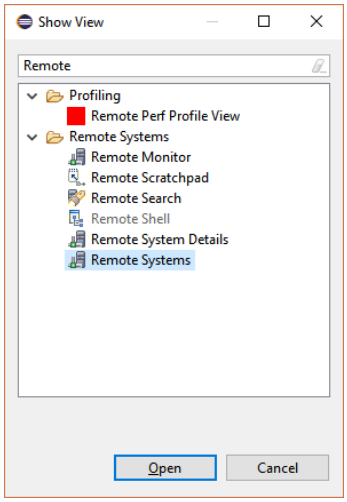

Figure 2: Eclipse Show View window

To create a new Remote connection, find the `Define a connection to remote system` tool button and press it.

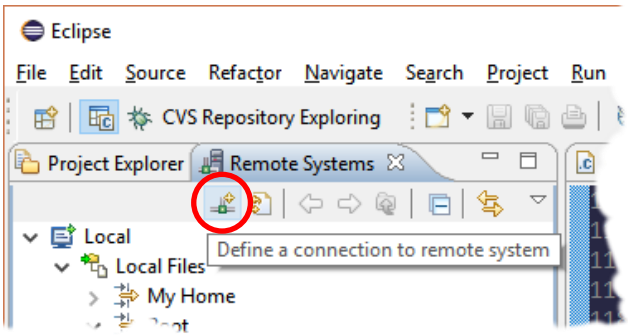

Figure 3: Eclipse Remote Systems view

The `New Connection` dialog will open.
On the first page, select `Linux` as Remote System Type and press `Next`.

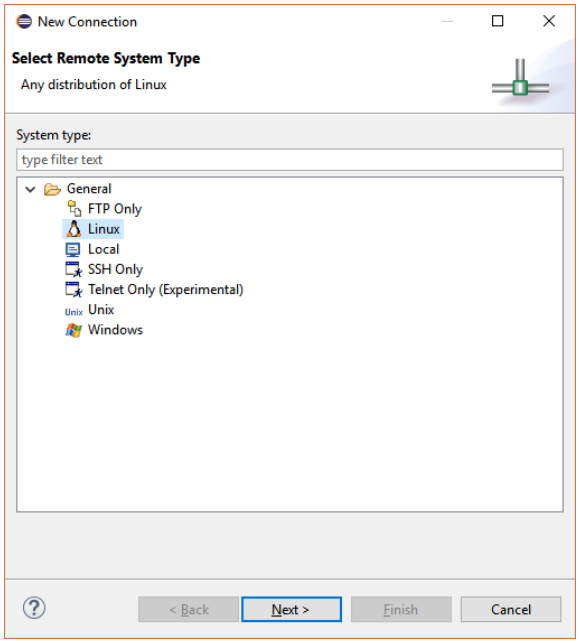

Figure 4: Select Remote System Type

On the next page, enter the APU IP address and a descriptive name for your connection.

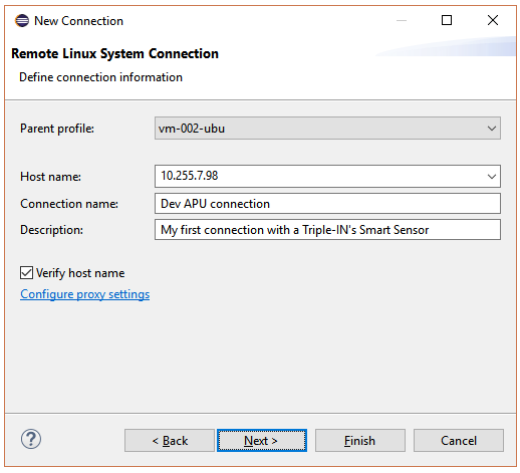

Figure 5: Connection information

Press the `Next` button to modify further parameters.

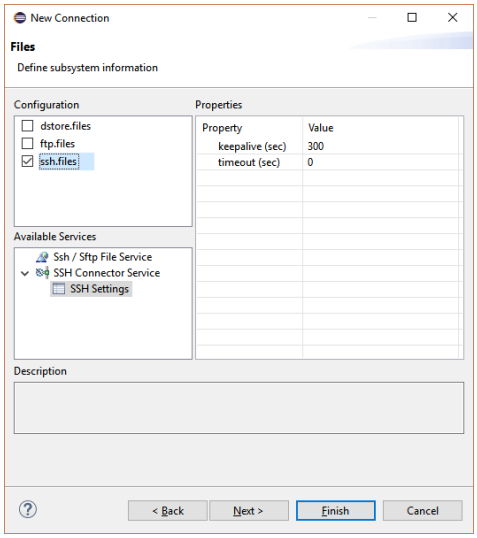

Figure 6: Define subsystem information: files

Choose `ssh.files` in the `Configuration` box and click `Next`.

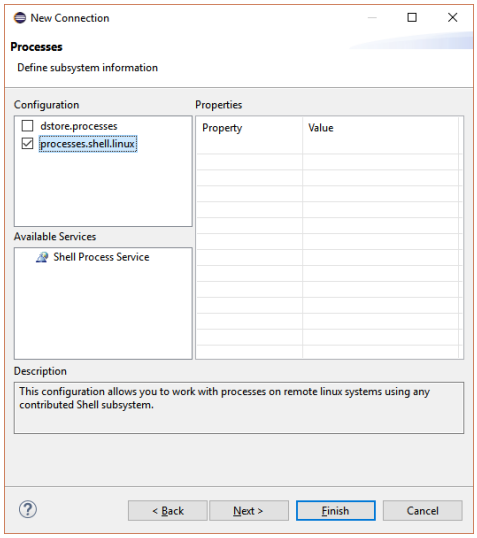

Figure 7: Define subsystem information: processes

Choose `processes.shell.linux` in the `Configuration` box and click `Next`.

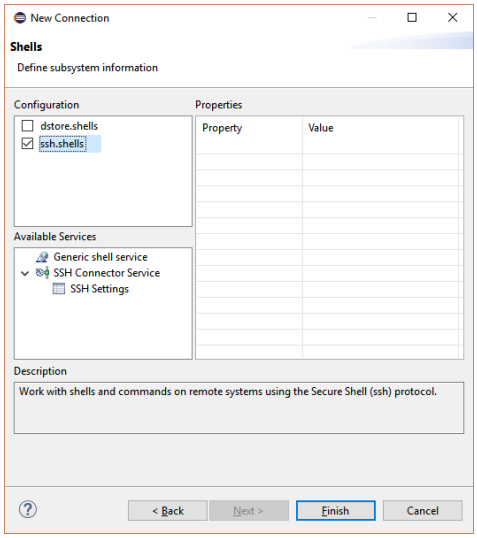

Figure 8: Define subsystem information: shells

Choose `ssh.shells` in the `Configuration` box and click `Finish`.

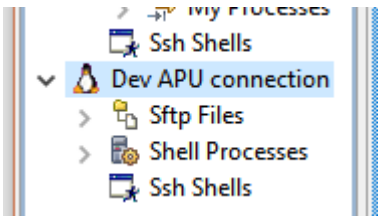

Figure 9: The new created connection

4.2 Connect to the newly created connection

Right click on the connection we created.

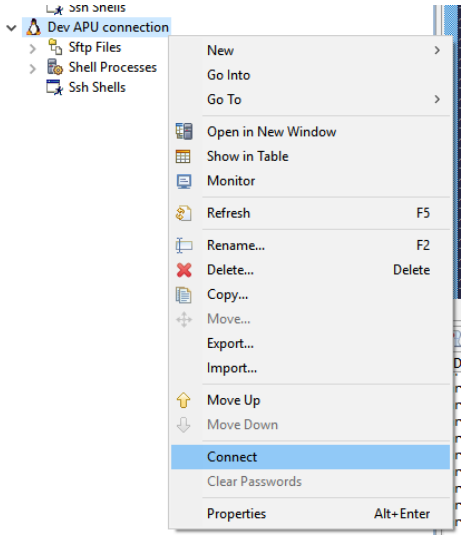

Figure 10: Remote connection contextual menu

Click on the `Connect` menu item.

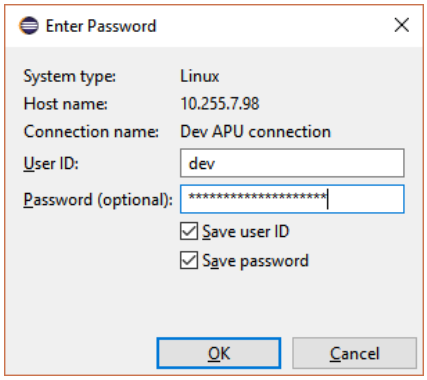

Figure 11: Remote connection credential dialog

Fill the `User ID` field with "**dev**" and the `Password` field with the one provided by Triple-IN, then click `OK` to establish the connection.
If no message is shown, the connection was successful. We can see if the connection is truly ok looking at the icons on the connection: they will now include a green arrow.

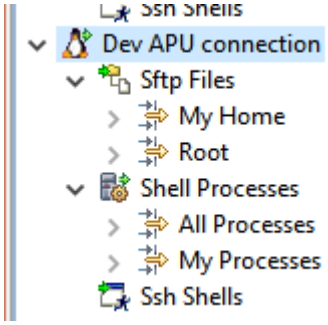

Figure 12: Green arrows show the connection is established

4.2.1 Host identification

If on the same computer we connected to a different Smart Sensor (or any device capable of SSH connections) which was configured with the same IP address, a warning will probably be shown. This warning is informing that the identification key of the device is changed. Normally we can press `Yes` without any worries, especially in a development environment.

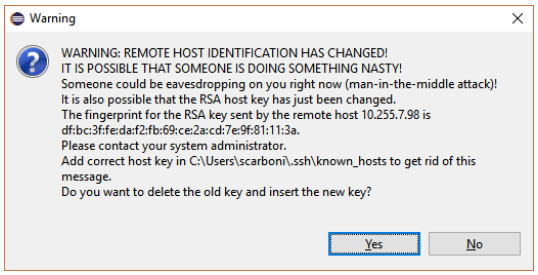

Figure 13: Device identification change warning

If for any reason we press the `No` button, the connection will not be established.

4.3 Browsing and managing files

Now that the connection has been established, we can access to the APU file system directly from Eclipse.
Expanding the `Sftp Files` item inside our connection we have access to the `dev` user Home directory and to the Linux root directory.

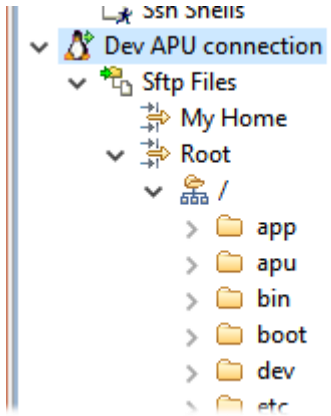

Figure 14: Access to APU file structure

Simple file operations are allowed here, including dragging files from Windows Resource Explorer. All these operations are possible only on files or folders on which the "**dev**" user has some authorization.

5 Setup an Eclipse project

To be able to compile a project to run on an APU, we have to setup a cross compiling project on Eclipse.

❖ Important

This chapter will assume a reader basic knowledge on Eclipse CDT.
For more information about Eclipse and the C/C++ developer toolset please refer to the Eclipse documentation.

5.1 Creating a new project

Start creating a new C++ Project using the `File->New->C++ Project…` menu command.

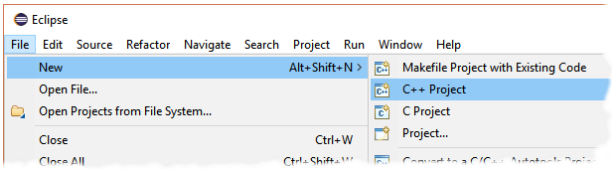

Figure 15: Create a new C++ project

In the C++ Project creation window, give your project a name and change the storing location if needed.
Take care of selecting the `Cross GCC` toolchain in the right box and then click `Next`.
You can choose the `Hello World C++ Project` project type if you want to have an initial source file. Normally an `Empty Project` is all you need.

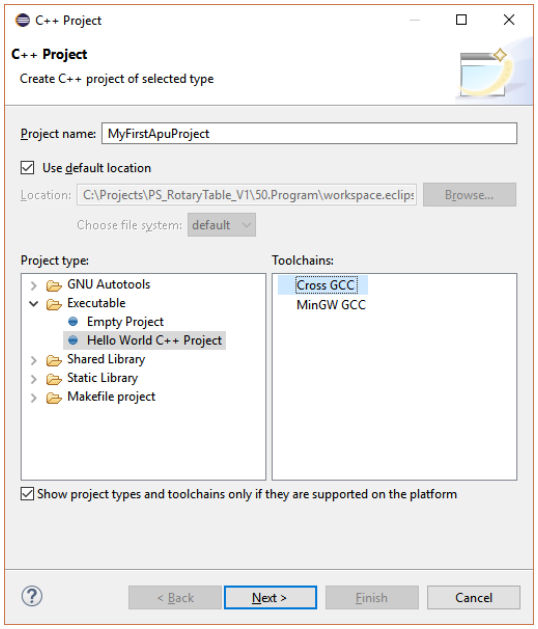

Figure 16: Create C++ project

If the `Hello World C++ Project` project type was chosen, the user has the possibility to set some properties related to the source file.

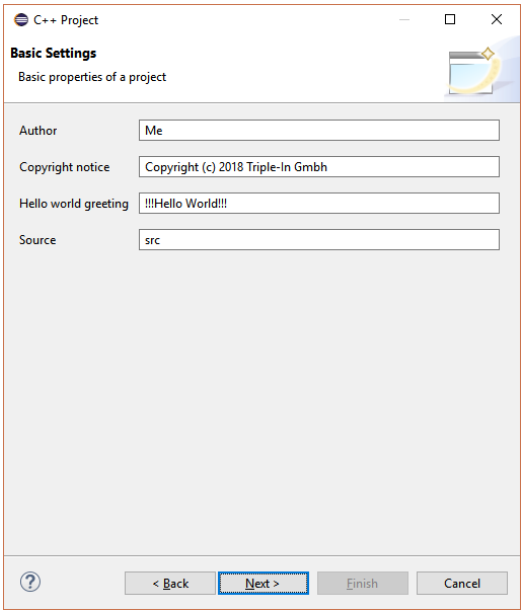

Figure 17: Hello World project properties

The next page will let us choose the configurations we would like to deploy on. Normally the default `Debug` and `Release` are enough.

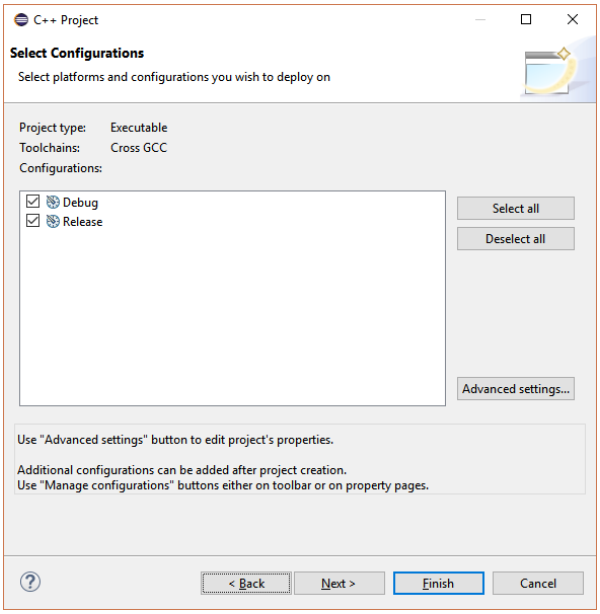

Figure 18: Deploy configurations

On the last page we need to setup the cross-compiling toolchain information.

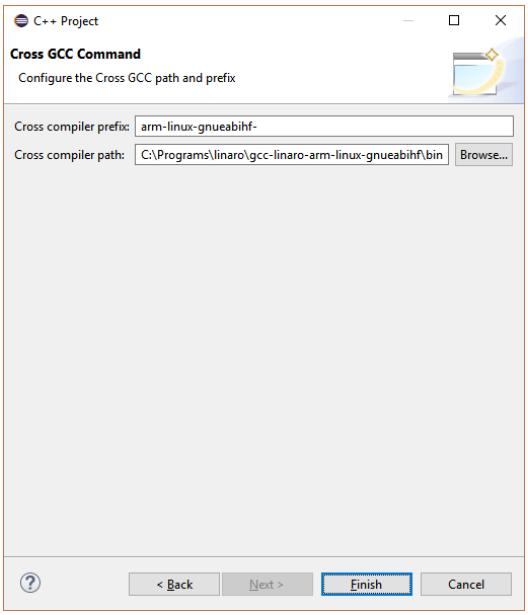

Figure 19: Project toolchain information

On the `Cross compiler prefix` we must write `arm-linux-gnueabihf-`. This is the prefix that every tool in the Linaro toolchain has.
The `Cross compiler path` is the one we defined in *Chapter 3* plus the `bin` subfolder in which every compiling tool is.
Click on the `Finish` button to confirm the creation of the new project.

5.2 Basic project configuration

When the project is created following the instructions described above, everything is ready for the first compilation.
If you'll try to compile the PSDemoProgram, the C and C++ dialect must be setup to the latest available one for the toolchain.
To do so, right click on the project in the `Project Explorer` and choose the last menu item, `Properties`.

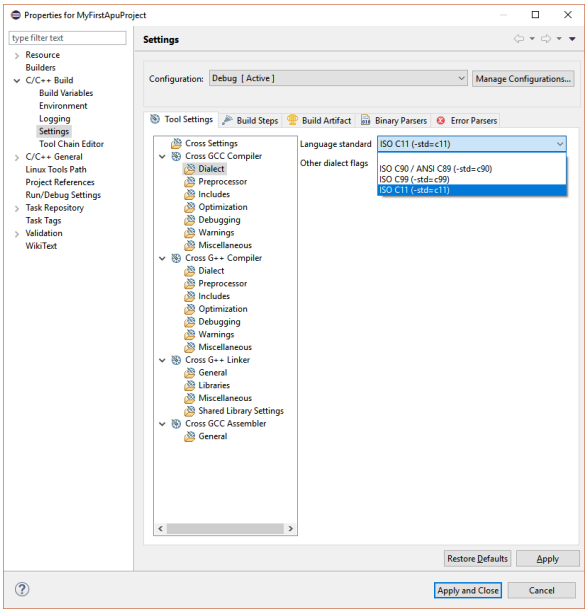

Figure 20: Project settings, changing C++ Dialect

Find and select `C/C++Build->Settings` on the left panel in the Properties dialog.
Under `Cross GCC Compiler` select the item `Dialect`. Select `ISO C11` from the list on the right.
Under `Cross G++ Compiler` select the item `Dialect`. Select `ISO C++1y` from the list on the right.

5.3 Preparing the project for debug

To debug an application, we need to create a Debug configuration for the program. Click on the `Run->Debug configurations…` menu.

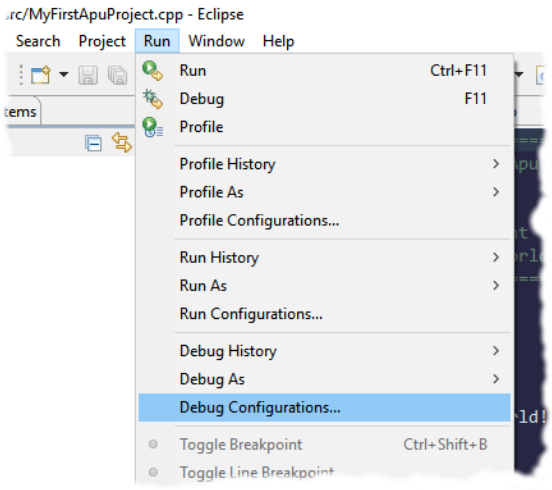

Figure 21: Debug configurations menu

In the `Debug Configurations` window that will open, select the `C/C++ Remote Application` in the left list and click on the `New launch configuration` button as indicated by the red arrow below.

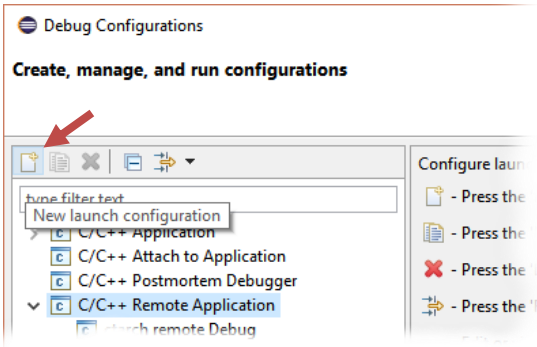

Figure 22: Create new Remote Application debug configuration

The new debug configuration will be created and selected for you.
On the right part of the window it is possible to setup all the parameters needed to enable the debug session.

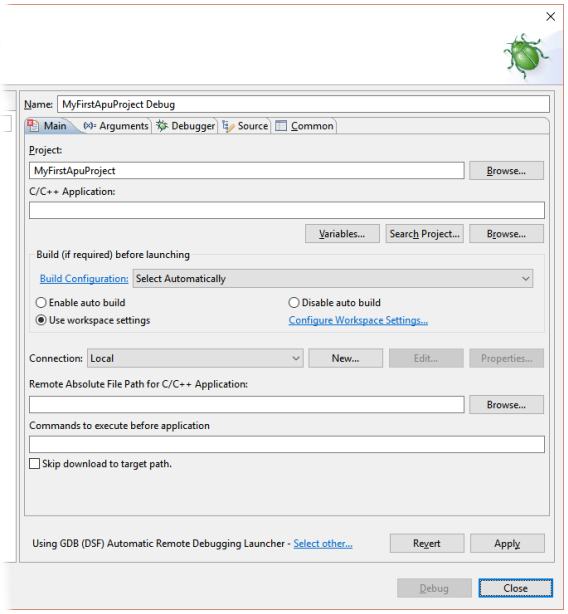

Figure 23: Debug configuration window

The default name of the configuration is set as `"App name" Debug`. The user can change it as needed.
To define the application to run we must set the `C/C++ Application`. Click on the `Search Project…` button and choose the Debug binary.

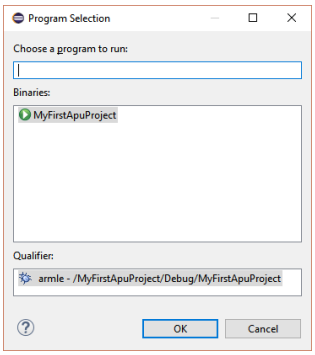

Figure 24: Search Project window

❖ Information

The Search project window will show a binary in the list only if the project was already compiled once!

Set the `Build Configuration` as `Debug`.
Now we need to setup a debug connection to work with. This operation is like the creation of a `Remote System` we described before, but it will be used only for debugging purposes.
Click on the `New…` button on the right of the `Connection` list.

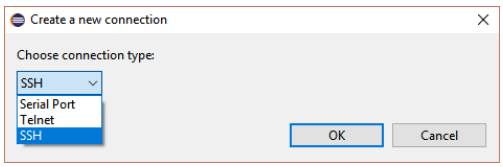

Figure 25: Debug connection type

Select `SSH` as the connection type and click `OK`.

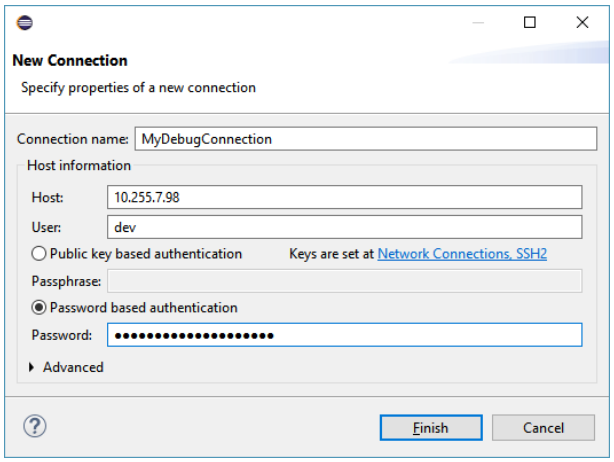

Figure 26: Debug connection information

Now set the `Connection name`, the sensor IP address in the `Host` field, `dev` in the `User` field and the provided password, selecting `Password based authentication`.
Clicking on `Finish` the debug connection will be created and selected in our Debug configuration.
Set the `Remote absolute file path for C++ Application`, pointing to the binary file. This could be `/home/dev/BINARY-NAME` or `/app/bin/BINARY-NAME`. The latter is preferred because it will be the only one used in production.
In the `Commands to execute before application` could be useful to insert this command:

```
chmod +x /home/user/BINARY-NAME
```

In this way the execute bit will be set for us.
Some other settings must be changed in the `Debugger` section.

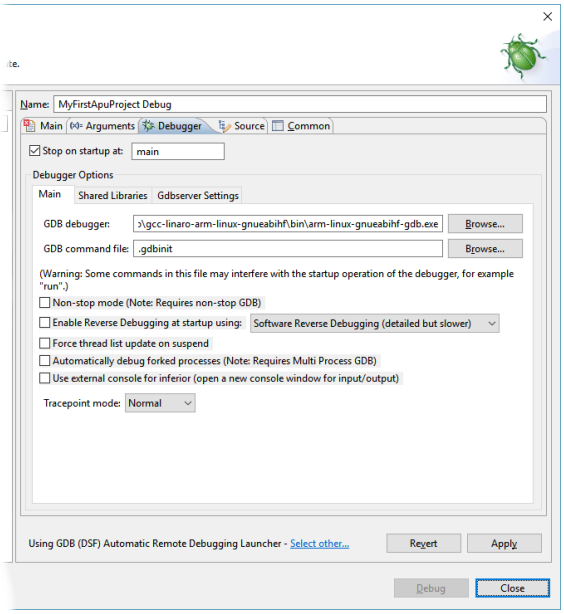

Figure 27: Debugger configuration section

Set the GDB debugger to `C:\Programs\linaro\gcc-linaro-arm-linux-gnueabihf\bin\arm-linux-gnueabihf-gdb.exe`, following what we decided in *Chapter 3*.
On the `GDB configuration file` field we must ensure to have a `.gdbinit` with at least the following contents:

```
set architecture arm
```

You can use the `Browse…` button on the right to search for the file if it's placed in a different folder.
Now we can click on `Apply` and then `Close` to confirm the configuration.

6 APU structure

6.1 Linux permissions

The developer has a special user to install and test his software.
The user credentials are:

- User: dev
- Password: *Provided by Triple-IN*

The console is available through the APU serial port or via SSH access.
During development, the custom application will probably be executed by the `dev` user. However, in a production environment the application will be run by the `app` user, which is a special user with no login access.

6.2 Directory structure

The directory structure accessible to the developer is under the `/app` folder.

```
/app
    /bin
    /data
    /etc
    /sbin
```

/app/bin

This folder is dedicated to the custom executables.

/app/data

This folder is reserved for storing data for the application.

/app/etc

This folder collects all the configuration files that can be customized.

/app/sbin

This folder contains some useful tools.

6.2.1 Permissions table

Folder	User	Permission
/app/bin	dev	rwx
	app	rx

Folder	User	Permission
/app/data	dev, app	rwx
/app/etc	dev	rwx
	app	rw
/app/sbin	dev	rwx
	app	rx

Figure 28: Permissions table

Important

Do not change the permissions for the /app directory and subdirectories. The permissions over the /app directory structure are restored on power on.

6.3 Custom application

The custom application is started automatically on power on, after all communication channels are up and running.
The custom application must be copied in `/app/bin` and the name must be `capp.sh` or `capp`. The file must have execution attribute enabled.
The start-up flow is the following:

1. The `/app/bin/capp.sh` is checked
2. If the file exists, it is executed. The result is discarded, so any possible error during the application execution is not reported
3. If the file do not exists, `/app/bin/capp` is checked
4. If the file exists, it is executed. The result is discarded, so any possible error during the application execution is not reported
5. If none of the files exist, no custom application is executed

The custom application is executed in the background, so the start-up procedure does not wait for it to succeed.
Despite that, it is strongly suggested to run the application as a daemon, forking from the main execution line.

7 Communication channels

7.1 Ethernet configuration

7.1.1 UDP/IP transport protocol

The UDP transport protocol can be used to send commands to the sensor, receive responses and receive the online scan stream.

7.1.2 TCP/IP transport protocol

The TCP transport protocol can be used to send commands to the sensor and receive responses. It is possible but not recommended to use TCP/IP to receive online scan stream.

7.1.3 Sensor IP address, Client IP address, Gateway IP and port

In the described communication pattern, the sensor provides some functionalities accessible through the Ethernet. Each one of these functionalities is identified as a "Service", and the control computer is the "Client".
The sensor has the possibility to start sending a scan data stream automatically on power on to a predefined Client, identified as the "Gateway" by a user defined combination of IP address and port. This specific function is called AutoStart (more information can be found in the PS Plus Programmers Manual).
The sensor socket addresses are a combination of an IP address and a port (which is mapped to the application program process). Every available combination identifies a Service.
The sensor has four different IP addresses configured and exposed to the outside:

- Predefined. It is automatically calculated from the serial number and cannot be changed nor disabled. The network mask is set to `255.255.0.0` and the address is calculated this way:

```
IP = 10.255.(serial / 100).(serial % 100)
```

- Custom. This address can be modified by the user through four sensor's User parameters. It is initially set with a default value calculated similarly to the Predefined:

```
IP = 10.0.(serial / 100).(serial % 100)
```

  For obvious reasons, this address cannot be set equal to the Predefined.

- Static or DHCP. This address can only be setup by a developer which has access to the APU "dev" user.
- Local. This is the localhost address (aka loopback), always present and normally used by the customer application.

7.2 Services

The following services are available by default on a sensor. As an example, we list the IP addresses for a sensor with the serial number as 1234:

	Service IP/Port	Protocol	Service	Description
1	10.255.12.34 6969	TCP/IP	Scan and commands	Command communication line for configuration and online data stream
		UDP/IP	Scan and commands	Command communication line for configuration and online data stream
2	10.0.12.34 1024	TCP/IP	Scan and commands	Command communication line for configuration and online data stream
		UDP/IP	Scan and commands	Command communication line for configuration and online data stream
3	Static or DHCP 6969,1024	TCP/IP	Scan and commands	Command communication line for configuration and online data stream
		UDP/IP	Scan and commands	Command communication line for configuration and online data stream
4	127.0.0.1 6968	TCP/IP	Scan and commands	Command communication line for configuration and online data stream
		UDP/IP	Scan and commands	Command communication line for configuration and online data stream
5	127.0.0.1 6967	TCP/IP	Scan and commands	This is normally used by the WebServer. Not available for other uses.
6	0.0.0.0 6996	UDP/IP	Announcement	A defined command sent to this channel will respond with the announcement message
7	0.0.0.0 3007	TCP/IP	Update	Used to send firmware updates to the sensor
8	0.0.0.0 22	TCP/IP	SSH	SSH access to the sensor
9	0.0.0.0 80	TCP/IP	Web interface	Access to the sensor web interface

Figure 29: Services table

The following table describes how a developer and a user may influence the services. The number in the first column refers to the number of the service in the previous table.

	Service IP/Port
1	Both developer and user cannot change this service in any way. This is always available
2	The user can change IP Address and port, like he can do in non-Plus sensors. The developer can disable this service completely, but the address will still be configured on the system. This behaviour allows the developer to use the IP/port defined by the user from the parameters to provide his own service
3	Normally disabled, this service can be enabled exclusively by the developer. The IP address will be chosen by the developer, but the ports are the predefined one (6969) and the one defined by the user through the parameters (default to 1024)
4	This is fixed and cannot be disabled or changed
5	This is fixed and cannot be disabled or changed. Normally is unavailable because already in use by the Webserver. Only when the Webserver is disabled (see point 9 on this list) this service is free to be used
6	This is fixed and cannot be disabled or changed
7	This is fixed and cannot be disabled or changed
8	This is fixed and cannot be disabled or changed
9	This is fixed and cannot be changed but the developer can disable it

7.3 Limitations

7.3.1 Concurrent connections

Update and Commands TCP/IP listening Services are limited to one connection at a time. This means that if a client is connected to TCP/IP Service, no other client can use that service.

7.3.2 Custom Service IP and Port

The IP address set by the user cannot be the same as the Predefined.
The Port set by the user cannot be one of: `22`, `80`, `3007`, `6969` and `6996`.

7.3.3 Suggestions

It is strongly recommended for the custom application to use one or both the local channels available as described in point 4 of the previous tables. This will keep the other channels available for communication with the outside.

It is strongly recommended to make use of the AutoScan feature (see PS Plus Programmers Manual) to reduce timing issues and reduce the workload for the application.

☞ Important

Please refer to the PS Plus Programmers Manual for more information about communicating with the sensor through one of its services.

8 Custom configuration

Some low-level settings can be changed by the developer to customize the behaviour and appearance of the system.
The `/app/etc/system.conf` special configuration file can be modified with the "**dev**" rights.

```
################################################################################
#
# system.conf
#
# Copyright 2017 Triple-In
#
# Version: 1
#
################################################################################
#
#                               WARNING!
#
# Modifying this file can make the sensor unusable from the network!
# Please be very careful when changing any of its parameter!
#
################################################################################

#
#  Custom network configuration
#

network.custom=no             # Enables the customization of the user network interface
network.addressmode=dhcp              # Type of the network configuration: dhcp or
static
network.address=192.168.42.42 # The IP address of the interface (used only in address
static mode)
network.netmask=255.255.255.0 # The network mask of the interface (used only in address
static mode)
network.network=192.168.42.0  # The network base address (used only in address static
mode)
network.broadcast=192.168.42.255      # The broadcast address inside the network (used
only in address static mode)
network.gateway=192.168.42.1  # The default gateway (used only in address static mode)
network.dnsmode=static                # Defines if the dns must be retrieved by dhcp or
set in address static mode
network.dns1=192.168.42.1             # The primary dns address (used only in dns static
mode)
network.dns2=192.168.42.254   # The secondary dns address (used only in dns static mode)

network.nouseraddr=no         # If set to yes, the address defined by the sensor
parameters is not setup
network.nouserservice=no              # If set to yes, the service on the address
defined by the sensor parameters is not started

#
#  Custom webpage configuration
#

webpage.enabled=yes   # If set to no, the http service will not be started at all
```

8.1 Network customization

To set up a customized network address, the `network.custom` parameter must be set to `yes`:

```
network.custom=yes
```

When the custom network is enabled, we can setup a network interface in different flavors.

8.1.1 DHCP address

To let the Ethernet interface get an address auto-assigned by a DHCP server:

```
network.addressmode=dhcp
```

When dhcp mode is set, the following parameters are not used:

```
network.address=
network.netmask=
network.network=
network.broadcast=
network.gateway=
```

8.1.2 Static address

To setup an interface with a static address, the `network.addressmode` parameter must be set accordingly:

```
network.addressmode=static
```

Then:

1. `network.address` must be set to the desired address
2. `network.netmask` must contain the network mask
3. `network.network` must contain the network base
4. `network.broadcast` must be set to the broadcast address of the network
5. `network.gateway` contains the primary gateway address

Here is an example:

```
network.address=192.168.42.42
network.netmask=255.255.255.0
network.network=192.168.42.0
network.broadcast=192.168.42.255
network.gateway=192.168.42.1
```

8.2 VPN client setup

It is possible for the developer to setup and enable a VPN connection where the APU act as a client.
To enable the VPN an OpenVPN configuration file named `openvpn.conf` or `config.ovpn` must be placed into the `/app/etc/` directory.
In case both files are present, `config.ovpn` is used.

> ❖ **Note**
>
> The content of the VPN configuration files must follow the rules of a standard OpenVPN configuration. Please check the OpenVPN documentation for more information.

8.3 DNS servers setup

DNS servers can be assigned either statically or dynamically.

8.3.1 DHCP assigned DNS

To let the DNS be configured by a DHCP server:

```
network.dnsmode=dhcp
```

8.3.2 Statically assigned DNS

The `network.dnsmode` parameter must be set as `static` and the `network.dns1` and `network.dns2` address parameters must be filled.

```
network.dnsmode=static
network.dns1=192.168.42.1
network.dns2=192.168.42.254
```

8.4 Web server setup

The APU provides a web page to access some information and configuration pages. Some behaviors of this web page can be modified.

8.4.1 Disabling the web server

The web server can be completely disabled (not starting at all the http service on the APU) setting to `no` the `webpage.enabled` parameter.

```
webpage.enabled=no
```

8.4.2 Customizing the web page information

If the web server is enabled, another possibility is given to the developer to customize the information shown on the web pages.
When the `brand.info` file is found in the `/app/etc` folder, the system will read it to configure some aspects of the webpage.
A file named `brand.info.EXAMPLE` is provided and can be used as a starting point.

```
brand-name=Triple-IN    # This is used for brand name in menu-bar
brand-link=http://www.triple-in.com

address-enabled=1
address-name=Triple-IN GmbH
address-street=Poppenbuetteler Bogen 64
address-city=D-22399 Hamburg
address-country=Germany

info-enabled=1
info-phone=0049 (0) 40 500 91998
info-fax=0049 (0) 40 527 34933
info-email=info@triple-in.de
info-website=http://www.triple-in.com

support-enabled=1
support-phone=0049 (0) 40 500 91998
support-email=support@triple-in.de
```

Branding

If the value of `brand-name` is left empty, no brand will be shown on the leftmost side of the menu bar.

```
brand-name=
```

If the brand name is defined, the `brand-link` field will be used to generate an `href` accordingly.

```
brand-name=Triple-IN
brand-link=http://www.triple-in.com
```

Figure 30: Brand name on the web page

All the following settings affect the contents of the `Contact` section in the web page.

```
address-enabled=1
info-enabled=1
support-enabled=1
```

When one of the `-enabled` fields is set to **0**, the section is removed from the Contact page.
If all the `-enabled` fields are set to **0**, the entire Contact section is removed from the web page.
If one section is enabled with `-enabled=1`, all the fields of the section will be shown and therefore must be defined.

A Table of figures