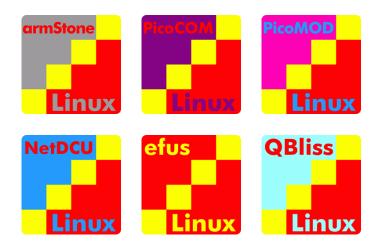
# **F&S Introduction to Eclipse**

## Debugging an Application

Version 1.3 (15.09.2022)





© F&S Elektronik Systeme GmbH Untere Waldplätze 23 D-70569 Stuttgart Germany

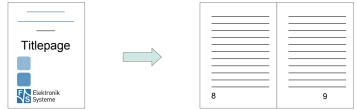
> Phone: +49(0)711-123722-0 Fax: +49(0)711-123722-99

## **About This Document**

This document describes how to debugging remote device using Eclipse CDT Automatic Remote Debugging Launcher under Linux.

#### Remark

The version number on the title page of this document is the version of the document. It is not related to the version number of any software release. The latest version of this



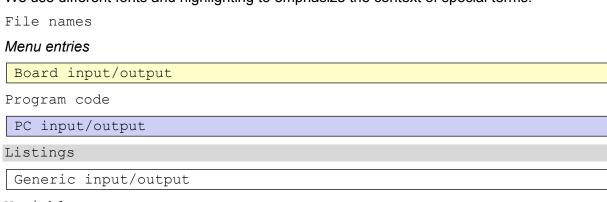
document can always be found at http://www.fs-net.de.

#### How To Print This Document

This document is designed to be printed double-sided (front and back) on A4 paper. If you want to read it with a PDF reader program, you should use a two-page layout where the title page is an extra single page. The settings are correct if the page numbers are at the outside of the pages, even pages on the left and odd pages on the right side. If it is reversed, then the title page is handled wrongly and is part of the first double-page instead of a single page.

#### **Typographical Conventions**

We use different fonts and highlighting to emphasize the context of special terms:



Variables

## History

Date	V	Platform	A,M,R	Chapter	Description	Au
2015-10-26	1.0	*	Μ	Layout	Converted to new F&S document layout, added and updated pictures	AD
2018-04-06	1.1		М	ALL	Modify the documentation, add more picture and explain more in details	PJ
2022-03-18	1.2		A,M	1.1	Modified section 1.1, add 1.2 for manual installation	AD
2022-08-19	1.3		М	4	Modified section 4, Yocto Debugger.	TG
V Version A.M.R Added, Modified, Removed						
Au Aut	,		leu			





## **Table of Contents**

Ab	out Tł	nis Document	iii	
His	story	V		
Tal	ole of	Contents	vii	
1	Intro	duction	1	
	1.1	Requirements	1	
	1.2	Yocto Tool Chain	1	
	1.3	Install Eclipse on Fedora	2	
	1.4	Install Eclipse from website	3	
2	Creat	e an Application	4	
	2.1	Create hello c file	12	
3	Remote Connection			
	3.1	Setup SSH connection on SBC/SOM	14	
	3.2	Setup SSH connection in Eclipse	15	
4	Setup	o Application settings	20	
5	Build	and Debug Application	25	
	5.1	Release build	25	
	5.2	Run Release build on device	25	
	5.3	Debug build	26	
	5.4	Debug build on device	26	
6	Appe	ndix	29	
	List of Figures			
	Important Notice			





## 1 Introduction

F&S offers a whole variety of Systems on Module (SOM) and Single Board Computers (SBC). There are different board families that are named NetDCU, PicoMOD, PicoCOM, armStone, QBliss, efus and PicoCore.

Linux is available for all of these platforms. Linux offers the integrated development environment (IDE) Eclipse for application development. Development environments include Eclipse CDT for C/C++.

This document describes how to debug an application on a remote device using the automatic remote debugging launcher plug-in.

### 1.1 Requirements

To develop an application following are necessary:

- Linux PC or VPC with fedora distribution
- Cross tool chain
- Buildroot or Yocto BSP

For successful quick start we offer listed software tools on our homepage in download area. For more information see documents "Quickstart with F&S Development Machine", "First Steps" and "Linux on F&S Boards"

#### Note:

On F&S development machine needed packages are preinstalled or can be installed easily. Cross tool chain and BSP are available too.

### 1.2 Yocto Tool Chain

The cross tool chain in case of yocto BSP is not a part of F&S development machine. But it can be created by building SDK. For this yocto images must be available. See section 10.2 of "Linux on F&S Board" for more details.

First add additional packages for debugging to your local configuration. Append to the configuration file conf/local.conf following lines:

EXTRA\_IMAGE\_FEATURES ?= "debug-tweaks tools-debug eclipse-debug ssh-server-openssh" CORE\_IMAGE\_EXTRA\_INSTALL += "openssh-sftp openssh-sftp-server"

Second build the SDK by command



bitbake -c populate\_sdk <image name>

After building cross tool chain can be found in "<buld-dir>/tmp/deploy/sdk" sub directory with name <Distro>-glibc-x86\_64-<image-name>-aarch64-<machine>-toolchain-5.4-zeus.sh

E.g, after building our standard images the tool chain is named "fus-imx-wayland-glibc-x86\_64-fus-image-std-aarch64-fsimx8mp-toolchain-5.4-zeus.sh"

To install the toolchain to directory of your choice execute the script.

### 1.3 Install Eclipse on Fedora

Install Eclipse on your development host with your Linux Distribution specific package manager system. On Fedora up to version 23:

sudo yum install eclipse-cdt

On new Fedora versions like 27 or 30 package manager dnf can be used to install new packages:

sudo dnf install eclipse-cdt

Additional following packages can be helpful for application development too:

- eclipse-rse Eclipse Remote System Explorer
- eclipse-linuxtools Linux specific Eclipse plugins

Then start Eclipse:

eclipse &

#### Note:

If Yocto toolchain is using eclipse must be started from the same terminal where SDK environment are set by command:

source <toolchain directory>/environment-setup-aarch64-poky-linux

Alternative create a shell script to start both command sequentially.

E.g.: eclipse-yocto.sh

```
#!/usr/bin/sh
source <path to installed yocto toolchain>/environment-setup-
aarch64-poky-linux
<path to eclipse>/eclipse
```



### 1.4 Install Eclipse from website

Download Eclipse IDE for C/C++ Developers package from website <u>https://www.eclipse.org/downloads/packages/</u> and install it in directory of your choice.

E.g., current version is IDE 2022-03 R and package name is eclipse-java-2022-03-R-linux-gtk-x86\_64.tar.gz.

For installation of additional packages "Install New Software..." dialog can be used. To start it go the menu entry "Help".



## 2 Create an Application

After all dependencies are installed, you can create a new project. From *File* select *New* and then C *Project*. Then click on *Next*.

New Project	$\sim$ ×
Select a wizard	-
Create a new C project	
Wizards:	
type filter text	
▶ 🧀 General	^
▼ 🧀 C/C++	
C Project	
C/C++ Project	
🖻 C++ Project	
🖻 Makefile Project with Existing Code	
▶ 🧀 RPM	
🔸 🗁 Shell Script	U
+ 🗁 Tracing	
	•
? < Back Next > Cancel Fi	nish

Figure 1: Create a new C project

On the next dialog fill in a project name and select *Empty Project* from *Executable*. Select *Cross GCC* as toolchain.

C Project							
C Project Create C project of selected type		Ď					
Project name: hello_world							
Use <u>d</u> efault location							
Location: /home/jakob/workspace/test/hello	_world	B <u>r</u> owse					
Choose file system: default 🔻							
Project type:	Toolchains:						
🕨 🧽 GNU Autotools	Cross GCC						
💌 🗁 Executable	Linux GCC						
<ul> <li>Empty Project</li> <li>Hello World ANSI C Project</li> <li>Shared Library</li> <li>Static Library</li> <li>Makefile project</li> </ul>							
Show project types and toolchains only if they are supported on the platform           ?         < Back							

Figure 2: Create empty C project

**Eclipse Application Development** 



C Project Select Configurations Select platforms and configurations you wish to deploy on Project type: Executable Toolchains: Cross GCC Configurations: 🔽 💸 Debug Select all 🔽 🛞 Release Deselect all Advanced settings... Use "Advanced settings" button to edit project's properties. Additional configurations can be added after project creation. Use "Manage configurations" buttons either on toolbar or on property pages. ? Cancel Finish < Back Next >

Then click on Next and select platforms and configurations you wish to deploy on.

Figure 3: Select configurations

Click on *Next* and setup the installed toolchain.

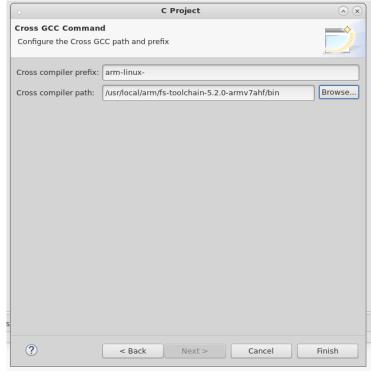


Figure 4: Configure Cross GCC



0	С	Project		$\sim$
Cross GCC Comman Configure the Cross G		x		Ď
Cross compiler prefix:				
Cross compiler path:				Browse
t				
?	< Back	Next >	Cancel	Finish

Figure 5: Configure Cross GCC - Yocto Toolchain

Click on *Finish* and you have successfully created your project.



For Yocto toolchain modify following setting too:

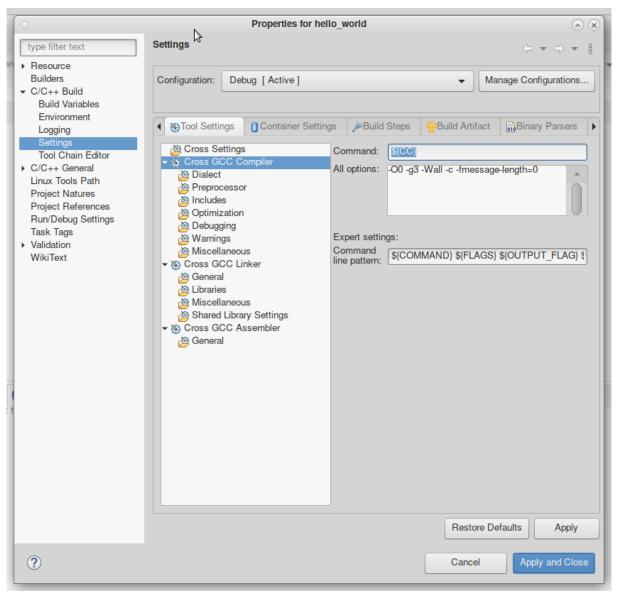


Figure 6: Cross GCC Compiler - Yocto Toolchain



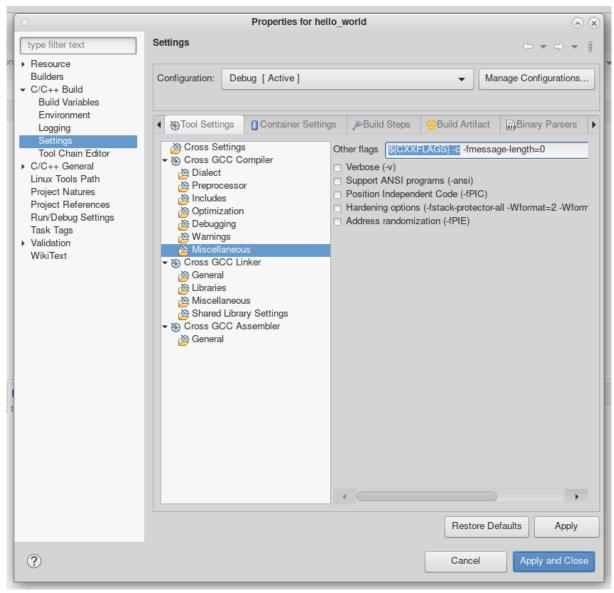


Figure 7: Cross GCC Compiler Flags - Yocto Toolchain



0		Properties for hel	lo_world			
type filter text	Settings	ß				( 8
<ul> <li>Resource Builders</li> <li>C/C++ Build Build Variables</li> </ul>	Configuration:	Debug [Active]			✓ Ma	anage Configurations
Environment Logging Settings Tool Chain Editor • C/C++ General Linux Tools Path Project Natures Project References Run/Debug Settings Task Tags • Validation WikiText	<ul> <li>Tool Settin</li> <li>Tool Settin</li> <li>Cross Set</li> <li>Cross GC</li> <li>Preproc</li> <li>Includes</li> <li>Optimiz</li> <li>Debugg</li> <li>Warning</li> <li>Miscella</li> <li>Shared</li> <li>Cross GC</li> <li>General</li> <li>Cross GC</li> <li>General</li> <li>Eibraries</li> <li>Miscella</li> <li>Shared</li> <li>Scross GC</li> <li>General</li> </ul>	ttings CC Compiler eessor s ation ing gs aneous CC Linker I s aneous Library Settings CC Assembler	gs Muild Command: All options: Expert settin Command line pattern:	\${CC}	PBuild Artifact	Binary Parsers         Binary Parsers         S(OUTPUT_FLAG) \$         S(OUTPUT_FLAG) \$         efaults
?					Cancel	Apply and Close

Figure 8: Cross GCC Linker - Yocto Toolchain



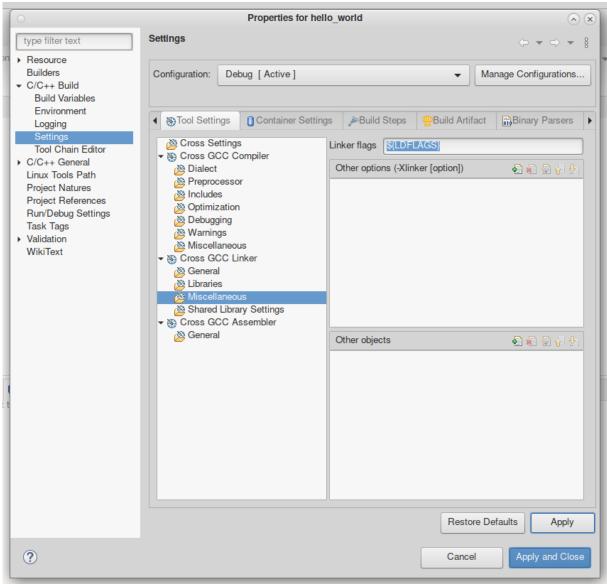


Figure 9: Cross GCC Linker Flags - Yocto Toolchain



0	Properties for hel	lo world		
type filter text	Settings	_		⊕ <b>-</b> ⇒ <b>-</b> ;
<ul> <li>Resource Builders</li> <li>C/C++ Build Build Variables</li> </ul>	Configuration: Debug [ Active ]		• Ma	anage Configurations
Build Vanables Environment Logging Settings Tool Chain Editor • C/C++ General Linux Tools Path Project Natures Project References Run/Debug Settings Task Tags • Validation WikiText	<ul> <li>Tool Settings</li> <li>Cross Settings</li> <li>Cross GCC Compiler</li> <li>Dialect</li> <li>Preprocessor</li> <li>Includes</li> <li>Optimization</li> <li>Debugging</li> <li>Warnings</li> <li>Wiscellaneous</li> <li>Cross GCC Linker</li> <li>General</li> <li>Libraries</li> <li>Miscellaneous</li> <li>Scoss GCC Assembler</li> <li>General</li> </ul>	Command: \${AS} All options:		Binary Parsers
			Restore D	efaults Apply
?			Cancel	Apply and Close

Figure 10: Cross GCC Assembler - Yocto Toolchain



## 2.1 Create hello c file

After you have successfully created your project right-click your project folder, choose *new* and select *source file*.

0	~	_		tort
File Edit Source F	New	>	Project	
📑 🕶 🔚 🐚 🤌	Go Into		° File	
ο Project Explorer δ	Open in New Window		🕒 File from Template	
• 😂 hello_world	Show in Local Terminal	>	🗳 Folder	
) 🔊 Includes	Copy Strg	+C		
	n Paste Strg	+V	▶ Header File	
	💢 Delete		😋 Source File	
	. Remove from Context		😂 Source Folder	
	Source Move Rename	<b>&gt;</b> F2	ි C/C++ Project ම Convert to a C/C++ Project (Adds C/C++ Nature)	
	🚵 Import		📑 Example	
	🗠 Export		🔁 Other	Strg+N

Figure 11: Create Source File

Select a name for your c file and select *finish*.

0	New Source File	$\odot$
Source File		=
Create a new s	source file.	C
Source folder:	hello_world	Browse
Source file:	hello_world.c	]
Template:	Default C source template	Configure
?	Cancel	Finish

Figure 12: New Source File

#### Write down your example code.

#include <stdio.h>

int main(void)



```
{
    int i, val = 0;
    printf("Hello World!\n");
    for(i = 0; i < 4; i++)
    {
        val = 2*i;
        printf("value: %d\n", val);
    }
    return 0;
}</pre>
```



Remote Connection

## 3 Remote Connection

### 3.1 Setup SSH connection on SBC/SOM

First of all you have to setup your F&S Board. Therefore you can have a look into F&S *FSiMX8M(M,N,P)/FSiMX6/SX/UL\_FirstSteps\_eng.pdf*. After that boot your F&S Board.

To work with SSH the board should have a valid date. This is necessary to create certificates for SSH. To setup a date you can use the following command:

date "2018-04-09 08:20"

Afterwards we have to enable the network interface. You can also set the network on command in UBoot to enable network interface at each boot. For further information please take a look in *FSiMX6/FSiMX8M(M,N,P)\_FirstSteps\_eng.pdf*.

Dynamically:

udhcpc

Static:

ifconfig eth0 10.0.0.84 up

The Root-Filesystem is read-only mounted but we have to modify something in the filesystem so we need it read-writeable.

mount -o remount,rw /

Open *sshd* config file

vi /etc/ssh/sshd\_config

and edit the following lines:

```
...
Port 22
AdressFamily any
ListenAddress 0.0.0.0
PermitRootLogin yes
```

Optional you can also allow to login without password, but we recommend you to not do this because it's a security risk. If you want to do it anyway add the following line to *sshd\_config* file.

```
PermitEmptyPasswords yes
```



...

After that you have to start the SSH server.

For Buildroot:

/etc/init.d/S50sshd start

For Yocto:

systemctl restart sshd.socket

Now the SSH server is running on our SBC/SOM. Now we can test it, therefore we are going back to our VM. To connect via ssh we open a Terminal, after that we are sending the following command:

ssh root@10.0.0.84

[jakob@localhost ~]\$ ssh root@10.0.0.84 The authenticity of host '10.0.0.84 (10.0.0.84)' can't be established. ECDSA key fingerprint is b1:b1:aa:83:12:d1:f1:21:7b:e3:6a:61:89:6e:31:ea. Are you sure you want to continue connecting (yes/no)? yes Warning: Permanently added '10.0.0.84' (ECDSA) to the list of known hosts.

Figure 13: SSH connection terminal

Now the SSH connection is successfully established and tested.

### 3.2 Setup SSH connection in Eclipse

Open Remote System Explorer. In the top right corner click *open Perspective* and select *Remote System Explorer*.

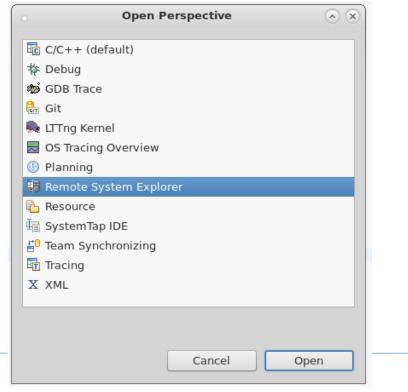




Figure 14: Open Perspective

Now we define a remote connection. Click in the left area, on the toolbar *Define a connection* to remote system. Select SSH Only and click Next >.

Í	New Connection         Image: Connection
1	Select Remote System Type
-	Connection for SSH access to remote systems
	System type:
	type filter text
·	* 🧽 General
	🔁 FTP Only
-	∆ Linux
-	E Local
i	SSH Only
k	📮 Telnet Only (Experimental)
1	Windows
ł	
1	
	(?) < Back Next > Cancel Finish
1	

Figure 15: New Connection – Remote Type

Enter the connection configuration of your device and select *Finish*.

	New Connection	$\sim$ $\times$
emote SSH Only Sy	stem Connection	
Define connection info	rmation	
Parent profile:	localhost	
Host name:	10.0.89	<b></b>
Connection name:	F&S Device	
Description:	Connection to F&S Board	
🗹 Verify host name		
Configure proxy settin	<u>gs</u>	
2	< Back Next > Cancel	Finish



It's better to setup a static IP address for the SBC/SOM so you don't have to change the ssh connection settings after every dhcp.

Test ssh connection, right-click your created remote connection and select Connect.

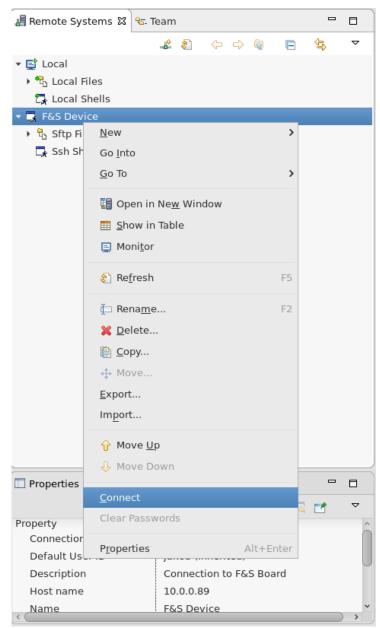


Figure 17: Connect to device



#### Remote Connection

Type in your login details and select OK.

o Enter	Password ×
System type: Host name: Connection name:	SSH Only 10.0.0.89 FS Device
User ID:	root
Password (optional):	****  Save user ID
	🗹 Save password
Ca	ancel OK

Figure 18: Login details





Use the Remote Systems tab to navigate through the target filesystem:

Figure 19: Target filesystem



## 4 Setup Application settings

Before you can setup the application settings be sure that you have built your Root-Filesystem (buildroot) or your meta-toolchain (Yocto) once. This is necessary because we need the compiled GDB debugger from this directory.

After the remote connection have been setup, switch back to the C/C++ perspective 🔤.

```
If you can't switch back to the C/C++ perspective do the following:
In the Menu-Bar select Window -> Perspective -> Open Perspecitve -> Other... ->C/C++
```

Select on the menubar *Run* and choose *Debug Configurations*.... After that select *C/C++ Remote Application* and click *New launch configuration*.

Debug Configurations				
Create, manage, and run configurat	ions	Ť.		
K      K	Configure launch settings from this dialog:			
Filter matched 6 of 6 items				
?		Close Debug		

Figure 20: New Debug Configuration

Setup debug configuration and apply it.

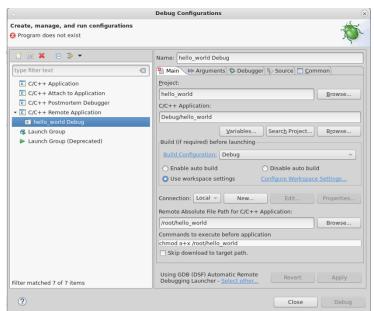
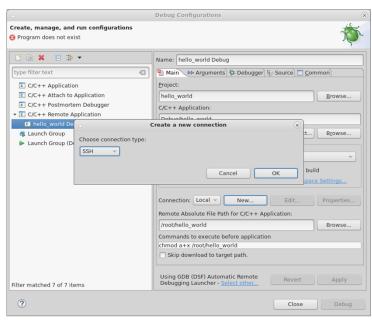


Figure 21: Setup Debug Configuration

**Eclipse Application Development** 





Select at Connection New... select connection type SSH and click OK.

Figure 22: Create remote connection

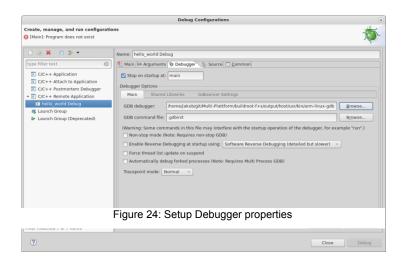
Setup the new connection and click Finish.

			Debug Configurations	i i		
create, manag	e. and run con	figurations				- <i>*</i>
Program does	0				( ) ( ) ( )	2
	New Connect					-
📑 📔 🗶 🛛	Specify prope	rties of a new conr	nection			
type filter text	Connection no	me: Remote devi	ico			non
C/C++ App			ice			
C/C++ Att						Browse
C/C++ Pos	Host:	10.0.0.89				
C/C++ Rer	User.	root				
c hello_wo	Public ke	v based authentica	ation Keys are set at	Network Connect	ions, SSH2	
Launch Gro Launch Gro			,			B <u>r</u> owse
Launch Gro	1	based authentica	tion			
			luon			· · ·
	Password:	••••				
	Advanced					Settings
	?			Cancel	Finish	Properties
			Remote Absolute Hie P	антогс/с++ ар	prication:	
			/root/hello_world			Browse
			Commands to execute	before applicatio	n	
			chmod a+x /root/hello			
			Skip download to ta	irget path.		
ilter matched 7	of 7 items		Using GDB (DSF) Auto Debugging Launcher -		Revert	Apply
?					Close	Debug
$\odot$						Debug

Figure 23: Setup remote connection



Switch to tab *Debugger* and select the GDB debugger path. The GDB debugger is located in the buildroot directory *.../output/host/usr/bin/arm-linux-gdb* or the meta-toolchain directory *.../5.4-zeus/sysroots/x86\_64-pokysdk-linux/usr/bin/aarch64-poky-linux/aarch64-poky-linux-gdb*. After that *Apply* the settings and *Close* it.



Select tab *Run* and choose *Run Configurations....* After that select *C/C++ Remote Application* and click *New launch configuration*.

reate, manage, and run configuratio		
	ns	
K      K	Configure launch settings from this dialog:	

Figure 25: New Run Configuration

Setup run configuration, Apply it and Close it.



	Run Configurations	
Create, manage, and run configurati Program does not exist	tions	
Image: Section 2016         Image: Section 20	Project: India_world CC++ Application: Release/helia_world Build Configuration: Release Build Configuration: Release © Enable auto build © Disable auto build © Use workspace settings.	Browse tt Browse
	Connection: Remote device	Properties
	/root/hello_world	Browse
	Commands to execute before application chmod a+ x /roothello_world Skip download to target path.	
Filter matched 8 of 8 items	Revert	Apply
?	Close	Run

Figure 26: Setup Run Configuration

ight-Click your project and select *Properties*. Choose *C/C++ Build* and select *Settings*. *Choose Configuration: Debug* and Add an include path. Include path is also from your buildroot directory .../output/host/usr/arm-buildroot-linux-gnueabihf/sysroot/usr/include or your meta-toolchain directory .../5.4-zeus/sysroots/aarch64-poky-linux/usr/include.

•		Properties for hello_world	$\overline{\mathbf{x}}$
type filter text 🛛 🛛	Settings		
<ul> <li>type filter text</li> <li>Resource Builders</li> <li>C/C++ Build Build Variables Environment Logging</li> <li>Settings</li> <li>Tool Chain Editor</li> <li>C/C++ General Linux Tools Path Project References Run/Debug Settings</li> <li>Task Repository Task Tags</li> <li>Validation WikiText</li> </ul>	Settings       Configuration:     Debug       Tool Settings     >>Build Step       Coross Settings     >> Social Settings       >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	s Prior Parsers Fror Parsers Include paths (-1) /home/jakob/git/Multi-Plattform/buildroot-f+s/output/host/usr/arm-buildroot-linux-gnue	Manage Configurations) 한 값 할 중 못
?		Restor	re <u>D</u> efaults <u>Apply</u>
0			

Figure 27: Add include path to Debug configuration



Switch configuration to Release and add the same path. Apply and Close the settings.

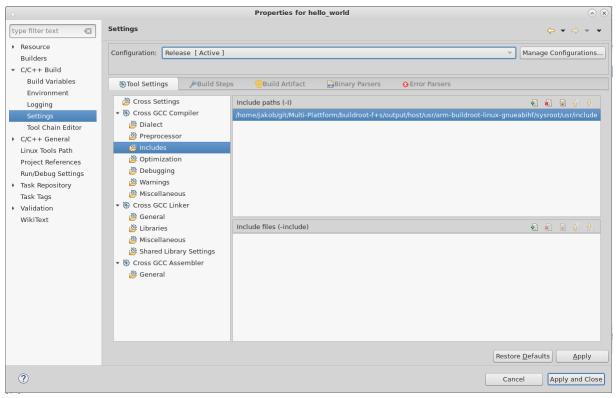


Figure 28: Add include path to Release configuration



## 5 Build and Debug Application

### 5.1 Release build

Select C/C++ perspective. Go to the downside arrow beside the hammer icon on the toolbar (right side) and choose Release. After that press the hammer icon.

Project Explorer 23 2 Palanza	🖓 🔻 🗂 🖪 hello_world.c 🛙	- 1
Q Tankara Q Tankara g Trolask g Tro	2 Comparison of the second	ب ن

Figure 29: Build Release configuration

### 5.2 Run Release build on device

To run the compiled example select the downside arrow beside the run icon (right side) and click *Run Configuration*.... Select your release configuration and click *Run*.

	Run Configurations		×
Create, manage, and run configurat	ions		
Image: Second	Name:       hello_world Release         Main:	Variables Starch Project Brown	
Filter matched 8 of 8 items		Revert App	ly .
?		Close	un

Figure 30: Run Release configuration



Build and Debug Application

If the remote connection is correctly configured, hello world will appear in the console windows in the bottom margin.

### 5.3 Debug build

Select C/C++ perspective. Go to the arrow beside the hammer icon (right side) and choose Debug. It should automatically build the example if not, press the hammer icon.

😁 • 🔝 🕼 👋 • 🚳 • 🖾 - 🖂 •	 \$ • 0 • 9 • 9 • 9 • 9 • 9 • 9 • 1 1 1 1 0 0 × 0 • 0 • 0 • 0 • 0 • 0 • 0 • 0 • 0	
Project Explorer 11 2 Rolcaso	 R hello world.c 31	
<ul> <li>Vario worki</li> <li>Surviva</li> <li>Surviva</li></ul>	<pre>intermediate and fail</pre>	a b v

Figure 31: Build Debug configuration

### 5.4 Debug build on device

To debug the built example select the downside arrow beside the debug icon (right side) and click *Run Configuration*.... Select your debug configuration and click *Run*.

	Debug Configurations	
Create, manage, and run configurat	ions	1
(i)      (ii)      (iii)      (iii)      (iiii)      (iiii)      (iiii)      (iiii)      (iiii)      (iiii)      (iiii)      (iiiii)      (iiiii)      (iiiii)      (iiiiii)      (iiiiiiii)      (iiiiiiiiii	Name: [helio_world Debug Main: (e) Arguments: (\$ Debugger ) Source Common Project: helio_world C(C++ Application: Debughelo_world Build (if required) before launching Build Configuration: Debug Genable auto build O lisable auto build O lisable auto build Connection: Remote Absolute File Path for C(C++ Application: /roothelio_world Connection: Remote Absolute File Path for C(C++ Application: /roothelio_world Connection Structure File Path for C(C++ Application: /roothelio_world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo world Commarks to execute before application choma d + x prochyleilo wo	Browse Properties Browse
ilter matched 8 of 8 items	Skip download to target path. Using GDB (DSF) Automatic Remote Debugging Launcher - <u>Select other</u> . Revert	Apply
?	Close	Debug

Figure 32: Debug configuration

**Eclipse Application Development** 



Confirm the Perspective Switch.

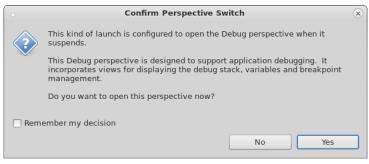


Figure 33: Confirm Perspective Switch

Now you're in Debug Perspective and you can debug your built example.

test - hello_world(hello_world.c - Eclipse							• • *		
H = 0 0 0 + 0 = 0 = 0 = 0 + 0 0 + 0 = 0 + 0 = 0 + 0 = 0 =		9 • 8 • <b>0</b> • 0 •					ck Access	er 16	4 87
🐐 Debug 🛙 🦌 🍺 🔻 🖻		🕬 Variables 😫 💁 Breakpoints 🔠 Registers 🍕 I	Expressions	🐑 Inte	ractive Console 🛋 Modu	les 🐁 🗖		~	
	-	Name Television P	int int			Value 0 0			ő
B hello_world c B 15 / 7 3 * hello_world.c					BE Outline 12		<u>≷</u> √° ● #	~	
<pre>4 * Created on: 00.04.2018 5 * Author: jakob 7 * 6 * 10 * 10 * 11 * 11 * 11 * 11 * 11 * 11</pre>									
13     val = 241; arristf*value: M(n*, val);       □ Console II ⊗ Tasks [] Problems O Executables @ Debugger Console [] Memory hollo world Debug (CC++ Remote Application) Remote Shell				~	• × % B	a 🖗 🗗 L		<b>*</b>	• •
helo, wend Désug ICK-+ Hennota Application) Hennota Skell funda ax /root/hell_world.gobserver : 3245 /root/hell_world.exit # chead ax /root/hello world.gobserver : 3245 /root/hello_world.exit Process /root/hello world created pide 426 Process /root/hello world created pide 426 Remote debugging from host 10.0.0.117									Ĵ

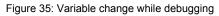
Figure 34: Debug Perspective



#### Build and Debug Application

To setup debug points right-click the line in your program where you want to set a debug point. Choose *Add Breakpoint...* After that setup Breakpoint properties and *Apply and Close* it. You also can setup a Breakpoint by left-click the corresponding line. With the Step options you can see variables are changing.

test - hello_world/hell	lo_world.c - Eclipse					• • ×
<u>File Edit Source Refactor Navigate Search Project Bun Window Help</u>						
🗂 🕶 🖬 🕼 👘 🕶 O 🕶 💁 🖉 🕶 💹 🖗 🖉 🐨 💭 🖗 🐨 📾 🖬 📾 🗰 🖓 🖘 🛄	1   9 • 8 • 9	⇔ + ⇔ +		Quick Acces	s 🕴 🖻	B B 🖗
🛊 Debug 🛱 🦌 👻 🖻 🗖	(*)= Variables 😫 💁 B	reakpoints 🛲 Registers 🍕 Expressions 👻	Interactive Console 🞽	Modules 🐁 📑 😑	11 II I	~ - 0
E hello_world Debug [C/C++ Remote Application]	Name	Type		Value		
* 😹 hello_world [426] [cores: 0]	00+ i	int		2		
👻 🧬 Thread #1 426 [core: 0] (Suspended : Step)	🕪 val	int		4		
main() at hello_world.c:19 0x10458						
📲 Remote Shell						
// home/jakob/git/Multi-Plattform/buildroot-f+s/output/host/usr/bin/arm-linux-gdb (7.10.1)						
						â
						U
	< C					)))
A hello_world.c 12			🗆 🔡 Outline 🛙	8 E 🕂 🛚 🛠 🗙	• ¥ *	~
5 */			n 🖬 stdio.h			
7 8			main(voi	d) : int		
9 #include <stdio.h> 10</stdio.h>						
110 int main(void)						
12 { 13 int i, val = 0;						
14 printf("Hello World!\n");						
$ \begin{array}{c} 15 \\ 16 \\ for(i = 0; i < 4; i++) \end{array} $						
17 { 18 val = 2*i;						
printf("value: %d\n", val);						
20 } 21 return 1;			-			
22 )						
			÷			
					-	
🖳 Console 🕴 🕢 Tasks 🔝 Problems 🕡 Executables 🙀 Debugger Console 🌔 Memory			<b>=</b> 74 92	🔒 📓 🐸 🚝 🛃 📑		• • •
hello_world Debug [C/C++ Remote Application] Remote Shell						~
<pre># chmod a+x /root/hello_world;gdbserver :2345 /root/hello_world;exit Process /root/hello world created; pid = 426</pre>						0
Listening on port 2345						
Remote debugging from host 10.0.0.117 Hello World!						
value: 0 value: 2						
Varue: 2						~





## 6 Appendix

## List of Figures

Figure 1: Create a new C project	4
Figure 2: Create empty C project	4
Figure 3: Select configurations	5
Figure 4: Configure Cross GCC	5
Figure 5: Configure Cross GCC - Yocto Toolchain	6
Figure 6: Cross GCC Compiler - Yocto Toolchain	7
Figure 7: Cross GCC Compiler Flags - Yocto Toolchain	8
Figure 8: Cross GCC Linker - Yocto Toolchain	9
Figure 9: Cross GCC Linker Flags - Yocto Toolchain	10
Figure 10: Cross GCC Assembler - Yocto Toolchain	11
Figure 11: Create Source File	12
Figure 12: New Source File	12
Figure 13: SSH connection terminal	15
Figure 14: Open Perspective	15
Figure 15: New Connection – Remote Type	16
Figure 16: SSH Connection settings	16
Figure 17: Connect to device	17
Figure 18: Login details	18
Figure 19: Target filesystem	19
Figure 20: New Debug Configuration	20
Figure 21: Setup Debug Configuration	20
Figure 22: Create remote connection	21
Figure 23: Setup remote connection	21
Figure 24: Setup Debugger properties	22
Figure 25: New Run Configuration	22
Figure 26: Setup Run Configuration	23
Figure 27: Add include path to Debug configuration	23
Figure 28: Add include path to Release configuration	24
Figure 29: Build Release configuration	25
Figure 30: Run Release configuration	25
Figure 31: Build Debug configuration	26
Figure 32: Debug configuration	26



#### Appendix

Figure 33: Confirm Perspective Switch	27
Figure 34: Debug Perspective	27
Figure 35: Variable change while debugging	28

### **Important Notice**

The information in this publication has been carefully checked and is believed to be entirely accurate at the time of publication. F&S Elektronik Systeme assumes no responsibility, however, for possible errors or omissions, or for any consequences resulting from the use of the information contained in this documentation.

F&S Elektronik Systeme reserves the right to make changes in its products or product specifications or product documentation with the intent to improve function or design at any time and without notice and is not required to update this documentation to reflect such changes.

F&S Elektronik Systeme makes no warranty or guarantee regarding the suitability of its products for any particular purpose, nor does F&S Elektronik Systeme assume any liability arising out of the documentation or use of any product and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

Products are not designed, intended, or authorised for use as components in systems intended for applications intended to support or sustain life, or for any other application in which the failure of the product from F&S Elektronik Systeme could create a situation where personal injury or death may occur. Should the Buyer purchase or use a F&S Elektronik Systeme product for any such unintended or unauthorised application, the Buyer shall indemnify and hold F&S Elektronik Systeme and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorised use, even if such claim alleges that F&S Elektronik Systeme was negligent regarding the design or manufacture of said product.

